POLITECNICO DI MILANO

School of Industrial and Information Engineering - Department of Electronics, Information and Bioengineering

"COUNT ON ME": AUTOMATED NEURAL NETWORK-BASED SYSTEMS FOR THE APOPTOTIC NUCLEI DETECTION

Supervisor:

Prof. Pietro Cerveri

Co-Supervisor:

Dr. Eleonora Dondossola

Co-Supervisor:

Dr. Stefano Casarin

Thesis by:

Federica Bussa, 897775

Academic Year: 2019-2020

Acknowledgements

I thank Professor Pietro Cerveri for providing me with the knowledge and for giving me the opportunity to work on such an interesting topic.

I thank Eleonora and Stefano for all their support and guidance throughout this project, and also for making so enjoyable the months spent in Houston.

Last, I thank my family and my friends for all the unconditional support they always gave to me despite my absence, especially in these last months.

Abstract

Introduction

Apoptosis is the programmed cell death, a highly regulated process responsible for cellular turnover in living tissues. Its alterations and associated events are involved in various form of disease, including cancer. Identifying apoptotic cells in pathological tissues provides valuable insights into the disease progression and helps evaluating the efficacy of putative therapies.

Various apoptosis detection modalities are current available, and they largely rely on empirical assessments from biologists. The process is laborious, time consuming and subjected to human error. To overcome this gap, we propose two automatic computerized systems to detect and quantify apoptotic cells in microscopic images to optimize time of the analysis, improve accuracy and abate the incidence of human errors.

Materials and Methods

The two proposed methods are based on deep learning algorithms [1] [2] and applied to confocal microscopic images of human metastatic melanoma cells [3]. Both of them are inserted in a work pipeline involving minimal data pre-processing and post-processing to extract parameters of interest.

The first method, based on object detection, consists in a VGG16-like convolutional neural network (CNN) trained as a binary classifier to identify apoptotic nuclei in small windows. Apoptotic and intact nuclei were manually cropped from the original images. The class imbalance issue was addressed by the construction of a balanced dataset by means of a dynamic under-sampling along the learning process (i.e. the network is trained with data that change over training iterations for the majority class) and a customized loss function in order to minimize false negative predictions.

Once the classifier was trained, it was systematically applied over test images in a sliding window fashion: the input image images were scanned with specific pixels resolution and for each position the CNN was applied. If the model identified an apoptotic event, the window was marked with a bounding box. In case of overlapping boxes, the nucleus was identified with their centroid.

The second method performs semantic segmentation over the entire image, distinguishing between intact nuclei, apoptotic fragments, and the remaining background. For this case, UNet architecture was chosen and adapted for the specific purpose, together with a customized loss function to target the class imbalance issue.

Results and Discussion

To evaluate the results of the first proposed detection framework, various standard performance parameters were used, resulting in satisfactory values for accuracy, precision and recall in validation samples, all above 0.98. Two input's sizes were investigated. In both cases, stride was set to half-size of the window or to a quarter window's size. A proper fit of these two parameters resulted essential for achieving satisfying results. The input size was proved to affect localization performances and a proper discrimination of events appearing really close each other as separated entities. The stride value instead was fundamental for a complete and accurate detection of all the apoptotic fragments.

The main disadvantage of the sliding window detection is the computational cost. Reducing the stride allows for a more accurate localization, but at the same time brings the disadvantage of increasing the computational time by a factor of 4. Choosing instead a larger stride, the number of windows to be predicted decreases, but the coarser granularity impacts both on detection and localization performance. Thus, a proper trade-off should be set between these two factors.

It resulted not possible to compare obtained results with a ground-truth, since provided annotations resulted only partially accurate, caused by the lack of sensitivity of the previous software used for the same analysis. Biologists considered satisfactory the obtained results after inspection.

Semantic segmentation performances were evaluated in terms of Intersection over Union, a common metric used to estimate the overlap between predictions and ground-truth masks, calculated separately for each class. The achieved value for intact nuclei segmentation in validation samples, equal to 0.8599 was retained satisfactory; for apoptotic nuclei instead, the achieved value of 0.5105 did not suggest as good performances at first glance. This was not confirmed in test phase, resulting in a correct detection of all apoptotic events reported in the available annotations and a number of

apparent false positive detections, consistent with the partial accuracy of annotations, then corrected as true apoptotic events after biologists' inspection. Thus, results confirmed a considerable accomplishment of the task, achieved with limited data employment and thanks to the extensive use of data augmentation. The computational time required for a single image was of the order of seconds, guarantying a significant time saving.

Comparing the two methods, the extra predictions in respect to the available annotations resulted consistent between the two, confirming the capability of detecting real apoptotic events in those positions. The first method resulted more sensible, with always a greater number of detections. The second method has instead the advantage of more precisely localizing apoptotic events and being computationally more efficient.

Conclusions

We present two CNN-based solutions for individuation of apoptosis in microscopic images. The proposed approaches can speed up biologists' work and reduce human errors' incidence.

In future developments, the image analysis framework of the first method will be improved for what concerns the accuracy-prediction time trade-off. Considering the appreciable prediction capabilities, a reduction of computational time will be targeted.

For the second method instead, an improvement of the learning process will be targeted, by properly redesigning loss and metric definitions, along with an increase in the quantity of input data and quality of corresponding labels.

The two frameworks will be also applied to new class of images to study different cells populations.

Sommario

Introduzione

L'apoptosi è la morte cellulare programmata, un processo altamente regolato responsabile del turnover cellulare nei tessuti viventi. La sua alterazione e gli eventi ad essa associati sono coinvolti in varie patologie, tra cui il cancro. L'identificazione di cellule apoptotiche in tessuti patologici fornisce preziose informazioni sulla progressione della malattia e aiuta a valutare l'efficacia di terapie putative.

Varie modalità di detezione per l'apoptosi sono attualmente disponibili, e si basano in gran parte su analisi empiriche da parte dei biologi. Il processo risulta laborioso, dispendioso in termini di tempo e soggetto a errori umani. Per superare questi limiti, proponiamo sistemi automatici computerizzati per rilevare e quantificare cellule apoptotiche in immagini microscopiche, con l'obiettivo di ottimizzare i tempi di analisi, migliorare l'accuratezza, e abbattere l'incidenza di errore umano.

Materiali e Metodi

I due metodi proposti sono basati su tecniche di Deep Learning [1] [2] applicate a immagini di microscopio confocale di cellule umane derivanti da melanoma metastatico [3].

Entrambi sono inseriti in una pipeline di lavoro che richiede minima preelaborazione delle immagini e post-processamento dell'output per l'estrazione dei parametri di interesse.

Il primo metodo, basato sul rilevamento di oggetti, consiste in una rete neurale convoluzionale (CNN) simile a VGG16, addestrata come classificatore binario per identificare nuclei apoptotici in piccole finestre. Nuclei intatti e apoptotici sono stati manualmente ritagliati dalle immagini originali.

Il problema dello sbilanciamento delle due classi è stato indirizzato con la costruzione di un dataset bilanciato tramite un sotto campionamento dinamico della classe dominante durante il processo di addestramento (i.e. la rete neurale è allenata con immagini della classe maggioritaria che cambiano con le iterazioni del training) e una funzione di perdita realizzata appositamente per minimizzare le predizioni falsi negativi.

Una volta addestrato il classificatore, questo è applicato sistematicamente sulle immagini di test con una procedura a sliding-window: l'immagine di input è scansionata con una

specifica risoluzione in numero di pixels, e per ogni posizione la CNN è stata applicata. Se il modello identificava un evento apoptotico, la finestra veniva segnata con un bounding-box. In caso di bounding-box sovrapposti, il nucleo veniva identificato con il loro centroide.

Il secondo metodo compie una segmentazione semantica sull'intera immagine, distinguendo tra nuclei intatti, frammenti apoptotici e il rimanente background. Per questo caso, un'architettura UNet è stata scelta e adattata al caso specifico, insieme a una funzione di perdita personalizzata per risolvere il problema dello sbilanciamento delle classi.

Risultati e Discussioni

Per valutare i risultati del framework di detezione proposto, sono stati usati diversi indici standard di performance, risultando in valori soddisfacenti per accuratezza, precisione e recall sui campioni di validazione, tutti sopra 0.98. Due dimensioni di input sono state investigate. In entrambi i casi, il passo della finestra è stato fissato a metà e un quarto della dimensione di input. L'assegnazione di un valore adeguato a questi due parametri è risultata essenziale per il raggiungimento di risultati soddisfacenti. La dimensione di input influenza le performance sulla localizzazione e un'appropriata discriminazione di eventi molto vicini tra loro come entità separate. Il passo della finestra invece è fondamentale per una rilevazione completa e accurata di tutti i frammenti apoptotici.

Lo svantaggio maggiore della detezione con sliding window è il costo computazionale. Riducendo il passo si ottiene una localizzazione più accurate, ma porta allo stesso tempo lo svantaggio di aumentate il tempo computazionale di un fattore 4.

Scegliendo invece un passo più largo, il numero di finestre da predire si riduce, ma la meno fine granularità impatta sia sulle performances di detezione che di localizzazione. Per questo motivo, un adeguato trade-off deve essere scelto tra questi due fattori.

Non è stato possibile comparare i risultati ottenuti con un gold-standard, dal momento che le annotazioni fornite sono risultate solo parzialmente accurate, a causa di una mancanza di sensitività del software precedentemente usato per la stessa analisi. I biologi hanno considerato soddisfacenti i risultati ottenuti dopo loro ispezione.

Le performances della segmentazione semantica sono state valutate in termini di Intersezione su Unione, una metrica comune usata in segmentazione per valutare il grado

di sovrapposizione tra la predizione del modello e le maschere di target, calcolata separatamente per ogni classe di interesse. Il valore raggiunto sulla segmentazione dei nuclei intatti, pari a 0.8599, è stato ritenuto soddisfacente; per la segmentazione dei nuclei apoptotici invece, il valore raggiunto, pari a 0.5105, non suggeriva performances così buone a primo impatto. Questo non è stato confermato nella fase di test, risultante in una corretta rilevazione di tutti gli eventi apoptotici riportati nelle annotazioni a disposizione, e a un apparente numero di predizioni false positive, consistente con la parziale accuratezza delle annotazioni, poi corrette come veri eventi apoptotici dopo ispezione da parte dei biologi. Conseguentemente, i risultati hanno confermato un considerevole compimento del task, raggiunto con un impiego limitato di dati e l'uso estensivo di data augmentation. Il tempo computazionale richiesto per una singola immagine è dell'ordine dei secondi, garantendo un considerevole risparmio di tempo per i biologi.

Confrontando i due metodi proposti, le predizioni in più rispetto alle annotazioni disponibili sono risultate consistenti tra i due approcci, confermando la capacità di entrambi di rilevare correttamente reali eventi apoptotici in quelle posizioni. Il primo metodo è risultato più sensibile, con un numero di predizioni sempre maggiori della segmentazione. Il secondo metodo ha invece il vantaggio di localizzare in maniera più precisa gli eventi apoptotici e di essere computazionalmente più efficiente.

Conclusioni

Presentiamo due soluzioni basate su CNN per l'individuazione di apoptosi in immagini microscopiche. Gli approcci proposti possono velocizzare il lavoro dei biologi e ridurre l'incidenza di errori umani.

In sviluppi futuri, il framework di analisi delle immagini sarà migliorato per quel che riguarda il trade-off tra accuratezza e tempo di predizione. Considerando le apprezzabili capacità di predizione, una riduzione del tempo computazionale verrà indirizzata.

Per il secondo metodo invece, un miglioramento del processo di addestramento verrà mirato, riprogettando le definizioni di funzione di perdita e metriche, insieme a un aumento della quantità di dati impiegati e qualità delle corrispondenti maschere.

I due frameworks saranno anche applicati a nuove classi di immagini per studiare diverse popolazioni cellulari.

Contents

List of Figures

List of Tables

List of Abbreviations

DL	Deep Learning
DNN	Deep Neural Network
ANN	Artificial Neural Network
NN	Neural Network
FFNN	Feed Forward Neural Networks
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
BN	Batch Normalization
ACAS	Automated Cellular Analysis System
SW	Sliding Window
TP	True Positive
FP	False Positive
FN	False Negative
IoU	Intersection over Union

1. Introduction

1.1 Apoptosis

Cell death can be caused by cell intrinsic or environmental perturbations, that lead to uncontrolled release of the pro-inflammatory cellular content, a process called necrosis. On the other hand, apoptosis is known as a mechanism of programmed cell death, which involves the regulated elimination of cells. It is considered a vital, spontaneous process in tissue homeostasis, as normal cellular turnover, but it can also be triggered in response to known stimuli (e.g., radiation or chemical-induced cell death), or when cells are damaged during disease progression. An unbalance of apoptosis (that has the opposite role in respect to mitosis), is observed in conditions such as neurodegenerative diseases, ischemic damage, autoimmune disorders and cancer [4].

1.1.1 Apoptosis History

The field of apoptosis has always attracted biologists' interest and has been independently described at least five times over the past 150 years [5], with initial studies starting in the mid-1800s. However, a detailed morphological description of this phenomenon is quite recent and dates back to 1972 [6].

The concept of apoptosis was originally mentioned by Carl Vogt in 1842 during his studies on developmental cell death in toads [7]. The first cellular description was given by Walther Flemming in 1885 that, with his drawings on cell shrinkage and apoptotic body formation, described spontaneous cell death as a physiological event, providing a description of the apoptosis' hallmarks for the first-time [8].

In mid 1900s Alfred Glucksman in 1951 described karyopyknosis as a distinct form of cell death, characterized by nuclear shrinkage and chromatin condensation [9].

Taking inspiration from these experiments, the pathologist John Kerr began to describe an unusual form of cell death seen following ischaemic injury in liver tissue and coined the term shrinkage necrosis in the 1960s [10].

In the 1970s Kerr, together with Andrew Wyllie and Alister Currie, characterized and defined this programmed form of cell death, calling it *apoptosis* (from Greek, this word was used to describe the dropping of petals from flowers, or leaves from trees). However,

this observational work was initially met with little appreciation, and it was only following the publication of Andrew Wyllie in 1980 on the induction of thymocyte apoptosis associated with the activation of an endogenous endonuclease [11], that these findings began to awaken interest and the biological importance of apoptosis was established [12].

After that, the elusive endonuclease DNA fragmentation factor subunit beta (DFFB, also known as CAD) cloning and characterization by Shige Nagata in 1998 [13] finally provided a biochemical marker for apoptosis, and the study of the role and underlying mechanism of this phenomenon became possible.

At this point, the study of apoptosis increased exponentially, resulting in publication of papers regarding aspects of this phenomenon from practically every field of modern biology. Of note, is the work of Bob Horvitz in 2002, which won the Nobel Prize for his fundamental contribution on the biology of apoptosis [14].

1.1.2 Morphological Description

Apoptosis exists in a delicate and tightly controlled balance with mitosis, and it follows highly controlled molecular events that induce defined morphological changes, observable by light or electron microscopy. The duration of each specific event is difficult to determine, but the overall process is very rapid (within 24h).

This phenomenon includes 4 sequential steps [15]:

I. Separation and cellular shrinkage
II. Nuclear or chromatin condensation
III. Cellular fragmentation and formation of apoptotic bodies
IV. Phagocytosis of apoptotic cells or bodies

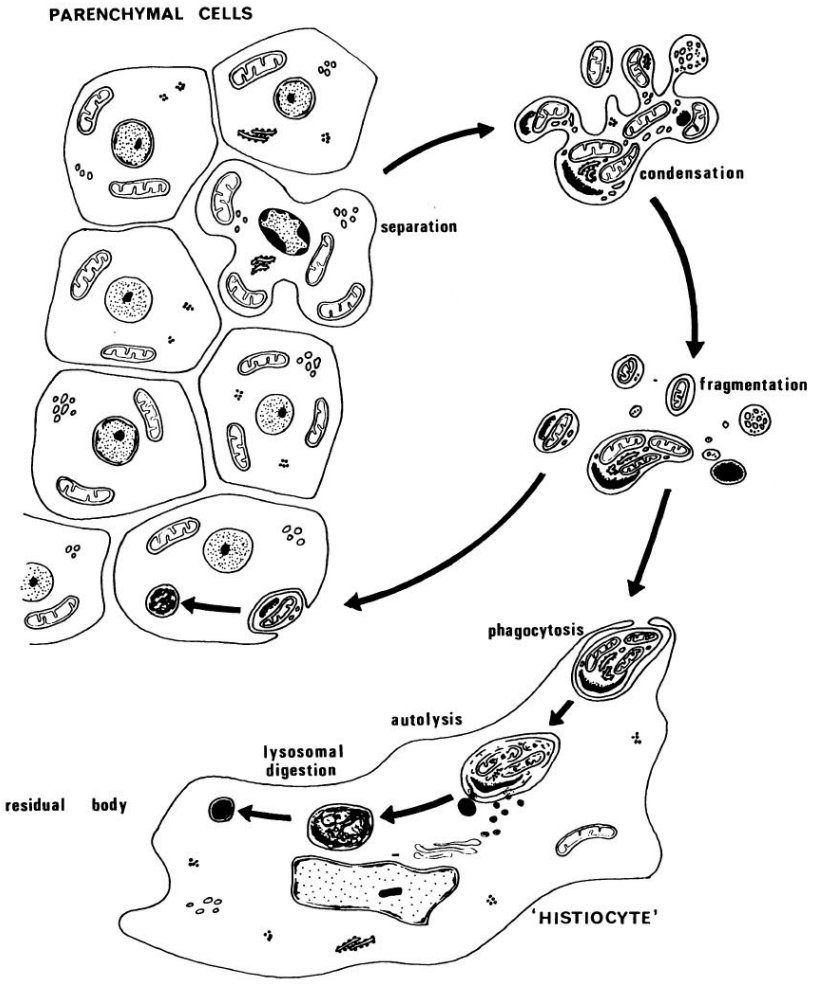


Fig. 1: After a separation from the other cells forming the tissue, the first visible morphological change corresponds to a decrease in overall cell size (condensation). The plasma membrane becomes more irregular, until the protuberances formed on the cellular surface, called blebs, separate into apoptotic bodies (fragmentation). Clusters of apoptotic fragments present in the intracellular space undergo phagocytosis by macrophages, parenchymal cells, and neoplastic cells, where they are subsequently degraded in phagolysosomes.

Step I: Separation and Cellular Shrinkage

A loss of cell junctions and other specialized plasma membrane structures occurs during the initial phase of apoptosis, which leads to a withdrawal mode of the cell from the surrounding. As a consequence of laminins and actin filaments cleavage, the cytoplasm begins to shrink, it becomes denser and organelles more tightly packed. The contraction of cytoplasmic volume is apparently associated with loss of intracellular fluid and ions [16].

Step II: Nuclear or Chromatin Condensation

This stage goes through very complex biochemical and molecular changes: chromatin condenses, it shatters in an orderly fashion into one or more masses, and migrates toward the periphery of the nuclear membrane. After that, the nucleus breaks into several fragments, which appear dense and dark under electron microscope investigation [17], [18].

Step III: Cellular Fragmentation and Formation of Apoptotic Bodies

Cells continue to shrink, packaging themselves into a form that allows for easy engulfment by macrophages, parenchymal cells and other neoplastic cells, with a more and more convoluted outline and extensive blebbing forming of the cell surface. Morphological changes in the membrane appear towards the end of the apoptotic process. As consequence, the cell breaks up into several membrane-bound smooth-surfaced 'apoptotic bodies' that contain a variety of tightly compacted organelles and/or some nuclear fragments. The content of each apoptotic body depends on the cytoplasmic protuberance components that give rise to it. For this reason, some bodies contain only cytoplasmic material, other present also condensed nuclear material.

Apoptotic cells also elicit biochemical changes on the plasma membrane surface that trigger the macrophage response and promote their phagocytosis, such as the translocation of phosphatidylserine from the inner leaflet of the cell to the outer surface [15], [19].

Step IV: Phagocytosis of Apoptotic Cells or Bodies

Apoptotic bodies are different in size and shape, and there is not a predefined number of apoptotic bodies formed from one cell. They are typically phagocytosed by neighbouring cells or macrophages. These cells are responsible for removing apoptotic fragments from tissues in a clean and tidy fashion that avoids many of the problems associated with necrotic cell death. Since no cellular content is released and apoptotic bodies are quickly phagocytosed, production of inflammatory factors by the engulfing cells and consequent inflammatory response are not triggered.

The endocytosed apoptotic bodies are indeed rapidly degraded by a series of enzymes within lysosomes, and the adjacent cells move around or, if necessary, proliferate to replace the gap created by the just deleted apoptotic cell [20], [21].

1.1.3 Molecular Mechanisms of Apoptosis

Apoptosis is a tightly controlled, energy-driven process involving a carefully orchestrated series of molecular events, which can be divided into two stages: induction and execution [22].

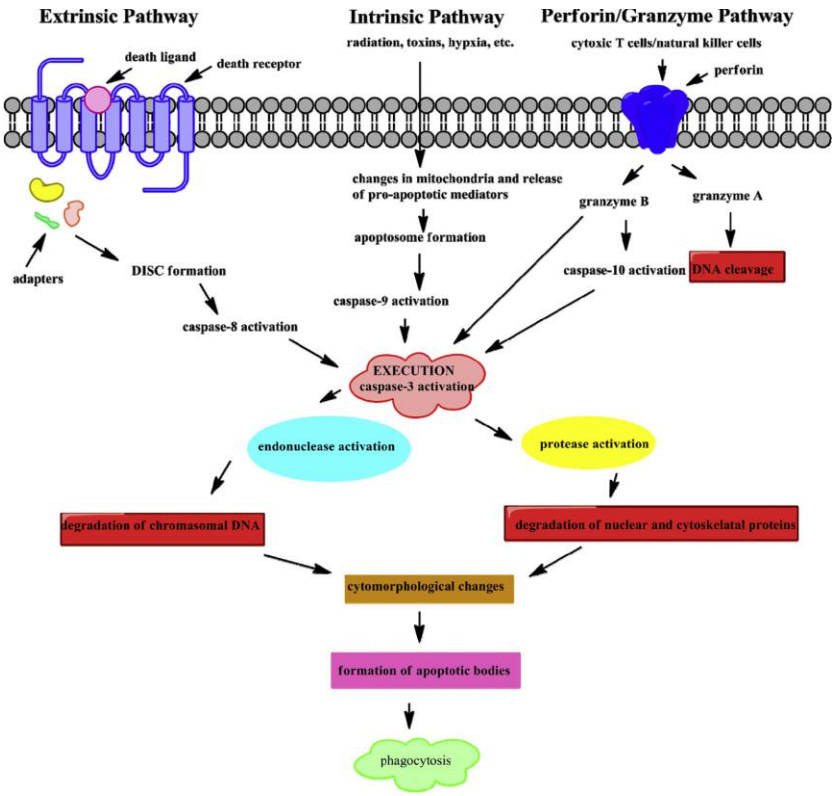


Fig. 2: Schematization of apoptotic pathways: induction of apoptosis can be achieved via the extrinsic, intrinsic, and perforin/granzyme pathways. Each pathway requires different stimuli and caspases to initiate apoptosis. However, these initiation pathways all end with the activation of caspase 3, beginning the execution phase of apoptosis. Activation of endonucleases and proteases result in the degradation of nuclear DNA and nuclear and cytoskeletal proteins. This gives rise to the classic cytomorphological changes seen in apoptotic cells, including cell shrinkage, chromatin condensation, formation of cytoplasmic blebs, and apoptotic bodies. Phagocytosis of the apoptotic bodies is the final step.

Induction

This step relies on death-inducing signals to stimulate the proapoptotic transduction cascades. Induction of apoptosis occurs through different apoptotic pathways: the intrinsic (or mitochondrial) and the extrinsic (or death receptor) pathways are the most common; in addition, T-cells can also initiate apoptosis through either granzyme A or B. Caspases are central to apoptotic mechanisms, as they are involved both in the initiation and in the execution processes.

Intrinsic pathway. The intrinsic pathway is initiated within the cell. There are a variety of nonreceptor-mediated stimuli, such as irreparable genetic damage, hypoxia, extremely high concentrations of cytosolic Ca2+ and severe oxidative stress that can trigger the initiation of this pathway.

These stimuli cause an increased permeability of the inner membrane of mitochondria, resulting in a loss of mitochondrial transmembrane potential, and leakage of proapoptotic molecules into the cytoplasm. Among them, cytochrome-c and second mitochondria-derived activator of caspase (SMAC) activate the caspase-

dependent mitochondrial pathway, and apoptosis inducing factor (AIF), that are instead released later during apoptosis.

This pathway is closely regulated by a group of proteins belonging to the Bcl-2 family (further addresses in 1.1.4).

Extrinsic pathway. Apoptosis is initiated by the activation of transmembrane receptors, known as death receptors, by death ligands. These death receptors have an intracellular death domain coupled to the apoptosis-inducing machinery. Activation of death receptors is controlled in many cases by inducible *de novo* expression of respective death ligands. Besides death receptors, there are also death effector downstream to the process.

When a death ligand binds to a death receptor, its intracellular domain recruits an adaptor protein and the whole ligand-receptor-adaptor protein complex is known as the death-inducing signalling complex (DISC). This initiates the assembly and activation of pro-caspase 8. The activated form of the enzyme, caspase 8 initiates apoptosis by cleaving other downstream or executioner caspases.

Granzyme pathway. This pathway is used to clear virus-infected or transformed cells: sensitized cytotoxic T-lymphocytes (CTLs) secrete perforin, inducing the formation of

pores in the plasma membrane of target cells and allowing serine proteases granzyme A and B to enter the target cell. Granzyme B directly activates caspase-3 which in turn leads to the classical execution pathway.
Granzyme A plays a role in CTL-induced apoptosis through the activation of a caspase-independent pathway.

Execution

The intrinsic and extrinsic pathways both end at the same execution phase, involving the activation of a series of execution caspases. These caspases activate cytoplasmic endonucleases, which degrade nuclear material, and proteases that break down nuclear and cytoskeletal proteins.
The most important executioner caspase is caspase-3, which is activated by initiator caspase-8 for the extrinsic pathway and caspase-9 for the intrinsic. Caspase-3 also plays a role in the reorganization and breakdown of the cell into apoptotic bodies.

1.1.4 Apoptosis in Cancer

A number of underlying principles can be identified as hallmarks of cancer. These principles are the rules that govern the multi-step transformation of human cells into malignant cancer, which are characterized by molecular, biochemical and cellular traits and acquired capabilities/alterations [23].
Tumorigenesis is based on subsequent genetic changes that confer different growth advantage and lead to the progressive conversion of normal human cells into invasive cancer. As a result, tumor cells have thus defects in common regulatory circuits that govern normal cell proliferation and homeostasis.
The numerous types of cancer create uncertainty about which and how many distinct regulatory circuits within each cell must be disrupted to become cancerous. However, there are six essential acquired capabilities that collectively dictate malignant growth and metastatic dissemination [23], which are summarized in Fig. 3:

i. Self-sufficiency in growth signals;
ii. Insensitivity to growth-inhibitory (antigrowth) signals;
iii. Evasion of programmed cell death (apoptosis);

iv. Limitless replicative potential;

v. Sustained angiogenesis;

vi. Tissue invasion and metastasis;

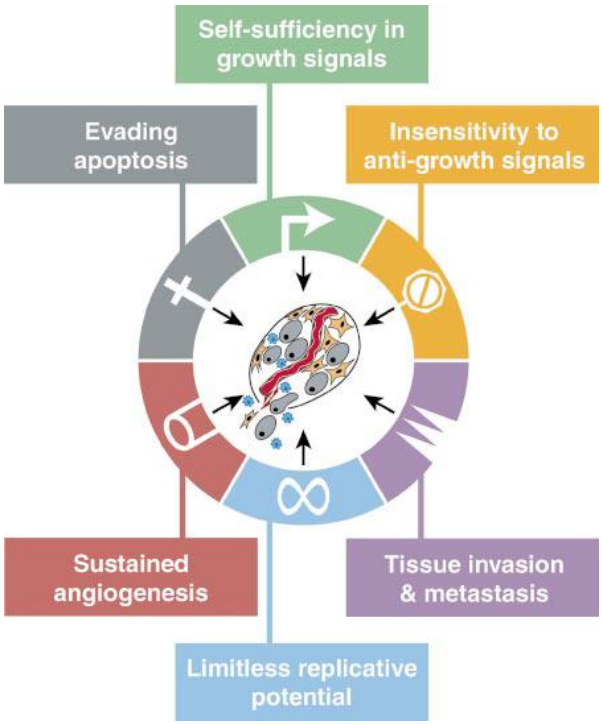


Fig. 3: Acquired Capabilities of Cancer. Cancers have acquired the same set of functional capabilities during their development, including evasion of apoptosis.

These distinct and complementary capabilities represent the foundations of cancer, being each of these physiological changes shared by almost all types of cancer. In the next paragraphs, we will specifically focus our attention on the evasion from apoptosis, indicating strategies by which this capability is acquired in cancer cells.

Evasion from Apoptosis

Apoptosis is a key player in tissue homeostasis, contributing to the regulation of normal cell populations. An adult human body generates about 60 billion cells per day, so, as consequence, an equal number of cells must die by apoptosis to maintain tissue homeostasis. Therefore, a deregulation of apoptosis can lead to an accumulation of cells. Apoptotic evasion, observed in cancer, plays an integral role in tumour growth, progression and resistance to therapy [24]. At the same time, apoptosis can be re-triggered in tumour cells resulting from application of therapeutic agents. For example, an increase

in apoptosis is shown after irradiation (which cause DNA damage), or chemotherapy (which generates cellular or DNA damage) and subsequent cell death [6].

Two classes of components are involved in acquiring resistance toward apoptosis at a molecular level:

- *Sensors* of the apoptotic pathway, are responsible for monitoring the extracellular and intracellular environment and to check is abnormalities are present, establishing whether a cell should go through apoptosis or not;
- *Effectors* of the apoptotic pathway, that are mediated by signals coming from the first class.

Sensors include:

i. Cell surface receptors (IGF-1R and IL-3R, FAS and TNF-R1) that bind survival/death ligands;

ii. Intracellular receptors, which monitor cell's wellbeing and activate death pathway upon detection of internal abnormalities, for example DNA damage, signalling imbalance, survival factor insufficiency, hypoxia, abrogation of cell–matrix or cell–cell adherence-based survival signals.

Effectors of apoptosis, responsible for cytochrome C release, include:

i. Members of the Bcl-2 family of proteins, with either pro-apoptotic (Bax, Bak, Bid, Bim) or antiapoptotic (Bcl-2, Bcl-XL, Bcl-W) functions that govern mitochondrial death signalling through cytochrome C release~~;~~

ii. The p53 tumour suppressor protein which upregulates the expression of proapoptotic Bax in response to DNA damage, in turn stimulating mitochondria to release cytochrome C;

iii. Intracellular proteases termed caspases, in particular caspases-8 and caspases -9, activated by death receptors such as FAS or by the cytochrome C released from mitochondria. These caspases trigger the activation of other effector caspases that execute the death program.

More specifically, Bcl-2 family has a regulatory role, controlling the rate of apoptosis by counterbalancing pro- and antiapoptotic proteins. About the latter, Bcl-2 is an inhibitor

of apoptosis, which suppresses two proapoptotic triggering proteins, Bax and Bak, embedded in the mitochondrial outer membrane.

Normally (if non inhibited) Bax and Bak disrupt the integrity of the outer mitochondrial membrane, causing the release of a proapoptotic signalling protein, cytochrome c. This activates a cascade of caspases that act via their proteolytic activities to induce apoptosis [25].

How cells acquire resistance to apoptosis

Alterations of components causing the inactivation of this apoptotic machinery support uncontrolled tumour proliferation. Thus, resistance to apoptosis can be acquired through a variety of strategies that translate into overexpression of antiapoptotic effectors or inhibition of proapoptotic effectors.

Tumour cells evolve a variety of strategies to limit or circumvent apoptosis. The first discovery that BCL-2 oncogene had anti-apoptotic activity when overexpressed [26], opened the investigation of apoptosis in cancer at the molecular level.

The most common alteration that brings a physiological cell into cancerogenous one relies on p53 mutations. It has been demonstrated that the functional inactivation of the p53 tumour suppressor protein leads to development of tumours, resulting in the removal of a key component of the DNA damage sensor that can induce the apoptotic effector cascade. Apoptotic signals evoked by other abnormalities, like hypoxia and oncogene hyper expression, use as mediator p53 protein and are impaired when p53 function is lost.

Tumours may achieve the same purpose by increasing expression of antiapoptotic regulators (Bcl-2, Bcl-xL) or of survival signals (Igf1/2), by downregulating proapoptotic factors (Bax, Bim, Puma), or by short-circuiting the extrinsic ligand-induced death pathway. Abrogation of FAS death signal result in evasion of apoptosis too.

1.2 Detection Techniques for Apoptosis

Detection of apoptosis in biological samples involves capturing morphological changes occurring at cellular level [27]. There is a large number of available assays to assess apoptosis, each of them specific for a certain phase in the cascade of apoptotic processes, and each one carrying advantages and disadvantages [28], [29]. Therefore, a proper choice of the method to detect apoptosis is extremely important.

Standard techniques, such as electron microscopy, TUNEL assays, flow cytometry, together with new emerging ones, such as microfluidic devices, single molecule spectroscopy, electrochemical methods, are here reported.

Electron microscopy. It is used to assess morphological changes of apoptosis such as membrane blebbing, chromatin condensation and organelles. If immunochemical staining is employed, then chemical information can be obtained in addition to morphological information. Advantages of this technique are the high spatial resolution, and the fact that cell morphology preserved. There are however some limitations here reported: cells are not left viable after analysis, making this an "end point" technique; cells are usually detected at the last stages of apoptosis, giving no information about the early stages; limited chemical information is provided; huge time and a high skill level are needed to prepare and measure cells.

Genomic and proteomic assays. Unlike microscopy investigations, they provide increased information about chemical content at the expense of spatial resolution. The drawback is that they are invasive methods, so cell viability is not preserved.
Several analytical methods are applied, such as the *DNA ladder assay*, that take advantage of the breaks in chromatin that result in the formation of DNA fragments. Unfortunately, DNA ladder assay suffers from low sensitivity, and apoptosis is detected only at late stages. Temporal resolution is low too.
Another method is represented by the *comet assay*, which detects different forms of DNA breakage. Its advantages are higher sensitivity, more specific information about the extent of damage and heterogeneity of DNA, and ease of use.
Protein-based assay detects the release of cytochrome c, up- or down-regulation of key inhibitory proteins, and the activation of caspases. Cytochrome c is one of the earliest detections of apoptosis, allowing higher temporal resolution.

Caspase activity assays. The ability of caspases to selectively cleave proteins when activated by apoptosis, makes them a good biochemical marker for detecting apoptotic events. Thus, sensitive caspase assay, combined with flow cytometry or microscopy, is able to detect intracellular caspase activation after induction of apoptosis.

One example is caspase activity investigated by fluorescence correlation spectroscopy (based on the use of new fluorescent probes specific to one or more caspases) which provides several advantages, including high temporal resolution, ability to leave cells viable for further analysis, no need for transfection of the cell.

Microfluidic devices. These emerging devices can perform on-chip flow cytometry. They are commonly made of traps designed to capture and hold cells in place. Once trapped, fluorescence imaging is used to monitor apoptosis.
Another type of microfluidic device uses affinity interactions to capture cells that undergo apoptosis. One advantage is that they allow long-term monitoring of captured cells as they progressed through the apoptotic process, providing precise information on the temporal dynamics of apoptosis.

Electrochemical methods. They represent a reasonable alternative method to detect apoptotic cells thanks to the fact that they do not require complex and expensive instrumentation. They are based on the use of electrochemical sensors, one type is made out of a gold electrode modified with a helix peptide ferrocene (Fc) which is used to detect active caspase-3 detection, resulting in the loss of reporter Fc. In this way, the decrease in the peak current is a result of caspase 3 activity and could therefore be used to detect apoptosis. The major advantages of this method include the simplicity and convenience (as it does not require complicated instrumentation, only a modified electrode that is easy to create), throughput and ease of use. The main limitation resides in the inability to measure apoptosis at the single cell level or provide cell-specific information over a large cell population.

Spectroscopic techniques. They rely on fluorescence detection of immunochemical labels or fluorescent probes that react to the cell environment during apoptosis. They are read out in a variety of formats such as flow cytometry, microscopy, etc. Several spectroscopic techniques are available in studying apoptosis, among which:
Labelled annexin V is applied in flow cytometry or light microscopy to detect mid and late stages of apoptosis. Annexin V binds to the exposed phosphatidylserine on the surface of the cell membrane during apoptosis.

TUNEL assay is based on DNA fragmentation. Using confocal microscopy, the TUNEL method provides in situ visualization of the DNA fragmentation process on a single-cell level. It has the potential to give rise to ultrastructural aspects of the apoptosis process. The TUNEL method can also be coupled with immunofluorescent labelling to increase information content.

Spectroscopic probes for membrane potential are also indicators of apoptosis in early stages. This because the membrane potential is an indicator for the increased permeability of mitochondria resulting in the release of cytochrome c.

Membrane potential can be tracked using several different dyes, most used probes are JC-1, rhodamine 123, and mitotracker red.

Flow cytometry. It is a common method used in conjunction with many other techniques in apoptosis detection. High cell counting, multi-parameter detection, possibility of sorting cells, cell throughput, are some properties that makes this technique well suited to cell analysis. Flow cytometry can measure apoptosis at a single point in time for a cell population, giving information of the ratio of apoptotic cells and cell viability. In addition, it can be used to assay caspase activity, DNA cleavage or viability dyes. Fluorescence-based assays that do not rely on morphological detection can be analysed by flow cytometry. For example, assays using Annexin-V, DNA content, caspase activity, membrane potential, and other probes can be used with flow cytometry to achieve large cell counts and detect rare cells in a population.

Instead, for analyses where morphology is an indicator of apoptosis or where the same cells have to be tracked over time, other forms of imaging detection and fluorescence microscopy are more suitable.

Among various techniques, mostly when cell morphology and topology are of interest, fluorescence-based assays coupled with image acquisition through confocal microscopy have been identified as the most informative method for apoptosis detection. A more detailed description of fluorescence confocal microscopy technique is provided in the following paragraphs.

1.2.1 Confocal Microscopy

Confocal laser scanning microscopy (CLSM) is an optical imaging technique which offers several advantages over conventional widefield optical microscopy, since it is capable of increasing resolution and contrast by means of a spatial pinhole that blocks out-of-focus light in image formation. Moreover, it allows to control the depth of field, eliminating or reducing background information away from the focal plane (that leads to image degradation), and to collect serial optical sections from thick specimens, and then reassembling the computer-stored images as a virtual 3-dimensional structure.
For these reasons, CLSM offers an alternative to traditional sectioning approaches.

Confocal microscopy is used in conjunction with fluorescent probes or antibodies to monitor the intracellular concentration of various physiologically important organic molecules. The choice of the fluorescent probe reflects the particular protein to be detected and can include activated caspases and histones, proteins that belong to the nucleus.

1.3 Limitation of Current Analysis

Once the microscopic images are acquired, biologists need to discern the apoptotic nuclei from the rest of cell population, and such discrimination is usually performed manually by experienced operators.
This leads to hours spent looking at images and counting each detected event, resulting in a very time-consuming operation. In addition, both cell classification and counting accuracy may depend on the capabilities and experiences of the operators, which generates inter-operator variability. Furthermore, manual counts can have a lower accuracy when performed serially for a huge amount of images, particularly on those reporting a highly dense population of cells.
Because of this, apoptotic detection could be a laborious procedure and can lead to fatigue-induced errors. Therefore, the necessity of an automated counting system is evident**.**
Among several automatic techniques, Deep Learning (DL) has shown successful results in many object detection problems, given its ability of automatically selecting the features

from the objects of interest, speeding up the processing of huge amount of data avoiding physiological human errors.

1.4 Proposed Work

Accordingly, this work proposes to develop an image analysis framework for automatically detecting apoptotic cells in microscopic images and discriminate between apoptotic and intact nuclei. Being able to automate the process can provide a huge time saving in the work of biologists and result in outputs that can be accurate and less prone to human errors and subjectivity, thus increasing statistical accuracy (because of the advantage of having a lower error rate per sample). Also, automated counting benefits from high reproducibility compared to manual counting.
In some cases, it is also possible to have a more in-depth analyses of the specimen, allowing for more detailed results (other parameters of interest) than simple counting.

The applied methodology of this thesis relies on deep learning techniques, which have shown a big success in the field of image processing and computer vision [30], among which convolutional neural networks (CNN) have become the preferable candidate for general object detection problems.

In this work two main applications of Deep Neural Networks have been investigated and employed to solve the problem of apoptosis detection: object detection and image segmentation. The proposed detection framework thus consists of two major independent components:

1. Apoptosis detection using a CNN-based classifier applied with a sliding-window technique;
2. Microscopic image segmentation, to discern apoptotic bodies from intact nuclei, count both of them, and compute useful indexes like the apoptotic ratio.

The first one is more a qualitative technique, i.e. it helps biologists to see whether apoptosis is present inside a cell population and localize them, but cannot provide useful parameter such as the computation of apoptotic ratio, which is important to understand whether the applied therapy has shown its effects.

For this reason, the second application was also developed, which is more useful for the extraction of such parameter since it is able to count both intact and apoptotic nuclei (hopefully in a more precise way in respect to sliding window technique since, because of the morphological nature of the apoptotic phenomenon and distribution of cells inside an image, it cannot be guaranteed that inside the inspected window always just one apoptotic event is present or more, and these two cases cannot be discriminate by the classifier).

The provided annotated data for the analysis made the sliding-window apoptosis detection technique more immediate, that is why it was the first one to be implemented. However, provided annotations were only partially accurate (sensitivity in annotated counts was about 57%) while specificity was very high (about 99%), this was sufficient to construct a model with better generalization capabilities, able to detect apoptosis which were lost in the analysis previously performed. So, the results obtained with this approach were useful also to guide a more precise manual labelling for image segmentation (second application).

2. Materials and Methods

2.1 Experimental Data

The images employed in this analysis are taken from organotypic in-vitro culture of human invasive and metastatic MV3 melanoma cells growing as 3D spheroids in extracellular matrix [3]. This culture strategy is then combined with in-vitro radiation therapy and automated image analysis to identify intact and apoptotic nuclei, as well as their position.

Thanks to their controlled experimental conditions and physiological similarities to in-vivo conditions, 3D spheroids represent an efficient solution to achieve reliable in vivo-like representation of therapy responses [31]. The 3D culture mimics the physiological cell-cell and cell-matrix contacts, and the matrix in which cells are embedded the physiological oxygen, growth factor and nutrient gradients. This allows for in-depth analysis of cell function, cancer resistance and therapy responses [32], avoiding in vivo strategies (with mouse models) that cannot be applied in routine-drug testing because of technical, financial and time considerations.

Organotypic 3D spheroid cultures are obtained by aggregating cells in suspension or forming multicellular colony, and subsequently embedding such cells or colony in a 3D basement membrane-like or fibrillar collagen-based ECM as physiological substrates for cell-matrix interactions, growth and invasion behaviors [33].

The resulting invasive cancer spheroid culture shows in-vivo like tumor organization and genetic heterogeneity and consequently, therapy responses with possible resistance mechanisms reflect physiological behaviors.

Despite the enormous impact of this technique on the investigation of cancer therapy response, current analysis tools of 3D cultures are restricted to spheroids growth and overall cell viability [32]. Even with high-throughput analysis approaches a fast estimate of the therapy response can be achieved, but still there is a lack in single cell analysis to monitor sub-regional heterogeneity of cell function.

Thus, semi-manual analysis of apoptosis and mitosis events by live-cell 3D confocal microscopy must be applied to investigate heterogenic therapy responses.

For these reasons, in vitro strategies involving spheroid growth and tissue invasion, combined with automated image analysis, are required for predicting therapy responses and identifying resistance at single cell level.

Monitoring the radiation response in 3D spheroid culture follows this pipeline:

i. Spheroid formation and collagen embedding

ii. *In vitro* irradiation therapy

iii. Microscopic detection of spheroid growth and invasion

iv. Automated image analysis to discriminate intact from apoptotic nuclei and identify their position, extracting single cell-based data on the surviving and apoptotic cell fractions, and their distribution in the invasion zone.

2.1.1 Spheroid Formation and Collagen Embedding

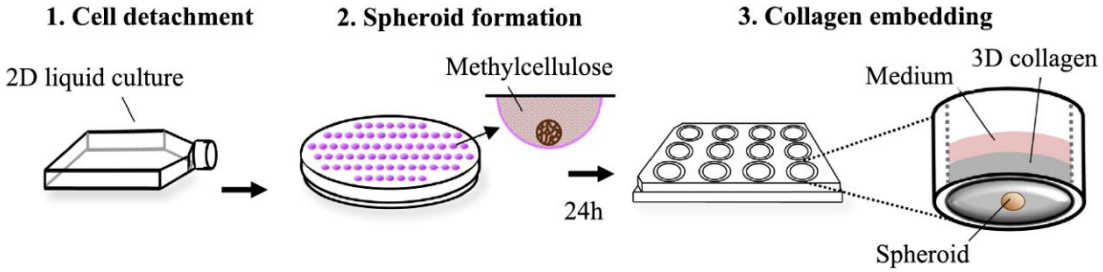


Fig. 4: 3D culture generation. 1. Cell detachment and mixing with methylcellulose and diluted collagen solution to provide spheroid formation; 2. Cell dropping into petri dish for 24h; 3. Spheroid embedding in collagen I solution.

Human invasive and metastatic MV3 melanoma cells were maintained in cell culture at 37°C and 5% CO2.

3D spheroids were generated by aggregation in a non-adhesive environment: cells from a sub-confluent 2D liquid culture were detached, counted, mixed with methylcellulose and diluted collagen solution to enhance spheroid formation.

Cell suspension was pipetted in small drops onto the inner surface of a cell culture dish and turned upside-down resulting in hanging drops.

After 24h, the cells aggregated to round and compact spheroids.

As last step, aggregated spheroids were embedded in collagen I solution which was allowed to polymerize at 37°C resulting in 3D fibrillar collagen containing one spheroid

per well. Spheroid cultures were maintained at 37°C and 5% CO2 to enable for growth and invasion into the 3D matrix.

2.1.2 Irradiation Therapy and Sample Preparation for Imaging

The cancer therapy provided in this study consisted in irradiation of prepared spheroids after 2 days of growth. Irradiation induces DNA damage and consequent cell death. It was performed with a single dose of 4Gy of X-rays at 320keV.

Spheroids were fixed with 4% PBS-buffered paraformaldehyde at the fifth day after irradiation, incubated with reagents for immunofluorescence staining, washed and then mounted on a glass slide, so that they were properly prepared for 3D confocal microscopy.

2.1.3 Confocal Microscopy of 3D Spheroids

After sample preparation was completed, images were acquired with confocal microscopy using a Leica SP5 laser scanner (three channels: H2B-eGFP, phalloidin and transmitted light).

3D images were obtained as 3 z-slices with inter-slice distance of 10 μm; therefore, the overall thickness of volumes was 20 μm.

Since the invasion zone of the spheroid increases with increasing depth, until a constant maximum extension between sequential slices is reached, imaging of 3D spheroids was standardized at the depth of maximum invasion radius (Fig. 5).

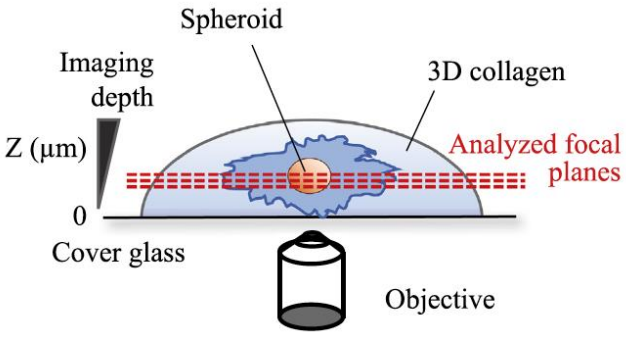


Fig. 5: Focal planes of invading spheroid imaged by 3D confocal microscopy.

2.1.4 Datasets

248 spheroids have been acquired, for a total of 744 single-channel, 16 bit-depth, gray-scale level intensity images (since for each spheroid 3 z-slices have been obtained), showing one quadrant of the invasion zone. Images are divided in 24 samples, each having a specific label and comprehensive of a number ranging between 4 and 6 spheroids. Moreover, for each sample, we have almost a correspondent number of spheroids which underwent the irradiation therapy (4Gy) and untreated spheroids (0Gy). Thus, the total number of spheroids 4Gy-irradiated is 122. Untreated spheroids are 126.

Statistics on the number of intact and apoptotic nuclei detected by ACAS

Considering the entire dataset of 248 spheroids, the total number of intact nuclei resulted to be 277’894 over 744 slices.

Mean value of intact nuclei per slice: 373.51 nuclei/slice

Standard Deviation of intact nuclei per slice: 266.12 nuclei/slice

The total number of apoptotic nuclei resulted to be 2’815 over 744 slices.

Mean value of apoptotic events per slice: 3.78 apoptotic events/slice

Standard Deviation of apoptotic events per slice: 4.03 apoptotic events/slice

Mean value and Standard Deviation of Apoptotic Ratio: 0.0159 ± 0.0199

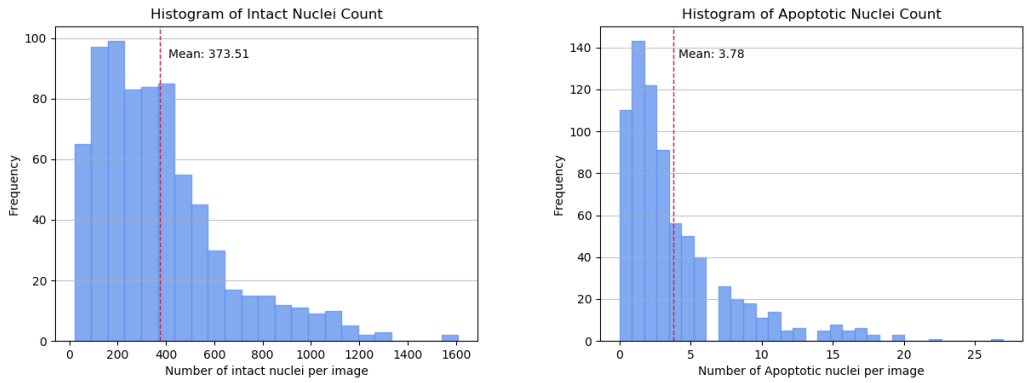


Fig. 6: Histograms of intact and apoptotic counts over the entire dataset.

Statistics on the number of intact and apoptotic nuclei detected by ACAS – 0Gy Dose

Considering the 126 untreated spheroids (0Gy dose), the total number of intact nuclei resulted to be 191’704 over 378 slices.

Mean value of intact nuclei per slice: 507.15 nuclei/slice

Standard Deviation of intact nuclei per slice: 289.61 nuclei/slice

The total number of apoptotic nuclei resulted to be 775 over 378 slices.

Mean value of apoptotic events per slice: 2.05 apoptotic events/slice

Standard Deviation of apoptotic events per slice: 1.93 apoptotic events/slice

Mean value and Standard Deviation of Apoptotic Ratio: 0.0061 ± 0.0102

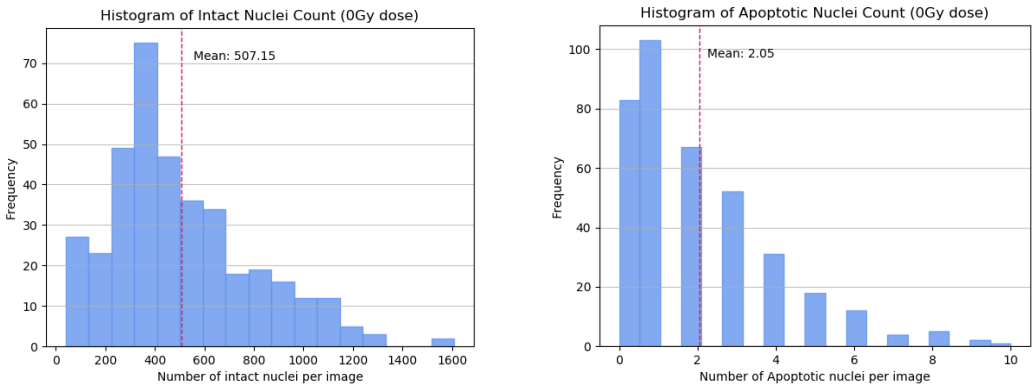


Fig. 7: Histograms of intact and apoptotic counts for 0Gy dose data.

Statistics on the number of intact and apoptotic nuclei detected by ACAS – 4Gy Dose

Considering the 122 irradiated spheroids (4Gy dose), the total number of intact nuclei resulted to be 86’190 over 366 slices.

Mean value of intact nuclei per slice: 235.49 nuclei/slice

Standard Deviation value of intact nuclei per slice: 140.89 nuclei/slice

The total number of apoptotic nuclei resulted to be 2’040 over 366 slices.

Mean value of apoptotic events per slice: 5.57 apoptotic events/slice

Standard Deviation of apoptotic events per slice: 4.79 apoptotic events/slice

Mean value and Standard Deviation of Apoptotic Ratio: 0.0260 ± 0.0224

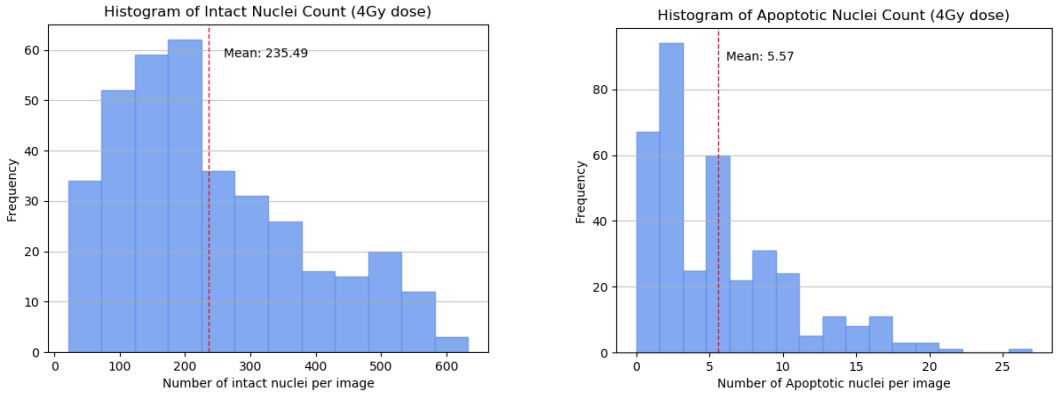


Fig. 8: Histograms of intact and apoptotic counts for 4Gy dose data.

A second dataset was provided in a second step. It was not exploited for Case Study 1 because apoptosis events were not annotated. It consists in multi-stack (not all equal in number as before) single-channel, 8 bit-depth, gray-scale level intensity volumes, coming from different cell populations, for a total of 372 images. They are all equal in size, 1024x1024 pixels, which made them more useful for Case Study 2.
Images were divided into three groups:

- High quality
- Medium quality
- Low quality

They were used mainly for test phase, then validated by an expert biologist.
A sub-sample of images belonging to high quality group was also used in a first stage to train UNet for semantic segmentation.

2.1.5 Automated Image Analysis

Acquired images were converted in TIFF format and processed through ImageJ, an open source image processing program based on Java for biological multidimensional images. For nuclei detection, H2B-eGFP channel was exploited (excitation 488um, emission 499-574um) since it shows the intensity of nuclear chromatin (which is a hallmark for apoptosis, characterized in initial stages by chromatin condensation).
After stitching together tiles from confocal microscope to form one coherent image for each z-stack, images of about 2000x2000 pixels were obtained, showing a section of the

spheroid. Centroid and radius of the core were manually identified and saved for later stages to automatically separate the core from the invasion zone and measure the distance of each nucleus from the spheroid center.

Automated segmentation of nuclei was performed through the Automated Cellular Analysis System (ACAS) provided by MetaVi Labs, Inc. (Austin, TX) and based on CUDA language and C#/.Net and GPU use, so that apoptotic fragments were discriminated from intact nuclei.

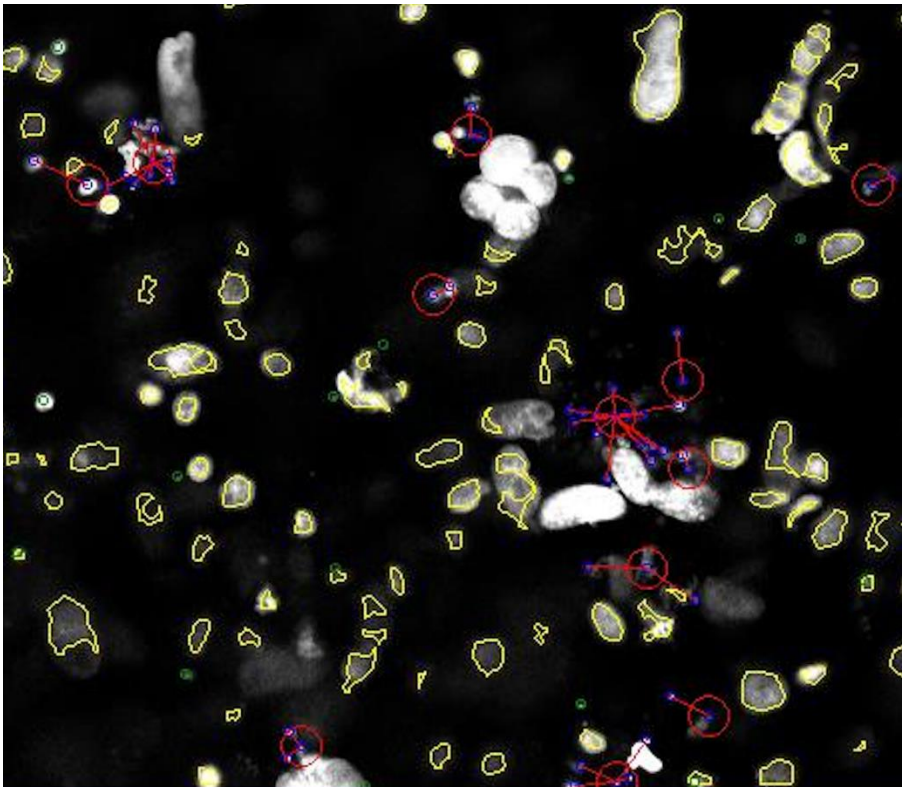
More specifically, the software generated the outputs shown in Fig. 9: number of intact nuclei, number of apoptotic events (consisting of several fragments located in proximity), x and y coordinate of each intact nucleus and centroid of the apoptotic event, and distance from the center of the core. The computed number of apoptotic events was divided by the total nuclei count (intact + apoptotic) to provide an estimation of the apoptotic ratio.

The absolute number of intact non-fragmented nuclei was taken in consideration as indicator of therapy resistance by comparing spheroids after treatment with non-treated controls. The pipeline followed by the ACAS software to detect apoptosis in these images includes these principal steps:

1) Identification of apoptotic fragments
2) Identification of intact nuclei
3) Fragments grouping and false-positive groups removal
4) Removal of nuclei overlapping with remaining fragments groups

An example of output provided by the ACAS software is schematized in Fig. 9.

a.



b.

7dpe_5dpi_22_4gy spheroid4__apoptotic

Slice	X-axis	Y-axis	Distance to center of core (um)
0	1258	339	841.410915929157
0	1586	572	868.461786147245
0	1306	1074	490.204708336464
0	1506	931	671.581954724939
0	1540	1136	648.483518423931

7dpe_5dpi_22_4gy spheroid4__intact

Slice	X-axis	Y-axis	Distance to center of core (um)
0	1756	1297	799.972930841287
0	1556	1297	648.607607326099
0	1781	1293	818.907605123749
0	1343	1291	487.446356334423
0	1510	1291	613.82722965723

c.

7dpe_5dpi_22_4gy spheroid4__report

Slice	Intact Nuclei	Apoptotic Nuclei	Apoptotic Ratio (%)
0	351	8	0.02228412
1	368	15	0.03916449
2	271	17	0.05902778
Average of all Z-Slices, Invasion Zone	330	13.33333	0.03883495

Fig. 9: Example of output from ACAS analysis for one image. a. For each image analyzed, ACAS gives in output the same image overlapped with the results from segmentation. Intact nuclei perimeters are highlighted in yellow, while apoptotic fragments' group are represented with a red circle (whose center indicates the center of mass of the group) and each blue dot represent one fragment found to belong to that group. Tables in b. and c. are simple examples of reports generated for each image: b. the first two annotate, for each slice (from 0 to 2, then 'slice 3' indicates the merge of all slices), x and y coordinates

of each detected nucleus or center of mass of the apoptotic group, together with their relative distance from the center of the spheroid core. c. The last report synthetizes the overall number of intact nuclei and apoptotic groups for each slice, and the computed apoptotic ratio as: Apoptotic Nuclei / (Intact Nuclei + Apoptotic Nuclei).

2.1.6 Challenges and Limitations of ACAS Software

We focus on the performances of the automated image analysis performed with ACAS, highlighting problems and limitations of the software.

Major difficulties in the analysis process of such images were the following:

i. background heterogeneity;

ii. large variations in diameter, shape and brightness of each nuclear fragment;

iii. variations in number and field size of a fragment group;

iv. variations in distances between the fragments;

v. variation in contrast of objects due to in-focus and out-of-focus position

vi. positional overlapping in the same z-slice of a 3D object.

The outlined issues imply that i) the apoptotic fragments are smaller in size and typically contain high intensity emission signal due to nuclear chromatin condensation; ii) intact nuclei are instead characterized by larger size, a wider pixel to pixel variation in their intensity profile, and nearly linear nucleus edge in larger round/ellipsoid shapes.

We outlined our discrimination to cope with the issues presented. Since one apoptotic event is composed by multiple apoptotic fragments, all particles located within a local radial distance of 20 μm between fragments were accounted as a single apoptotic event.

To avoid potential false-positive classification, the algorithm excluded events when the fragment was isolated from other or did not fulfil the size and intensity profile of an apoptotic fragment. Additionally, bright pixels of an intact nucleus could mimic apoptotic fragments resulting in false positive detection. These false-positive apoptotic events were removed by background corrections.

Automated analysis of non-treated controls (0Gy) and irradiated (4Gy) spheroids required a processing time of about 116 seconds per image, leading to outputs represented in number of alive and dead cells per image.

Comparing these outputs with manual analysis by an experienced operator as reference, it was found that sensitivity of the software in detecting intact nuclei was pretty good

(91.6% for 0 Gy spheroids and 87.2% for 4 Gy irradiated spheroids), and also specificity was acceptable (0 Gy: 84.6%; 4 Gy 81.9%).

Apoptosis recognition instead was characterized by a very low sensitivity of 36.6% for untreated samples and 57.1% for irradiated samples, but higher specificity of 97% (0 Gy) and 99% (4 Gy). Even though not optimal, these results were acceptable for the specific goal of the paper [3], since it was based on computing indexes for single cell resistance and heterogeneity of the therapy response. This was done by partitioning results obtained for two different cell populations, for example apoptotic index for treated and untreated spheroids, so that in the end the final result was insensitive to the lack of sensitivity for the analysis of each population. In conclusion, the performances of this software were acceptable for the analysis of this case since the detection of changes in apoptotic frequency was reliable thanks to the high specificity, but clearly demonstrate a sensible limitation of the software in detecting apoptotic events, which leads to exclude its use in other kind of analysis where a complete identification of all the apoptotic events is strictly required. The current work overcomes these limitations by proposing a more sensible, yet still specific, apoptosis detection framework exploiting deep learning techniques, in particular Deep Convolutional Neural Networks (CNN).

2.2 Deep Learning and Neural Networks

Over the past years, Deep Learning (DL) algorithms have achieved successful performances in biomedical image analysis, in terms of image recognition tasks, localization, segmentation [34]. These results are accompanied by the simplicity of such techniques, since they do not need complicate or long pre-processing. Deep Neural Networks (DNN) are able to automatically detect features of interest from the objects to be analysed.

For these reasons, DL represents a class of powerful methods able to extract information from medical data and integrate them in ad hoc model.

Models that will be analysed belong to the general class of Artificial Neural Networks.

2.2.1 Artificial Neurons and Neural Networks

Artificial Neural Networks (ANNs, or simply NNs) are computational models that emulate the nervous system to perform a large variety of tasks. More, they are well suited to solve problems like pattern recognition, clustering and classification. A general NN is composed by processing units, called *neurons*. They are typically connected to each other in a particular configuration called *architecture*. They form a *network*, where each neuron is also called *node*. The links between nodes have associated weights, which determine the strength of their connection.

Neurons

Each neuron, whose structure is illustrated in Fig. 10, carries the inputs coming from the other neurons which is connected to, calculates the weighted summation of such inputs, subtracts a bias and computes its output with a proper activation function.

The bias is useful to construct a successful learning model, because it shifts the activation function helping in getting better fit for the data (i.e. a better prediction function at the output). In practice, it is considered as a fixed input signal $x_0 = 1$ weighed such as $w_{k0} = b_k$, so the neuron structure is given by:

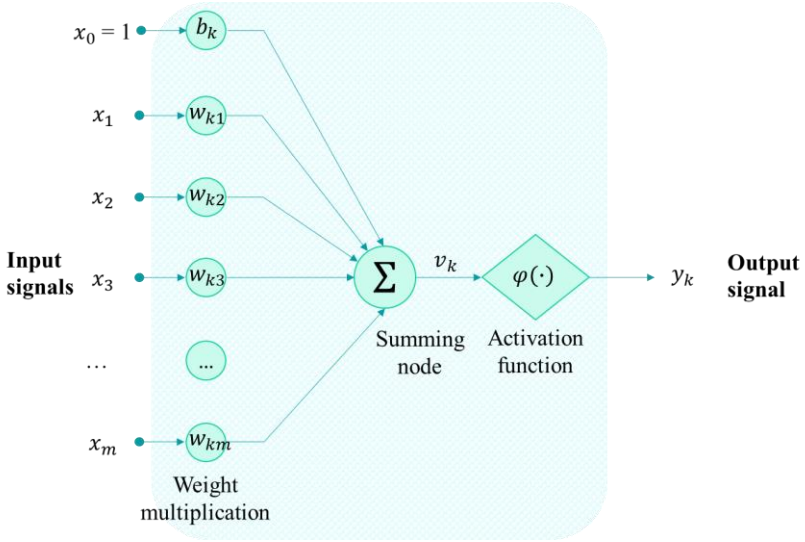


Fig. 10: Neuron structure with all the sequential mathematical steps highlighted.

Mathematically, the working principle of a generic k-th neuron is outlined in Fig. 7 and can be described by the formula:

$$y_k = \varphi\left(\sum_{i=1}^{m} x_i \cdot w_{ki} + b_k\right) \quad (1)$$

where:

x_i represent the input signals for neuron k; w_{ki} are the respective weights for each input of the neuron; b_k is the bias for neuron k; φ is the activation function that triggers the neuron's activation (generally it is common to all neurons belonging to a given layer).

Along this chain, the activation function controls the behaviour of the neuron, deciding whether it is activated or not depending on the result of the weighted summation of inputs and bias addition.

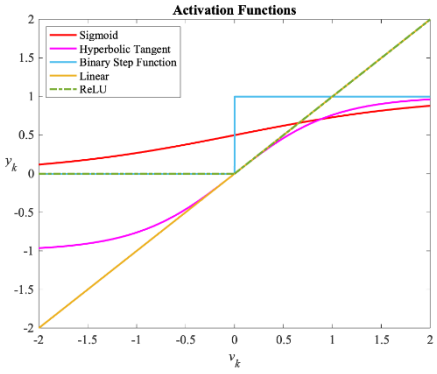
The goal of the activation function is to introduce nonlinearities in the output signal. This is useful in recognizing nonlinear relationships between inputs and corresponding outputs [35]. In fact, if we suppose that just linear activation functions are used, the output of the NN will be just a linear combination of the input vector, no matter the number of neurons and hidden layers composing the network. NN with just linear activation functions acts as a linear regression model with limited performances in extracting complex information from data.

Activation functions play an important role also in the learning process, which is based on an iterative weights' update, as it will be described later. In particular, the learning process follows a gradient descent method called back-propagation, which is based on the derivative of the activation function to compute the update for the weights of the network. Therefore, the activation functions play a determinant role in the NN, defining the output of each neuron and in turn controlling the behaviour of the overall network.

The most common activation functions are reported in Fig. 11 (a), along with their derivatives (Fig. 11 (b)) and they are listed below:

- Binary step function
- Linear activation function
- Sigmoid activation function
- Hyperbolic tangent activation function
- SoftMax activation function
- Rectified Linear Unit (ReLU) activation function

a.



b.

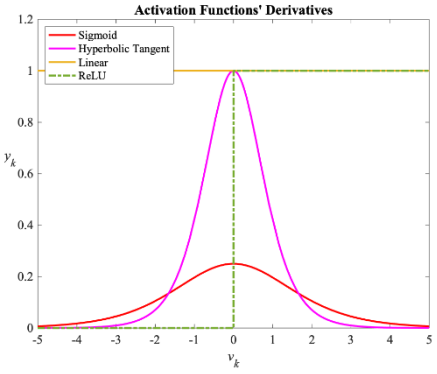


Fig. 11: a. Summary of neuronal activation functions; b. Summary of neuronal activation functions' derivatives.

Binary step function. This is the simplest threshold-based activation function, mathematically defined by the formula:

$$y_k = \begin{cases} 1 & if\ v_k \geq 0 \\ 0 & if\ v_k < 0 \end{cases} \tag{2}$$

and reported in Fig. 11 (a) in blue.

Thus, it produces a binary output, making it suitable as activation function for output neurons for classification, it cannot be used in hidden neurons because the activation function must be differentiable to allow the back-propagation optimization to work properly. Having 0 derivative, the gradient descent is not able to make progress in updating the weights.

Linear activation function. The neuron output is proportional to the input. One example is the identity function, where:

$$y_k = v_k \tag{3}$$

showed in Fig. 11 (a) in yellow.

In this case the NN is just outputting a linear function of the input. Using a linear activation function, the output of the network is a linear combination of the inputs independently from the number of layers.

The derivative of a linear function is constant (Fig. 11 (b), yellow). Therefore, the gradient is a constant value independent from the inputs, implying that the backpropagation

optimization updates the network parameters always by the same quantity for a given layer, so the NN would not improve its error and would not be able to understand how to change its parameters to provide a better prediction.

Sigmoid activation function. It is the most common type of activation in feed forward NNs and in the output layer of deep NNs. It is described mathematically by the formula:

$$y_k = \frac{1}{1 + e^{-v_k}} \tag{4}$$

It is a S-shaped nonlinear function, as reported in Fig. 11 (a) in red, bounded between 0 and 1, it guarantees training stability over linear activation function that instead produces a non-bounded output.
Advantageously, this function has a derivative related to the value of inputs (Fig. 11 (b), red), allowing back-propagation.
However, it also suffers from some limitations which involve the vanishing gradient problem, because for very low or very high input values, the derivative is practically flat, and the network does not learn or is too slow in reaching an accurate prediction. As further limitation, it gives non-zero centred output, causing the gradient updates to propagate in different directions.

Hyperbolic tangent (tanh) activation function. It is a smoother zero-centred function bounded between -1 and 1, whose mathematical formulation is given by:

$$y_k = \frac{2}{1 + e^{-2v_k}} - 1 \tag{5}$$

and reported in Fig. 11 (a) in purple.
The tanh function was proposed to overcome some limitations of the sigmoid: its gradient is steeper, as shown in Fig. 11 (b) in purple, allowing faster convergence. The zero centred output also helps the back-propagation process. However, the vanishing gradient problem remains.

SoftMax activation function. It is used in multi-class models to return the probability distribution for each class. The SoftMax function is a generalization of the logistic

function and normalizes the output for each class between 0 and 1, with the sum of the probabilities for each class being equal to 1. The SoftMax is computed using the relationship:

$$y_k = \frac{e^{x_k}}{\sum_{j=1}^{m} e^{x_j}} \quad for\ k\ =\ 1, \dots, m \tag{6}$$

Typically, it appears at the output layers of the deep learning architectures. The main difference between the Sigmoid and SoftMax is that the former is used in binary classification while the latter is used for multivariate classification tasks.

Rectified Linear Unit (ReLU) Activation Function. It performs a threshold operation on input values, setting to zero the negative ones and leaving them identical if positive, thus the mathematical formulation is:

$$y_k = max(0, v_k) \tag{7}$$

The ReLU activation function is reported in Fig. 11 (a) in green. It is the most successful and widely used activation function for deep learning applications, given the better performance and generalization compared to the previously cited activation functions [36]. In fact, ReLU solves the vanishing gradient problem seen with Sigmoid and Tanh, because it has gradient equal to 1 for positive inputs (Fig. 11 (b), green), allowing the network to converge more quickly.
Another advantageous property is the fact that having a gradient equal to zero for negative values introduces sparsity, keeping only the useful links and therefore the network becomes less dense, decreasing computational costs.

Among the activation functions presented, we chose case-by-case the most suitable one depending on the properties of the problem we aim to solve, privileging the ones that are proved to lead to faster convergence and maximal performance gained by the NN (for example, ReLU for neurons in hidden layers of Convolutional NN).

Network's Architecture

As already mentioned, the architecture of a NN indicates its overall structure, i.e. how many units it has and how these units are connected to each other. NNs are organized into groups of units called layers, as shown in Fig. 12.

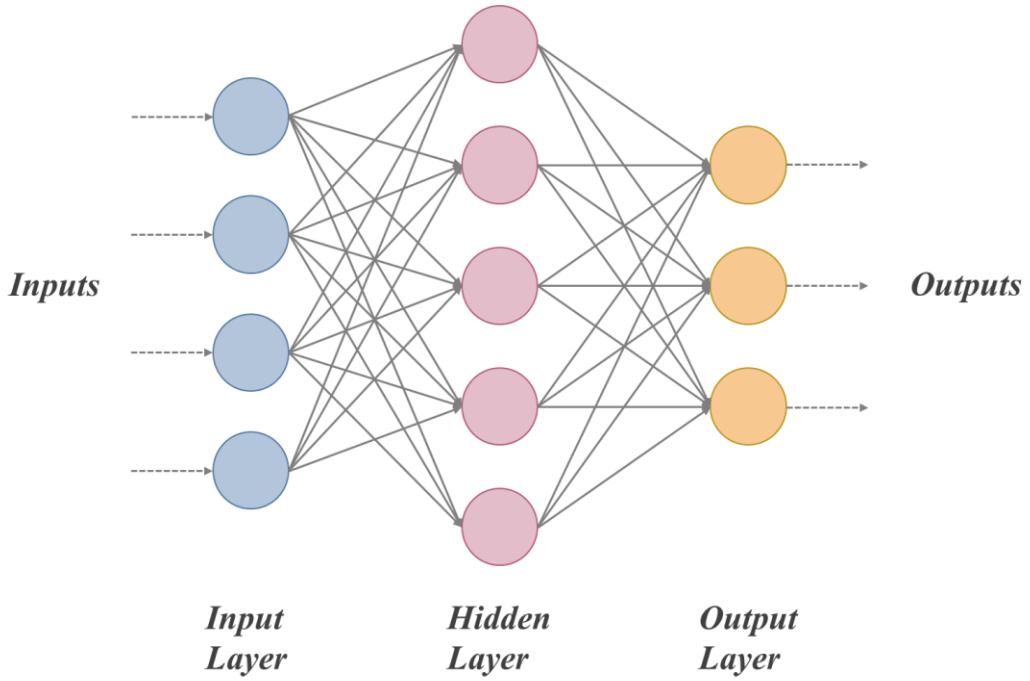


Fig. 12: Generic Neural Network Architecture.

These layers are arranged in a chain structure and with each layer being a function of the layer that precedes it. In general, a NN is constituted of three types of layers (they are a group of neurons sharing the same kind of activation function and position in the hierarchical organization of the network):

i. Input layer: it represents the input pattern to be provided to the network (represented by the dotted arrows in Fig. 12), with no computation performed at this level. The dimensionality is equal to the number of inputs (for example, in Fig. 12 the dimensionality of the input layer is 4).

ii. Hidden layer(s): they are the intermediate layers between input and output, they are not exposed to the outer world but just part of the abstraction provided by the NN, which in turn can be seen as a black box. They are responsible for all the mathematical computation performed by the NN, extracting features from the inputs and transferring the result to the output. In Fig. 12, the

architecture is composed by a single hidden layer with dimensionality equal to 5.

iii. Output layer: it produces the final result, bringing up the information learnt by the network to the outer world (dotted arrows on the right).

Input and output layers are univocally defined, while the NN architecture can be configured with one or more hidden layers. The larger the number of hidden layers, the longer the NN will take to produce the desired output, but on the other side, the higher the complexity of problems the NN can solve. In fact, the more the input data are nonlinearly separable, the more hidden layers will be needed to perform the task.
NNs with a large number of hidden layers are also called Deep Neural Networks (DNN). In these architectures, each hidden layer learns how to extract a distinct set of features, and the deeper the layer, the more complex the features that layer recognizes, because it aggregates and recombines features from the previous layer. This is known as *feature hierarchy* and makes DNNs capable of handling very large, high-dimensional data, representing them more accurately and concisely (i.e. with fewer units and weights). Many real-world problems, such as image recognition tasks, are efficiently solved with deep architectures [37].

2.2.2 Feed Forward and Convolutional Neural Networks

The way these layers are connected to each other defines the topology of the network. We concentrate on feed-forward networks (FFNN), in which information flows from the input to the output, without any cycles or feedback connections (in that case the NN would be called Recurrent NN). These networks are generally function approximators.
A specialized kind of FFNN, particularly useful for image recognition and computer vision problems (up to the limit that they can beat human recognition power as well [38]), is the Convolutional Neural Network (CNN).
A CNN is a deep FFNN where hidden neurons are connected only to a subset of neurons of the previous layer, resembling the connectivity pattern of neurons in the visual cortex [39]. Vision entails various processing steps that progressively elaborate the visual signal. Here, extraction of features is performed through neurons that respond only to a restricted region of the visual field, known as receptive field.

Accordingly, CNNs are able to extract spatial features from the image seen in input thanks to this sparse connectivity, and this extraction of feature is performed hierarchically (the more going in depth in the network, the more complex the feature learnt).

The structure of a CNN (Fig. 13) is composed by the following sequence of layers:

i. *Convolutional layers* that apply filters to the input image decomposing it into features (the process is called *feature learning*);
ii. *Batch Normalization layer* (not shown in figure) that normalizes features coming from the preceding layer by recentring and rescaling their values;
iii. *Pooling layer* that down-samples those features (without loss of information) to decrease computational costs;
iv. *Fully connected layers* at the end of the network that take the output of the last convolution/pooling block and classify the content of the image according to nonlinear combinations of those high-level features extracted by the preceding blocks.

Of them, the first three constitute the convolutional block, while the last one forms the dense bock.

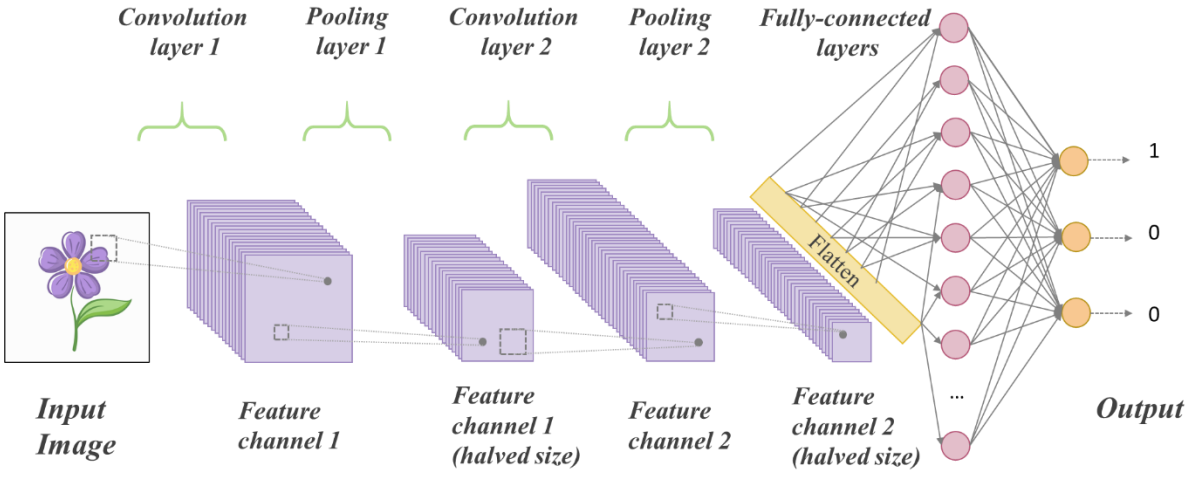


Fig. 13: General Convolutional Neural Network architecture (only two 'Conv/Pool' blocks).

A more detailed description is offered below.

i. Convolutional Layer. It consists of a set of learnable filters, or kernels, smaller respect to the spatial dimensions of the image, which perform the convolution operation. The kernel is shifted along both spatial dimensions, with a given stride, each time performing the convolution, i.e. a matrix multiplication between its values and the portion of image superimposed, until the entire image is covered.
The result of this operation is an output called feature map, expressing the response of that filter at every spatial position. To ensure that dimensionality of the output remains the same as the input, a proper padding is applied.
Each convolutional layer has a certain number of filters, each producing an independent two-dimensional feature map. The output volume is produced by stacking these feature maps along a third dimension (the hypothetical channel dimension for the image). This is fed to successive layers.

Main properties of this layer are therefore:

- Local processing applied to the input image: neurons belonging to convolutional layers receive inputs from a restricted subarea of the image (or feature map if the convolutional layer is more in depth). This strategy reduces the number of connections and, consequently, the computation efforts.
- Shared weights: the convolutional layer can be thought as if numerous copies of the same neuron are created, since they share the same number and values of weights, which are the filter parameters. This allows to maintain the effective number of parameters to be learnt relatively small, thus simplifying and speeding-up the process of learning.

ii. Batch Normalization Layer. Batch Normalization (BN) is a technique that standardizes the inputs to a layer, making this step as part of the model architecture, and performing it for each mini batch of input data. This has the advantage of stabilizing the learning process and reducing the number of iterative steps required to train the network.
Training DNN is challenging because the input distribution of each layer is different during training with every mini batch of data given in input, as the parameters of the previous layers change. This slows the training down, requiring lower learning rates and a more careful parameter initialization. This phenomenon is known as internal covariate shift. BN reduces the range in which hidden unit value shifts around [40].

BN thus allows to use higher learning rates and to be less careful about parameters' initialization. Overfitting is reduced with BN, as it has a slight regularization effect.
To increase stability of the NN, BN whiten each feature layer independently by subtracting to each neuron input $x^{(k)}$ the batch mean and dividing by the batch standard deviation:

$$\hat{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}} \tag{8}$$

However, this is not sufficient, because simply normalizing each input of a layer may change what the layer represents. Furthermore, weights in the next layer become no longer optimal. To solve this issue, the transformation inserted in the network should represent an identity transform.
We introduce two trainable parameters for each layer that provide a linear transformation that scales and shift the normalized output $\hat{x}^{(k)}$ to a mean $\beta^{(k)}$ and standard deviation $\gamma^{(k)}$:

$$y^{(k)} = \gamma^{(k)} \cdot \hat{x}^{(k)} + \beta^{(k)} \tag{9}$$

These parameters are learnt during the training as model parameters, restoring the representation power of the network.

iii. Max Pooling Layer. A pooling layer is a sort of down sampling aimed to reduce the resolution of the feature maps without loss of information. This in turn reduces the number of parameters and overall computation complexity in the network; moreover, it allows a better control for overfitting problems [41].
The most common pooling layer (which better performs respect to other) is Max Pooling, which gives in output the maximum value from the portion of feature map covered with the kernel. It requires two hyper-parameters to be set a priori: size of the kernel, and stride. Their typical values are: filter size = 2x2; stride = 2.
In this way, a stack of feature maps with dimensions H x W x N, where N indicates the number of feature maps, is down sampled to H/2 x W/2 x N.

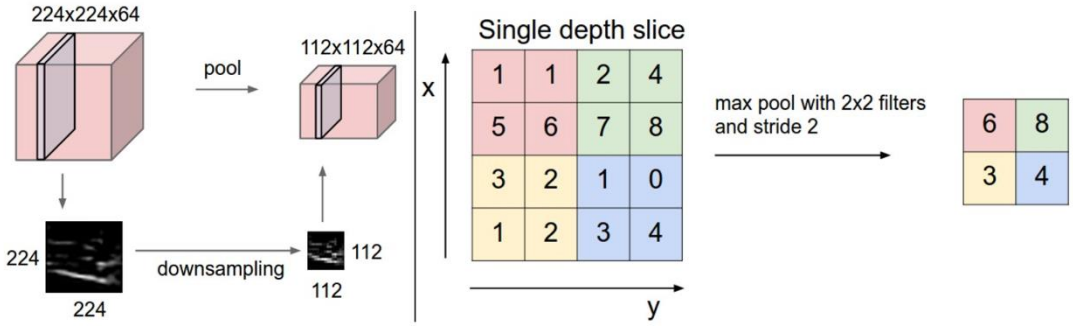


Fig. 14: Pooling layer down samples the volume spatially, independently in each depth slice of the input volume. Left: input volume of size [224x224x64] is pooled with filter size 2, stride 2 into output volume of size [112x112x64]. Right: max pooling with a stride of 2, i.e. each max is taken over 4 numbers (little 2x2 square). [42]

iv. Fully Connected Layer. It composes the fully connected NN structure that drives the final classification decision. Its input comes from the last pooling block, whose output is flattened into a one-dimensional vector. Neurons in a fully connected layer have full connections to all neurons composing the previous layer and perform high-level reasoning. This is a way to learn non-linear functions of the high-level features represented by the output of the last convolutional layer. Training this part of the network makes the model able to distinguish between dominating features in images and classify their content accordingly with Softmax activation function. Thus, the fully connected output layer has a number of neurons equal to the number of classes to predict, each outputting the associated probability for the input of belonging to that class.

Once the network architecture is chosen, the NN must learn to accomplish the desired task. This is done in the training phase, where the NN simply looks to a set of training examples in an iterative procedure trying to predict the correct label for each of them. After each step of the training phase, the performances reached by the NN must be evaluated, consisting in a validation phase.

2.2.3 Training Phase

Training NNs is an iterative process, which starts initializing the parameters of the NN randomly, and culminates in the optimal values of the weights such as the NN is able to perform the desired task.

NN learns through the minimization of an error function, also called loss or cost function. This function is computed as the difference between the actual NN output and the expected result and it is parameterized in the weights.

The general expression of the cost function $J(\theta)$ represents the sum of the error, i.e. difference between predicted $\hat{y}^{(i)}$ and expected $y^{(i)}$ value, over the total number of training samples m:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost\left(\hat{y}^{(i)}, y^{(i)}\right) \tag{10}$$

The process of $J(\theta)$ minimization is performed with the gradient descent optimization algorithm, in which the update for the weights is proportional to the negative of the gradient of the cost function computed in respect to the weights. The general formula for the weights' update is therefore:

$$\Delta \boldsymbol{w} = -\gamma * \nabla J(\boldsymbol{w}) \tag{11}$$

The step size at each iteration is determined by a tuning parameter of the optimization algorithm that is called learning rate, γ. This value has to be set-up appropriately. If it is too high, it makes the loss not converging to the minimum, while if it is too, it makes the training process unacceptably slow. Typically, the value of the γ is chosen manually.

Since the neural activation functions are likely to be nonlinear (see pros and cons in section 2.2.1), $J(\theta)$ is a non-convex function, so, during minimization with gradient descent, the function may get stuck in a local minimum. To prevent this, a basic impulse to the loss function is provided (called momentum) in a given direction, thus helping the loss not to get stuck in a local minimum. This method is called stochastic gradient descent.

Backpropagation is a method to compute the partial derivatives of the cost function with respect to any weight in the network. To perform this, it uses a technique called chain rule [43]; it computes the gradient one layer at a time and propagating backward the error distributing it across the contributing weights. Backpropagation generalizes the gradient computation in the delta rule.

2.2.4 Validation Phase

Another concept that accompanies training is referred to the evaluation of the performances reached by the network during iterations. This aims to assess how good the model has become in correctly accomplishing the task.
To do this, we have first to define a new set of data, which is independent from the training samples used to adjust the value of weights. This new set of data is called validation dataset, and it is defined as the dataset that provides unbiased evaluations of the model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. In other terms, the model should become able to generalize its capabilities of performing a given task also on data which are not used to "learn" how to accomplish that given task. To reach generalization capabilities, the validation set results are used to fine-tune the model hyperparameters.
Validation dataset goes together with the definition of a validation metric, which depends on the task.

2.3 Software Development

2.3.1 Software Framework

We implemented our CNN in Python [44], using two open-source machine learning libraries: TensorFlow and Keras. They are very suitable for numerical computing as DL requires and allow for a fast and easy model development and evaluation.
DL models were trained and evaluated in the online browser-based platform *Google Colab*, a free cloud service in which the development deep learning applications is accelerated using a free Tesla K80 GPU.

Two cases studies are proposed in this thesis, both based on the use of CNNs. The first one involves the use of a CNN for classification purposes to detect apoptosis in small windows of the image using a sliding window approach. The second case study uses a

particular structure of CNN, called UNet, to perform semantic segmentation distinguishing between three classes: intact nuclei, apoptotic nuclei and background.

2.3.2 Case Study 1: Apoptosis Detection with VGG-16 CNN

For this application, an object-detection-based strategy is used.
Object detection is a fundamental recognition problem in computer vision, which aims to find whether there are represented instances of certain categories in a given image or not. If present, it also provides precise localization of each object instance (for example through a bounding box) [45]. Since DL has emerged as powerful solution for learning complex representations automatically from data, these techniques have provided the major improvement in this field. [46]

Object detection combines two types of problems closely related: image classification and object localization.

- Image classification involves recognizing the content inside an image, assessing the presence of objects of interest, and assigning one or more class labels (depending on how many and which objects are represented in the image);
- Object localization instead refers to identify the location of instances inside the image, thus making the overall task more challenging than simple classification.

DNNs generally achieve better performances than conventional cell detection approaches. These involve firstly the extraction of one or multiple hand-crafted features with classical image processing techniques, and after that a classifier working on these feature vectors to identify regions containing target cells, but they have shown some limitations [47]. DNNs instead provide the advantage of evolving from fixed types of extracted features toward problem-specific features which are learnt automatically from training data. This is why they were preferred in this project.

The idea is that DNN detectors are trained as classifiers in the image pixel space, either as a pixel labelling or as a region proposal network. Thus, these methods predict the coordinates (x, y) of those cells directly on the image.

We realized a so-called sliding window method, which reduces the problem of object detection to image classification in sliding window fashion. The NN is trained to classify the content of a fixed-size rectangular window, that gives in output the probability for the window to belong to a specific class; then the NN analyses all possible regions by scanning the image and classifying each window independently (Fig. 15 (a)).

Depending on the application, we can choose to repeat this procedure different times, scanning the image several times with a different window size, depending if the detecting objects have various dimensions inside the image. The more the types of windows, the longer the detection will take.

Once all windows have been exploited, we obtain an output like the one reported in Fig. 15 (b), with multiple overlapped windows predicting the presence of the object of interest. Among the overlapping windows, the one with the highest detection probability was kept, and the others discarded, giving the final detection (Fig. 15 (c)).

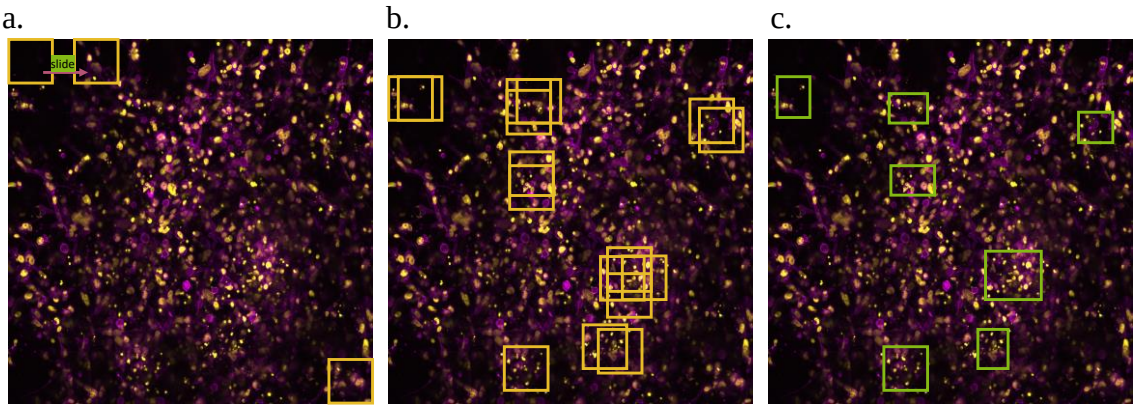


Fig. 15: Example of sliding window principle: (a) fixed size squared window scanning the entire image, each time making the prediction, (b) positive predictions across the entire image, (c) non-maximal suppression result.

We chose to employ the sliding window technique because it is a simple and effective method that works properly in apoptosis detection for two main reasons: first, the classifier is faster to be train compared to other architectures, which instead involve a very huge number of parameters (e.g. YOLO architecture has about 272 million parameters, compared to the 50 million of our architecture, which can be further reduced to 9 million by reducing the number of hidden units, as it will be described later). This implies the need for a lower number of images during training to achieve good performances and avoid overfitting. Second, this strategy allows to localize, drawing a

bounding box around the object of interest without needing that the coordinates of such boxes are labelled for training examples.
Finally, the “bounding-box” window dimension was chosen on the basis of biological and network-based considerations.

VGG-16 CNN Architecture

The DNN architecture chosen as window-classifier is a customized version of the VGG-16, a CNN model proposed by K. Simonyan and A. Zisserman in [1].
We chose this type of network for two reasons: first, similar networks already proved successful results in similar tasks [48]; second, VGG16 has been successfully applied along with the sliding window paradigm for similar applications [49].
The peculiarity of this network, behind its success over the past proposed architectures, is the attention on the convolutional depth, which was proved to enormously affect the accuracy of the prediction in image recognition. This is accompanied with the choice of very small convolutional filters, with a kernel size of 3x3 in each layer. These two adjustments showed a significant improvement in performances, leading to final architectures with depths of 16 or 19 layers (thus, VGG-16 and VGG-19 are defined).

The network architecture is shown in Fig. 16:

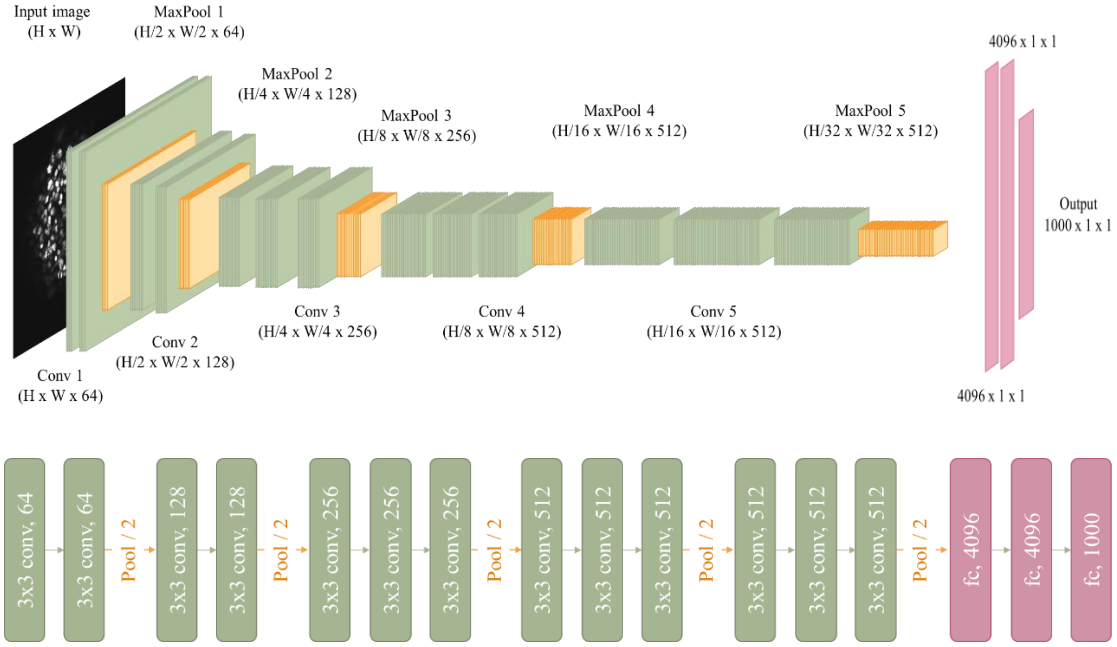


Fig. 16: VGG16 architecture. Each green block represents a stack of Convolutional Layer + Batch Normalization + ReLU activation. Each orange block is a stack of MaxPool layers (equal in number to the number of feature maps from the previous convolutional block). Each pink layer represents one fully connected layer.

It is composed by two parts: the convolutional part which works as feature extractor, and the fully connected layer part which works as classifier.

Convolutional part

The fixed-size input image (single-channel square image, whose size is given by the inspected window: 64x64 and 128x128 window pixel-sizes are inspected) is passed through a stack of convolutional-pooling blocks, as shown in Fig. 24. The first two blocks are composed by two convolutional layers followed by MaxPooling; the last three are instead made from three convolutional layers and one MaxPooling.

We customized these blocks by adding BN after each convolutional layer, but before their activation function (ReLU), as suggested in [40].

From a block to another, MaxPool layer halves the resolution. The number of MaxPooling layers (i.e. the number of blocks) defines the resolution decrease factor: with five of these layers, the resolution is decreased by a factor $2^5 = 32$. Thus, the input image size must be multiple of 32 to match the network architecture.

The filters have small 3x3 receptive field, which is the smallest size to capture the notion of left/right, up/down, and center. The convolution stride is fixed to 1 pixel; spatial padding of each convolutional layer is such that the spatial resolution is preserved after convolution, i.e. the padding is 1 pixel for a 3x3-kernel size convolutional layer.
Stacking two or three consecutive convolutional layers with receptive filed of 3x3 provides an effective receptive field of 5x5 and 7x7, respectively. Thus, the number of parameters is decreased. In fact, assuming that both the input and the output of a 3x3 convolution stack has C channels, the stack is parametrized by:

$$n_{weigths} = n_{Conv-stacks} * (size_{filter}^2 * n_{Channels}^2) \quad (12)$$

Thus, the stack of two convolutional layers with filters 3x3 requires $2 * (3^2 * C^2) = 18\,C^2$ weights compared to $1 * (5^2 * C^2) = 25\,C^2$ weights for a single 5x5 convolutional layer, i.e. 39% more.

The stack of three convolutional layers with filters 3x3 requires $3 * (3^2 * C^2) = 27\,C^2$ weights compared to $1 * (7^2 * C^2) = 49\,C^2$ weights for a single 7x7 convolutional layer, i.e. 81% more.

This results in the observation that increasing the depth, together with reducing filter size, significantly improves accuracy over the prediction, as a regularization effect is imposed, leading to better performances.

The number of channels (i.e. feature maps) of convolutional layers is set to 64 for the first block in the original architecture, and then it increases by a factor of 2 after each max-pooling layer but it is kept constant from the fourth to the fifth block, reaching 512 feature maps whose dimension is given by (W/32, H/32, 512) for an input image size (W, H). In our customized implementation, the initial number of feature maps in the first convolutional layer is set as hyperparameter to be tuned, starting from 64 as the original work and then reducing it up to the limit where performances start to decrease.

Spatial down-sampling is carried out by five MaxPooling layers at the end of each block. MaxPooling is performed over a 2x2 pixel window, with stride 2.

Fully connected part

The stack of convolutional and MaxPooling layers is followed by three fully connected layers: in the original architecture, the two hidden layers have 4096 hidden units each, while the output layer performs the classification and thus contains 1000 neurons (reflecting the number of classes). All hidden layers are equipped with the rectification (ReLU) non-linear activation function, while the final layer was provided with SoftMax activation function. In our case, the same number of fully connected layers was kept, while the number of hidden units was treated as an hyperparameter to be set. Activation function for hidden layers was left to ReLU. The output layer is instead composed by a single neuron, activated with sigmoid function, as the network is required to perform a binary classification. Thus, the output of this neuron represents the probability (ranging between 0 and 1) of the input window to contain apoptotic fragments.

A threshold was applied to discriminate between the two outputs of binary classification. Threshold was empirically set to 0.5. Thus, images whose output is below the threshold, are assigned to class 0 (apoptosis is not present), otherwise class 1 (apoptosis is present), as shown in Fig. 17.

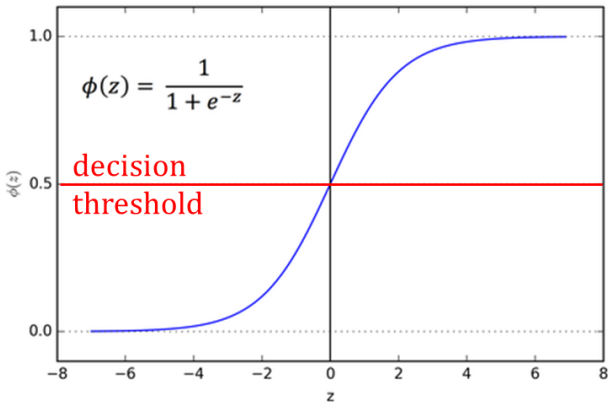


Fig. 17: Sigmoid activation function with decision threshold to discriminate the two classes.

Data Preparation

Data preparation involves two steps: image pre-processing and cropping.

Images are converted in double precision floating-point format and normalized with a linear rescaling function so that all pixel values are within the range (0, 1).

Computing the image histogram, which shows the grayscale value distribution, an elbow-shape can be observed in the lower intensities range, attributable to background heterogeneities. These were removed by pushing the corresponding pixel values where histogram overcomes a given threshold (set empirically) toward zero. After that, the intensity distribution is re-mapped with a linear function again over the range (0, 1).
Finally, gamma correction was performed, a power transformation used to increase brightness and contrast. Accordingly, the pixel value is raised to power γ, empirically set to 0.75.

Fig. 18 and 19 show these three steps applied to two images from the dataset.

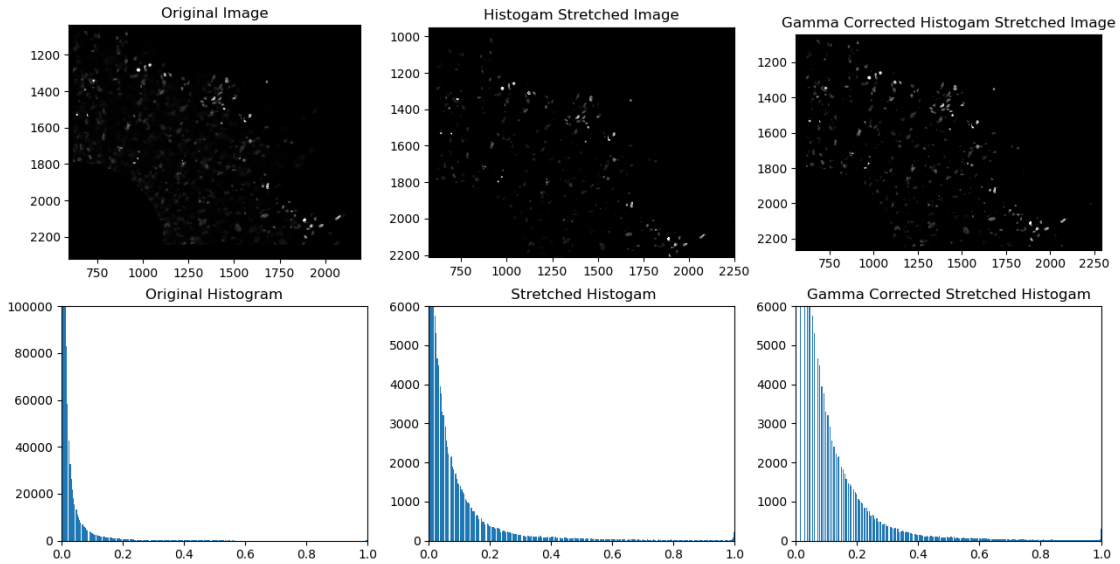


Fig. 18: Example of pre-processing steps applied to one image. The different columns show image and corresponding histogram after 1) Normalization, 2) Histogram Stretching, 3) Gamma Transformation

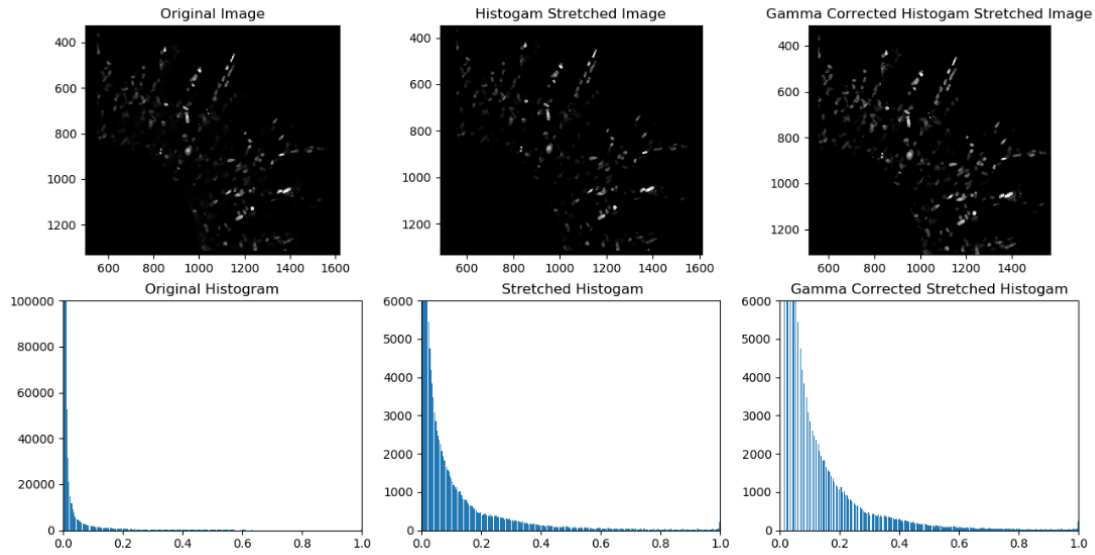


Fig. 19: Example of pre-processing steps applied to one image. The different columns show image and corresponding histogram after 1) Normalization, 2) Histogram Stretching, 3) Gamma Transformation

Second step of data preparation is cropping: to train the classifier, we need to give in input to the NN an image with the same size of the desired window to use for the investigation with sliding window approach in test phase.

Two window sizes are investigated: 64x64 pixels, and 128x128 pixels. These values derive from above cited considerations: first, we chose the minimum input size multiple of 32 pixels which was compatible with the distance between fragments clustered by ACAS software as belonging to the same apoptotic event. Thus, a window of 64x64 pixels was considered. In a second step, to speed up the sliding-window scanning, also window size of 128x128 pixels was investigated.

Training and validation images are obtained cropping each image with a corresponding window centred at annotated coordinates of each nucleus. Then, each window is labelled accordingly: class 0, if the coordinates were referring to an intact nucleus, class 1 if the coordinates were referring to an apoptotic event.

As consequence of this approach, since apoptosis involves a very small number of cells over the entire population, the problem of a highly unbalanced dataset was experienced. Unbalanced data is a huge problem in DL [50]. Possible solutions for handling class imbalance can be divided in two categories: data-level techniques, that act on data by

means of under-sampling the majority class or oversampling the minority class, and algorithm-level techniques that act on the learning process, by adjusting the definition of the cost function [51].

Both these techniques have been investigated in this application.
Acting on data, a particular training strategy was implemented: at each iteration of the training procedure, the NN learns on a training set composed by all the original instances of class 1, and a corresponding number of class 0 randomly picked among the entire dataset without replacement (meaning that the instances of class 0 are seen by the network just one time along the training procedure, while class 1 are seen a number of times equal to the training iteration steps).
Regarding the algorithm-level one, we acted on the definition of the loss function to update the weights, giving more importance to misclassifications occurred for class 1.

Another limitation, deriving from provided annotations, was the lack of sensitivity in apoptosis detection by ACAS. While for those windows that were labelled class1, we can be quite sure that they contain apoptotic fragments (because specificity of the detection was greater than 99%), we cannot draw the same conclusions for class 0: even if the window is probably centred on an intact nucleus, we cannot exclude the presence of an overlapped apoptotic nucleus or fragment within the same window. This is a problem, since those windows can bias the classifier toward incorrect predictions.
To minimize the probability of occurrence, instances of class 0 are taken from untreated spheroid images, where the rate of apoptosis occurrence is much lower than in irradiated samples.
About 1500 windows of class 1 and 50’000 windows of class 0 are obtained, coming from images of spheroid samples from 1 to 20. The remaining part of the dataset was used for test phase in sliding window fashion. These are split between training and validation sets with a proportionality factor of 80% and 20% of the overall dataset.

Training

The NN is trained with about 2400 images per iteration for 200 epochs. The optimization algorithm used to minimize the loss function and adjust the network parameters was

Adam [52]. Since the problem can be reduced to binary classification, the most suitable loss function is the binary cross-entropy, to which we add a proper weight β to manage the problem of class imbalance. The weighted binary cross entropy is thus defined by:

$$WBCE = -\frac{1}{N}\sum_{i=1}^{N} \beta\, y_i \log \hat{y}_i + \frac{1}{\beta}(1 - y_i)\log\,(1 - \hat{y}_i) \tag{13}$$

Where $y_i \; and \; \hat{y}_i$ are the ground-truth label and the predicted output for each image I belonging to the mini-batch (composed by N elements).
The weight β was considered as a parameter to be tuned to deal with the problem of unbalance data.

Validation

The model is continuously evaluated at each step of the training procedure on a validation dataset. Regarding the definition of proper metrics, when building binary classifiers, a common way to summarize the performances is by using a confusion matrix (Fig. 20):

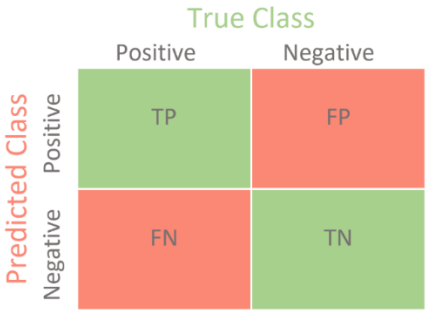


Fig. 20: Confusion Matrix.

The most used evaluation metrics to assess NN performances are derived from the confusion matrix. More specifically, we evaluate the NN performance in terms of Accuracy, Precision and Recall, defined as follows:

$$ACC = \frac{TP + TN}{N} \tag{14}$$

Accuracy measures how many observations, both positive and negative, were correctly classified. The problem with this metric is that it is blind to imbalance problems.

$$PRE = \frac{TP}{TP + FP} \tag{15}$$

Precision measures how many observations predicted as positive are in fact positive.

$$REC = \frac{TP}{TP + FN} \tag{16}$$

Recall (or Sensitivity) measures how many observations out of all positive observations have been classified as positive.

2.3.3 Case Study 2: Semantic Segmentation with UNet

Semantic segmentation is a fundamental aspect of image processing, defined as the process of labelling each pixel of an image with a corresponding class of what is being represented. A NN trained for semantic segmentation receives in input an image and gives in output a 2D mask of the same image's size partitioned into multiple segments or objects, that associates to each pixel a different class. Thus, it acts as a pixel-level image classifier. Through this dense prediction, it produces fine-grained results, because it automatically includes in its output the localization information.

The advantage is that it enables us to decompose an image into meaningful parts and lets us to understand a scene at a more granular level, defining shapes and boundaries of objects in an image.

This approach is more suitable when the extraction of more parameters is required, such as the number of cells inside the image to compute apoptotic ratio. To extract such parameter, semantic segmentation was combined with image post-processing techniques. Several deep learning architectures have obtained outstanding results in cellular segmentation and classification. Among these, the UNet architecture has become a widely used tool for biomedical segmentation and analysis [2], and for this reason this was the NN architecture chosen to achieve the desired task. UNet belongs to the category of

Encoder-Decoder based models [53] which maps data-points from an input domain to an output domain (equal to the input) via a two-stage network:

i. The encoding network, which compresses the input into a feature vector representation capturing the useful information to predict the output.
ii. The decoding network, which aims to predict the output from the feature vector representation enabling precise structure's localization.

UNet Architecture

UNet takes name from its symmetric shape, as shown in Fig. 21. This results from putting together the usual convolutional contracting network (made by a succession of convolutional and pooling layers) with another symmetric convolutional network where pooling is replaced by up-sampling operation, thus increasing the resolution of the output features.

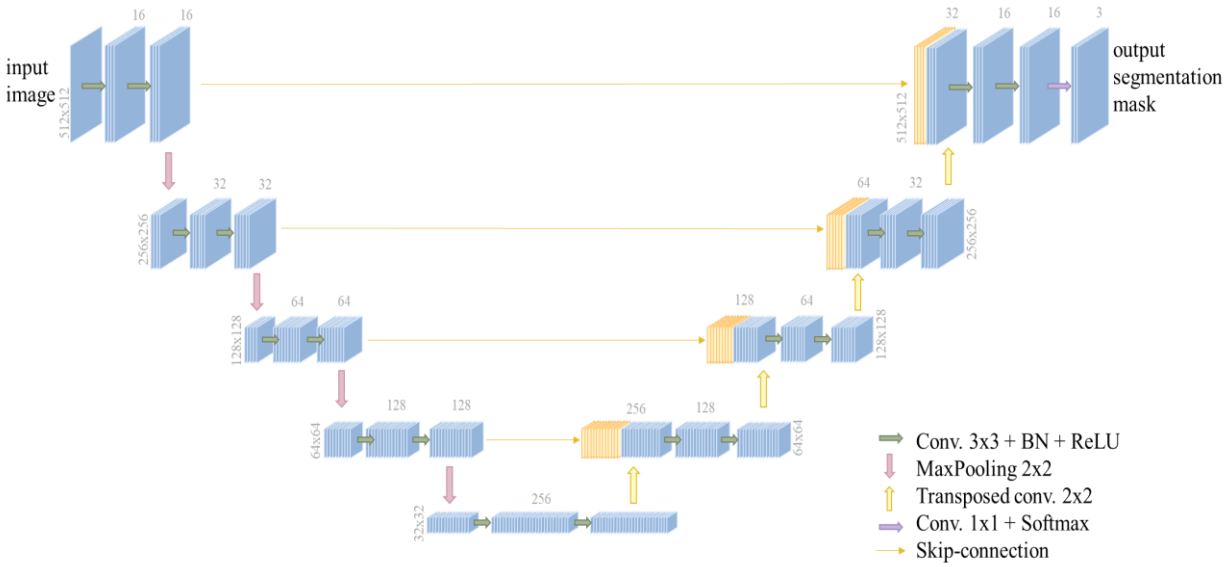


Fig. 21. U-net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. Orange boxes represent copied feature maps. The arrows denote the different operations.

To better localize, the intermediate high-resolution features from the contracting path are combined with the corresponding up-sampled output in the expanding path through skip-connections that implement a concatenation operation. These skip connections allow the network to propagate context information to higher resolution layers.

A successive convolutional layer takes the up-sampled output concatenated with to the high-resolution features and learns to assemble a more precise output based on this information.

There are several advantages in using UNet architecture for semantic segmentation tasks. First, the use of skip connections enables the combination of location information from the down-sampling path with the contextual information in the up-sampling path, useful to predict a good segmentation map. Second, the architecture has no dense layer, so the number of parameters is limited, allowing for a fast training procedure and prediction.

Moreover, the training strategy adopted in the original paper [2] relies on the use of data augmentation to learn from the available annotated images more effectively. The use of massive data augmentation is important in biomedical domain, where the number of annotated samples is usually limited. DNNs are heavily reliant on a large amount of data to avoid overfitting because of the enormous number of parameters to optimize [54]. Data augmentation represents an effective methodology to improve the NN's capability of generalization when the amount of available data is limited.

Encoding / Contracting path

The contracting path follows the typical architecture of a convolutional network already described, except from the fully connected part. Input size is set to 512x512 pixels.

In our implementation, UNet is composed by 4 equal blocks, each one consisting in the repetition of two convolutional layer with filter size equal to 3, unitary stride and adequate padding to maintain the output feature map with the same input size. Convolutional layers have identity activation function, then followed by a BN layer and ReLU activation function. The two convolutional layers belonging to the same block share the same number of feature channels.

Last, a 2x2 max pooling operation with stride 2 for down-sampling follows the convolutional filtering. At each down-sampling, the number of feature channels of the following convolutional layer (in the successive block) is doubled.

In the original architecture, the number of convolutional filters in the first block was set to 64. For computational reasons, mainly due to RAM constrains, we limited the number of feature channels in the first block to 16.

Bottleneck

The bottleneck is between the contracting and expanding paths; it is built from two convolutional layers each followed by BN and ReLU activation function.

Decoding / Expanding path

The decoding part recreates an output mask with the original spatial resolution of input image to enable precise localization of the extracted features.

Analogously to the encoding part, it is composed by 4 blocks, where now MaxPooling is replaced by an up-sampling layer that restores spatial resolution, called Transposed-Convolutional layer. Up-sampling doubles the spatial resolution and halves the feature maps. These feature maps are concatenated through skip-connections with the ones coming from the corresponding encoding block.

These concatenated maps are then given in input to the cascade of two convolutional layers followed by BN and ReLU activation function.

At the final layer, a 1x1 convolution is used to map the final feature maps to the desired number of classes (as in Fig. 21, in our implementation, the number of classes are 3). At the end, SoftMax activation function is used to predict the probability of each pixel to belong to each class.

Transposed Convolutional Layer. The transposed convolutional layer performs an up sampling of the input feature maps [55]. Transposed convolution is a more advantageous technique in respect to predefined interpolation methods (e.g. nearest neighbor, bilinear or bicubic interpolation), because it has learnable parameters, so it learns to up-sample in an optimal way during the training process. The hyperparameters that define this layer are the same as a standard convolutional layer, but they have different meaning. They are chosen accordingly to provide an output whose dimensions are doubled in respect to the input feature map: kernel size was set to k=2, padding equal to p=0 and stride equal to s=2 for both spatial dimensions.

Implementing a transposed convolutional layer consists in a four-steps process, as shown in Fig. 22:

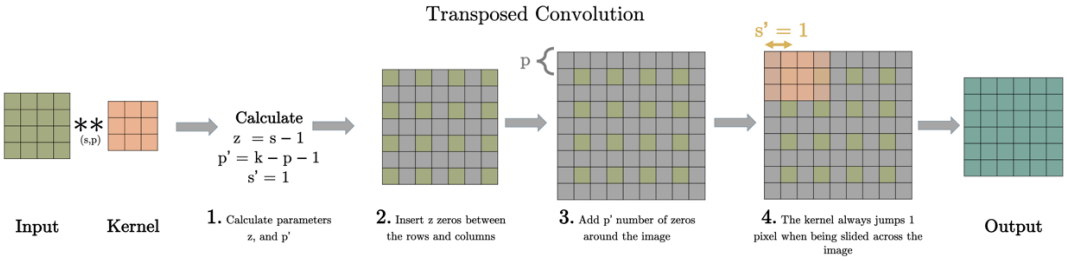


Fig. 22: Transposed Convolution operation.

1. Once decided the values for k, s and p, the values for z and p' are derived
2. z number of zeros are inserted between each rows and columns of the input, increasing the size of the input to

$$(2 * i - 1) \times (2 * i - 1) \tag{17}$$

3. The input image is padded with p' number of zeros
4. A standard convolution is carried on the new image with stride s' = 1

Thus, for a given size of input *(i)*, kernel *(k)*, padding *(p)* and stride *(s)*, the size of the output feature map *(o)* generated is given by:

$$o = (i - 1) * s + k - 2p \tag{18}$$

Concatenation through skip connections. Skip Connections merge feature maps coming from the transposed convolutional layer (after BN and ReLU are applied) with 'intermediate' feature maps from the encoding blocks with correspondent spatial dimensions. This doubles the number of feature maps in input for the following convolutional layer, ensuring maximum information flow.

There are several advantages in using skip-connections in this kind of models: first of all, the fact that we stack the input of deep hidden layers together with low-level feature maps is useful for recovering spatial information lost during down sampling and reconstructing target content at the original spatial resolution with fine-grained details. Thus, skip connections allow to build more compact models with respect to other ones deprived of skip connections on equal performances.

Second, **s**kip connections enable reusability of features from the earlier layers, **s**tabilizing training and convergence. They have an uninterrupted gradient flow from the first layer to the last layer, which tackles the vanishing gradient problem. The loss landscape changes also significantly when introducing skip connections.

Data Preparation

Image pre-processing was the same performed in Case Study 1.

Cropping was performed, selecting ROI in correspondence of the center of the image.

This was preferred to resizing techniques, as they would have involved a considerable degradation of the image input resolution, with the risk of compromising apoptotic fragments identification. In fact, apoptosis consists in very tiny objects, made of very few pixels per fragments. Resizing an image from 2000x2000 pixels (in average) to 512x512 pixels therefore involves an unacceptable decrease in spatial resolution that affects the information content of our target.

Last step of data preparation is manual annotation to create labels for training and validation instances. 90 images have been annotated among the entire dataset.

In order to reduce as much as possible class imbalance problems, images containing the highest number of annotated apoptosis events have been chosen.

When images coming from the second dataset were exploited, data pre-processing was also the same. Instead, cropping was not performed, as the original size was closer to the input size of the network, and magnification factor in these microscopic images was higher than the first dataset, so resizing was considered a reasonable solution. It was performed with spline interpolation of the third order.

As we have already introduced the advantages of using data augmentation when training a NN, this approach was followed after training/validation split (in this case about 70% of instances – 63 images – were employed for training and 30% – 27 images – for validation) to ensure independency between the two datasets.

Data augmentation does not involve geometrical deformation or blurring of the input images that could compromise the information content, but randomly vertical and/or horizontal flipping, together with random rotation with angular resolution of 15° and nearest neighbour interpolation.

From each original image, other 9 were obtained (together with their segmentation maps which underwent the same type transformations). Thus, the overall datasets are composed by 630 images for training set and 270 images for validation set.

Training

Training procedure was carried out for 300 epochs, with batch size equal to 4. Loss was minimized according to ADAM optimizer, with learning rate set to 0.005

Loss function is defined separately for intact and apoptotic masks, and then combined together with proper weight factors set empirically starting from values of class occurrence for apoptotic and intact pixels.

For semantic segmentation, region-based loss functions are well suitable since they aim to minimize the mismatch or maximize the overlap regions between ground truth and predicted segmentation masks. Among these, two types of loss are taken into account:

Dice loss [56] directly optimizes the Dice coefficient which is the most commonly used segmentation evaluation metric.

$$DICE = \frac{1}{N}\sum_{i=1}^{N}\left(1 - \sum_{Class} w_{Class} \cdot \frac{2\sum_p y(p)\,\hat{y}(p)}{\sum_p y(p) + \sum_p \hat{y}(p)}\right) \tag{19}$$

It calculates the similarity between ground-truth mask and the output segmentation mask from the network, separately for each class. Then, a proper weight is added for each of the class to be discriminated (for background pixels, w is set to 0).

Tversky loss sets different weights to false negative (FN) and false positive (FP), which is different from Dice loss using the equal weights for FN and FP.

$$TL = \frac{1}{N}\sum_{i=1}^{N}\left(1 - \sum_{Class} w_{Class} \cdot \frac{\sum_p y(p)\,\hat{y}(p)}{\sum_p y(p)\,\hat{y}(p) + \beta\sum_p (1 - y(p))\hat{y}(p) + (1-\beta)\sum_p y(p)(1-\hat{y}(p))}\right) \tag{20}$$

Tversky loss can be seen as a generalization of Dice loss to address the issue of data imbalance and achieve better trade-off between precision and recall. This is done by adding a weight to false positives and false negatives through β coefficient.
When β = 1/2, It can be solved into Dice loss. For this case, β = 0.3 as in the original paper [57].

Both of them are measurements of overlap, meaning that they measure the discrepancy between predicted mask resulting from the network segmentation and the actual one.

Validation

To assess segmentation performances, a proper metric which is also a measurement of overlap is employed, called Intersection over Union:

$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \tag{21}$$

Fig. 23 illustrates the concepts of intersection and union between two bounding boxes (considering one of them as the ground-truth, the other as the network prediction).
IoU is a good metric for measuring overlap between ground-truth and predicted masks, evaluating the overlap between the two areas.

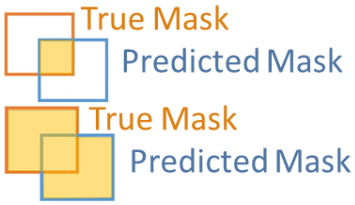


Fig. 23: Concepts of Intersection and Union of predicted and ground-truth mask.

Results section will present separately the outcome of both case studies.

3. Results and Discussion

3.1 Case Study 1: Apoptosis Detection with VGG16 CNN

In order to detect the apoptotic nuclei, a CNN-based binary classifier is trained to discern whether apoptotic fragments are contained or not inside a fixed-size square window, giving in output the probability that apoptosis is present within that window. Once the CNN is trained, to finally quantify the apoptotic events of the overall cell population, the entire image is scanned by the input window to the CNN in sliding window paradigm.
Two window-sizes were investigated: 64x64 pixels and 128x128. The first one corresponds to the maximum extent of apoptotic events in those images used for training (multiple of 32 for NN architectural considerations). The second one is investigated in attempt to reduce the computational time required in SW-test phase.

3.1.1 Results of Training and Test for 64x64 Input Images' Size: Summary of Dataset, Model and Algorithm Hyperparameters.

Dataset

The final dataset used for training and validation is composed by:

- 1468 crops of class apoptosis coming from annotations of treated spheroids, with a preponderant percentage of apoptosis centred in the middle of the window and the remaining part randomly shifted around the centre (not to bias the model in predicting apoptotic class even if fragments do not fall perfectly in the middle of the window during the SW approach);
- 13008 crops of class intact from treated-spheroids population; it was opportunely checked that no annotated apoptosis was falling inside;
- 31831 crops of class intact from untreated-spheroids population.

We could not exclude the presence of some non-annotated apoptotic event inside a such large population of 'intact' crops. However, those samples would weigh less during the learning process since the network recognizes them one time only during the training phase. These data were opportunely split in training and validation sets with respective ratios of 80% and 20%.

Model Hyperparameters

After several trials, starting with the original network architecture, we reduced the number of filters in the first convolutional block to 8, thus decreasing the number of parameters to avoid overfitting, without compromising the network's performance. Since the reduced input size, the last block of 3 convolutional layers and MaxPooling is removed. The other hyperparameters of the convolutional block remained the same as the original architecture.

Concerning the fully connected part, the number of layers remained the same, i.e. 3. The number of hidden neurons starting from 4096 in 1 and 2 was reduced to 256 without losing performance.

Algorithm Hyperparameters

The model is trained for 200 epochs, with mini-batch size equal to 32, i.e. the number of training samples used to estimate the loss before one update of the weights for each iteration. The batch size is an important hyperparameter since it influences the dynamics of the learning algorithm, controlling the accuracy of the error estimation and how quickly the model can learn the task. It was proved that such value leads to good training stability and generalization performances.

The loss function used for binary classification is the weighted binary cross-entropy (Eq. 13). It was preferred to the classical binary cross-entropy because the multiplicative factor β for positive targets gives more importance to their misclassification in loss computation, resulting in a reduced number of false negative detections.

The optimum value for such multiplicative factor was found empirically and it was set to 4.

The loss function was minimized with Adam optimization algorithm [52]. The learning rate, which controls how much the model weights can change during their update in response to the estimated loss, was set to 0.001 at the beginning of the learning process and configured with sigmoidal drop up to the final value of 0.0001 at the last iteration.

Results of the training process are shown in Fig. 24:

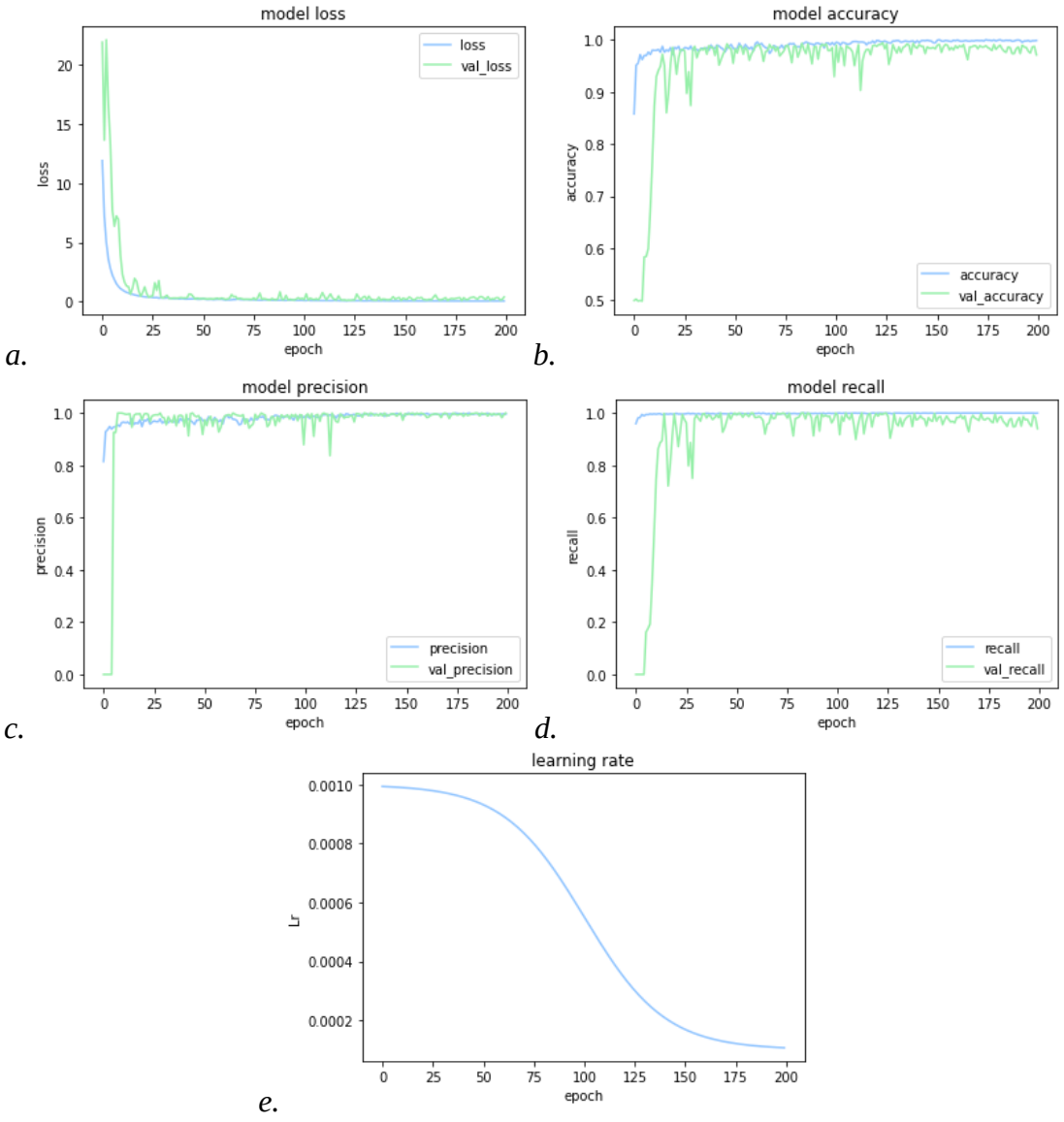


Fig. 24: a. Loss, b. Accuracy, c. Precision, d. Recall and e. Learning Rate trends during training epochs

Model performances are evaluated in terms of accuracy, a classical metric for classification problems which indicate the goodness of the model in making a correct prediction, but since we are dealing with a very unbalanced problem it resulted not sufficient to validate the model, thus precision (quantifying the ability of the model to identify as relevant instances only those that actually were) and recall (quantifying the ability of the model to find all the relevant instances within the dataset) were added.

The minimum value for the validation loss was reached at epoch 179, therefore that model was saved for inference. The results show that satisfactory performances are achieved

both on the training and on the validation sets. Table 1 reports achieved values for loss and performance metrics used for classification in both sets:

	Loss (WBCE)	Accuracy	Precision	Recall
Training	0. 0589	0.9995	0.9990	1.0000
Validation	0.0950	0.9907	0.9872	0.9955

Table 1: Summary of loss's and metrics' values for training and validation sets.

In order to find the optimal value for weight β in WBCE loss (Eq. 13), an empirical procedure was followed. This consists in testing several models, each of them being trained with a different value of β, on a set of entire images where apoptoses were manually labelled (for a total of 12 images) and scanned with the SW technique with stride equal to half-size of the window.

For each of these images, to assess the performance of each model, the F1-score has been calculated, which is more suitable than accuracy when class distributions are uneven. Obtained values are then averaged over the entire test set.

Values for beta which were investigated range between 1 and 33, where 33 corresponds to the mean ratio between intact and apoptotic windows among test images. Results (summarized in Table 2) have shown that a value for β equal to 4 maximises F1-score.

Beta	F1-score
1	0.6165
2	0.6096
4	0.6463
10	0.6429
16	0.5939
19	0.6099
33	0.6232

Table 2: Computed F1-score for several models trained with different values of beta-weight coefficient in loss.

It is interesting to show feature maps from the various convolutional layers of the CNN for a validation sample containing an apoptotic event correctly classified by the model (output > 0.99). Fig. 25 shows that the trained filters capture the content inside the image useful to predict the correct class. This means that the model bases its prediction on those pixels which really define the apoptotic event.

Fig. 26, 27, 28, 29 report instead different examples of apoptosis identification from different images (each of them being one z-shack of a different tumour). In each of them, those windows predicted by the model to contain apoptosis are highlighted: true positive (TP) predictions (i.e. a window predicted to contain apoptosis which correspond an annotated event by ACAS software) are identified in green; false positive (FP) predictions (i.e. window predicted to contain apoptosis which are not detected by ACAS) are identified in red. False negatives (FN) are not reported.

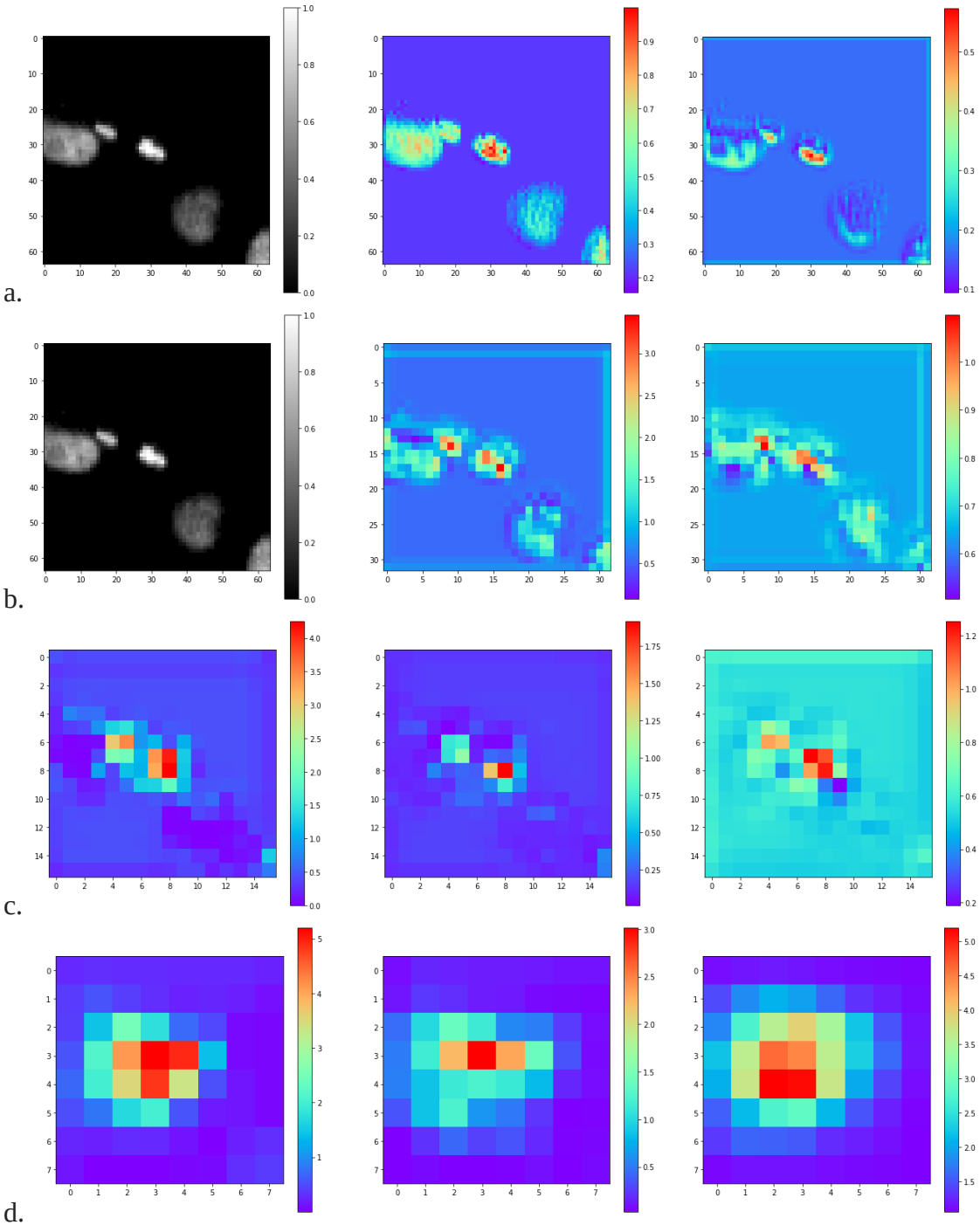


Fig. 25: Feature maps from convolutional layer: a. input image and output from the filters in the first convolutional block; b. input image and output from the second convolutional block; c. output from the third convolutional block; d. output from the fourth convolutional block. Higher values for the output feature maps (coloured in red in figures) indicate that those pixels 'weigh' more for the window classification.

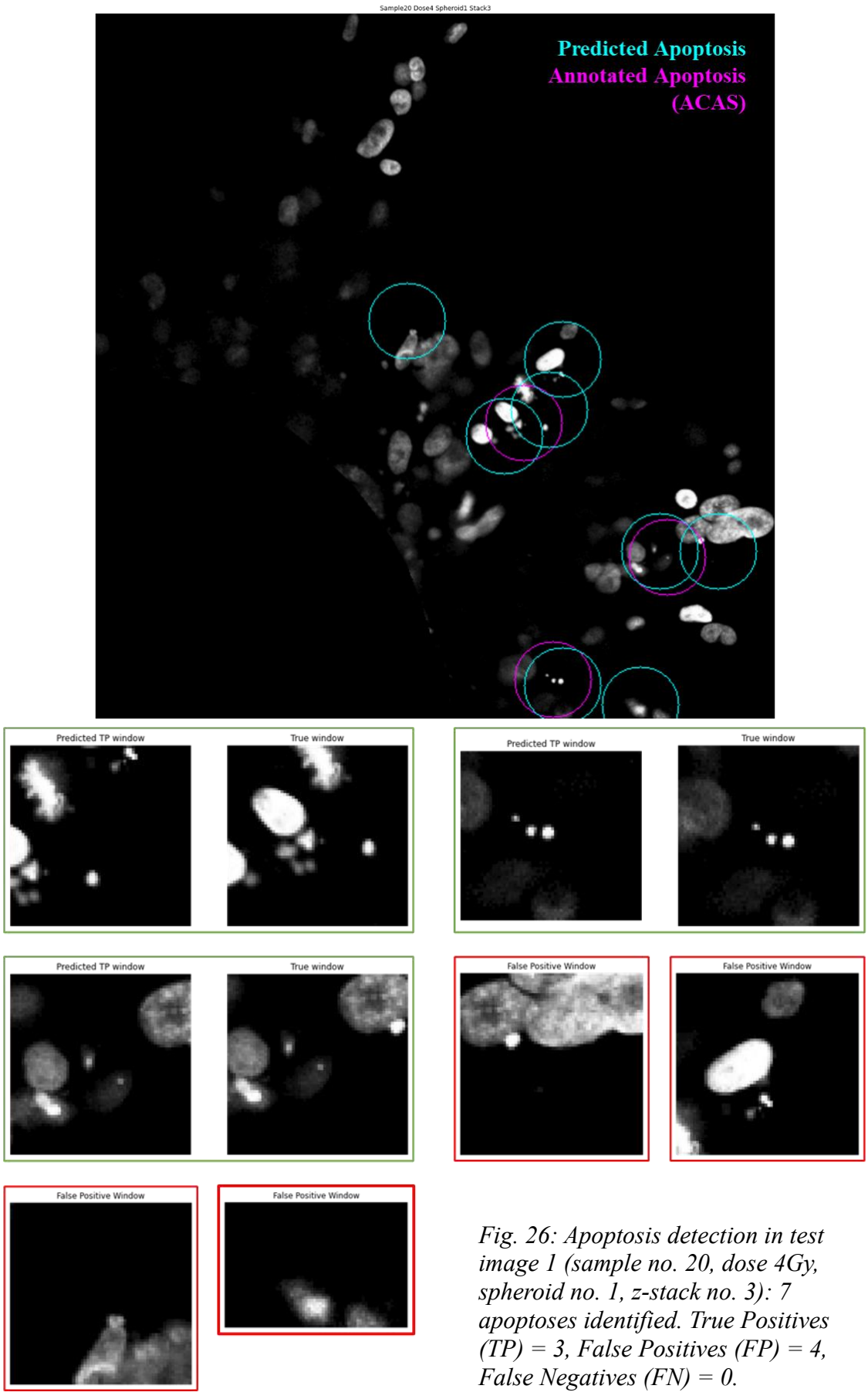


Fig. 26: Apoptosis detection in test image 1 (sample no. 20, dose 4Gy, spheroid no. 1, z-stack no. 3): 7 apoptoses identified. True Positives (TP) = 3, False Positives (FP) = 4, False Negatives (FN) = 0.

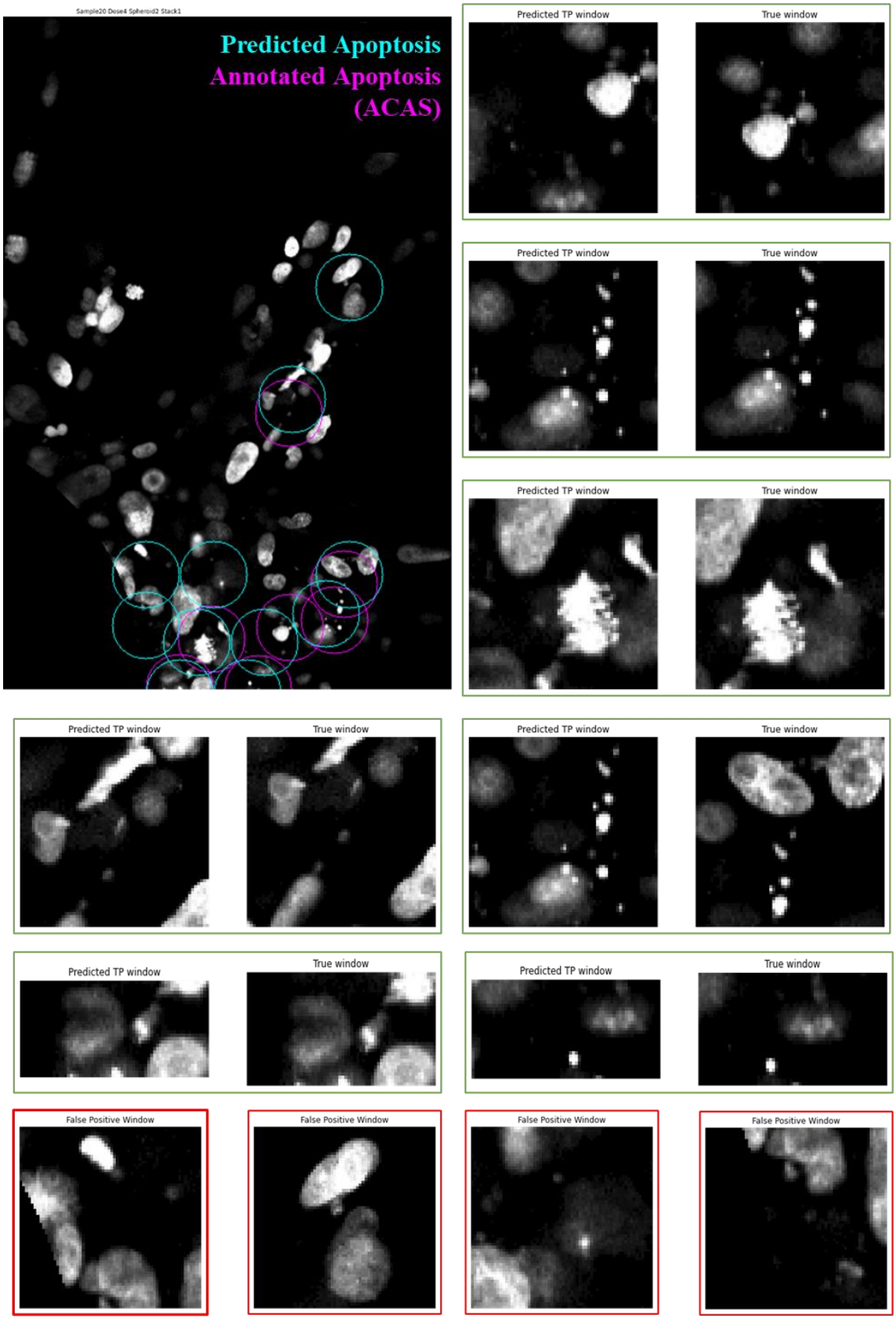


Fig. 27: Apoptosis detection in test image 2 (sample no. 20, dose 4Gy, spheroid no. 2, z-stack no. 1): 11 apoptosis identified. True Positives (TP) = 7, False Positives (FP) = 4, False Negatives (FN) = 0.

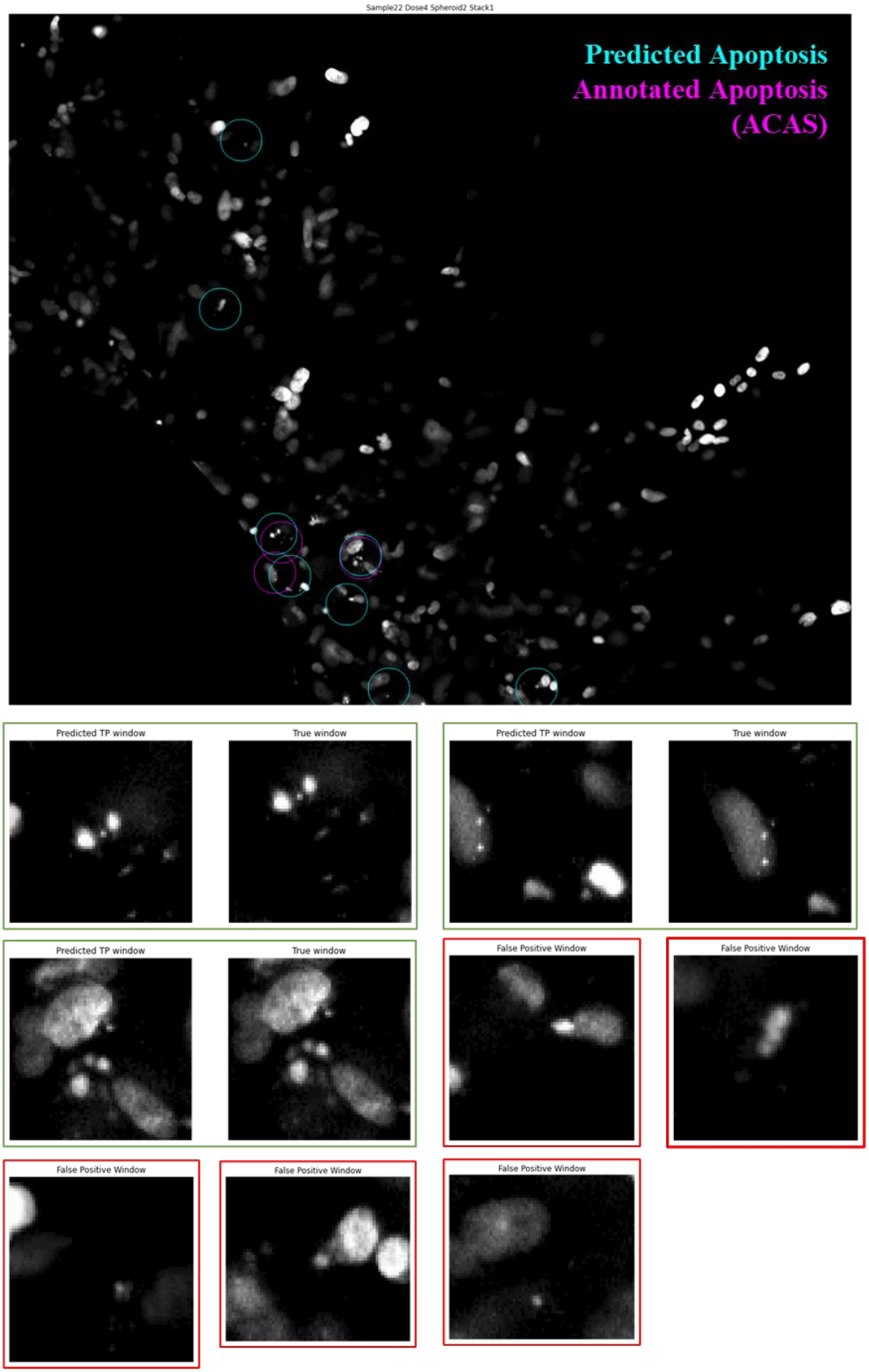
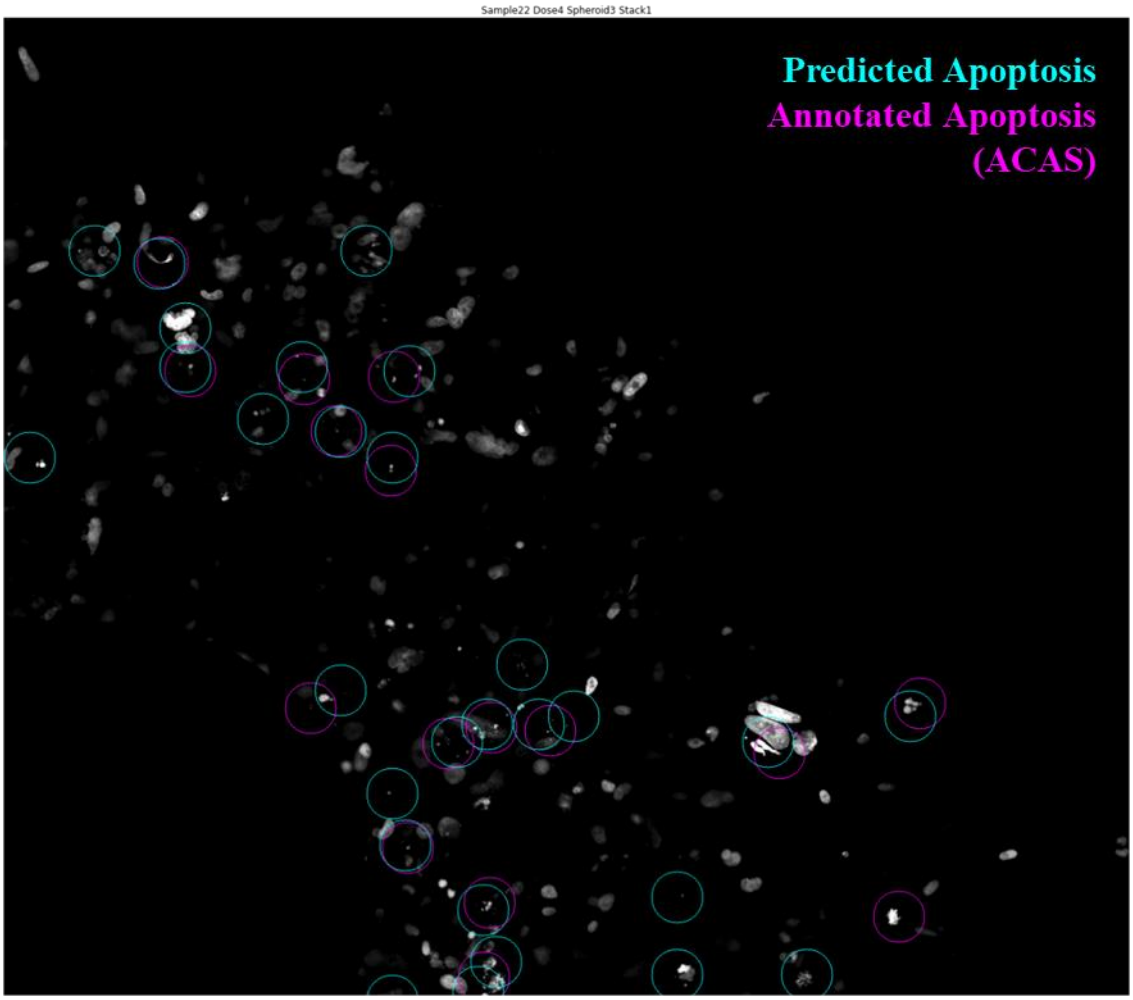
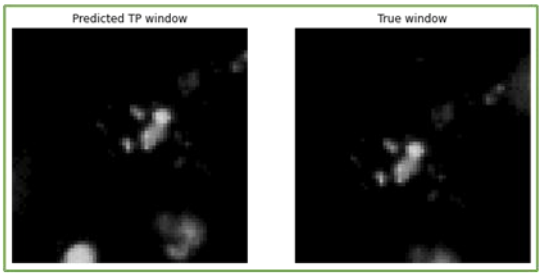
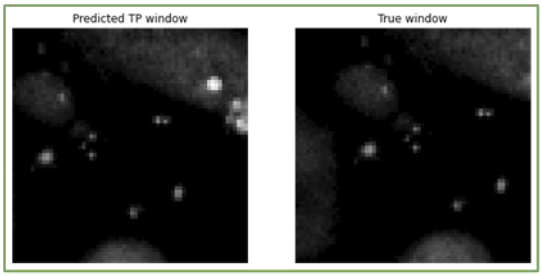
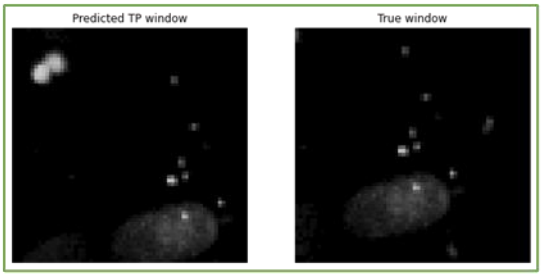


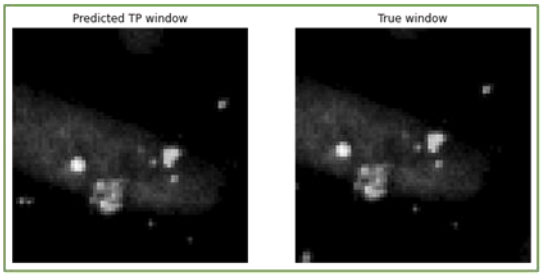
Fig. 28: Apoptosis detection in test image 3 (sample no. 22, dose 4Gy, spheroid no. 2, z-stack no. 1): 8 apoptoses identified. True Positives (TP) = 3, False Positives (FP) = 5, False Negatives (FN) = 0.

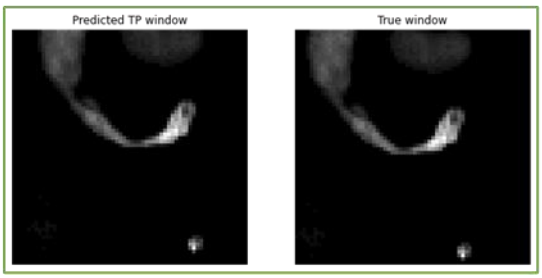

Sample22 Dose4 Spheroid3 Stack1
Predicted Apoptosis
Annotated Apoptosis
(ACAS)

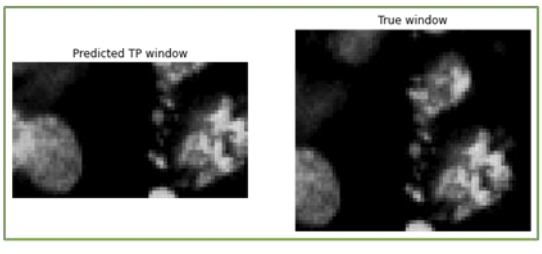

Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window

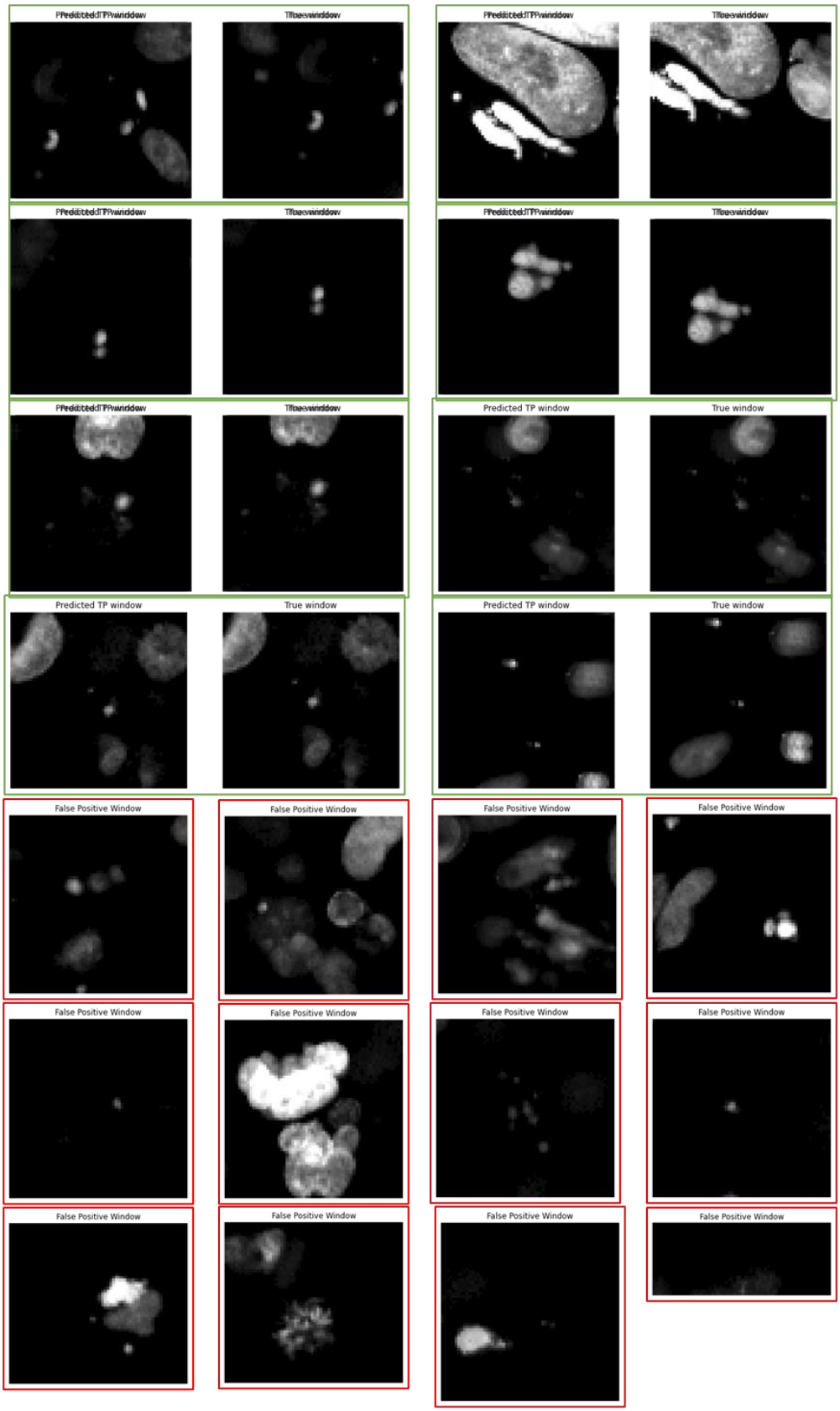


Fig. 29: Apoptosis detection in test image 4 (sample no. 22, dose 4Gy, spheroid no. 3, z-stack no. 1): 26 apoptoses identified. True Positives (TP) = 14, False Positives (FP) = 12, False Negatives (FN) = 2.

Table 3 summarizes the effectiveness of our detection approach in terms of True Positives (TP), False Positives (FP) and False Negatives (FN) for the four test images above presented.

Image	Detected Apoptoses	Annotated Apoptoses	TP	FP	FN
s.20-sph.1-4Gy-stack3	7	3	3	4	0
s.20-sph.2-4Gy-stack1	11	7	7	4	0
s.22-sph.2-4Gy-stack1	8	3	3	5	0
s.22-sph.3-4Gy-stack1	26	16	14	12	2

Table 3. Summary of True Positives (TP), False Positives (FP) and False Negatives (FN) for the four test images compared to ACAS output.

From these results, a consistent false positive number is observable. This is mainly due to the low sensitivity in apoptosis detection by ACAS software.

The outcome of the model went under examination of expert biologists, which indeed confirmed that most of the predictions classified as containing apoptotic events, do actually contain apoptosis, not detected by the software previously used.

Based on their corrections, the actual number of TP and FP is thus adjusted:

Image	Detected Apoptoses	TP	FP	FN
s.20-sph.1-4Gy-stack3	7	6	1	0
s.20-sph.2-4Gy-stack1	11	10	1	0
s.22-sph.2-4Gy-stack1	8	7	1	0
s.22-sph.3-4Gy-stack1	26	26	0	2

Table 4. Summary of True Positives (TP), False Positives (FP) and False Negatives (FN) for the four test images adjusted after biologists' inspection.

To minimize FNs' detection, we chose a weight factor equal to 4 in the loss formula with satisfying results (see Table 4). However, it was observed that a reduction in FN number was possible also by halving the stride at which the window is scanned along the image

during the SW paradigm. For demonstration, we show in Table 5 a comparison between results obtained in 4 test images using the model trained with simple cross-entropy loss function (β=1) scanned over the images with stride = 32 and stride = 16.

	Stride = 32				Stride = 16			
	TP	FP	FN	Time	TP	FP	FN	Time
s.23-sph.1-slice2	**1**	**6**	**4**	**79s**	**5**	**15**	**0**	**304s**
s.23-sph.6-slice1	**2**	**4**	**2**	**49s**	**4**	**13**	**0**	**186s**
s.23-sph.6-slice3	**8**	**14**	**6**	**54s**	**15**	**27**	**0**	**195s**
s.24-sph.1-slice3	**2**	**2**	**3**	**56s**	**4**	**4**	**0**	**215s**

Table 5: Comparison of results of the model trained with binary cross-entropy loss tested over four images with stride = 32 and stride = 16.

FNs number is reduced to zero in all cases. However, computational time required to scan the overall window is quadruplicated (as the number of windows to be investigated). Also, the number of FPs increased. For these reasons, the choice of a model trained with weighted cross-entropy loss (β=4), used in test images with stride equal to 32 was preferred.

The mean computational time required to scan an entire image of about 1800 pixels per size is about 50 seconds. The proposed method can help the work of biologists, avoiding them to scan manually such images to count apoptosis, but resulted a bit slow. For this reason, the effect of increasing the size of the window for predictions was investigated as possible alternative, with the opportunity of reducing by a factor of 4 the computational time required per image.

3.1.2 Results of Training and Test for 128x128 Input Images' Size: Summary of Dataset, Model and Algorithm Hyperparameters

Dataset

The final dataset used for training and validation is composed by:

- 1466 crops class apoptosis coming from annotation of treated spheroids;
- 19963 crops class intact from treated spheroids population;
- 14740 crops class intact from untreated-spheroids population.

As in the previous case, data were opportunely split in training and validation sets with respective ratios of 80% and 20%.

Model Hyperparameters

The model architecture and hyperparameters were the same discussed in the case of 64x64 input. In this case, the last block of 3 convolutional layers and MaxPooling is maintained as in the original architecture. After several trials, starting with the original network architecture, we reduced the number of filters in the first convolutional block to 16, thus decreasing the number of parameters to avoid overfitting, without compromising network performance. Concerning the fully connected part, the number of layers remained the same, fixed to 3. The number of hidden neurons starting from 4096 in the first and second layers was reduced to 1024 without losing performance.

Algorithm Hyperparameters

The model is trained for 200 epochs, with mini-batch size equal to 32. The loss function used for binary classification is the weighted binary cross-entropy with weight β equal to 4. The loss function was minimized with Adam optimization algorithm. The learning rate was set to 0.001 at the beginning of the learning process, it was configured with sigmoidal drop up to the final value of 0.0001 at the last iteration.

Results of the training process are shown in Fig. 30:

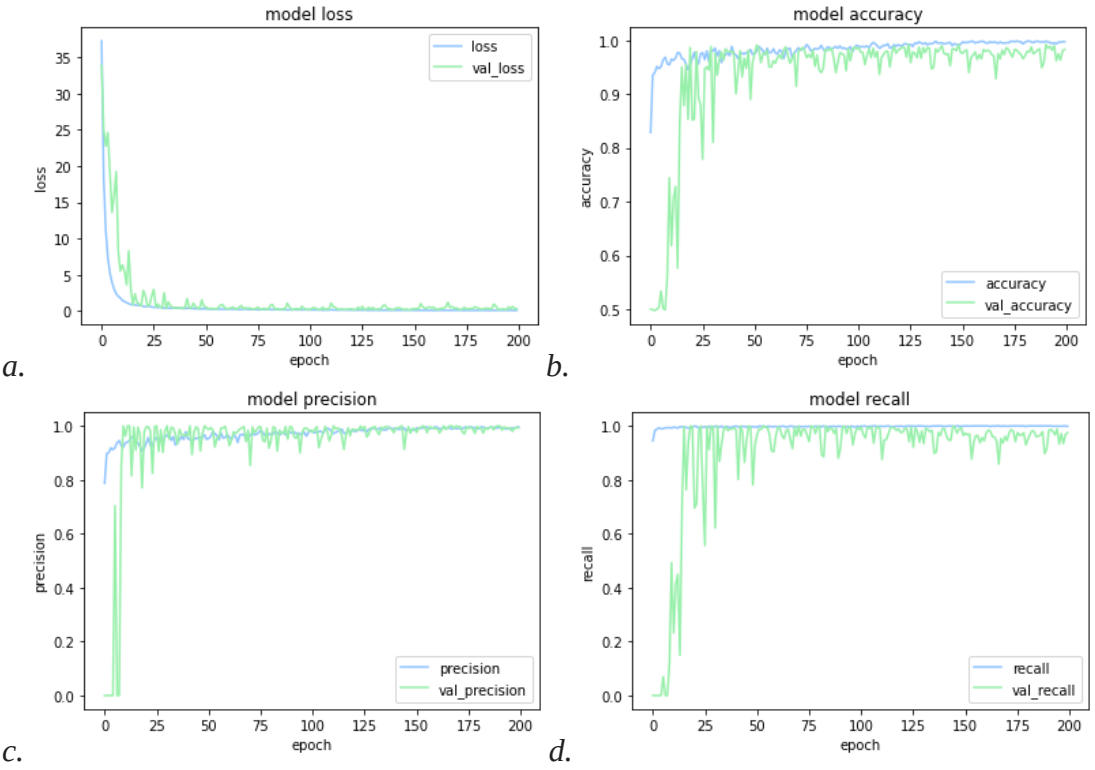


Fig. 30: a. Loss, b. Accuracy, c. Precision and d. Recall trends during training epochs

The minimum value for the validation loss was reached at epoch 149, therefore that model was saved for inference. The results show that satisfactory performances are achieved both on the training and on the validation sets. Table 6 reports the achieved values for loss and performance metrics used for classification in both sets:

	Loss (WBCE)	Accuracy	Precision	Recall
Training	0. 1178	0.9922	0.9846	0.9992
Validation	0.1590	0.9919	0.9906	0.9938

Table 6: Summary of loss's and metrics' values for training and validation sets

Fig. 31, 32, 33, 34 show different examples of apoptosis identification from the same images analysed in the 64x64 input case. In each of them, those windows predicted by the model to contain apoptosis are highlighted: true positive (TP) predictions (i.e. a window predicted to contain apoptosis which correspond an annotated event by ACAS software) are identified in green; false positive (FP) predictions (i.e. window predicted to

contain apoptosis which are not detected by ACAS) are identified in red. False negatives (FN) are not reported.

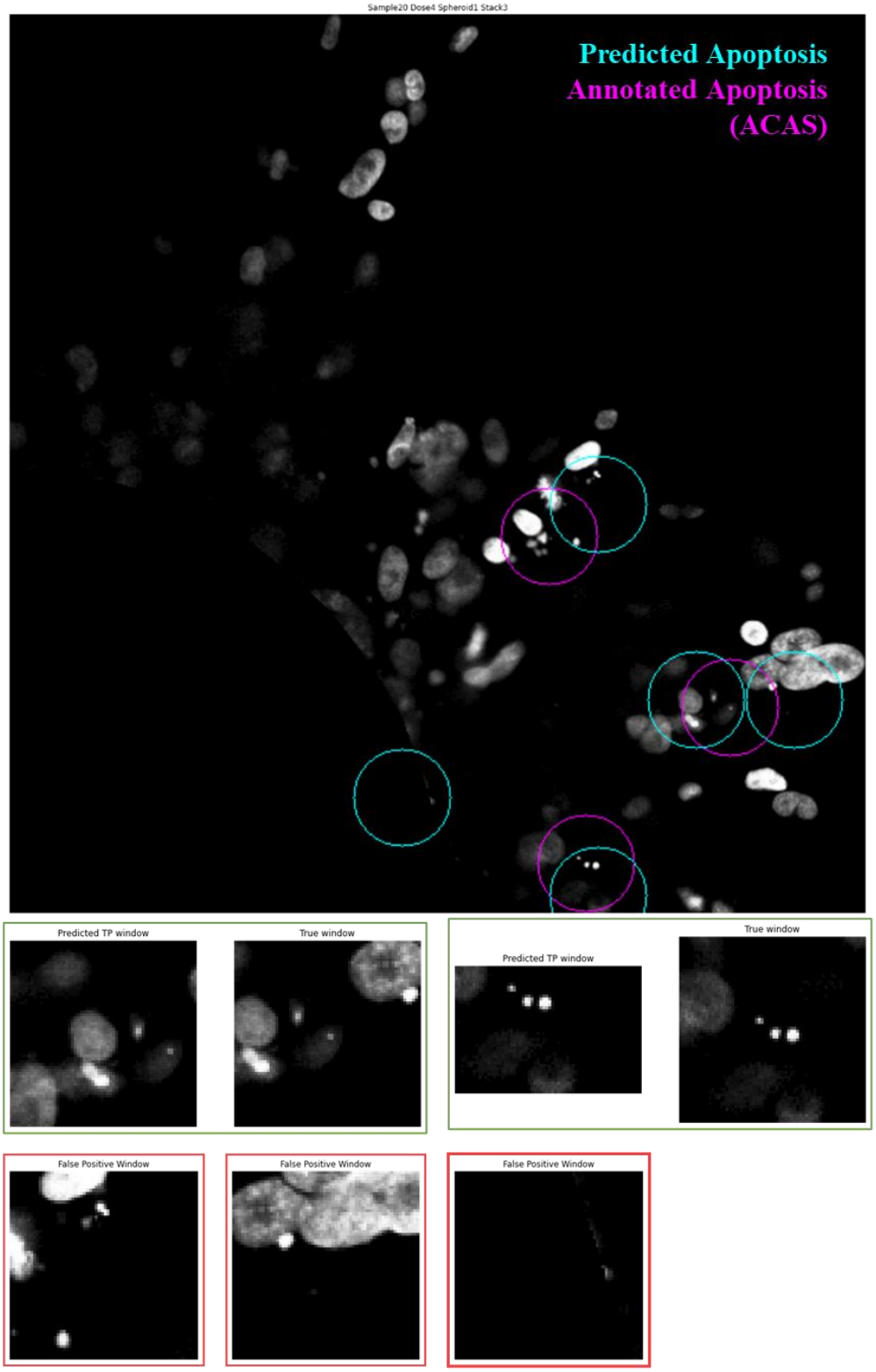


Fig. 31: Apoptosis detection in test image 1 (sample no. 20, dose 4Gy, spheroid no. 1, z-stack no. 3): 5 apoptoses identified. True Positives (TP) = 2, False Positives (FP) = 3, False Negatives (FN) = 1

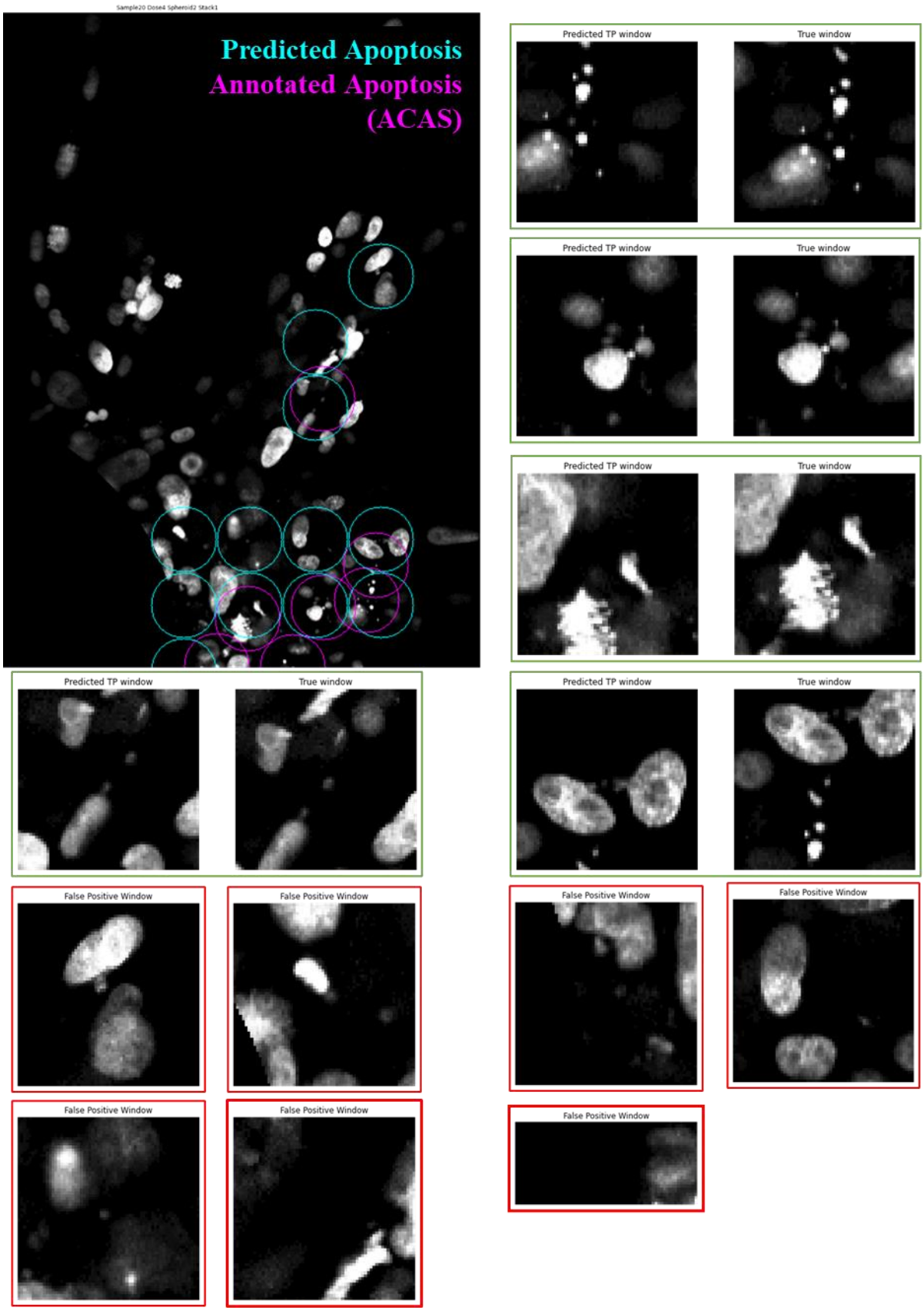


Fig. 32: Apoptosis detection in test image 2 (sample no. 20, dose 4Gy, spheroid no. 2, z-stack no. 1): 13 apoptoses identified. True Positives (TP) = 5, False Positives (FP) = 7, False Negatives (FN) = 2.

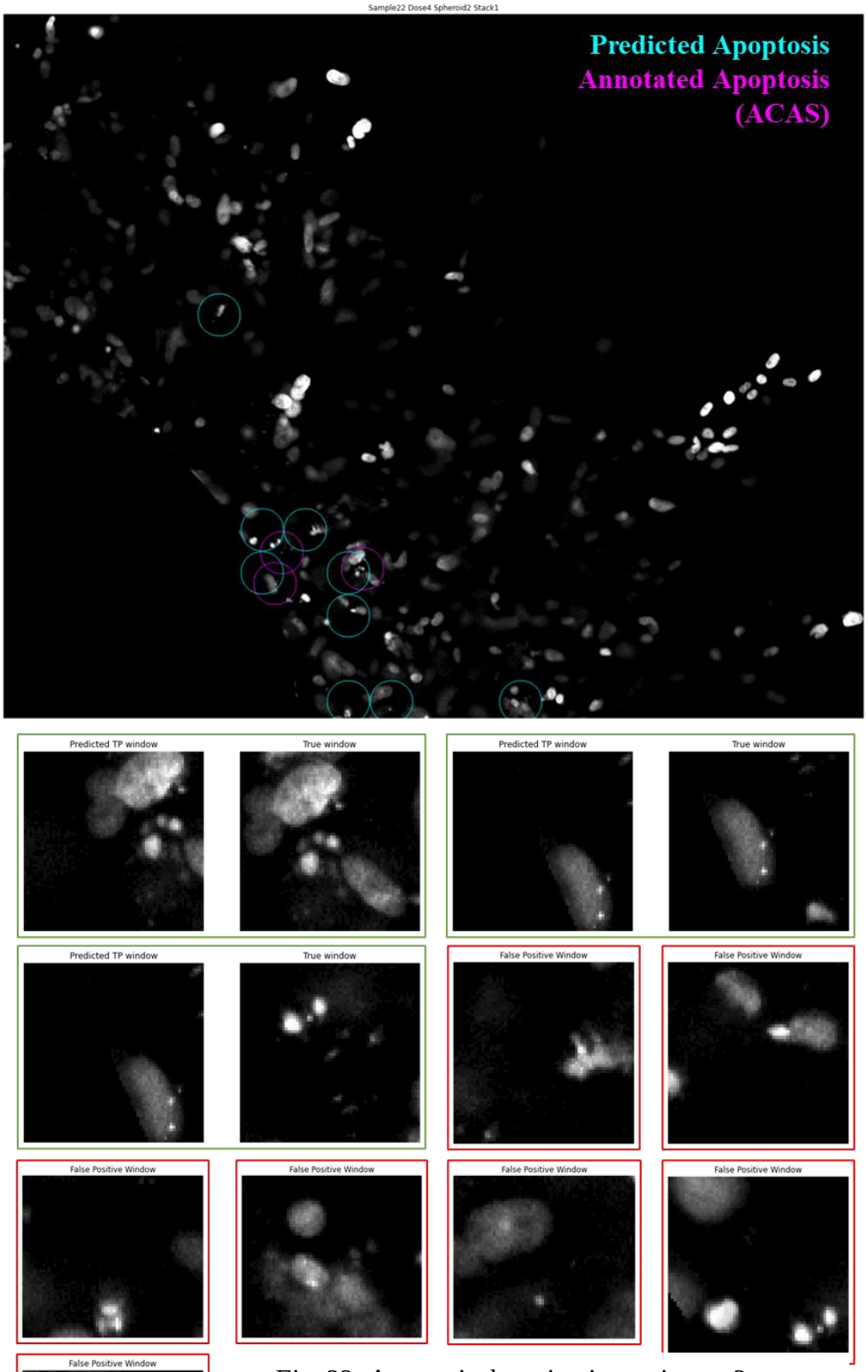
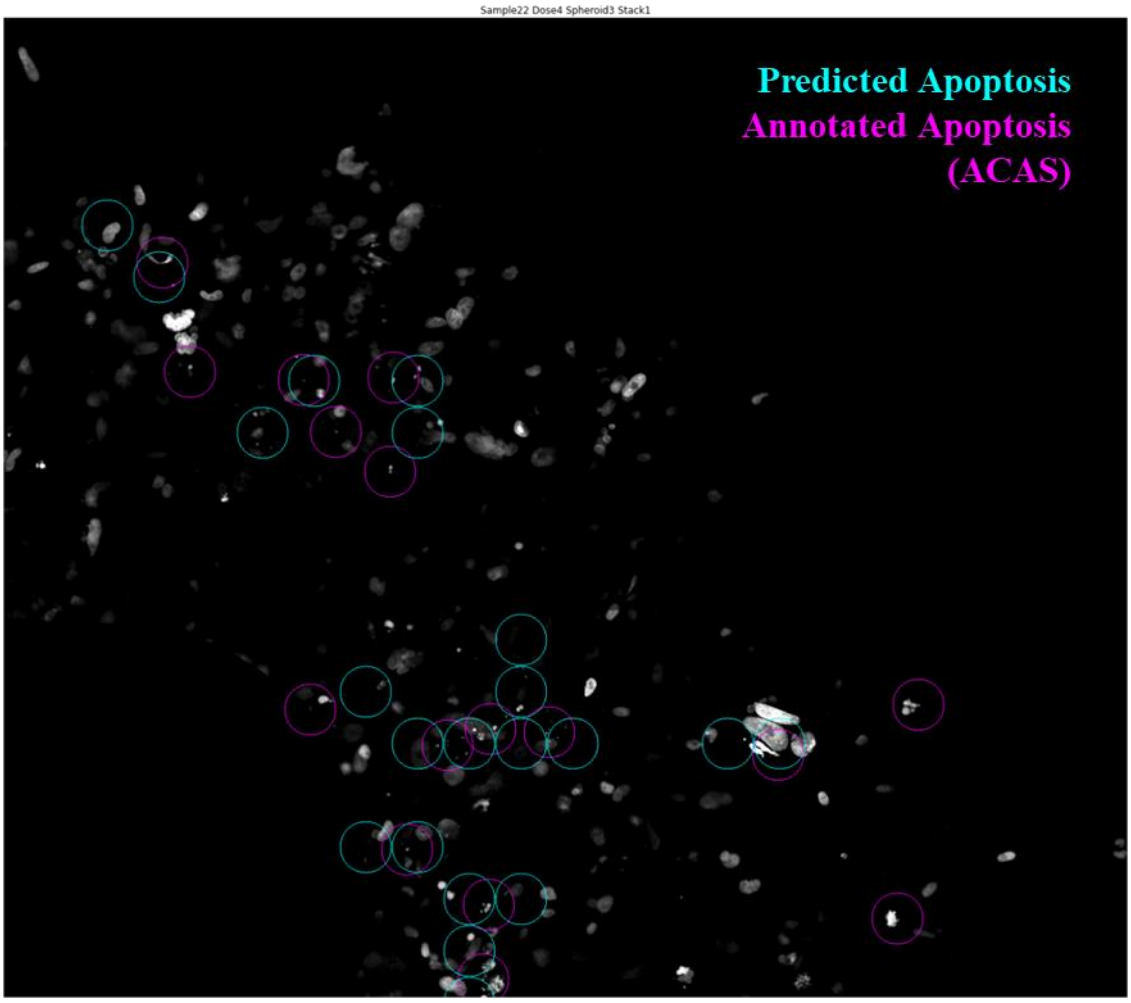
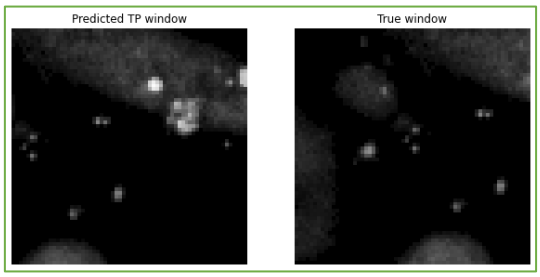
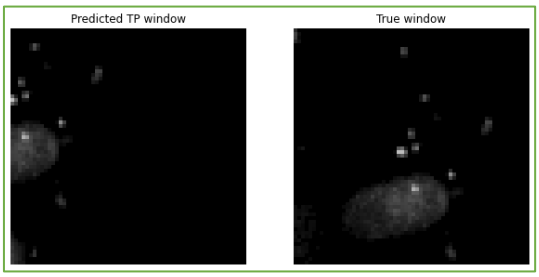
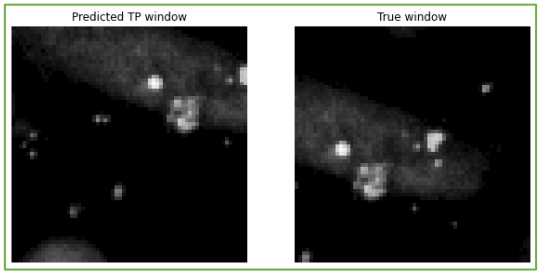


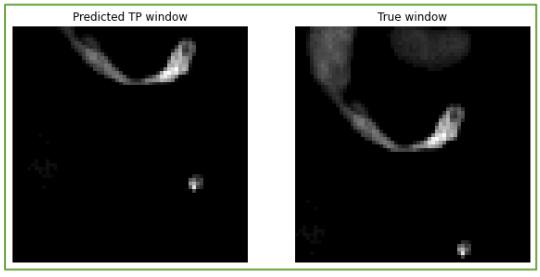
Fig. 33: Apoptosis detection in test image 3 (sample no. 22, dose 4Gy, spheroid no. 2, z-stack no. 1): 10 apoptoses identified. True Positives (TP) = 3, False Positives (FP) = 7, False Negatives (FN) = 0.


Sample22 Dose4 Spheroid3 Stack1
Predicted Apoptosis
Annotated Apoptosis
(ACAS)


Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window


Predicted TP window
True window

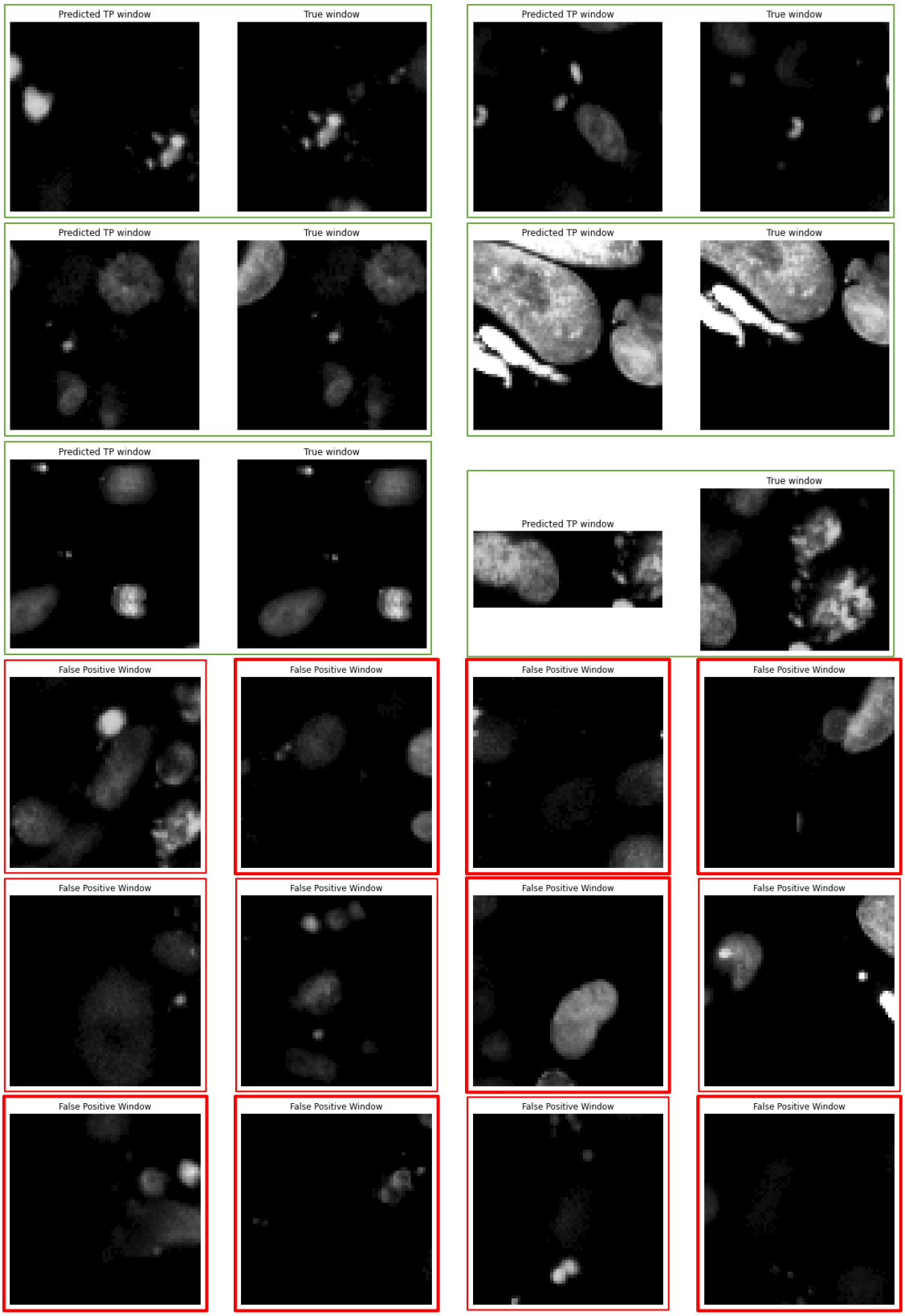


Fig. 34: Apoptosis detection in test image 4 (sample no. 22, dose 4Gy, spheroid no. 3, z-stack no. 1): 22 apoptoses identified. True Positives (TP) = 10, False Positives (FP) = 12, False Negatives (FN) = 6

Results show a worsening in the performances during SW test, with increased number of both FP and FN. A possible reason is the large gap between window size and stride compared to the dimensionality of apoptotic fragments to be detected, which are in the order of few pixels. Reducing the stride to 32 pixels resulted in an improvement of the performances, however computational time quadruplicated, resulting in no advantage compared to the 64x64 input-sized model applied with equal value of stride.

	Stride = 64				Stride = 32			
	TP	FP	FN	Time	TP	FP	FN	Time
s.20-sph.1-slice3	2	3	1	12s	3	3	0	46s
s.20-sph.2-slice1	5	7	2	12.5s	7	10	0	49s
s.22-sph.2-slice1	3	7	0	11s	3	7	0	48s
s.22-sph.3-slice1	10	12	6	8.5s	14	17	2	32s

Table 7: Comparison of results of the model tested over four images with stride = 64 and stride = 32.

The final outcome of this approach consists in a list of x and y coordinates reporting the centroids of windows (clustered if their distance is below a given threshold) which have detected an apoptotic event inside their content (Fig. 35 and Table 8). Such results, then validated by expert biologists, demonstrated that the implemented system is able to detect apoptosis more efficiently and with higher sensitivity than other software commercially available. It was not possible with this approach to further investigate the cell population, for example computing other useful indexes such as the apoptotic ratio, which instead became feasible with the implementation of the segmentation model.

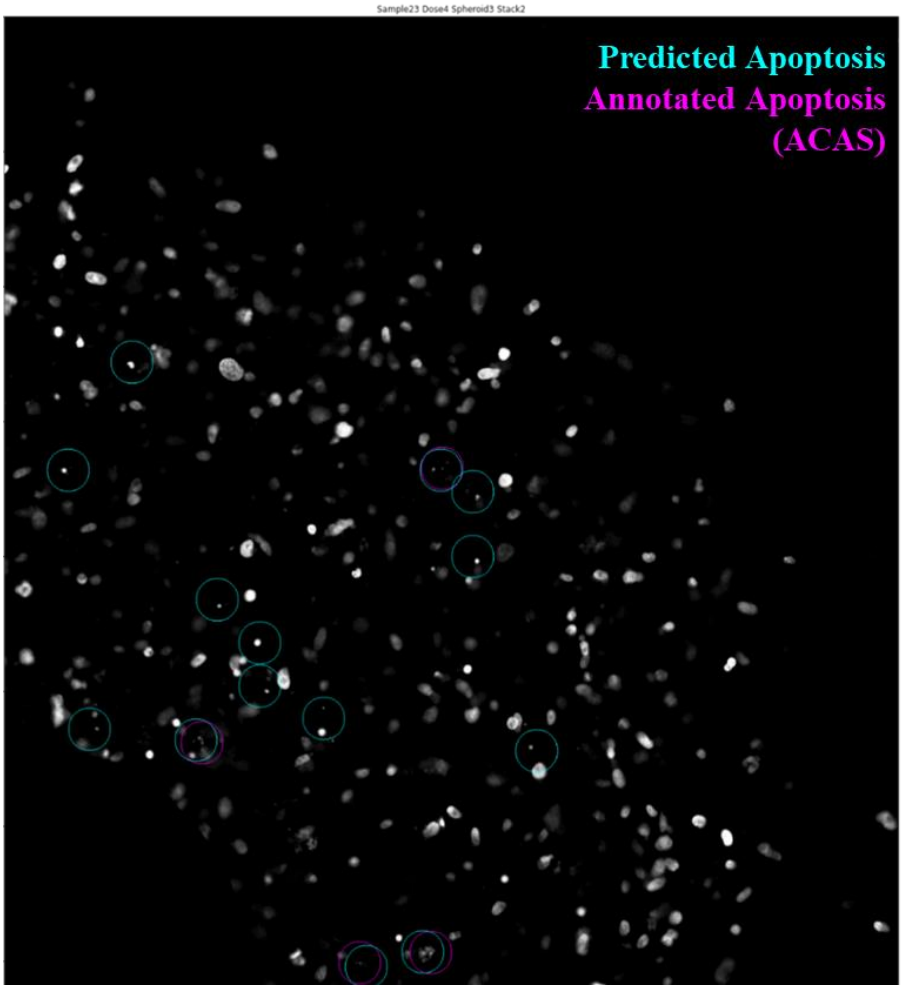


Fig. 35: Test image with predicted apoptotic windows.

Annotated Coordinates	Predicted Coordinates
(641, 1387)	(629, 1386)
(658, 669)	(656, 672)
(297, 1076)	(288, 1072)
(534, 1402)	(544, 1408)
	(800, 1088)
	(384, 992)
	(704, 704)
	(480, 1040)
	(128, 1056)
	(704, 800)
	(320, 864)
	(96, 672)
	(384, 928)
	(192, 512)

Table 8: Predicted coordinates for image in Fig. 35

3.2 Case Study 2: Semantic Segmentation with UNet

Counting the number of cells in microscopic images is essential for biological studies, allowing for the computation of useful indexes to quantify therapy effects, such as the already mentioned apoptotic ratio, that the previous method was not able to extract. For this reason, an automated image processing tool is developed to speed-up the work of biologists in the computation of such parameter. To achieve this goal, a CNN was trained to segment microscopic images, discriminating between intact nuclei, apoptotic fragments and the remaining background pixels.
After sematic segmentation is achieved, some post-processing is applied on the predicted output mask to extract the useful parameters.

Dataset

Input images also come from Dataset 1, which underwent the same pre-processing pipeline applied in the previous case study. Among the entire dataset, those images containing the largest number of annotated apoptosis were chosen, to limit class imbalance issues in the constructed dataset.
About 90 images were selected; they were cropped in the middle to fit the input size for the network of 512x512 pixels, and manually labelled sample by sample. Last, they were manually split in training and validation sets, and augmented by a factor of 10, thus maintaining the two datasets independent from each other.

Model Hyperparameters

The original architecture of UNet was maintained with all its hyperparameters. Only the initial number of filters in the first encoding block was reduced from 64 to 16 because of computational RAM constrains. This is turn translated in a faster training procedure, since the number of parameters to be optimized was much lower.

Algorithm Hyperparameters

Several combinations of values have been tested to tune algorithm hyperparameters. An appropriate choice of batch size and learning rate values was proved to be fundamental for achieving satisfactory results during the training process and a good network's optimization. In particular, it was found that choosing small batch sizes (below 16

samples per batch) and learning rates not too small (above 0.005) leads to greater generalization capabilities.

For these reasons, the final training strategy resulted in the following configuration: the batch size was set to 4, the learning rate to 0.005 at the beginning of the learning process and configured with cosine annealing up to the final value of 0.0005**.**

The loss function used for this case is a combination of the Dice loss calculated for intact mask, and Tversky loss calculated for apoptotic mask. Both of them belong to the category of region-based loss, aimed to maximize the overlap between ground-truth and predicted segmentation masks. Therefore, being metric-sensitive losses, they directly optimize the target metric, which makes them more suitable in medical segmentation tasks where performances are mostly evaluated in terms IoU or Dice score.

Moreover, such loss overcomes the imbalance issue of the target classes (intact and apoptotic) in respect to background, always preponderant in this type of images. Thus, Dice loss and Tversky loss only consider the proper segmentation class, overcoming imbalance issues with respect to background class.

Tversky loss is an asymmetric similarity measure, more suitable than common Dice loss for apoptosis class because it better mitigates the imbalance of such class in respect to intact class, achieving a better trade-off between precision and recall when adopted in fine-grained objects segmentation [57]. By adding two parameters in its computation, false negative can be penalized more. This becomes useful in yielding better segmentation results for the minority apoptotic class than using the common Dice loss.

These two computed losses are then combined in a weighted average, where the weight for each class is proportional to the inverse of class occurrences, with their sum normalized to 1. Resulting values computed for each sample in the mini batch, are then averaged over the batch size N:

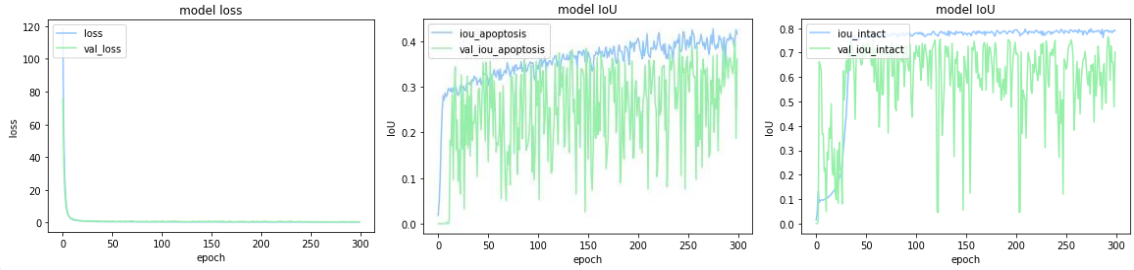
$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(1 - w_{Class} \cdot \mathcal{L}_{Class}) \tag{22}$$

This final loss is minimized by Adam optimization algorithm.

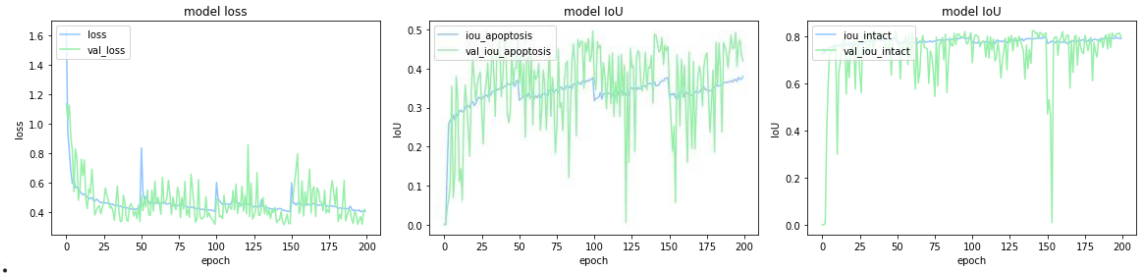
The model is trained for 300 epochs but saved at each iteration when either apoptosis IoU or intact IoU for validation set reached the maximum value up to that point.

Results of different training processes are shown in Fig. 36:

a.



b.



c.

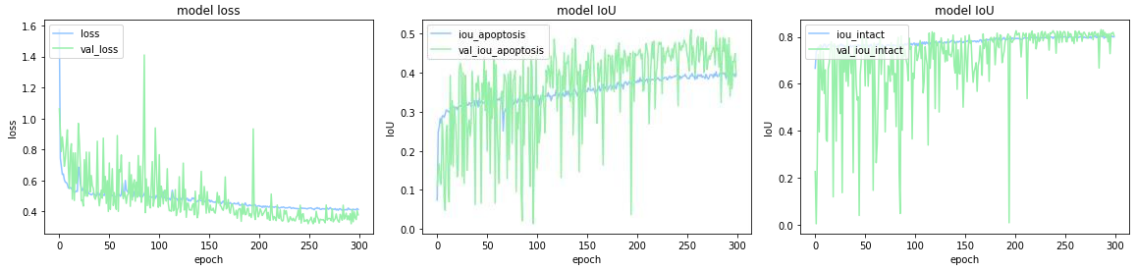


Fig. 36: Loss, IoU metrics for intact and apoptosis classes trends during training epochs. Training configuration: a. batch size = 8, lr = 0.001 – 0.0001, sigmoidal decay, number of training epochs = 300; training performed with a reduced number of samples
b. batch size = 4, lr = 0.005 – 0.001, cosine decay with restart every 50 epochs, number of training epochs = 200
c. batch size = 4, lr = 0.005 – 0.0005, single cosine decay, number of training epochs = 300

Semantic segmentation performances are evaluated in terms of IoU, as mentioned above, considering it also as a good criterion to choose the combination of weights at the end of each iteration of the training phase when the best performances are achieved over validation samples, independent from the training ones. The training configuration that achieved the best performance in terms of validation IoU was the one above mentioned, leading to a learning trend shown in Fig. 36 (c).

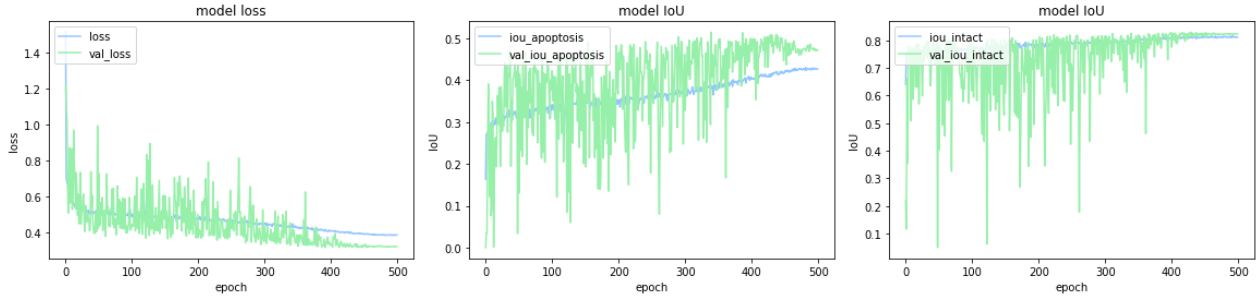
The maximum value for the validation apoptosis IoU was reached at epoch 255 with an average value over validation images of 0.5105. The maximum value for the validation intact IoU was reached instead at epoch 287, with an average value of 0.8599. Since the

apoptotic class resulted the most delicate in achieving good performances, the first model was considered for inference. The correspondent value for intact IoU was 0.8304.

These results indicate a satisfactory accomplishment of the task in the segmentation of intact cells, less for apoptotic fragments. Looking at the same metric over training samples, the maximum value that was reached during training, equal to 0.4026, clearly shows a lack of capability to capture the complexity of training data, and therefore to perfectly fit the expected prediction. The model complexity in terms of number of convolutional layers and encoding/decoding blocks could not be increased to better fit training samples for computational requirements, and also because the limited number of data would have caused overfitting problems.
Instead, the generalization capability of the model in handling unseen data was quite good, as the higher value achieved in validation metric for apoptotic mask suggests. Therefore, an increase in model complexity would have compromise such generalization capability, together with the fact that the training size should have been increased too in order to avoid overfitting. It has to be notice however, that the gap between validation IoU and training IoU is also to attribute to the presence of empty apoptotic masks in training samples, avoided when manually sampling the validation set, which lowers the final averaged value over instances.
Validation metrics appear fluctuating a lot over training iterations; lowering the learning rate did not improve this trend, causing instead the problem of getting stuck into a local minimum.
Batch size, limited to a value below 16 by the available GPU memory, proved to influence the time of convergence of the training process. The limited range prevented from low generalization but leaded to relative slow convergence of the training algorithm.

More specifically, a positive trend is visible in metrics' values over iterations (Fig. 36 (c)), indicating that their values would have continued to increase if the training process was longer. Results of two longer training procedures are shown in Fig. 37:

a.



b.

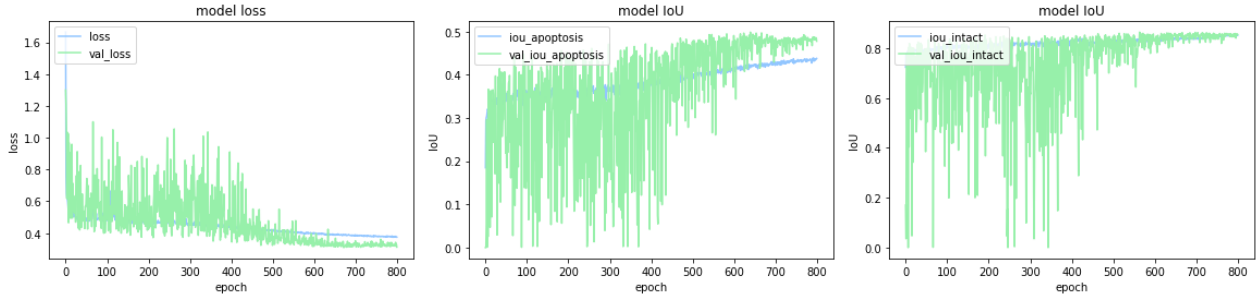


Fig. 37: Loss, IoU metrics for intact and apoptosis classes trends during training with

a. batch size = 4, lr = 0.005 – 0.0005, single cosine decay, training epochs = 500

b. batch size = 4, lr = 0.005 – 0.0005, single cosine decay, training epochs = 800.

These results still show a positive trend of metrics' values over iterations, suggesting to increase even more the number of training iterations. This was not possible for computational limits imposed by Colab.

Finally, it was observed that the mean IoU over training and validation samples were not optimized at the same time for the two classes, suggesting that the model was not able to learn an objective distinction of the two classes. This questioned the quality of manual labelling, affected by human subjectivity in precisely define the contours of cells which appear fading in the background.

Segmentation results over validation images are reported in Fig. 38 and Fig. 39.

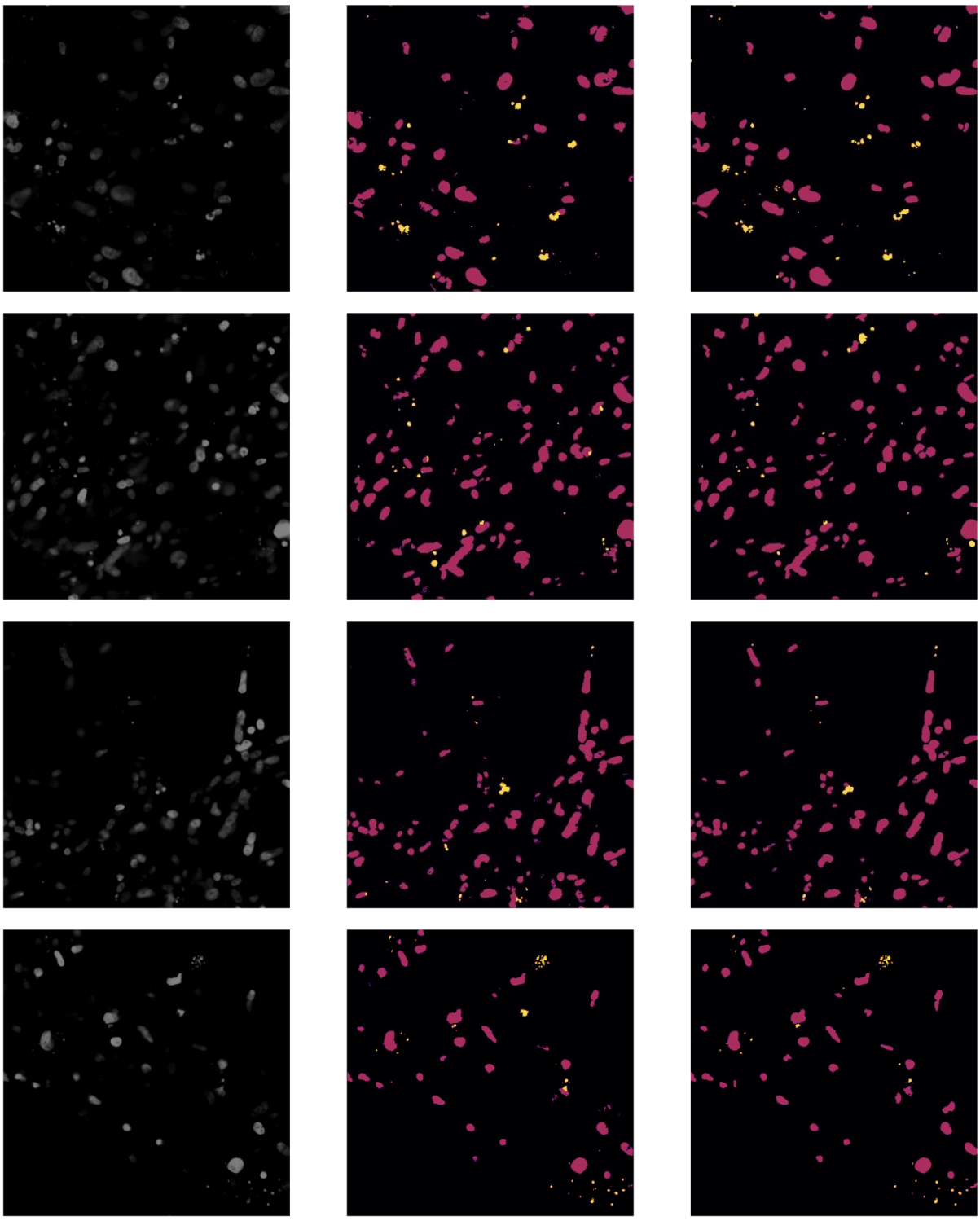


Fig. 38: Output predictions for 4 validation samples, one for each row: the first column reports the input image, second column the predicted output mask, the third column the corresponding ground-truth mask. Legend: pink regions identify intact nuclei, yellow regions the apoptotic fragments, black regions indicate the a-cellular background.

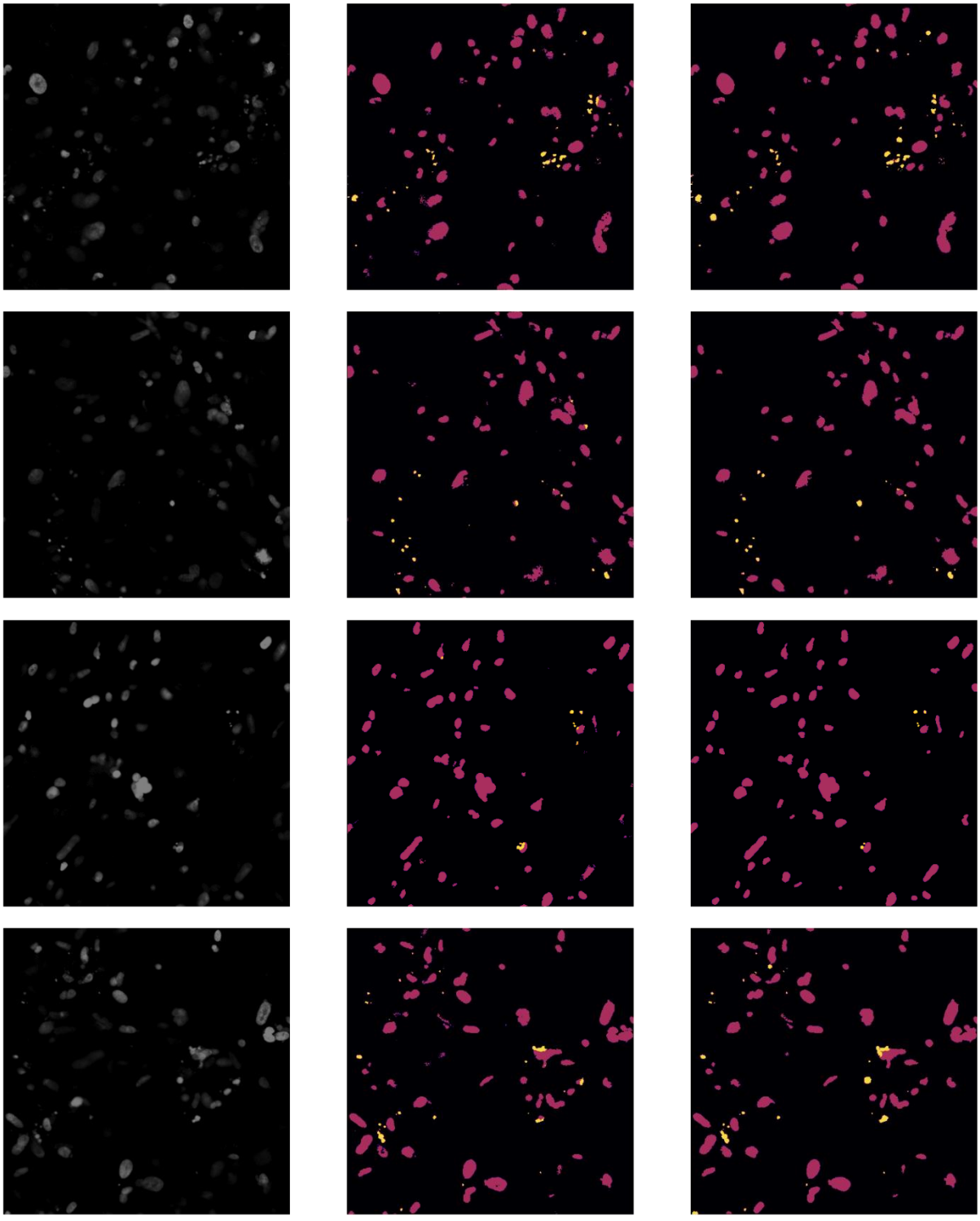
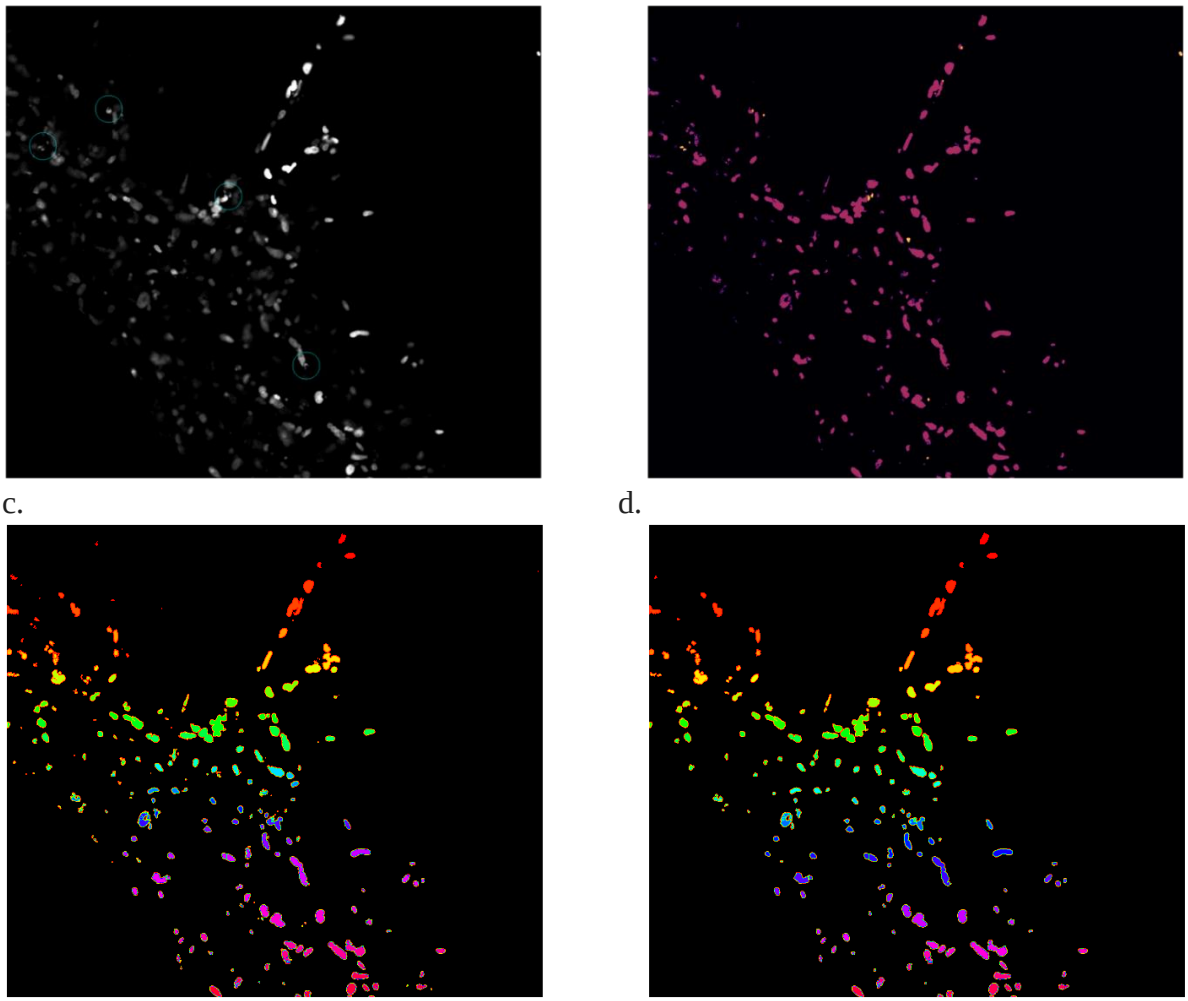


Fig. 39: Output predictions for 4 validation samples, one for each row: the first column reports the input image, second column the predicted output mask, the third column the corresponding ground-truth mask. Legend: pink regions identify intact nuclei, yellow regions the apoptotic fragments, black regions indicate the a-cellular background.

Once the segmentation mask was obtained by the model prediction, the number of intact and apoptotic cells was calculated. Obtained intact masks underwent simple post-processing involving filling holes appearing in the segmented nuclei and removal of small objects having an area in terms of count of pixels lower than a certain threshold, considering them just noise coming from the background which was wrongly counted as an intact nucleus. For apoptosis mask instead, apoptotic segments with a relative distance below a given threshold were clustered to be counted as a single apoptotic event, excluding isolated fragments, and computing the centre of mass for each cluster.

The resulting output is shown in Fig. 40, 41, 42, 43 for some test images: the first row reports on the left the input image with a circle in correspondence of an apoptotic event detected, centred in the resulted centroid of fragments; on the right, the obtained segmentation mask is shown; the second row reports the input mask directly from the model output on the left, with each instance coloured with a different colour, while post-processed mask with noisy pixels filtered out is shown on the right.

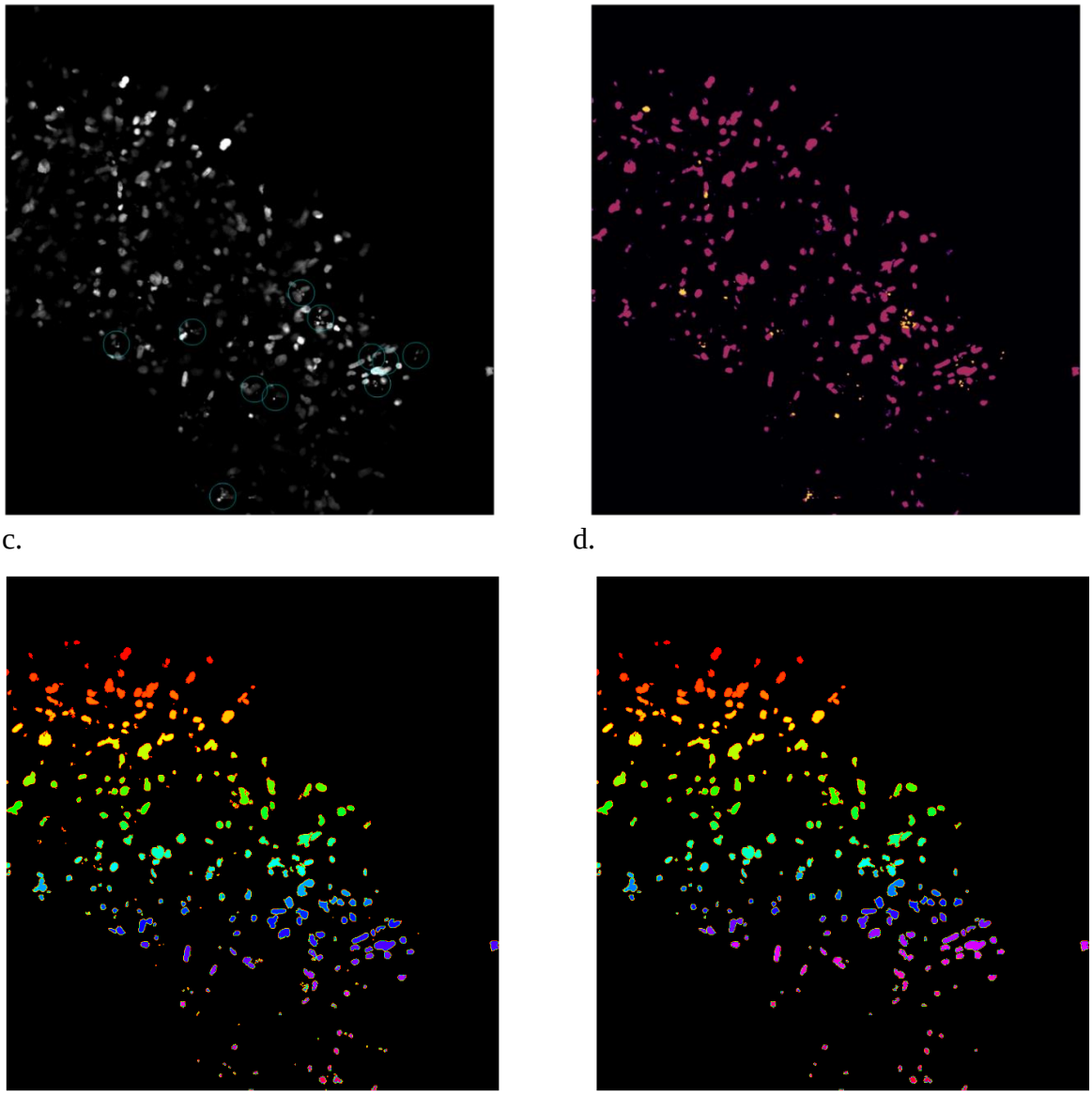
a. b.



c. d.

Fig. 40: Output prediction for image 'Sample 1, Dose 4Gy, Spheroid 1, Slice 1': a. input image with apoptotic events detected circled in light blue; b. predicted output mask (intact nuclei in purple, apoptotic fragments in yellow, background in black); c. predicted intact mask (each colour indicate a different instance); d. post-processed intact mask.

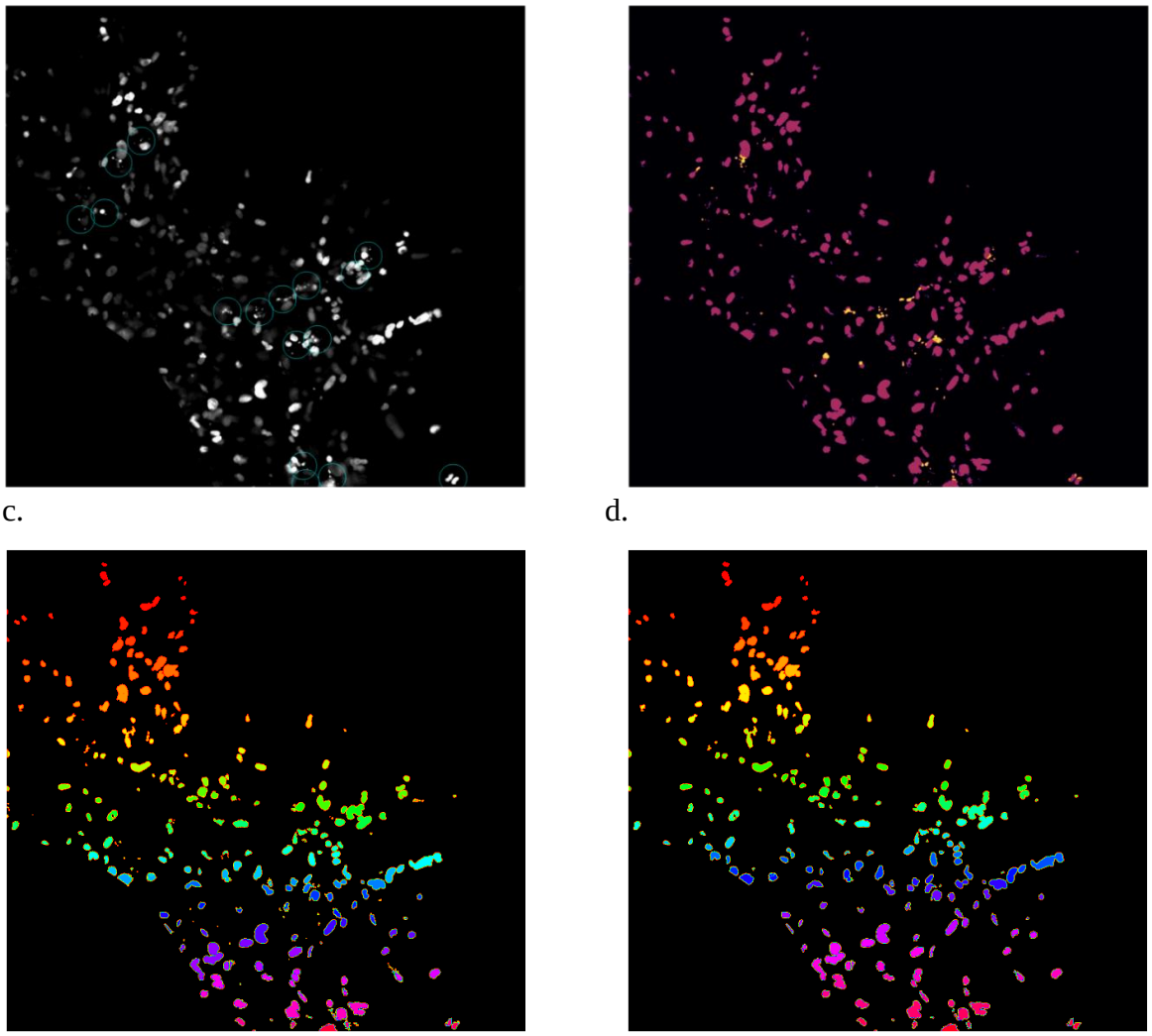
a. b.



c. d.

Fig. 41: Output prediction for image 'Sample 1, Dose 4Gy, Spheroid 2, Slice 3': a. input image with apoptotic events detected circled in light blue; b. predicted output mask (intact nuclei in purple, apoptotic fragments in yellow, background in black); c. predicted intact mask (each colour indicate a different instance); d. post-processed intact mask.

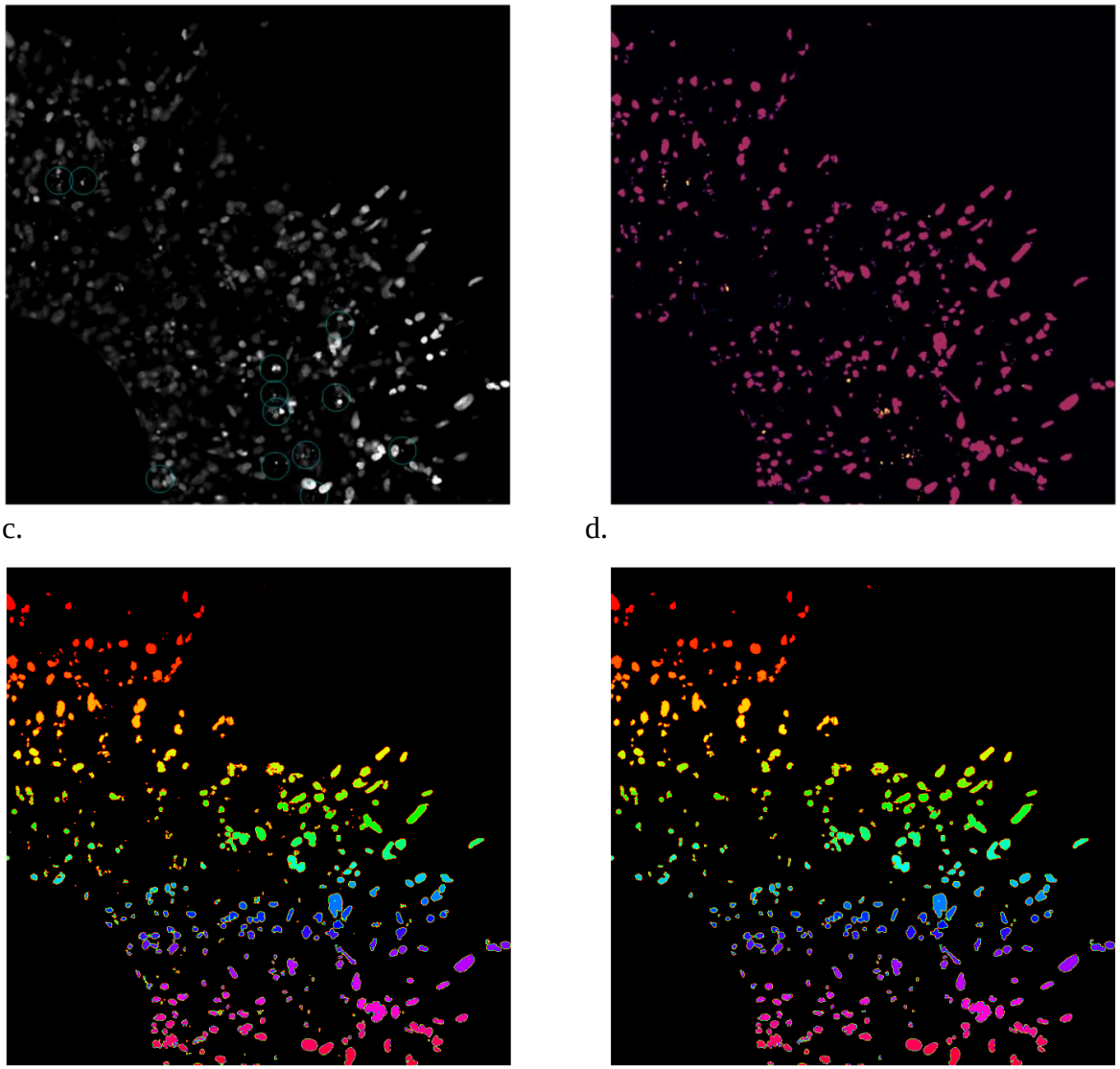
a. b.



c. d.

Fig. 42: Output prediction for image 'Sample 1, Dose 4Gy, Spheroid 3, Slice 3' a. input image with apoptotic events detected circled in light blue; b. predicted output mask (intact nuclei in purple, apoptotic fragments in yellow, background in black); c. predicted intact mask (each colour indicate a different instance); d. post-processed intact mask.

a. b.



c. d.

Fig. 43: Output prediction for image 'Sample 2, Dose 4Gy, Spheroid 2, Slice 1': a. input image with apoptotic events detected circled in light blue; b. predicted output mask (intact nuclei in purple, apoptotic fragments in yellow, background in black); c. predicted intact mask (each colour indicate a different instance); d. post-processed intact mask.

Obtained results are compared to software annotation. Always remembering the low software's performances in detecting apoptosis, the number of revealed apoptosis from the segmentation mask was always higher than the value reported in annotations. Therefore, discrepancies between the two values were considered acceptable. Retaining instead more reliable the reported number of intact nuclei, it resulted infeasible to perfectly predict the original number of annotated nuclei, revealing performance

limitations discussed after the observation of training metrics' values. A more accentuated underestimation of the number of intact nuclei was observed when cellular population was denser, imputable to the fact that overlapped nuclei, despite a good segmentation, accounted for a single instance. Simple algorithms can be applied to overcome this limitation, such as the watershed transformation.

A comparison between annotated and predicted counts of images shown in Fig. 40, 41, 42, 43, together with the computation of apoptotic ratio, is shown in Table 9.

Image	Annotated # Intact Nuclei	Annotated # Apop. Nuclei	Predicted # Intact Nuclei	Predicted # Intact Nuclei	Predicted Apop.Ratio
Sample 1, 4Gy, sph. 1, slice 1	187	0	179	4	0.0219
Sample 1, 4Gy, sph. 2, slice 3	238	6	241	11	0.0437
Sample 1, 4Gy, sph. 3, slice 3	235	10	228	16	0.0656
Sample 2, 4Gy, sph. 2, slice 1	439	4	372	12	0.0313

Table 9: Comparison between annotated and counted number of intact nuclei for images in Fig. 40, 41, 42, 43

Results were retained satisfying by biologists after inspection. The model prediction, despite not so high values reached in metrics indexes to assess performances, resulted adequate for correctly quantifying the number of intact cells and the number of apoptotic events.

The overall computational time for each image, including the same pre-processing pipeline applied for training and validation images, model prediction and post-processing of the predicted segmentation, is in the order of seconds, providing a significant time saving compared to manual procedures.

The model was tested also over images belonging to the second dataset provided. Results are shown in Fig. 44, respectively for high quality (Fig. 44 (a)), medium quality (Fig. 44 (b)), and low-quality (Fig. 44 (c)) images.

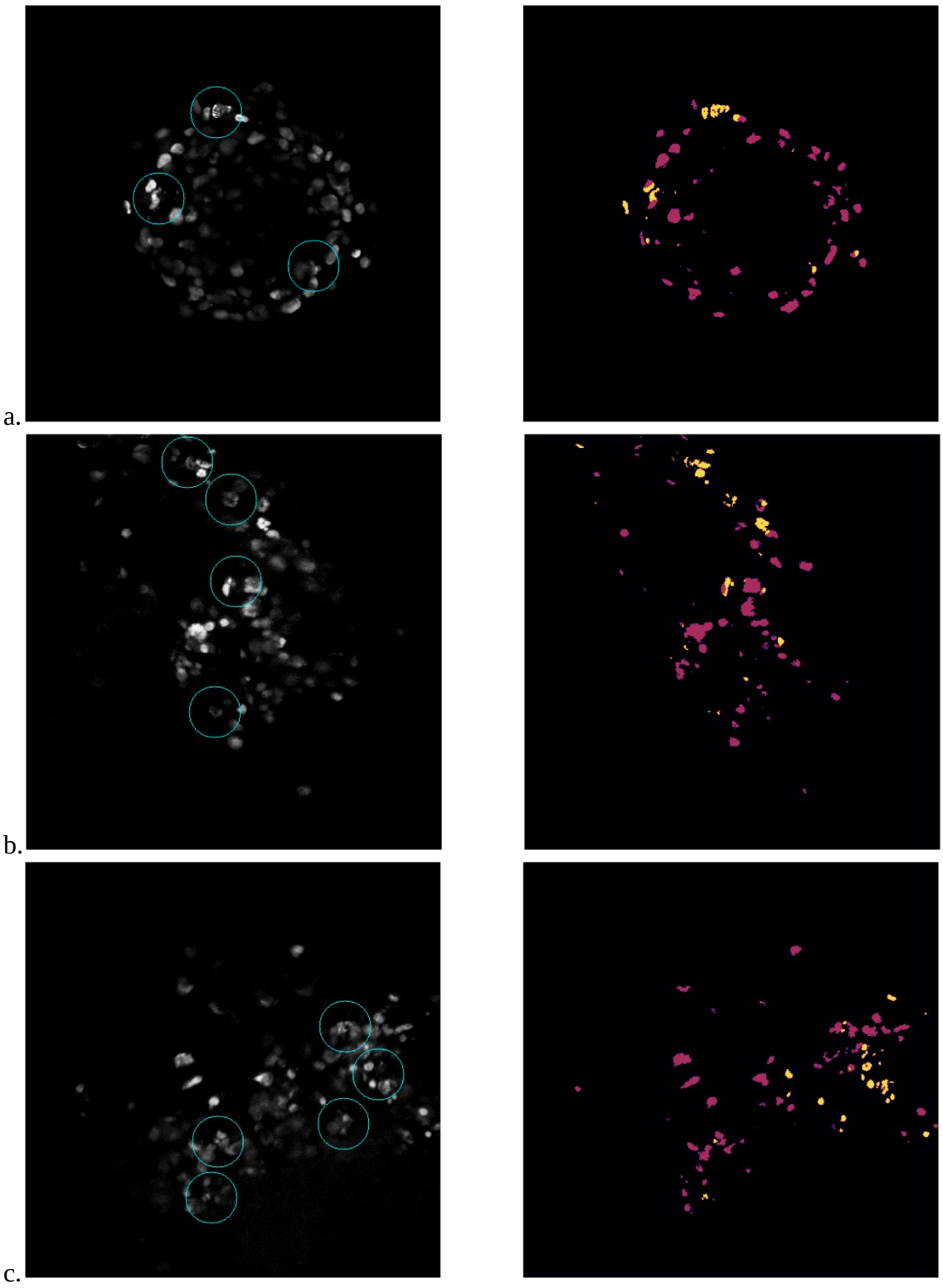
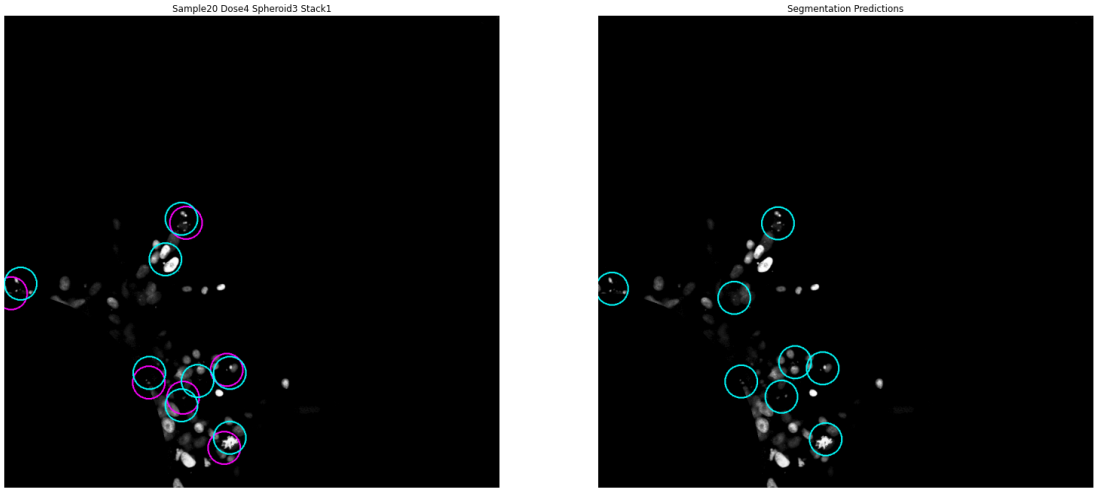


Fig. 44: Model prediction for a high-quality (a), medium quality (b) and low quality (c) image of dataset 2. First column reports the input image with the computed apoptotic centroid circled in light blue. Second column reports the output segmentation mask: intact nuclei in pink, apoptotic fragments in yellow, background in black.

3.3 Comparison of the Proposed Methods

A comparison between the two proposed approach is proposed in Fig. 45 and Fig 46, which illustrate in the first column the predictions made by the SW approach, on the second column the results of semantic segmentation after clustering apoptotic fragments detected for the same image:

a.



b.

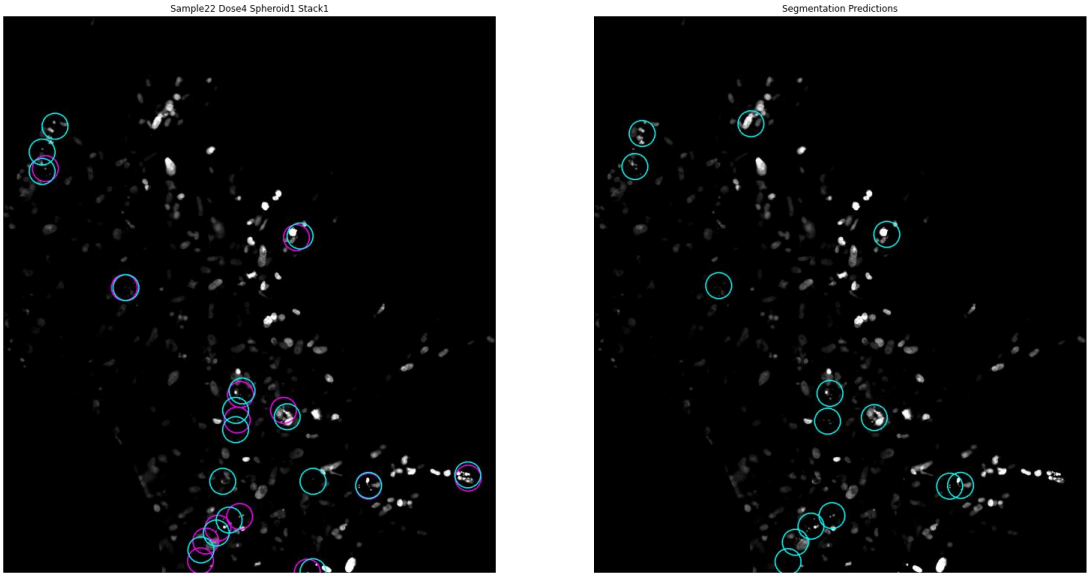
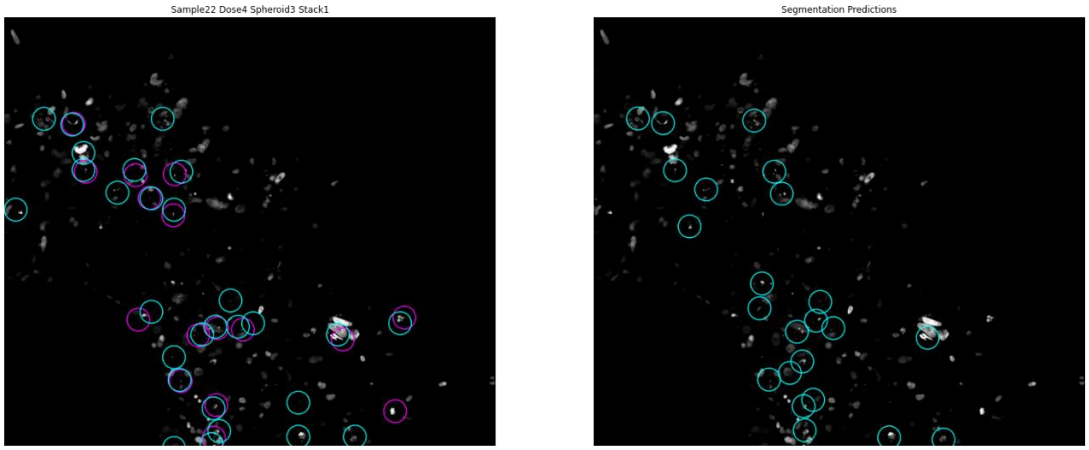


Fig. 45: Comparison between results obtained with SW detection and semantic segmentation for two images. Light-blue circles indicate the respective apoptosis events predicted; purple circles in first column reports ACAS annotations for each image.

a.



b.

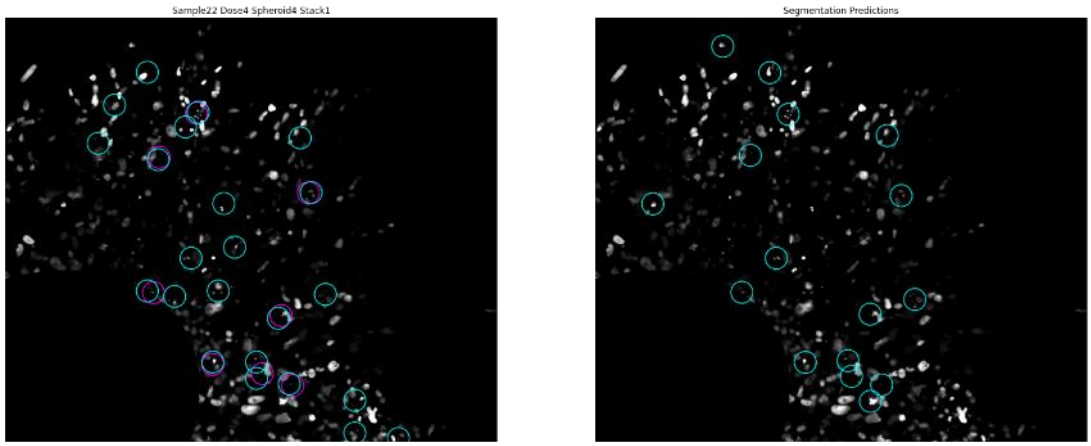


Fig. 46: Comparison between results obtained with SW detection and semantic segmentation for two images. Light-blue circles indicate the respective apoptosis events predicted; purple circles in first column reports ACAS annotations for each image.

From these figures, the comparison figured out that the extra predictions in respect to the available annotations was consistent between the two methods, confirming the presence of real apoptotic events in those positions. Also, the first method resulted more sensible in detecting apoptosis, with always a greater number of detections. If those predictions are correct detections, the lack of capability of the semantic segmentation model to detect them could be imputed to the lower performances reached during training.

However, the latter method has the advantage of more precisely localizing apoptotic events, being the centroid computed from the coordinates of detected fragments and not windows. Also, it resulted computationally more efficient, requiring just few seconds in order to detect apoptosis in the entire image.

4. Conclusions

The aim of this thesis was to build automated systems to help and speed-up the work of biologists for what concerns apoptosis detection among cellular nuclei in confocal microscopic images and calculation of correlated indexes, useful to quantify the effect of anti-cancer therapies.

Apoptosis has become an important target in cancer treatments, which attempt to impair the unregulated tumoral overgrowth by triggering the programmed death. Therefore, being able to monitor apoptosis results extremely important to quantify such therapy responses. Having tools that can help in discriminating apoptotic nuclei from the rest of population can relieve biologists of hours spent in manual detection by simply eye inspection, with the collateral risk of not being not too accurate because of human subjectivity.

In this scenario, deep learning emerges as a good solution, given the ability of producing fast and accurate analysis results, already demonstrated in many literature-annexed studies, with limited data. Generalization capabilities of CNN models are perfectly suited for this kind of problem, as apoptotic events appear very different one from the other in terms of number of fragments, dimensions, and sparsity.

Two case studies have been developed: the first one adopts an object detection technique, with a binary classifier trained to detect apoptotic fragments in small regions and then applied in sliding window fashion to test images. Once all the predictions are done, location of detected events is reconstructed.

To cope with the flaws of the first methodology, a second approach performs semantic segmentation distinguishing between three classes (intact, apoptotic nuclei and background).

Both of them are inserted in a working pipeline where minimal data pre-processing and post-processing to extract parameters of interest was required. For each application, a proper CNN architecture has been chosen and tuned for the specific case, together with proper loss and metrics definitions.

In the first case study, training results were successful, with validation metrics' values for accuracy, precision and recall above 0.98. At inference moment, an apparent high false positive number was observed, resulting instead in correct predictions when results were presented to biologists. This was justified by the fact that the ground-truth prediction used for comparison came from another automated software with lower sensitivity. Thus, the implemented system resulted much more efficient in accurately detecting all apoptosis inside the image.

The main disadvantage of the sliding window detection is the computational cost. Reducing the stride of the sliding window means to have a finer granularity allowing for a more accurate localization, but at the same time brings the disadvantage of increasing the computational time by a factor of 4. Choosing instead a larger stride, the number of windows to be predicted decreases, but the coarser granularity may impact both on detection and localization performance.

Thus, a proper trade of should be set between the accuracy of the detection (maximized when input size is set to 64x64 and stride equal to 32 pixels) and the time required for predictions (minimized when input size is 128x128 and stride 64). The first one was preferred for a correct accomplishment of the task, leading to inference time in the order of 50 seconds per image.

It resulted not possible to compare obtained results with a ground-truths labels, since provided annotations resulted only partially accurate, caused by the lack of sensitivity of the software previously used for the same analysis.

Obtained results have proved to be effective in relieving biologists from manually annotating cells, but some limitations are shown for what concerns a consistent time saving, since prediction times per image are comparable to those required for manually annotating images with the same size.

Future improvements will regard an improvement of the accuracy-prediction time trade-off. Considering the appreciable prediction capabilities, a reduction of computational time will be targeted, by privileging fully convolutional architectures that can implement the SW detection more efficiently, given the lower number of weights, therefore leading to faster predictions. Another improvement will be guided through a better minimization of false negative number, together with preventing an increase in false positive predictions.

In the second case study, semantic segmentation performances were evaluated in terms of IoU, separately for each class. Achieved values for intact nuclei segmentation in validation samples were retained satisfactory; for apoptotic nuclei instead, they did not suggest as good performances at first glance. This was not confirmed in test phase, resulting in a correct detection of all apoptotic events reported in available annotations and a number of apparent false positive detections, consistent with the partial accuracy of annotations, then corrected as true apoptotic events after biologists' inspection. Thus, results confirmed a considerable accomplishment of the task, achieved with limited data employment and thanks to the extensive use of data augmentation.

Future developments will be focused on the improvement of learning process, with a better hyperparameters' optimization to target an increase in performance metrics over training and validation samples. Different type of losses can be tuned to the specific case (focal Tversky loss, boundary loss) to reach the desired minimum. Also, more suitable metrics can be computed which can quantify the model performances more precisely than IoU. The amount of data used in training and validation sets will be increased, and quality of manual labels improved.

Finally, the two frameworks will be applied to new class of images (MPM) to study different cells populations.

References

[1] Simonyan, K., Zisserman, A., Very Deep Convolutional Networks for Large-Scale Image Recognition

[2] Ronneberger, O., et al., U-Net: Convolutional Networks for Biomedical Image Segmentation, 2015, arXiv:1505.04597

[3] Veelken, Cornelia, et al. "Single cell-based automated quantification of therapy responses of invasive cancer spheroids in organotypic 3D culture." Methods 128 (2017): 139-149.

[4] Elmore, Susan. "Apoptosis: a review of programmed cell death." *Toxicologic pathology* 35.4 (2007): 495-516.

[5] Cotter, Thomas G. "Apoptosis and cancer: the genesis of a research field." *Nature Reviews Cancer* 9.7 (2009): 501-507

[6] Kerr, John FR, Andrew H. Wyllie, and Alastair R. Currie. “Apoptosis: a basic biological phenomenon with wide ranging implications in tissue kinetics.” British journal of cancer 26.4 (1972): 239-257.

[7] Vogt, C. Untersuchungen über die Entwicklungsgeschichte der Geburtshelferkröte. (Alytes obstetricians) 130 (Jent und Gassman, 1842).

[8] Flemming W. Uber die Bildung von Richtungsfiguren in Saugethiereiern beim Untergang Graaf’scher Follikel. Arch Anat EntwGesh 221 (1885).

[9] Glucksmann, A. Cell death in normal vertebrate ontogeny. Biol. Rev. 26, 59–86 (1951).

[10] Kerr, J. F. A histochemical study of hypertrophy and ishaemic injury or rat liver with special reference to changes in lysosomes. J. Pathol. Bacteriol. 90, 419–435 (1965)

[11] Wyllie, A. H. Glucocorticoid-induced thymocyte apoptosis is associated with endogenous endonuclease activation. Nature 284, 555–560 (1980).

[12] Wyllie, A. H., Kerr, J. F. & Currie, A. R. Cell death: the significance of apoptosis. Int. Rev. Cytol. 68, 251–306 (1980).

[13] Enari, M. et al. A caspase-activated DNase that degrades DNA during apoptosis, and its inhibitor ICAD. Nature 391, 43–50 (1998).

[14] Horvitz, H. R. Nobel lecture. Worms, life and death. Biosci. Rep. 5, 239–303 (2003).

[15] Brown, D.A., Yang, N. & Ray, S.D. Apoptosis, Manchester University College of Pharmacy, Fort Wayne, IN, USA, Elsevier Inc., 2014

[16] Häcker, Georg. "The morphology of apoptosis." *Cell and tissue research* 301.1 (2000): 5-17.

[17] Wyllie, Andrew H. "Apoptosis: an overview." British medical bulletin 53.3 (1997): 451-465.

[18] Wyllie, A. H., et al. "Chromatin cleavage in apoptosis: association with condensed chromatin morphology and dependence on macromolecular synthesis." The Journal of pathology 142.1 (1984): 67-77.

[19] Orrenius, Sten, Pierluigi Nicotera, and Boris Zhivotovsky. "Cell death mechanisms and their implications in toxicology." Toxicological Sciences 119.1 (2011): 3-19.

[20] Krysko, Dmitri V., et al. "Apoptosis and necrosis: detection, discrimination and phagocytosis." Methods 44.3 (2008): 205-221.

[21] Savill, John. "Recognition and phagocytosis of cells undergoing apoptosis." British medical bulletin 53.3 (1997): 491-508.

[22] Wong, Rebecca SY. "Apoptosis in cancer: from pathogenesis to treatment." Journal of Experimental & Clinical Cancer Research 30.1 (2011): 87.

[23] Hanahan, Douglas, and Robert A. Weinberg. "The hallmarks of cancer." cell 100.1 (2000): 57-70.

[24] Kerr, J. F., Winterford, C. M. & Harmon, B. V. Apoptosis. Its significance in cancer and cancer therapy. Cancer 73, 2013–2026 (1994).

[25] Hanahan, Douglas, and Robert A. Weinberg. "Hallmarks of cancer: the next generation." cell 144.5 (2011): 646-674.

[26] Vaux, David L., Suzanne Cory, and Jerry M. Adams. "Bcl-2 gene promotes haemopoietic cell survival and cooperates with c-myc to immortalize pre-B cells." Nature 335.6189 (1988): 440-442.

[27] Banfalvi, Gaspar. "Methods to detect apoptotic cell death." Apoptosis 22.2 (2017): 306-323.

[28] Sgonc, R., and J. Gruber. "Apoptosis detection: an overview." Experimental gerontology 33.6 (1998): 525-533.

[29] Martinez, Michelle M., Randall D. Reif, and Dimitri Pappas. "Detection of apoptosis: A review of conventional and novel techniques." Analytical methods 2.8 (2010): 996-1004.

[30] Voulodimos, Athanasios, et al. "Deep learning for computer vision: A brief review." Computational intelligence and neuroscience 2018 (2018).

[31] L.C. Kimlin, G. Casagrande, V.M. Virador, In vitro three-dimensional (3D) models in cancer research: an update, Mol. Carcinog. 52 (3) (2013) 167–182.

[32] R. Edmondson et al., Three-dimensional cell culture systems and their applications in drug discovery and cell-based biosensors, Assay Drug Dev. Technol. 12 (4) (2014) 207–218.

[33] M. Vinci et al., Advances in establishment and analysis of three-dimensional tumor spheroid-based functional assays for target validation and drug evaluation, BMC Biol. 10 (2012) 29.

[34] Lee, June-Goo, et al. "Deep learning in medical imaging: general overview." Korean journal of radiology 18.4 (2017): 570-584.

[35] Sharma, Sagar. "Activation functions in neural networks." Towards Data Science 6 (2017).

[36] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Haifa, 2010, pp. 807–814.

[37] Najafabadi, Maryam M., et al. "Deep learning applications and challenges in big data analytics." Journal of Big Data 2.1 (2015).

[38] Topol, Eric J. "High-performance medicine: the convergence of human and artificial intelligence." Nature medicine 25.1 (2019): 44-56.

[39] Lindsay, Grace. "Convolutional neural networks as a model of the visual system: past, present, and future." Journal of Cognitive Neuroscience (2020): 1-15.

[40] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." arXiv preprint arXiv:1502.03167 (2015).

[41] Wu, Haibing, and Xiaodong Gu. "Max-pooling dropout for regularization of convolutional neural networks." International Conference on Neural Information Processing. Springer, Cham, 2015.

[42] https://cs231n.github.io/convolutional-networks/#pool

[43] Nielsen, Michael A. Neural networks and deep learning. Vol. 2018. San Francisco, CA: Determination press, 2015.

[44] http://www.python.org

[45] Zhao, Zhong-Qiu, et al. "Object detection with deep learning: A review." IEEE transactions on neural networks and learning systems 30.11 (2019): 3212-3232.

[46] Wu, Xiongwei, Doyen Sahoo, and Steven CH Hoi. "Recent advances in deep learning for object detection." Neurocomputing (2020).

[47] Xue, Yao, and Nilanjan Ray. "Cell Detection in microscopy images with deep convolutional neural network and compressed sensing." arXiv preprint arXiv:1708.03307 (2017).

[48] Cireşan, Dan C., et al. "Mitosis detection in breast cancer histology images with deep neural networks." International conference on medical image computing and computer-assisted intervention. Springer, Berlin, Heidelberg, 2013.

[49] Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

[50] Kotsiantis, Sotiris, Dimitris Kanellopoulos, and Panayiotis Pintelas. "Handling imbalanced datasets: A review." GESTS International Transactions on Computer Science and Engineering 30.1 (2006): 25-36.

[51] Mahani, Aouatef, and Ahmed Riad Baba Ali. "Classification Problem in Imbalanced Datasets." Recent Trends in Computational Intelligence. IntechOpen, 2019.

[52] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[53] Minaee, Shervin, et al. "Image segmentation using deep learning: A survey." arXiv preprint arXiv:2001.05566 (2020).

[54] Wong, Sebastien C., et al. "Understanding data augmentation for classification: when to warp?." 2016 international conference on digital image computing: techniques and applications (DICTA). IEEE, 2016.

[55] Dumoulin, Vincent, and Francesco Visin. "A guide to convolution arithmetic for deep learning." arXiv preprint arXiv:1603.07285 (2016).

[56] Sudre, Carole H., et al. "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations." Deep learning in medical image analysis and multimodal learning for clinical decision support. Springer, Cham, 2017. 240-248.

[57] Salehi, Seyed Sadegh Mohseni, Deniz Erdogmus, and Ali Gholipour. "Tversky loss function for image segmentation using 3D fully convolutional deep networks." International Workshop on Machine Learning in Medical Imaging. Springer, Cham, 2017.