POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Biomedica

Tesi di Laurea

Applicazione di reti bayesiane all'analisi automatica di immagini istopatologiche

Relatori
prof. Santa Di Cataldo
dott. ing. Francesco Ponzio

Candidato
Piero Policastro
matricola: 255612

Anno accademico 2019-2020

Sommario

Una delle più grandi differenze tra un essere umano e un software con intelligenza artificiale, sta nel fatto che quest'ultimo non può in alcun modo esprimere un dubbio su una questione che gli si pone dinanzi.

L'incapacità di poter esprimere l'incertezza induce a ritenere il sistema come non sufficientemente maturo.

Attraverso metodi che permettono di calcolare l'incertezza questo scoglio può essere superato.

I sistemi Bayesiani stanno destando sempre di più l'interesse del mondo scientifico proprio per la loro capacità di poter affrontare questo tipo di valutazioni. Di seguito ne verrà discusso lo stato dell'arte e le possibili applicazioni nel campo della medicina per l'interpretazione di immagini istologiche.

Le analisi saranno svolte principalmente sulla valutazione dell'incertezza sulla predizione. Verrà inoltre approfondito anche un altro ambito applicativo delle reti bayesiane: il datacleaning. Con questo temine si indica l'attività di eliminazione da un dataset degli elementi che non rispettano un determinato criterio. Questo principio nella maggior parte dei casi ha l'obiettivo di individuare la presenza di errori o rumore che renderebbero il dataset inconsistente.

Sarà dunque l'incertezza la caratteristica discriminante tra i vari elementi che si hanno a disposizione.

Verrà inoltre introdotto un applicativo, sviluppato dal candidato in Python, che permetterà di compiere delle analisi bayesiane su immagini ad alta risoluzione di pazienti affetti da adenocarcinoma o adenoma al colon-retto.

In conclusione verrà presentato un ulteriore strumento, sviluppato nello stesso linguaggio di programmazione del precedente, che utilizzando l'incertezza e un apposito algoritmo di selezione di soglia, permetterà di compiere datacleaning in maniera automatica su un dataset di immagini.

Ringraziamenti

Il candidato ringrazia la professoressa Santa di Cataldo e il professor Francesco Ponzio per l'opportunità di crescita concessagli e per la disponibilità mostrata.

E' doveroso ringraziare HPC POLITO per la dotazione hardware concessa che ha permesso di compiere agevolmente task complessi.

Desidero ringraziare la mia famiglia che durante questo percorso, anche se a molti chilometri di distanza, mi ha sempre appoggiato e sorretto soprattutto nei momenti di difficoltà.

Ringrazio Maria Adriana, la mia fidanzata, per la sensibilità unica che in tante situazioni ha dimostrato e Padre Ivan, il mio zio acquisito, per avermi insegnato a osservare la realtà con occhi diversi.

Un ultimo grazie va agli amici di vecchia data e a quelli incontrati in questi anni a Torino, per aver condiviso con molti momenti importanti della vita.

Indice

Elenco delle tabelle 6

Elenco delle figure 7

1 Introduzione e background 9
- 1.1 Machine learning e Deep learning: nozioni fondamentali 9
- 1.2 Elementi principali reti neurali 10
- 1.3 Istologia e deep learning 12
- 1.4 Teoria Modelli Bayesiani 14
  - 1.4.1 Overfitting e regolarizzazione 16
- 1.5 Convolutional neural network 18
  - 1.5.1 Funzioni di attivazione 20
- 1.6 Bayesian Convolutional Neural Networks 22
- 1.7 Modelli 23
  - 1.7.1 KL divergence 24
  - 1.7.2 Drop Out 25
- 1.8 Data augmentation 26

2 Il Dataset 27
- 2.1 Informazioni generali 27
  - 2.1.1 Tumore colon-retto 28
  - 2.1.2 Adenoma e Adenocarcinoma 29
  - 2.1.3 Whole slide images 30
  - 2.1.4 Openslide 32
- 2.2 Creazione dei dataset 33

3 Metodi 37
- 3.1 KL method 37
  - 3.1.1 Tensorflow piplines 37

3.1.2 TFRecods . . . 38
3.2 Drop-out method . . . 39
3.2.1 Keras . . . 39
3.2.2 PyTorch . . . 40

4 Risultati 43
4.1 Prestazioni KL . . . 44
4.2 Prestazioni drop out . . . 45
4.3 Considerazione su incertezza . . . 46
4.4 ROI cropping . . . 48
4.5 Data cleaning manuale . . . 49
4.5.1 Analisi dataset . . . 49
4.5.2 Prestazioni drop out . . . 50
4.6 Data cleaning automatico . . . 51
4.6.1 Otsu . . . 52
4.6.2 New threshold . . . 54
4.6.3 Comparazione tra metodi di soglia . . . 55
4.6.4 Prestazioni drop out . . . 56

5 Bayesian framework 59
5.1 Introduzione . . . 59
5.1.1 PyQt5 . . . 60
5.1.2 OpenSeadragon . . . 60
5.2 Analisi . . . 61
5.2.1 Analisi ui . . . 61
5.2.2 Struttura del programma di analisi . . . 66
5.3 Data cleaning . . . 68
5.3.1 Data cleaning ui . . . 69
5.3.2 Struttura del programma di datacleaning . . . 74

6 Conclusioni 77
6.1 Sviluppi futuri . . . 78

Bibliografia 81

Elenco delle tabelle

1.1 Stuttura modello KL 24
1.2 Modello drop out, strati di convoluzione 25
1.3 Modello drop out, strati densi 25
2.1 Numerosità dataset 33
2.2 Numerosità immagini estratte per classe 33
2.3 Composizione finale dataset 35
4.1 Accuratezza modello con Kullback–Leibler divergence 44
4.2 Accuratezza modello drop out dataset completo 45
4.3 Parametri data augmentation 45
4.4 Accuratezza modello drop out con data augmentation dataset completo 46
4.5 Accuratezza modello drop out su dataset con data cleaning manuale 51
4.6 Accuratezza modello drop out con data augumentation su dataset con data cleaning manuale 51
4.7 Numerosità nuovi dataset con soglia "Otsu" e "New" 55
4.8 Prestazioni del modello drop out con dataset ottenuto con "Otsu threshold" 56
4.9 Prestazioni del modello drop out con data augmentation, utilizzando dataset ottenuto con "Otsu threshold" 56
4.10 Prestazioni del modello drop out dataset ottenuto con "New threshold" 57
4.11 Prestazioni modello drop out con data augmentation su dataset ottenuto con "New threshold" 57

Elenco delle figure

1.1 Funzione di attivazione Relu 21
2.1 Suddivisione dei pazienti nei tre dataset. 34
4.1 Valori di incertezza totale delle tiles con predizione errata . . . 47
4.2 Fig.(a) Tessuto colon-retto con presenza di vasi sanguigni, Fig.(b) Stroma presente in tessuto colon-retto, Fig.(c) Crop raffiguranti maggiormente vetro. 47
4.3 Paziente 16, file numero 2. 50
4.4 In alto istogramma incertezza epistemica, seguito da istogramma incertezza aleatoria, in basso istogramma incertezza totale. 53
4.5 Visualizzazione delle soglie ricavate con le metodologie automatiche sull'istogramma dell'incertezza totale derivante dal training set. 54
4.6 Grafici a torta percentuale degli elementi rimossi nel training set con il metodo di: Otsu (a) e New threshold (b). 55
5.1 menu analisi, e possibile settaggio dei parametri. 63
5.2 Vista al termine dell'analisi. 64
5.3 Visualizzazione singola classe, attraverso l'utilizzo della barra degli strumenti. 64
5.4 Visualizzatore ad alta risoluzione su server locale. 65
5.5 Organizzazione dei file creati dal software di analisi. 67
5.6 Singolo elemento file json derivante dall'analisi. 68
5.7 Pagina dedicata all'estrazione delle tiles. 70
5.8 Pagina dedicata al training. 71
5.9 Pagina dedicata allo studio dell'incertezza. 72
5.10 Pagina dedicata al datacleaning. 73
5.11 Pagina dedicata alla fase di testing. 74
5.12 Singolo elemento file json derivante dal datacleaning. 76

Capitolo 1

Introduzione e background

1.1 Machine learning e Deep learning: nozioni fondamentali

Negli ultimi anni la crescente potenza dei computer e dei dispositivi atti a svolgere calcoli, come ad esempio le Graphics Processing Unit (GPU), hanno fatto sì che molti argomenti concepiti solo a livello teorico fossero percorribili anche per quel che riguarda la pratica, caso eclatante è i machine learning.

Con il termine machine learning si intende la capacità di una macchina di estrarre conoscenza autonomamente dallo studio di dati.

L'apprendimento automatico può essere diviso in tre macro aree:

1. apprendimento supervisionato, in cui al sistema oltre che l'input, si somministra anche l'output desiderato, in questo modo la macchina riesce a estrarre attraverso processo iterativo, il collegamento che esiste tra input e output,

2. apprendimento non supervisionato, si fornisce solo l'input al modello, senza alcun tipo di categorizzazione dei dati, uno dei possibili casi di applicazione può essere il clustering,

3. reinforcement learning, in cui un agente prende delle decisioni che vengono valutate attraverso un interprete che conferisce una ricompensa all'agente in caso di decisione corretta,

La crescente espansione del machine learning ha portato le società a costruire hardware apposito, che permettesse di migliorare ulteriormente i tempi di esecuzione, esempio sono le molto recenti Tensor Process Unit (TPU).

La crescita esponenziale della potenza di calcolo però è stata accompagnata anche dalla capacità dell'uomo di raccogliere e gestire grandi quantità di dati.

Sempre più frequente è, infatti, il riferimento ai Big Data, espressione con cui si indica la grande quantità di dati a cui si può accedere per cercare di estrarre nuova conoscenza.

La combinazione di queste due variabili ha portato alla ribalta un ambito del machine learning che inizialmente era stato accantonato: il deep learning. Su di esso si basano molti sistemi che utilizziamo quotidianamente, come ad esempio la trascrizione del parlato.

Il deep learning prende ispirazione dalla struttura del cervello umano, precisamente cerca di replicare le connessioni che sono presenti tra i vari neuroni e il modo in cui viene gestita gerarchicamente l'informazione.

Le Convolutional Neural Network prendono spunto infatti, proprio dal nostro apparato visivo, esse sono un particolare tipo di rete che permette di estrarre dalle immagini delle features, progressivamente più complesse, che poi potranno essere utilizzate per identificare oggetti.

Questo tipo di reti si sono dimostrate molto performanti e quindi vengono comunemente utilizzate in vari ambiti, come ad esempio la guida autonoma, o le si possono anche trovare in software atti all'identificazione di soggetti oppure oggetti. Un altro campo applicativo è quello medico, nel quale ci si può imbattere in software per l'identificazione di aree di interesse da immagini estratte da dispositivi medici.

In un mondo in cui l'uomo ha sempre una maggiore necessità di interagire con macchine che in autonomia possano prendere decisioni, o dalle quali possa scaturire un'informazione decisiva come ad esempio una diagnosi, cresce il bisogno di avere sistemi sempre più affidabili e accurati.

1.2 Elementi principali reti neurali

Come già detto, le reti neurali prendono spunto dall'organizzazione dei neuroni all'interno del cervello degli esseri viventi, è necessario quindi che il sistema creato possa generare un determinato output una volta ricevuto in ingesso un segnale.

I collegamenti tra i vari neuroni, che nel cervello sono detti sinapsi, nei sistemi ad apprendimento automatico verranno chiamati pesi. Attraverso questi sarà possibile conferire una determinata importanza alla connessione tra i due neuroni. I pesi quindi sono uno degli elementi più importanti in cui è custodita la conoscenza del modello.

Generalmente, sono caratterizzati da un numero, in base alla grandezza dello stesso, il neurone successivo riceverà un input più o meno grande.

Ogni neurone è associato a una funzione di attivazione, quest'ultima in base al segnale che riceverà in input, dato dalla somma di tutti gli output degli elementi precedenti moltiplicati per i rispettivi pesi, calcolerà il segnale di uscita. Queste funzioni verranno approfondite in un paragrafo successivamente.

Un ulteriore elemento fondamentale per l'apprendimento è la funzione obiettivo, più comunemente detta "loss function". Attraverso lo studio della suddetta funzione si può valutare in che misura la rete stia approssimando l'informazione presente nei dati. E' possibile dividere le varie tipologie di loss a seconda di quale è lo scopo del sistema: regressione o classificazione.

Con il primo termine si intende l'ambito dell'apprendimento supervisionato che ha come obiettivo quello di studiare funzioni continue, in modo da poter predire possibili andamenti futuri.

Le funzioni obiettivo maggiormente associate a questa tipologia di sistema sono l'errore quadratico medio, cioè la media del quadrato della differenza tra i valori predetti e quelli reali e l'errore assoluto medio, in questo caso è la media del valore assoluto della differenza tra la predizione e il valore reale.

L'obiettivo della classificazione è, invece, quello di predire la classe di appartenenza di un elemento. La classe predetta è ricavata dalla percentuale di appartenenza che è restituita tipicamente dall'ultimo layer avente come funzione di attivazione la funzione softmax o la sigmoide. La funzione loss comunemente utilizzata, in caso di problemi di classificazione binaria, è l'entropia incrociata, anche detta "Cross Entropy".

$$CrossEntropy = -(ylog(p) + (1-y)log(1-p))$$

Questa funzione incrementa di valore quando la predizione si allontana dalla classe reale dell'elemento. Per problemi multiclasse si ricorre invece alla "Categorical Cross Entropy", evoluzione del caso precedente ma studiata per l'analisi di problemi con più classi.

Sono necessari quindi degli algoritmi che modificando i pesi, riescano a minimizzare la funzione loss. Gli "optimization alghoritm" attraverso un

processo iterativo riescono a svolgere il suddetto lavoro, sono indispensabili per l'apprendimento del modello, dato che i pesi nell'istante iniziale sono assegnati in modo random.

Il metodo del gradiente è la tecnica più famosa, attraverso il calcolo del gradiente della funzione obiettivo rispetto ai pesi, permette di trovare il valore minimo della loss.

Un'altra tipologia è quella del gradiente stocastico che riprendendo la metodologia precedentemente trattata, prende in considerazione solo una parte degli elementi presenti nel dataset, selezionati in maniera stocastica.

Il "learning rate" è una costante, che associato agli algoritmi di ottimizzazione permette di aggiornare i pesi con una velocità a piacimento, potrebbe infatti capitare che i pesi vengano aggiornati così rapidamente da non poter raggiungere mai il minimo della funzione costo. Impostando quindi valori tendenzialmente piccoli di questa costante si riesce a evitare questo scenario

Sono stati sviluppati degli algoritmi di ottimizzazione che non hanno la necessità di indicare il valore di learning rate, infatti esso viene aggiornato in maniera automatica durante il processo iterativo di apprendimento, diminuendo gradualmente fino al raggiungimento della convergenza del sistema. Adam, è uno dei metodi appena citati. E' stato progettato principalmente per il deep learning e, come detto, adatta il tasso di apprendimento durante il processo stesso.

1.3 Istologia e deep learning

Il processo di cura di un paziente parte dalla corretta diagnosi del male che lo ha colpito, in questo modo si può procedere alla corretta somministrazione della terapia.

La strada comunemente percorsa per ottenere una diagnosi per molti tipi di malattie è studiare, con l'utilizzo di un microscopio, parti di tessuto che si ritiene possa essere interessato.

Il processo di estrazione di campioni di tessuto dalla zona di interesse comunemente è detto biopsia.

Il processo di analisi nella maggior parte dei casi è compiuto da un medico esperto in istopatologia, che in base ai propri studi e considerata l'esperienza, classifica il tessuto incriminato.

Alle volte può essere molto complesso identificare correttamente la presenza o l'assenza di particolari pattern che possano portare a una corretta diagnosi. Il punto di vista di un solo esperto quindi può ritenersi non sufficiente

per una corretta analisi.

In questo contesto sono entrati in maniera sempre più importante i software con intelligenza artificiale, questi infatti riescono a integrare informazioni variegate, ciò porta a ottenere in fase di classificazione risultati molto accurati che possono aiutare il medico durante il processo di diagnosi.

L'ambito in cui l'utilizzo di reti neurali artificiali sta avendo sempre più importanza è quello dell'elaborazione di immagini mediche, principalmente in radiologia, in istologia e in istopatologia. I sistemi basati sul machine learning infatti sono risultati molto prestanti per quanto riguarda il riconoscimento di pattern che comunemente non sarebbero stati considerati.

Uno ostacolo che in passato ha bloccato l'espansione dei sistemi sopra descritti sta nel fatto che questi non erano adatti a gestire una grande quantità di dati e inoltre raggiungevano prestazioni sempre migliori in base alla qualità dei dati che hanno in input.

Le informazioni sono ritenute di qualità quando al loro interno non è presente del rumore, cioè dell'informazione errata che potrebbe inficiare sull'apprendimento del sistema automatico, e quindi indurlo in errore. Esempio di quanto detto fino ad ora sono le immagini medicali, infatti al loro interno molto spesso possono essere presenti elementi non desiderati, oppure tessuto sano e malato che si alterna velocemente rendendo molto complesso la creazione di una mascherina per separare ciò che si è classificato.

Il deep learning a differenza di quanto detto finora, riesce a lavorare con grandi quantità di dati, utilizzando alcune tecniche, riesce anche a rendere marginale il problema del rumore.

Nuove ricerca provano a utilizzare sistemi intelligenti per collegare informazioni che fino ad ora erano considerate slegate, si è infatti arrivati a integrare alle classiche immagini mediche, dati derivanti da studio nel genoma del paziente in analisi.

Questo potrà aprire nuovi scenari e modi di visualizzare le stato clinico del paziente mettendo solide basi per la medicina personalizzata.

Futuri sviluppi potrebbero riguardare i wearable devices, si potranno quindi avere ancora più informazioni sullo stile di vita del paziente, questo porterà a un'ulteriore crescita di questo settore che già al momento sta mostrando ottimi risultati.

1.4 Teoria Modelli Bayesiani

I sistemi bayesiani stanno riscontrando sempre più interesse da parte del mondo scientifico principalmente per due fattori: la possibilità di calcolare l'incertezza sulla predizione e la capacità di avere intrinsecamente un termine di regolarizzazione.

Dare la possibilità a un sistema autonomo di poter manifestare la propria incertezza rispetto a una predizione fa sì che questi strumenti possano essere percepiti in maniera diversa da quanto fatto fino ad ora, conferendogli un'idea di maggiore robustezza.

Il segreto che si nasconde dietro i modelli che permettono di calcolare l'incertezza sta nella concezione completamente nuova che viene conferita ai pesi, essi infatti da semplici valori deterministici, diventano delle distribuzioni di probabilità.

In fase di training non sarà necessario quindi andare a determinare un valore singolo ma una distribuzione, che nel caso di una gaussiana sarà caratterizzata da media (μ) e varianza(σ^2).

La legge che è alla base delle considerazioni appena fatte è la legge di Bayes:

$$p(\omega|D) = \frac{p(D|\omega)p(\omega)}{p(D)}$$

La suddetta legge afferma che la probabilità posteriore è uguale alla probabilità condizionata per la probabilità priori dei pesi diviso per la probabilità a priori dei dati.

Con il termine probabilità condizionata si intende la probabilità che accada un evento nota una particolare condizione.

I termini di probabilità priori, invece, non sono condizionati da evidenze, rappresentano quindi l'informazione che già si ha su quella determinata variabile.

Applicando questo principio nell'ambito del machine learning sorge un termine di intrattabilità, infatti la distribuzione priori dei dati non può essere calcolata in un numero finito di operazioni.

Sono allora state intraprese due distinte strade per il superamento di questa problematica, entrambe però basate sul concetto di approssimazione:

1. variational inference: si utilizza un'altra distribuzione di probabilità per approssimare la vera distribuzione posteriore;

2. Monte Carlo sampling: si utilizza il campionamento per ovviare alla questione dell'intrattabilità;

A livello computazionale risulta più percorribile la prima opzione, quindi inizialmente è stata sviluppata la prima idea per andare ad applicare il teorema di Bayes sul machine learning.

La nuova distribuzione di probabilità, ipotizzandola di tipo gaussiano, potrà essere scritta come:

$$q_\theta(\omega|D)$$

Il termine θ indica l'insieme di due variabili media e varianza che caratterizzano la gaussiana.

Lo scopo dell'algoritmo di apprendimento sarà quello di calcolare μ e σ in modo da far avvicinare il più possibile la nuova distribuzione a quella vera.

Per calcolare la distanza tra due distribuzioni si utilizza la KullbackLeibler divergence. L'equazione da ottimizzare può essere scritta, quindi, come segue:

$$\theta_{opt} = argminKL[q_\theta(\omega|D)||p(\omega|D)]$$

Dato che per calcolare la KL divergence c'è bisogno di risolvere un integrale, è necessario ricorrere a un'altra approssimazione. In conclusione l'equazione da minimizzare può essere scritta:

$$\theta_{opt} = argmin - (\mathbb{E}_{q_\theta(\omega|D)}[logp(w \cap D)] + \mathbb{E}_{q_\theta(\omega|D)}[logq(w|D]) + logp(D)$$

In questa equazione si ripresenta un termine di intrattabilità, proprio come per l'equazione di partenza, l'ultimo logaritmo dipende dalla distribuzione di probabilità dei dati.

Questa problematica per parecchi anni ha bloccato lo sviluppo della tecnica bayesiana applicata al machine learning, fino a quando Blundell et al. 2015 riuscì a superare il problema integrando le due strade che inizialmente sembravano separate.

La tecnica del Monte Carlo sampling venne quindi applicata su una distribuzione che approssimava la distribuzione reale.

Recentemente si è scoperto che è possibile avere gli stessi risultati di una rete bayesiana utilizzando però le componenti classiche di una rete. Ciò è possibile con l'utilizzo del drop-out in fase di inferenza. Questo aspetto verrà discusso nel sotto paragrafo seguente riguardante specificatamente il drop-out.

Grazie a questa nuova rappresentazione dei pesi o all'utilizzo del drop out in fase di inferenza, classificando più volte lo stesso input è possibile ottenere risultati diversi.

Nel primo caso questo risultato si può ottenere grazie al concetto di peso come distribuzione. Campionando queste funzioni continue, ogni volta che si farà inferenza avremo delle reti diverse. Da ciò nasce la possibilità di ottenere risultati diversi partendo però da un singolo modello.

Per quanto riguarda il secondo caso invece, lasciando attivo il drop out anche in fase di inferenza, ogni volta che si effettuerà una classificazione si utilizzerà una rete diversa.

1.4.1 Overfitting e regolarizzazione

Uno dei problemi principali delle reti neurali è detto overfitting. Con questo termine si indica la capacità della rete di riconoscere molto bene solo i dati presenti nel training set, ottenendo però prestazioni molto scarse in termini di accuratezza su dati diversi da quelli da cui ha imparato.

Questo problema può presentarsi nella maggior parte dei sistemi di apprendimento automatico, quindi anche nel machine learning e nel deep learning.

L'overfitting principalmente può presentarsi in due casi: la prima è l'evenienza in cui la complessità del modello che si sta utilizzando non è proporzionale alla quantità di informazioni presenti nel dataset, il secondo è l'eccessiva durata del processo di training.

Per ovviare a questo problema si può ricorrere all'arresto anticipato del training. Questo metodo però non è sempre la strada migliore da intraprendere; sono state quindi sviluppate varie tecniche che vengono dette di regolarizzazione.

Attraverso l'utilizzo di queste metodologie, si ha un miglioramento delle prestazioni dei modelli ad apprendimento automatico, tuttavia i tempi di esecuzione del training si dilungano.

Una delle tecniche che consente di ottenere un effetto di regolarizzazione, è quella di andare ad aggiungere alla funzione costo una particolare componente. Questo nuovo elemento dipende dal valore dei pesi, ed è moltiplicato per una costante che permette di regolare l'intensità che si vuole conferire a questa nuova componente.

Recentemente sono state scoperte altre due tecniche che permettono di ottenere risultati simili senza la necessità di modificare la funzione obiettivo: il drop out e l'ancora più recente batch normalization. Le suddette metodologie, permettono di ottenere una regolarizzazione, senza il bisogno di modificare la funzione obiettivo.

Si è scoperto che i sistemi bayesiani possiedono intrinsecamente un termine di regolarizzazione, attraverso prove sperimentali è possibile notare infatti che l'effetto dell'overfitting su di esse è marginale.

Drop out

Con il drop out è possibile andare a "spegnere" un determinato numero di neuroni, per ottenere un effetto di regolarizzazione, si va quindi a modificare direttamente la rete. Per ogni epoca, infatti, si seleziona un certa percentuale di nodi all'interno dei layer nascosti che verranno disattivati.

Il processo di apprendimento continua con una nuova epoca disattivando altri elementi e verrà interrotto al raggiungimento del numero massimo di epoche.

Tipicamente viene utilizzato solamente in fase di training, tuttavia da recenti evidenze scientifiche si è scoperto come l'impiego di questa tecnica anche in fase di inferenza possa portare ad ottenere gli stessi risultati che si possono ottenere con i sistemi bayesiani puri.

Yarin Gal e Zoubin Ghahramani hanno dimostrato attraverso una ricerca[1], come l'applicazione del drop out prima di ogni layer è matematicamente equivalente a un sistema bayesiano che utilizza come distribuzioni dei pesi delle distribuzioni gaussiane.

Batch normalization

La procedura solitamente compiuta sui dati, prima che questi vengano trasmessi al modello, sia nella fase di training che in fase di inferenza è quella di scalare i valori tra 0 e 1, questa operazione prende il nome di normalizzazione.

La funzione di questa metodologia è duplice: velocizzare il raggiungimento della convergenza della rete e facilitare la comparazione dei dati a disposizione.

Il procedimento di apprendimento continua iterativamente fino a che non è soddisfatto il criterio di interruzione, se impostato, oppure fino a che si raggiunge il numero massimo di epoche preimpostate.

Esiste l'eventualità che una delle funzioni di attivazione presenti in un layer restituisca valori molto diversi da quelli presenti nello stesso livello. Attraverso il batch normalization si può andare a normalizzare anche i dati in uscita dai singoli layer della rete, evitando quindi la problematica appena enunciata.

Inizialmente si riteneva che questa tecnica andasse a minimizzare il problema della "Covariate Shift". Con questo termine è inteso il caso in cui

i dati presenti nel training set e nel test set non rispettano lo stesso tipo di distribuzione. E' quindi altamente probabile che il modello generato con un dataset affetto da questa problematica sia caratterizzato da prestazioni scarse soprattutto in quanto a generalizzazione.

Recenti studi hanno smentito questa teoria iniziale ed hanno dimostrato invece che il miglioramento delle performance che si ottiene con il batch normalization sia dovuto all'ottimizzazione della funzione obiettivo.

Riassumendo, risultano evidenti i miglioramenti delle performance generati da questa tecnica, tuttavia il mondo scientifico è ancora diviso sulle motivazioni che portano a ciò.

L'informazione però derivante da dati non è persa, infatti si aggiungono altri parametri che durante il processo di training verranno aggiornati a seconda proprio delle normalizzazioni che verranno eseguite.

Questa tecnica garantisce:

1. aumento velocità del training;

2. utilizzo di learning rate più alti;

3. robustezza al modello, quindi meno dipendente dai parametri.

Con il termine batch si intende un insieme di dati che vengono passati in successione durante la fase di training; questo permette di dividere il dataset in piccole parti.

In questo modo anche se la dotazione hardware che si ha a disposizione non è il meglio sul mercato, si riesce comunque a svolgere il training anche su dataset molto grandi.

Per completare un'epoca sarà necessario far passare tutti i batch attraverso la rete. Come si può intuire dal nome, le operazioni che vengono svolte dalla metodologia trattata in questo paragrafo, verranno svolte per ogni batch.

1.5 Convolutional neural network

Come spesso accade nel campo scientifico, si riesce a estrarre nuova conoscenza osservando ciò che avviene in natura. Le conoscenze acquisite potranno poi essere utilizzate nei più svariati ambiti

Lo stesso è successo con le Convolutional Neural Network, esse infatti provano a replicare ciò che era stato scoperto nel 1960 da Hubel e Wiesel per quanto riguardava il collegamento tra sistema nervoso e apparato visivo.

Dal loro studio si è evidenziato come a livello cerebrale l'informazione derivante dal sistema visivo attivasse solo particolari aree della corteccia visiva, e quanto più l'informazione era simile tanto più le aree che si attivavano a livello neuronale erano vicine.

Per quanto riguarda l'ambito dell'intelligenza artificiale, inizialmente per lo studio delle immagini si utilizzava le reti multilayer perceptron.

Queste però presentavano molti problemi, soprattutto durante l'analisi di immagini di grandi dimensioni perché necessitavano di un numero veramente molto elevato di nodi e questo accresceva in modo esponenziale la quantità di pesi.

Altra problematica molto grave che caratterizzava questo modello per lo studio delle immagini era che tutta l'informazione derivante dai patterns all'interno dell'immagine andava persa dato che il primo step per analizzarla era quello di trasformare la stessa in un vettore. Questo svantaggio era molto vistoso quando si andava a valutare l'accuratezza nel caso in cui in analisi erano presenti immagini molto grandi che quindi al loro interno potevano presentare allo stesso tempo pattern diversi .

Partendo da queste problematiche si è pensato a delle tecniche che permettessero di estrarre automaticamente delle features, e che poi attraverso l'utilizzo di queste potessero classificare le immagini. Utilizzando questo paradigma si riesce a risparmiare una grande quantità di tempo di cui necessitava l'ingegnerizzazione delle features.

Le Cnn sono delle reti di tipo feed-forward cioè ogni neurone è collegato a tutti i neuroni del layer successivo. Tipicamente l'architettura può essere divisa in due parti: la prima parte di feature learning in cui attraverso i filtri si estrae informazione, successivamente una sezione dedicata alla classificazione.

Questa struttura però comporta dei problemi infatti risulta essere molto incline all'overfitting, soprattutto nel caso di architetture di tipo deep, è quindi necessario ricorrere a metodi di regolarizzazione per eliminare queste problematiche.

L'operazione alla base di questo tipo di reti è la convoluzione che, dal punto di vista teorico monodimensionale, consiste nel calcolare l'integrare del prodotto tra due funzioni, di cui una viene traslata.

Per quanto riguarda le immagini, per compiere la convoluzione c'è bisogno di una struttura chiamata filtro che servirà, facendolo scorrere su tutta l'immagine, a riconoscere particolari features. Il risultato di questa operazione verrà salvato in una struttura chiamata feature map. I filtri, anche detti kernel, sono caratterizzati dai pesi; durante il processo di training questi valori si modificheranno in modo da riuscir ad estrarre feature informative.

Il risultato dell'operazione di convoluzione sull'immagine si può vedere come una nuova immagine con dimensione inferiore a quella di partenza. Questa caratteristica dipende dalla grandezza del passo con si fa scorrere il filtro sull'immagine, detto "stride", e dalla dimensione del kernel.

$$Featuremap = \frac{Image_{in} - kernel_{dim} + 2 * Padding}{Stride} + 1$$

Il padding è una tecnica che permette di ottenere delle feature map di dimensioni superiori, aggiungendo degli zeri intorno all'immagine di partenza.

Nella fase di training si andranno a modificare i valori dei pesi in modo tale da poter estrarre delle feature dalle immagini.

Nella maggior parte delle volte per un layer si possono utilizzare più di un kernel. Ne deriva, quindi, la possibilità di avere più feature map dallo stesso layer alle quali poi verranno applicate i kernel dei layer di convoluzione successivi.

I layer di convoluzione sono seguiti comunemente da un layer detto di Pooling; questo permette in prima istanza di ridurre la dimensionalità della feature map ma anche di evitare l'overfitting aumentando il livello di astrazione.

Per questa tipologia di layer non sono necessari pesi, basterà designare la grandezza del kernel e lo stride; questa finestra verrà quindi fatta traslare su tutta la feature map, e potranno essere compiute principalmente due operazioni:

1. Max Poolig, si va a prelevare il valore massimo del pixel all'interno di quella finestra.

2. Averege Pooling, si calcolerà la media dei valori dei pixel.

3. Min Pooling, sarà selezionato solo il valore minimo dei pixel.

In entrambi i casi alla fine di questa operazione si avrà una feature map di dimensioni inferiori a quella iniziale.

1.5.1 Funzioni di attivazione

Le funzioni di attivazione sono tra le più importanti componenti per quanto riguarda la costruzione di un sistema di machine learning.

Come accaduto per il concetto alla base delle CNN si è cercato di mimare un comportamento che comunemente avviene all'interno delle cellule cerebrali.

I neuroni per produrre un potenziale d'azione necessitano di una sollecitazione, essi infatti hanno bisogno di essere stimolati oltre una determinata soglia. Su questo concetto si basa il significato della funzione di attivazione, il neurone emette un output se stimolato entro certi livelli.

Inizialmente ci si serviva della funzione gradino che però presentava un problema quando la si abbinava al back propagation. La derivata della funzione di attivazione appena enunciata poteva essere molto elevata.

Successivamente la funzione di attivazione che divenne di uso comune nel machine learning fu la sigmoide, essa aveva il grande vantaggio di permettere il calcolo dei gradienti facilmente.

Per il deep learning questa funzione risultava inadeguata perché causava il problema del "vanishing gradinet". Studiando le reti di tipo profondo, si è notato che i primi layer hanno una capacità di apprendere inferiore rispetto a quelli successivi. In generale la capacità di modificare i pesi diminuisce nello spostarsi dagli ultimi layer fino a giungere ai primi. Nelle reti con molti layer il problema della scomparsa dei gradienti è molto accentuato.

Per superare questa problematica si può ricorrere alla funzione rampa comunemente chiamata "Rectified Linear Units" o più comunemente ReLU. Per valori negativi di input, l'output è zero, mentre tra zero e uno è presente una rampa, viene infatti anche detta funzione rampa.

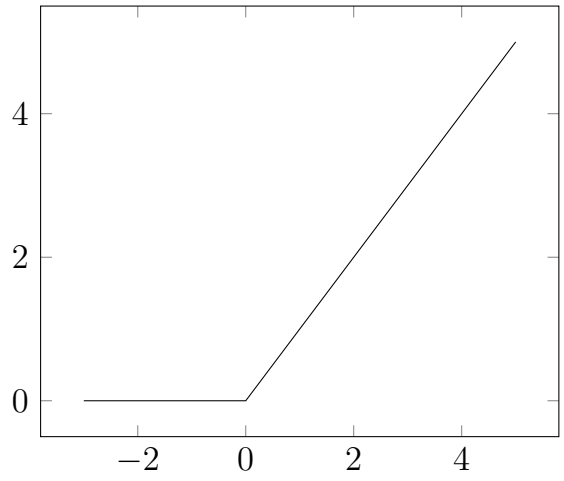


Figura 1.1. Funzione di attivazione Relu

Un'altra funzione di attivazione molto utilizzata per il layer di output nei sistemi di machine learning e deep learning, soprattutto per problemi multiclasse, è la softmax.

Essa restituisce dall'ultimo layer dei valori che sommati raggiungono 1, e che indicano la percentuale di appartenenza ad una classe. Tale informazione può essere interpretato come probabilità di appartenere.

Erroneamente l'output di questa funzione di attivazione può essere inteso come una percentuale che indica quanto la rete è sicura della classificazione, tuttavia un elemento che ottiene in uscita dalla softmax un valore elevato per una determinata classe, può tuttavia avere un valore di incertezza elevato.

1.6 Bayesian Convolutional Neural Networks

Il processo di apprendimento di sistema supervisionato può essere diviso principalmente in 2 parti: training e test.

Durante la prima parte, si forniscono dei dati in input alla rete, grazie ad essi si otterrà un output che verrà confrontato con il taget.

In questa prima comparazione ci sono le fondamenta dell'apprendimento della rete. In base a questa infatti si andranno ad aggiornare i pesi dei collegamenti tra i neuroni che hanno portato a quel determinato risultato, partendo dalla fine della rete fino all'inizio.

Quello appena enunciato è un principio molto importante dei sistemi di machine learning, viene denominato back propagation e, attraverso esso, la macchina riesce a imparare. Il processo di training è iterativo, infatti la rete impara lentamente, quindi è necessario che essa riceva in ingresso un numero ben determinato di volte gli input.

Una volta concluso il training si passa alla fase di test, durante la quale la rete viene utilizzata per classificare elementi mai fornitigli durante il processo di apprendimento. In questa fase è possibile, quindi, valutare la capacità della rete di generalizzare, da ciò si potrà comprendere se il modello ha realmente imparato.

Nell'utilizzo finale, può capitare che si richieda alla macchina di esprimersi su oggetti che non ha mai visto. La suddetta evenienza è la normale routine di questi sistemi.

Sorge però un problema quando ciò che le si presenta è qualcosa di completamente diverso da quello da cui il sistema ha tratto la sua conoscenza.

Come già detto l'apprendimento supervisionato basa il training su set di dati categorizzati in classi, la problematica precedentemente espressa riguarda l'evenienza in cui, durante la fase di inferenza, si faccia classificare al modello un dato che appartenga a una classe diversa da quelle inizialmente presenti in fase apprendimento nel dataset.

In questa eventualità il modello sarà obbligato ad assegnare l'input a una delle classi, e non avrà in alcun modo la possibilità di esprimere la propria difficoltà derivante dallo svolgimento di questa operazione.

Per cercare di superare questo tipo di problema si può ricorrere a sistemi che permettono di stimare l'incertezza sulla predizione; quest'ultima può essere ripartita in due componenti:

1. incertezza aleatoria, deriva dall'incertezza del modello che si sta utilizzando.

2. incertezza epistemica, riassume l'incertezza derivante dalla carenza di dati.

Queste quantità vengono valutate attraverso l'utilizzo di particolari tecniche che permettono di avere più output, per lo stesso input.

Classificando quindi un singolo elemento più volte, sarà possibile avere risultati diversi.

Per ottenere i risultati sopra citati si è superato il classico modo in cui si è pensato a una rete.

I pesi, nelle reti classiche, sono dei valori numerici che permettono di dare o meno 'importanza' al collegamento che c'è tra 2 neuroni. Questo concetto ora viene superato, i pesi infatti in questa nuova concezione di rete derivano dal campionamento di una distribuzione continua, che caratterizza ogni collegamento tra neuroni.

1.7 Modelli

Per fare un'analisi di tipo bayesiano si può ricorrere a due diverse metodologie. La prima prevede l'utilizzo di particolari layer che permettono di calcolare ciò che è stato esplicato dal punto di vista teorico, utilizzando quindi la Kl divergence; la seconda invece prevede l'utilizzo delle componenti comunemente utilizzate per la costruzione di una CNN.

La particolarità della seconda tipologia sta proprio nel fatto che anche se si utilizzano le componenti classiche di una rete di tipo deep, si può raggiungere lo stesso risultato utilizzando il drop out in fase di inferenza.

1.7.1 KL divergence

E' stato utilizzato Python come linguaggio di programmazione per implementare questo tipo di rete, precisamente si è fatto ricorso alla libreria open-source di Google, dedicata all'intelligenza artificiale, chiamata Tensorflow.

Questa libreria mette a disposizione molti strumenti per la creazione di modelli di machine learning e deep learning; inoltre, anche se ancora in fase beta, permette di accedere a metodi basati sui sistemi bayesiani. La libreria utilizzata per questo tipo di operazione si chiama *tensorflow probability*.

Per i layer di convoluzione si è utilizzato "conv2dFlipout", attraverso questi è possibile passare come argomento anche la distribuzione di probabilità dei pesi; per quanto riguarda invece i layer densi si è dovuto ricorrere a "DenseFlipout".

Di seguito è riportata la tabella con la struttura finale del modello, l'organizzazione può essere vista come la ripetizione di un insieme di blocchi che nella tabella sono numerati da uno a cinque.

Con la dicitura layer A e layer B si intendono layer di convoluzione uno consecutivo all'altro; il numero riportato indica la numerosità del filtro.

L'ultima colonna si riferisce invece allo stride nel layer di maxpooling che si trova alla fine di ogni elemento, la dimensione dello stesso è uguale per tutti i layer ed è 2x2 pixel.

Elemento	layer A	layer B	Stride
1	16	16	2
2	32	32	2
3	64	64	2
4	128	128	2
5	256	256	2

Tabella 1.1. Struttura modello bayesiano con KL divergence

Dopo l'elemento numero cinque è presente un layer detto "flatten" che permette di trasformare l'informazione in lineare, esso infatti andrà a rimuovere tutte le dimensioni tranne la prima.

L'output del layer flatten viene quindi passato ad un "DenseFlipout" layer da 128 e infine ad un altro dello stesso tipo da 3 che permette di discernere tra le tre classi.

1.7.2 Drop Out

Per l'Implementazione di questa tipologia di modello sono state utilizzate due API, oltre a Tensorflow, si è ricorso a Pytorch, libreria open-source utilizzata da Facebook e Twitter; anche in questo caso il linguaggio di programmazione è Python.

La struttura del modello drop out è più complessa, alla colonne della tabella precedente viene aggiunta una colonna che indica la percentuale di drop out.

Elemento	layer A	layer B	Stride	Drop-rate
1	16	16	2	0.15
2	32	32	2	0.25
3	64	64	2	0.25
4	128	128	1	0.25
5	256	256	-	0.3
6	1024	-	2	-

Tabella 1.2. Struttura della componente di convoluzione del modello drop out

Come per la rete precedente il layer successivo è un flattern che permette di rendere l'informazione lineare, in seguito sono presenti una serie di layer dense che per chiarezza sono riportati nella seguente tabella:

Elemento	Dense layer	Drop-rate
1	1024	0.35
2	364	0.25
3	256	-

Tabella 1.3. Struttura della componete lineare del modello drop out

In generale la funzione utilizzata è la Relu, rappresentata nell'immagine 1.1.

L'ultimo layer anche in questo caso è un dense layer di numerosità 3, la funzione di attivazione inoltre è la softmax.

1.8 Data augmentation

Quando si ha a che fare con sistemi di machine learning bisogna fronteggiare la difficoltà nel reperire dati.

Come si è già detto, la quantità di dati che si raccolgono al giorno d'oggi è molto elevata ma, spesso è difficile ottenerli in tempi ragionevoli; sono stati studiati, quindi, approcci che permettessero, partendo dai dati che si hanno a disposizione, di ottenere nuova informazione.

Il data augumentation, ci permette inoltre di evitare l'overfitting, quest'ultima è una problematica molto ricorrente nei sistemi ad apprendimento automatico, verrà discussa in un apposito paragrafo nel capitolo dedicato ai modelli Bayesiani.

Modificando a ogni iterazione le immagini che si sottopongono al modello in fase di training, il modello è spinto a imparare le vere features racchiuse nelle immagini. Le operazioni maggiormente utilizzate nel caso di data augumentation di immagini sono:

1. Shear range: un intervallo di valori entro il quale si desidera deformare le proprie immagini, come se si stesse applicando uno sforzo di taglio.

2. Zoom range: le immagini possono venir ingrandite con valori che vanno nel range designato.

3. Flip: possibilità di capovolgere l'immagine.

4. Brightness range: variazione della luminosità.

5. Rotate range: ruotare di un determinato angolo.

Sono caratterizzati tutti da intervalli perché a ogni iterazione i valori che verranno applicati alle immagini varieranno in quel determinato range in modo casuale.

Di solito, accompagna queste opzioni la possibilità di riempire gli spazi privi di informazione che si vengono a creare ad esempio dopo un'eventuale rotazione. Tra le alternative a disposizione c'è la possibilità di colmare le aree in questione, riflettendo l'immagine, oppure si può semplicemente copiare il pixel immediatamente prossimo alla zona priva di informazione.

Capitolo 2

Il Dataset

2.1 Informazioni generali

L'oggetto dell'analisi che verrà svolta nei prossimi capitoli è il tessuto epiteliale del colon. Esso è stato suddiviso in tre classi: adenoma, adenocarcinoma o tessuto sano.

Per la creazione del dataset sono state utilizzate immagini di tipo whole slide image (WSI), provenienti dall'Università di Leeds; quest'ultima dispone di un sito[1] dedicato alla virtual patology che offre circa 150 terabytes di materiale riguardante l'istologia.

In quanto algoritmi di DL di tipo supervisionato, come precedentemente accennato, necessitano di una grande quantità di dati opportunamente etichettati. Nel contesto dell'analisi automatica di WSIs, questo si traduce nell'individuare all'interno di queste ultime, delle regione in interesse (ROIs) omogenee in termini di tessuto.

La prima azione applicata sulle immagini è detta, quindi, "ROI cropping", si tratta di isolare le parti di tessuto che sono interessate dalle due tipologie di tumore e dunque di salvare questo nuovo file di più piccole dimensioni.

Questo processo verrà meglio approfondito nel capitolo seguente dedicato al datacleaning.

Le immagini che verranno usate per la creazione del dataset provengono da 27 pazienti estratti casualmente dalla banca dati, precisamente 9 per classe. L'operazione del "cropping" su un'immagine di grandi dimensioni porterà,

[1] Virtual Pathology at the University of Leeds

come si vedrà di seguito, ad avere associato ad ogni paziente un numero variabile di file whole slide.

2.1.1 Tumore colon-retto

L'apparato digerente ha come obiettivo quello di approvvigionare tutti gli altri sistemi dei nutrienti necessari alla vita. Questo compito viene svolto attraverso un particolare tipo di tessuto epiteliale che permette il passaggio delle molecole desiderate, e impedisce invece il trasbordo di elementi patogeni.

La struttura di questo apparato è cilindrica e nella parte inferiore risulta ripiegata più volte su se stessa. Nella sezione iniziale, prossima alla bocca, si trova la faringe, seguita da esofago, stomaco, intestino tenue, e infine intestino crasso.

Sezionando il condotto digestivo, è possibile notare l'avvicendarsi di quattro tonache concentriche:

1. mucosa: è lo strato più interno, esso quindi è a diretto contatto con ciò che viene ingerito, svolge la funzione di assorbimento e secrezione, e per aumentare la superficie atta a svolgere queste funzioni ha una struttura ripiegata;

2. sotto mucosa: avvolge lo strato elencato al punto precedente, è la sede dei vasi sanguigni e nervosi;

3. muscolare: è composto da muscolatura liscia, ha il duplice scopo di impedire il reflusso delle sostanze nutrienti e di permettere il movimento delle stesse durante la digestione;

4. sierosa: è lo strato più esterno ed è composto da tessuto connettivo;

Prendendo in analisi l'ultima parte dell'apparato digerente, l'intestino crasso, può essere diviso in tre parti: cieco, colon e infine retto.

La funzione di queste tre componenti è quella di andare ad estrarre la maggior parte dell'acqua che si trova in ciò che verrà successivamente espulso attraverso l'ano.

Con il termine tumore al colon-retto si fa riferimento, alla lesione che colpisce le cellule della mucosa della zona in questione. Questa tipologia è una delle più comuni in occidente, ciò si può ricondurre allo stile di vita e alla dieta.

A livello mondiale la suddetta tipologia di neoplasia è la quarta più diffusa dopo le patologie che attaccano il polmone, fegato e stomaco. Osservando soltanto l'universo femminile, questa patologia si colloca al terzo posto dopo il tumore al seno e al polmone [4].

Esaminando in particolare l'Europa si nota come questo tipo di patologia sia molto ricorrente, infatti rappresenta la seconda tipologia di neoplasia più diffusa, precisamente la terza per gli uomini dopo quella che colpisce la prostata e polmone, invece per le donne la seconda dopo il tumore al seno [5].

La sopravvivenza a 5 anni è in continuo aumento, dato che proporzionalmente è aumentato lo screening sulle persone maggiormente interessate da questa problematica, cioè la popolazione con età compresa tra i 50 e i 75 anni.

Il primo stadio di questo tipo di controllo si compie attraverso la palpazione dell'intestino, alla ricerca di agglomerati sospetti. Successivamente se il medico trova delle zone sospette si può intervenire attraverso diverse tecniche tra le quali l'ecografia all'addome oppure la Tomografia Computerizzata con contrasto, in questo modo si potrà valutare l'estensione della lesione e la presenza di metastasi.

La colonscopia è la tecnica specifica per lo studio di questo tipo di problematiche, attraverso una fibra ottica è possibile illuminare all'interno del delle cavità dell'apparato digestivo e grazie a una videocamera visualizzare la mucosa.

Esiste inoltre la possibilità si collegare, alla classica strumentazione in uso per la colonscopia un ulteriore elemento che consente di estrarre dei campioni di tessuto, questi permetteranno di compiere analisi istologiche e quindi di avere una diagnosi certa [12].

Le analisi istologiche sono quindi compiute attraverso l'utilizzo del microscopio da parte di personale specializzato.

2.1.2 Adenoma e Adenocarcinoma

Con il termine neoplasia si fa riferimento a un insieme di cellule che mostrano delle caratteristiche diverse dalla controparte sana. Queste presentano una spiccata capacità riduttiva e inoltre non rispettano il normale ciclo di vita di una cellula.

Per etichettare una cellula come neoplastica, normalmente, un esperto si basa sullo studio della forma e del colore. A seguito dell'analisi di tali elementi potrà essere fatta una diagnosi.

L'origine dei tumori può essere ricondotta a mutazioni che avvengono a livello del DNA, le quali sommandosi nel tempo possono portare alla generazione di neoplasie.

Le mutazioni però sono processi naturali, non sempre infatti si arriva alla generazione di lesioni. Sono inoltre presenti particolari processi di protezione che l'organismo può attuare per l'individuazione delle mutazioni e la conseguente riparazione.

I tumori in generale si possono dividere in due categorie: benigni e maligni.

I primi si caratterizzano per il basso potere infiltrante e la bassa capacità moltiplicativa, i tumori maligni, invece, hanno elevate proprietà di infiltrazione e possono creare metastasi facilmente.

Con il temine infiltrante si intende la capacità della neoplasia di interessare tessuti nelle vicinanze del punto di origine. Le metastasi invece permettono al tumore di spostarsi attraverso i vettori presenti nel corpo umano, come ad esempio il sangue, e di arrivare anche in tessuti molto lontani dal focolaio iniziale.

Il termine "adenas" in greco significa ghiandola, da esso deriva parte del nome sia dei tumori benigni che di quelli maligni del sistema esocrino.

L'adenoma è una lesione di tipo benigno che può originarsi da cellule epiteliali di organi ghiandolari esocrini oppure da cellule epiteliali di tessuti con proprietà secretorie come ad esempio la mucosa del colon o dell'esofago.

Con il termine carcinoma si fa riferimento a qualsiasi tipo di tumore di tipo maligno che si è originato da tessuto epiteliale. Con il temine adenocarcinoma quindi ci si riferisce a una lesione maligna, precisamente al tessuto epiteliale con proprietà secretorie.

Le due categorie di neoplasie non sono però due mondi a sé stanti, infatti alle volte, gli adenomi possono trasformarsi in adenocarcinomi.

In entrambi i casi per avere una diagnosi sicura è necessario andare a studiare i tessuti interessati attraverso la biopsia e quindi lo studio al microscopio degli stessi.

2.1.3 Whole slide images

La digital pathology basa le proprie fondamenta negli anni sessanta. Durante quel decennio si ebbero le prime acquisizioni digitalizzate.

Per uno sviluppo più completo di questa pratica si è dovuto attendere gli anni duemila, infatti, in un primo momento la strumentazione era di difficile reperibilità o troppo costosa.

La radiologia è stato l'ambito che ha subito uno sviluppo più accentuato, questo è accaduto grazie a diversi fattori.

Il primo sicuramente è da imputare al numero maggiore di acquisizioni che si fanno rispetto all'istologia, infatti quando possibile si cerca di evitare una biopsia. Un altro motivo che ha favorito lo sviluppo della radiologia è lo spazio di archiviazione. In media un'acquisizione radiologica ha una dimensione quattro volte inferiore ad una atta alla virtual patology, come ad esempio un'immagine istologica.

Con il termine whole slide images si fa riferimento a una particolare tipologia di file che è sempre più utilizzata nell'ambito medico e della ricerca grazie alla capacità di inglobare quantità considerevoli di informazioni, in strutture ben organizzate e di dimensioni meno importanti.

Questi file vengono generati grazie all'utilizzo di particolari microscopi ad alta risoluzione, le immagini di piccole dimensioni che derivano dalle singole acquisizioni fatte con questo strumento vengono poi sommate, fino ad arrivare a generare un singolo file di grandi dimensioni che contiene tutte le acquisizioni fatte.

Possono inoltre essere raccolte nello stesso file immagini a diversi fattori di magnificazione.

Alla fine del procedimento di acquisizione si avranno delle immagini con struttura "piramidale", che nella modalità non compressa avrebbero potuto occupare anche una quantità di memoria dell'ordine della decina di Giga byte.

La struttura è detta piramidale perché nello stesso file sono presenti immagini a risoluzione diversa a seconda della "profondità" a cui si ispeziona lo stesso.

Di solito si possono avere tre o più livelli e il layer zero è quello caratterizzato dalla risoluzione maggiore.

In aggiunta alle immagini, in questi file è conservata anche una grande quantità di meta-dati, come ad esempio il tempo di esposizione o la grandezza di acquisizione delle immagini dal microscopio che potranno essere poi utilizzati in fase di analisi.

I vantaggi nell'utilizzare questo tipo di file sono:

1. trasportabilità: essendo i file altamente efficienti in quanto ad archiviazione, sono facili da condividere, questo aspetto è molto importante soprattutto nel campo scientifico;

2. facilità di archiviazione: non sono necessarie strutture particolari per poterli conservare;

3. possibilità di fare comparazioni: avendo le immagini in formato digitale è facile poter confrontare file diversi allo stesso tempo, molto difficile invece è fare questo tipo di analisi ricorrendo direttamente al microscopio;

4. isolare le zone di interesse: grazie ad opportuni software, andare ad annotare direttamente sull'immagine aree di studio utili per una classificazione o per successivi approfondimenti è molto facile.

Un'altra peculiarità deriva dal concetto base della digital patology, infatti attraverso questo tipo di file si possono sviluppare software con intelligenza artificiale che aiutano ad ottenere una diagnosi in tempi brevi, vedasi il caso di classificatori di immagini basati su sistemi ad apprendimento automatico.

2.1.4 Openslide

Per lavorare con questo particolare tipo di file si è utilizzato Openslide, una libreria di Python, scritta in linguaggio C.

Questa libreria permette di lavorare con diversi formati di file di tipo WSI, tra questi anche il formato Aperio, contrassegnato dalla dicitura ".svs". Le singole immagini immagazzinate in questo particolare formato sono di tipo tiff, altri formati di file WSI permettono di utilizzare codifiche di compressioni ancora più pesanti come ad esempio jpeg, questo però si ripercuote sull'immagine con la conseguente perdita di informazioni.

Una delle funzioni principali di open slide è quello di poter andare a selezionare un particolare livello e da esso ottenere un'immagine della dimensione che si desidera in una particolare posizione.

Questa funzione è alla base degli algoritmi sviluppati per ottenere il dataset, infatti per poter lavorare su questi file il primo step che compiuto è stato quello di andare a dividere l'immagine a risoluzione maggiore, ciò del livello 0, in piccole tiles.

E' possibile inoltre creare le tiles per un determinato livello con overlap, cioè si può spostare la finestra che genera l'immagine di un determinato numero di pixel che è inferiore alla grandezza dell'immagine stessa, in questo modo si possono avere in immagini diverse, parti di tessuto uguale.

Nella libreria open slide inoltre sono presenti anche altri metodi che permettono di interfacciare i file svs con script in linguaggio java, ciò è molto utile per poter andare a creare dei visualizzatori con risoluzione adattiva al livello di profondità a cui si sta svolgendo l'analisi, questo strumento è stato utilizzato nel tool di analisi per permettere una visualizzazione a massima risoluzione del file selezionato.

2.2 Creazione dei dataset

Come detto nell'introduzione i pazienti analizzati sono 27, ad ognuno di essi è stato associato un numero. Il singolo soggetto, inoltre, è caratterizzato da un numero variabile di file svs.

Classe	Files svs
AC	23
H	61
AD	27

Tabella 2.1. Numerosità file svs per classe.

Ogni singolo file svs, ha una risoluzione variabile, anche per ciò che attiene a file associati allo stesso paziente.

Per quanto riguarda l'estrazione delle tiles si è lavorato sempre sul livello a massima risoluzione, ritagliando l'immagine con una finestra di 256x256 pixel senza overlap.

Le dimensioni delle tiles sono state scelte per due motivi: garantire che nel singolo crop sia contenuta l'informazione necessaria all'apprendimento; migliorare le prestazioni in quanto a velocità di creazione, infatti la libreria *Openslide* permette di ottenere prestazioni migliori se si sceglie una dimensione della finestra che rispetta una specifica equazione.

Le immagini così ottenute non sempre erano della dimensione desiderata, dato che la risoluzione dell'immagine di partenza spesso non era proporzionata; è stato necessario quindi andare a selezionare solo le tiles che rispettassero questa proporzionalità.

Classe	Tiles estratte	Tiles 256x256
AC	41908	40151
H	30205	27933
AD	32985	31122

Tabella 2.2. Numerosità immagini estratte per classe.

E' stato inoltre creato un ulteriore dataset rimpicciolendo la risoluzione da 256 a 64, in questo modo si riesce a ridurre il carico computazionale generato dal dataset. In seguito verrà fatta una comparazione delle prestazioni dei sistemi di deep learning sui set a diversa risoluzione.

Il dataset iniziale è stato suddiviso in tre sotto gruppi:

1. training set: racchiude le informazioni attraverso le quali il sistema ad apprendimento automatico potrà imparare;
2. validation set: composto da elementi non presenti nel training set, permetterà di andare a valutare in fase di training le prestazioni del sistema, e sarà utile per la variazione degli hyperparameters;
3. test set: verrà utilizzato nella fase finale per valutare se la rete è in grado di generalizzare o meno, anche questo deve essere composto da elementi che in fase di training il sistema non ha ricevuto in input.

Questa suddivisione è comunemente utilizzata per l'organizzazione dei dati quando si utilizza il machine learning.

I tre set inoltre sono stati creati partendo da pazienti diversi, facendo in modo che un soggetto potesse risultare in uno solo dei set.

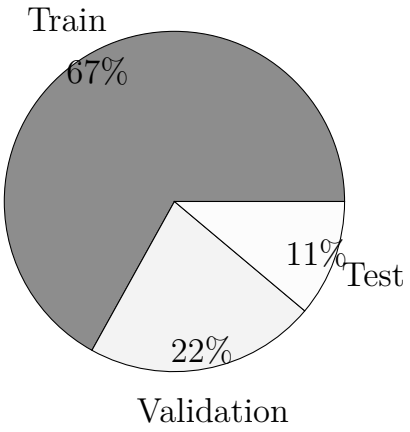


Figura 2.1. Suddivisione dei pazienti nei tre dataset.

Per la creazione definitiva dei dataset si è andato a selezionare quale fosse la classe meno numerosa all'interno dello specifico set. Così facendo si è scelta la soglia minima di immagini per classe in modo da avere le classi equamente bilanciate.

In conclusione nella tabella 2.3 è possibile consultare la dimensione finale del dataset.

Set	Classe	Totale
Train	15000	45000
Val	6400	19200
Test	2700	8100

Tabella 2.3. Dataset finale.

Capitolo 3

Metodi

3.1 KL method

Le reti di tipo bayesiano hanno sì il grande vantaggio di poter calcolare l'incertezza sulla predizione, ma hanno anche uno svantaggio ovvero il peso computazionale. Anche disponendo di hardware molto prestante, infatti, la computazione da svolgere è molto elevata.

Per velocizzare la fase di training, cioè per ottenere tempi di training ragionevoli, Google ha creato dei metodi che permettono alla rete di tipo deep di essere più veloce.

La prima tecnica che si può utilizzare è quella delle "pipelines", attraverso le quali si riesce a ridurre il collo di bottiglia che è lo scambio dati tra componenti del computer.

Un altro modo per rendere le operazioni più veloci è quello di organizzare i dati in particolari strutture che permettono una lettura molto veloce; questa opzione è approfondita nel paragrafo seguente dei TFRecords.

3.1.1 Tensorflow piplines

Nel modulo *tf.data*, tensorflow mette a disposizioni dei metodi che permettono di gestire in maniera intelligente lo scambio dati che avviene in fase di training tra CPU e GPU all'interno del computer su cui si sta svolgendo l'analisi.

Tipicamente la fase di training è composta dall'alternanza di due fasi:

1. caricamento dati: presa in carico dalla central processing unit, una volta conclusasi, i dati vengono inviati all'altra componente interessata;

2. training: questo processo è delegato alla GPU che attraverso il grande numero di core riesce a svolgerlo velocemente.

Questi due processi, anche nel caso di mini-batch, si alternano fino al termine dei dati in input.

Il problema sorge nel momento in cui la GPU finisce il suo compito e la CPU inizia a processare nuovi dati, si creano infatti delle fasi di inattività delle due componenti.

Per evitare queste fasi di inattività si possono utilizzare le pipelines, che permetto di gestire i due processi sopra enumerati in maniera asincrona, in questo modo i tempi di training diminuiscono drasticamente.

E' necessario quindi creare una nuova struttura chiamata "Iteratore" consente lo scambio dati tra i due processi garantendo robustezza al sistema.

In ogni caso alle volte la fase di caricamento dei dati può rappresentare il collo di bottiglia al training.

3.1.2 TFRecods

I TFRecords sono un particolare tipo di file che rispetta dei protocolli di serializzazione, essi sono dei file di tipo binario.

Attraverso queste particolari strutture è possibile velocizzare la lettura dei dati da disco, soprattutto nei casi in cui si sta facendo stream di dati online o si utilizzano vecchi tipi di dispositivi di storage. Si riesce quindi a superare il limite che si riscontrava, anche se in maniera minore, con le pipelines.

Tipicamente questi file hanno una grandezza di circa 300MB, nel caso in cui il dataset abbia una dimensione maggiore, si possono creare più TFRecord, che poi in fase di training verranno passati tutti al modello attraverso un'apposita funzione.

C'è però anche un altro tipo di vantaggio nell'utilizzo dei TFRecords, essi permettono al sistema di non dover caricare in memoria tutti gli elementi che lo compongono ma, rispettando il concetto di batch, caricheranno in memoria un batch per volta, riducendo al minimo anche il consumo di memoria.

Le molte peculiarità di questo tipo di file sono accompagnate però da aspetti negativi:

1. conversione del dataset: è necessario convertire tutto il dataset a propria disposizione in questo nuovo tipo di file. Il processo ha una durata abbastanza elevata, nel caso del dataset che si è precedentemente utilizzato, infatti, sono state necessarie circa 10 ore;

2. carenza di documentazione: essendo una componente di tensorflow relativamente nuova, la documentazione che si può trovare online è abbastanza carente o di difficile comprensione; la comprensione del funzionamento di questa tecnica di bufferizzazione quindi non risulta sempre chiara e lineare.

3.2 Drop-out method

Per l'implementazione di questo modello è stato utilizzato libreria open source per l'apprendimento automatico e le reti neurali, scritta in Python: *Keras*.

Essa si basa su tensorflow, ma ambisce a un livello di astrazione più alto, infatti anche se le funzionalità nella maggior parte dei casi sono condivise, con Keras è possibile farlo molto più velocemente.

Negli ultimi anni questa libreria sta assumendo un'importanza sempre maggiore, questo è stato confermato dalla release di tensorlow 2.0 rilasciata pochi mesi fa.

3.2.1 Keras

Keras raccoglie delle Application Programming Interface (API) che appartengono alla libreria Tensorflow. Da quest'ultima eredita gran parte delle funzioni, tuttavia se ne differenzia poiché rende l'esperienza di programmazione più immediata.

Un esempio può essere il caricamento dei dati da passare in input ad una rete che può essere svolto con l'utilizzo di solo due funzioni:

1. flow_from_directory: permette di leggere i file presente in una directory e indicizzarli.

2. imagedatageneretor: genera un "iteretor" che permette di passare in input i dati al modello. Attraverso la suddetta funzione è possibile anche attivare il data augmentation aggiungendo poche righe di codice.

E', inoltre, presente un set di funzioni obiettivo tra le quali scegliere. Si riduce, dunque, anche il tempo di programmazione per questa parte che comunemente non è immediata.

Altra peculiarità la si può ritrovare nelle "callback". Sono infatti presenti delle funzioni che permettono di gestire operazioni che devono avvenire durante il training del modello, come ad esempio l'arresto anticipato che

permette di evitare l'overfitting o il salvataggio automatico dei pesi nel caso in cui siano presenti dei miglioramenti della rete.

C'è, in aggiunta, l'opportunità di scrivere delle "callback" personalizzate. Questa opzione è molto importante se si intende gestire un processo durante il training, è infatti stata utilizzata nell'applicativo di datacleaning per la visualizzazione in tempo reale dell'andamento della funzione di costo e dell'accuratezza.

3.2.2 PyTorch

PyTorch è una libreria open source che raggruppa vari metodi riguardanti il machine learning ed anche il deep learning.

Inizialmente il framework è stato creato da tre ricercatori, successivamente dato il grande interesse generato nei più svariati ambiti, anche al di fuori della ricerca, è stato rilevato da Facebook che tutt'oggi lo impiega nelle proprie analisi.

Questa libreria negli anni si è fatta sempre più strada, fino a diventare quasi uno standard per la creazione di modelli di deep learning.

I vantaggi di quest'ultima sono molteplici, essa infatti è costruita per permettere di compiere calcoli di tipo algebrico in maniera pratica e veloce, risulta un ottimo sostituto della libreria "Numpy", con il plus che i calcoli sono svolti dalle GPU.

Da questa caratteristica ne consegue che gli algoritmi creati utilizzando questa libreria riescono a svolgere le operazioni molto più rapidamente. In ogni caso, a differenza di Tensorflow, la visualizzazione o la modifica di matrici o di vettori risulta molto veloce e facile, dato che Numpy e PyTorch hanno una elevata compatibilità.

A differenza di altre librerie dedicate all'apprendimento automatico, quella di analisi in questo paragrafo è stata concepita principalmente per la programmazione in linguaggio Python.

La maggior parte dei metodi utili alla creazione di un modello di machine learning o di deep learning si possono trovare nella classe "torch.nn".

PyTorch mette, a disposizione in particolare una serie di metodi atti allo studio di immagini, come ad esempio "Imageloader" che una volta indicato un percorso all'interno delle cartelle, associa automaticamente le immagini nelle varie sottocartelle alla rispettiva classe.

Sono presenti inoltre altre "utilities" che permettono di suddividere il dataset, una volta caricato, in batch di una dimensione a piacere che dipenderà

dalle prestazioni dell'hardware a disposizione; con la possibilità ad ogni epoca di rimescolare questi gruppi di immagini.

Come per la controparte in Tensorflow, sono disponibili delle funzioni loss e degli algoritmi di ottimizzazione già implementati, che quindi attraverso la scrittura di poche righe di codice sono permesse la loro implementazione.

Capitolo 4

Risultati

Una volta terminata la creazione del dataset e la definizione dei modelli si è passati alla valutazione delle prestazioni degli stessi in termini di accuratezza.

Le prove sono state eseguire su "Google Colab", un apposito sito sviluppato dall'omonima azienda per scopi di ricerca, che garantisce entro determinati limiti di tempo, di eseguire codice Python su hardware molto prestante.

Per quanto riguarda il reparto GPU Colab mette a disposizione delle schede grafiche Nvidia TESLA K80, con 24 GB di memoria RAM GDDR5 e 4992 Cuda core; queste specifiche sono molto importanti per sviluppare reti di deep learning in breve tempo.

Si è, inoltre, fatto ricorso al sistema di super calcolo dell'HPC Polito, esso mette a disposizione una quantità elevata di potenza di calcolo che viene normalmente utilizzata da ricercatori.

Il sistema è composto da tre cluster: Casper, Hactar e Legion.

Quest'ultimo dispone di hardware molto recente soprattutto per quanto riguarda il comparto GPU, infatti sono installate le Nvidia Tesla V100 con 32 GB di memoria ram e 5120 cuda cores.

Prima di procedere è lecito fare delle considerazioni sui set di dati a disposizione:

1. velocità di training: tra i due dataset a risoluzione differente si è infatti notato una grande differenza in quanto a rapidità nel portare a termine il training. Il set con risoluzione 64x64 è risultato nettamente più veloce;

2. accuratezza: non sono risultate differenze sostanziali tra i set di immagini in quanto ad accuratezza, in generale il set a 256x256 aveva prestazioni leggermente inferiori a quello con risoluzione minore.

Per le considerazioni appena fatte si è continuato lo studio utilizzando solo il dataset a risoluzione minore cioè 64x64 pixel. Utilizzare immagini a più alta densità di pixel, infatti, non ha portato ad alcun beneficio.

E' importante evidenziare che nelle tabelle riportate di seguito il valore di accuratezza sul test set è stato calcolato classificando l'intero set cinque volte e poi calcolando la media dei valori di accuratezza.

4.1 Prestazioni KL

Le prestazioni del modello generato sono valutate in accuratezza e sono riportate nella tabella 4.1

Set	Accuratezza
Train	83,5%
Validation	83,5%
Test	71,4%

Tabella 4.1. Valori accuratezza sui dataset del modello KL

Come è intuibile dai valori di accuratezza si nota che il sistema sta apprendendo senza che sia presente overfitting; una volta raggiunto questo picco di prestazioni sul train e sul validation, però, si denota un rapido peggioramento delle stesse.

Il problema riscontrato nell'utilizzo di questa tecnica è che anche se il training viene compiuto abbastanza velocemente, sono presenti molti problemi di compatibilità con altre funzioni di tensorflow o keras che poi saranno indispensabili per continuare l'analisi.

Si è quindi provato ad aggiornare il modello utilizzando maggiormente metodi appartenenti a keras, quando presenti.

Nel nuovo modello sono stati riscontrati due problematiche importanti:

1. tempi computazionali: questo può essere imputato al non utilizzo delle pipelines e dei TFRecords, non essendo compatibili pienamente con keras, che evidentemente sotto questo punto di vista avevano una grande rilevanza;

2. convergenza del sistema: con le nuove funzioni è risultato ulteriormente più difficile arrivare alla convergenza del sistema.

Da queste problematiche è nata la scelta di continuare lo studio sulle reti bayesiane soltanto con l'utilizzo del modello con drop out in quanto garantisce tempi di esecuzione ragionevoli e mostra una maggiore probabilità di convergere verso un risultato.

4.2 Prestazioni drop out

Anche in questo caso si ricorre alla tabella 4.2 per mostrare le prestazioni della rete allenata attraverso il modello con Drop-out.

Set	Accuratezza
Train	86,3%
Validation	73,23%
Test	68,1%

Tabella 4.2. Valori accuratezza sul dataset del modello con drop out.

Osservando che i valori del validation set e del test set risultano abbastanza distaccati da quelli del training, si può notare che il modello tende all'overfitting.

Si è provato, quindi, ad utilizzare il data augmentation per ottenere un miglioramento.

Di seguito sono riportati i valori utilizzati per applicare il data agumetnation:

Tecnica	Valore
Shear range	0,2
Zoom range	0,2
Brightnessrange	0,5-1
Flipout	True
Fill mode	Nearest

Tabella 4.3. Parametri utilizzati per data augmentation.

Le modifiche sono state apportate in modalità "online", cioè i dati presenti nel dataset non hanno subito alterazioni, ma le immagini sono state modificate soltanto una volta lette.

I risultati ottenuti da questo ulteriore sistema sono riportati nella tabella 4.4.

Set	Accuratezza
Train	84,3%
Validation	71,6%
Test	65,5%

Tabella 4.4. Valori accuratezza sul dataset del modello con Drop-Out, con attivo data augmentation.

Si nota che le prestazioni tendenzialmente sono in linea con il caso precedente, inoltre l'accuratezza del training set è inferiore rispetto al modello inizialmente indicato. Questo è da imputare principalmente all'attivazione del data augmentation che ad ogni iterazione somministra immagini diverse in input.

4.3 Considerazione su incertezza

In questo paragrafo si andranno ad analizzare le prime considerazioni che è possibile fare grazie alle reti bayesiane.

Partendo dal modello con drop out si è classificato il test set imponendo un Monte Carlo samlpe di 50. Si sarebbe potuto seguire lo stesso procedimento anche con la rete che utilizza la KL.

Grazie all'applicativo illustrato successivamente, verrà generato un dizionario in cui ad ogni immagine sarà associato un valore di incertezza.

Utilizzando un istogramma è possibile mostrare l'incertezza totale delle tiles che sono state classificate erroneamente all'interno del test set.

Dall'immagine si denota che è presente una correlazione tra predizione sbagliata e alta incertezza. Nel grafico infatti è presente un picco intorno allo 0,5 mentre tendenzialmente valori bassi di incertezza sono associati a predizioni corrette.

Per comprendere meglio cosa fosse considerato incerto per la rete, sono state selezionate tutte le immagini con un'incertezza totale superiore allo 0,6. Le stesse, quindi, sono state rappresentate nella figura 4.2.

Risulta evidente che nelle immagini più incerte sono presenti:

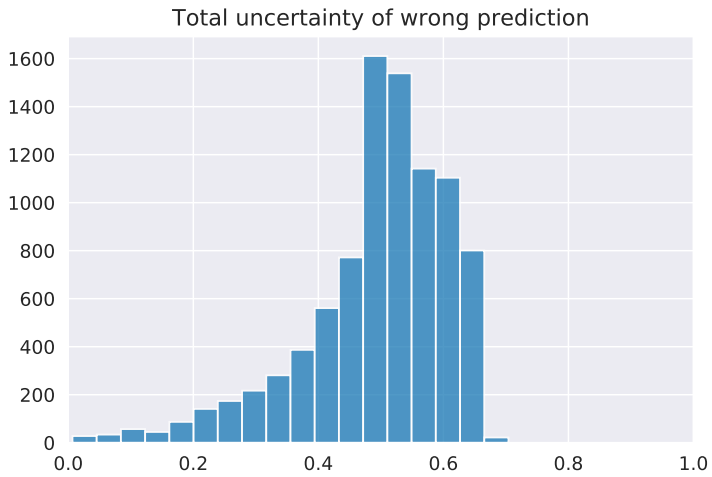
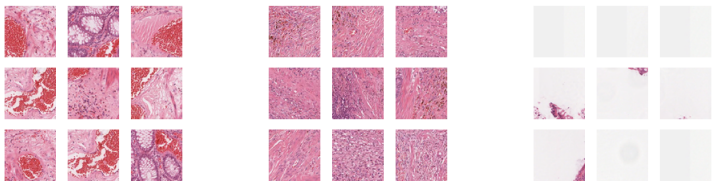


Figura 4.1. Valori di incertezza totale delle tiles con predizione errata

1. vasi sanguigni: caratterizzati dal colore rosso vivo, essi rappresentano fonte di errore perché, anche se presenti in percentuali differenti, sono comuni a tutti i tessuti;

2. sezioni con assenza di tessuto: non è detto che il tessuto ricopra l'intera superficie del vetrino, queste zone bianche sono ancora una volta presenti in tutti i tessuti in analisi;

3. stroma: particolare tipo di tessuto connettivo con funzioni strutturali, è principalmente composto da cartilagine che può essere anche organizzata in fasci.



(a) (b) (c)

Figura 4.2. Fig.(a) Tessuto colon-retto con presenza di vasi sanguigni, Fig.(b) Stroma presente in tessuto colon-retto, Fig.(c) Crop raffiguranti maggiormente vetro.

4.4 ROI cropping

Requisito fondamentale per ottenere un buon modello è quella di avere a disposizione un dataset di qualità.

Spesso, infatti, i dati possono essere:

1. spuri o rumorosi: si possono individuare in essi errori e outliers;
2. inconsistenti: non c'è coerenza tra i dati presenti nello stesso dataset. Un esempio potrebbe essere l'eventualità in cui si siano utilizzate scale diverse per una stessa feature;
3. incompleti, nel caso in cui ad un elemento sono assegnati più valori, alcuni di questi non sono presenti.

La presenza di queste problematiche all'interno di un insieme di dati porta conseguentemente a ottenere pessime prestazioni durante l'apprendimento di una rete neurale.

Per quanto riguarda l'ambito delle immagini, nella maggior parte dei casi la procedura di assegnazione a una determinata classe è delegata a un esperto. Quest'ultimo attraverso la propria conoscenza, suddivide le immagini tra le classi in analisi.

Nel caso delle immagini di tipo WSI, la procedura da seguire per la creazione del dataset è detta "ROI cropping", l'esperto individua le zone di tessuto affette da tumore e le ritaglia.

Le immagini così ottenute saranno quindi assegnate alla rispettiva categoria.

Image scope è uno strumento che fa fronte a questa necessità, attraverso esso, il medico potrà generare dei file svs di più piccola dimensione rispetto al file iniziale che tuttavia contengono per la maggior parte una sola classe.

Nel ritaglio di forma rettangolare che si ottiene, delle volte sono presenti anche delle componenti non desiderate. E', infatti, raro trovarsi in presenza di aree di interesse che combacino in maniera esatta alla zona di ritaglio.

Dal processo appena descritto si origina la problematica del rumore all'interno del dataset.

Nasce quindi il bisogno di intervenire su questa problematica in modo tale da rimuovere ciò che non corrisponde alla reale classe in cui è stato categorizzato.

Tipicamente durante il "Knowlege discovery", il processo di raffinamento dei dati è molto importante, spesso, viene considerato quello sul quale si concentrano più risorse.

Per ovviare a questo problema, nell'ambito delle image processing, si utilizzano delle tecniche automatiche che permettono di calcolare alcuni parametri propri delle immagini. I suddetti valori, per immagini appartenenti alla stessa classe, si può ipotizzare che rimangano uguali. L'individuazione degli outliers si basa sul calcolo di parametri come la media e varianza.

Altra tipologia di tecnica utilizzata per la rimozione del rumore è il clustering. Si va quindi a dividere il set di elementi in sottogruppi in uno spazio multidimensionale, gli elementi che hanno una distanza elevata dal sottogruppo di appartenenza sono ritenuti errori.

Queste tecniche però non tengono conto delle particolari strutture che possono trovarsi all'interno delle immagini comunemente chiamate "patterns".

L'alternativa a questo tipo di pratica è far visionare le immagini ad un esperto il quale, attraverso la sua esperienza, deve indicare quali siano le immagini che rappresentano rumore.

Nella seguente paragrafo si esporrà come è possibile fare datacleaning su un insieme di immagini attraverso l'informazione che si ottiene da un sistema bayesiano; saranno inoltre confrontate le prestazioni e i tempi di esecuzione rispetto alla pratica manuale.

4.5 Data cleaning manuale

Nel caso in cui non si volesse ricorrere a una tecnica di cleaning automatica si può ricorrere al data cleaning manuale.

Per svolgere questa operazione è necessaria quindi una cospicua quantità di tempo da parte di personale specializzato in quanto devono essere visualizzate tutte le immagini presenti nel dataset. E' necessario, quindi, controllare l'effettiva corrispondenza con l'opportuna classe.

Per ottenere una comparazione tra la tecnica automatica e quella manuale, il processo di selezione è stato svolto sul dataset precedentemente creato.

Successivamente sono state valutate le prestazioni che è possibile ottenere con il nuovo set di immagini.

Per svolgere questo processo è stato necessario l'intervento di un esperto che, attraverso la propria esperienza, ha individuato le immagini del dataset che contenevano più rumore in modo da poterle eliminare.

4.5.1 Analisi dataset

Dallo studio di tutti i file svs è emerso che il paziente 16 è quello caratterizzato da più rumore, esso presenta infatti un tumore non ancora differenziato.

In questa immagine il tessuto sano e quello malato si alternano in maniera non netta, inoltre si può notare la presenza massiccia di stroma.

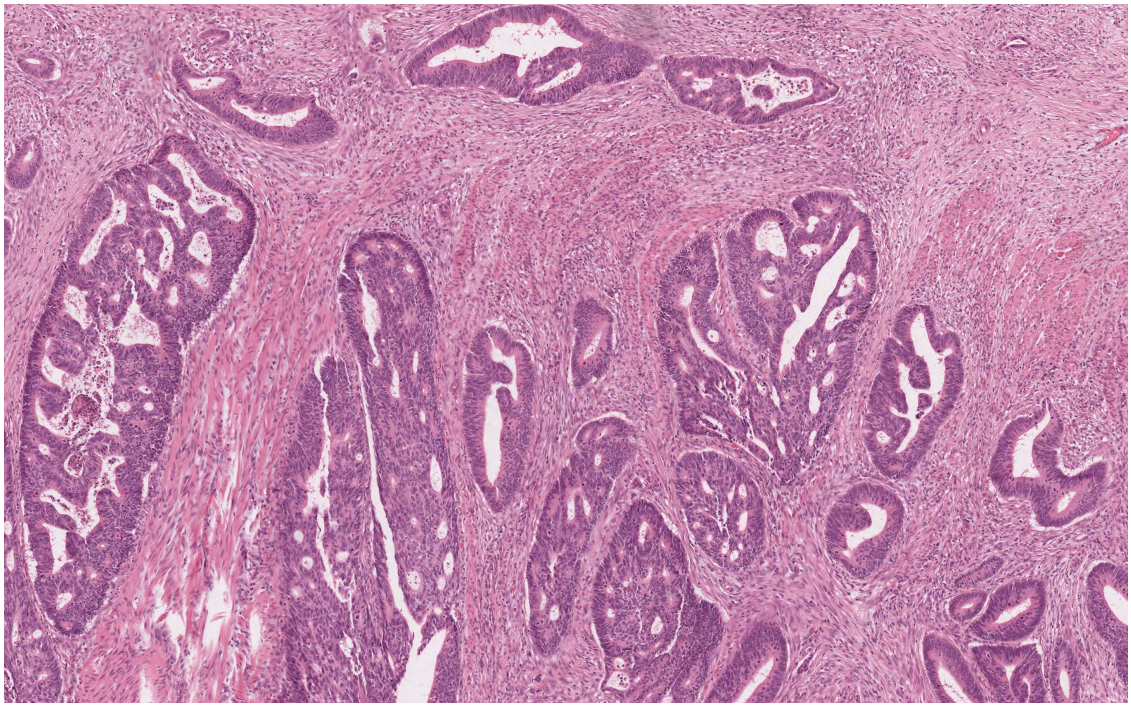


Figura 4.3. Paziente 16, file numero 2.

E' stato quindi necessario rimuovere completamente il paziente 16, appartenente alla classe adenocarcinoma che, come mostrato dalla tabella 2.2, è la classe che presenta una maggiore numerosità di immagini estratte.

Oltre al paziente 16, sono state rimosse dal dataset iniziale anche due file svs del paziente 40 e due del paziente 43.

Esse infatti anche se mostravano in maniera minore la presenza di tessuti non correlati alla classe di appartenenza degli stessi, risultavano comunque per la maggior parte compromesse.

La durata di analisi delle immagini è stata di circa un giorno. E' stato inoltre necessario andare a ricreare le tiles con la conseguente spesa in termini di tempo.

4.5.2 Prestazioni drop out

Sono state valutate le prestazioni di un nuovo modello con drop out, allenato con il dataset ottenuto dopo la selezione da parte dell'esperto.

Set	Accuratezza
Train	87,3%
Validation	85,4%
Test	74,4%

Tabella 4.5. Valori accuratezza sul dataset del modello con Drop-Out, dopo il data cleaning manuale.

Si denota un evidente aumento delle prestazioni per quanto riguarda tutti e tre i set di dati, maggiormente per quanto riguarda il test set.

Per completezza è stato, inoltre, allenato un ulteriore modello però con il data augmentation attivo.

Set	Accuratezza
Train	79,7%
Validation	77,1%
Test	76,1%

Tabella 4.6. Valori accuratezza sui dataset del modello con Drop-Out con attivo data agumetation, dopo il datacleaning manuale.

E' evidente come il data cleaning e il data agumetation abbiano portato la rete ad ottenere ottimi risultati in quanto a generalizzazione, evitando l'overfitting.

4.6 Data cleaning automatico

In questo paragrafo verrà esplicato come è stato possibile utilizzare un sistema bayesiano per fare datacleaning in maniera automatica. Inoltre sarà possibile comprendere se questa tecnica può fornire risultati simili alla controparte manuale.

Il primo step è stato allenare un modello bayesiano sul dataset che si ha a disposizione, questo verrà quindi utilizzato per classificare le immagini presenti all'interno del training set o del validation imponendo un Monte Carlo samples di 50.

Per avere un metro di paragone utile con le prestazioni delle reti precedentemente allenate con la completezza del dataset, il test set è rimasto inalterato.

Per ogni immagine sono state ricavate 50 classificazioni da modelli tendenzialmente diversi. Sula scia dei risultati ottenuti è stato possibile fare statistica estraendo i valori di incertezza aleatoria e epistemica, come riportato nell'articolo "Uncertainty quantification using Bayesian neural networks in classification: Application to ischemic stroke lesion segmentation".

Per l'implementazione è stata utilizzata la libreria di Python *Numpy*.

Le formule di seguito riportate fanno riferimento a una singola immagine, ipotizzando di inserire in un vettore i risultati delle 50 classificazioni, in questo caso quindi n = 50.

$$epistemic = \frac{1}{n}\sum_{i=0}^{n} x_i^2 - (\frac{1}{n}\sum_{i=0}^{n} x_i)^2$$

$$aleatoric = \frac{1}{n}(\sum_{i=0}^{n} x_i(1 - \sum_{i=0}^{n} x_i))$$

Al termine della classificazione del set di immagini si otterrà un file di testo contenente un json che segue le struttura in figura 5.12.

Partendo da questo file si è potuto rappresentare attraverso degli istogrammi i valori di incertezza delle tiles (figura 4.4).

Osservando gli istogrammi dell'incertezza totale di vari dataset è possibile notare un'importante caratteristica: sono bimodali. E' quindi possibile discernere tra tiles certe e incerte anche solo osservando questo tipo di grafico.

Partendo da questa proprietà, si è andato quindi a sviluppare un algoritmo per l'individuazione della soglia ideale di separazione tra le classi.

Sono state individuate quindi due metodologie, la prima basata sul metodo di Otsu, la seconda cerca di trovare un giusto compromesso tra rigore teorico e numerosità del dataset.

Nel software di datacleaning la scelta della soglia finale è lasciata comunque all'utente che, a seconda delle condizioni in cui si trova, può ritenere più consono l'utilizzo di una o dell'altra.

4.6.1 Otsu

Per l'individuazione della soglia si è fatto ricorso al metodo di Otsu, comunemente usato nella sogliatura automatica delle immagini, attraverso l'utilizzo dell'istogramma delle luminosità.

Presupposto fondamentale per l'utilizzo di questa tecnica è che nell'immagine siano presenti due classi. Partendo da ciò si andrà a calcolare la soglia che permette meglio di separarle.

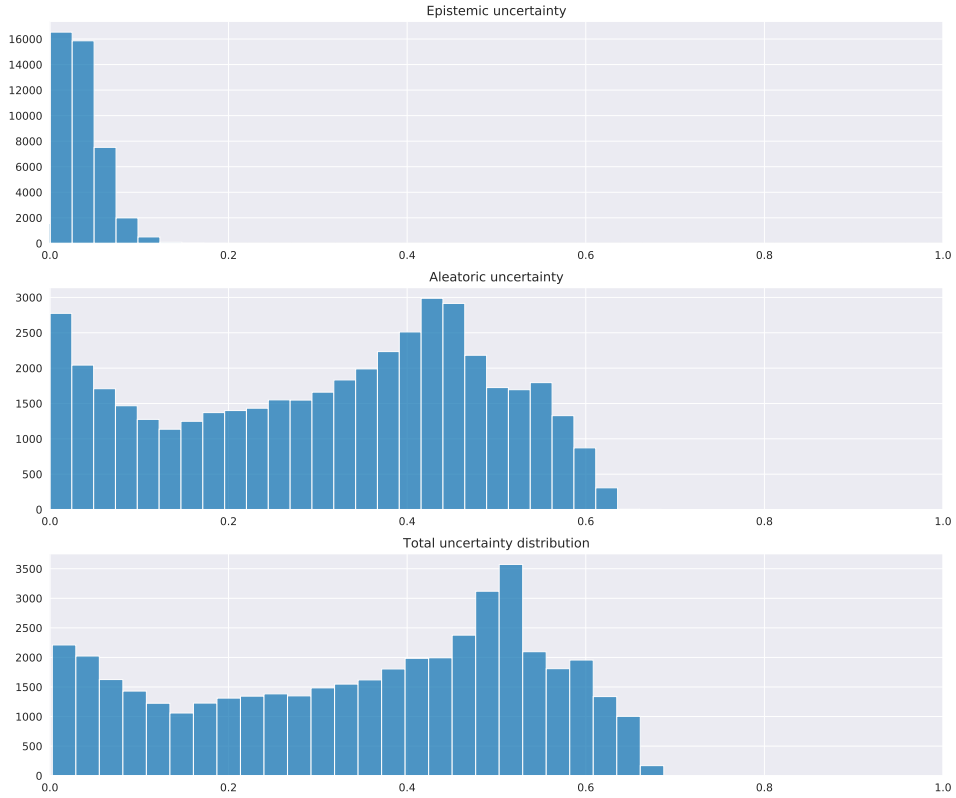


Figura 4.4. In alto istogramma incertezza epistemica, seguito da istogramma incertezza aleatoria, in basso istogramma incertezza totale.

Osservando l'istogramma dell'incertezza si nota che il requisito della presenza di due classi è soddisfatta, si è proceduto quindi allo studio dell'istogramma dell'incertezza totale.

L'obiettivo sarà quindi quello di minimizzare la varianza intra-classe, questa può essere definita come la somma pesata della varianza all'interno delle due classi.

Con il teorema di Otsu si dimostra che minimizzare la varianza intra-classe equivale a massimizzare la varianza interclasse:

$$\sigma^2 = p_1(k)p_2(k)[\mu_1(k) - \mu_2(k)]^2$$

Trovando il massimo di questa equazione, si sarà individuata la soglia ideale.

La varianza inter-classe e la soglia calcolata con il metodo di Otsu sono riportate in figura 4.5, rispettivamente in colore arancione e in colore rosso.

4.6.2 New threshold

E' stata inoltre studiata una metodologia di sogliatura automatica alternativa rispetto a quella precedentemente enunciata, in quanto potrebbe risultare alle volte necessario essere più conservativi nella costruzione del nuovo dataset, soprattutto nel caso in cui si sta lavorando su set di dati di piccole dimensioni.

Partendo dalla soglia di Otsu (T_1), si è andato a calcolare il picco immediatamente successivo nell'istogramma, in questo modo si otterrà un intervallo compreso tra T_1 e il massimo valore successivo.

Lo step seguente è stato andare a trovare il punto di massima variazione all'interno di questo intervallo in modo da evitare ad esempio i casi in cui il massimo valore di incertezza si trovasse in corrispondenza di un plateau.

Utilizzando quindi la funzione "diff" della libreria di Python *Numpy* si è ottenuto un vettore contenente al proprio interno i valori di variazione.

Individuando il massimo in questo vettore si è identificata una nuova soglia T_2, che sarà quindi compresa tra T_1 e il picco di incertezza.

Nella figura 4.4, la nuova soglia è indicata con la dicitura "New thresold" ed è associata al colore nero.

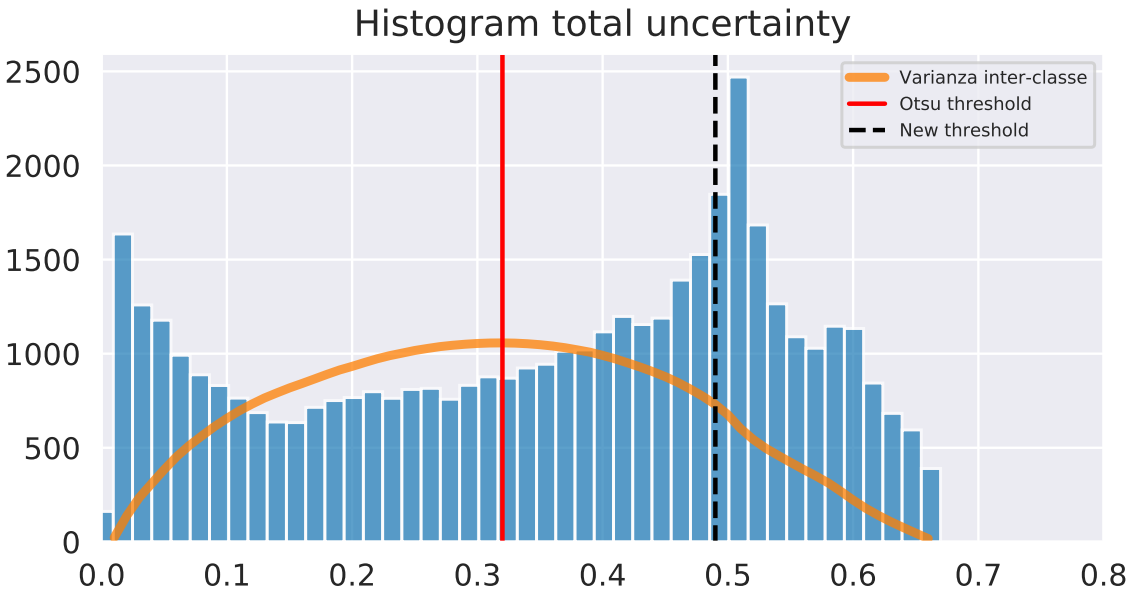


Figura 4.5. Visualizzazione delle soglie ricavate con le metodologie automatiche sull'istogramma dell'incertezza totale derivante dal training set.

4.6.3 Comparazione tra metodi di soglia

Come è facilmente intuibile, la scelta della soglia si ripercuoterà sulla numerosità del dataset che verrà generato.

Nelle tabella seguente è indicato il numero di tiles che avranno i nuovi dataset con l'utilizzo di una delle soglie utilizzate.

Set	Dimensione iniziale	Otsu	New threshold
Train	45000	17751	30359
Validation	19200	8776	13582

Tabella 4.7. Dimensione nuovi dataset a seconda della soglia selezionata.

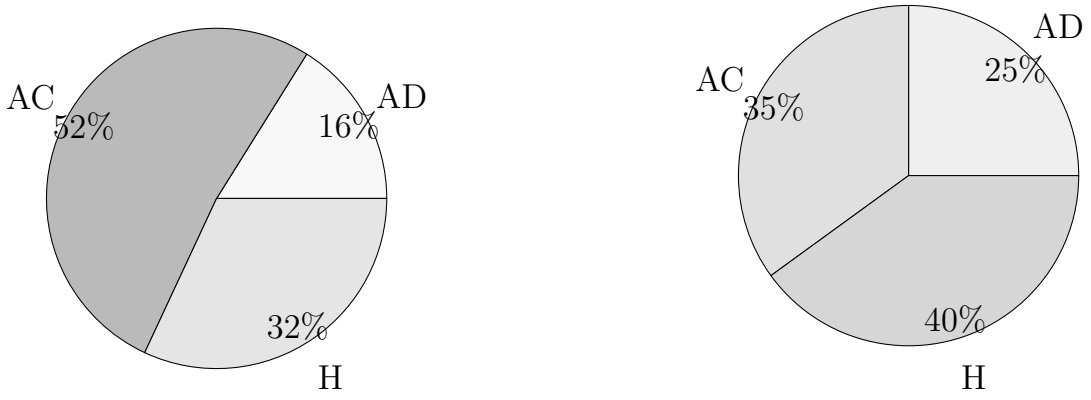


Figura 4.6. Grafici a torta percentuale degli elementi rimossi nel training set con il metodo di: Otsu (a) e New threshold (b).

Nei grafici a torta riportati nella figura 4.6 è indicata la percentuale di immagini eliminate per classe, per entrambi i casi in analisi.

E' importate notare come nel caso della prima soglia automatica, le immagini che si andranno a rimuovere apparterranno per la maggior parte alla classe AC, mentre per quanto riguarda la soglia "New", la rimozione tra le classi è più bilanciata.

La classe dell'adenoma risulta in entrambi i casi quella con la percentuale di rimozione più bassa, ciò sta a significare che la rete fa maggiore difficoltà a differenziare tra adenocarcinoma e sano.

Nel software sviluppato per il datacleaning, di cui successivamente si discuterà, si lascia all'utente la scelta della soglia tra le due che vengono automaticamente ricavate.

4.6.4 Prestazioni drop out

Otsu threshold

Si è generato quindi un nuovo dataset selezionando solo le tiles che avessero incertezza inferiore a quella indicata da quella di Otsu(0,32).

Set	Accuratezza
Train	95,2%
Validation	90,5%
Test	63,5%

Tabella 4.8. Valori di accuratezza del modello drop out, sul dataset ottenuto selezionando solo le immagini con incertezza inferiore a "Otsu threshold".

Set	Accuratezza
Train	91,72%
Validation	90,2%
Test	67,9%

Tabella 4.9. Valori di accuratezza del modello drop out con data augmentation, sul dataset ottenuto selezionando solo le immagini con incertezza inferiore a "Otsu threshold".

Osservando le tabelle possiamo affermare che il dataset ottenuto con la soglia di Otsu è più coerente, infatti nel training e nel validation set si ottiene un'accuratezza molto alta, però l'aver ridotto drasticamente la dimensione del dataset ha fatto si che le prestazioni di generalizzazione peggiorassero rispetto all'utilizzo del dataset completo. Il nuovo dataset infatti rappresenta soltanto 39,5% di quello iniziale.

New threshold

Nella seguente tabella sono riportate le prestazioni del modello con drop out, utilizzando in fase di training il dataset generato selezionando tutte le tiles

caratterizzate da un'incertezza minore di "New threshold".

Set	Accuratezza
Train	91,72%
Validation	87,5%
Test	71,5%

Tabella 4.10. Valori di accuratezza del modello drop out sul dataset ottenuto selezionando solo le immagini con incertezza inferiore a "New threshold".

Le prestazioni generali hanno subito un miglioramento, come ipotizzabile. Avere un dataset coerente ha portato all'accuratezza del 91,7% sul training set.

Di seguito sono indicati i valori di accuratezza utilizzando sempre il nuovo dataset, ma con l'aggiunta del data agumetation i parametri sono stati settati come indicato in tabella 4.3.

Set	Accuratezza
Train	89,29%
Validation	85,65%
Test	79,25%

Tabella 4.11. Valori di accuratezza del modello drop out con data augmentation, sul dataset ottenuto selezionando solo le immagini con incertezza inferiore a "New threshold".

Risulta chiaro come la somma tra datacleaning con sistema bayesiano e data agumetation, abbia portato a risultati ottimi soprattutto in quanto a prestazioni di generalizzazione, è presente infatti un incremento del 11% circa sul test set.

Capitolo 5

Bayesian framework

5.1 Introduzione

L'obiettivo finale della tesi è stato quello di creare degli strumenti "user friendly" che permettessero, anche a chi non ha conoscenza nell'ambito della programmazione, di poter consultare le informazioni che solo un sistema di tipo bayesiano può fornire.

A tale scopo si è creato un "framework" cioè un insieme di classi e metodi, in linguaggio Python, utile per la gestione di "whole slide images", ma anche per l'implementazione di reti bayesiane. Essendo i suddetti codici scritti nel rispetto del paradigma del linguaggio a oggetti, i metodi e le classi possono essere facilmente utilizzati anche in altri progetti oltre ai due applicativi di seguito enunciati.

Si è, quindi, sviluppato un software per all'analisi di immagini di tipo istologico che permettesse di visualizzare oltre alla mappa delle predizioni anche una mappa delle incertezze.

Il secondo software invece consente di svolgere datacleaning automaticamente attraverso una rete di tipo bayesiano e, una volta salvato il nuovo dataset, di creare una nuova rete con il conseguente miglioramento delle prestazioni.

Partendo dal codice utilizzato per la parte teorica della tesi, si è creata una controparte con interfaccia grafica che, attraverso pochi semplici step, desse l'opportunità di ripercorrere gli stessi passi.

I due applicativi e l'intero framework sono disponibili in un repository

pubblica "GitHub"[1], raggiungibile attraverso il URL riportato in nota. [2]

5.1.1 PyQt5

Le API su cui si è basato lo sviluppo della parte grafica degli strumenti sopra citati si chiama *PyQt5*. Quest'ultima è una libreria Python che utilizza Qt GUI framework.

Qt è una libreria scritta in linguaggio C++, molto utilizzata nell'ambito industriale, che permette la creazione di interfacce attraverso l'impiego di particolari elementi grafici.

I widget sono gli elementi fondanti della progettazione grafica, attraverso essi sarà possibile inserire pulsanti, testo, immagini, e molte tipologie di elementi che si possono trovare nei software di uso comune.

PyQt5 eredita un'importante proprietà da Qt, può essere considerata come una libreria con una grandissima duttilità, essa si adatta a molti sistemi operativi come ad esempio Windows, Linux, Mac Os, ma anche Android e IOS.

Questo framework oltre a mettere a disposizione le classiche funzioni, permette di gestire i thread in maniera molto efficiente. La suddetta proprietà verrà utilizzata abbondantemente in entrambi gli applicativi in modo da poter offrire un'esperienza di utilizzo fluida e veloce, rendendo inoltre il software reattivo anche quando si sta svolgendo in background un task pesante.

5.1.2 OpenSeadragon

Seadragon è una tecnologia che permette la visualizzazione di grafica o immagini superando le problematiche generate nella gestione di file di grandi dimensioni.

Questa API si basa su due piattaforme: una che utilizza Javascript e XML e un'altra che adopera Microsoft's Silverlight. Nel primo caso è necessario soltanto avere un web plug-in appropriato, nel secondo invece è necessario aver installato l'applicazione di Microsoft: Silverlight.

OpenSeadragon è un visualizzatore "open-source web based" basato su seadragon e javascript che permette di visualizzare immagini ad alta risoluzione,

[1]Piattaforma online che permette la condivisione e lo sviluppo simultaneo da parte di più programmatori di software.

[2]https://github.com/Pier45/WSI_analysis

riducendo al minimo la latenza e i tempi di caricamento che comunemente sarebbero associati a questo tipo di file.

Una volta selezionato un file per la visualizzazione, il software crea una collezione di immagini detta "Deep Zoom Collection" a risoluzione di 256x256 pixel per tutti i layer presenti nello stesso.

Il segreto che si nasconde dietro alle ottime prestazioni che si ottengono con questa tecnologia sta nel mostrare un numero di immagini della collezione che sia proporzionale alla risoluzione dello schermo. In questo modo il computer non dovrà gestire una quantità inutile di dati derivanti da immagini con risoluzione più elevata da quella mostrata nello schermo.

Questo visualizzatore verrà utilizzato nello strumento di analisi al fine di garantire una visualizzazione alternativa e a più alta risoluzione.

5.2 Analisi

L'obiettivo dell'applicativo è quello di analizzare file in formato svs e quindi di mostrare un'immagine in uscita con la categorizzazione dei tessuti.

Per la classificazione è stata utilizzata una rete bayesiana che ha permesso di calcolare l'incertezza sulla predizione. Questa informazione è stata utilizzata per la selezione delle tiles da mostrare oppure da rimuovere in quanto ad alta incertezza.

Tra le opzioni disponibili nel tool è presente la possibilità di cambiare modello oppure di selezionare il numero di Monte Carlo tra un set di possibili valori.

E' inoltre possibile visualizzare una singola classe per volta e, se necessario, lo strumento può anche restituire una mappa dei valori dell'incertezza.

Ogni classe è stata associata a un colore e l'immagine di partenza è stata convertita in scala di grigi, così da agevolare la visualizzazione.

Nell'applicativo è inoltre disponibile un'apposita sezione per visualizzare l'immagine con OpenSeadragon.

Lo strumento è stato sviluppato per l'analisi di tessuto colon-rettale.

5.2.1 Analisi ui

L'interfaccia dello strumento è stata pensata per essere molto intuitiva. Si può dividere in tre parti:

1. menu bar: nella parte alta sono presenti 5 opzioni, "File", "Analysis", "View", "Options" e infine "Help";

2. tool bar: comunemente è la sezione dell'interfaccia che mette a disposizione le funzioni maggiormente utilizzate. Anche nel suddetto caso in quest'area sono presenti le funzioni che si adopereranno più spesso e sono individuabili anche nella menu bar. Queste funzioni sono associate a un'icona che permetterà di intuire velocemente a cosa è riferita;

3. area per la visualizzazione dei risultati: in quest'area sarà disponibile in prima istanza l'immagine completa del file svs che si andrà ad analizzare; inoltre una volta compiuta l'analisi sarà possibile visualizzare in questa zona sia i risultati della classificazione che la mappa delle incertezze.

Dato che lo scopo dell'analisi è quello di andare a studiare una singola immagine, la prima operazione da compiere è quella di andare a selezionare il file svs desiderato, in questo modo si sbloccheranno gli altri menu che prima risultavano non accessibili.

Questo step può essere compiuto sia da menu bar che da tool bar cliccando la voce associata all'icona di una cartella.

Appena selezionato il file desiderato, verrà generata un'immagine alla risoluzione desiderata che sarà subito visualizzata. Si aprirà una nuova finestra di piccole dimensioni contenente una progress bar che indicherà l'andamento del processo di creazione delle tiles.

Una volta completata la procedura, la finestra verrà chiusa e sarà possibile passare allo step successivo.

Di default è selezionato il livello a minore risoluzione, questo per rendere l'analisi più veloce sia nella fase di classificazione che di creazione delle tiles.

Tuttavia, grazie all'apposito menu Analysis, selezionando la voce "slow mode", è possibile passare al livello a maggiore risoluzione. Da ciò ne consegue una maggior durata dei processi prima citati ma una mappa finale delle predizioni e delle incertezze più fitta.

Dal menu dedicato all'analisi è, inoltre, possibile modificare delle impostazioni riguardanti lo strumento. Sono presenti, infatti, l'opzione per selezionare un altro modello in formato "h5" e anche la facoltà di selezionare un Monte Carlo sample tra tre possibili valori: 5, di default selezionato, 25 e 50.

L'ultima voce, dello stesso menu, permette di avviare la classificazione e di procedere allo studio del file svs precedentemente selezionato. Questa opzione è associata all'icona di una freccia di colore verde, che è presente anche nella toolbar. Si può avviare l'analisi anche semplicemente attraverso la combinazione di tasti ctrl+s.

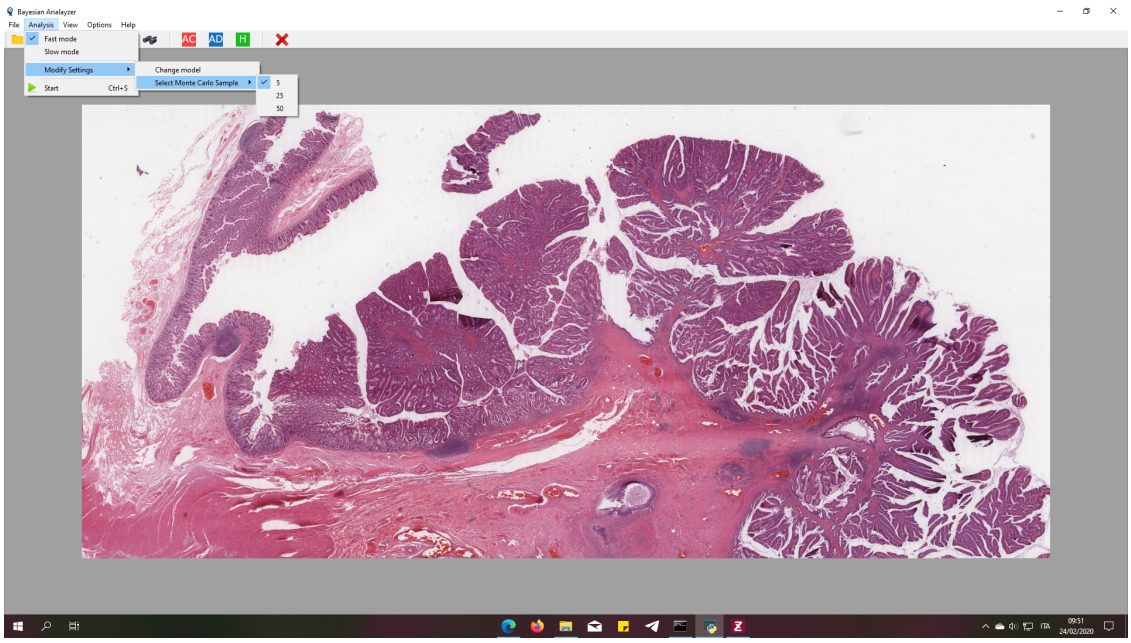


Figura 5.1. menu analisi, e possibile settaggio dei parametri.

Una volta premuto sul tasto di avvio, come per quanto successo per la creazione delle tiles, si aprirà una nuova finestra con una progress bar associata a una percentuale di completamento. Una volta ultimato il processo la finestra con la barra si chiuderà in automatico e nella zona di visualizzazione sarà accessibile la classificazione 5.2.

Dall'immagine si nota che ad ogni classe è associato un colore, rosso per l'adenocarcinoma, blu per l'adenoma e infine verde per il tessuto sano.

E' possibile inoltre constatare che non tutta l'immagine è associata a una classe, l'algoritmo, infatti, tiene in conto anche l'incertezza. Le tiles che il software ritiene incerte non vengono associate ad una classe, e quindi non saranno mostrate in questa visualizzazione.

Per agevolare la comprensione, l'immagine del tessuto, è stata trasformata in scala di grigi, in questo modo le classi in cui sono state classificate le tiles sono più evidenti.

Nella tool bar è possibile selezionare una delle tre classi, cliccando sugli appositi pulsanti oppure attraverso la voce "View" dalla menu bar.

Dalla medesima voce del menu è inoltre possibile accedere alla visualizzazione dell'incertezza calcolata per ogni singola tiles.

Sono disponibili tre opzioni:

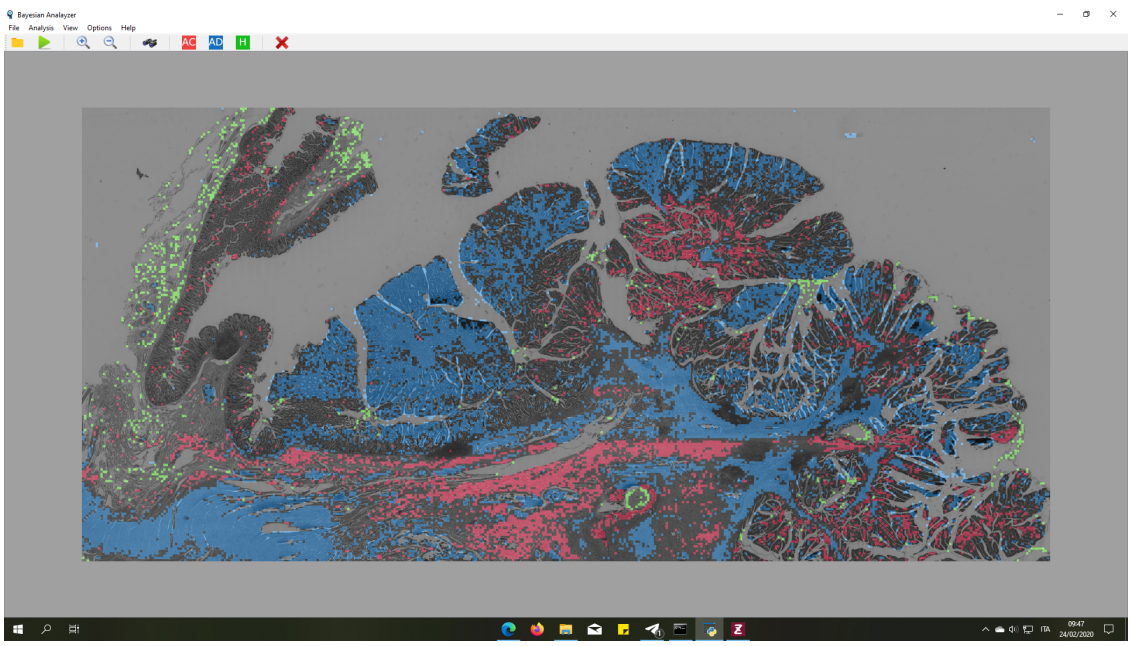


Figura 5.2. Vista al termine dell'analisi.

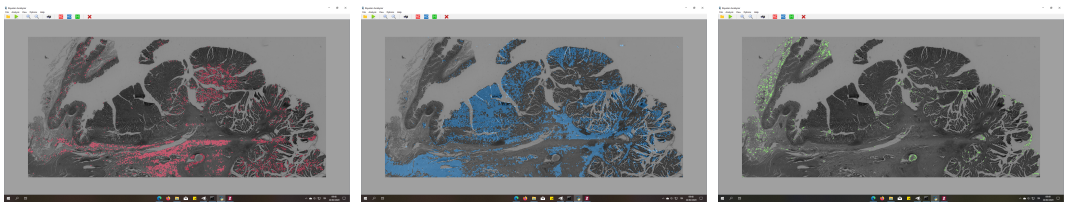


Figura 5.3. Visualizzazione singola classe, attraverso l'utilizzo della barra degli strumenti.

1. total uncertainty: sarà visibile l'incertezza totale cioè la somma delle due componenti di seguito riportate;

2. aleatoric uncertainty: si visualizzerà solo l'incertezza aleatoria dovuta alla rumorista dei dati;

3. epistemic uncertainty: si visualizzerà l'incertezza epistemica cioè l'incertezza dovuta al modello utilizzato;

Nel caso in cui fosse necessario è possibile, attraverso la voce "View with no overlay", visualizzare l'immagine su cui si sta compiendo l'analisi non in scala di grigi ma RGB.

Nel menu "View" sono disponibili altre alternative; sono, infatti, presenti due voci: zoom in e zoom out, associate a una lente di ingrandimento con un segno più e l'altra con il segno meno. Cliccando su una delle lenti di ingrandimento è possibile zoomare una parte dell'immagine per visualizzare in maniera più ravvicinata una specifica zona. Questa opzione è raggiungibile anche attraverso la tool bar.

La voce "Normal size" permette di tornare alle dimensioni iniziali dell'immagine, attraverso l'utilizzo di particolari librerie Python, le immagini inizialmente si adatteranno alla risoluzione dello schermo su cui sono state riprodotte. Nel caso in cui le immagini siano di dimensione molto superiore alla risoluzione del monitor verrà calcolato un fattore di scala che sarà utilizzato per scalare l'immagine in modo da renderla completamente visibile all'interno dell'area di visualizzazione.

Passando alla voce successiva nella menu bar, è presente "Options". Attraverso esso saranno accessibili due voci, la prima che permetterà di accedere a un particolare visualizzatore di file svs ad alta risoluzione, seguita poi dalla voce "About".

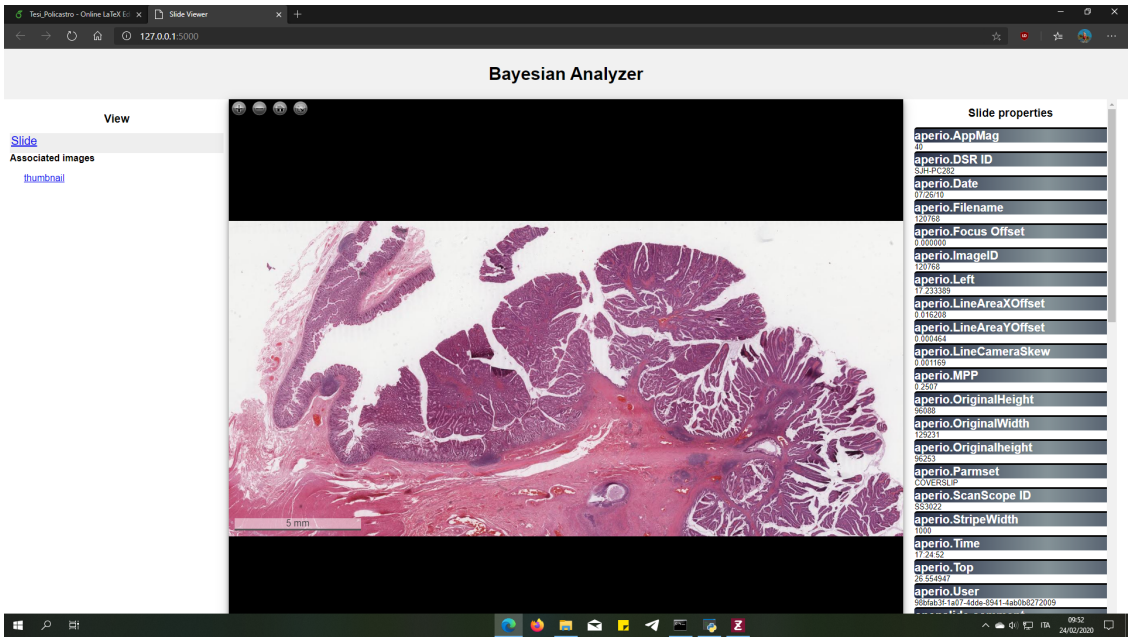


Figura 5.4. Visualizzatore ad alta risoluzione su server locale.

Per accedere al visualizzatore ad alta risoluzione sarà necessario quindi cliccare sull'apposita voce nella menu bar oppure cliccare sul binocolo all'interno della toolbar.

Comparirà una finestra informativa che spiegherà brevemente come adoperare il visualizzatore, intanto in backgoud attraverso un thread verrà avviato un server locale, grazie alla libreria di Python *flask*, precisamente alla porta 5050.

Una volta cliccato "ok" sul messaggio di dialogo si aprirà automaticamente un tab nel browser attraverso cui sarà possibile visualizzare il file "svs" con una risoluzione adattiva.

In questa pagina web inoltre sulla destra sarà possibile visualizzare tutti i metadati associati all'immagine, invece sulla sinistra saranno visualizzate le immagini associare al file svs. Molto spesso, infatti, i file "Aperio" possono immagazzinare anche altre immagini di piccola dimensione in cui sono raffigurati ad esempio i risultati di altri tipi di analisi.

Ritornando all'applicativo di analisi in "Options" è possibile accedere attraverso la seconda voce del menu a una breve introduzione allo strumento di visualizzazione avanzato.

Ultima voce della menu bar è "About", essa permette di accedere a un sommario delle funzioni racchiuse nello strumento di analisi.

5.2.2 Struttura del programma di analisi

E' importante esplicare l'organizzazione oltre che dell'interfaccia anche dei file, nonché le funzioni più importanti che sono state utilizzate nel software di analisi.

Durante il processo di creazione tiles, il software autonomamente ricava la potenza di calcolo del pc in cui viene utilizzato. E', infatti, capace di individuare il numero di threads del pc, e quindi di dividere il task fra i vari "workers".

L'algoritmo è stato sviluppato dal candidato ed è capace di elaborare il lavoro che si deve svolgere e, in base alla mole di quest'ultimo, dividerlo ai vari lavoratori.

Questa funzione permette all'applicativo di non richiedere particolare hardware per funzionare e, in caso di computer molto prestante, di ottenere risultati in maniera molto rapida.

A conferma di ciò, è possibile individuare all'interno della cartella che il software genera per la gestione dell'analisi, un numero di cartelle che iniziano con "p" proporzionale alla potenza di calcolo del pc. Il caso in figura 5.5 mostra una cartella in cui sono presenti 8 sottocartelle, infatti il computer su cui è stata svolta l'analisi è equipaggiato con un processore Intel i7-7700 [13], CPU caratterizzata da 8 thread.

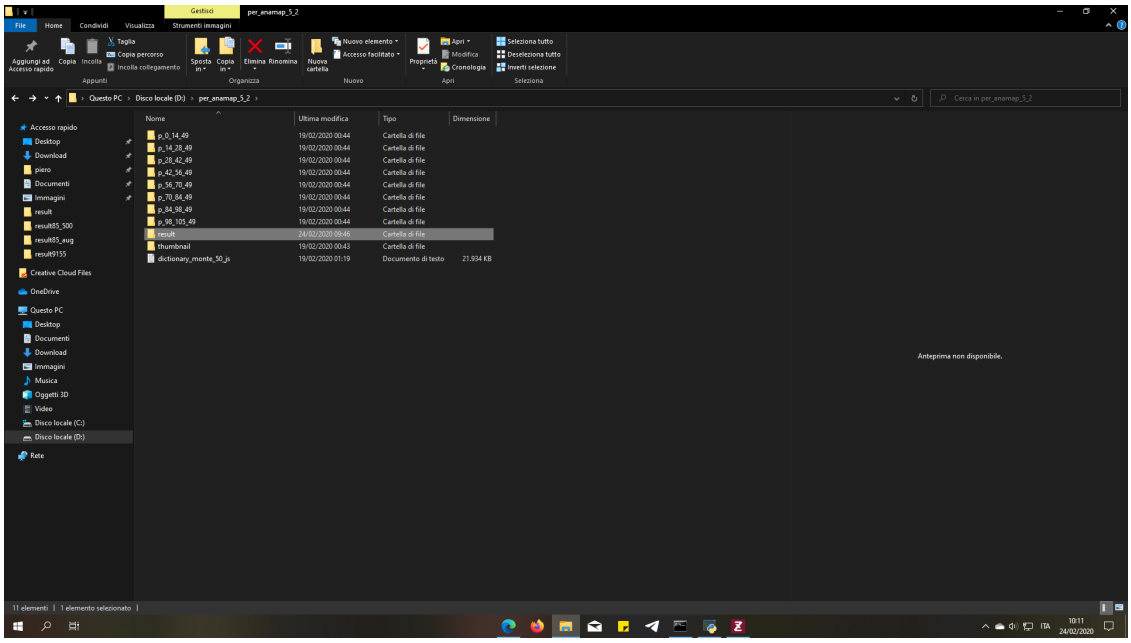


Figura 5.5. Organizzazione dei file creati dal software di analisi.

Nella stessa cartella inoltre è presente una cartella denominata "result", al suo interno è possibile trovare:

1. Pred_class.png: l'immagine in analisi con sovrapposto la classificazione delle tiles, quindi con tutte e tre le classi;

2. AC.png , AD.png, H.png: l'overlay della singola classe;

3. Uncertainty: la cartella contiene tre file raffiguranti l'incertezza totale, l'aleatoria e l'epistemica.

Salvando le immagini in queste cartelle è possibile quindi visualizzare il risultato della classificazione anche senza l'utilizzo del tool.

C'è, inoltre, un ultimo file in questa cartella, rinominato:

$$dictionary_monte_\{numeromontecarlo\}.txt$$

In esso sono presenti i dati relativi ad ogni tiles che compone il dataset in analisi. Il file ha la struttura riportata in figura 5.6.

Il file è in formato json e, come si può vedere, a ogni tile associa una chiave, in questo caso "100". Alla "key" è associato come valore un dizionario

```
{
  "100": {
        "im_path":"C:/Users/10002_2/p_0_11_28\\
           tile_100_3_15.png",
        "shape_x": 64,
        "shape_y": 64,
        "col": 3,
        "row": 15,
        "class": "AC",
        "epi": 0.1828,
        "ale": 0.3624
        }
}
```

Figura 5.6. Singolo elemento file json derivante dall'analisi.

contenente il path dove è possibile trovare la tile, la dimensione in pixel, la posizione della tile utilizzando le coordinate del file svs; infine si può trovare la classe predetta e i valori di incertezza epistemica e aleatoria.

Il salvataggio della dimensione in pixel della tile é un passo importante per la creazione delle immagini finali, infatti nella fase di creazione del dataset, le tiles che si originano non sono sempre della dimensione desiderata.

L'eventualità di non avere altezza e larghezza uguali ha imposto l'utilizzo dello "zero padding" durante la classificazione, in modo da poter ottenere una classificazione anche per queste tiles non regolari.

Durante la fase di creazione delle immagini risultanti dall'analisi con lo strumento sarà necessario creare una maschera da sovrapporre all'immagine iniziale delle dovute proporzioni, qui entrano in gioco le specifiche precedentemente salvate.

5.3 Data cleaning

L'obiettivo del secondo software sviluppato è di ridurre le lungaggini e il consumo di risorse necessarie a compiere datacleaning manualmente su un dataset di immagini.

Per svolgere questa operazione è stato opportuno, quindi, creare un interfaccia grafica utilizzando la libreria *PyQt5* che permettesse, attraverso un

processo guidato, di replicare l'analisi fatta dal punto di vista teorico nei capitoli precedenti.

Nell'applicativo sono integrate funzioni utilizzate per lo strumento di analisi, come ad esempio i metodi adoperati per la creazione delle tiles. E' stato necessario, tuttavia, aggiungere nuovo codice al fine di permettere il training del modello e anche per lo studio dell'incertezza.

Il software idealmente può essere diviso in sezioni, la prima è dedicata alla creazione delle tiles. Sarà quindi possibile selezionare i file svs da cui dovranno essere estratte le immagini per poi poter fare il training del modello.

La sezione successiva è dedicata proprio al training. E' inoltre possibile settare vari paramenti, di seguito se ne parlerà più approfonditamente.

Una volta ottenuto il modello è necessario classificare il dataset su cui si desidera attuare l'operazione di datacleaning, in questo modo si otterranno i dati necessari per la sezione successiva in cui si dovrà selezionare sotto quale soglia le tiles sono ritenute incerte.

In ultima analisi è presente un'area dedicata al testing, quest'ultima verrà utilizzata per poter convalidare le prestazioni sotto l'aspetto della generalizzazione.

5.3.1 Data cleaning ui

L'interfaccia grafica di questo programma è strutturata in maniera completamente diversa dal software dedito all'analisi, si è infatti utilizzato il widget "Tabs", che permette di suddividere le funzionalità all'interno di pagine.

Nella parte alta è presente anche questa volta una menu bar con due voci, attraverso "About" sarà possibile accedere a un'introduzione al software in cui sono spiegati i passaggi principali per utilizzare il tool.

La prima finestra ha come titolo *Get Tiles*. Utilizzando l'apposito pulsante nella parte alta della pagina è possibile selezionare la cartella nella quale si desiderano salvare tutti i dati che scaturiranno dall'analisi.

Di seguito la pagina è divisa in tre sezioni ognuna dedicata ad una particolare parte del dataset: training, validation, test.

Con il pulsante "Select folder" è possibile selezionare una cartella all'interno della memoria del computer, questa deve però rispettare una particolare organizzazione, infatti è indispensabile che i file "svs" siano suddivisi in tre cartelle rinominate con il nome della classe: AC, AD e H.

Conclusa la selezione, l'elenco dei file sarà visibile nella relativa sezione, come mostrato nella figura 5.7

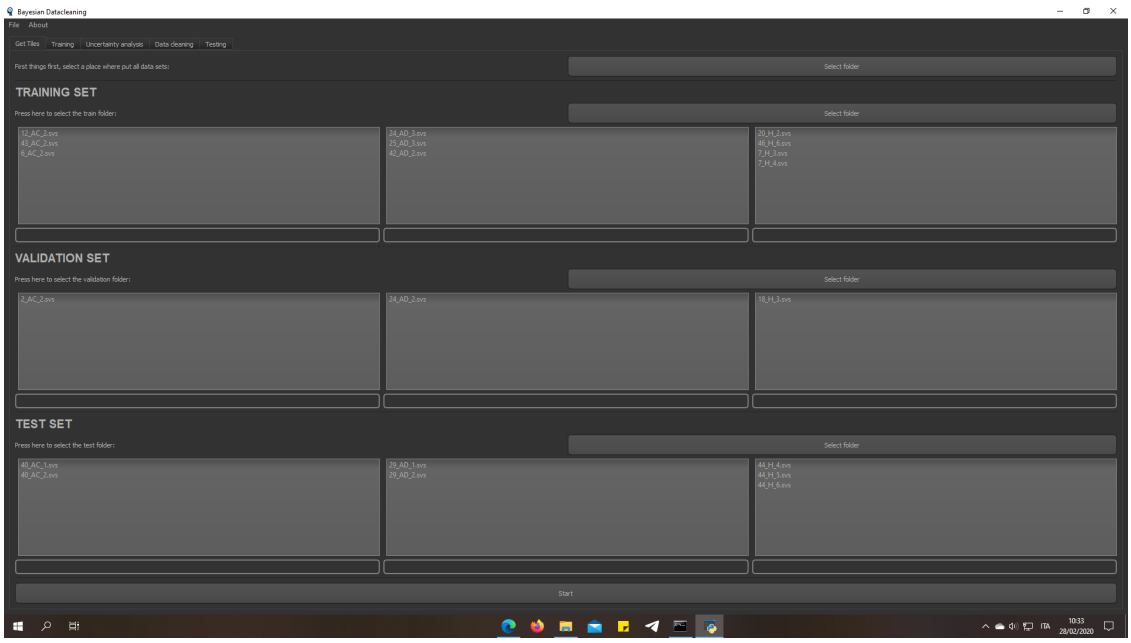


Figura 5.7. Pagina dedicata all'estrazione delle tiles.

Si può quindi procedere all'avvio della creazione delle tiles, una volta premuto "Start", sotto ogni lista sarà visibile una barra di avanzamento che permetterà di intuire lo stato del processo.

Conclusasi questa prima parte si passa alla seconda finestra: *Training*.

Fondamentale è la selezione del modello che si desidera utilizzare. Si può scegliere tra il modello basato su drop out o quello che implementa la KL. E' inoltre possibile selezionare il numero di epoche entro il quale si vuole allenare la rete.

Se necessario, si può anche modificare la dimensione dei batch nel caso in cui il dispositivo attraverso il quale si sta svolgendo l'analisi non abbia a disposizione un quantitativo di RAM adeguato alla dimensione di default indicata dal software.

Nell'evenienza in cui l'utente imponga una determinata soglia, sia per il numero di epoche che per la dimensione dei batch, il valore designato andrà a sostituire quello di default indicato a fianco al tasto "ok".

E' inoltre possibile attivate il "Data augmentation" inserendo una spunta nell'apposito campo, in questo modo si abiliterà questa funzione durante il training.

Cliccando "Start" è possibile avviare il training. Nella parte sottostante una volta avviato il processo sarà possibile visualizzare oltre al time stamp

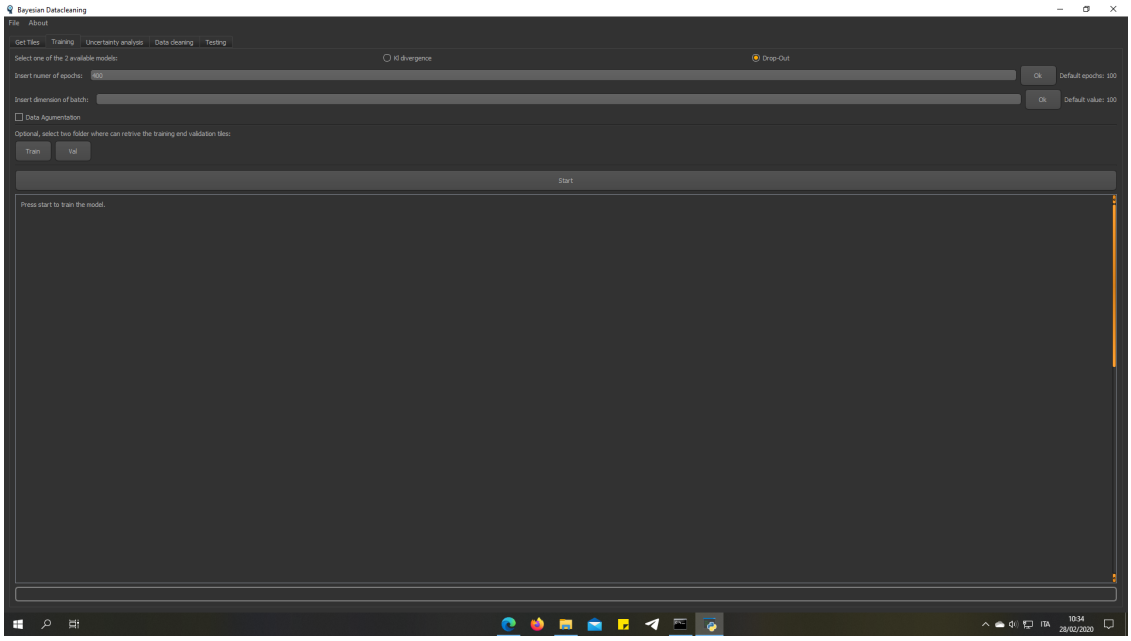


Figura 5.8. Pagina dedicata al training.

dell'avvio, anche il progresso batch per batch dell'accuratezza.

Alla conclusione di ogni epoca verrà mostrata una riga di ricapitolazione con indicato l'accuratezza finale sia del validation che del training set. Questo processo continuerà fino alla conclusione della fase di training.

Opzionale è la possibilità di selezionare una particolare cartella per il training set e il validation set; questo risulterà utile in un secondo momento in cui il data cleaning sarà ultimato e si avrà a disposizione un nuovo set per allenare la rete. Nella parte inferiore inoltre è presente una barra di avanzamento che renderà di facile intuizione la percentuale di epoche completate.

Proseguendo alla terza finestra *Uncertainty analysis* c'è la possibilità classificare uno dei tre set utilizzando la rete precedentemente allenata.

A discrezione dell'utente è la possibilità di modificare il Monte Carlo sample, che di default è 5.

Anche in questo caso, come per l'applicativo di analisi, verrà creato un file json *dictionary_monte_{numero MonteCarlo}_js.txt* nel quale saranno raccolte le informazioni derivanti dalla classificazione delle immagini. Il suddetto file avrà una struttura più complessa rispetto a quello creato dal software di analisi.

Questo file verrà utilizzato nella finestra finale *Data cleaning* per procedere alla parte più importante del processo di datacleaning.

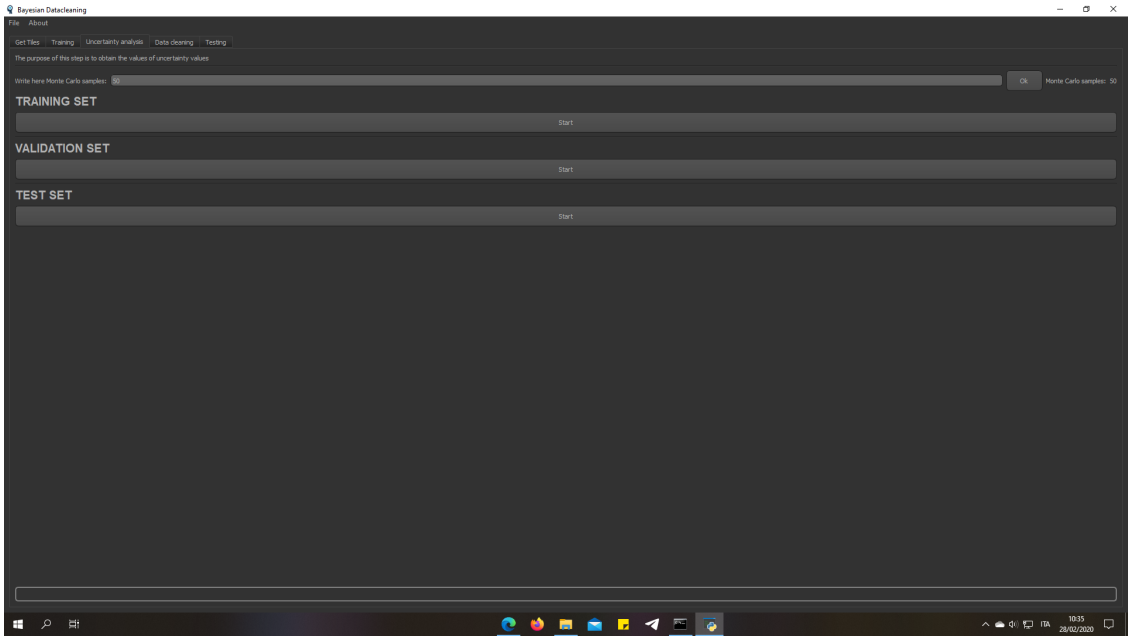


Figura 5.9. Pagina dedicata allo studio dell'incertezza.

Per prima cosa nella parte alta della pagina si deve selezionare quale set si vuole andare ad analizzare tra il training set e il validation set. Naturalmente saranno accessibili solo se nella fase precedente sono stati classificati.

Nell'istante in cui si seleziona il set, compariranno gli istogrammi dell'incertezza totale, dell'aleatoria e dell'epistemica nella parte sottostante.

Osservando al di sotto dell'istogramma dell'incertezza totale sarà visibile una didascalia riportante in numero di elementi che caratterizza di dataset selezionato.

Proseguendo verso il basso è presente una sezione che permette di selezionare una soglia attraverso la quale si desidera separare le immagini incerte da quelle certe. Sono disponibili due opzioni:

1. auto: l'algoritmo per prima cosa calcolerà la soglia attraverso il metodo di Otsu. In questo modo, come già spiegato, si massimizza la varianza inter-classe. Se la nuova soglia non garantisce una numerosità totale pari al 60% del dataset iniziale si calcolerà una soglia più conservativa cercando il punto in cui si ha la più grande variazione come spiegato dal punto di vista teorico nel paragrafo "New threshold";

2. manual: una volta cliccato su questo pulsate, comparirà un elemento in cui poter inserire il valore di soglia. Prima di procedere all'analisi verrà

fatta una verifica sulla pertinenza della soglia, dovrà essere compresa tra 0 e 1, in caso contrario nel grafico non verrà mostrata la nuova soglia.

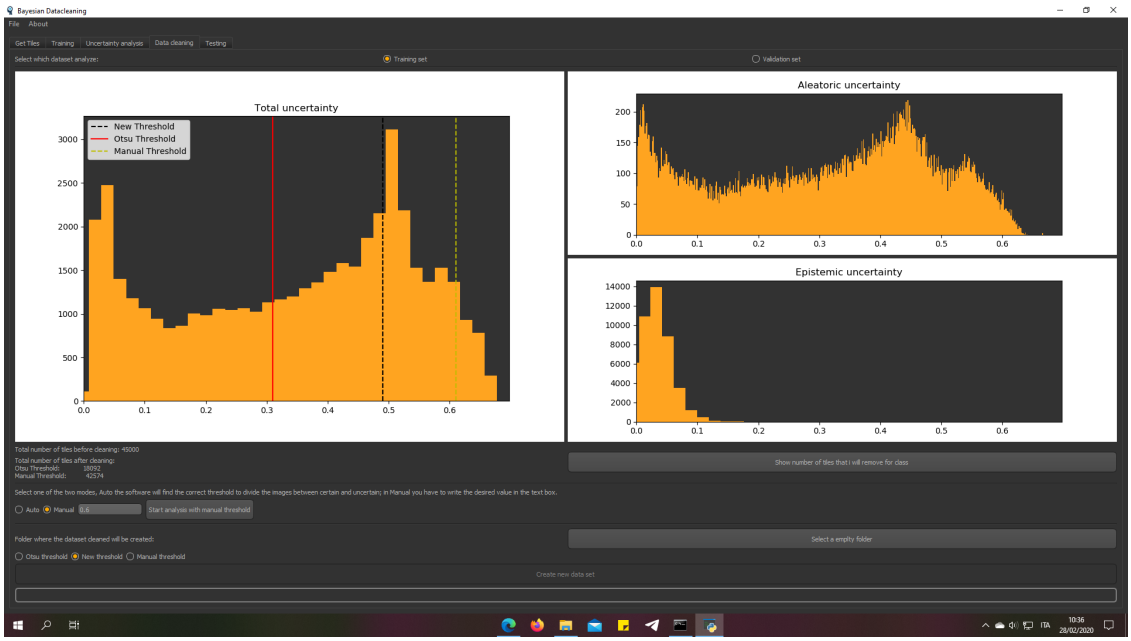


Figura 5.10. Pagina dedicata al datacleaning.

La didascalia che prima riportava solo il numero di immagini iniziali, verrà aggiornata una volta selezionata la soglia desiderata. Saranno elencati, quindi, nella suddetta area il numero di immagini che si otterranno se si selezionerà una delle due soglie automatiche.

Di fianco a questa didascalia verrà mostrato inoltre un pulsante che permetterà di controllare a quale classe appartengono le tiles rimosse. Si è utilizzato un grafico a torta per questa visualizzazione. Nel caso in cui siano state calcolate 2 soglie, cioè sia "Otsu" che "New", saranno quindi visibili due grafici a torta nella finestra appena indicata.

Nell'istogramma dell'incertezza totale, inoltre, sarà visibile una linea verticale in corrispondenza delle varie soglie. Nel grafico è presente una legenda in cui è facile comprendere la corrispondenza tra colore e soglia.

Per la creazione del nuovo dataset sono necessari altri due passaggi. Si dovrà selezionare una cartella vuota nella quale si andrà a creare il nuovo dataset.

E' necessario, selezionare qual è la soglia che si desidera tra le disponibili. Concluso quest'ultimo passaggio cliccando sul tasto "Create" si avvierà il

processo di creazione del nuovo dataset; lo stato dello stesso sarà visibile attraverso la barra di avanzamento nella parte inferiore.

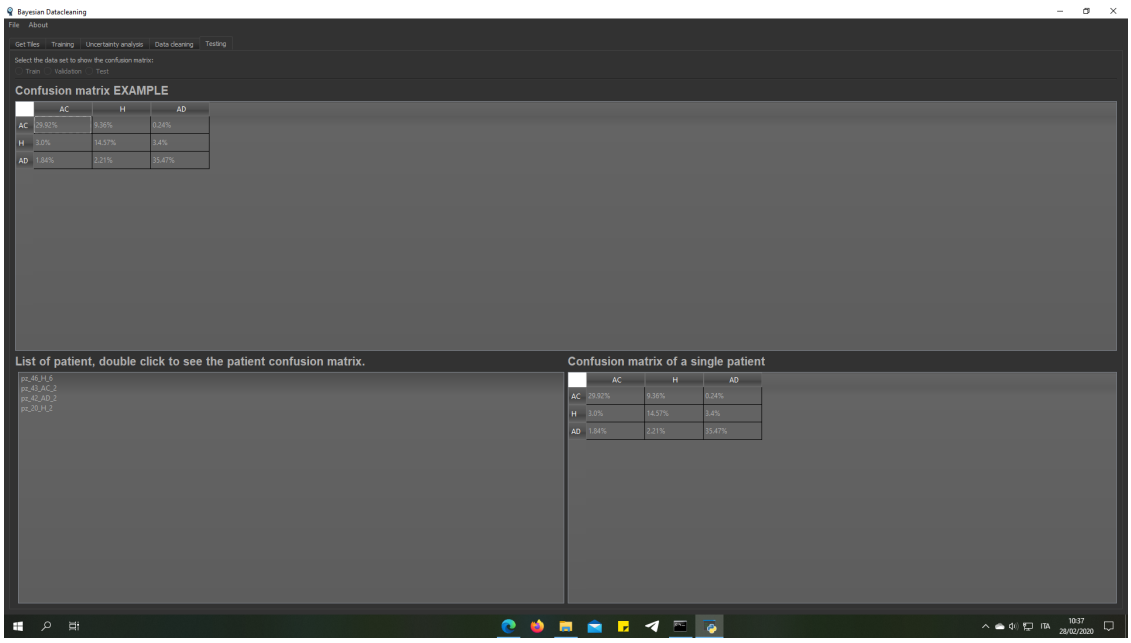


Figura 5.11. Pagina dedicata alla fase di testing.

Attraverso l'ultima pagina del software per il datacleaning *Test*, è possibile consultare le confusionmatrix dei vari data set. La visualizzazione sarà accessibile soltanto ad avvenuta classificazione nella pagina *Uncertainty analysis*, questo processo attiverà il "check box" nella parte alta.

Una volta selezionato quale dataset si desidera analizzare, è possibile visualizzare nella tabella appena sottostante la matrice generale.

Nella sezione inferiore è presente una lista contenente i file che sono presenti in quel dataset. Cliccando due volte su una di queste voci sulla destra verrà mostrata la confusionmatrix del singolo paziente.

5.3.2 Struttura del programma di datacleaning

In questo paragrafo verranno esplicate, soprattutto dal punto di vista della programmazione, tutte le scelte e le metodologie utilizzate per la realizzazione dell'applicativo di datacleaning.

Partendo dalla pagina *Get tiles*, una volta cliccato sul pulsante per avviare l'analisi, verranno create tre cartelle all'interno di quella nella quale si desidera salvare tutti i dati che scaturiranno dal software.

Le tre cartelle saranno rinominate AC, AD e H e conterranno i rispettivi set di immagini.

Grazie alla classe di PyQt5 *QThread* è possibile gestire in maniera intelligente i processi. In totale per il processo di creazione delle immagini verranno avviati nove thread. Nel caso in cui questi processi vengano avviati su una macchina con specifiche non sufficienti si creerà una coda di esecuzione che avrà come unico lato negativo il prolungamento dei tempi di esecuzione ma il processo comunque verrà portato a termine.

Passando alla pagina *Training* è importante dire che la possibilità di visualizzazione in tempo reale dei risultati è possibile grazie alla creazione di una callback da parte del candidato che permette quindi questo scambio di informazioni durante il processo di allenamento. Keras infatti mette a disposizione apposite librerie per la creazione di "chiamate" durante il processo di allenamento.

Al termine del processo di training verrà salvato nella cartella di lavoro il modello in formato "h5" con il seguente nome: *Model_{Drop oppure Kl}_{timestamp}.h5*. In questo modo sarà possibile allenare diverse reti che avranno un nome diverso.

Oltre al modello è possibile trovare nella cartella del dataset analizzato anche un file di testo con all'interno un dizionario json avente due chiavi "acc" e "loss". Questo file potrà essere utilizzato nel caso in cui, terminato il training, si vogliano rianalizzare i valori di accuratezza e della funzione obiettivo.

Il modello una volta salvato verrà caricato quando l'utente, passando alla pagina successiva, vorrà classificare uno dei tre dataset.

Al termine dell'analisi il file json generato avrà la seguente struttura:

Come è possibile vedere dalla figura 5.12, oltre ai campi già presenti nel file json derivante dall'applicativo di analisi, si aggiunge "name", che indica il nome del file svs a cui è collegata la tile in analisi e "true_class", cioè la reale classe di appartenenza dell'immagine.

Grazie a questo file di testo nelle pagine *Data cleaning* e *Test* è possibile visualizzare gli istogrammi e anche le confusionmatrix, sia quella generale del dataset che quelle per paziente.

```
{
    "0": {
        "name": "pz_42_AD_2",
        "true_class": "AD",
        "im_path": "C:/Users/piero/Documents/test\\train/
           AD\\pz_42_AD_2_tile_0_0_0.png",
        "shape_x": 64,
        "shape_y": 64,
        "col": 0,
        "row": 0,
        "pred_class": "AC",
        "epi": 0.0028,
        "ale": 0.1229
    }
}
```

Figura 5.12. Singolo elemento file json derivante dal datacleaning.

Capitolo 6

Conclusioni

Il filo conduttore che collega tutti gli argomenti trattati in questa tesi è lo studio dell'incertezza e, conseguentemente, le modalità attraverso le quali è possibile adoperare questa nuova tipologia di informazioni, fino ad ora, non facilmente consultabili con i sistemi ad apprendimento automatico.

Attraverso tecniche come il drop out o la KL divergence è possibile calcolare sia l'incertezza epistemica che l'incertezza aleatoria. In questo modo oltre all'incertezza totale si può effettuare uno studio particolareggiato sull'incertezza dovuta al modello e quella generata dal rumore presente nei dati.

Per la consultazione di queste informazioni è stato realizzato un framework che ha permesso di creare due applicativi atti a mostrare in pochi semplici passaggi quanto discusso a livello teorico precedentemente.

Dallo studio si può concludere che i sistemi bayesiani possono far compiere al deep learning un grande passo in avanti. Si è notato che attraverso le metodologie suddette, è possibile individuare quando una predizione probabilmente non è corretta, considerando proprio il valore di incertezza totale 4.1.

Questa capacità è molto importante nei casi in cui non sono ammessi errori, come ad esempio nel campo medico, ma anche in ambiti che stanno acquistando sempre più importanza come nel caso della guida autonoma.

E' possibile, utilizzando le reti bayesiane, compiere datacleaning su un dataset, nella maggior parte dei casi all'interno dei dati utilizzati sono presenti dei campioni spuri erroneamente assegnati ad una classe oppure campioni rumorosi.

Applicare questa tecnica all'insieme di immagini a disposizione ha permesso inoltre di avere un significativo miglioramento delle prestazioni in accuratezza. La funzione di "pulizia" dei dati può oltretutto essere molto utile nel caso di creazione di data warehouse, in questo modo si potranno archiviare soltanto informazioni utili.

Durante lo studio, tuttavia, sono state evidenziate delle limitazioni in particolare per quanto riguarda la tecnica che ricorre all'utilizzo della Kullback–Leibler divergence. Le problematiche riscontrate sono principalmente due: la prima è il tempo necessario per concludere il training, la seconda invece riguarda l'ottimizzazione dei parametri.

Per svolgere il processo di apprendimento in tempi ragionevoli bisognerà o aumentare la potenza di calcolo dell'hardware attualmente esistente, oppure ampliare la compatibilità di tecniche come ad esempio i TFRecords, in modo tale da velocizzare il processo di lettura quanto più possibile.

Al momento sono già disponibili delle metodologie per ottimizzare la velocità del processo di training dei modelli, ma purtroppo sia la compatibilità con altre librerie che la carenza di informazioni sull'utilizzo delle suddette tecniche, rende complessa l'applicazione delle stesse.

Per quanto concerne la seconda questione, è stato necessario svolgere svariati tentativi prima di poter trovare una combinazione corretta dei parametri caratteristici del modello che permettessero allo stesso di raggiungere la convergenza.

In conclusione, dagli studi compiuti si può affermare che l'interesse crescente intorno ai sistemi bayesiani porterà sicuramente negli anni a venire grossi sviluppi sia dal punto di vista teorico che nell'ambito delle applicazioni pratiche.

6.1 Sviluppi futuri

Il paradigma seguito per la progettazione del framework che si è sviluppato è la modularità. E' possibile pertanto rimuovere o aggiungere nuove componenti rispettando gli input dei vari metodi senza la necessità di apportare troppe modifiche al codice base.

Di seguito verranno discusse delle possibili funzionalità che potranno essere facilmente implementate all'interno del framework e quindi sia nello strumento atto all'analisi, sia per quello per il datacleaning.

Una possibile implementazione di cui possono beneficiare entrambi gli applicativi è il miglioramento del processo di creazione delle tiles. Nelle due

interfacce grafiche, infatti, non sono disponibili molte opzioni per modificare i parametri che regolano questa prima fase.

La libreria *Openslide* mette a disposizione la possibilità di selezionare la grandezza della finestra con la quale si vuole andare a ritagliare l'immagine presente a un determinato livello del file piramidale. Sarebbe quindi interessante lasciare la scelta del livello e la grandezza della finestra all'utente, così da poter studiare anche l'effetto di questi parametri sulle prestazioni dei modelli di deep learning.

Un'altra opzione disponibile della suddetta libreria è la possibilità di creare le tiles con un determinato overlap, come risultato si avranno, quindi, delle immagini con delle sezioni uguali. Questa implementazione potrebbe risultare particolarmente utile nello strumento di data cleaning, in quanto se si è in presenza di un dataset di dimensioni modeste, attraverso la generazione di tiles con sovrapposizione, si potrà ottenere un dataset più vasto.

Osservando nello specifico l'applicativo dedicato al data cleaning, nella pagina dedicata al training, potrebbe essere realizzata un'apposita sezione per modificare i parametri dei modelli, senza dover quindi ricorrere a modifiche nel codice. Nella suddetta area potrebbe essere presente la possibilità di modificare:

1. il numero di layer: in questo modo si potrà modificare la struttura propria della rete, adattando la complessità della stessa a seconda delle necessità;

2. la dimensione del kernel: negli strati di convoluzione, questo è un parametro molto importante che può inficiare sulle prestazioni del modello di deep learning;

3. lo stride;

4. i parametri del max pooling: variando anche in questo caso la dimensione del filtro oppure la possibilità di disattivarlo per un determinato layer;

5. il batch normalization: attivarlo o disattivarlo dopo un layer di convoluzione.

Passando alla pagina rinominata *Data cleaning* dell'omonimo software, considerata la struttura del file json 5.12 che si ottiene come risultato dalla classificazione di un set di immagini, è possibile pensare a nuove visualizzazioni. Un esempio potrebbe esse l'individuazione del paziente con tiles più incerte.

Analizzando invece le possibili implementazioni riguardanti lo strumento di analisi saranno possibili miglioramenti soprattutto per quanto riguarda la visualizzazione dei risultati.

Un possibile sviluppo potrebbe essere quello di creare un visualizzatore che permetta di modificare in tempo reale la soglia con la quale una determinata area dell'immagine è ritenuta incerta, la scelta in questo modo sarebbe gestita dall'utente.

Sarebbe inoltre utile aggiungere una visualizzazione già presente per lo strumento di datacleaning: l'istogramma delle incertezza. In questo modo sarebbe di facile intuizione la robustezza generale delle predizioni ottenute.

Bibliografia

[1] Yarin Gal, Zoubin Ghahramani, *Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning*, 2015.

[2] Kumar Shridhar, Felix Laumann, Adrian Llopart Maurin, Martin Olsen, Marcus Liwicki*Bayesian Convolutional Neural Networks with Variational Inference*, 2018

[3] Ioffe, Sergey; Szegedy, Christian (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.*

[4] World Cancer Report 2014, World Health Organization, 2014, pp. Chapter 1.1, Fig. 1.1.4. Estimated world cancer incidence proportions by major sites, in both sexes combined, in men, and in women, 2012.

[5] World Cancer Report 2014, World Health Organization, 2014, pp. Chapter 1.1, Fig. 1.1.12. Europe. Estimated cancer incidence proportions by major sites, in both sexes combined, in men, and in women, 2012.

[6] Fabio A. Spanhol, Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte, 2016, *A Dataset for Breast Cancer Histopathological Image Classification.*

[7] Jun Xu a,n, Xiaofei Luo a, Guanhao Wang a, Hannah Gilmore b, Anant Madabhushi, 2016,*A Deep Convolutional Neural Network for segmenting and classifying epithelial and stromal regions in histopathological images.*

[8] John Arevaloa, Angel Cruz-Roaa, Viviana Ariasb, Eduardo Romeroc, Fabio A. González, 2015,*An unsupervised feature learning framework for basal cell carcinoma image analysis.*

[9] *Overview and Future Promise of an Exciting New Technique*, 2016, Hayit Greenspan, Bram van Ginneken, Ronald M. Summers.

[10] *Weight Uncertainty in Neural Networks*, 2015, Charles Blundell Julien Cornebise, Koray Kavukcuoglu, Daan Wierstra.

[11] *ScanNet: A Fast and Dense Scanning Framework for Metastatic Breast Cancer Detection from Whole-Slide Images*, 2017, Huangjing Lin1, Hao

Chen1,2, Qi Dou1, Liansheng Wang3, Jing Qin4, Pheng-Ann Heng1.
[12] Sezione del sito dell'AIRC sul cancro del colon-retto.
[13] Sito Intel, https://www.intel.it/content/www/it/it/products/processors/core/i7-processors/i7-7700.html