UNIVERSITÀ DI PARMA

Dipartimento di Ingegneria e Architettura

Corso di Laurea in Ingegneria Informatica

Analisi del segnale pressorio proveniente dal microcircolo per la diagnosi preventiva della sepsi

Use of microcirculation pressure signal analysis for preemptive diagnosis of sepsis in ICU

Relatore:

Prof. Stefano Cagnoni

Tesi di Laurea di:

Federico Zucchi

ANNO ACCADEMICO 2020-2021

Indice

Not all who wander are lost

J.R.R. Tolkien

Ringraziamenti

Ringrazio il mio relatore, il professor Cagnoni, per avermi dato la possibilità di svolgere questo lavoro di Tesi ed avermi supportato e consigliato lungo tutto lo sviluppo del progetto.
Ringrazio il professor Bocchi dell'Università di Firenze per avermi fornito il dataset.
Ringrazio la dottoressa Bignami per il supporto medico e la preziosa disponibilità.
Ringrazio tutta la mia famiglia per avermi sempre sostenuto e saggiamente consigliato durante tutto il percorso, senza di voi non sarei qui ora.
Ringrazio i Pensionati dell'Olmo ed i borettesi: Zio, Nixon, Pamp, Lollo, Catz, Barbio, Mafo, Massi e Simo per la preziosa amicizia
Ringrazio tutti i compagni di università in particolare Sacco e Matti per aver reso le tante ore di studio più sopportabili
Ringrazio tutti gli altri amici: Alle, Chiara, Gio, Gibbo ed Alessia con i quali siamo riusciti a vederci poco nell'ultimo periodo ma, nonostante ciò, mi hanno sempre dimostrato grande vicinanza.
Ringrazio Martina che mi è stata vicina in quest'ultimo e duro periodo della mia carriera universitaria supportandomi sempre con immensa pazienza.

Capitolo 1

Presentazione della Tesi

Il lavoro di Tesi tratta dell'analisi e dello sviluppo di un modello di deep learning che sfrutta principalmente strati di tipo LSTM (Long Short-Term Memory) per la predizione dell'insorgenza di sepsi in pazienti in terapia intensiva.
La sepsi è un problema sanitario tanto importante quanto complesso da individuare nelle sue fasi iniziali e colpisce dal 13% al 39% dei pazienti ricoverati. Essa è mortale in più del 30% dei pazienti nei quali si manifesta. Per questo, riuscire ad anticiparne i sintomi è di fondamentale importanza per il personale sanitario che può attuare un trattamento farmacologico atto a bloccarla prima che raggiunga stadi avanzati.
Molti dei lavori oggi presenti in letteratura cercano di predire l'insorgenza di sepsi analizzando i cosiddetti EHR (electronic health records) dei pazienti. Essi sono valori discreti, spesso disponibili come media su periodi temporali di alcuni minuti, che descrivono lo stato di salute del pazienti. Queste metriche sono state estensivamente testate ed in letteratura, ad oggi, vi sono diversi modelli di ML che ne fanno uso per predire la sepsi, anche se, molto spesso, non fanno altro che sintetizzare in un modello le relative linee guida per la diagnosi, utilizzandole a priori per definire la ground truth nel training set. In questo lavoro si è cercato di raggiungere lo stesso obiettivo a partire dai soli segnali fisiologici dei pazienti acquisiti durante il loro monitoraggio.

In particolare, sono state analizzate le forme d'onda dei segnali pressori (fotopletismografici, o PPG) acquisiti mediante pulsiossimetri da dito. Questi dispositivi misurano, in modo pratico e non invasivo, il volume di sangue presente ad ogni istante nella parte più superficiale del sistema circolatorio: il microcircolo.
Il dataset utilizzato è MIMIC-III Waveform Database Matched Subset: un insieme di 22137 registrazioni di segnali fisiologici raccolti da circa 10282 pazienti in terapia intensiva fra il 2001 e il 2012 al Beth Israel Deaconess Medical Center (BIDMC, Boston, MA, USA). Questi pazienti sono stati divisi in due gruppi: uno contenente pazienti deceduti in ospedale per sepsi e l'altro, di controllo, contenente pazienti ricoverati per malattie mentali, ipotizzando che questi ultimi non avessero disturbi riconducibili alla sepsi e potessero dunque fungere da gruppo di controllo per la costruzione di un classificatore. I segnali PPG seguono, generalmente, in ogni ciclo cardiaco, un andamento formato da due picchi corrispondenti alla fase sistolica, in cui il cuore si contrae, e quella diastolica, dove il muscolo cardiaco si rilassa.
L'individuazione di questi picchi nel segnale è di cruciale importanza, in quanto la loro rispettiva posizione e la loro forma forniscono intrinsecamente informazioni sullo stato di salute del paziente e sono forti indicatori dell'insorgenza di sepsi.
I pulsiossimetri da dito sono strumenti non invasivi e perciò molto sensibili al rumore prodotto, soprattutto, da movimenti del paziente o interferenze esterne. Per questo motivo, prima di effettuare qualsiasi analisi sul segnale PPG è necessario applicare delle tecniche di pulizia dei dati. In questa tesi si sono testate diverse metriche statistiche come la curtosi, la skewness ed il confronto, tramite Dynamic Time Warping, con un ciclo ideale appositamente generato come somma di sinusoidi. Tutte queste metriche si sono rivelate ottimi indicatori della qualità del singolo ciclo. Tuttavia, per la variazione fisiologica della frequenza cardiaca, il calcolo dei valori di tali metriche richiede una preelaborazione e una accurata segmentazione in cicli del segnale, per poter ottenere la quale sono richieste le stesse caratteristiche di 'pulizia'

del segnale che l'applicazione delle metriche vorrebbe garantire. Per questo motivo, a partire dall'osservazione dei segnali nel dataset, si è sviluppo un algoritmo di pulizia ad hoc basato sull'identificazione dei soli picchi sistolici o diastolici e, successivamente, sull'eliminazione di parti del segnale dove si riscontrano ampie deviazioni, in termini di distanza reciproca o ampiezza, dei picchi rispetto a condizioni compatibili con la fisiologia umana.
Per quanto riguarda la classificazione dei segnali, si è sviluppato un autoencoder, basato su LSTM, allenato sui segnali dei pazienti di controllo. Un autoencoder è un modello di machine learning addestrabile in modo autosupervisionato, suddivisibile in un encoder che comprime i dati in ingresso ed un decoder che, partendo dai dati compressi, cerca di ricostruire il segnale di partenza nel modo più fedele possibile, minimizzando il cosiddetto errore di ricostruzione. A seguito dell'allenamento, l'autoencoder crea una sua rappresentazione interna compressa dell'input. Questa rappresentazione è stata utilizzata come base di confronto per definire il concetto di "paziente non settico" e poterlo rapportare ai casi in cui sia presente la sepsi.
I segnali non puliti dei pazienti settici e di controllo sono stati divisi in segmenti di quaranta secondi e poi forniti in input all'autoencoder per valutare se vi fossero delle differenze, in termini di funzione di distribuzione di probabilità (PDF) dell'errore di ricostruzione dei segmenti, fra i due gruppi. Ciò è stato fatto allenando un classificatore XGBoost su una rappresentazione, ottenuta tramite PCA, delle distribuzioni dell'errore relative ai singoli pazienti. L'F1-Score ottenuto dal classificatore in cross-validation è pari al 61%, dato insufficiente per la diagnosi ma che conferma l'ipotesi che l'autoencoder ricostruisca con diversa fedeltà i due gruppi di pazienti e che questi, quindi, siano effettivamente discriminabili a partire da un modello basato sull'insieme di controllo. Inoltre, l'andamento dell'F1-Score in funzione del numero di componenti principali utilizzate dimostra che l'autoencoder può essere utilizzato come metodo di pre-processing per i segnali. Si è verificato infatti che i tracciati per i quali l'errore di ricostruzione supera una certa soglia sono caratterizzati da assenza di segnale o da un livello di rumore tale da causare

un forte deterioramento delle prestazioni del classificatore.
Tutti i segnali dei pazienti settici e di controllo caratterizzati da rumore al di sotto della soglia individuata per l'errore di ricostruzione dell'autoencoder, sono stati poi utilizzati per lo sviluppo di un classificatore basato su reti LSTM. Quest'ultimo ha ottenuto un'accuratezza sul test-set, bilanciato fra pazienti settici e di controllo, pari al 76% permettendo quindi di distinguere in modo soddisfacente i due gruppi.
In futuro si prevede di ampliare il modello includendo i dati clinici dei pazienti oltre a valutare un'analisi nel tempo dello scostamento del segnale rispetto alle condizioni di normalità, per evidenziare in modo precoce l'insorgenza della sepsi e consentire un intervento tempestivo, condizione fondamentale per la sopravvivenza del paziente.

1.1 Struttura dell'elaborato

Nel Capitolo 2 viene presentato un excursus teorico sul problema della sepsi e come, ad oggi, è stato affrontato in letteratura.
Il Capitolo 3 approfondisce i principali strumenti tecnici e teorici che sono stati utilizzati nella Tesi.
Nel Capitolo 4 viene descritto il dataset utilizzato per le analisi.
Il Capitolo 5 affronta le problematiche del dataset ed i metodi di pulizia del segnali fisiologici.
Nel Capitolo 6 è riportato il modello di AI sviluppato ed i relativi risultati sul dataset.
Il Capitolo 7 trae le conclusioni del lavoro di Tesi ed indica alcuni possibili sviluppi futuri.

Capitolo 2

Introduzione

Grazie alle moderne tecnologie in ambito medico vengono raccolti sempre più dati provenienti da molteplici strumenti. In particolare, i dati acquisiti nei reparti di terapia intensiva, *intensive care units (ICU)* in inglese, comprendono una vasta gamma di informazioni sullo stato di salute del paziente. Questi dati possono essere poi utilizzati per lo sviluppo di intelligenze artificiali per assistere i medici nella diagnosi di complicanze. In particolare, nelle *ICU*, vengono raccolti due tipi di dati: gli *electronic health records (EHRs)* e le forme d'onda derivanti dal monitoraggio dei parametri vitali dei pazienti. I primi sono dati discreti e sono già stati studiati a lungo mentre i secondi sono stati esplorati in minor misura ma offrono una quantità molto elevata di informazioni sullo stato del paziente e meritano, dunque, un approfondimento. In questo lavoro ci siamo concentrati su un problema di particolare interesse medico: la sepsi. Quest'ultima è definita come disfuzione organica, che può portare a morte, innescata da una risposta disregolata del corpo ad un'infezione [28]. La sepsi è un problema sanitario di elevatissima importanza [24]: essa ha un'incidenza che varia dal 13.6% al 39.3% nei pazienti ricoverati ed è mortale dal 25.8% al 35.3% dei casi a seconda delle diverse aree geografiche [25]. Nel tempo sono state formulate numerose defizioni di sepsi; in questo lavoro è stata considerata la definizione *Sepsis 3* [28], la più aggiornata e quella maggiormente adottata.

Negli ultimi 20 anni i reparti di terapia intensiva sono stati indicati come quelli dove l'intelligenza artificiale (AI) può maggiormente contribuire ad aiutare i medici nella diagnosi precoce di patologie [13]. Inoltre, in questi particolari e critici reparti vengono acquisite enormi quantità di dati che spesso non vengono conservate e vanno perse [4].
Come già detto sono stati diversi i tentativi di prevedere l'insorgenza di sepsi utilizzando gli EHR[21]. Alcuni di questi studi hanno dato ottimi risultati anticipando lo sviluppo di sintomi severi di sepsi di 3 o 4 ore sfruttando diversi parametri vitali fra i quali[9]:

1. Battito cardiaco
2. Temperatura corporea
3. Pressione sistolica e diastolica
4. SpO2
5. Numero di globuli bianchi
6. Età

In questo contesto sono stati testati vari modelli di AI . Nella maggior parte dei casi la metrica utilizzata per misurare le prestazioni del sistema informatico è l'AUROC (*Area Under Receiver Operator Curve*).
In particolare Kam et al. [15] propongono un approccio, tramite reti *LSTM* sul dataset MIMIC-II, per la previsione dell'insorgenza di sepsi con 3 ore di anticipo sfruttando 9 variabili vitali: la pressione sistolica e diastolica, la frequenza del battito cardiaco, la temperatura corporea, il ritmo respiratorio, il numero di globuli bianchi, il pH, l'ossigenazione del sangue e l'età. Questo studio ha ottenuto un AUROC pari a 0.929, superando quello di *InSight* [3]: noto modello per la previsione di sepsi estensivamente testato su dataset come MIMIC-III [5][17] e CRMC[19].
Un fondamentale problema di questi approcci che sfruttano gli EHRs per la predizione di sepsi è quello della *circolarità*, come segnalato da Schamoni et

al.[26]: considerando che non è possibile conoscere l'esatto momento in cui la sepsi è cominciata, per approssimare un punto di partenza plausibile si utilizzano dei criteri clinici come, per esempio, Sepsis-3 [28]. Utilizzare gli stessi criteri clinici che si usano per la diagnosi per assegnare le label ai dati del *training set* di un sistema di *machine learning* può essere circolare. Questo comporta il rischio di non apprendere nuova conoscenza sui dati ma solo replicare i criteri di classificazione iniziali. Nel migliore dei casi un classificatore replicherebbe le linee guida per la definizione di sepsi senza riconoscere nuove caratteristiche realmente importanti per il riconoscimento preventivo della patologia.
Un approccio che elabora solo le forme d'onda, come quello presentato in questo lavoro, è esente da questa circolarità in quanto le informazioni usate per la definizione di sepsi non sono esplicitamente disponibili nei dati da elaborare ma devono essere estratte a partire da conoscenze di più alto livello.

Capitolo 3

Stato dell'arte

In questo capitolo sono presentate le tecnologie e le tecniche adottate per lo sviluppo del progetto. Vengono introdotte le due principali librerie Python utilizzate, PCA per l'analisi delle componenti discrete, le reti *Long-Short Term Memory (LSTM)* per l'analisi di sequenze di dati ed una particolare architettura di AI: gli autoencoder.

3.1 Librerie Utilizzate

Per la realizzazione di questo lavoro è stato utilizzato il linguaggio *Python* ed alcune librerie che offrono funzioni di alto livello per la lettura dei segnali fisiologici, l'elaborazione di tali dati e l'implemetazione dei modelli di AI che saranno descritti nel Capitolo 6 e mediante i quali si sono classificati i pazienti. Descriveremo ora, brevemente, le principali librerie e funzione utilizzate.

3.1.1 WFDB

La libreria *Waveform Database Software Package* (WFDB)[1] per Python è una raccolta di funzioni per elaborare ed analizzare i segnali fisiologici. Le

[1] https://www.physionet.org/content/wfdb-python/3.4.1/

componenti principali della libreria Python sono ispirate alla versione originale in C ma presentano alcune differenze nell'implementazione oltre alla mancanza di alcuni funzioni, come *wabp* che permette di individuare i picchi nei segnali PPG/ABP, disponibile in C ma non in Python. La versione C offre vantaggi in termini di velocità ma risulta meno intuitiva. Per quanto riguarda il lavoro svolto, la libreria WFDB è stata utilizzata per la lettura dei segnali in quanto la loro memorizzazione segue una struttura specifica e la lettura richiede funzioni ad hoc. Il risultato della lettura è un *array* della libreria *Numpy* [2]

3.1.2 Keras

Keras[3] è una libreria open-source che fornisce un'interfaccia Python per lo sviluppo di reti neurali. In questo lavoro è stato utilizzata, come *backend*, la libreria *TensorFlow*[4].
Keras contiene implementazioni dei più comuni tipi di layer, funzioni di attivazione e componenti delle reti neurali. Oltre a queste offre supporto per modelli di AI meno comuni, in particolare le reti neurali ricorrenti LSTM utilizzate in questo progetto e descritte nella Sezione 3.4.

3.2 PCA

L'analisi delle componenti principali (*principal component analysis* o *PCA* in inglese) è una tecnica atta alla riduzione del numero di variabili che descrivono un insieme di dati che minimizza la perdita di informazioni. In termini statistici lo scopo è quello di creare un nuovo insieme di variabili non correlate che massimizzano la varianza.
PCA è definita come un trasformazione lineare ortogonale che esprime le variabili di partenza secondo un sistema di coordinate tale che la variabile

[2]https://numpy.org/doc/stable/index.html
[3]https://keras.rstudio.com/
[4]https://www.tensorflow.org/

con la maggiore varianza viene proiettata sul primo asse, quella seconda per varianza sul secondo asse e così via.
Per calcolare il nuovo spazio vettoriale, si utilizza la matrice di covarianza di $\mathbf{x}_i$ che è l'insieme di variabili dello spazio di partenza. Per prima cosa si individuano gli autovalori della matrice di covarianza, il cui numero corrisponderà al massimo fra il numero delle variabili e il numero delle istanze comprese nel dataset da cui vengono derivti. L'autovalore più grande corrisponde alla dimensione $\mathbf{w_1}$ che conserva la maggior varianza o prima componente principale. Il secondo autovalore per quantità di varianza alla seconda componente e così via.
Per ognuno degli autovalori viene calcolato il corrispondente autovettore in modo da ottenere la base su cui proiettare i dati per ottenere le nuove variabili $\mathbf{w}$. Questa matrice appena ottenuta a partire dagli autovettori è definita matrice di rotazione V. Per ottenere le nuove variabili $w_1, w_2, \ldots, w_n \in W$, nel nuovo spazio vettoriale, a partire da quelle originali $x_1, x_2, \ldots, x_n \in X$ si effettua l'operazione matriciale $W = V \bullet X$.
A livello intuitivo si tratta di trovare la direzione lungo la quale proiettare lo spazio di partenza in modo da preservare la massima varianza o, in altre parole, individuare l'asse lungo il quale è possibile tracciare un ellissoide contenente il maggior numero di punti. Una volta trovato il primo se ne cerca un altro ortogonale al primo che soddisfa la stessa condizione rispetto alla varianza rimanente e così via. Un esempio è visibile in Figura 3.1. A sinistra è rappresentato un dataset insieme a tre differenti assi. A destra invece il risultato della proiezione del segnale originale sui corrispondenti assi. Si osserva che la proiezione sulla linea continua presenta massima varianza, quella sulla linea tratteggiata meno ed infine sulla linea punteggiata si ha minima varianza.
In ultima analisi decidere il numero di componenti o equivalentemente la quantità di varianza da mantenere significa scegliere lungo quanti e quali assi calcolare la proiezione nello spazio di arrivo rispetto a quello di partenza tenendo conto che il valore degli autovalori è proporzionale alla varianza

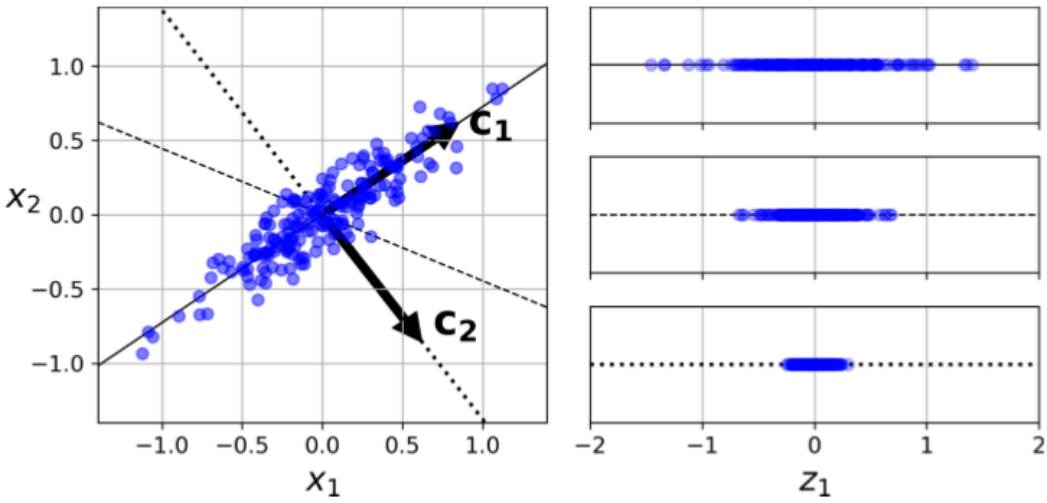


Figura 3.1: Selezione del sottospazio lungo il quale proiettare [10]

rilevabile lungo il corrispondete autovalore. Questo consente di garantire la massima conservazione di informazioni se, fissato N, si selezionano gli N autovettori corrispondenti agli N autovalori più grandi.

3.3 Recurrent neural network

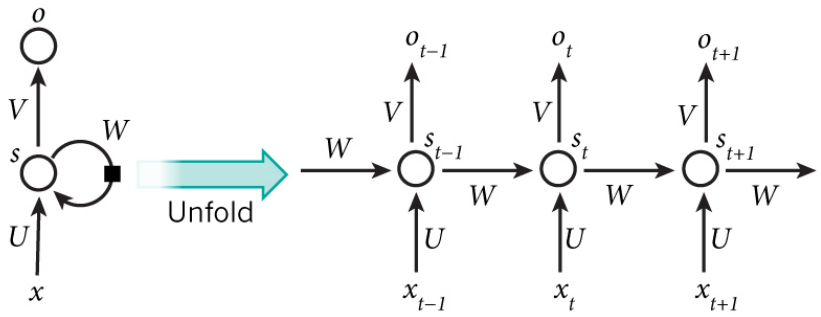


Figura 3.2: Dispiegamento di una RNN

Le reti neurali classiche *Multilayer perceptron (MLP)* non hanno memoria degli stati passati ed ogni neurone modifica il proprio stato indipendentemente dagli ingressi precedenti.

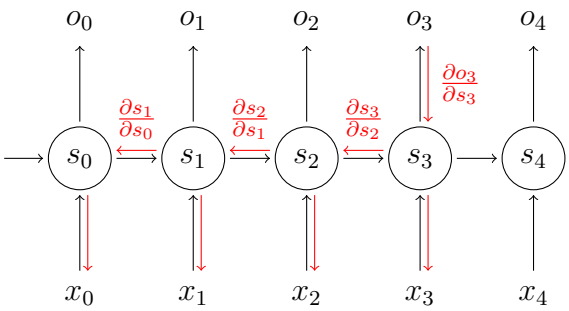


Figura 3.3: Esempio di passo di BPTT. Ad ogni input vengono calcolate le derivate parziali rispetto allo stato corrente e tutti quelli precedenti. L'errore al tempo t viene quindi propagato anche a tutti gli elementi precedenti

Una tipologia di reti neurali che tiene conto del fattore temporale della sequenza di input sono le *recurrent neural network (RNNs)*[8]. Una rete ricorrente è un sistema dinamico con *feedback*, che può essere dispiegato nel tempo come in Figura 3.2 per ottenere una MLP equivalente. E' possibile, infatti, approssimare tale sistema con una catena di elementi *feed-forward* attraverso cui si propaga lo stato. Tale catena è lunga in modo proporzionale alla durata della memoria del sistema. Il metodo di apprendimento è la backpropagation. In questo caso l'architettura della rete contiene intrinsecamente l'informazione temporale. Inoltre, il gradiente ad ogni istante dipende dallo stato corrente e da tutti quelli precedenti. Il metodo di apprendimento è chiamato *Backpropagation Through Time (BPTT)* e procede come mostrato in Figura 3.3. Il comportamento della rete ricorrente è modellato dalle seguenti equazioni:

$$\begin{aligned} s^t &= f(Ux^t + Ws^{t-1}) \\ o^t &= Vs^t \end{aligned} \tag{3.1}$$

dove le matrici U, V, W costituiscono rispettivamente i pesi delle connessioni *input-to-hidden*, *hidden-to-output* e *hidden-to-hidden*.

3.4 LSTM

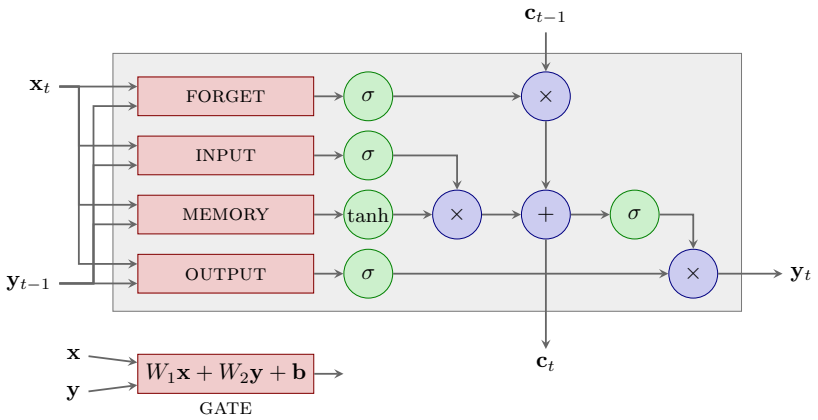


Figura 3.4: Struttura di una cella *LSTM*

Le *RNN* discusse nella Sezione 3.3 riescono potenzialmente ad avere memoria degli stati precedenti ed apprendere dipendenze in sequenze indefinitamente lunghe. Nella realtà questo non è vero a causa del *vanishing gradient*, problema accuratamente analizzato da Bengio et al. [2]. In sintesi, questa difficoltà è motivata dal fatto che per $\tau \ll t$, $\left|\frac{\partial C_t}{\partial a_\tau}\frac{\partial a_\tau}{\partial W}\right| \to 0$, in altre parole il gradiente ha maggior peso sugli stadi vicini e tende a 0 per quelli più distanti. Nel tentativo di mitigare questa problematica si sono sviluppate le *Long Short Term Memory (LSTM)*. Quest'ultime sono RNN dove al tempo t non corrisponde un singolo neurone ma una struttura più complessa. Nella forma più classica ogni cella è composta da 3 elementi o *gate*:

1. **Forget Gate**: essa si occupa di decidere se mantenere o dimenticare l'informazione proveniente dall'istante precedente

2. **Output Gate**: misura quanto i valori della cella contribuiscono all'uscita della rete *LSTM*

3. **Input Gate**: misura l'importanza delle informazioni che la cella riceve in input

In termini matematici le equazioni che descrivono una *LSTM*, con i suoi *gate*, sono le seguenti:

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(W_n x_t + U_n h_{t-1} + b_n) \\ h_t &= o_t \circ \tanh(c_t) \end{aligned} \tag{3.2}$$

Dove c è il canale di carry che permette alle celle LSTM di comunicare fra loro e $\circ$ rappresenta il prodotto elemento per elemento.

3.5 Autoencoder

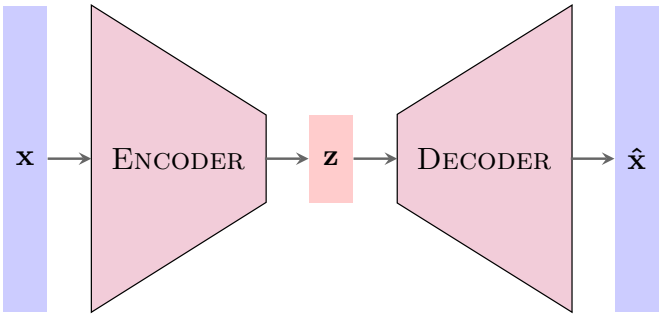


Figura 3.5: Struttura di un *autoencoder*

Gli autoencoder sono particolari tipologie di reti neurali dove l'input corrisponde all'output desiderato. Il loro obiettivo è quello di ridurre la dimensionalità dell'input e, a partire da questa rappresentazione, ricostruire l'ingresso. La parte della rete che si occupa di "comprimere" l'input è chiamata *encoder*, la parte che cerca di ripristinarlo è denominata *decoder*. La

rappresentazione intermedia viene definita come *spazio latente* e può essere interpretata come un "codice" più compatto con il quale la rete rappresenta l'ingresso. In Figura 3.5 è riportato graficamente quanto appena descritto.
Una volta allenata la rete è possibile utilizzare l'*encoder* per ottenere un sistema di compressione. La capacità di quest'ultimo di comprimere l'informazione contenuta nell'input è però fortemente dipendente dai dati utilizzati per il training dai quali sono state estratte le *feature* oltre ad essere *lossy*: l'output dell'encoder sarà comunque una rappresentazione approssimativa dell'input.

3.5.1 Autoencoder con LSTM

In questa sezione discuteremo una tipologia particolare di autoencoder utilizzata nel sviluppo del progetto: un modello *encoder-decoder* basato su reti LSTM.
Facciamo un esempio di autoencoder formato da due *layer* LSTM per l'encoder ed altri due per il *decoder*. I parametri citati in questo paragrafo fanno rifermento alla libreria Keras descritta nella Sezione 3.1.2 e con la quale si è sviluppato il modello.
L'input della prima LSTM è un tensore 3-D nella forma `[batch×timesteps×feature]`, in output vi sarà un tensore di dimensione `[batch×timesteps×units]`, dove `units` è il numero di unità LSTM.
Essendo il primo layer connesso ad un secondo strato LSTM il parametro `return_sequence` è impostato a `True` in modo che ogni cella generi un output ad ogni *timestep* e non solo nell'istante finale di elaborazione dei dati.
L'uscita della seconda LSTM dell'encoder produce un vettore di dimensione uguale al numero di celle di quest'ultima e comunque minore rispetto al numero di *feature* iniziali. Esso è la rappresentazione codificata delle *feature* in ingresso cioè, il sopracitato spazio latente. Questo passa attraverso un *layer* `RepeatVector` di Keras che lo replica per un numero di volte pari al numero di *timestep* iniziali in modo da ottenere un tensore tridimensionale per l'ingresso del primo layer LSTM del decoder. Quest'ultimo è identico

all'encoder ma con le due reti LSTM scambiate.
Il tensore in uscita viene moltiplicato per una matrice di livelli *Dense* per riportarlo alla dimensione iniziale. Il prodotto avviene vettore per vettore, ovvero ad ogni timestep, grazie al layer `TimeDistributed`.

Capitolo 4

Analisi del dataset

4.1 Descrizione

Il dataset utilizzato per lo sviluppo del progetto è *MIMIC-III Waveform Database* [1] che contiene 67830 registrazioni di segnali fisiologici raccolti da circa 30000 pazienti in terapia intensiva fra il 2001 e il 2012 al *Beth Israel Deaconess Medical Center (BIDMC, Boston, MA, USA)*. In particolare è stato utilizzato un sottoinsieme di questi dati, proveniente da *MIMIC-III Waveform Database Matched Subset* [20]. Quest'ultimo contiene 22137 record di segnali fisiologici e 22247 record numerici discreti da 10282 pazienti in terapia intensiva. Esso è definito come "*matched*" perchè associa le forme d'onda del *MIMIC-III Waveform Database* con i dati clinici dell'originale *MIMIC-III Database* [11][14] .
Il dataset è accessibile solo dopo aver seguito un corso sull'uso etico dei dati medici e si è superato l'esame per ottenere il certificato.
Le apparecchiature della terapia intensiva non sono direttamente collegate ai sistemi di database dell'ospedale. L'informazione relativa al numero del paziente e il suo nome devono essere dunque inserite dall'operatore. Dato che questi dati non sono critici per il paziente spesso sono omessi dagli operatori rendendo impossibile il collegamento fra paziente in ICU e lo stesso paziente

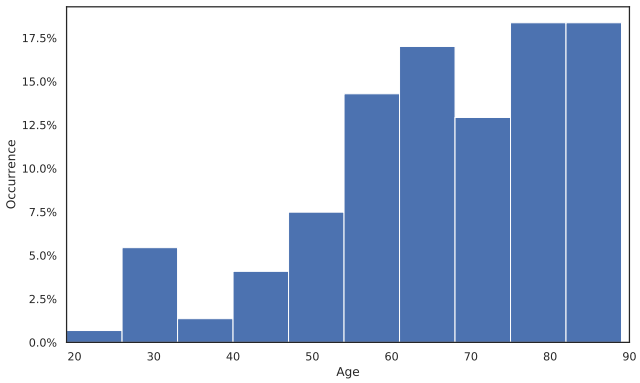
[1] https://physionet.org/content/mimic3wdb/1.0/

in altri reparti dell'ospedale. In questo dataset sono disponibili tutti i pazienti per i quali è stato possibile collegare l'identità ai dati letti dal sistema di monitoraggio della terapia intensiva. I dati sono organizzati in cartelle nella forma `pXXNNNN/pXXNNNN-YYYY-MM-DD-hh-mm` dove *pXXNNNN* è il *subject_id* del paziente, *YYYY-MM-DD* è la data surrogata, ovvero generata tramite una traslazione casuale rispetto alla data reale e *hh-mm* è l'orario reale dell'inzio del monitoraggio. Per lo stesso paziente lo *shift* casuale è uguale per ogni registrazione in modo che rimangano cronologicamente ordinate.
Ad ogni paziente corrispondono fino ad 8 segnali fisiologici; fra questi si trovano spesso ECG, ABP (*Arterial Blood Pressure*) e PPG (*Fingertip Photoplethysmogram*). I dati numerici invece includono tipicamente la pressione sistolica e diastolica, il ritmo del battito cardiaco e della respirazione, l'ossigenazione ed altri, se disponibili. In questo progetto non sono stati utilizzati i dati numerici in quanto si voleva testare un modello facente uso della sola componente proveniente dalla forma d'onda del segnale PPG.
I pazienti compresi nel *dataset* sono stati divisi in gruppo di Controllo e Sepsi secondo i seguenti criteri:

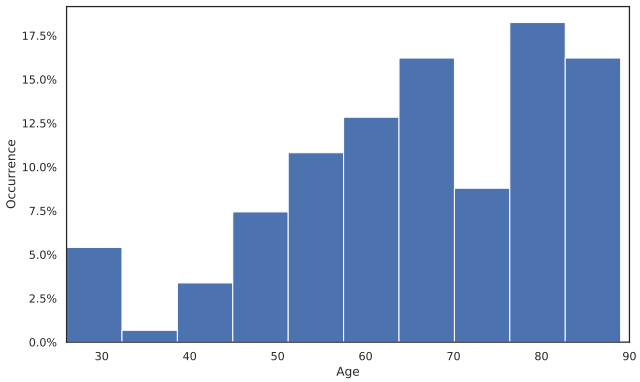
1. Gruppo settici:
    - Singolo ricovero ospedaliero
    - Singolo ricovero in ICU
    - Soggetto deceduto in ospedale
    - Codici di diagnosi (ICD-9): 99591 (Sepsis), 99592 (Severe sepsis), 78552 (Septic shock)
    - Soggetto presente nel MIMIC-III Waveform Database Matched Subset.
2. Gruppo di controllo:
    - Singolo ricovero ospedaliero

- Singolo ricovero in ICU
- Soggetto non deceduto in ospedale
- Codici di diagnosi (ICD-9): 311 (Depressive disorder NEC), 3051 (Tobacco use disorder), 30000 (Anxiety state NOS), 2948 (Other persistent mental disorders due to conditions classified elsewhere), 3004 (Dysthymic disorder)
- Nessuna diagnosi di sepsi
- Soggetto presente nel MIMIC-III Waveform Database Matched Subset.

Come si osserva dalle caratteristiche del gruppo di controllo, i codici di diagnosi scelti sono tutti riconducili a disturbi mentali. Si è dunque ipotizzato che tali pazienti non avessero disturbi riconducibili alla sepsi nel momento di ammissione alla terapia intensiva.
Dal momento che il numero di soggetti di controllo che corrispondevano ai criteri era molto più alto di quello dei pazienti settici, i soggetti di controllo sono stati limitati a 40 soggetti per ciascun codice ICD-9, in modo da bilanciare il dataset.
Il *subset* così selezionato consta di 200 pazienti per il gruppo di controllo e 178 per il gruppo dei settici. Una volta eliminati i soggetti per i quali il segnale PPG non è disponibile, si riducono rispettivamente a 154 e 164 .
I singoli segnali sono stati campionati ad una frequenza di 125 Hz.
In Figura 4.2 è riportato il sesso dei pazienti di controllo e di quelli settici; si osserva che per i primi vi è una prevalenza di donne mentre per i secondi di uomini. Seppur con un campione statisticamente poco rilevante questo conferma quanto ottenuto da altri studi che segnalano una maggior mortalità della sepsi negli uomini rispetto alle donne[18][22][27].
In Figura 4.1 è riportata la distribuzione per età dei pazienti di controllo e di quelli settici: le differenze fra i due gruppi non sono significative, evidenziando solo una maggior presenza di over 70 nei settici.

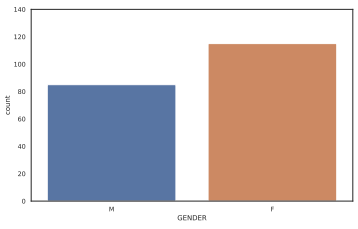


(a) Età pazienti di controllo

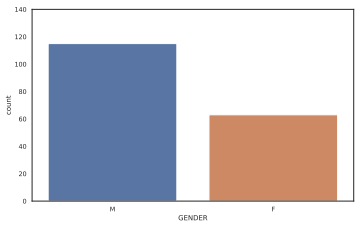


(b) Età pazienti settici

Figura 4.1: Distribuzione di età per i pazienti di controllo (a) e per i pazienti settici (b)

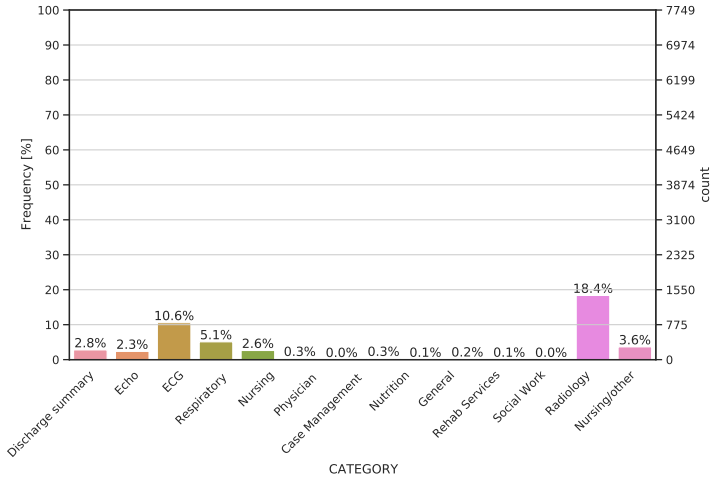


(a) Sesso dei pazienti di controllo

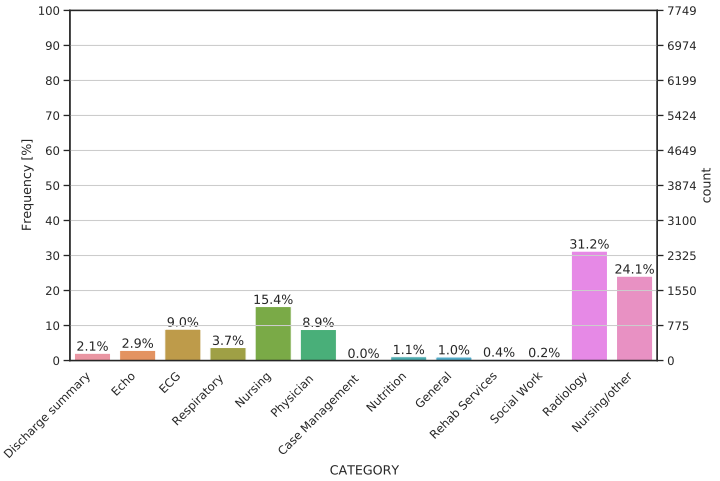


(b) Sesso dei pazienti settici

Figura 4.2: Sesso dei pazienti di controllo (a) e dei pazienti settici (b)



(a) Referti dei pazienti di controllo



(b) Referti dei pazienti settici

Figura 4.3: Categorie dei referti relativi ai pazienti di controllo (a) e settici (b)

Capitolo 5

Pulizia del dataset

5.1 Descrizione del segnale PPG

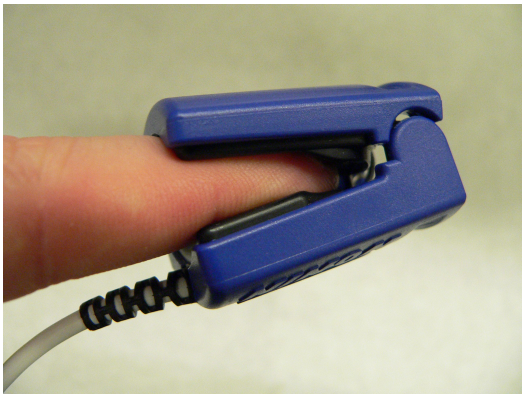


Figura 5.1: Esempio di pulsiossimetro

I segnali PPG seguono generalmente l'andamento mostrato in Figura 5.3. La lettura dei questi segnali viene eseguita in modo non invasivo tramite un pulsiossimetro (visibile in Figura 5.1). Quest'ultimo viene stretto sul polpastrello e invia un impulso luminoso verso la pelle e misura la quantità di luce assorbita. In questo modo consente di misurare il volume del sangue nel letto micro-vascolare dei tessuti. [1]. Quando il cuore pompa il sangue nel resto del corpo si verifica un conseguente aumento di circolo sanguigno anche nei capillari più superficiali della pelle; questo incremento porta ad un maggiore

assorbimento della luce emessa dal pulsiossimetro. Similmente, quando il sangue ritorna al cuore tramite il reticolo venoso, il volume del sangue diminuisce così come, proporzionalmente, la quantità di luce assorbita. Come si osserva da Figura 5.2 la forma d'onda PPG misurata è composta da una componente pulsante (AC) che riflette i cambiamenti sincroni al ciclo cardiaco nel volume di sangue ad ogni battito ed una componente quasi-statica (DC) che contiene informazioni relative alla respirazione, la termoregolazione ed il sistema simpatico [1].

La componente AC del segnale PPG consta di due fasi principali a seconda

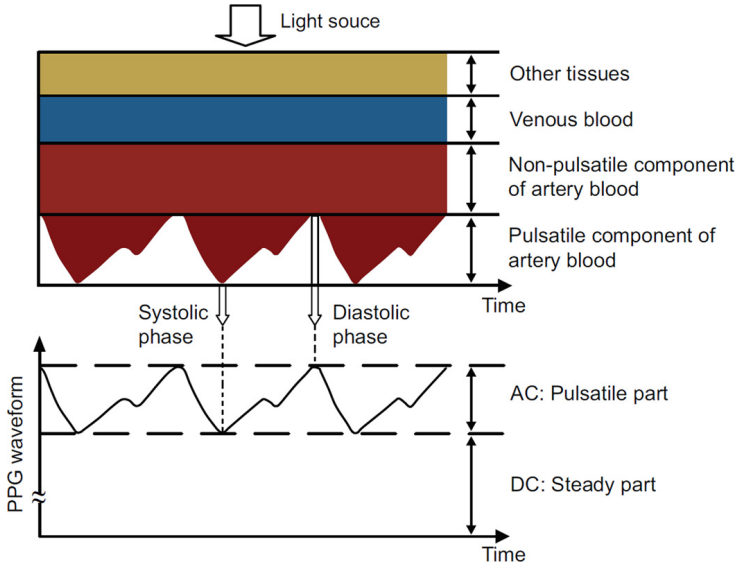


Figura 5.2: Variazione nella luce assorbita nei tessuti

della variazione di volume sanguigno individuato: la fase sistolica e quella diastolica, visibili in Figura 5.3. La prima fase, o fronte di salita, inizia da un avvallamento e termina con un picco detto sistolico. La fase diastolica contiene anch'essa un picco, definito, appunto, diastolico e termina con un ulteriore avvallamento concludendo così il ciclo cardiaco.

La componente AC è influenzata da molteplici sistemi del corpo umano, fra i quali quello arterioso, venoso, autonomo e respiratorio. Per questo motivo la sua forma e alterazione fornisce notevoli informazioni sullo stato complessivo di un soggetto e può essere indicativa di alcune patologie.

L'individuazione dei picchi e delle valli permette di dividere il segnale in blocchi contenenti, ognuno, un ciclo cardiaco. Questa divisione è importante in quanto le variazioni nel segnale PPG come tempo di salita, l'ampiezza dei picchi e la forma possono fornire informazioni sul circolo sanguigno e, di conseguenza sullo stato di salute del paziente. Per ottenere questa segmentazione in modo automatico è quindi necessario definire algoritmi,detti di *peak detection*, in grado di identificare i picchi rilevanti del segnale.

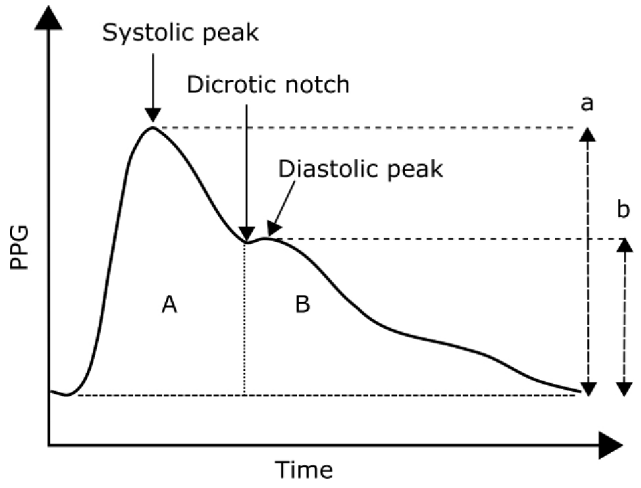


Figura 5.3: Esempio di segnale PPG [23]

5.2 Metodi di *peak detection*

Allo scopo di segmentare il segnale nei singoli cicli cardiaci è necessario individuare i picchi sistolici ed i punti di passaggio fra i due segmenti, detti *trough*. Per fare ciò sono stati testati diversi algoritmi, che ora descriveremo, per poi arrivare all'utilizzo di un approccio misto, sviluppato su misura considerando il dataset sul quale si è lavorato.

5.2.0.1 Ricerca dei minimi e massimi locali con soglia fissa [7]

Questo metodo si basa sull'individuazione dei massimi e minimi locali. Dapprima il segnale viene filtrato con un filtro passa-banda (0.5-8 Hz) per migliorare il segnale durante l'individuazione dei picchi; successivamente si individuano i massimi locali maggiori di δ, soglia per la quale un picco può essere considerato accettabile in termini fisiologici. In formule, $v[n]$ è un picco corrispondente ad un massimo locale e maggiore di δ:

$$v_{th} = v[n] > \delta \tag{5.1}$$

Per individuare i *trough* mediante questo approccio si è invertito il segnale . Tuttavia, se il segnale è rumoroso, in ogni ciclo si individueranno più massimi locali oltre a quello sistolico e diastolico rendendo la segmentazione scorretta e fuorviante per i modelli informatici di classificazione che ne fanno uso.

5.2.0.2 Individuazione dei picchi tramite clustering

Un altro approccio adottato quindi è stato l'individuazione di tutti massimi locali del segnale, sulla cui base si sono estratte alcune *feature* relative all'ampiezza del picco e alla media della differenza di ampiezza fra picchi successivi. In base a tali informazioni è stato allenato un classificatore *KMeans* per l'individuazione di due cluster, uno contenente i picchi sistolici e l'altro quelli diastolici. La stessa procedura è stata ripetuta sui minimi locali per individuare i *troughs*.
In Figura 5.4 è riportato un esempio di divisione in cluster su segnali già puliti. Si osserva, guardando i due blocchi divisi secondo una linea verticale, che vi è netta distinzione e l'algoritmo produce ottimi risultati. Essi sono visibili in Figura 5.5 dove in blu sono riportati i punti di picchi sistolici ed in arancione i picchi diastolici. Un difetto di questo metodo è il fatto che su segnali molto lunghi, come quelli presenti nel dataset, richiede tempi elevati e occupa molta memoria rendendo pressoché impossibili le successive elaborazioni.

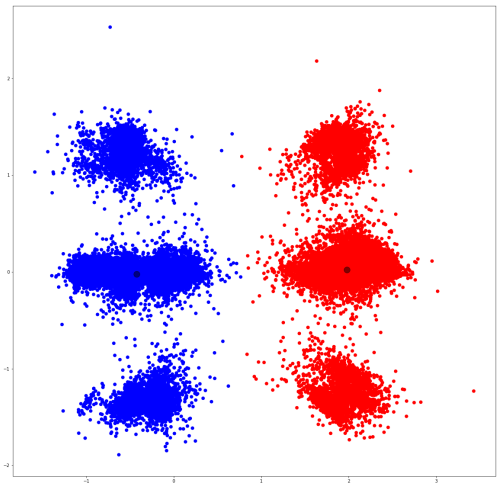


Figura 5.4: Esempio di applicazione del clustering per l'individuazione dei picchi

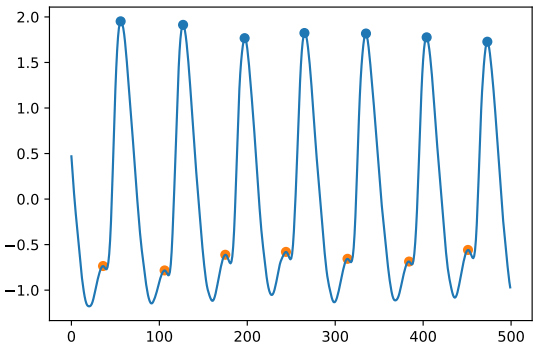


Figura 5.5: Applicazione dell'algoritmo di peak detection basato sul clustering ad un segmento d'esempio

5.3 *Signal Quality Indexes (SQIs)*

I segnali PPG sono spesso soggetti a rumore in quanto la loro lettura viene effettuata nelle condizioni più disparate e tramite strumenti non invasivi. In questo contesto sono state testate diverse metriche per comprendere se esse potessero costituire degli indici per valutare la qualità del segnale PPG. Tali *SQIs* sono descritti in Mohamed E. [6] e riportati qui. Durante la trattazione, per "segnale filtrato" si intenderà l'utilizzo di un filtro passa banda di tipo *Butterworth*, realizzato come composizione di un filtro passa alto con frequenza di taglio ad 1 Hz e ordine 1 e un filtro passa basso con frequenza di taglio di 20 Hz e ordine 4.

5.3.0.1 *Perfusion*

Questo indice è quello più frequentemente utilizzato fino ad ora per la valutazione dei segnali PPG e rappresenta la variazione di luce pulsante assorbita quando il dispositivo è posizionato sul polpastrello. In altre parole è il rapporto fra la componente pulsante e quella semi-stazionaria precedentemente descritta nella Sezione 5.1. Matematicamente è definito come:

$$P_{SQI} = [(y_{max} - y_{min})/|\bar{x}|] \times 100 \tag{5.2}$$

Dove $\bar{x}$ è la media del segnale mentre y è il segnale filtrato.

5.3.0.2 *Skewness*

La *skewness* o *momento statistico del 3° ordine* è una misura della simmetria della distribuzione di probabilità, in formule:

$$S_{SQI} = 1/N \sum_{i=1}^{N} [x_i - \hat{\mu}_x/\sigma]^3 \tag{5.3}$$

dove $\hat{\mu}_x$ e σ sono rispettivamente il valore medio atteso e la deviazione standard attesa del segnale mentre N è la lunghezza del segnale

5.3.0.3 Curtosi

La curtosi o *momento statistico del 4° ordine* è una misura statistica della distribuzione dei dati attorno alla media, matematicamente è definibile

$$K_{SQI} = 1/N \sum_{i=1}^{N} [x_i - \hat{\mu}_x/\sigma]^4 \tag{5.4}$$

dove $\hat{\mu}_x$ e σ sono rispettivamente il valore medio atteso e la deviazione standard attesa del segnale mentre N è la lunghezza del segnale.

5.3.0.4 Entropia

Questa misura quantifica la distanza fra la funzione di densità di probabilità (PDF) del segnale ed una distribuzione uniforme. In altre parole misura l'incertezza del segnale. In formule:

$$E_{SQI} = -\sum_{n=1}^{N} [x[n]^2 \log_e(x[n]^2)] \tag{5.5}$$

Dove x è il segnale ed N la lunghezza del segnale

5.3.0.5 Zero crossing rate

Quantifica la tendenza del segnale a cambiare segno, passando da positivo a negativo; è definito come:

$$Z_{SQI} = 1/N \sum_{n=1}^{N} [\mathbb{I}[s_t s_{t-1} < 0]] \tag{5.6}$$

Dove s_t è il campione del segnale filtrato al tempo t ed N è la lunghezza del segnale. $\mathbb{I}$ è una funzione che assume il valore 1 se la condizione fra parentesi quadre è verificata, 0 altrimenti.

5.3.0.6 Rapporto segnale-rumore

Misura la quantità di rumore rispetto al segnale. In questo contesto è rappresentata dal rapporto fra la deviazione standard del segnale filtrato e la

deviazione standard del segnale originale:

$$N_{SQI} = \sigma^2_{signal}/\sigma^2_{noise} \tag{5.7}$$

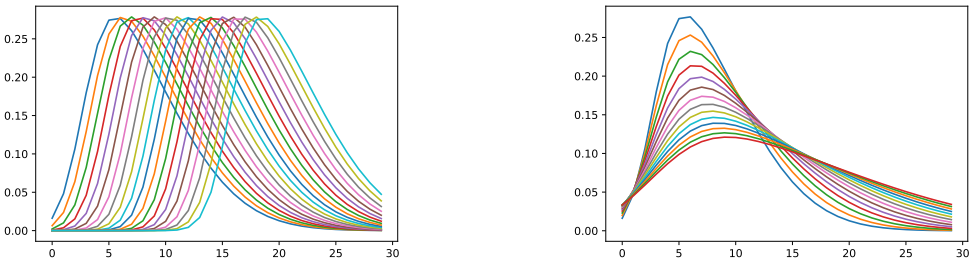
5.3.0.7 Potenza relativa

A differenza delle metriche viste fino ad ora questa analizza il segnale nel dominio della frequenza. Siccome l'energia dei picchi sistolici e diastolici è concentrata maggiormente nella banda 1-2.25 Hz [7]; è stata rapportata la densità di energia spettrale in questa fascia di frequenze rispetto a quella di tutto il segnale (0-8 Hz):

$$N_{SQI} = \sum_{f=1}^{2.25Hz} PSD / \sum_{f=0}^{8Hz} PSD \tag{5.8}$$

5.3.0.8 DTW

Una valutazione differente della qualità del segnale PPG è stata fatta mediante l'utilizzo del *Dynamic Time Warping (DTW)* per confrontare il singolo battito del segnale PPG con un template generato a priori; questo metodo è ispirato da Q Li et al. [16].
DTW è stato utilizzato per ottenere una misura della distanza fra i punti corrispondenti del singolo ciclo del segnale PPG e il template.
Per ogni ciclo è stato generato un template della lunghezza corrispondente al segmento in analisi e valori di ampiezza compresi fra 0 e 1; usando poi la funzione `MinMaxScaler()` della libreria *Python Scikit-learn* è stato scalato il segmento PPG nello stesso range di ampiezze. Si sono testati due *template* diversi: uno ottenuto come unione di distribuzioni *skew* e l'altro come unione di due segnali sinusoidali.
La distribuzione *skew*, su cui si basa il primo *template*, è calcolata come mostrato nel codice *Python* in Figura 5.7 dove e rappresenta la posizione del picco della funzione e w la scala della distribuzione sull'asse delle ascisse. In Figura 5.6 è riportato il grafico della funzione di *skew* variando il parametro e a sinistra e il parametro w a destra.



(a) Variazione parametro e (b) Variazione parametro w

Figura 5.6: Variazione dei parametri e e w nella distribuzione *skew*

Il primo template è stato poi ottenuto come somma di due distribuzioni *skew*, una con parametro e posto ad 1 e scala a 2.5 ed una con posto e a 3 e scala di 3. Successivamente sono state unite considerando il massimo in ogni posizione in modo da avere il picco sistolico in corrispondenza del punto di massimo. Il *template* così ottenuto è visibile in Figura 5.8.
Il secondo template, mostrato in Figura 5.9, è ottenuto come somma di sinusoidi nel modo seguente:

$$template = \sin(2\pi 2t - \pi/2) + \sin(2\pi t - \pi/6) \tag{5.9}$$

In particolare, dopo diversi test, si è osservato che l'approccio che produce risultati più corretti e verosimili è quello basato su due funzioni di skew, perciò per la valutazione della qualità del segnale si è utilizzata quest'ultima.

5.3.1 Risultati

Tutti gli SQI sono stati calcolati sia sul singolo battito che come media dei cicli sul segmento di 5000 campioni. I risultati non sono qui riportati a causa del loro numero molto elevato. Le metriche di skewness e curtosi si sono rivelate ottime nel distinguere segmenti di bassa ed alta qualità e anche il confronto con il template ideale mediante DTW ha prodotto risultati di pari qualità. In Figura 5.10 sono visibili le tre metriche appena citate calcolate come media, su un segmento di 5000 sample, dei valori per ogni battito. Si

```
def skew_func(x, e, w, a):
t = (x - e) / w
omega = (1 + erf((a * t) / np.sqrt(2))) / 2
gaussian_dist = 1/(np.sqrt(2 * np.pi)) * np.exp(-(t ** 2) /
                                    2)
return 2 / w * gaussian_dist * omega
def ppg_absolute_dual_skewness_template(width,e_1=1,w_1=2.5,
                                    e_2=3,w_2=3, a=4):

x = np.linspace(0, 11, width, False)
p_1 = skew_func(x, e_1, w_1, a)
p_2 = skew_func(x, e_2, w_2, a)
p_ = np.max([p_1, p_2], axis=0)
```

Figura 5.7: Codice per la generazione del template formato da due funzioni di skew

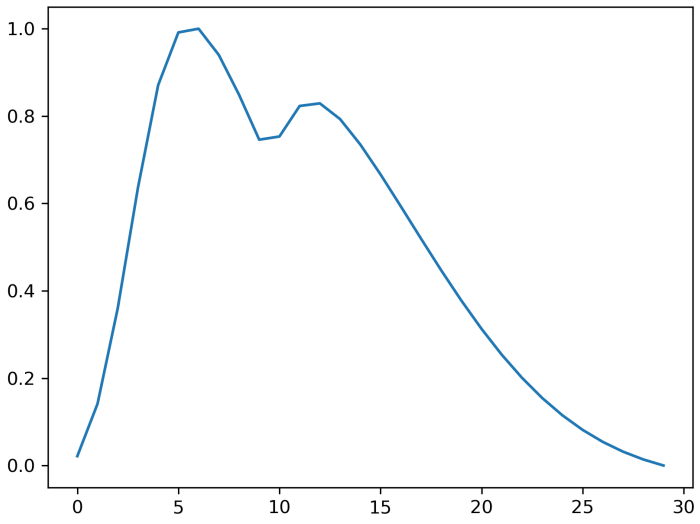


Figura 5.8: Template generato come unione di due funzioni di skew

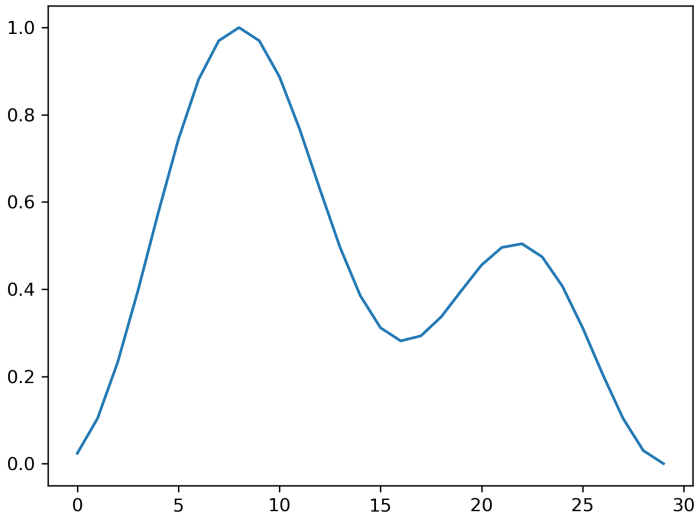


Figura 5.9: Template generato come unione di due sinusoidi

osserva che l'andamento delle tre è pressoché identico a conferma della loro intercambiabilità. Ciò che si può inoltre osservare è un effetto di slivellamento che porta le metriche ad assumere valori anche lontani fra di loro. Questo fa si che in un intorno la metrica sia un buon indicatore della qualità del segnale ma globalmente non possa essere significativa ed utilizzabile. L'effetto di trascinamento è dovuto alla variabilità di frequenza dei battiti, per ovviare a questo problema sarebbe necessario segmentare preventivamente il segnale nei singoli cicli e poi ottenere gli SQIs su questi ultimi. Ciò non è possibile in quanto lo scopo di valutazione della qualità della registrazione è proprio finalizzato a rendere il segnale più pulito per essere fornito come input ad uno degli algoritmi di *peak detection*.

Questa circolarità dunque richiede un sistema differente di valutazione della qualità del segnale. L'approccio che verrà proposto nel prossimo paragrafo si pone l'obiettivo di ovviare ai problemi appena evidenziati rendendo anche la pulizia sufficientemente veloce per essere eseguita sulla grande mole di dati qui in analisi.

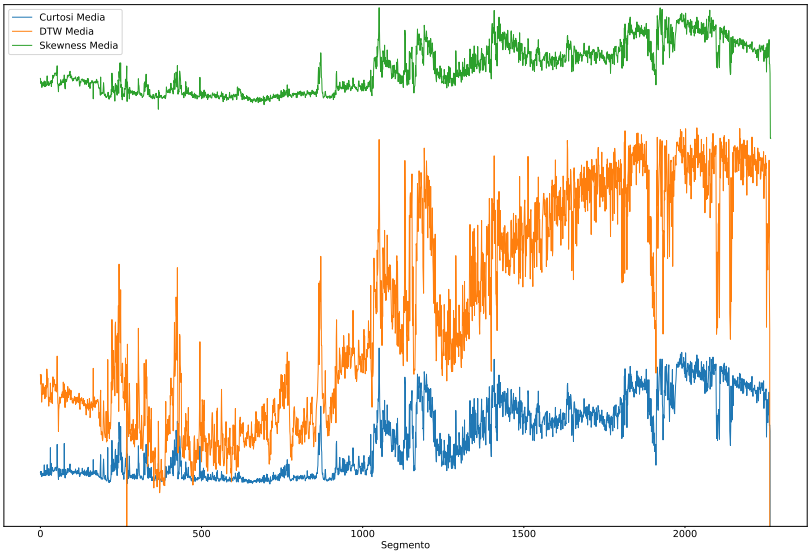


Figura 5.10: Esempio di calcolo degli SQI ed effetto di deriva dell'errore

5.4 Pulizia e individuazione dei picchi tramite soglia a deviazione standard

Siccome i metodi di *peak detection* e SQIs descritti e testati non hanno ottenuto risultati soddisfacenti si è scelto di sviluppare un algoritmo di peak detection dedotto a partire dall'osservazione dei segnali presenti nel dataset.

Tale metodo consta delle seguenti fasi:

1. Eliminazione dei periodi in cui la lettura non è presente
2. Filtraggio del segnale
3. Individuazione dei massimi locali
4. Divisione del segnale in segmenti
5. Eliminazione dei segmenti senza picchi

6. Calcolo dei valori di deviazione standard

7. Eliminazione degli *outlier*

In prima battuta si sono eliminati dal segnale tutti i tratti per i quali la registrazione non è presente. Questi derivano generalmente da una disconnessione dello strumento e non sono utili alla classificazione. Successivamente si è filtrato il segnale con un filtro passa banda di tipo *Butterworth*; tale filtro è stato realizzato come combinazione di un filtro passa alto con frequenza di taglio ad 1 Hz e ordine 1 di un filtro passa basso con frequenza di taglio di 20 Hz e ordine 4.
In Figura 5.11 è riportato un esempio di filtraggio secondo il modello appena descritto. Il segnale filtrato è a media nulla, inoltre, i picchi sistolici e diastolici vengono evidenziati per favorirne l'individuazione. Il filtraggio inoltre riduce il rumore prodotto dall'interpolazione facendo quindi anche da *smoothing*. Questa operazione preliminare permette anche di ridurre il rumore eliminandolo al di fuori della banda di interesse per il segnale. L'individua-

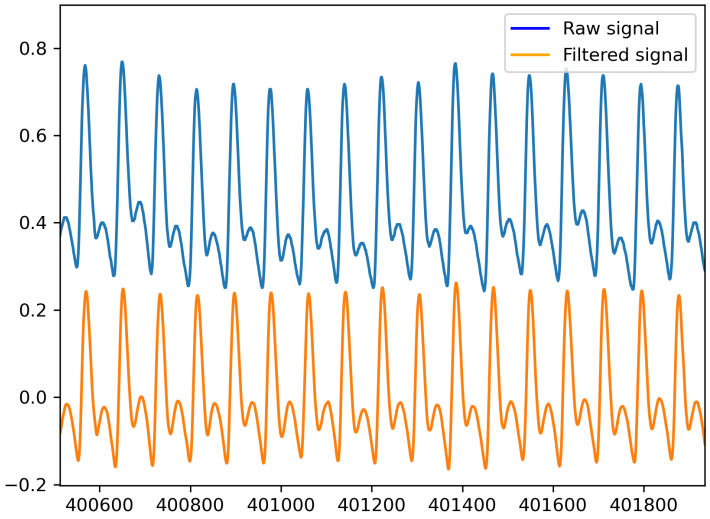


Figura 5.11: Esempio di filtraggio di un segnale PPG

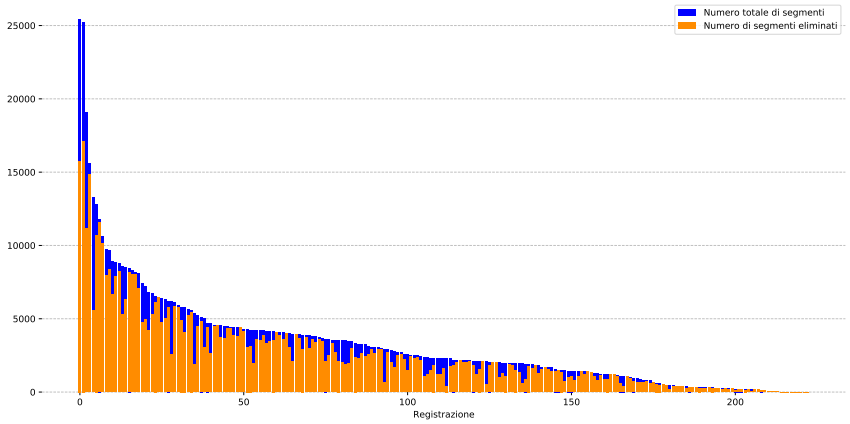
zione dei picchi è stata eseguita sfruttando la libreria *SciPy*[1], in particolare la funzione `find_peaks()`. Quest'ultima prende in input un array 1-D e trova tutti i massimi locali confrontando i campioni vicini. Per adattare la funzione al problema in questione la distanza minima fra due picchi, affinché un picco sia considerato valido, cioè non introdotto dal rumore, è stata fissata a 30 *sample* (0.24 secondi o circa 240 battiti al secondo) in modo da escludere picchi troppo ravvicinati che non sono realistici considerando il fenomeno fisico in analisi. Inoltre, dopo alcuni tentativi, si è regolata la *prominence* a 0.14. Essa definisce quanto un picco "risalta" rispetto ai punti di un suo intorno ed è misurata come la distanza, in verticale, fra un picco e la sua linea di contorno minima, ovvero quella che non contiene picchi con valore maggiore rispetto a quello in analisi.
Come passaggio successivo il segnale è stato diviso in finestre di dimensione fissa di 5000 segmenti, equivalenti a 40 secondi con una frequenza di campionamento di 125 Hz. Questa precisa suddivisione si è scelta come compromesso in modo da non rendere troppo grande la dimensione dell'input del modello di AI che verrà descritto nel Capitolo 6 e mediante il quale si è eseguita la classificazione, allo stesso tempo ipotizzando tuttavia che la lunghezza della finestra sia sufficiente per riconoscere all'interno del segnale un'indicazione di sepsi.
Considerando finestre di 40 secondi, se per ogni sotto-segmento di 3 secondi consecutivi non sono stati trovati picchi, si scarta l'intera finestra. Questo è stato fatto considerando che, generalmente, finestre che non presentano picchi validi per diversi istanti sono costituite da rumore e sono dunque da scartare. Per trovare i punti di *trough* si è semplicemente invertito il segnale e si è ripetuta la procedura allo stesso modo.
Una volta individuati tutti i punti di interesse si sono calcolate due metriche:

1. Lo scarto medio dei picchi rispetto alla media.

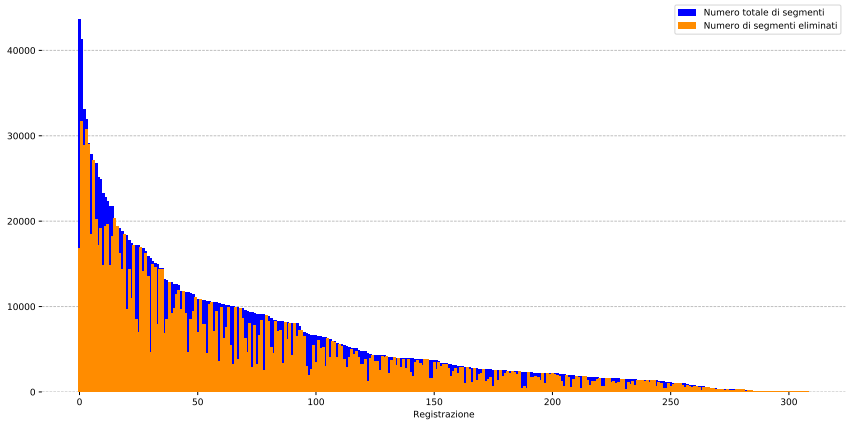
2. Lo scarto medio della distanza fra picchi consecutivi rispetto alla media.

[1]https://docs.scipy.org/doc/scipy/index.html

Questi valori consentono di definire quando un segnale ha un andamento poco uniforme che denota, nella maggior parte dei casi, rumore. Per scegliere quali segmenti scartare si sono fatti vari tentativi analizzando diversi valori di soglia. La soglia trovata per la prima metrica è 0.15 mentre per l'altra è 10. Una volta trovati questi valori si è eseguito l'algoritmo sui pazienti sani e quelli settici. In Figura 5.12a è riportato l'istogramma dei segmenti eliminati rispetto a quelli totali nei pazienti di controllo. In media sono stati eliminati l'81.5% dei segnali per singola registrazione. Una scrematura così forte ha permesso di mantenere solo i segnali di maggior qualità escludendo quelli che contenevano rumore. Al termine della pulizia sono rimasti comunque un totale di 148345 segmenti, corrispondenti a circa 1648 ore di registrazione.
In Figura 5.12b è riportato l'istogramma dei segmenti eliminati rispetto a quelli totali nei pazienti settici. In media sono stati eliminati il 79.2% dei segnali. Al termine della pulizia sono rimasti un totale di 424196 segmenti, corrispondenti a circa 4713 ore di registrazione.
Si osserva che la percentuale di segmenti eliminati è pressoché identica fra pazienti sani e settici; questo dato conferma l'ipotesi che il rumore sia equamente distribuito nei due gruppi e non sia dunque un fattore di distinzione.



(a) Gruppo di controllo



(b) Gruppo sepsi

Figura 5.12: Grafico del numero di segmenti mantenuti rispetto a quelli eliminati, per ogni registrazione, nel gruppo di controllo (a) e dei settici (b). Il grafico è ordinato per numero totale di segmenti, rappresentato in blu. Il numero di segmenti eliminati è mostrato in arancione.

Capitolo 6

Sviluppo del modello

In questo capitolo verranno discussi i modelli e le metodologie applicate per lo sviluppo del classificatore, in particolare il *training* dell'autoencoder, le modalità di estrazione delle *features*, le tecniche di riduzione della dimensionalità dei dati e il classificatore finale che ha prodotto i migliori risultati nella classificazione. In particolare è stato prima sviluppato l'autoencoder allenato sui pazienti di controllo, quest'ultimo è stato utilizzato per classificare i dati grezzi. I migliori segmenti, classificati dall'autoencoder come puliti, sono poi forniti in input ad una rete LSTM che li classifica in base al gruppo di appartenenza.

6.1 Sviluppo dell'autoencoder

Nella prima fase si è allenato un *autoencoder* come descritto nella Sezione 3.5.1. Esso è composto da due layer LSTM bidirezionali (*BiLSTM*) per l'encoder ed altri due, sempre bidirezionali, per il decoder. La scelta di guardare al passato e al presente degli input con delle reti BiLSTM consente di osservare il segnale in input sia nell'ordine originale che in quello inverso permettendogli di apprendere da variabili sia passate che future nella sequenza. L'autoencoder è stato allenato a partire dai pazienti di controllo per creare un modello del soggetto non settico. Si è ipotizzato è che il modello in que-

stione commetta maggiori errori nella ricostruzione dei segnali fisiologici dei pazienti settici rispetto a quelli di controllo cui il quale è stato allenato. I dati dei pazienti di controllo sono stati puliti secondo quando descritto nella Sezione Il dataset così risultate è stato standardizzato in modo da avere una distribuzione Gaussiana con media nulla e deviazione standard pari a 1 applicando uno `StandardScaler` della libreria *sklearn*.
Successivamente è stato eseguito un *resampling*: si è osservato che anche dimezzando il numero di campioni le caratteristiche e la forma d'onda del segnale rimanevano pressoché immutate. Questo ha ridotto il numero di campioni per segmento a 2500.
Adottare questo accorgimento ha permesso di diminuire i tempi di training delle reti *BiLSTM* che hanno il doppio dei parametri rispetto alle normali LSTM, oltre che facilitare la convergenza della rete riducendo la dimensionalità dello spazio delle features.
Sono state testate varie reti variando il numero di parametri e di layer LSTM, in Figura 6.1 è riportato il modello che ha prodotto i migliori risultati. Esso comprende 512 celle BiLSTM per il primo *layer* dell'encoder e 256 per il secondo layer. Il decoder è poi composto da 256 celle nel primo strato e da 512 nel secondo.

Come funzione di loss è stato utilizzato l'errore quadratico medio fra il segnale in ingresso e lo stesso ricostruito dalla rete in output.
In figura 6.2 è riportato il grafico che mostra la *reconstruction loss* della rete per il *training* ed il *validation set*. Il *fit* del modello è stato eseguito con ottimizzatore `Adam`, *learning rate* pari a 0.001 e *clipnorm* pari a 1 per mitigare un fenomeno di *exploding gradient* rilevato durante il training e tipico delle RNN per input di grosse dimensioni come segnalato da Graves et al. e qui riportato:

> *One difficulty when training LSTM with the full gradient is that the derivatives sometimes become excessively large, leading to numerical problems. To prevent this, [we] clipped the derivative of the loss with respect to the network inputs to the LSTM layers (before the sigmoid and tanh*

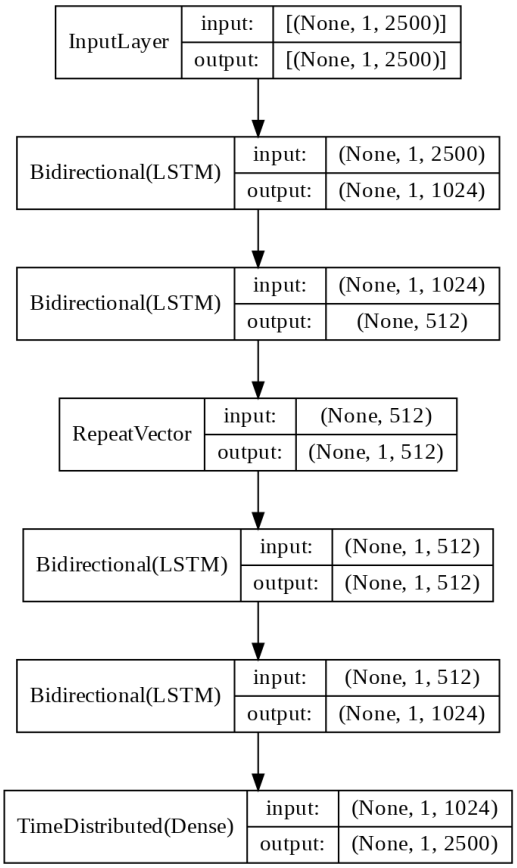


Figura 6.1: Struttura dell'*autoencoder* che ha prodotto i migliori risultati

functions are applied) to lie within a predefined range.[12].

Il training è stato impostato a 200 epoche con `EarlyStopping` pari a 10 sul *validation loss*. Questo ha portato la rete a convergere in 114 epoche con *reconstruction loss* sul *training set* di 0.0663 e 0.0780 sul *validation*. In Figura 6.3 viene mostrato un segmento ricostruito

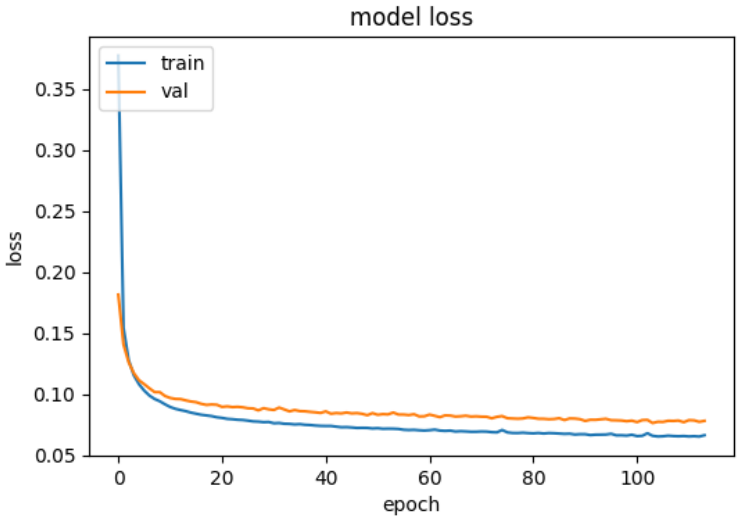


Figura 6.2: Train e validation loss dell'autoencoder

dalla rete, non visto durante il training, in questo caso la loss è pari a 0.056. Si osserva che il segnale è ricostruito in modo sufficientemente preciso, anche se a volte la rete inserisce alcuni picchi diastolici spuri, amplificando quelli che sono i reali picchi del segnale come mostrato in Figura 6.3 nel picco a tempo 400. Interessante osservare come si comporta la rete in presenza di un paziente settico con delle anomalie sui picchi diastolici come in Figura 6.4. La *loss* in questo caso arriva a 0.76 e l'autoencoder non riesce a ricostruire correttamente i picchi diastolici.

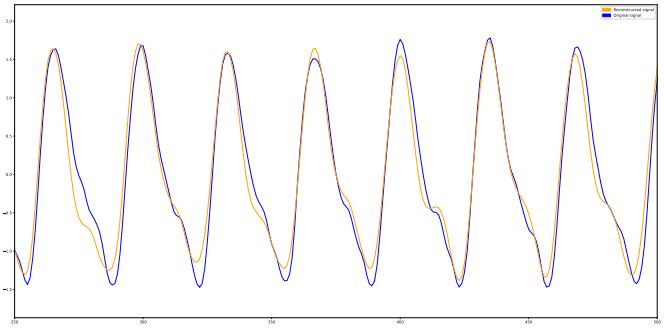


Figura 6.3: Ricostruzione di un segmento non visto in *training*

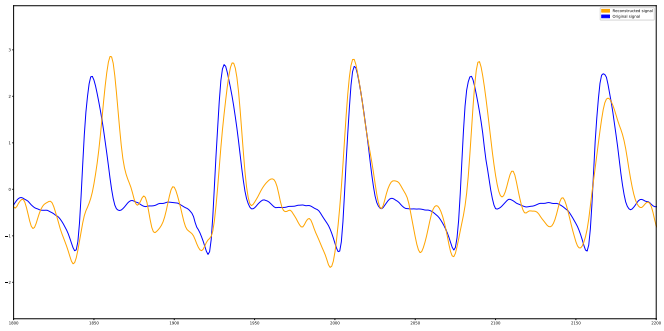


Figura 6.4: Ricostruzione di un segmento di un paziente settico

6.2 Classificazione mediante PDF della *loss* dell'autoencoder

Una volta sviluppato l'autoencoder descritto nella Sezione 6.1 sono stati forniti in input i dati non puliti, ma ugualmente normalizzati, dei pazienti sani e di quelli settici. Le registrazioni di quest'ultimi sono state unite in un unico record in ordine temporale secondo le date fittizie del dataset.
Ogni segmento ha fornito una *reconstruction loss*, i singoli errori sono stati raccolti in una funzione di distribuzione di probabilità (*PDF*) con numero di *bin* pari a 80. Questo valore è stato scelto in modo che non si avessero segnali troppo brevi che potessero presentare un numero di campioni inferiore o troppo vicino al numero dei *bin* portando a generare istogrammi caratterizzati da ampia presenza di valori nulli.
In Figura 6.5 è riportato il grafico degli istogrammi dell'errore per i pazienti di controllo, mentre in Figura 6.6 quelli per i pazienti settici. Ogni paziente è rappresentato da una curva di differente colore. Si osserva che gli andamenti sono simili: la maggior parte degli errori si concentrano nell'intervallo fra 0 e 0.3. Dalla visualizzazione dei segnali si osserva inoltre che valori maggiori di 0.5 rappresentano segnali sporchi (o solo rumore) e valori intermedi segnali con anomalie. Proprio in questi intervalli di segnali anomali ma non particolarmente interessati dal rumore possono essere concentrate le differenze chiave nelle forme d'onda che permettono di distinguere i pazienti di controllo da quelli settici.

6.2.1 Riduzione della dimensionalità

A partire dalle 80 *features* della PDF, utilizzando PCA, è stata diminuita la dimensione degli istogrammi. In Figura 6.7 è riportato il grafico con 2 componenti principali, che mantengono circa il 55% della varianza totale. Si può notare un gruppo isolato in basso a sinistra di pazienti settici. Questo supporta l'ipotesi che vi siano differenze significative in termini di distribuzione dell'errore fra i due gruppi.

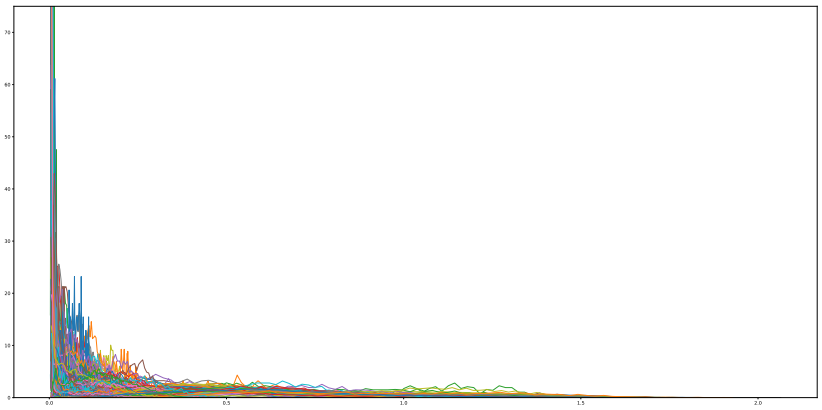


Figura 6.5: PDF della loss dei pazienti di controllo

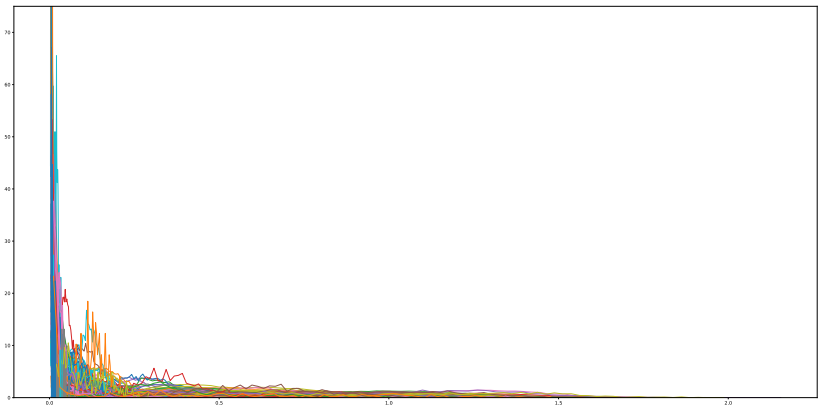


Figura 6.6: PDF della loss dei pazienti settici

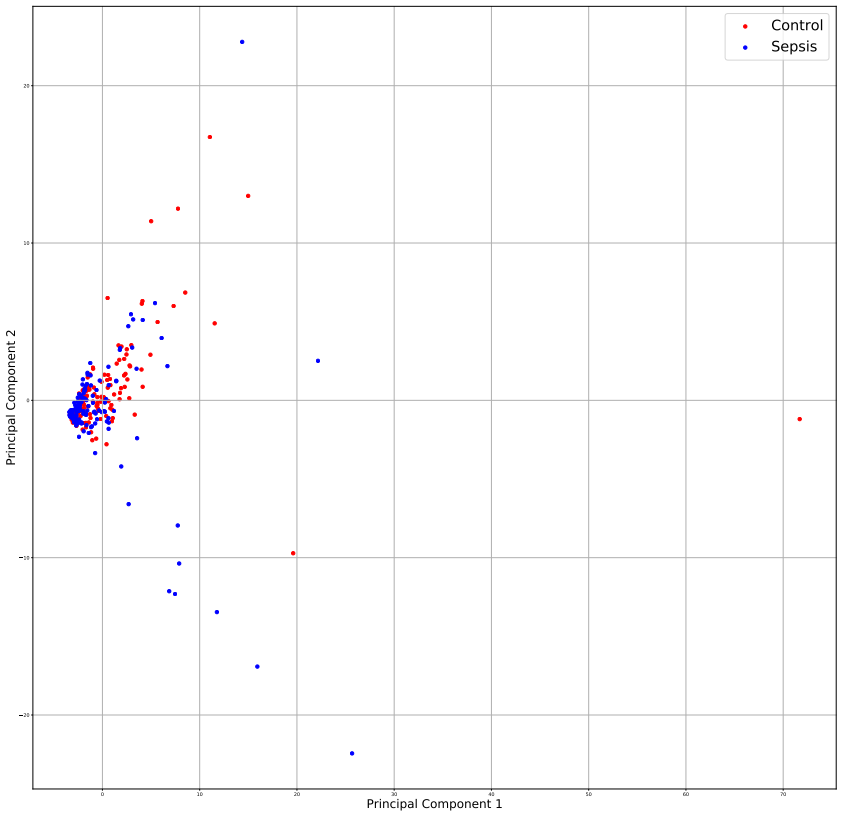


Figura 6.7: PCA a 2 componenti con tutti gli 80 *bin* considerati

6.2.2 Risultati mediante PDF della loss e XGBoost

Una volta allenato l'autoencoder sui pazienti di controllo e calcolata la PDF dell'errore di ricostruzione dell'autoencoder con un istogramma a 80 *bin* si è sviluppato un classificatore basato su ML sulla distribuzione di probabilità della *loss* allo scopo di verificare se vi fossero delle differenze fra i gruppi di pazienti settici e di controllo. Il classificatore utilizzato è *XGBoost*, che costruisce una serie di alberi di decisione e valuta in parallelo le predizioni di ogni singolo classificatore per effettuare una scelta finale. XGBoost ottiene generalmente risultati migliori rispetto ai semplici alberi di decisione ma, per contro, perde l'interpretabilità dei risultati, ovvero la possibilità di ricostruire il percorso logico seguito per raggiungere il risultato finale. Come metrica di valutazione delle performance del classificatore si è scelto l'F1-Score in quanto il dataset è leggermente sbilanciato con 143 pazienti settici e 153 di controllo. L'algoritmo è stato eseguito variando il numero di componenti principali considerate, partendo da 1 ed arrivando fino a conservarle tutte e 80. Ogni test, fissato il numero di componenti principali, è stato eseguito in *cross-validation* per 5 esecuzioni indipendenti variando test e validation set. Il numero di alberi decisionali su cui viene valutato l'input è stato scelto pari a 400 e la massima profondità pari a 150 nodi.

In Figura 6.8 è riportato il grafico dell'F1-Score e la varianza mantenuta al variare del numero di componenti principali conservate. Si osserva che l'accuratezza varia fra il 50% e il 60% ed in particolare raggiunge picchi significativi quando le componenti principali mantenute sono fra le 2 e le 10. Successivamente cala quando si riduce con PCA a più di 40 componenti. Nel grafico è anche riportata la quantità cumulativa di varianza mantenuta da ogni insieme di variabili. I risultati migliori si ottengono conservando fra il 95% e il 98% della varianza. Da quanto si evince da questo esperimento vi sono comunque delle differenze in termini di capacità di ricostruzione dell'autoencoder fra i pazienti di controllo, con il quale è stato allenato, ed i pazienti settici. Nel prosieguo di questo capitolo verranno investigate ulteriormente queste differenze allo scopo di costruire un modello con una maggior capacità

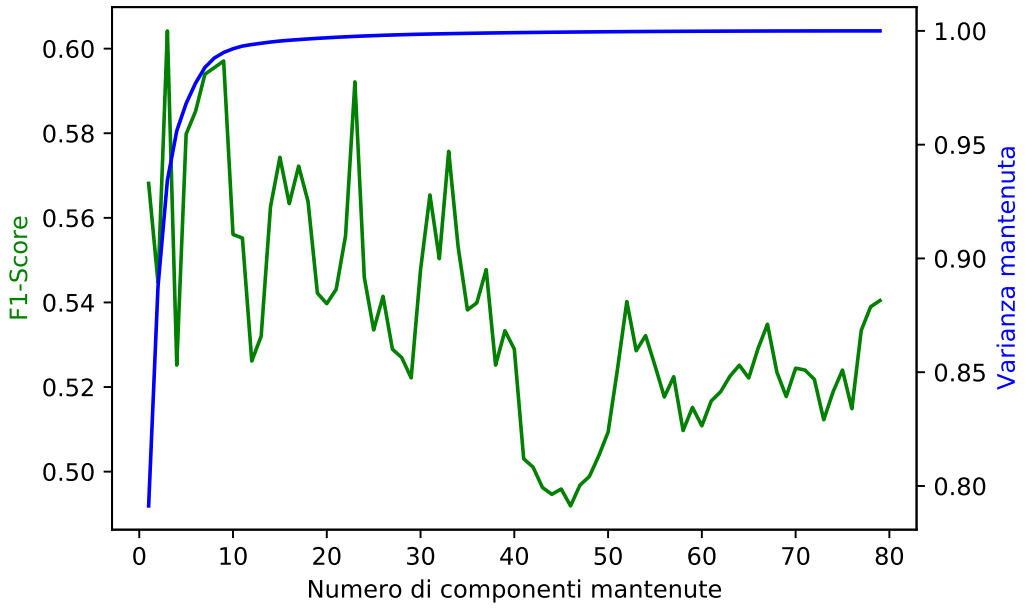


Figura 6.8: F1-Score e varianza mantenuta al variare del numero di componenti principali conservate

predittiva che faccia uso dell'autoencoder come metodo di ricostruzione del segnale.

6.3 Classificazione mediante reti LSTM

Una volta verificato che la ricostruzione dei segnali in esame mediante *autoencoder* allenato con i soli pazienti di controllo, conserva ancora delle differenze fra i due gruppi di interesse si sono approfondite queste diversità alla ricerca di casi patologici riconducibili alla sepsi.
Una volta calcolata la PDF dell'errore di ricostruzione dell'autoencoder come descritto nella Sezione 6.2 si sono eliminati tutti i segmenti per i quali tale errore è maggiore di 0.3, valore per il quale si è osservata una diminuzione della qualità del segnale o, in altri termini, rumore. L'eliminazione di questi segmenti equivale ad eliminare la coda della PDF descritta in precedenza e mostrata in Figura 6.6 e 6.5. I segmenti così selezionati sono stati forniti in input ad un modello come mostrato in figura 6.9. Quest'ultimo è costituito da tre *layer* LSTM, due da 512 unità ed una da 256 ed uno strato con un singolo neurone con funzione di attivazione *sigmoid*. In Figura 6.10 sono riportati i risultati relativi al training del modello.
Sul test-set l'AUROC ha raggiunto il 75%, questo conferma la buona capacità predittiva del modello. L'F1-Score si attesta all'85% e le metriche di *precision* e *recall* sono, corrispondetemene, dell' 83% e dell' 87%. Un discorso differente va fatto per la specificità e la sensibilità. La prima raggiunge il 50% e la seconda l'85%, questo si traduce in una maggiore performance del modello nell'identificare correttamente i pazienti malati con un basso tasso di falsi negativi ed una minor capacità di classificazione dei pazienti sani con un alto tasso di pazienti sani.
Successivamente è stato ripetuto il training sullo stesso modello bilanciando il dataset. I risultati del training sono visibili in Figura 6.11. In questo caso si è ottenuto, sul test set, un F1-Score pari al 76% ed un valore di AUROC dell'83%, più elevato rispetto al caso non bilanciato. Anche in questo caso il

classificatore è stato quindi in grado di garantire una buona distinzione fra i due gruppi.

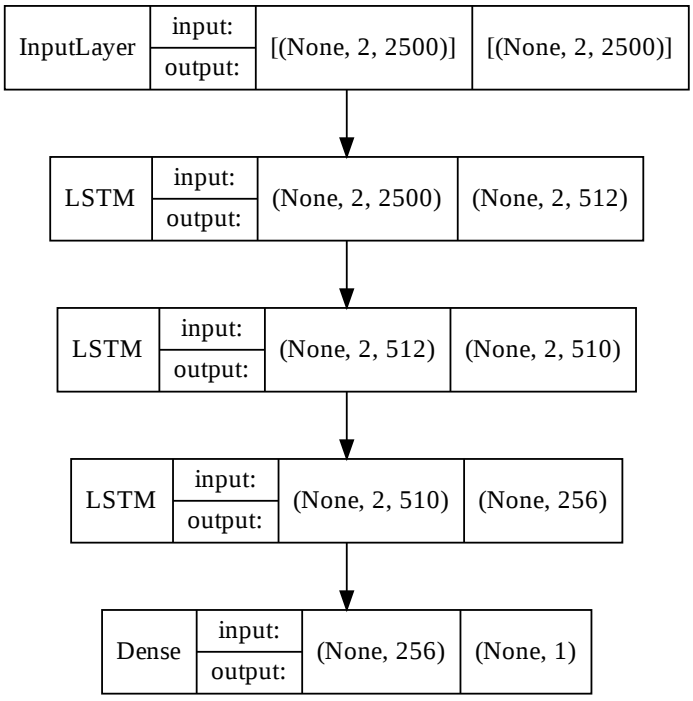


Figura 6.9

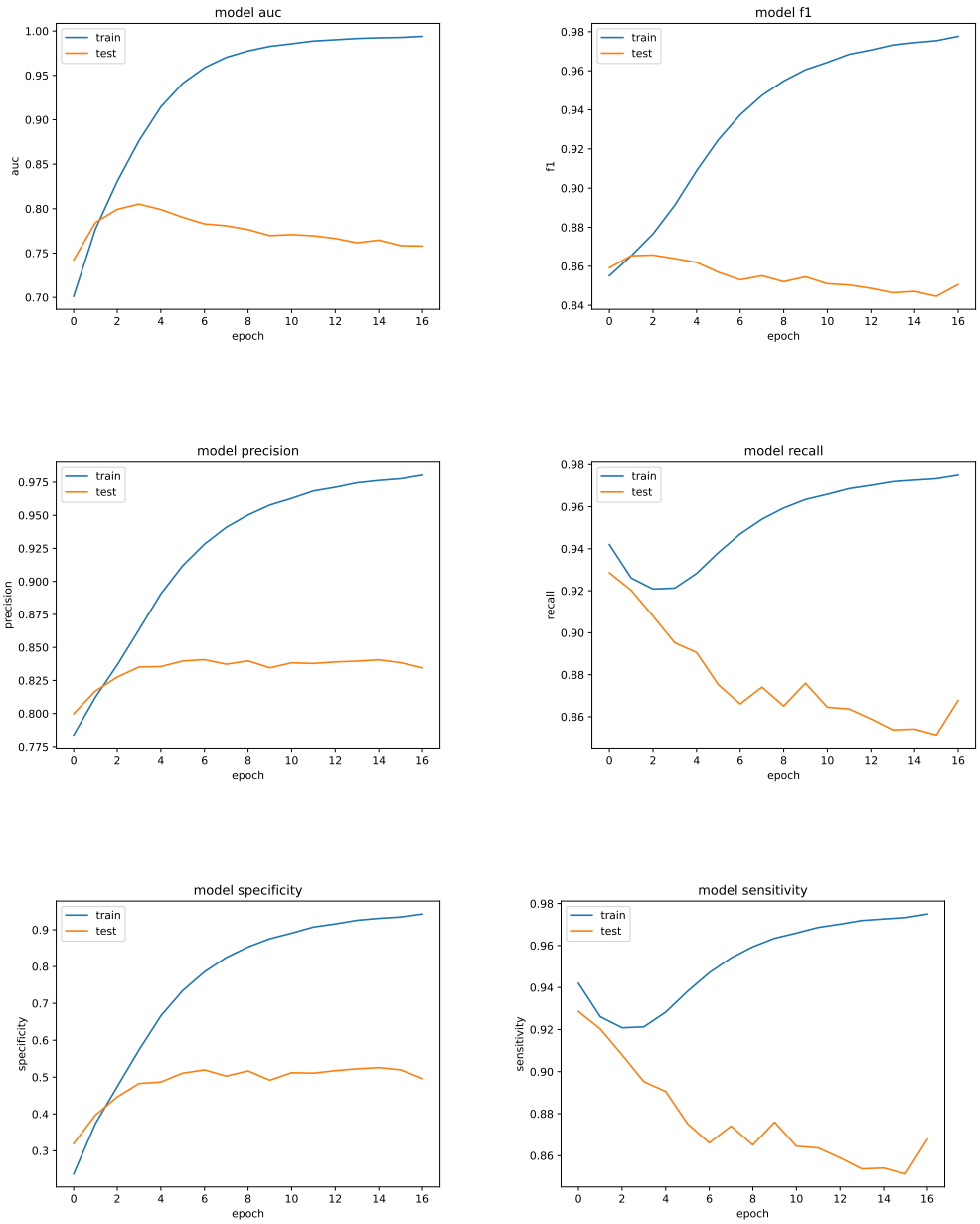


Figura 6.10: Grafici relativi al training del modello con dataset non bilanciato

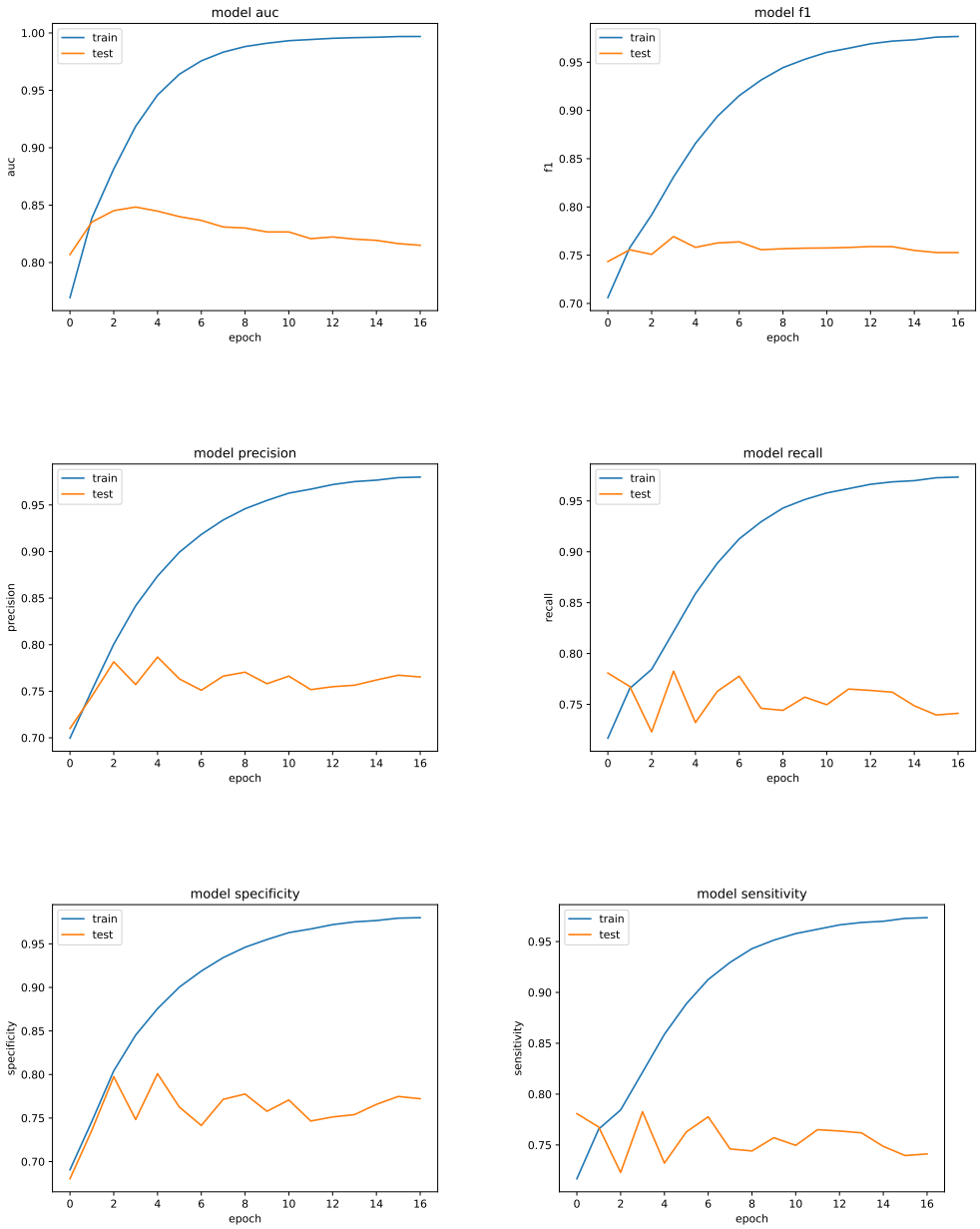


Figura 6.11: Grafici relativi al training del modello con dataset bilanciato

Capitolo 7

Conclusioni

Il lavoro di Tesi ha condotto allo sviluppo di un modello AI che sfrutta principalmente strati di tipo LSTM (Long Short-Term Memory) per la predizione dell'insorgenza di sepsi in pazienti in terapia intensiva, a partire dai segnali pressori letti con tecniche non invasive.
Una parte fondamentale del lavoro è stata quella preliminare di pulizia dei dati. Questa ha condotto allo sviluppo di un sistema intermedio, sempre basato su reti neurali, in grado di quantificare il rumore presente nei segnali pressori. Tale sistema, basato su *autoencoder*, ha permesso di eliminare parti del segnale non riconducibili a fenomeni fisiologici o, in altre parole, i segmenti rumorosi delle registrazioni dei pazienti. Inoltre l'autoencoder ha mantenuto delle differenze fra i segnali dei pazienti settici e quelli di controllo. Tali differenze sono state esplorate riducendo, lungo le sue componenti principali, l'errore di ricostruzione dei segnali commesso dall'autoencoder. Si è dimostrato, mediante la realizzazione di un classificatore XGBoost, che per un numero limitato di componenti principali è possibile distinguere i pazienti di controllo da quelli settici. Al crescere delle componenti questa differenza si annulla. Ciò è giustificato dal fatto che le differenze fra i due gruppi sono minime e quindi da ricercare nelle sole caratteristiche più rilevanti. Una volta dimostrato che i due gruppi di pazienti, dopo il passaggio nell'autoencoder, mantengono delle differenze, si sono approfondite tali peculiarità mediante

un nuovo modello di AI. Quest'ultimo, costituito da strati LSTM, ha permesso di classificare i pazienti settici e quelli di controllo con un'accuratezza del 76% ed un AUROC dell'83% permettendo in modo soddisfacente di distinguere i due gruppi.
In futuro è possibile studiare in modo più approfondito l'autoencoder come approccio alla modellazione del segnale e confrontarlo con i sistemi già presenti in letteratura per valutarne le prestazioni. Inoltre si può ampliare il modello proposto includendo i dati clinici dei pazienti in quanto molte delle architetture in letteratura ne fanno uso per migliorare la capacità predittiva del sistema. Uno studio successivo del modello presentato in questo lavoro di Tesi consiste sicuramente nel valutare un'analisi nel tempo dello scostamento del segnale rispetto alle condizioni di normalità, per evidenziare in modo precoce l'insorgenza della sepsi e consentire un intervento tempestivo, condizione fondamentale per la sopravvivenza del paziente.

Bibliografia

[1] John Allen. “Photoplethysmography and its application in clinical physiological measurement”. In: *Physiological measurement* 28.3 (2007), R1.

[2] Yoshua Bengio, Patrice Simard e Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.

[3] Jacob S Calvert et al. “A computational approach to early sepsis detection”. In: *Computers in biology and medicine* 74 (2016), pp. 69–73.

[4] Leo Anthony Celi et al. ““Big data” in the intensive care unit. Closing the data loop”. In: *American journal of respiratory and critical care medicine* 187.11 (2013), p. 1157.

[5] Thomas Desautels et al. “Prediction of sepsis in the intensive care unit with minimal electronic health record data: a machine learning approach”. In: *JMIR medical informatics* 4.3 (2016), e5909.

[6] Mohamed Elgendi. “Optimal signal quality index for photoplethysmogram signals”. In: *Bioengineering* 3.4 (2016), p. 21.

[7] Mohamed Elgendi et al. “Systolic peak detection in acceleration photoplethysmograms measured from emergency responders in tropical conditions”. In: *PLoS One* 8.10 (2013), e76585.

[8] Jeffrey L Elman. “Finding structure in time”. In: *Cognitive science* 14.2 (1990), pp. 179–211.

[9] Lucas M Fleuren et al. “Machine learning for the prediction of sepsis: a systematic review and meta-analysis of diagnostic test accuracy”. In: *Intensive care medicine* 46.3 (2020), pp. 383–400.

[10] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* ” O’Reilly Media, Inc.”, 2019.

[11] Ary L Goldberger et al. “PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals”. In: *circulation* 101.23 (2000), e215–e220.

[12] Alex Graves. “Generating sequences with recurrent neural networks”. In: *arXiv preprint arXiv:1308.0850* (2013).

[13] C William Hanson III e Bryan E Marshall. “Artificial intelligence applications in the intensive care unit”. In: *Critical care medicine* 29.2 (2001), pp. 427–435.

[14] Alistair EW Johnson et al. “MIMIC-III, a freely accessible critical care database”. In: *Scientific data* 3.1 (2016), pp. 1–9.

[15] Hye Jin Kam e Ha Young Kim. “Learning representations for the early detection of sepsis with deep neural networks”. In: *Computers in biology and medicine* 89 (2017), pp. 248–255.

[16] Qiao Li e Gari D Clifford. “Dynamic time warping and machine learning for signal quality assessment of pulsatile signals”. In: *Physiological measurement* 33.9 (2012), p. 1491.

[17] Qingqing Mao et al. “Multicentre validation of a sepsis prediction algorithm using only vital sign data in the emergency department, general ward and ICU”. In: *BMJ open* 8.1 (2018), e017833.

[18] Greg S Martin et al. “The epidemiology of sepsis in the United States from 1979 through 2000”. In: *New England Journal of Medicine* 348.16 (2003), pp. 1546–1554.

[19] Andrea McCoy e Ritankar Das. “Reducing patient mortality, length of stay and readmissions through machine learning-based sepsis prediction in the emergency department, intensive care unit and hospital floor units”. In: *BMJ open quality* 6.2 (2017), e000158.

[20] B Moody et al. *The MIMIC-III Waveform Database Matched Subset, physionet. org, 2017. doi: 10.13026.*

[21] Michael Moor et al. “Early prediction of sepsis in the ICU using machine learning: a systematic review”. In: *Frontiers in medicine* 8 (2021), p. 348.

[22] Nosheen Nasir et al. “Mortality in sepsis and its relationship with gender”. In: *Pakistan journal of medical sciences* 31.5 (2015), p. 1201.

[23] Michael Paul et al. “Modeling photoplethysmographic signals in camera-based perfusion measurements: optoelectronic skin phantom”. In: *Biomedical optics express* 10.9 (2019), pp. 4353–4368.

[24] Kristina E Rudd et al. “Global, regional, and national sepsis incidence and mortality, 1990–2017: analysis for the Global Burden of Disease Study”. In: *The Lancet* 395.10219 (2020), pp. 200–211.

[25] Yasser Sakr et al. “Sepsis in intensive care unit patients: worldwide data from the intensive care over nations audit”. In: *Open forum infectious diseases*. Vol. 5. 12. Oxford University Press US. 2018, ofy313.

[26] Shigehiko Schamoni et al. “Leveraging implicit expert knowledge for non-circular machine learning in sepsis prediction”. In: *Artificial intelligence in medicine* 100 (2019), p. 101725.

[27] Jörg Schröder et al. “Gender differences in human sepsis”. In: *Archives of surgery* 133.11 (1998), pp. 1200–1205.

[28] Mervyn Singer et al. “The third international consensus definitions for sepsis and septic shock (Sepsis-3)”. In: *Jama* 315.8 (2016), pp. 801–810.