Relazione del progetto relativo l'insegnamento di Applicazioni dell'Intelligenza Artificiale per la sessione autunnale a.a. 2023/2024.

Corso di Laurea Magistrale in Informatica Applicata
Università degli studi di Urbino "Carlo Bo"

Relatore:
Matteo Marco Montanari
(Matricola 323293)

Docenti:
Prof. Stefano Ferretti
Prof.ssa. Sara Montagna

1) Obiettivo del progetto

Si vuole progettare e addestrare una rete neurale profonda basata su Convolutional Neural Networks e Long Short Term Memory Networks per la valutazione di qualità di segnali ECG dinamici e di lunga durata. Consultando la letteratura di riferimento, l'obiettivo del progetto è replicare in buona parte il lavoro svolto da He, C. et Al. intitolato *"Dynamic Electrocardiogram Signal Quality Assessment Method Based on Convolutional Neural Network and Long Short-Term Memory Network"* [1] su un diverso dataset e utilizzando il framework PyTorch al posto di TensorFlow. Il dataset scelto per il progetto è *"Brno University of Technology ECG Quality Database (BUT QDB)"* [2] raccolto tramite la banca dati PhysioNet [3].

2) Dataset utilizzato

Il dataset utilizzato per il progetto è *"Brno University of Technology ECG Quality Database (BUT QDB)"* [2] raccolto tramite la banca dati PhysioNet [3].

BUT QDB è un database creato dal team di cardiologia del Dipartimento di Ingegneria Biomedica della Brno University of Technology, allo scopo di valutare la qualità di segnali ECG. I dati comprendono 18 registrazioni a lungo termine di segnali ECG a derivazione singola e i dati associati di un accelerometro a 3 assi, raccolti da 15 soggetti (9 donne, 6 uomini) di età compresa tra 21 e 83 anni. Le registrazioni sono state effettuate tra agosto 2018 e ottobre 2019 mentre i soggetti svolgevano normali attività quotidiane (*"free living conditions"*). I dati sono stati raccolti utilizzando un ECG mobile e un accelerometro (Bittium Faros 180) con una frequenza di campionamento di 1.000 Hz per i segnali ECG e 100 Hz per i segnali dell'accelerometro.

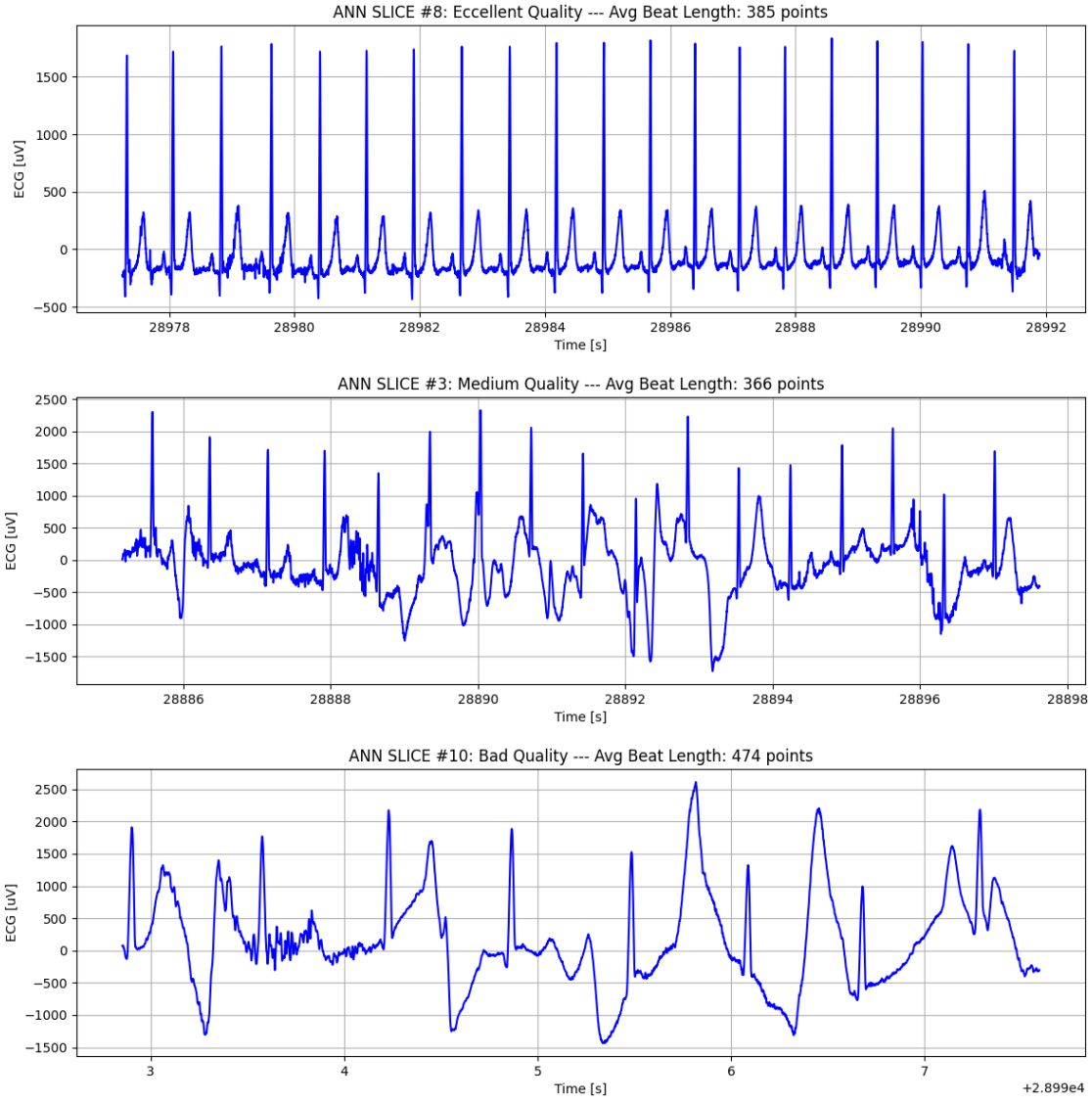
Per questo progetto, i dati degli accelerometri non sono stati utilizzati perché non rilevanti agli scopi previsti.

Tutti i dati sono forniti nel formato WaveForm Database (WFDB) e sono distribuiti liberamente sotto licenza Creative Commons Attribution (CC-BY-4.0) attraverso la banca dati PhysioNet.

Tre segnali, dalla durata minima di 24 ore, sono stati completamente annotati in termini di qualità del segnale ECG. I restanti 15 segnali sono stati annotati in due segmenti selezionati, ciascuno della durata di 20 minuti. Inoltre, sono stati annotati anche cinque segmenti aggiuntivi di scarsa qualità del segnale. La qualità del segnale è stata classificata come segue:

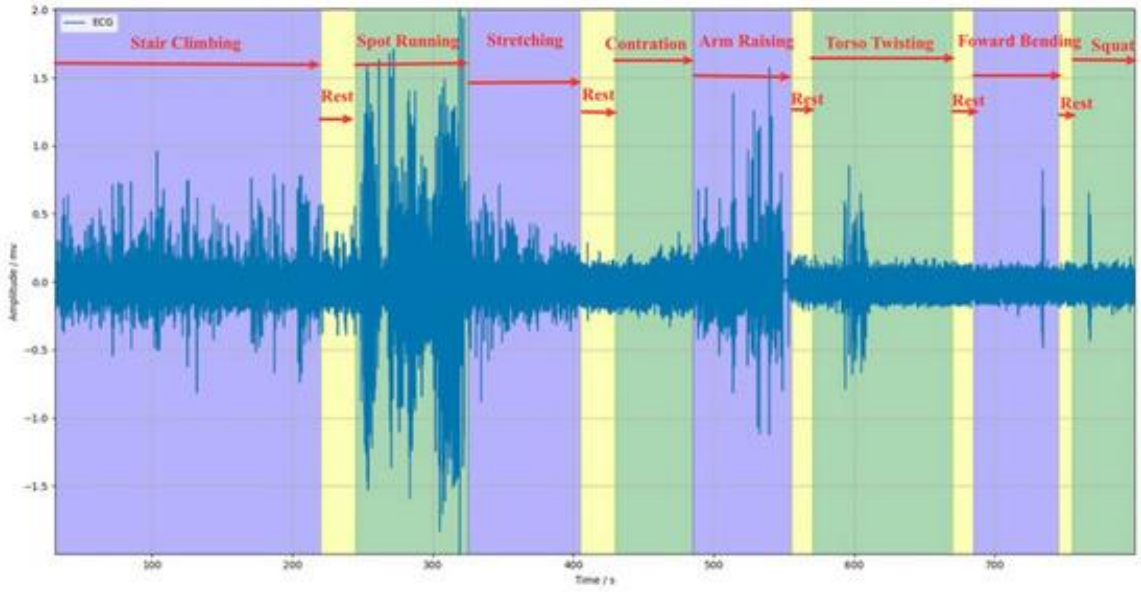
1. **Classe 1 (Eccellent Quality)**: indica che tutte le forme d'onda significative dell'ECG (onda P, onda T e complesso QRS) sono chiaramente visibili e i loro inizi e offset possono essere rilevati in modo affidabile;
2. **Classe 2 (Medium Quality)**: indica che vi è un maggiore livello di rumore rispetto alla classe precedente e i punti significativi dell'ECG non possono essere individuati in modo affidabile, ma il segnale consente comunque un rilevamento QRS affidabile;
3. **Classe 3 (Bad Quality)**: indica che i complessi QRS non possono essere rilevati in modo affidabile e il segnale non è adatto per ulteriori analisi.

La figura seguente mostra una porzione di segnale ECG per ognuna delle 3 classi di qualità individuate.



In particolare, le registrazioni ECG che compongono questo database sono state raccolte per un periodo di tempo prolungato mentre i soggetti svolgevano normali attività quotidiane (*"free living conditions"*). Per questo motivo il segnale prodotto è chiamato *ECG dinamico di lunga durata*.

La figura seguente mostra un esempio di segnale *ECG dinamico* con annotate le *attività* corrispondenti svolte dal paziente in questione.



Il database è destinato allo sviluppo e al confronto di algoritmi per valutare la qualità delle registrazioni ECG. Una delle caratteristiche uniche di questo database è che la qualità dei segnali ECG è annotata campione per campione (i segnali hanno una qualità variabile nel tempo).

Le annotazioni di *qualità* del segnale ECG sono state registrate in un file CSV con 12 colonne. Per ogni annotatore sono state assegnate tre colonne mentre le ultime tre colonne sono per il consenso (3 colonne x 3 annotatori + 3 colonne x consenso). La prima colonna per ogni annotatore contiene l'indice che identifica il primo campione del segmento annotato; la seconda colonna contiene l'indice che identifica il campione finale del segmento annotato; e la terza colonna contiene la classe di qualità assegnata (1, 2 o 3; 0 significa che la qualità non è stata annotata in questo segmento).

Per il progetto si è deciso di considerare solamente le annotazioni relative alle colonne di consenso (ignorando quelle dei singoli annotatori).

La figura seguente mostra un file di annotazione per uno dei segnali ECG che compongono il dataset (le ultime tre colonne sono quelle di consenso).

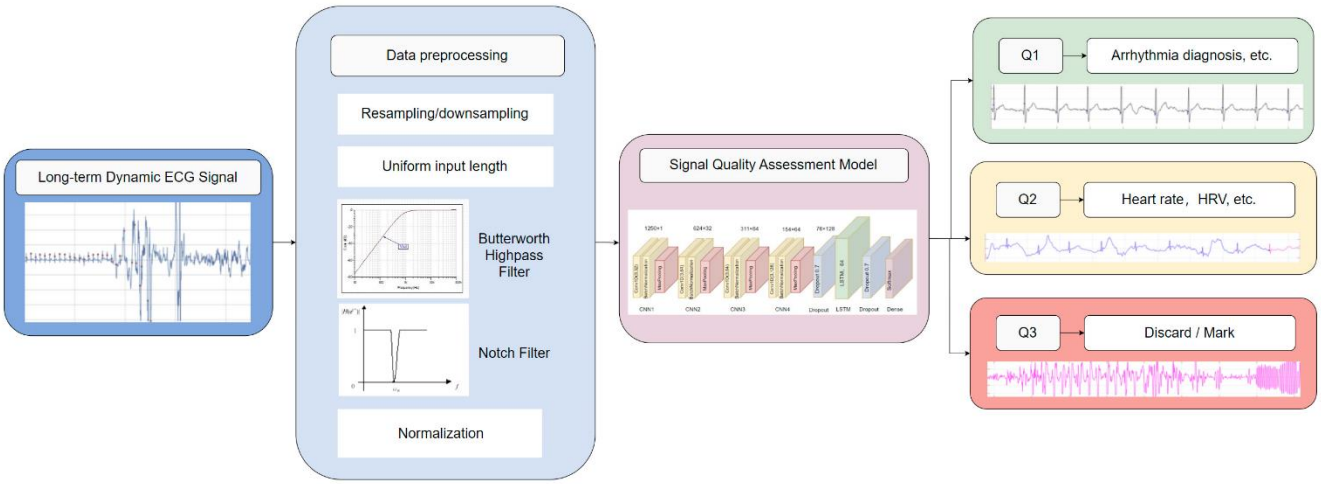
File: <base> / 104001 / 104001_ANN.csv (7,287 bytes)

1	28800000	0	1	28800000	0	1	28800000	0	1	28800000	0
28800001	29028285	1	28800001	29029015	1	28800001	29028996	1	28800001	29028996	1
29028286	29041643	2	29029016	29038208	2	29028997	29036404	2	29028997	29038208	2
29041644	29830120	1	29038209	29833496	1	29036405	29041070	1	29038209	29041070	1
29830121	29843829	2	29833497	29842799	2	29041071	29041373	2	29041071	29041373	2
29843830	30000000	1	29842800	30000000	1	29041374	29833350	1	29041374	29833350	1
30000001	57600000	0	30000001	57600000	0	29833351	29842799	2	29833351	29842799	2
57600001	57702386	2	57600001	57614269	1	29842800	29847171	1	29842800	30000000	1
57702387	57704739	3	57614270	57614654	2	29847172	29847839	2	30000001	57600000	0
57704740	58265306	2	57614655	57617910	1	29847840	30000000	1	57600001	57614172	1
58265307	58268218	3	57617911	57618319	2	30000001	57600000	0	57614173	57614654	2
58268219	58800000	2	57618320	57620535	1	57600001	57614172	1	57614655	57617910	1
58800001	87178000	0	57620536	57621106	2	57614173	57614637	2	57617911	57618319	2

3) Stato dell'arte

Il progetto ricalca in buona parte il lavoro svolto da He, C.;Wei, Y.;Wei, Y.; Liu, Q.; An, X. intitolato *"Dynamic Electrocardiogram Signal Quality Assessment Method Based on Convolutional Neural Network and Long Short-Term Memory Network"* e pubblicato su "*Big Data and Cognitive Computing".*

Nella figura seguente è mostrata la pipeline proposta nell'articolo, che è la stessa seguita per la realizzazione di questo progetto. Nella prima fase, più segnali ECG dinamici di lunga durata vengono sottoposti inizialmente ad una fase di preprocessing (sottocampionamento, finestratura, filtraggio, normalizzazione) per adattare il dataset originale al modello di rete neurale che si vuole utilizzare. Nella seconda fase, una volta preprocessato il dataset, viene addestrata una rete neurale deep su tali dati in modo da ottenere un modello tramite il quale poter fare classificazione di qualità del segnale ECG. Tale classificazione sarà poi utile per lo studio di fattibilità di analisi specifiche su questi tracciati ECG.



In particolare, la fase di ***preprocessing*** è composta dalle seguenti sottofasi:

1. Un segnale ECG dinamico di lunga durata viene prima suddiviso in segmenti utilizzando una finestra scorrevole di 10 secondi (senza sovrapposizioni) per garantire che i dati in ingresso al modello siano di dimensione costante.
2. I segmenti così ottenuti vengono poi sottocampionati per ridurre la frequenza di campionamento di tutti i segnali a un valore costante di 250 Hz (rispetto all'originale di 1000 Hz). In questo modo i dati potranno essere elaborati in maniera più efficiente.

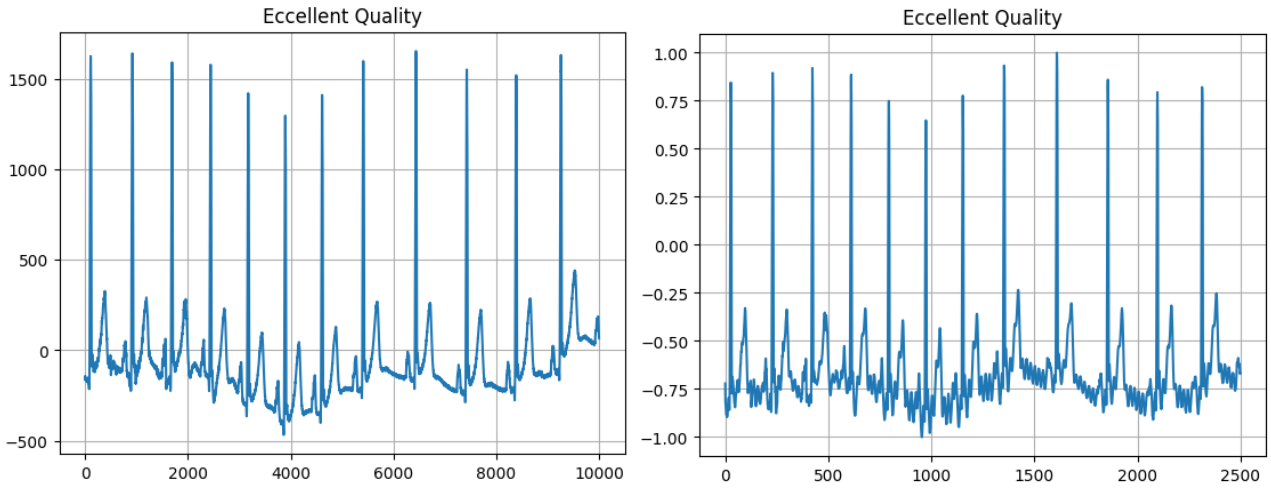
3. In seguito, tali dati vengono filtrati mediante un filtro passa-alto di tipo Butterworth da 0,67 Hz e successivamente passano attraverso un ulteriore filtro Notch da 50 Hz per ridurre le interferenze più facilmente rimovibili generate durante la misurazione.
4. I segnali filtrati vengono quindi normalizzati nell'intervallo [−1,1] attraverso una normalizzazione min–max:

$$x = \frac{x - min}{max - min}$$

5. Infine, i dati vengono suddivisi randomicamente in dati di addestramento, test e validazione, rispettivamente nel rapporto: 70% - 20% - 10%.

Il preprocessing dei dati ha lo scopo di accelerare la velocità di convergenza del modello, ridurre la scomparsa o l'esplosione del gradiente, migliorare la capacità di generalizzazione del modello e aumentarne l'adattabilità.

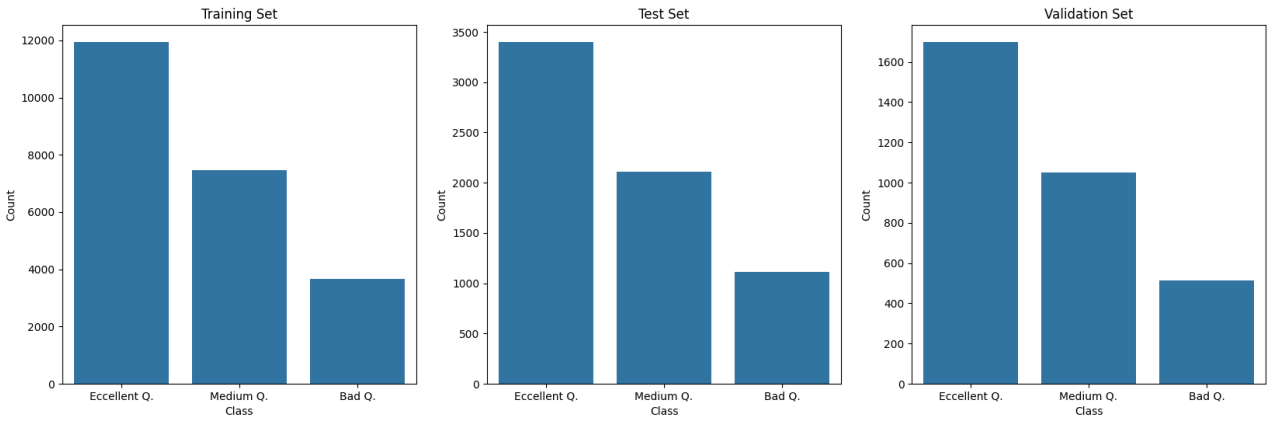
Nella figura seguente viene mostrato un campione di segnale ECG prima e dopo la fase di preprocessing. I dati processati saranno l'input per il modello di rete neurale che andremo ad addestrare.



Prima del preprocessing

Dopo il preprocessing

Dopo la fase di preprocessing è possibile notare che il dataset utilizzato, così come è stato costruito, risulta fortemente sbilanciato sulle diverse classi di qualità del segnale ECG. Questo aspetto dovrà essere tenuto in considerazione nella fase di addestramento del modello. Nella figura seguente sono mostrati tre istogrammi (uno per ogni insieme tra addestramento, test e validazione) che rappresentano la distribuzione dei campioni preprocessati sulle tre classi di qualità individuate.



Seguendo il lavoro proposto nell'articolo, si è scelto di addestrare una rete neurale profonda composta da strati convoluzionali (CNN) e LSTM. I layer CNN sono stati utilizzati per automatizzare il processo di *Feature Extraction* a partire dai dati grezzi, senza doverlo effettuare manualmente ma facendo apprendere alla rete i parametri corretti tramite gli strati convoluzionali. Il layer LSTM è stato utilizzato con lo scopo di catturare non solo le variazioni individuabili tra battito e battito, ma anche le fluttuazioni della frequenza cardiaca a lungo termine rilevate sullo stesso segnale a diversi istanti di tempo.

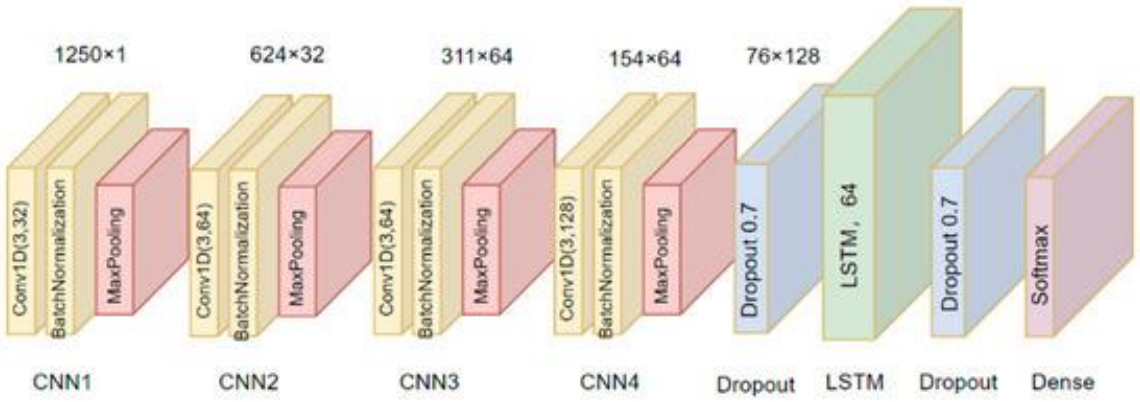
La rete neurale proposta in questo progetto utilizza anche una serie di *layer di normalizzazione batch (BNL)* subito dopo ciascun livello convoluzionale. Questo layer viene introdotto per migliorare ulteriormente la stabilità e l'efficienza del processo di training normalizzando ogni batch di dati. Subito dopo tale livello è applicato uno strato di Max-Pooling per evidenziare le caratteristiche del segnale più importanti e per ridurre il costo computazionale. Attraverso quattro blocchi di layer convoluzionale, di normalizzazione batch e di pooling, la rete individua una gerarchia di features da fornire come input allo strato LSTM. Quest'ultimo livello, in quanto rete neurale ricorrente (RNN), è progettato per individuare le dipendenze a lungo termine nelle serie temporali come i segnali ECG. Infine, l'output di classificazione è ottenuto tramite uno strato completamente connesso e le probabilità corrispondenti a ciascuna classe sono calcolate

utilizzando la funzione di attivazione SoftMax. Per ridurre l'overfitting si utilizzano due strati di *dropout* con probabilità 0.7: uno prima e uno dopo il layer LSTM. Per l'addestramento si utilizzano l'ottimizzatore *Adam* e la funzione *Cross Entropy* come funzione di perdita. Infine, per migliorare l'efficienza di training e la capacità di generalizzazione del modello, si è scelto di utilizzare la strategia *Early Exit*: quando le prestazioni sul *validation set* non mostrano alcun miglioramento per più epoche consecutive, il training viene terminato in anticipo per evitare l'*overfitting* sui dati di addestramento.

Pertanto, *l'**architettura*** di rete neurale utilizzata è la seguente:

1. Strato ***Convoluzionale*** 1D con 32 kernel di dimensione 3 seguito da uno strato di Batch Normalization e uno di Max Pooling di dimensione 2.
2. Strato ***Convoluzionale*** 1D con 64 kernel di dimensione 3 seguito da uno strato di Batch Normalization e uno di Max Pooling di dimensione 2.
3. Strato ***Convoluzionale*** 1D con 64 kernel di dimensione 3 seguito da uno strato di Batch Normalization e uno di Max Pooling di dimensione 2.
4. Strato ***Convoluzionale*** 1D con 128 kernel di dimensione 3 seguito da uno strato di Batch Normalization e uno di Max Pooling di dimensione 2.
5. Strato di ***Dropout*** con probabilità pari a 0.7.
6. Singolo strato ***LSTM*** con 64 neuroni.
7. Strato di ***Dropout*** con probabilità pari a 0.7.
8. Strato ***Denso*** finale con 64 neuroni e funzione di attivazione SoftMax.

La figura seguente mostra graficamente l'architettura di rete neurale appena descritta:



4) Codice sviluppato

Il codice è stato sviluppato in Cloud tramite il framework PyTorch su Google Colaboratory ed è reperibile al repository pubblico GitHub: https://github.com/MatteoMarcoM/Progetto_AIA [4].

Nella figura seguente viene mostrato uno spezzone di codice Python che utilizza il framework PyTorch per implementare la rete neurale seguendo l'architettura descritta nella sezione precedente.

```python
class CNN_LSTM(nn.Module):
    def __init__(self, lstm_hidden_size, lstm_num_layers, num_classes):
        super(CNN_LSTM, self).__init__()

        self.lstm_hidden_size = lstm_hidden_size
        self.lstm_num_layers = lstm_num_layers
        self.num_classes = num_classes
        # length of a single sample
        self.ecg_sample_len = int(df_train.shape[1] - 1)   # 2500

        # Definizione dei layer CNN
        self.conv1 = nn.Conv1d(in_channels=1, out_channels=32, kernel_size=3)
        self.bn1 = nn.BatchNorm1d(32)
        self.pool1 = nn.MaxPool1d(kernel_size=2)

        self.conv2 = nn.Conv1d(in_channels=32, out_channels=64, kernel_size=3)
        self.bn2 = nn.BatchNorm1d(64)
        self.pool2 = nn.MaxPool1d(kernel_size=2)

        self.conv3 = nn.Conv1d(in_channels=64, out_channels=64, kernel_size=3)
        self.bn3 = nn.BatchNorm1d(64)
        self.pool3 = nn.MaxPool1d(kernel_size=2)

        self.conv4 = nn.Conv1d(in_channels=64, out_channels=128, kernel_size=3)
        self.bn4 = nn.BatchNorm1d(128)
        self.pool4 = nn.MaxPool1d(kernel_size=2)

        # Calcolo della dimensione dell'input per LSTM dopo i layer CNN
        self.lstm_input_size = self._get_conv_output_size()

        # Definizione dei layer LSTM e Dropout
        self.dropout1 = nn.Dropout(0.7)  # Dropout prima del LSTM
        self.lstm = nn.LSTM(input_size=self.lstm_input_size, hidden_size=self.lstm_hidden_size, num_layers=self.lstm_num_layer
        self.dropout2 = nn.Dropout(0.7)  # Dropout dopo il LSTM

        # Definizione dello strato completamente connesso
        self.fc = nn.Linear(self.lstm_hidden_size, self.num_classes)
```

5) Risultati ottenuti

Gli indicatori di performance utilizzati per valutare il modello sono quelli standard utilizzati nei problemi di classificazione come quello analizzato. In particolare, la metrica più rilevante per valutare complessivamente l'efficacia del modello su dataset sbilanciati come quello utilizzato è la MacroF1:

$$Accuracy = \frac{TP_1 + TP_2 + TP_3}{TP_1 + TP_2 + TP_3 + FN_1 + FN_2 + FN_3 + FP_1 + FP_2 + FP_3}$$

$$MacroPrecision = \frac{Precision_1 + Precision_2 + Precision_3}{3}$$

$$\text{MacroRecall} = \frac{Recall_1 + Recall_2 + Recall_3}{3}$$

$$MacroF1 = \frac{F1_1 + F1_2 + F1_3}{3}$$

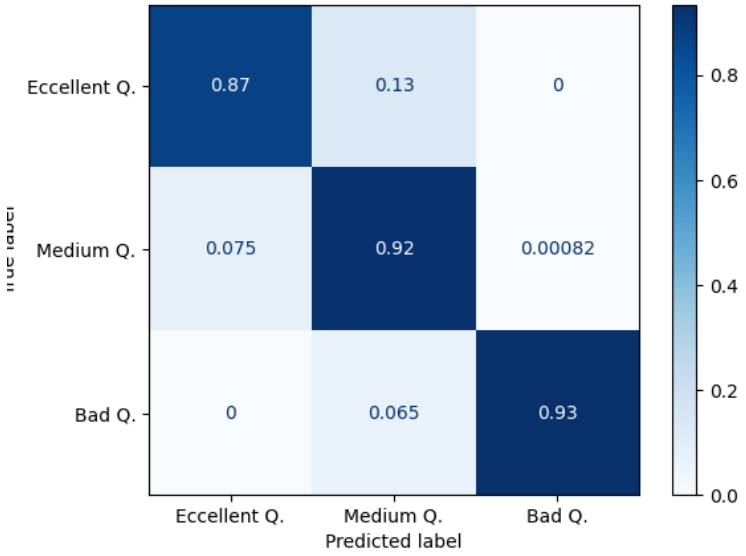
*E inoltre (TP, TN, FP, FN sono calcolati tramite la tecnica **one-versus-all**):*

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \qquad (i = 1,2,3)$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \qquad (i = 1,2,3)$$

$$F1_i = 2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i} \qquad (i = 1,2,3)$$

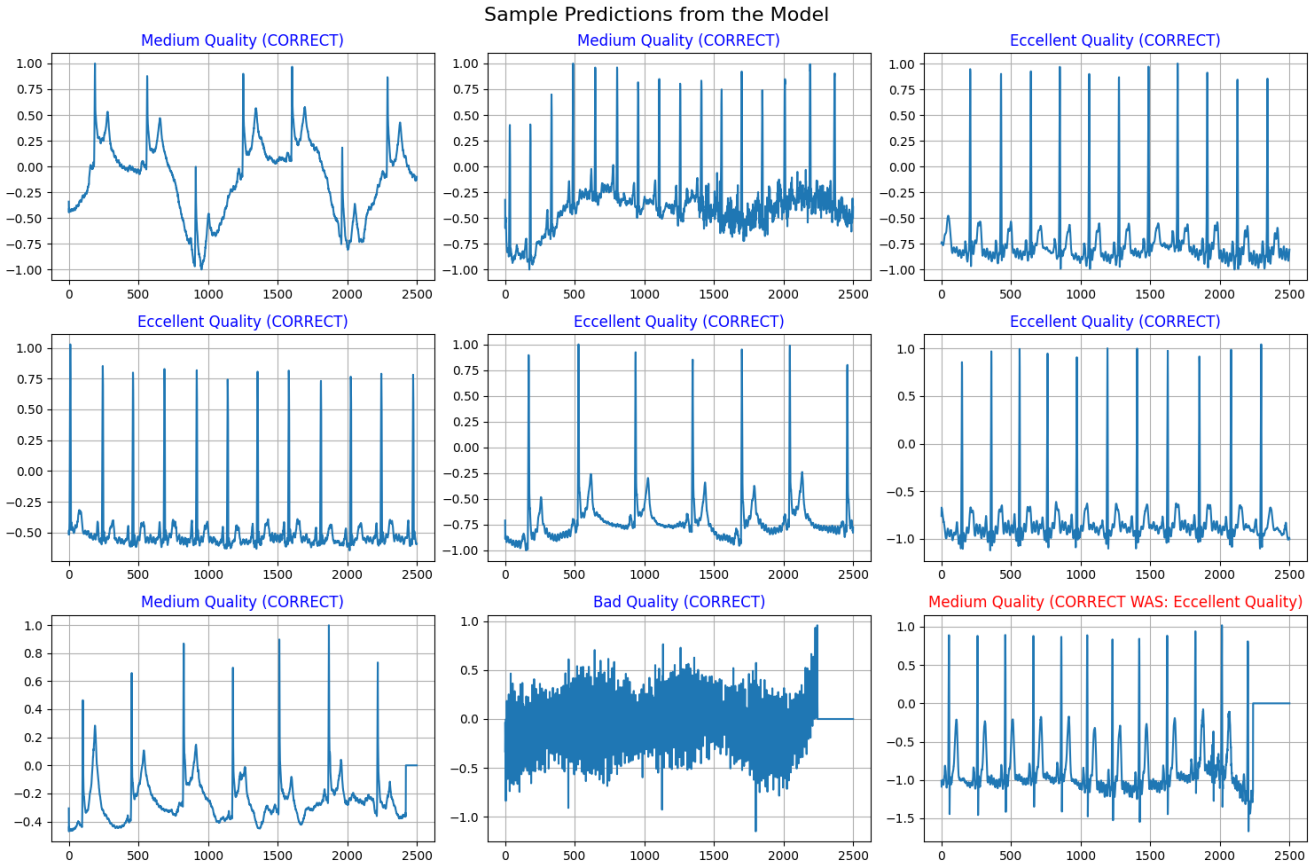
La figura seguente mostra la *matrice di confusione*, che ci permette di valutare l'accuratezza complessiva della rete neurale in riferimento al *test set*.



La figura seguente mostra tutte le metriche di valutazione del modello sopra riportate, applicate alla rete neurale sul *dataset di test* (una volta completato l'addestramento).

	precision	recall	f1-score	support
Eccellent Q.	0.95	0.87	0.91	11821
Medium Q.	0.79	0.92	0.85	7326
Bad Q.	1.00	0.93	0.97	3922
accuracy			0.90	23069
macro avg	0.91	0.91	0.91	23069
weighted avg	0.91	0.90	0.90	23069

La figura seguente mostra 9 campioni di *test* estratti casualmente, assieme alla classe predetta dalla rete una volta addestrata sui dati di training.



Questi risultati mostrano come la rete neurale sviluppata ed addestrata sul dataset di riferimento ottiene delle ottime prestazioni sia in termini di accuratezza globale sia per quanto riguarda la Macro Average F1, ottenendo dei valori sul *test set* rispettivamente di 90% e 91%. Questo è un buonissimo risultato anche considerando lo sbilanciamento iniziale del dataset, fatto che conferma l'efficacia dei metodi e delle tecniche utilizzate in questo progetto.

Infine, gli *sviluppi futuri* di un progetto di questo tipo potrebbero riguardare l'utilizzo di altre tipologie di rete neurale da utilizzare per creare modelli di classificazione, oltre alla possibilità di applicare la rete neurale sviluppata per questo progetto ad altri dataset.

6) Riferimenti Bibliografici

[1] He, C.;Wei, Y.;Wei, Y.; Liu, Q.; An, X. Dynamic Electrocardiogram Signal Quality Assessment Method Based on Convolutional Neural Network and Long Short-Term Memory Network. *Big Data Cogn. Comput*. 2024, 8, 57. https://doi.org/10.3390/bdcc8060057.

[2] Nemcova, A., Smisek, R., Opravilová, K., Vitek, M., Smital, L., & Maršánová, L. (2020). Brno University of Technology ECG Quality Database (BUT QDB) (version 1.0.0). *PhysioNet*. https://doi.org/10.13026/kah4-0w24.

[3] Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., ... & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220.

[4] Matteo Marco Montanari, Project for the Exam of Application of Artificial Intelligence, a.y. 2023/2024. GitHub Repository. https://github.com/MatteoMarcoM/Progetto_AIA

Termine della relazione