

Framework Convolutional Neural Network

Il modello utilizzato per la classificazione binaria (tracciato Normale o tracciato Anormale) di ECG è una rete convoluzionale sequenziale con 5 layer, che ha la seguente struttura:

-I primi 4 layer sono composti da:

- Conv1D
- BatchNorm1D
- MaxPool1D
- Funzione di attivazione RELU

-L'ultimo layer è composto:

- AveragePooling1D
- Layer Flatten
- Layer Dense con funzione di attivazione Soft Max

Implementazione della CNN

```
'''layer 1'''
cnn = Sequential()
cnn.add(Conv1D(2**num_unit, kernel_size=initial_kernel_size, strides=strides, input_shape=(10800,1)))
cnn.add(BatchNormalization())
cnn.add(MaxPooling1D(pool_size=maxpooling_poolsize))
cnn.add(Activation(activation))

'''layer 2'''
cnn.add(Conv1D(2**num_unit, kernel_size=kernel_size))
cnn.add(BatchNormalization())
cnn.add(MaxPooling1D(pool_size=maxpooling_poolsize))
cnn.add(Activation(activation))

'''layer 3'''
cnn.add(Conv1D(2**(num_unit+1), kernel_size=kernel_size))
cnn.add(BatchNormalization())
cnn.add(MaxPooling1D(pool_size=maxpooling_poolsize))
cnn.add(Activation(activation))

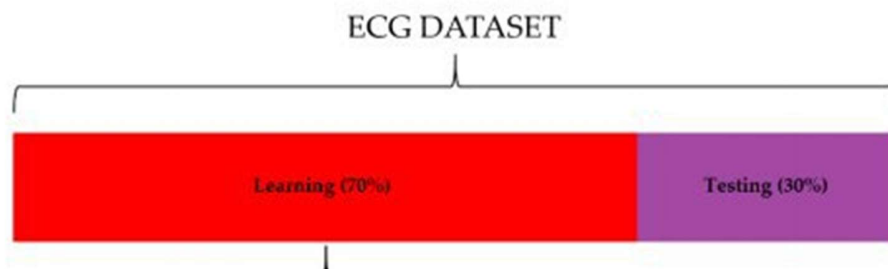
'''layer 4'''
cnn.add(Conv1D(2**(num_unit+2), kernel_size=kernel_size))
cnn.add(BatchNormalization())
cnn.add(MaxPooling1D(pool_size=maxpooling_poolsize))
cnn.add(Activation(activation))

'''layer 5'''
cnn.add(AveragePooling1D(pool_size = avg_poolsize))
cnn.add(Flatten())
cnn.add(Dense(n_classes, kernel_initializer=kernel_initializer, activation='softmax'))
return cnn
```

Training , Validation and Testing

prima di effettuare il training , la validazione e il test l' intero dataset e stato diviso in due parti così divise:

- 70% di learning set
- 30% di test set



Per la **validazione** degli iperparametri del modello è stata utilizzata la *convalida incrociata K-Fold* con $k=10$ sul set di Learning.

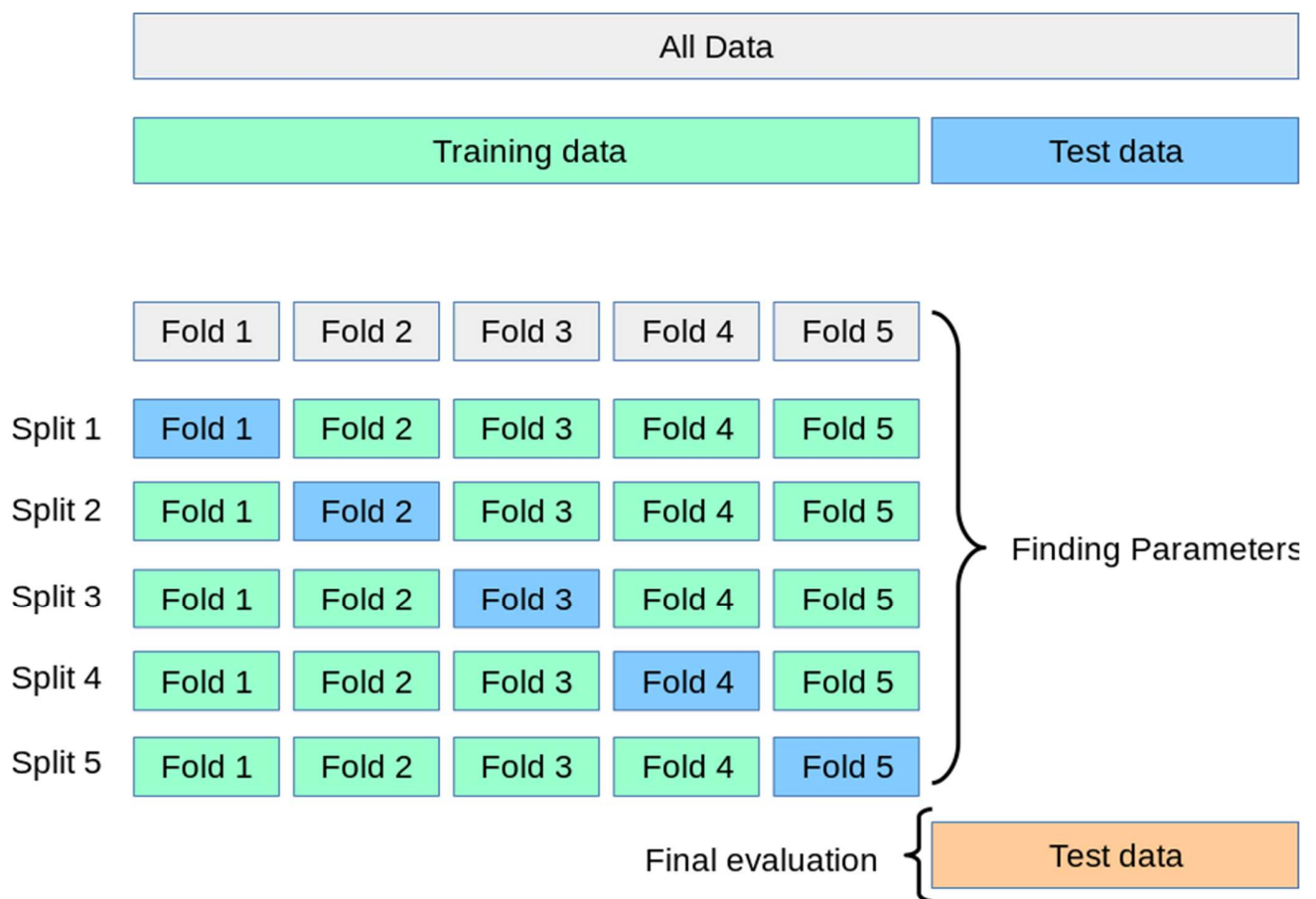
A ogni iterazione del k-Fold si sono effettuate le seguenti operazioni:

- Creazione della CNN
- Il learning set è stato diviso ulteriormente in k parti dove $k-1$ parti sono state utilizzate per l' addestramento(trainingSet) e la restante parte come validation set

```
for train, val in kfold.split(LearningX, LearningY):  
    model = CNN_loader()  
  
    trainingSet_X = LearningX[train]  
    trainingSet_y = LearningY[train]  
  
    validation_X = LearningX[val]  
    validation_Y = LearningY[val]
```

- Fit del modello con il *training set* e il *validation set*
- E' stata calcolata l' accuratezza e il recall

```
loss, acc, recall = model.evaluate( validation_X, validation_Y, batch_size=batch_size, verbose=1)  
ACCURACY.append(acc)  
LOSSLESS.append(loss)  
RECALL.append(recall)  
  
print("Model Results: ")  
print("lossless = " + str(loss))  
print("accuracy = " + str(acc))  
print("recall = " + str(recall))
```



Alla fine delle varie iterazioni si è calcolata la media e la deviazione standard delle accuratèzze e del recall.

Dopo aver effettuato la **validazione** è stata effettuata la **predizione**.

Per la **predizione** la rete è stata addestrata sull' intero learning set e successivamente si è effettuata la predizione per i dati del test set utilizzando il modello appena creato.

Figura1

Nella figura1 è presentato un esempio di convalida incrociata k-fold con k=5.

In fine si è visualizzata la matrice di confusione e le seguenti metriche:

- Precision
- Recall
- F1

Scelta iperparametri

Per la scelta degli iperparametri si è fatto riferimento all' articolo

"Automatic ECG Diagnosis Using Convolutional Neural Network di Roberta Avanzato and Francesco Beritelli " e gli iperparametri scelti inizialmente sono stati i seguenti:

```
#Parametres
#-----
EPOCHS = 100
BATCH_SIZE = 24
MAXPOOL_POOL_SIZE = 4
AVGPOOL_POOL_SIZE = 2
KERNEL_SIZE = 4
OPTIMIZER = tensorflow.keras.optimizers.Adam
ACTIVATION = 'relu'
KERNEL_INITIALIZER = 'normal'
STRIDES = 4
NUM_UNIT = 7 # 2^NUM_UNIT 128 unit
LOSS_FUNCTION = 'binary_crossentropy'
INITIAL_KERNEL_SIZE = 80
LEARNING_RATE = 0.1

#-----

def CNN_LOAD():
    model = createCNN(num_unit = NUM_UNIT,
                      initial_kernel_size = INITIAL_KERNEL_SIZE,
                      strides = STRIDES ,
                      kernel_initializer = KERNEL_INITIALIZER,
                      maxpooling_poolsize = MAXPOOL_POOL_SIZE ,
                      activation= ACTIVATION,
                      avg_poolsize = AVGPOOL_POOL_SIZE,
                      kernel_size = KERNEL_SIZE)
    model.compile(optimizer=OPTIMIZER(lr = LEARNING_RATE), loss=LOSS_FUNCTION, m
    return model
```

I risultati ottenuti con questi iperparametri sono stati i seguenti:

	0	1
0	221	46
1	52	473

	precision	recall	f1-score	support
N	0.81	0.83	0.82	267
A	0.91	0.90	0.91	525
accuracy			0.88	792
macro avg	0.86	0.86	0.86	792
weighted avg	0.88	0.88	0.88	792

Direcotry ".\tests\test_with_initial_params"

Test effettuati per il miglioramento del modello

Per migliorare i risultati ottenuti (visti in precedenza) sono stati effettuati una serie di test cambiando gli iperparametri della CNN.

I risultati considerati più rilevanti sono riassunti nella seguente tabella.

	Testing Result			Validation Result	
Test	Recall A	Recall N	Accuracy	avg_accuracy	avg_recall
1	0,99	0,40	0,80	0,834 +/- 0,029	0,956 +/- 0,002
2	0,90	0,83	0,88	0,889 +/- 0,023	0,968 +/- 0,002
3	0,91	0,72	0,85	0,841 +/- 0,036	0,961 +/- 0,002
4	0,94	0,77	0,89	0,861 +/- 0,045	0,964 +/- 0,002
5	0,95	0,77	0,88	0,754 +/- 0,096	0,901 +/- 0,003
6	0,95	0,85	0,92	0,765 +/- 0,072	0,891 +/- 0,003
7	0,98	0,51	0,83	0,845 +/- 0,038	0,925 +/- 0,003
8	0,91	0,78	0,87	0,856 +/- 0,026	0,967 +/- 0,003

Dataset con segmenti di 15 secondi

Il ogni tracciato è stato suddiviso in segmenti di 15 secondo e ogni segmento è stato etichettato.

In tutto si sono ottenuti i 5280 segmenti di cui :

- 2105 segmenti etichettati come N
- 3175 segmenti etichettati come A

L'intero dataset è stato suddiviso in una parte di learning e una parte di test con una percentuale di 70% per il learning e 30% per il test.

Sia nel learning set che nel test set sono state mantenute le proporzioni iniziali tra segmenti etichettati come A e segmenti etichettati come N.

Eseguendo il test con i gli iperparametri di partenza (seguendo l' articolo) si sono ottenuti i seguenti risultati:

Testing Result			Validation Result	
Recall A	Recall N	Accuracy	avg_accuracy	avg_recall
0.91	0.89	0.90	0.880 +/- 0.063	0.970 +/- 0.002

Sono stati effettuati altri test cambiando gli iperparametri e i risultati sono presentati nella seguente tabella.

	Testing Result			Validation Result	
Test	Recall A	Recall N	Accuracy	avg_accuracy	avg_recall
1	0,88	0,86	0,88	0,784 +/- 0,103	0,927 +/- 0,002
2	0,90	0,88	0,89	0,892 +/- 0,022	0,971 +/- 0,004
6	1.00	0,30	0,72	0,788 +/- 0,076	0,868 +/- 0,002
7	0,99	0,38	0,75	0,728 +/- 0,065	0,901 +/- 0,003

Successivamente al test set sono stati eliminati alcuni segmenti etichettati come A per ottenere una proporzione di 60% di segmenti etichettati come N e il 40% di segmenti etichettati come A.

Eseguedo il test con i gli iperparametri di partenza (seguendo l' articolo) si sono ottenuti i seguenti risultati:

Testing Result			Validation Result	
Recall A	Recall N	Accuracy	avg_accuracy	avg_recall
0,90	0,92	0,91	0,909 +/- 0,010	0,969 +/- 0,003

Si sono effettuati altri test cambiando gli iperparametri e i risultati ottenuti sono presentati nella seguente tabella:

	Testing Result			Validation Result	
Test	Recall A	Recall N	Accuracy	avg_accuracy	avg_recall
1	0,75	0,94	0,87	0,781 +/- 0,102	0,907 +/- 0,002
4	0,91	0,85	0,88	0,887 +/- 0,021	0,967 +/- 0,001
6	0,80	0,96	0,89	0,821 +/- 0,037	0,866 +/- 0,004
7	0,83	0,88	0,86	0,851 +/- 0,019	0,899 +/- 0,003
9	0,97	0,63	0,76	0,847 +/- 0,045	0,926 +/- 0,005
10	1.00	0,51	0,70	0,858 +/- 0,044	0,926 +/- 0,003
12	0,90	0,91	0,91	0,864 +/- 0,034	0,942 +/- 0,003
13	0,90	0,80	0,94	0,751 +/- 0,083	0,900 +/- 0,004

Lettura ECG classificati come Normali ma che sono Anomali

In ogni cartella relativo un file di testo in cui sono indicati il numero del tracciato , il numero del segmento all'interno del tracciato con il campione di partenza e il campione finale e le annotazioni su quel segmento.