

# Tool CNN per analisi e predizione di crisi epilettiche

**Manuel Cretone, Emilio Silvestri**  
**Tutor: prof Marco Piangerelli**  
Group project





# OUTLINE

## Introduzione

Epilessia

Il progetto: obiettivi

Workflow progetto

## Reti Neurali

Reti Neurali

Reti Neurali Convoluzionali

## Training

Dataset

## Tecnologie

Backend

Frontend

## Il software

Tool analisi

Tool predizione

Tool training

# EPILESSIA

Malattia neurologica che colpisce 50 milioni di persone nel mondo.

Caratterizzata da specifici eventi clinici: crisi epilettiche

- ▶ Alterazione del normale funzionamento dell'encefalo
- ▶ Tipi di crisi epilettiche:
  - ▶ Parziali: una singola parte
  - ▶ Generalizzate: entrambi gli emisferi

# EEG E EDF

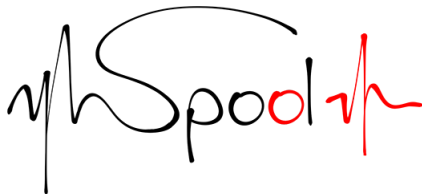
Diagnosi epilessia tramite analisi elettroencefalogramma  
Segnali generati dalla differenza di potenziale degli elettrodi applicati allo scalpo

- ▶ Registra attività elettrica dell'encefalo
- ▶ Riproduce segnali su tracciati grafici

Uso di file .edf (European Data Format)

- ▶ Archiviazione dei segnali registrati
- ▶ Caratteristiche della registrazione (frequenza di campionamento, durata...)

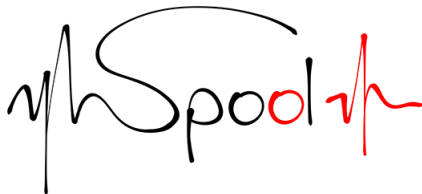
# IL PROGETTO: OBIETTIVI



Il programma realizzato ha tre scopi principali:

- ▶ Offrire una visione grafica di file .edf contenenti tracciati elettroencefalografici
- ▶ Ridurre la fase di individuazione di crisi epilettiche all'interno di un tracciato EEG
- ▶ Offrire un ambiente di allenamento di reti neurali personalizzate per l'individuazione di crisi epilettiche

# IL PROGETTO: OBIETTIVI



Il programma realizzato ha tre scopi principali:

- ▶ Offrire una visione grafica di file .edf contenenti tracciati elettroencefalografici
- ▶ Ridurre la fase di individuazione di crisi epilettiche all'interno di un tracciato EEG
- ▶ Offrire un ambiente di allenamento di reti neurali personalizzate per l'individuazione di crisi epilettiche

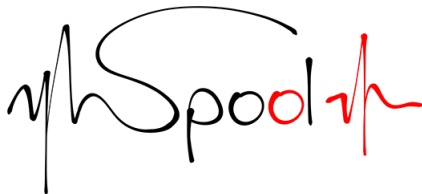
# IL PROGETTO: OBIETTIVI



Il programma realizzato ha tre scopi principali:

- ▶ Offrire una visione grafica di file .edf contenenti tracciati elettroencefalografici
- ▶ Ridurre la fase di individuazione di crisi epilettiche all'interno di un tracciato EEG
- ▶ Offrire un ambiente di allenamento di reti neurali personalizzate per l'individuazione di crisi epilettiche

# IL PROGETTO: OBIETTIVI



Il programma realizzato ha tre scopi principali:

- ▶ Offrire una visione grafica di file .edf contenenti tracciati elettroencefalografici
- ▶ Ridurre la fase di individuazione di crisi epilettiche all'interno di un tracciato EEG
- ▶ Offrire un ambiente di allenamento di reti neurali personalizzate per l'individuazione di crisi epilettiche



# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

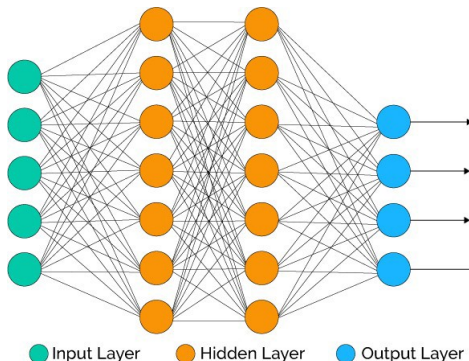
# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

# WORKFLOW PROGETTO

- ▶ studio EEG e crisi epilettiche
- ▶ studio teorico CNN
- ▶ studio tool di sviluppo
- ▶ creazione di un dataset di allenamento bilanciato
- ▶ creazione e allenamento rete per le predizioni
- ▶ costruzione di architettura web backend e frontend

# RETI NEURALI: OVERVIEW



- ▶ Modelli computazionali basati su neuroni
- ▶ Ispirate al funzionamento biologico del cervello
- ▶ Applicate nella risoluzione di problemi ingegneristici, informatici, di simulazione...

# RETI NEURALI: OVERVIEW

## Funzionamento neurone:

- ▶ Una o più connessioni in ingresso e uscita
- ▶ Somma dei segnali in ingresso moltiplicati per i pesi delle connessioni
- ▶ Aggiunta di un eventuale *bias*
- ▶ Funzione di attivazione e segnale in uscita

## Tipologie di rete:

- ▶ Feedforward: layer di neuroni con connessioni in una sola direzione
- ▶ Ricorsive: connessioni con neuroni dello stesso livello o all'indietro



# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test



# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*

Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

# RETI NEURALI: TRAINING

## Tipologie di allenamento:

- ▶ Supervised Learning
- ▶ Unsupervised Learning
- ▶ Semi-supervised Learning
- ▶ Apprendimento per rinforzo

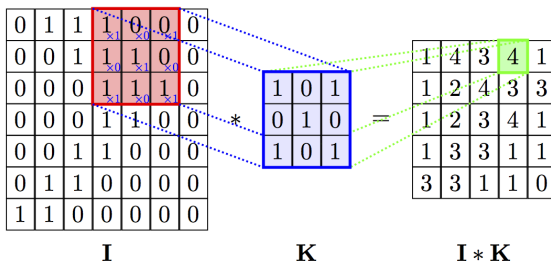
## Passi allenamento supervisionato:

- ▶ Inizializzazione
- ▶ Feed-forward
- ▶ Loss function → Calcolo errore
- ▶ Differentiation → Diminuzione loss
- ▶ Backpropagation e weights update

Ripetuti per ogni  
*epoca*  
Validation e Test

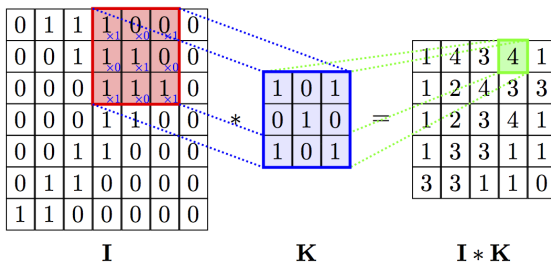
# RETI NEURALI CONVOLUZIONALI

- ▶ Adatte ad input di grandi dimensioni
- ▶ Riconoscimento di *pattern* nell'architettura
- ▶ Applicazione di *kernel* per individuazione di features nei dati



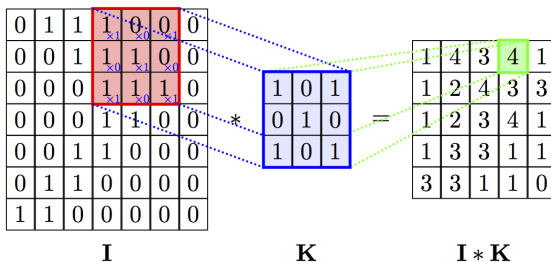
# RETI NEURALI CONVOLUZIONALI

- ▶ Adatte ad input di grandi dimensioni
- ▶ Riconoscimento di *pattern* nell'architettura
- ▶ Applicazione di *kernel* per individuazione di features nei dati



# RETI NEURALI CONVOLUZIONALI

- ▶ Adatte ad input di grandi dimensioni
- ▶ Riconoscimento di *pattern* nell'architettura
- ▶ Applicazione di *kernel* per individuazione di features nei dati



# RETI NEURALI CONVOLUZIONALI

Parametri layer convoluzionale:

- ▶ Input (I)
- ▶ Profondità (K)
- ▶ Stride (S)
- ▶ Zero-Padding (P)

Dimensione del volume di output:

$$O = \frac{(I - K - 2P)}{S} + 1 \quad (1)$$



# RETI NEURALI CONVOLUZIONALI

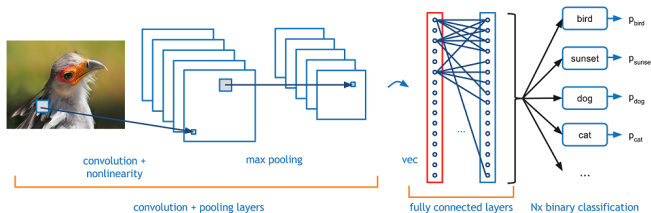
Parametri layer convoluzionale:

- ▶ Input (I)
- ▶ Profondità (K)
- ▶ Stride (S)
- ▶ Zero-Padding (P)

**Dimensione del volume di output:**

$$O = \frac{(I - K - 2P)}{S} + 1 \quad (1)$$

# RETI NEURALI CONVOLUZIONALI



- Convolutional layers
- Pooling Layers (max, average...)
- Fully Connected Layers



# MODELLO PROPOSTO

```
<bound method Module.state_dict of ConvNet(
  (layer1): Sequential(
    (0): Conv1d(23, 10, kernel_size=(10,), stride=(5,), padding=(9,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=15, stride=10, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv1d(10, 5, kernel_size=(5,), stride=(4,), padding=(4,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=10, stride=5, padding=0, dilation=1, ceil_mode=False)
  )
  (drop_out): Dropout(p=0.5)
  (fc1): Linear(in_features=35, out_features=15, bias=True)
  (fc3): Linear(in_features=15, out_features=2, bias=True)
  (soft): Softmax())
>>
```

## ► Sequential:

- Convolution
- ReLU activation
- MaxPool

## ► Dropout

- Fully-connected
- Softmax



# MODELLO PROPOSTO

```
<bound method Module.state_dict of ConvNet(
  (layer1): Sequential(
    (0): Conv1d(23, 10, kernel_size=(10,), stride=(5,), padding=(9,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=15, stride=10, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv1d(10, 5, kernel_size=(5,), stride=(4,), padding=(4,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=10, stride=5, padding=0, dilation=1, ceil_mode=False)
  )
  (drop_out): Dropout(p=0.5)
  (fc1): Linear(in_features=35, out_features=15, bias=True)
  (fc3): Linear(in_features=15, out_features=2, bias=True)
  (soft): Softmax())
>>
```

## ► Sequential:

### ► Convolution

### ► ReLU activation

### ► MaxPool

### ► Dropout

### ► Fully-connected

### ► Softmax

# MODELLO PROPOSTO

```
<bound method Module.state_dict of ConvNet(
  (layer1): Sequential(
    (0): Conv1d(23, 10, kernel_size=(10,), stride=(5,), padding=(9,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=15, stride=10, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv1d(10, 5, kernel_size=(5,), stride=(4,), padding=(4,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=10, stride=5, padding=0, dilation=1, ceil_mode=False)
  )
  (drop_out): Dropout(p=0.5)
  (fc1): Linear(in_features=35, out_features=15, bias=True)
  (fc3): Linear(in_features=15, out_features=2, bias=True)
  (soft): Softmax())
>>
```

## ► Sequential:

- Convolution
- ReLU activation
- MaxPool

## ► Dropout

- Fully-connected
- Softmax

# MODELLO PROPOSTO

```
<bound method Module.state_dict of ConvNet(
  (layer1): Sequential(
    (0): Conv1d(23, 10, kernel_size=(10,), stride=(5,), padding=(9,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=15, stride=10, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv1d(10, 5, kernel_size=(5,), stride=(4,), padding=(4,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=10, stride=5, padding=0, dilation=1, ceil_mode=False)
  )
  (drop_out): Dropout(p=0.5)
  (fc1): Linear(in_features=35, out_features=15, bias=True)
  (fc3): Linear(in_features=15, out_features=2, bias=True)
  (soft): Softmax())
>>
```

## ► Sequential:

- Convolution
- ReLU activation
- MaxPool

## ► Dropout

## ► Fully-connected

## ► Softmax



# MODELLO PROPOSTO

```
<bound method Module.state_dict of ConvNet(
  (layer1): Sequential(
    (0): Conv1d(23, 10, kernel_size=(10,), stride=(5,), padding=(9,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=15, stride=10, padding=0, dilation=1, ceil_mode=False)
  )
  (layer2): Sequential(
    (0): Conv1d(10, 5, kernel_size=(5,), stride=(4,), padding=(4,))
    (1): ReLU()
    (2): MaxPool1d(kernel_size=10, stride=5, padding=0, dilation=1, ceil_mode=False)
  )
  (drop_out): Dropout(p=0.5)
  (fc1): Linear(in_features=35, out_features=15, bias=True)
  (fc3): Linear(in_features=15, out_features=2, bias=True)
  (soft): Softmax())
>>
```

## ► Sequential:

- Convolution
- ReLU activation
- MaxPool

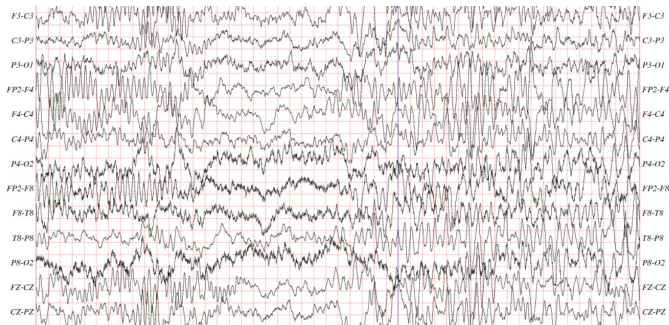
## ► Dropout

- Fully-connected
- Softmax





# DATASET



- ▶ Children's Hospital Boston database (22 pazienti totali)
- ▶ Pazienti 1,2,3,5,7 e 8
- ▶ File con almeno un evento di crisi
- ▶ Frequenza di campionamento 256Hz
- ▶ 23 canali di campionamento

# TRAINING DELLA RETE

## ► Preparazione del dataset:

- Isolamento dei segnali di crisi
- Suddivisione in finestre da 30 secondi, con stride di 1
- Eliminazione segnali pre e post ictali (5 minuti)
- Selezione finestre non di crisi

## ► Training:

- 30 epoche
- Cross Entropy Loss
$$loss(x, class) = -x[class] + \log(\sum_j \exp(x[j]))$$
- Adam Optimizer (Adaptive moment estimation)

## ► Validazione: 80% training set, 20% validation set

## ► Test su file esterni al training set

# BACKEND

- ▶ Python
- ▶ Django Web Framework
- ▶ PyTorch
- ▶ PyEdflib



**django**

**PYTORCH**




# IL SOFTWARE

- ▶ Tool Analisi
- ▶ Tool Predizione
- ▶ Tool Training



# TOOL ANALISI



[HOME](#)
[EEG ANALYSIS](#)
[PREDICT](#)
[TRAIN](#)

## EEG ANALYSIS

Upload your edf file

chb01\_03.edf  
 Upload your file

**UPLOAD**

Select the desired channel or all channels

Select channel ▼

Insert the start second of the slice you want to visualize

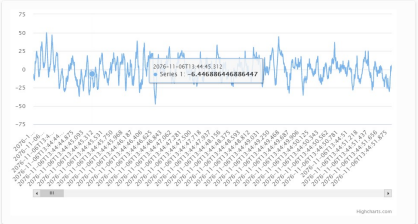
Start Time

Insert the length (in seconds) of the slice you want to visualize

Window Dimensions

➤ **EEG visualization with insert parameters**

Visualize your edf file, you can choose single channel or entire EEG trace. The chart shows the selected slice of the trace according to start and length parameters.



2076-11-06T13:44:45.312  
Series 1: -6.446886446886447

highcharts.com

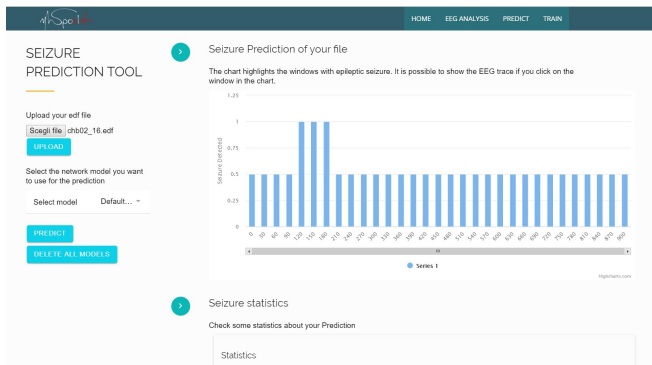
➤ **EEG Statistics**

Visualize the statistics of uploaded file. Statistics are calculated on the whole file.

## Servizi offerti:

- ▶ Visualizzazione EEG singolo canale / intero tracciato
- ▶ Statistiche intero file
- ▶ Visualizzazione grafico distribuzione valori

# TOOL PREDIZIONE



## Servizi offerti:

- ▶ Visualizzazione predizione crisi per ogni finestra
- ▶ Statistiche predizione
- ▶ Grafico EEG finestra selezionata

# TOOL TRAINING

The screenshot shows the 'n1Spool' EEG Analysis Tool Training interface. The top navigation bar includes links for HOME, EEG ANALYSIS, PREDICT, and TRAIN. The main content area is divided into two sections: 'Dataset Creation' and 'Setting your Network'.

**Dataset Creation**

Convert all the uploaded files in a training dataset. You can choose window dimension and the stride for seizure windows. Applying a little stride will overlap windows creating a bigger dataset.

Input fields for window dimension and stride are set to 30 and 20 respectively. A 'CREATE DATASET' button is present. Below the button, it says 'DATASET CREATED'.

**Setting your Network**

**Adding your Convolutional**

Output Convolutional :

Number of kernel to apply (feature detectors)

**Kernel dimension:**

width of each kernel to apply

**Stride :**

Number of signals you want the filter to shift in every movement

**Padding:**

add some zeroes around the signal window

**Pool Kernel dimension:**

## Servizi offerti:

- ▶ Inserimento file di training e creazione dataset
- ▶ Creazione rete personalizzata
- ▶ Training della rete creata

# VANTAGGI

- ▶ visualizzazione tracciati EEG
- ▶ calcolo statistiche su EEG
- ▶ predizione di crisi epilettiche attraverso reti neurali preallenate
- ▶ creazione rete neurale personalizzata e predizione

# SVILUPPI FUTURI

- ▶ Analisi di diversi tipi di file (.csv, .txt...)
- ▶ Deployment su server remoto
- ▶ Maggiore personalizzazione delle reti
- ▶ Ottimizzazione e diversificazione delle reti pre-allenate offerte

# SVILUPPI FUTURI

- ▶ Analisi di diversi tipi di file (.csv, .txt...)
- ▶ Deployment su server remoto
- ▶ Maggiore personalizzazione delle reti
- ▶ Ottimizzazione e diversificazione delle reti pre-allenate offerte

# SVILUPPI FUTURI

- ▶ Analisi di diversi tipi di file (.csv, .txt...)
- ▶ Deployment su server remoto
- ▶ Maggiore personalizzazione delle reti
- ▶ Ottimizzazione e diversificazione delle reti pre-allenate offerte

