Tool CNN per analisi e predizione di crisi epilettiche

Manuel Cretone, Emilio Silvestri
Tutor: prof Marco Piangerelli

Group project

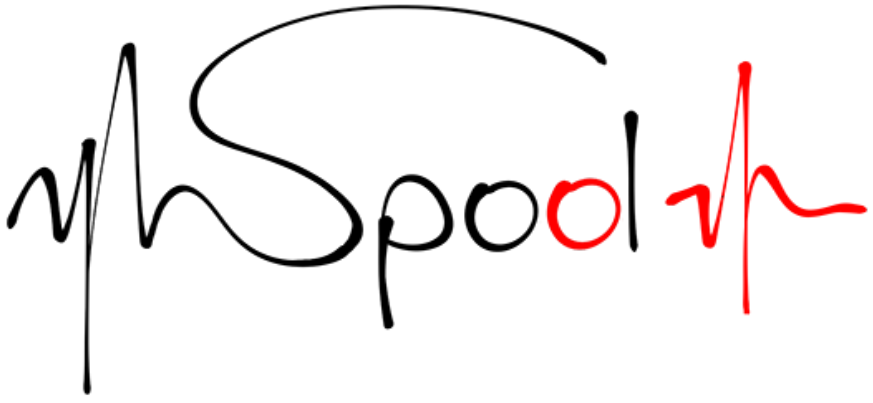
Outline

- Introduzione
- Il software
  - Tool analisi
  - Tool predizione
  - Tool training
- Basi teoriche
  - Epilessia
  - Reti neurali
  - Reti neurali Convoluzionali
- Training
  - Dataset
- Tecnologie
  - Backend
  - 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

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

SVILUPPI FUTURI

- Deployment su server remoto
- Maggiore personalizzazione delle reti
- Ottimizzazione e diversificazione delle reti pre-allenate offerte

Tool analisi

blablabla

Tool predizione

blablabla

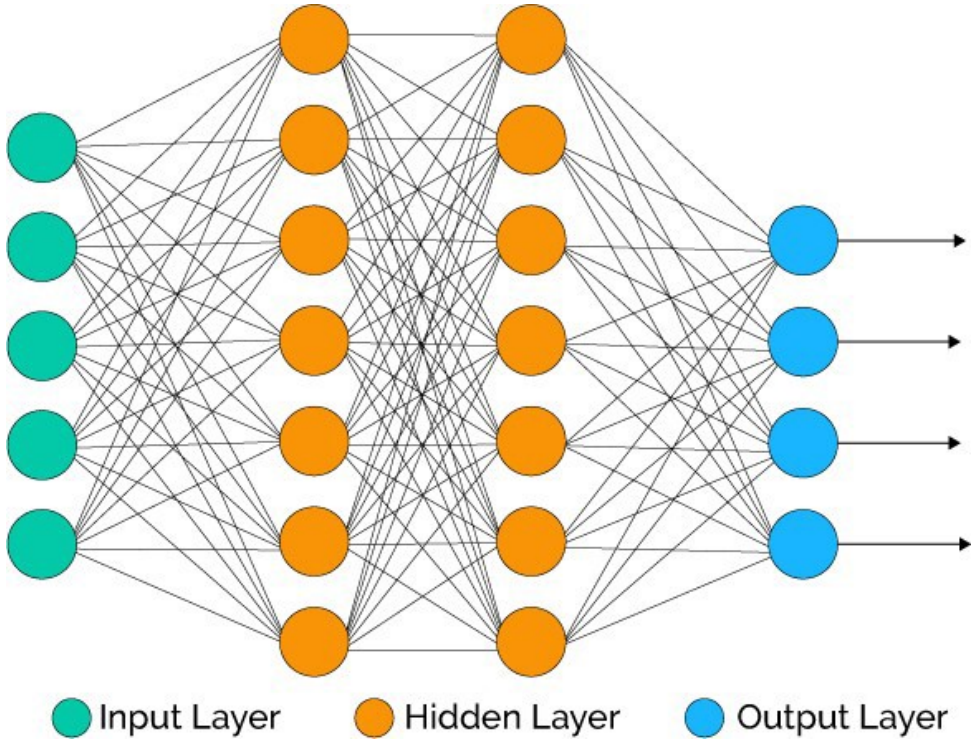
Tool training

blablabla

Epilessia

bla bla

Reti neurali



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

Reti neurali

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

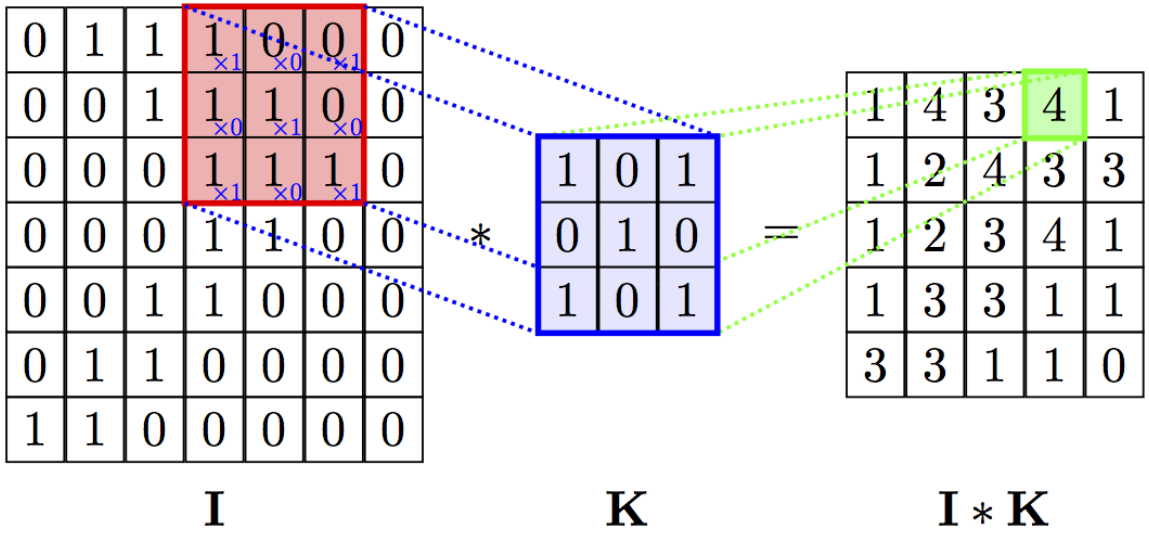
Passi allenamento supervisionato:

- Inizializzazione
- Feed-forward
- Loss function
- Differentiation
- 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

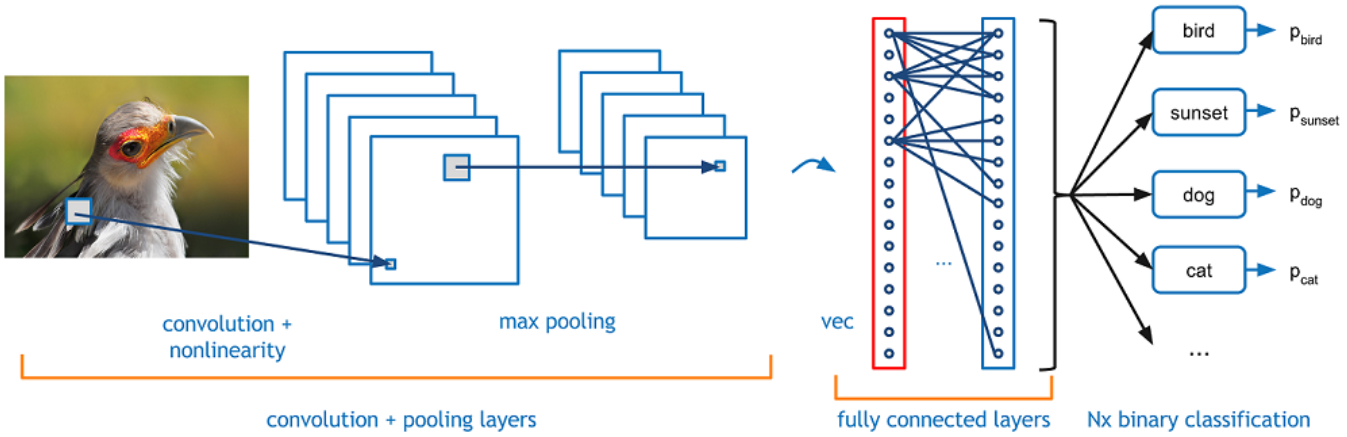
Parametri layer convoluzionale:

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

Dimensione del volume di output:

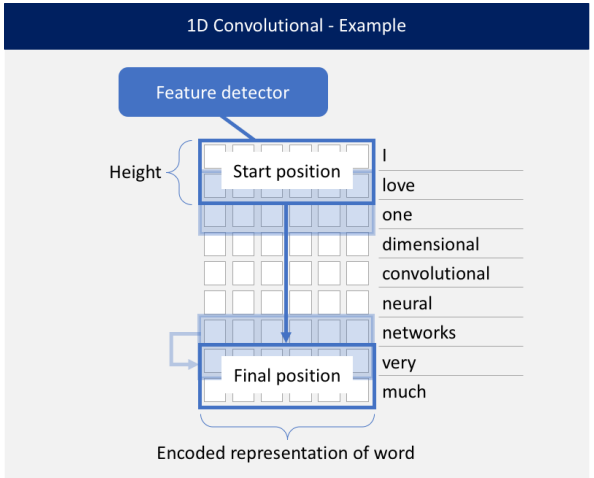
$$O = \frac{(I - K - 2P)}{S} + 1 \tag{1}$$

Reti neurali Convoluzionali

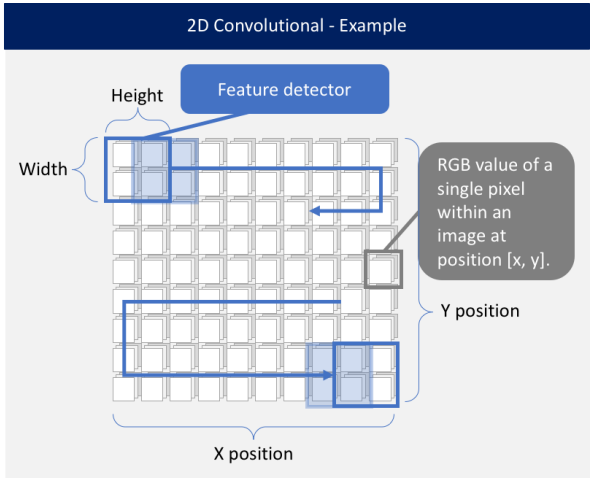


- Convolutional layers
- Pooling Layers (max, average...)
- Fully Connected Layers

Convoluzioni 1D



In this example for natural language processing, a sentence is made up of 9 words. Each word is a vector that represents a word as a low dimensional representation. The feature detector will always cover the whole word. The height determines how many words are considered when training the feature detector. In our example, the height is two. In this example the feature detector will iterate through the data 8 times.



In this example for computer vision, each pixel within the image is represented by its x- and y position as well as three values (RGB). The feature detector has a dimension of 2 x 2 in our example. The feature detector will now slide both horizontally and vertically across the image.

Conv 1D (segnali, NLP...)

- Input bidimensionale *(C, W)*
- Il kernel scorre lungo la dimensione W

Conv 2D (immagini)

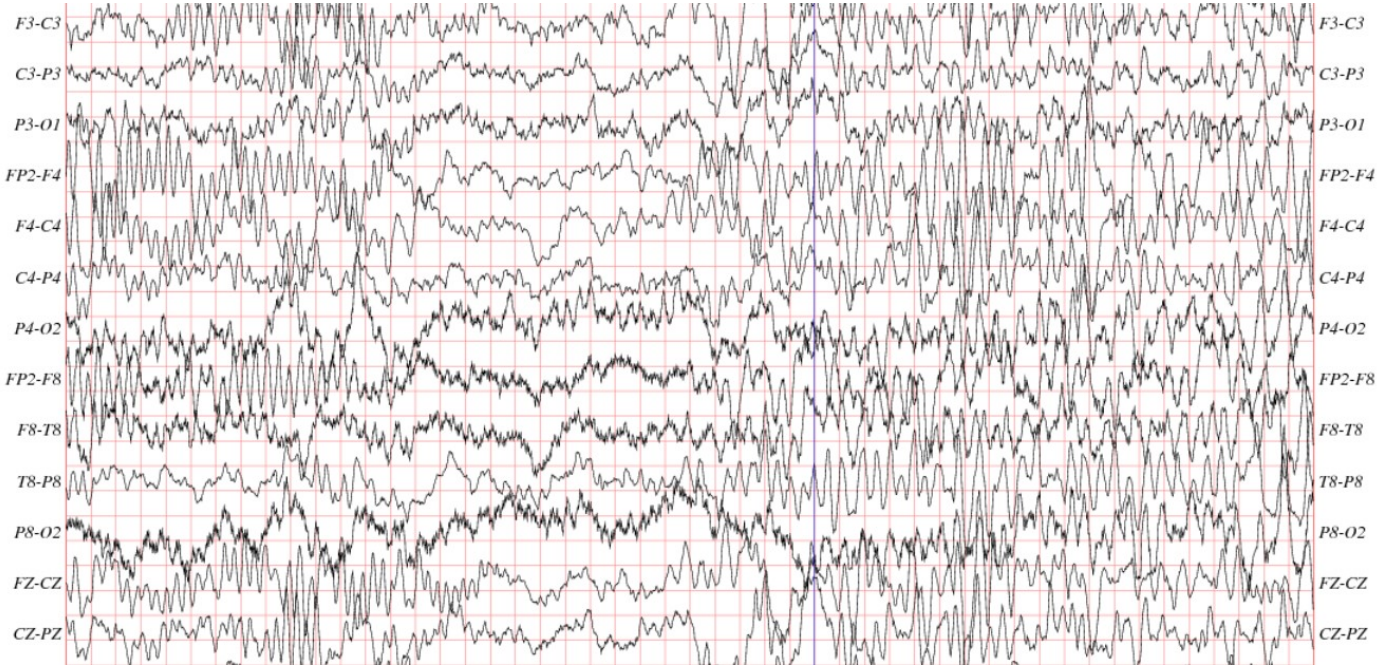
- Input tridimensionale *(C, H, W)*
- Il kernel scorre lungo le dimensioni H e W

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 overlapping di 29
  - Eliminazione segnali pre e post ictali (5 minuti)
  - Selezione finestre non di crisi
- **Validazione**: 80% training set, 20% validation set
- **Test** su file esterni al training set

Backend

django, pytorch, pyedflib

Frontend

angular, ionic , highchart


thank you
danke
gracias
merci
obrigado
dziękuję
teşekkür ederim
go raibh maith agat
mochchakkeram
sukriya
kop khun krap
terima kasih
grazie
arigatô
dakujem
tapadh leat
bedankt
hvala
mauruuru
ngiyabonga
спасибо
мерси
謝謝
감사합니다
ありがとう
dank je
takk
tack
taiku
sagolun
Баярлалаа
рахмат
ευχαριστώ