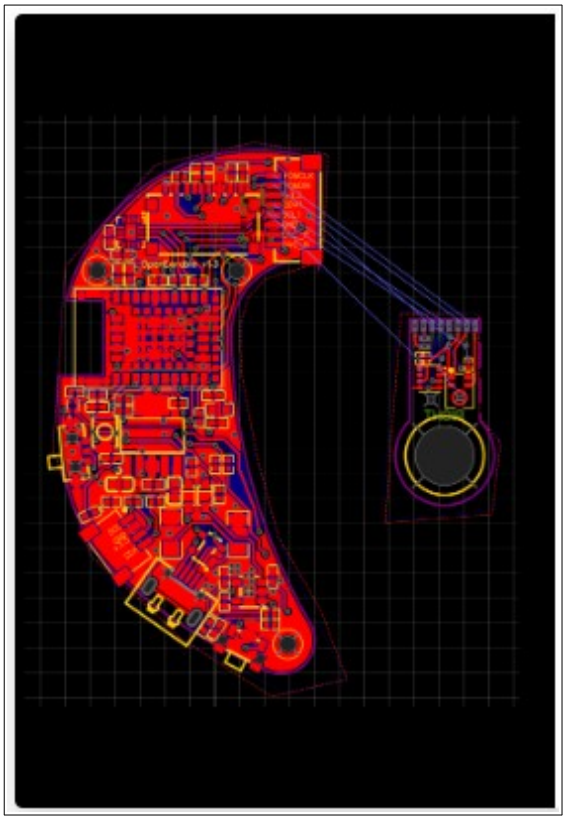
Progetto Tirocinio: OpenEarable

INTRO

OpenEarable è una nuova piattaforma open source basata su Arduino per applicazioni di rilevamento basate sull'orecchio. È dotata di una serie di sensori e attuatori: un'unità di misura inerziale a 9 assi, un sensore di pressione e temperatura del canale uditivo, un microfono a ultrasuoni rivolto verso l'interno, nonché un altoparlante, un pulsante e un LED RGB controllabile. OpenEarable offre una piattaforma di rilevamento aperta e di uso generale per la ricerca e lo sviluppo di dispositivi auricolari.



COMPONENTI

Diversi **SENSORI** con possibilità di rilevare oltre 30 fenomeni:

- **Microfono con capacità a ultrasuoni** (Modello Knowless SPH0641LU4H-1). Questo microfono è orientato verso il condotto uditivo, ed è in grado di catturare segnali audio a ultrasuoni. Sono supportati per la registrazione su scheda microSD interna fino a 62,5 kHz. Potenziali applicazioni sono: autenticazione forma del canale uditivo, monitoraggio cardiaco e tracciamento del movimento acustico.
- **Unità di misura inerziale a 9 assi** (Modello BMX160). Composta da accelerometro, giroscopio e magnetometro (IMU). Consente lo streaming dati IMU in tempo reale fino a 50 Hz tramite BLE. Possibili applicazioni sono: monitoraggio frequenza respiratoria, monitoraggio dell'allenamento e fitness e supporto per rianimazione cardiopolmonare.
- **Sensore di pressione del canale uditivo** (Motore Bosch BMP280). Esso è rivolto verso l'interno e consente di misurare i cambiamenti di pressione nel canale uditivo sigillato. Attraverso il cuscinetto auricolare in schiuma è possibile ottenere misurazioni precise fino a 30 Hz in streaming tramite BLE. Possibili applicazioni sono: interazioni EarRumble, rilevamento del mangiare e rilevamento gesti della lingua.

Diversi **ATTUATORI** usati per riprodurre audio e fornire feedback visivo:

- **Oratore:** è un altoparlante interno che supporta la riproduzione di file audio dalla scheda microSD interna. È possibile generare toni a frequenza costante direttamente sul dispositivo o attivare otto diversi jingle tramite BLE.
- **Led RGB:** è posizionato nell'involucro principale dietro al dispositivo. Il colore può essere configurato liberamente tramite BLE.
- **Pulsante e interruttore:** utilizzato per attivare eventi notificati tramite BLE. Interruttore utilizzato per accendere/spegnere il dispositivo.

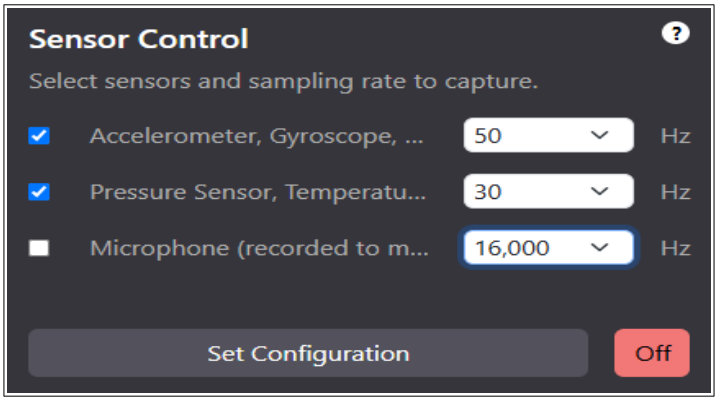
PROVE DI FUNZIONAMENTO

Si può verificare il funzionamento del dispositivo attraverso la Dashboard disponibile nel sito di OpenEarable. In particolare essa è divisa in varie parti:

- Sezione di connessione del dispositivo.

- Sezione di controllo dei sensori:

  - Accelerometro, Giroscopio e Magnetometro: 0 – 50 Hz
  - Pressione e Temperatura: 0 – 30 Hz
  - Microfono: 16,0 – 62,5 Hz



- Sezione di controllo del LED.

Il risultato che si verifica è il seguente:

CARATTERISTICHE DI DISPOSITIVO

- Indirizzo IP BLE Device: 54:EE:39:71:B5:21, OpenEarable-B521
- Caratteristiche UUID:

DEVICE INFO SERVICE	45622510-6468-465a-b141-0b9b0f96b468
DEVICE IDENDTIFIER CHARACTERISTIC	45622511-6468-465a-b141-0b9b0f96b468
FIRMWARE REVISION CHARACTERISTIC	45622512-6468-465a-b141-0b9b0f96b468
HARDWARE GENERATION CHARACTERISTIC	45622513-6468-465a-b141-0b9b0f96b468
BATTERY SERVICE	0000180f-0000-1000-8000-00805f9b34fb
BATTERY LEVEL CHARACTERISTIC	00002a19-0000-1000-8000-00805f9b34fb
BATTERY STATE CHARACTERISTIC	00002a1a-0000-1000-8000-00805f9b34fb
PARSE INFO SERVICE	caa25cb7-7e1b-44f2-adc9-e8c06c9ced43
SCHEME CHARACTERISTIC	caa25cb8-7e1b-44f2-adc9-e8c06c9ced43
SENSOR SERVICE	34c2e3bb-34aa-11eb-adc1-0242ac120002
SENSOR CONFIGURATION CHARACTERISTIC	34c2e3bd-34aa-11eb-adc1-0242ac120002
SENSOR DATA CHARACTERISTIC	34c2e3bc-34aa-11eb-adc1-0242ac120002

BUTTON SERVICE	29c10bdc-4773-11ee-be56-0242ac120002
BUTTON STATE CHARACTERISTIC	29c10f38-4773-11ee-be56-0242ac120002
LED SERVICE	81040a2e-4819-11ee-be56-0242ac120002
LED STATE CHARACTERISTIC	81040e7a-4819-11ee-be56-0242ac120002
AUDIO SERVICE	5669146e-476d-11ee-be56- 0242ac120002
AUDIO SOURCE CHARACTERISTIC	566916a8-476d-11ee-be56- 0242ac120002
AUDIO STATE CHARACTERISTIC	566916a9-476d-11ee-be56- 0242ac120002

- Struttura IMU con Sensor ID = 0 (accelerometro, giroscopio, magnetometro):

Byte 0 - 3	Byte 4 - 7	Byte 8 - 11	Byte 12 - 15	Byte 16 - 19	Byte 20 - 23	Byte 24 - 27	Byte 28 - 31	Byte 32 - 35
ACC X	ACC Y	ACC Z	GYRO X	GYRO Y	GYRO Z	MAG X	MAG Y	MAG Z
float	float	float	float	float	float	float	float	float

- Struttura BME con Sensor ID = 1 (pressione e temperatura):

Byte 0 - 3	Byte 4 - 7
PRESSURE	TEMPERATURE
float	float

- Calcolo RPY

$$roll = \arctan\left(\frac{ACC_Y}{ACC_z}\right)$$

$$pitch = \arctan\left(\frac{-ACC_X}{\sqrt{ACC_Y{}^2 + ACC_Z{}^2}}\right)$$

$$yaw = yaw_{prev} + GYRO_Z * dt$$

OBIETTIVI DEL PROGETTO

Il progetto consiste nello studio del dispositivo OpenEarable che comprende:

- studio del dispositivo con la DashBoard
- scrittura programma per estrapolare i dati inerziali, RPY, temperatura e pressione
- verifica andamento dei dati attraverso grafici
- registrazione delle attività con costruzione di una rete neurale
- estrazione del modello migliore di training
- riconoscimento delle attività in tempo reale

INTRO PROGETTO

Studiando il funzionamento del dispositivo attraverso la DashBoard e collezionando le informazioni ho costruito un programma per estrarre i dati di accelerometro, giroscopio, magnetometro, pressione e temperatura dall'OpenEarable. Questi dati, soprattutto quelli inerziali, sono stati collezionati in file csv che descrivono le attività che ho registrato indossando nell'orecchio il dispositivo. Queste attività sono: **TESTA FERMA, ROTAZIONE, NODDING** e **INCLINAZIONE**. La registrazione di questi dati è durata circa 5 minuti per ciascuna attività. Inoltre, con un opportuno programma, ho plottato i dati visualizzando l'andamento di essi e, attraverso un modello 3d del dispositivo, ho visualizzato con un altro grafico i dati di RPY attraverso le rotazioni in tempo reale del dispositivo. Ho costruito una rete neurale per addestrare l'IA a riconoscere le attività in maniera autonoma e i risultati ottenuti sono stati alla base della costruzione del miglior modello attraverso una matrice di confusione. Questo modello, poi, è stato convertito ed usato per predire in tempo reale le attività precedentemente registrate.

DESCRIZIONE DEL CODICE

Per facilitare la comprensione del codice ho deciso di spiegarlo e dividerlo in fasi:

FASE 1: Riconoscimento del dispositivo e connessione.

Le funzioni che definiscono questa fase sono:

- **MAIN:** Permette di raccogliere la lista dei dispositivi BLE in zona e di connettersi al dispositivo OpenEarable attraverso l'indirizzo IP. Inoltre attende l'input dell'utente per avviare la ricezione dei dati dai sensori del dispositivo.
- **SCAN:** Esegue una scansione dei dispositivi BLE per 10 secondi e ritorna la lista dei dispositivi trovati.
- **FIND:** Filtra solo i dispositivi BLE i cui indirizzi MAC corrispondono a quelli presenti nella lista addresses. Restituisce la lista dei dispositivi desiderati.
- **CONNECT:** Definisce la connessione del dispositivo mandando dei messaggi a terminale sulla riuscita della connessione a dispositivo. Legge il livello della batteria e successivamente definisce la connessione attraverso l'accensione del led con il colore verde, salvato con lo stato "connected".
- **DISCONNECT:** Termina la connessione con il dispositivo.

FASE 2: Raccolta dati proveniente dai sensori.

Questa fase si attiva quando, da terminale, compare "Enter recording name". A questo punto attraverso il tasto "Invio" iniziamo la raccolta dei dati:

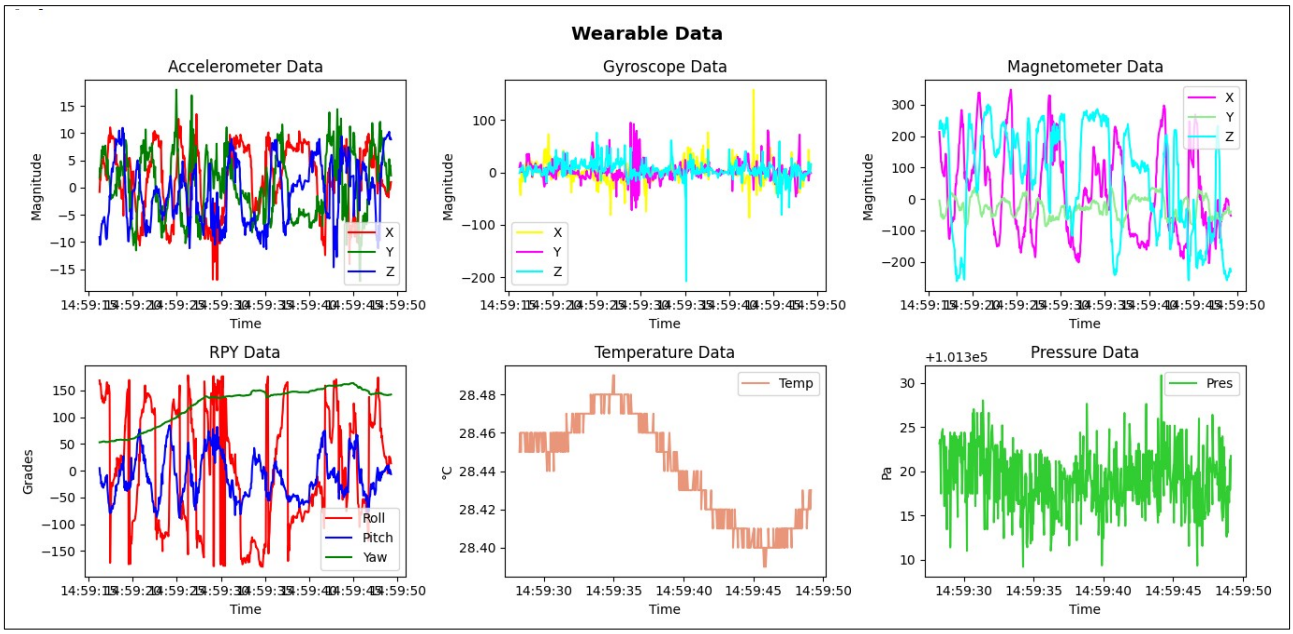
- **RECEIVE INERTIAL DATA:** Permette di raccogliere i dati comunicando con il sensore IMU e BME dell'OpenEarable. Cambia stato in "recording", cambiando il led da verde a rosso. Permette la creazione dei file csv separati: uno contenente i dati di imu e rpy e l'altro contenente i dati di pressione e temperatura. Questo processo di ricezione dei dati dura fino a che non si stoppa la simulazione.
- **IMU DATA CALLBACK:** Ogni volta che riceve i dati imu, attraverso questa funzione, questi vengono parsati da binario, calcola RPY attraverso i dati inerziali e li scrive su csv.
- **BME DATA CALLBACK:** Estrae i dati ambientali di pressione e temperatura.

FASE 3: Analisi grafica dei dati di IMU e BME.

Le funzioni che permettono questo sono:

- **ANIMATE:** Funzione necessaria per aggiornare i dati e visualizzarli in tempo reale durante la simulazione, mantenendo solo gli ultimi 600 campioni dei file csv di IMU e BME.
- **LIVE PLOTTING:** Genera un aggiornamento dinamico ogni 50 millisecondi e visualizza la finestra con i grafici.

Queste funzioni sono chiamate da un main e questo funziona separato rispetto al programma di collezione dei dati. È utile questa fase per verificare l'andamento dei dati nelle varie attività che si vogliono registrare e vedere le differenze tra le stesse. Un esempio applicativo che si può vedere è il seguente:



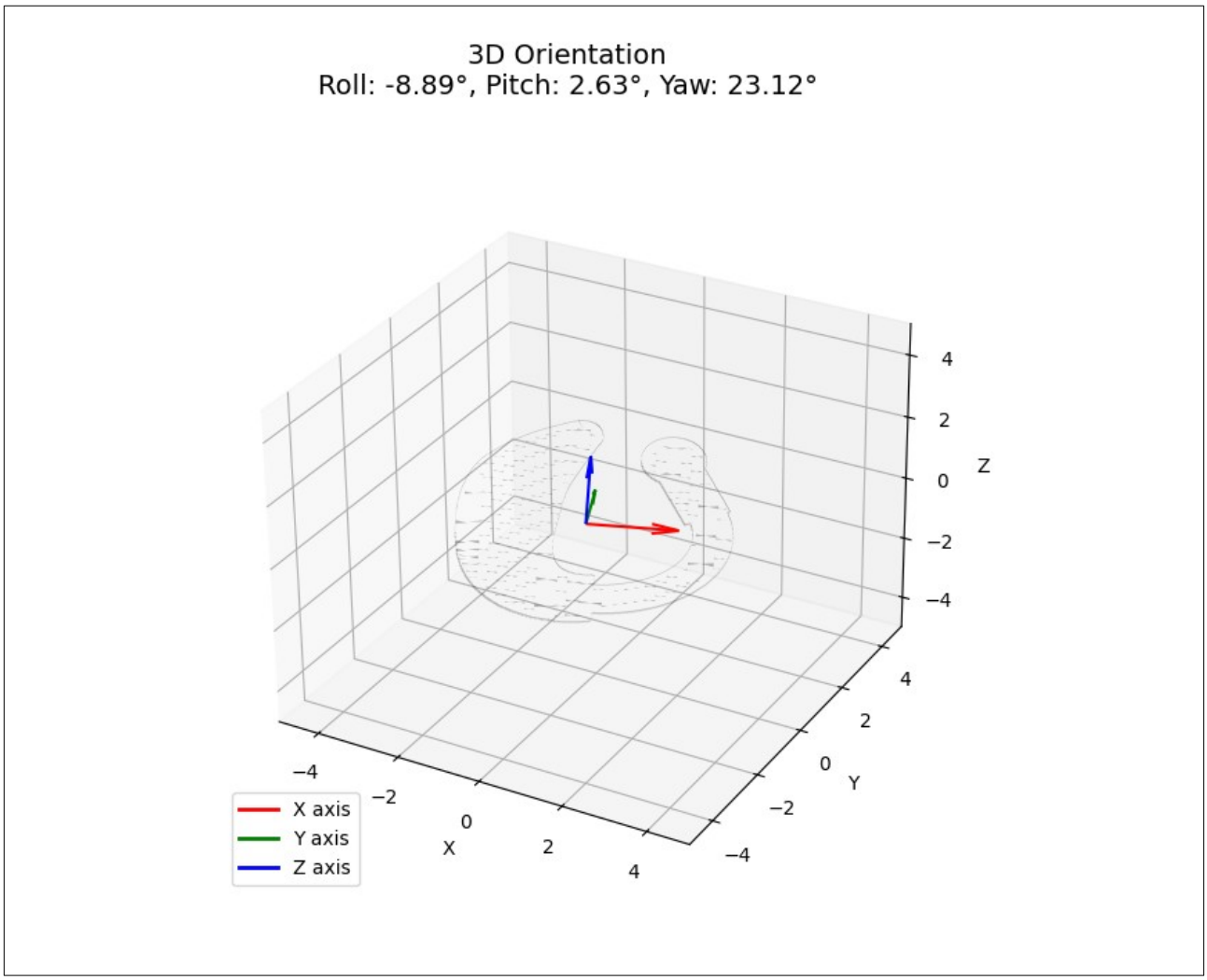
FASE 4: Analisi grafica del modello 3d dell'OpenEarable.

È formata dalle seguenti funzioni:

- **LOAD OBJ:** Legge il file obj del modello 3d dell'OpenEarable e restituisce i vertici e le facce dell'oggetto.
- **ROTATE MODEL:** Applica le rotazioni 3d al modello attraverso le combinazioni delle rotazioni lungo le coordinate X, Y, Z, scalando e ruotando i vertici.

- **ANIMATE:** Fa partire la rotazione del modello a partire dall’ultima riga del file csv. Converte gli angoli in radianti e ruota i vertici del modello 3d in base all’orientamento. Disegna il modello ruotato e ruota i vettori X, Y, Z del modello in base alla matrice di rotazione R, mostrando come il modello si orienta nello spazio. Inoltre mostra i valori di RPY e il loro aggiornamento in tempo reale nel corso della rotazione del modello.

Queste funzioni sono chiamate da un main e questa fase è utile per verificare l’andamento del modello 3d secondo i dati di RPY. Un esempio applicativo che si può vedere è il seguente:



FASE 5: Training del programma e creazione matrici di confusione.

Questa fase comprende due classi:

- **CSV DATA MODULE:** è un modulo utile per lavorare con dati temporali provenienti dall'OpenEarable con la possibilità di eseguire una valutazione tramite k-fold cross-validation. Nel dettaglio, questo modulo comprende queste funzioni:

  - **CREATE DATASET:** legge dati da file CSV contenenti i dati temporali di accelerometro e giroscopio e applica la codifica one-hot;

  - **WINDOWING:** Segmenta il dataframe in finestre temporali;

  - **SETUP:** Esegue la cross-validation, suddividendo i dati in k-fold;

  - **LABELS ENCODING:** Codifica le etichette come one-hot;

  - **DATALOADER:** Fornisce i dati per addestramento e validazione;

- **CNN:** è un modello di rete neurale. Gestisce l'addestramento, la validazione e il test, calcolando la perdita e le metriche come la precisione e il f1-score. Inoltre, include il salvataggio dei modelli migliori tramite checkpoint e la generazione di matrici di confusione alla fine del test.

Queste classi sono richiamate opportunamente da una funzione main che gestisce il processo di k-fold cross-validation per addestrare e testare modelli CNN su dati letti da file CSV. Per ogni fold si ha:7

- viene creato un modello CNN con i dati del fold attuale;

- viene impostato un callback di checkpoint per salvare il miglior modello in base alla perdita di validazione;

- al termine dell'addestramento, il modello viene testato sui dati di validazione.

I risultati che si ottengono da questo programma sono la creazione di matrici di confusione aventi tre differenti risultati sul riconoscimento delle attività.

FASE 6: Conversione modello ckpt in onnx e riconoscimento delle attività.

La conversione viene fatta attraverso due funzioni principali:

- **LOAD MODEL:** Funzione che carica il modello precedentemente addestrato dal file checkpoint, che contiene i pesi e le configurazioni del modello.
- **TORCH TO ONNX:** Converte il modello PyTorch in formato ONNX.

L'esecuzione del programma viene effettuata attraverso la funzione main().

Il riconoscimento delle attività viene fatto sulla base del modello ONNX, attraverso la fase di connessione e raccolta dei dati del dispositivo.

LIBRERIE UTILIZZATE

Le librerie installate per l'implementazione del codice sono le seguenti:

- **ASYNCIO:** usata per la gestione asincrona delle operazioni come la comunicazione con dispositivi BLE.
- **NUMPY:** Usata per la manipolazione di array e operazioni numeriche.
- **BLEAK:** Gestisce la comunicazione con dispositivi BLE.
- **PANDAS:** Usata per la manipolazione dei dati in Python. Utilizzata per caricare, pulire, esplorare e trasformare i dati in formato CSV, Excel e altro.
- **SCIKIT-LEARN:** Utilizzata per l'apprendimento automatico. Include funzioni di pre-processing dei dati, come normalizzazione, encoding delle etichette e divisione di dati in set di allenamento e test.
- **TORCH:** Utilizzata per facilitare la costruzione, l'addestramento e l'inferenza di modelli di deep learning.
- **LIGHTNING:** Utilizzata per la gestione di training, validazione e test in modo più organizzato e più facilmente scalabile.
- **SEABORN e MATPLOTLIB:** librerie per la visualizzazione dei dati. Permette di creare grafici 2D o 3D e utile per la visualizzazione di dati statistici
- **ONNX:** Mi permette di convertire i modelli e lavorare con modelli pre-allenati.
- **ONNX RUNTIME:** motore di esecuzione che esegue modelli ONNX in modo ottimizzato e ad alte prestazioni.

RISULTATI OTTENUTI E CONSIDERAZIONI

Ho addestrato il programma su quattro attività che sono ben distinte poiché una è statica (testa ferma) e tre sono dinamiche (nodding, rotazione, inclinazione). I risultati ottenuti dalle matrici di confusione sembrano andare bene poiché le attività sono abbastanza riconosciute dal programma. La predizione delle attività, durante la raccolta dei dati, invece, presenta qualche problema nel riconoscimento delle attività. Ad esempio mentre mentre si fa nodding e ci si ferma rileva che sto inclinando la testa invece di rilevare la testa ferma. Quindi il programma è migliorabile su diversi aspetti:

- Costruzione di una rete neurale più precisa e affidabile
- Predizione delle attività, dove alcune volte possono insorgere delle “allucinazioni”