UNIVERSITÀ DEGLI STUDI
DI MILANO

Advanced Intelligent Systems, 2024

Affective Computing and Stress Recognition

Understanding and Modeling Stress in VR Environments and User Engagement

Dr.ssa *Susanna **Brambilla*** *susanna.brambilla@unimi.it*
Dr. *Marco Ligabie* *marco.ligabue@unimi.it*
Prof. *N. A. Borghese* *alberto.borghese@unimi.it*

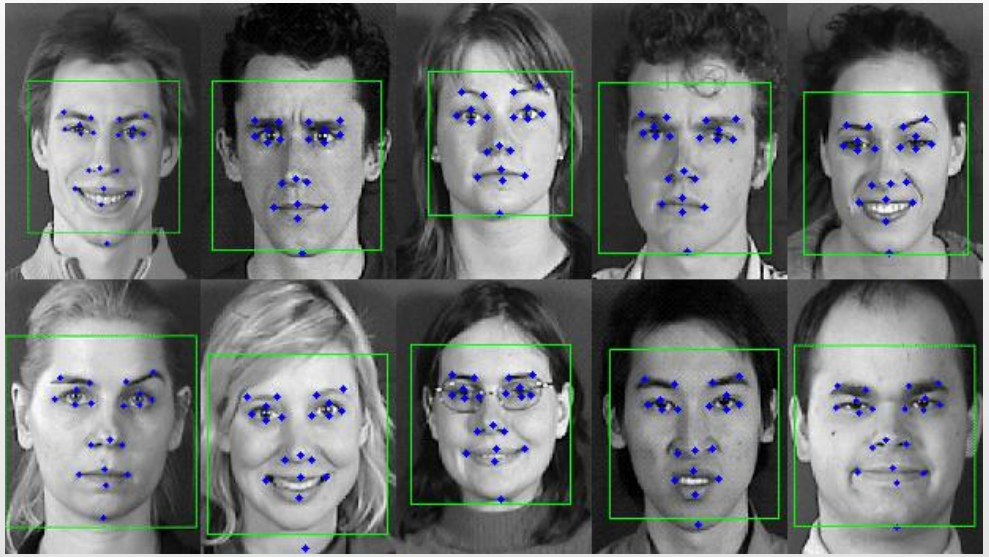
Affective Computing (Picard R.,1995)

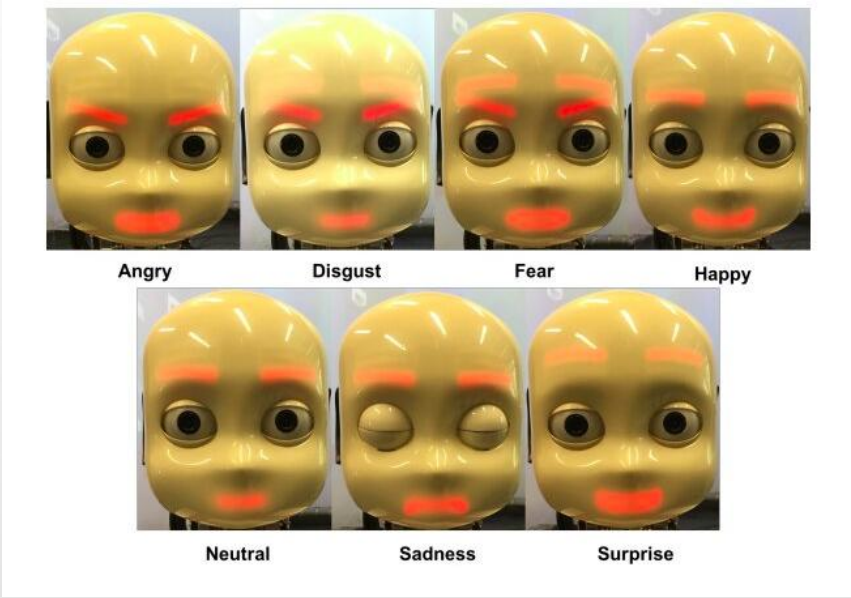
Affective Computing (AC) aims at developing systems which can **automatically recognize emotions** to give **adequate responses**

AC can improve human-computer interactions

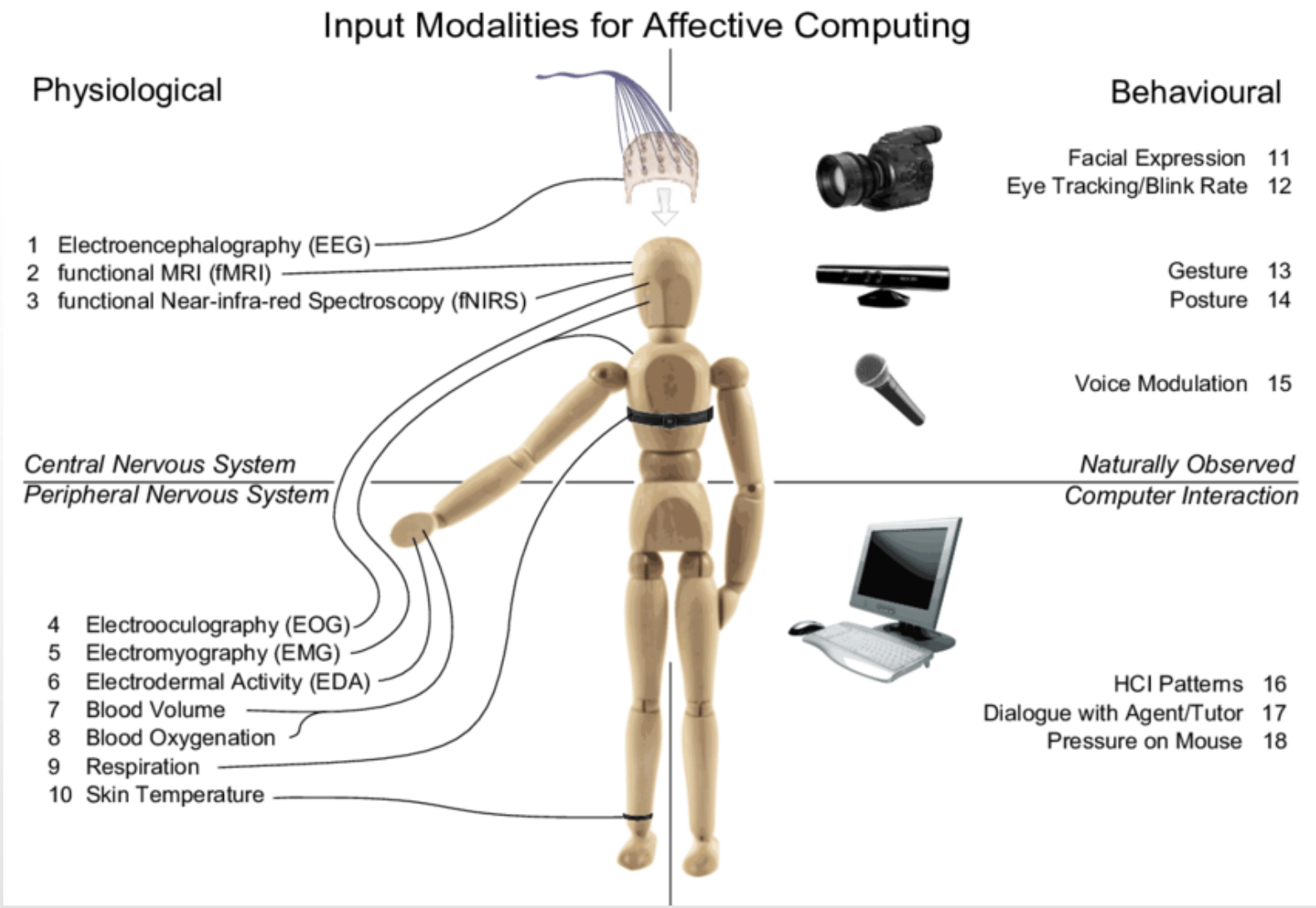
Computers:

- **recognize emotions**
- **expressing emotions**





AC techniques



AC applications

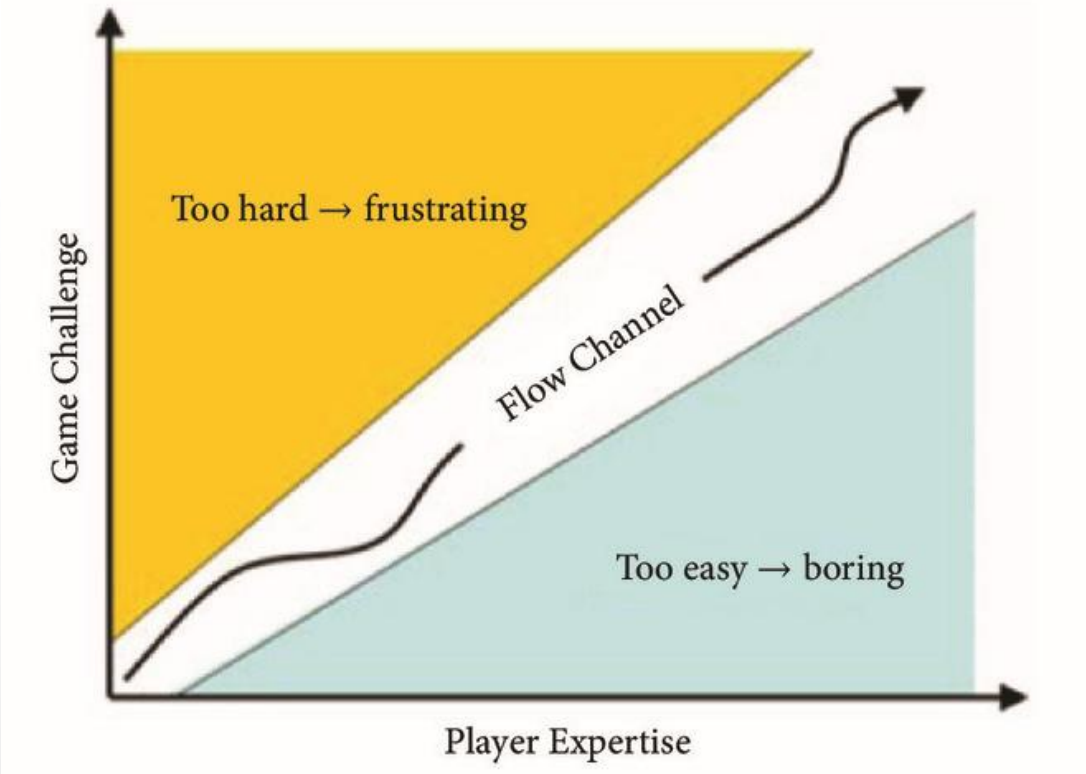
The Flow Theory

Definition:

- mental state of **deep focus and immersion** where individuals feel **fully absorbed** in an activity
- **time seems** to **fade away**, and the person becomes **entirely engaged** in the present moment

Challenge-Skill Balance: the activity must match the individual's skill level:

- **Too easy → Leads to boredom**
- **Too hard → Causes frustration or anxiety**



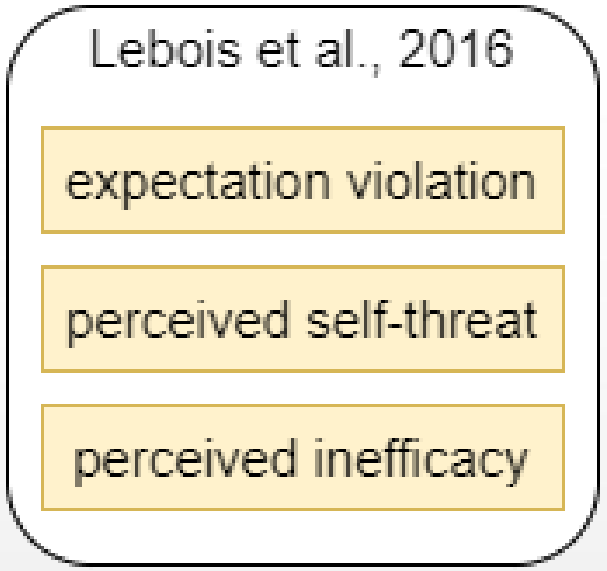
What is stress?

Stress is the body's response to external or internal demands that disrupt its equilibrium

A natural **physiological reaction to perceived challenges or threats,** both real and imagined

The Brain's Role:

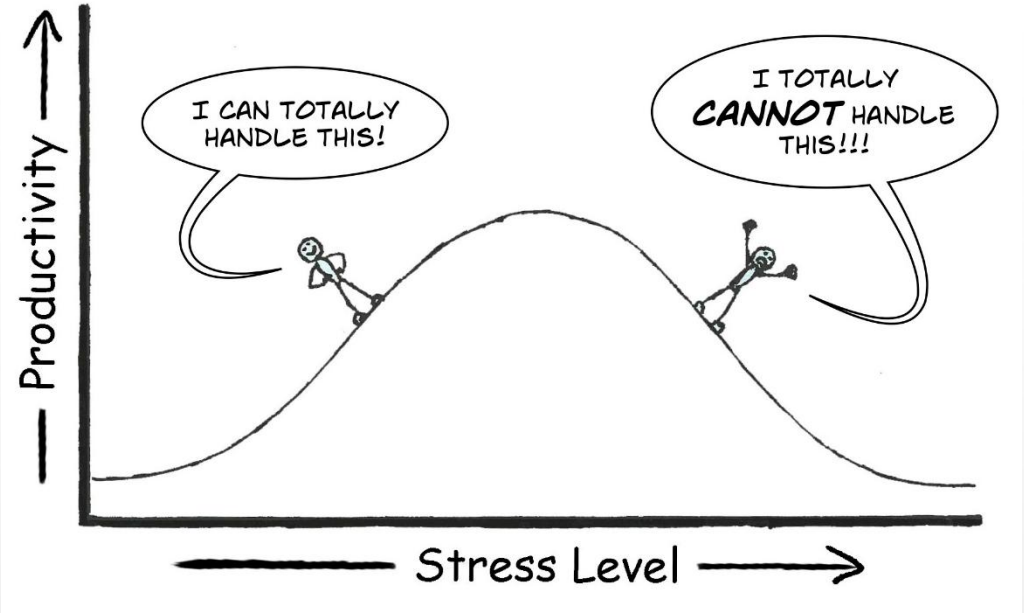
- The brain constantly predicts what's needed to **maintain balance** in the body
- Stress occurs when there's a **gap between expectations and reality**, or when we feel **unprepared to cope**



Positive vs. Negative stress

Eustress (Positive Stress): A type of stress that **motivates us** and helps us **perform better**
Example: Preparing for an important presentation or challenge

Distress (Negative Stress): Stress that **overwhelms us, leading to anxiety, fear,** or a feeling of **being out of control**
Example: A constant heavy workload, financial pressures, or chronic illness



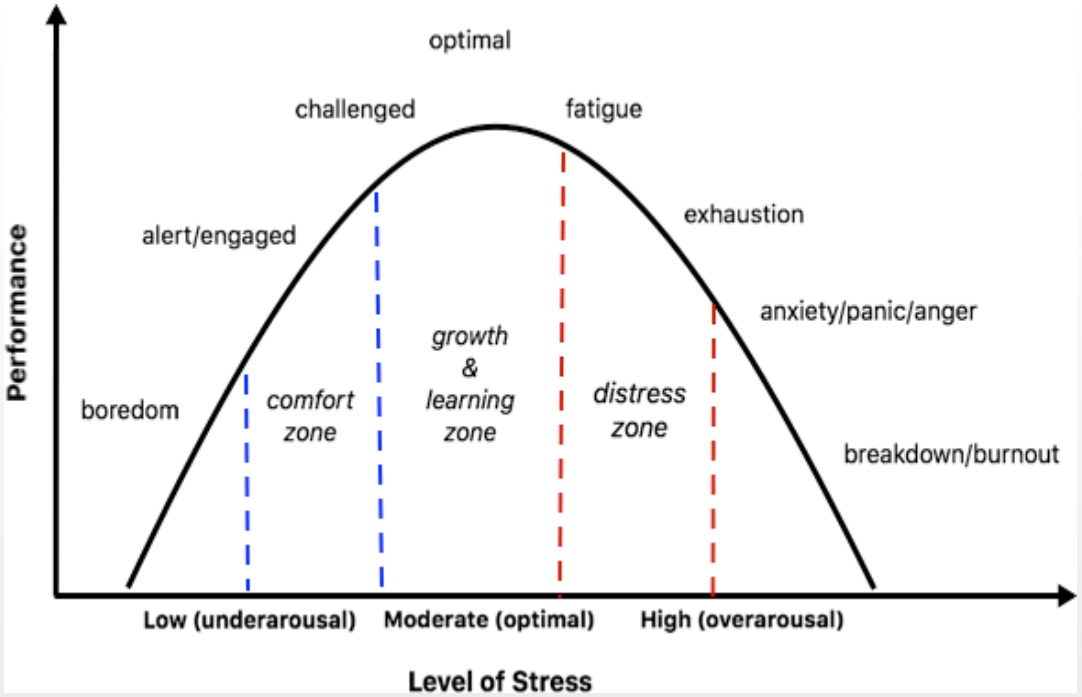
Why stress?

Stress can play a pivotal role in **maintaining motivation and focus**, acting as a **dynamic balance** between challenge and enjoyment

Flow occurs when stress is optimal—not too high or too low:

- It keeps individuals **alert and engaged**, driving **effort and performance**
- Properly calibrated stress can enhance **experiences**, fostering **learning, creativity, and productivity**

Yerkes-Dodson Law (1908) →



Responses to Stress

Physiological responses	Behavioral responses
Increased heart rate: Prepares the body for action by improving blood circulation.	**Avoidance or withdrawal:** Escaping perceived stressors or social interactions.
Faster breathing: Enhances oxygen intake to meet increased energy demands.	**Hypervigilance:** Scanning the environment for potential threats.
Muscle tension: Prepares for physical action but may cause stiffness.	**Erratic actions**: Nervous habits like pacing or fidgeting.
Release of stress hormones (e.g., cortisol, adrenaline): Increases alertness and energy.	**Changes in communication:** Speaking faster or less coherently under pressure.
Dilated pupils: Improves vision to detect potential threats.	**Task performance changes**: Improved for some tasks, degraded for others.

These reactions prepare the body to **face or flee** from the stressor (fight or flight response)

AC and Virtual Reality

- High degree of **immersion**
- Elicit **stronger emotional reactions**
- Realistic experience in a **controlled and safe** setting
- Gather **different types of data**

Applications of Stress Measurement in VR

Training and Education:

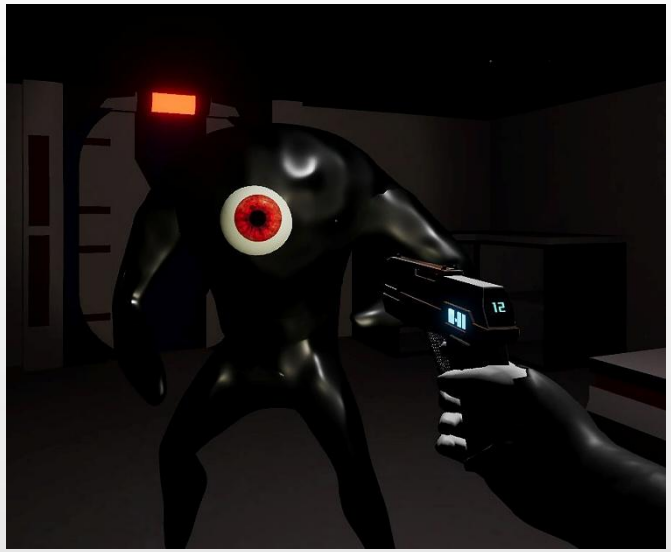
- **Simulations:** stress is induced to create high-pressure environments (e.g., training pilots, medical professionals)
- **Adaptive training:** Using stress data to adjust difficulty levels in real-time to maintain the optimal learning experience

Therapy:

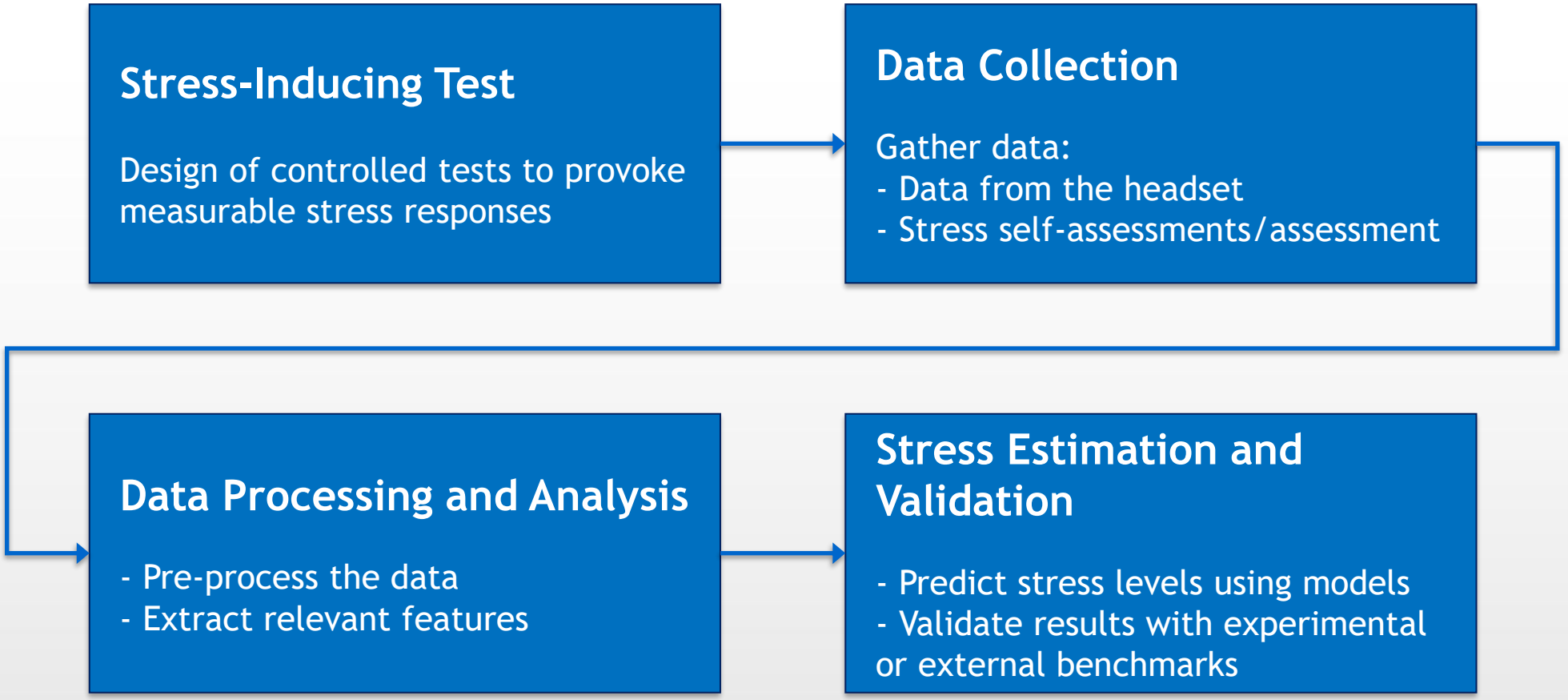
- **Exposure Therapy:** controlled exposure to stressful situations in VR helps treat anxiety disorders, phobias, and PTSD
- **Stress Management:** virtual environments can help individuals practice relaxation techniques and stress-coping strategies

Entertainment:

- **Adaptive gameplay** that adjusts difficulty based on stress levels enhances the player's experience mantaining them in the flow



Stress system pipeline



Design and Stress Tracking

Develop engaging VR environments to induce stress effectively

1. **Cognitive Stressor Tasks (eg. Stroop Test):** StroopTestVR
2. **Trier Social Stress Test (TSST):** TSSTVR
3. **Ad-hoc simulations**

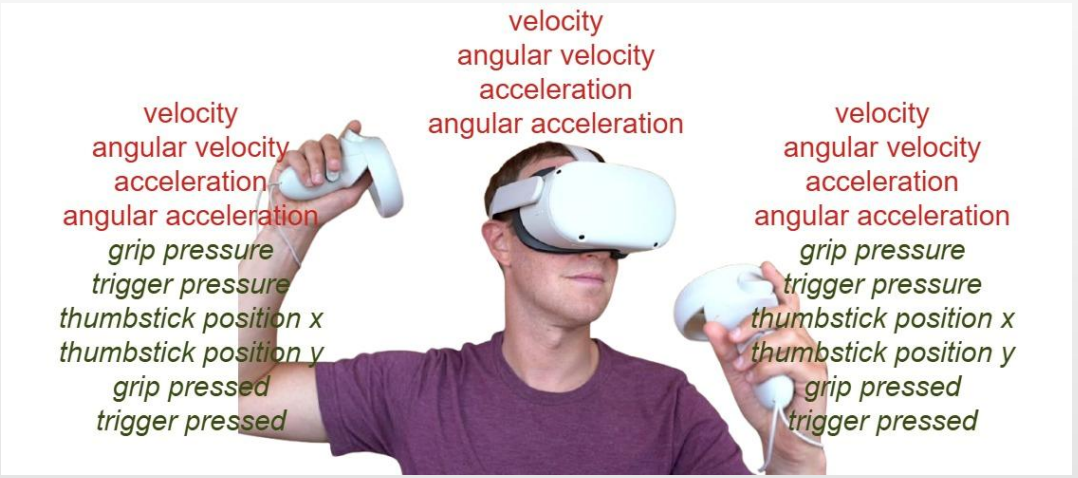
Integrate advanced stress-tracking systems to monitor users during gameplay or simulation:

- **Behavioral Indicators:** User actions, patterns, and decision-making during scenarios
- **Environmental Contexts:** The nature of in-scenario events that might trigger stress responses

Data Collection

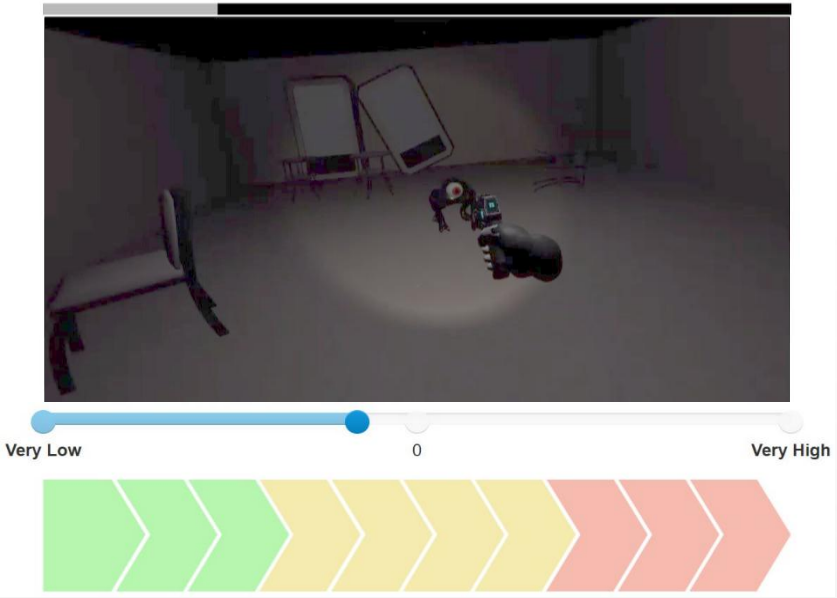
Data	Description
Face-tracking	Tracks users' facial expressions as Action Units (AU) weighted between 0 and 1
Eye-tracking	Identifies gaze position by monitoring eye movements
Voice Signal	Integrated microphones capture speech
Head-tracking/Hand-tracking	Tracks six degrees of freedom (6DoF), including rotational and positional head/controllers movements
Finger-tracking	Detects finger placement on individual buttons

Quest Pro



Stress assessment

DANTE (Dimensional Annotation Tool for Emotions) is tool for annotating and analyzing emotional responses



Developed By:

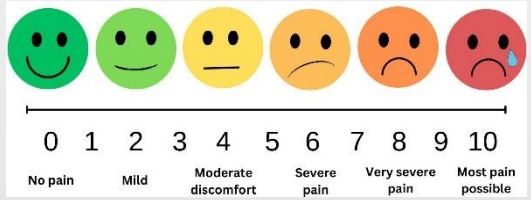
- **PHuSe Lab**, Università degli Studi di Milano, Italy

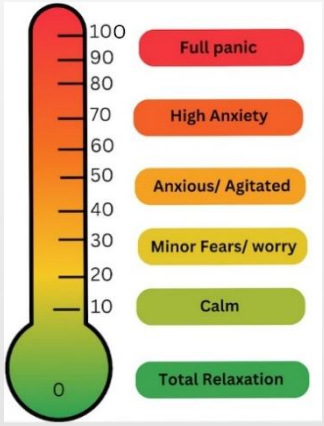
Purpose:

- **Analyze and annotate** affective responses to stimuli
- **Continuous self-report** of perceived stress by participants

Standard questionnaires:

- **STAI-Y1 (State-Trait Anxiety Inventory - Form Y1):** measures current anxiety in response to a situation using a 4-point Likert scale
- **VAS (Visual Analog Scale)**: participants mark their current stress level on a continuous line from 0 to 100 on a visual scale
- **SUDS (Subjective Units of Distress Scale)**: measures the intensity of distress in a given moment on a scale from 0 (No distress) to 100 (Maximum distress)





Data Processing

Characteristics of VR Data:

- **Temporal dependence:** data is collected in sequences over time, where timing is crucial
- **Multimodal nature:** data comes from multiple sensors
- **High Dimensionality:** combining modalities results in large datasets

***Preprocessing Techniques*:**

- **Time window selection:** define specific time intervals to capture relevant patterns
- **Feature Extraction:** deriving meaningful features from raw data
- **Feature Selection:** filter out irrelevant data to focus on high-impact indicators for stress assessment

Feature Extraction

***Eye Gaze Features*:**

- **Fixation duration and frequency**: measure how long and how often a user focuses on specific points
- **Scan Path:** the sequence of gaze movements, showing attention shifts

Facial Expression Features:

- Analysis of **Facial Action Units (AUs) across different facial areas,** which can indicate stress-related expressions

Body Movement Features:

- **fine-grained analysis** of hand and head movements for signs of stress, hesitation, or fatigue

Controller Interaction Features:

- **Button Press Timing & Patterns**
- **Force applied on triggers:** measurement of force applied, indicating frustration or stress.

For all data types, **statistical features** such as mean, variance, peak values can be used to analyze trends, variability, and extreme event

Feature Selection

Why it matters:

- improves model performance by **removing redundant or irrelevant data**
- **reduces computational cost** and **prevents overfitting**

Filter Methods:

Use **statistical measures** (e.g., variance to remove features that do not change much, and covariance to eliminate redundant features that are highly correlated)

Wrapper Methods:

Evaluate subsets of features using **predictive models** to determine the best performing combination

Embedded Methods:

Feature selection is **integrated during model training**, such as using LASSO regularization to shrink unimportant features' coefficients

Time Series Modeling for Stress estimation

Modeling Approaches:

1. **Traditional Time Series Models:**
   - **Hidden Markov Models (HMMs):** Ideal for identifying discrete affective states over time by modeling transitions between hidden states
   - **Kalman Filter:** Used for state estimation in dynamic systems, the Kalman Filter helps track and predict stress levels by filtering out noise from sensor data

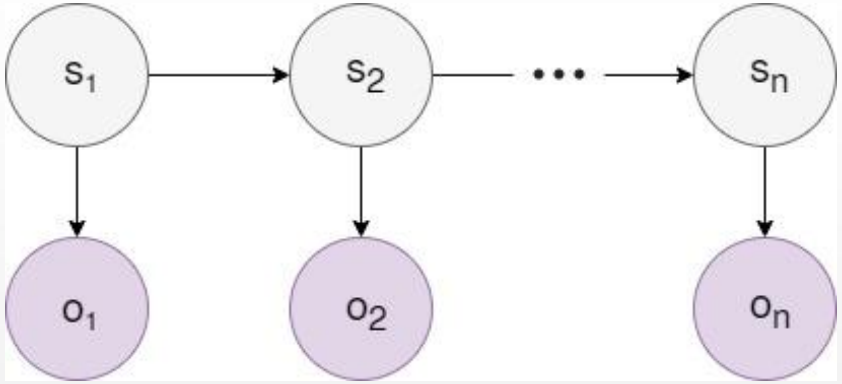
2. **Deep Learning for Time Series:**
   - **Recurrent Neural Networks (RNNs):** Effective for sequential data that has long-term dependencies
   - **Long Short-Term Memory (LSTM):** Specialized in handling long-range dependencies and retaining information over time

3. **Anomaly Detection:**
   - **Variational Autoencoders (VAEs):** Used for anomaly detection by learning a probabilistic model of the data, which helps identify outliers or abnormal patterns in stress-related data

State Space Models

State Space Model (SSM) for stress level dynamics in which each hidden state s_t generates physiological and behavioral observations (vector o_t) at each time step.

SMM to perform posterior inference about hidden states (stress level state estimation).



Problems:

- **Classification**, stress as a discrete variable → **Hidden Markov Model (HMM):** models transitions between discrete states and the likelihood of observing certain behaviors at each state
- **regression,** stress as a continuous variable → **Kalman Filter (KF):** estimates the stress level over time, accounting for noise in the observations and making real-time predictions based on past data

Variational AutoEncoders (VAEs)

VAEs* are a type of **generative model** that **learns a probability distribution over the latent space** (map the input to a **distribution),** allowing the **reconstruction** of the input data and **generation** of new data

Unsupervised Learning: work without labeled data, making them suitable for various applications

Architecture

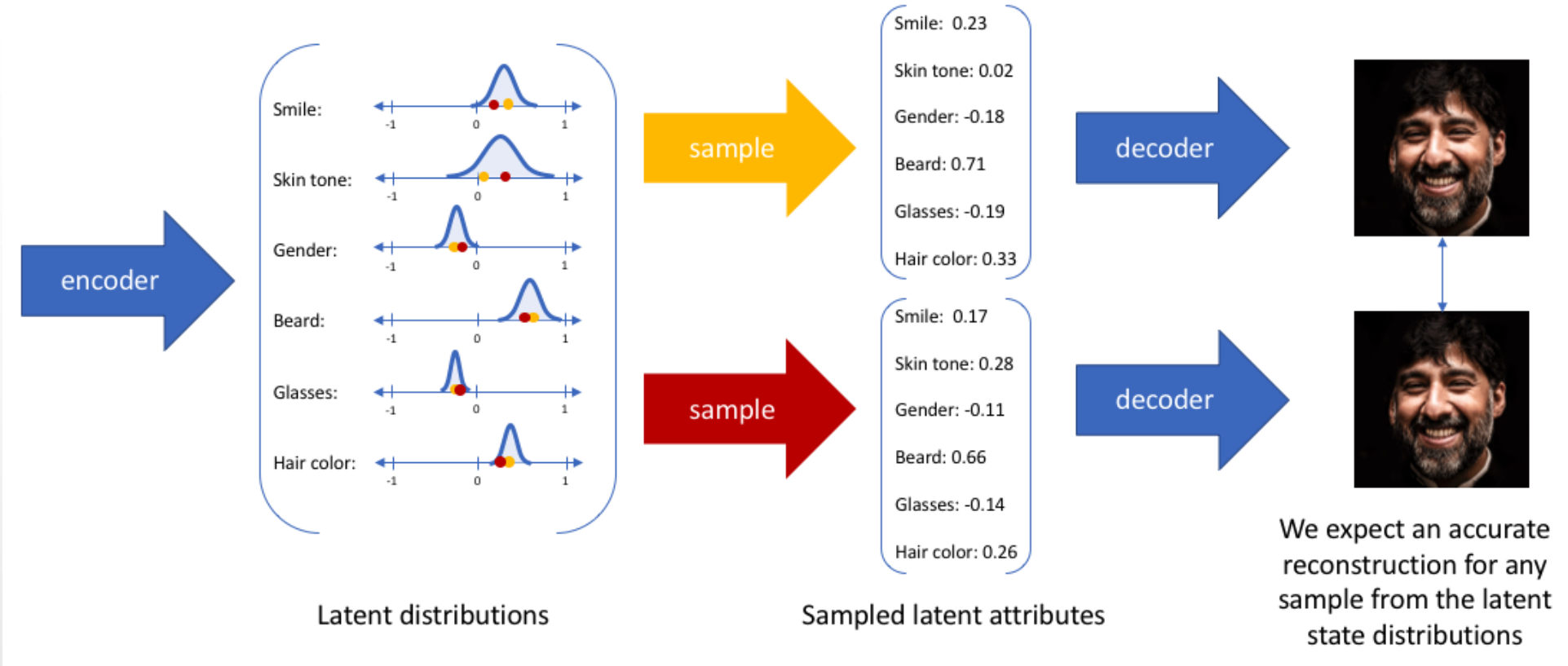
1. **Encoder (network):** maps the input data to the parameters of a **probability distribution** over the latent space: mean $\mu(x)$ and variance $\sigma(x)$

2. **Latent Space (Vectors z):** represents a **probabilistic space** from which we can sample new values to generate similar data

3. **Decoder (network):** reconstructs data from a sample taken from the latent space, enabling both **reconstruction and generation**

* tutorial-on-vae

VAE architecture



Anomaly detection

VAEs can model normal motion patterns, and **deviations** from these patterns (such as stress-induced movements) can be flagged as **anomalies** based on latent space deviations

- **Training:** exclusively on data without stress or with known low stress levels in order to learn the latent space representation of typical, stress-free movement patterns
- **Reconstruction:** reconstruct normal data with minimal error
- **Anomaly detection**: the VAE is trained, new motion data can be input into the model to detect anomalies

A **threshold for the reconstruction error or latent space distance** is defined → If the error or distance exceeds a threshold, the data point is classified as an anomaly

Anomaly detection in facial skin temperature using VAE

Open issues

Multimodal Fusion:
Combining data from multiple sensors (e.g., gaze, facial expressions, movement) into a unified model.

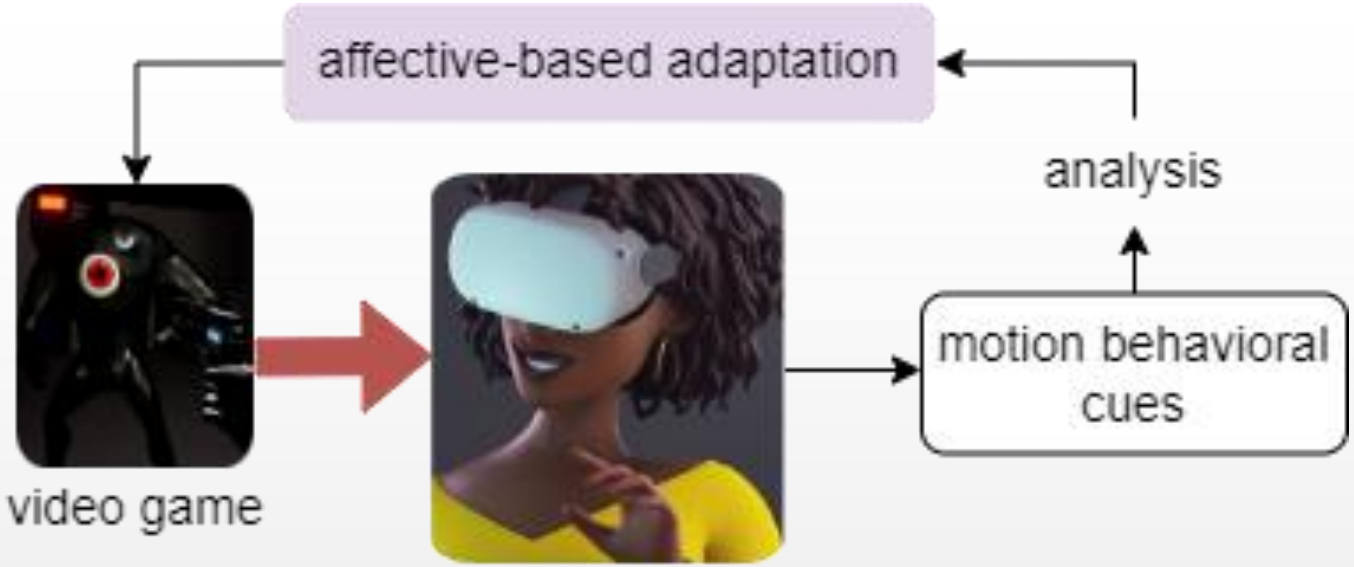
Generalization Across Subjects:
Adapting models to individual variability in stress responses.

Data Requirements:
Need for large datasets to capture diverse stress scenarios.

Real-Time Forecasting:
Achieving fast, accurate stress level predictions in real-time.

Main goal

Affective Gaming (AG):



Idea: design and develop **a system that dynamically adapts** a virtual reality **game content** based on players' stress in real-time

Thanks for your attention!

LSTM per la predizione dello stress

Dott. Marco Ligabue marco.ligabue@unimi.it
Dott. Susanna Brambilla susanna.brambilla@unimi.it
Prof. N. A. Borghese alberto.borghese@unimi.it

aislab

POM9
Playlab fOr inNovation in Games

UNIVERSITAS STUDIORUM MEDIOLANENSIS

Recurrent Neural Networks

Le RNN sono reti neurali progettate per elaborare dati sequenziali, utilizzando connessioni ricorrenti per considerare contesti temporali.

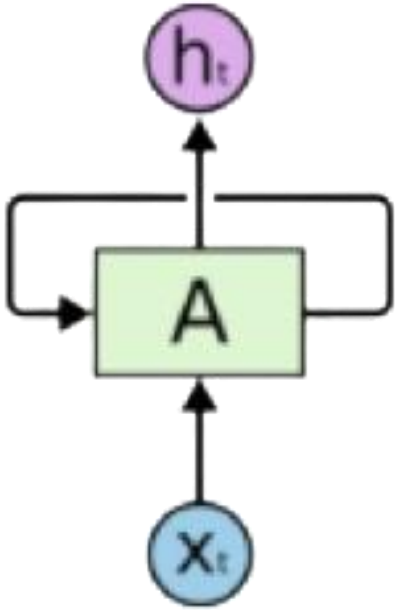
L'output a ogni passo dipende sia dall'input corrente sia dallo stato precedente.

$$h_t = \sigma(W_x x_t + W_h h_{t-1} + b_h)$$

dove W_x e W_h sono matrici dei pesi, b_h è il bias e

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Sono adatte a problemi sequenziali: testo, segnali temporali, audio.



Svantaggi delle RNN

Problema del Vanishing Gradient: durante l'addestramento, i gradienti si riducono esponenzialmente in rete, rendendo difficile l'apprendimento di dipendenze a lungo termine.

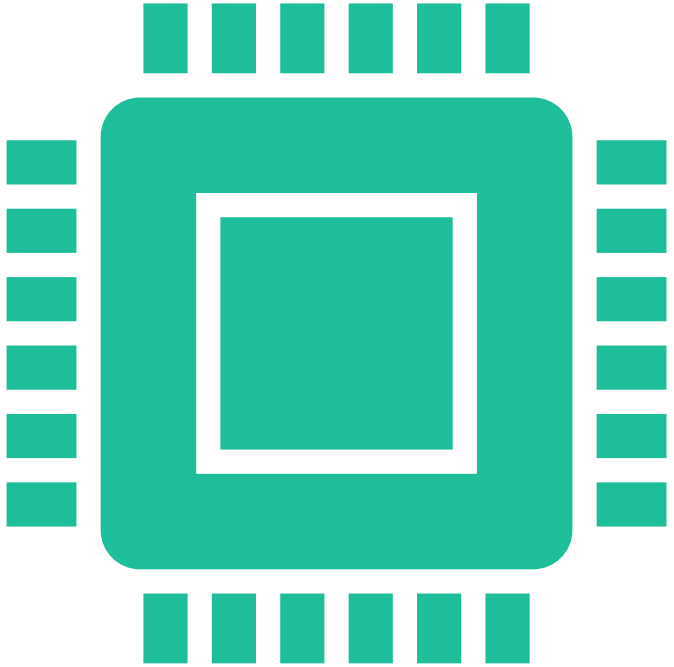
Le LSTM sono state sviluppate per superare questo problema.

Introduzione alle LSTM

Le LSTM (Long Short-Term Memory) sono una particolare RNN progettata per gestire meglio le dipendenze a lungo termine.

Includono celle di memoria che aiutano a 'ricordare' informazioni per periodi prolungati, superando i limiti delle RNN standard.

Hanno ampie applicazioni nella classificazione, nell'elaborazione dei dati, nell'analisi di serie temporali, nel riconoscimento vocale, nella traduzione automatica, nel rilevamento dell'attività vocale, nel controllo dei robot, nei videogiochi e nella sanità.



Vantaggi delle LSTM

Memoria a Lungo Termine: Grazie alle celle di memoria, le LSTM possono mantenere informazioni utili per periodi prolungati.

Adatte a Dati Sequenziali: Ideali per analizzare dati sequenziali o temporali, come testi, serie storiche, e audio.

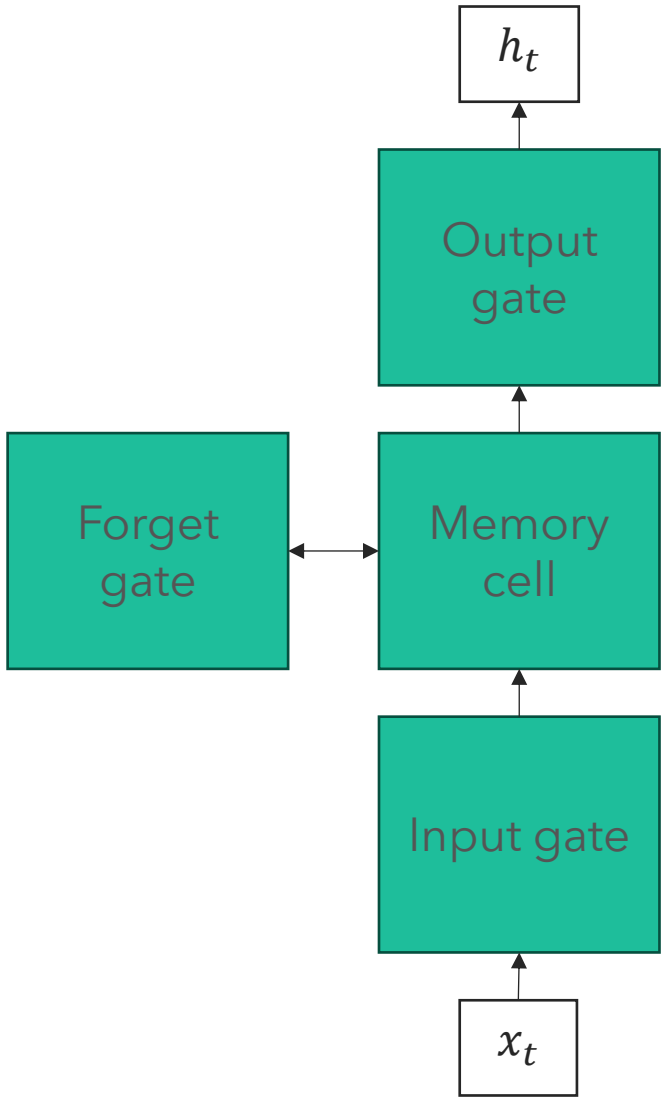
Risoluzione (parziale) del Vanishing Gradient Problem: La struttura dei gate delle LSTM permette di conservare gradienti di apprendimento, rendendo possibile la cattura di pattern su lunghe sequenze.

Architettura

L'architettura di una **LSTM** è composta da una struttura ricorsiva che include:

- **Cella di memoria**: dove vengono memorizzate informazioni utili nel tempo.
- **Tre gate principali**:
  - **Input Gate**
  - **Forget Gate**
  - **Output Gate**

Ogni gate utilizza combinazioni di x_t (input attuale) e h_{t-1} (stato nascosto precedente).



Input Gate

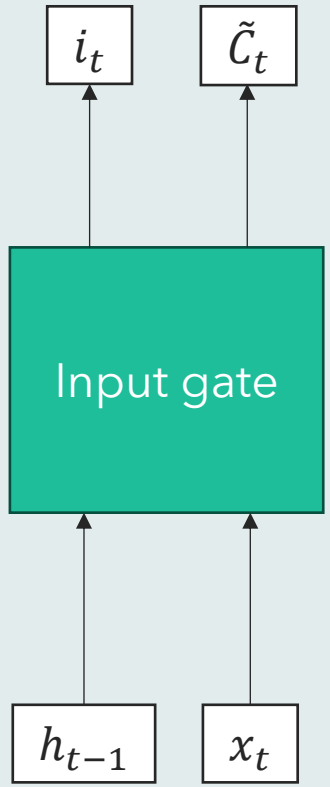
L'**input gate** è il componente che regola quali informazioni del nuovo input x_t devono essere aggiunte allo stato della cella C_t.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

Dove W_i è la matrice dei pesi appresa durante l'apprendimento.

Viene quindi calcolato un *candidate cell state* ($\tilde{C}_t$) che rappresenta i nuovi valori proposti per la cella di memoria.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

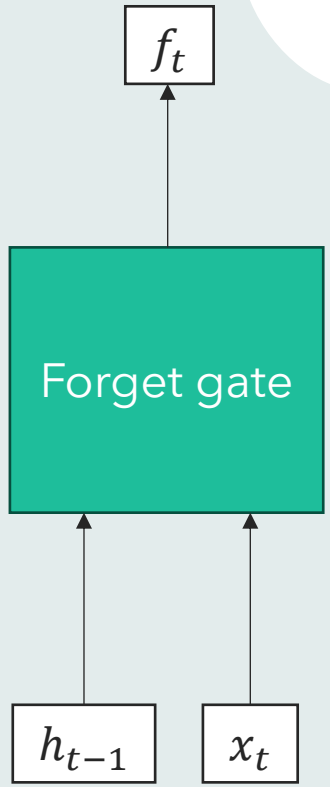


Forget Gate

Il **forget gate** è il componente che decide quali informazioni dello stato della cella precedente (C_{t-1}) devono essere dimenticate o mantenute.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

Dove W_f è la matrice dei pesi appresa durante l'addestramento.

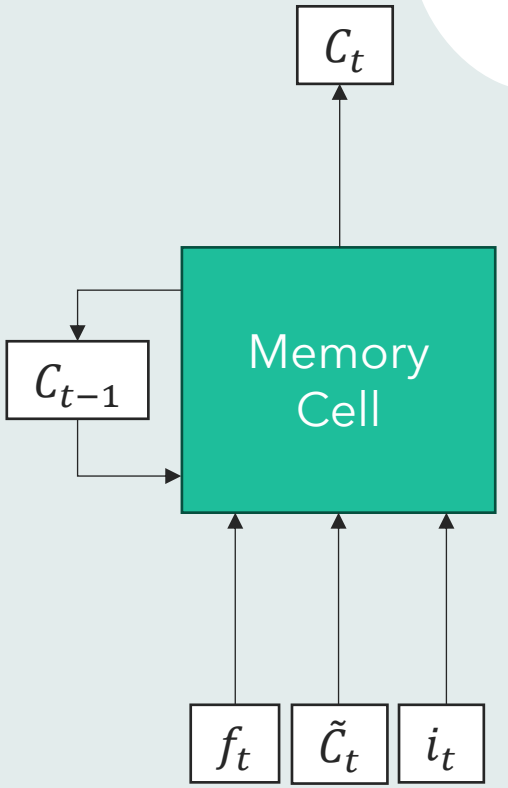


Memory Cell

La **memory cell** C_t rappresenta lo stato interno della rete LSTM, che integra memoria a lungo termine e dinamiche temporali.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Lo stato della cella combina le informazioni da mantenere (C_{t-1}) e le nuove informazioni proposte ($\tilde{C}_t$), pesate rispettivamente da f_t e i_t.



Output Gate

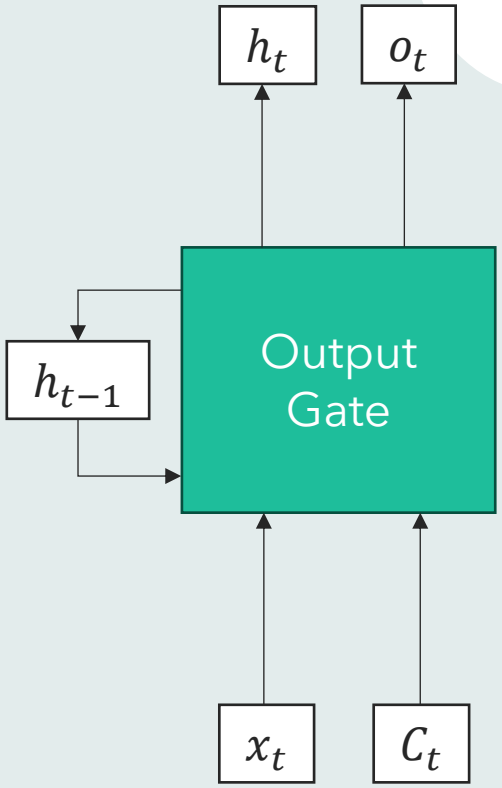
L'**output gate** regola quali informazioni dello stato della cella C_t devono essere trasferite allo stato nascosto h_t, rappresentando l'output della cella.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

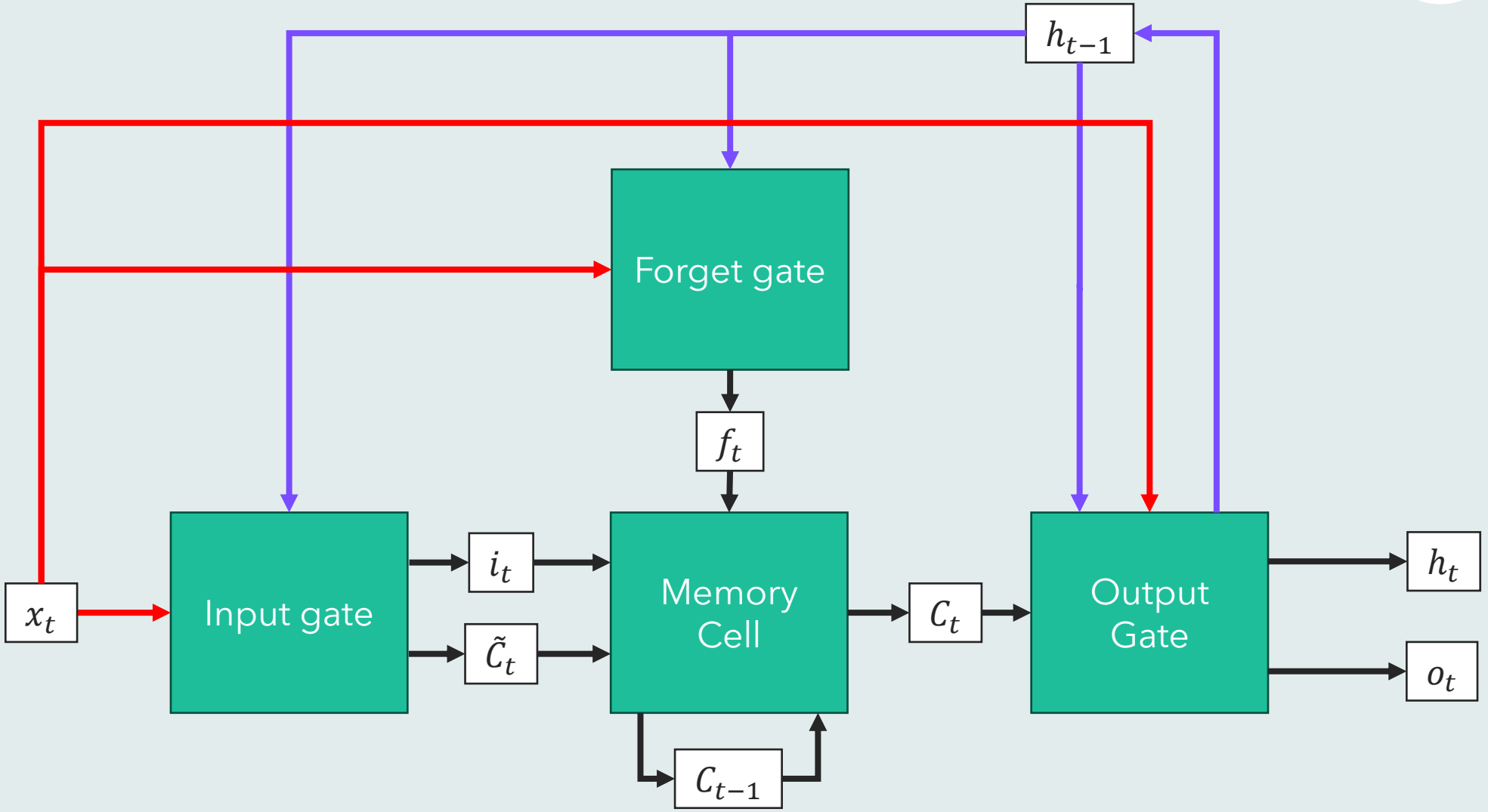
Dove W_o è la matrice dei pesi appresa durante l'addestramento.

Lo stato nascosto viene calcolato applicando una funzione $\tanh$ su C_t, modulata da o_t:

$$h_t = o_t \cdot \tanh(C_t)$$



Schema LSTM


h_{t-1}
Forget gate
f_t
x_t
Input gate
i_t
$\tilde{C}_t$
Memory Cell
C_t
Output Gate
h_t
o_t
C_{t-1}

Parameter Tuning

- **Numero di layer**: Strati della rete che determinano la profondità del modello.
- **Hidden States**: Numero di neuroni nella cella, determina la capacità di memoria.
- **Learning Rate**: Tasso di apprendimento che influenza la velocità di convergenza.
- **Dropout Rate**: Riduce overfitting spegnendo connessioni intra-layer randomicamente.
- **Weight Decay**: Aggiunge regolarizzazione L2.

Suggerimenti per il Tuning:

- Usare tecniche come **early stopping** per evitare overfitting.
- Provare diverse combinazioni di parametri per trovare la configurazione più adatta al problema in esame.

Aggiornamento dei pesi in una Rete Neurale

Gradiente: $$\nabla E(w) = \frac{\delta E(\{w\}|\dots)}{dw_1} + \frac{\delta E(\{w\}|\dots)}{dw_2} + \frac{\delta E(\{w\}|\dots)}{dw_3} + \frac{\delta E(\{w\}|\dots)}{dw_4} + \dots.$$

Modifico il valore dei pesi di una quantità proporzionale alla pendenza della funzione costo rispetto a quel parametro.
Estensione della tecnica del gradiente a più variabili.

$$\mathbf{\Delta w = -\eta\, \nabla E(w)} \Leftrightarrow \Delta w_{ij} = -\eta \frac{\delta E(\{w\}|\dots)}{dw_{ij}} \qquad \eta < 1$$

ADAM (Adaptive Moment Estimator)

ADAM è un algoritmo di ottimizzazione che serve per **addestrare i modelli di apprendimento automatico**, aggiornando pesi e bias in modo efficiente per minimizzare la funzione di perdita.

Introduce di fatto un **learning rate variabile**.

Parametri di ADAM:

- β_1 : Fattore di decadimento per il primo momento.
- β_2 : Fattore di decadimento per il secondo momento.
- η : Learning rate iniziale.
- ε : Termine di stabilizzazione per evitare divisioni per zero.

ADAM (Adaptive Moment Estimator)

1. Stima del primo momento (media mobile dei gradienti)

$$M_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

2. Stima del secondo momento (media mobile del quadrato dei gradienti)

$$V_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

3. Correzione del bias

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \qquad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

4. Aggiornamento del parametro

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}$$

ADAM (Adaptive Moment Estimator)

Prima:

$$\theta_t = \theta_{t-1} - \eta \cdot g_t$$

Dopo:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}$$

Stateful LSTM

Una *Stateful LSTM* è una variante della LSTM che conserva l'*hidden state* e il *cell state* tra i batch consecutivi. Questo permette al modello di apprendere dipendenze temporali a lungo termine senza ripartire da uno stato iniziale a ogni batch.

Caratteristiche principali:

- **Persistenza dello stato**: lo stato viene passato tra batch successivi, mantenendo la continuità temporale.
- **Gestione sequenziale**: ideale per serie temporali o dati con una struttura ordinata nel tempo (e.g., predizione di stress, analisi di segnali).
- **Performance migliorata**: evita la perdita di informazioni utili sulle dipendenze a lungo termine.

Possibili svantaggi

- In modelli stateful, errori accumulati nello stato possono compromettere l'apprendimento.
- Sequenze di lunghezza variabile richiedono padding o troncamento.
- Le LSTM elaborano sequenze **passo per passo**, risultando inefficaci per sequenze lunghe.
  - Si possono addestrare con più sequenze temporali in parallelo passando un **batch** con un *timestamp* per ogni sequenza ad ogni step.
  - Possibili soluzioni: minLSTM e PerceiverIO

IL NOSTRO ESPERIMENTO

IL VIDEOGIOCO

Ambientazione: il giocatore è un astronauta che deve scappare da una stazione spaziale, affrontando gli alieni invasori in un ambiente ostile, sopravvivendo con risorse scarse a disposizione.

- First-person **survival horror**
  - ➢Ottimo per evocare forti reazioni emotive.

Tipi di dati

Label

Autovalutazione tramite DANTE visionando la propria sessione di gioco

Movimento

Controller e Headset:

- Posizione, Velocità, Velocità angolare
  - ➢ Vector3
- Rotazione
  - ➢ Quaternion

Eye-tracking

Eye-space (solo rot.), head-space e world-space

- Posizione
  - ➢Vector3
- Rotazione
  - ➢Quaternion

Tipi di dati

Bottoni

- Button pressure
  - ➢ float (range 0-1)
- Button pressed
  - ➢ int [0-1]
- Thumbstick position
  - ➢ Vector2

Face

- 63 Action Unit basate su FACS (Facial Action Coding System)
  - ➢float(range 0-1)

External

Dati di contesto e semantici

Preparazione dei dati

1

Estrazione finestre temporali

- Dati divisi per tipologia
- Estratte finestre di 0.5s con overlap 50%

2

Estrazione features

- Media
- Mediana
- Varianza
- RMS
- Deviazione standard
- Massimo
- Minimo
- Somma

3

Selezione features

- Selezione tramite analisi varianza e covarianza

Parametri utilizzati

Optimizer Adam (Adaptive Moment Estimator)

- Learning rate: 1e-05
- Weight Decay: 0.001

LSTM

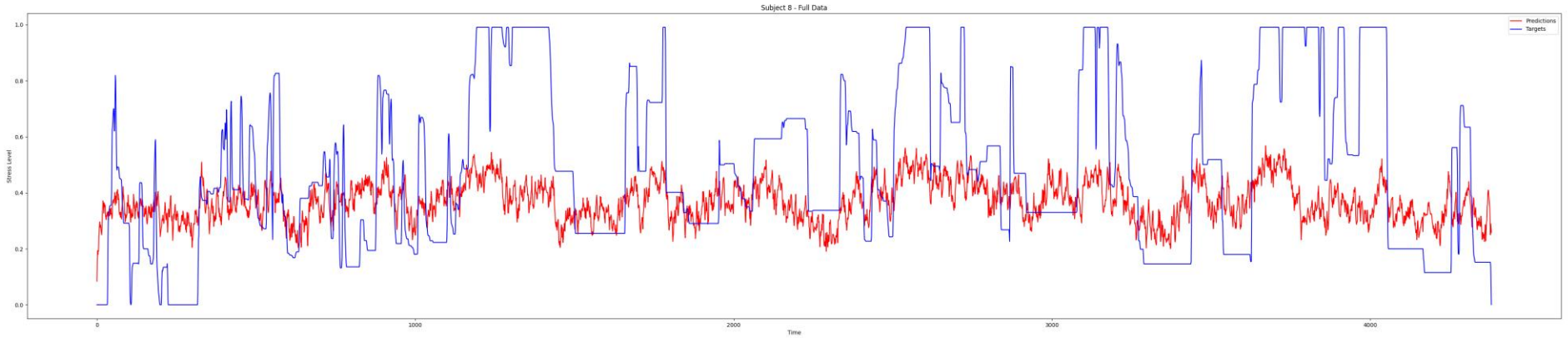
- Hidden Size: 150
- Layers: 2
- Dropout: 0.6
- Layer Feed Forward per previsione stress

Addestramento

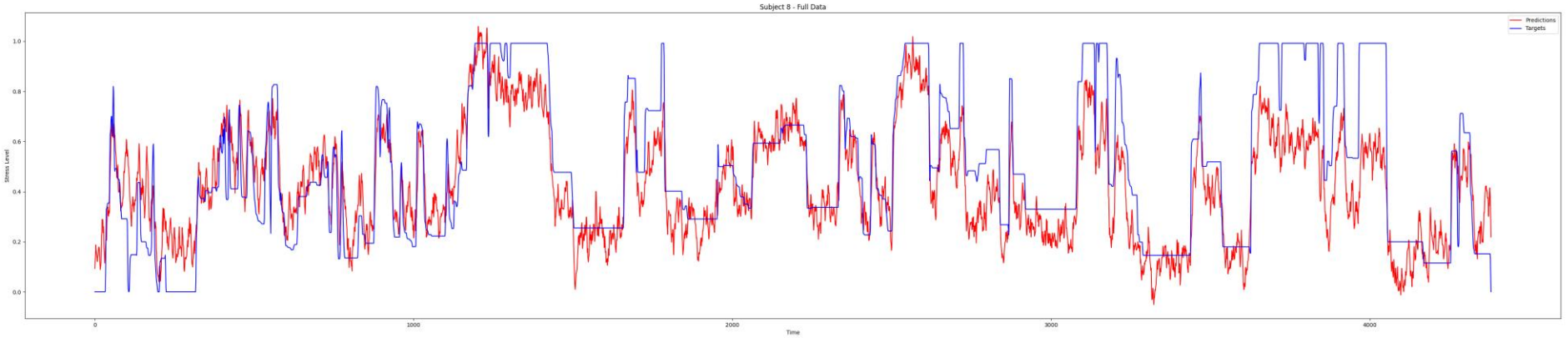
- Epochs: 35
- Batch Size: 1

Utilizzo Stress

Without feature selection

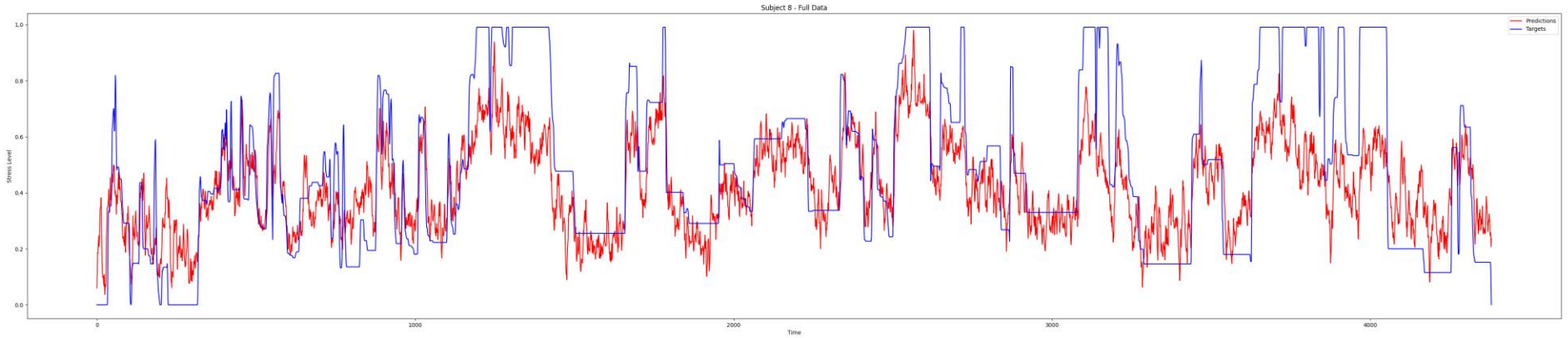


With feature selection

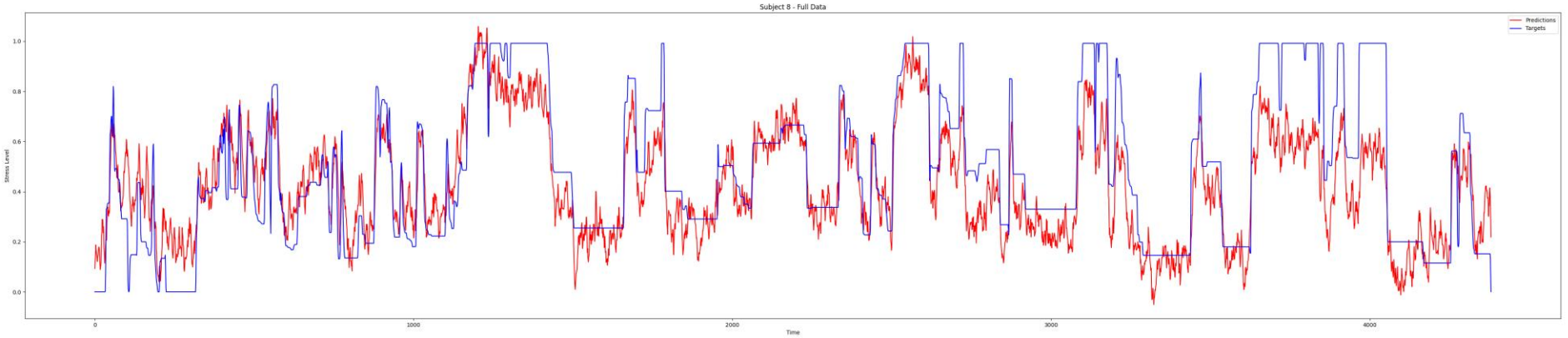


Utilizzo Stress – Soggetto bello

Using eye-tracking data

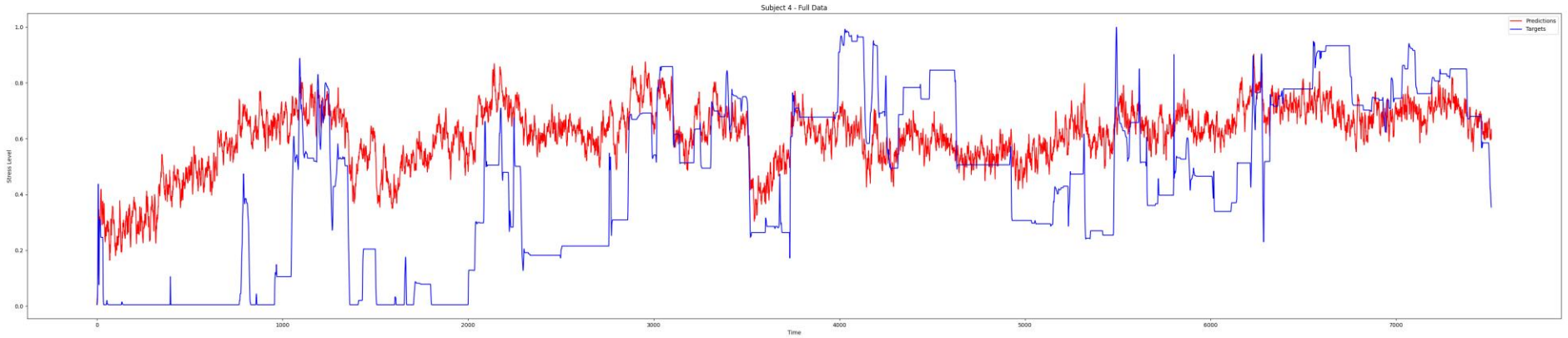


Using buttons data

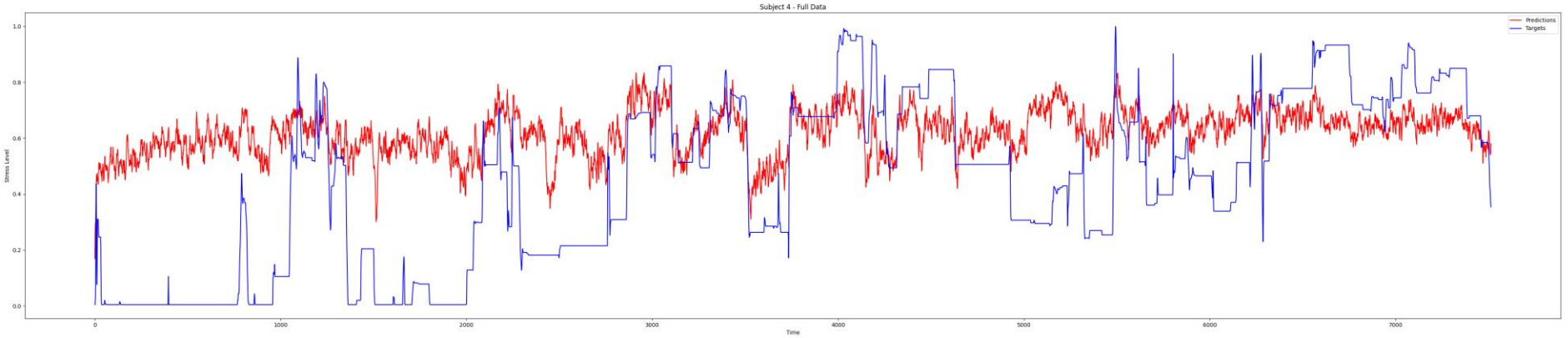


Utilizzo Stress – Soggetto brutto

Using eye-tracking data

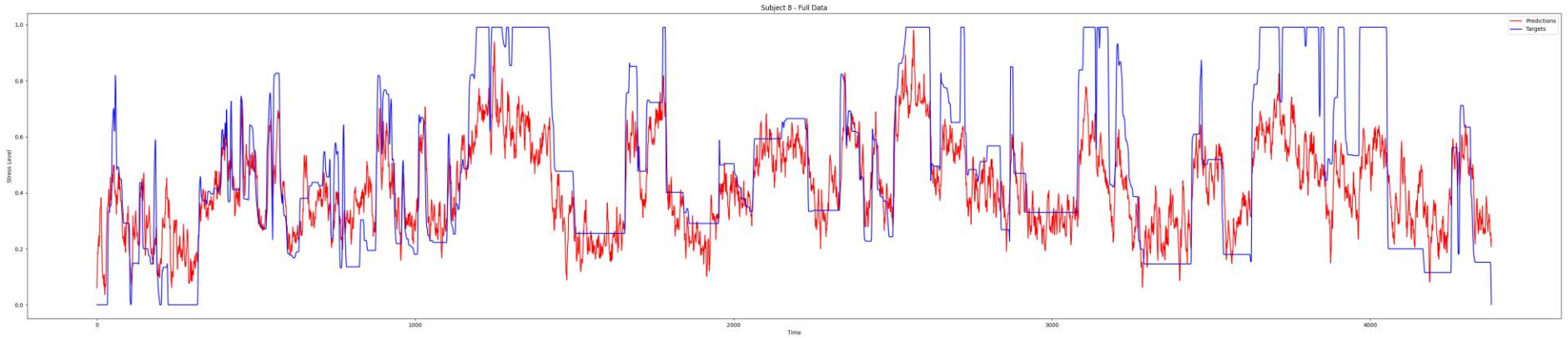


Using buttons data

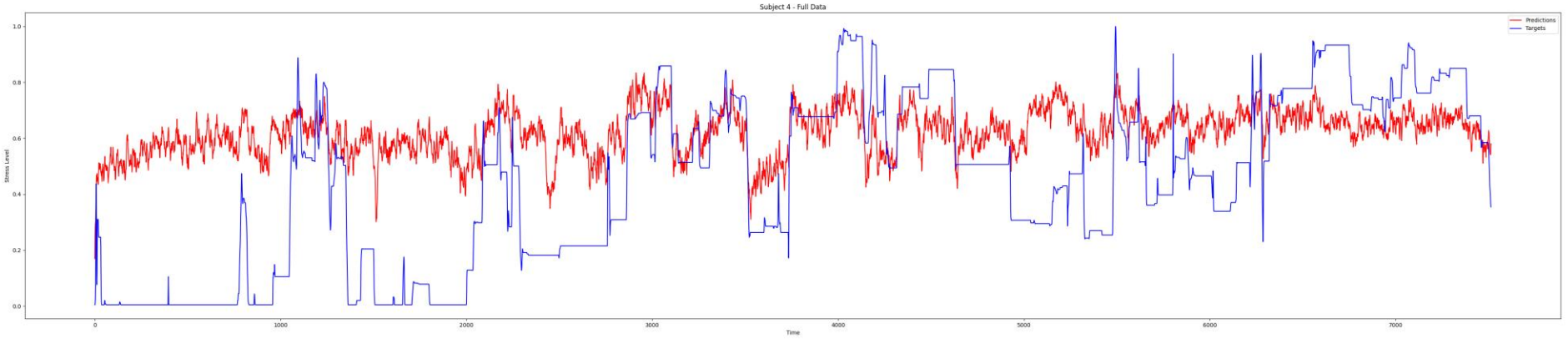


Utilizzo Stress – Soggetto brutto

Using eye-tracking data



Using buttons data



Osservazioni

Soggetti efficaci con un tipo di dati lo sono anche con gli altri, e viceversa.

Possibili cause:

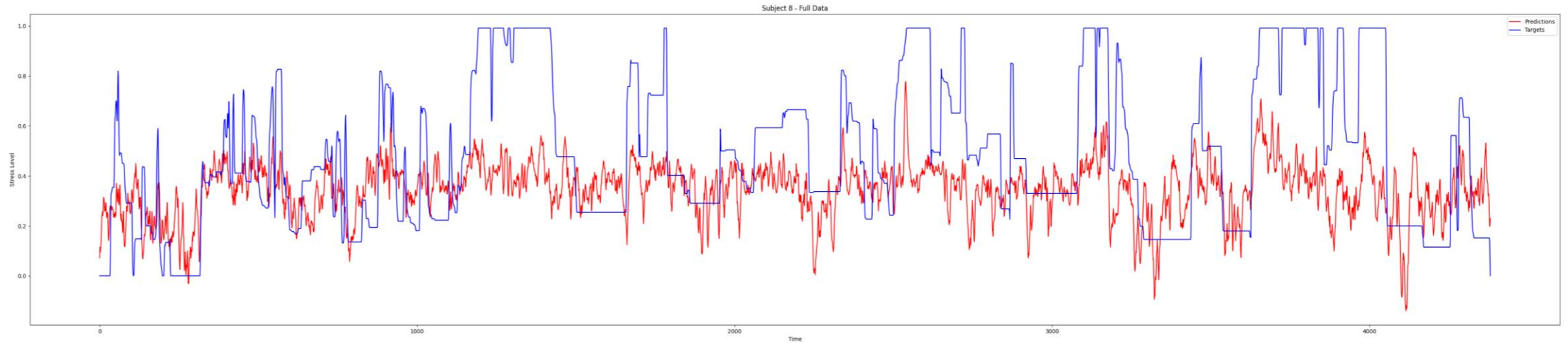
- I risultati dipendono fortemente dalla accuratezza delle label (autovalutazioni)
- Vi sono soggetti più espressivi che quindi aiutano il modello a riconoscere lo stress
- Le feature estratte non sono significative per tutti i soggetti
- Sequenze temporali troppo lunghe

Osservazioni – Dati facciali

Si è riscontrato una scarsa efficacia dei dati relativi alle espressioni facciali

Possibili problematiche:
- Raccolti durante comunicazione verbale
- Facilmente simulabili o falsificabili
- Limitazione hardware nell'accuratezza dell'acquisizione (es. dovuta a presenza di occhiali)

Possibili sviluppi:
- Divisione in gruppi muscolari significativi (es. contorno occhi)
- Migliorare estrazione e selezione features (attualmente la selection ne scarta l'89%)



Sviluppi «futuri»

Rielaborazione dei dati

- Estrazione accelerazioni
- PCA
- Conversione quaternioni

Selezione features «globale» per generalizzazione su più soggetti

Test con nuovi modelli

- minLSTM
- PerceiverIO

Gating tra le diverse tipologie di dati

Grazie per l’attenzione!