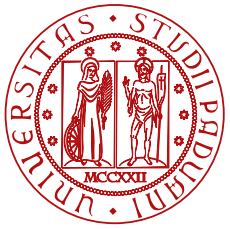

Università degli Studi di Padova

Tesi Magistrale in Ingegneria Meccatronica

Miglioramento della Predizione del Movimento Umano tramite l'Integrazione di Descrizioni delle Azioni

Candidato

Pietro Mercorella

ID Studente 2056786

Relatore

Monica Reggiani

Università di Padova

Correlatore

Stefano Michieletto

Università di Padova

Michael Vanuzzo

Università di Padova

Anno Accademico
2023/2024

Abstract

Il problema dell'anticipazione del movimento umano consiste nel prevedere i movimenti futuri di una persona basandosi su quelli passati. La capacità di fare previsioni accurate permette di migliorare l'interazione uomo-macchina, rendendo questi sistemi più reattivi e sicuri. Tuttavia, ottenere sistemi predittivi che siano accurati e robusti è ancora un problema aperto, in quanto è difficile modellare la complessità e la variabilità del movimento umano. I modelli attuali allo stato dell'arte si basano solo sui dati delle pose passate della persona, senza includere informazioni sul contesto. Nonostante ciò, questo non è sufficiente, in quanto tali modelli falliscono nel prevedere con accuratezza i movimenti che includono transizioni tra azioni, ad esempio passare dal camminare al sedersi.

L'obiettivo di questa tesi è quello di realizzare un modello innovativo di predizione del movimento umano, integrando l'informazione dei movimenti passati con la descrizione testuale della semantica del movimento stesso. Questa integrazione permette di aggiungere contesto ai movimenti e migliorare l'accuratezza delle previsioni, specialmente nelle transizioni tra diverse azioni. Il modello di anticipazione del movimento si basa sul Deep Learning e l'elaborazione delle informazioni testuali avviene tramite l'uso di un Large Language Model. Nel lavoro di questa tesi sono stati condotti numerosi esperimenti utilizzando diversi modelli di anticipazione e molteplici modelli di linguaggio. Per determinare l'efficacia dell'integrazione delle descrizioni testuali, è stata confrontata l'accuratezza delle previsioni dei modelli basati solo sulle pose passate con quelli che integrano anche l'informazione semantica.

Tale approccio rappresenta uno strumento fondamentale per il futuro sviluppo di algoritmi di previsione del movimento umano, risultando applicabile in modo trasversale per il miglioramento di diversi modelli predittivi. Questa integrazione permette di aprire nuove possibilità per applicazioni avanzate in vari settori come la robotica, la sorveglianza e l'interazione uomo-macchina. L'approccio proposto rappresenta un passo avanti verso sistemi predittivi più intelligenti e versatili, capaci di comprendere e anticipare i movimenti umani in maniera più efficace e affidabile.

Indice

Lista delle Figure

Lista delle tabelle

Lista degli Acronimi

IA Intelligenza Artificiale

ML Machine Learning

DL Deep Learning

HMP Human Motion Prediction

NLP Natural Language Processing

LLM Large Language Model

GT Ground Truth

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

GRU Gated Recurrent Units

NLP Natural Language Processing

CNN Convolutional Neural Network

PVRED Position-Velocity Recurrent Encoder-Decoder

QT Quaternion Transformation

BERT Bidirectional Encoder Representations from Transformers

MSE Mean Squared Error

DNN Deep Neural Network

MPJPE Mean Per Joint Position Error

AMASS Archive of Motion Capture As Surface Shapes

BABEL Bodies, Action and Behavior with English Labels

SMPL Skinned Multi-Person Linear model

MLM Masked Language Modeling

NSP Next Sentence Prediction

ZV Zero Velocity

JAD Joint Angle Difference

MAE Mean Angular Error

1

Introduzione

Quando si pensa all'Intelligenza Artificiale (IA), l'immaginario collettivo si popola immediatamente di tecnologie avanzate, robot capaci di comprendere e decidere azioni, e un mondo futuristico in cui macchine e esseri umani convivono. Tuttavia, l'IA e il suo utilizzo sono molto più concreti e attuali di quanto si possa immaginare, trovando applicazione in numerosi settori della vita quotidiana. Gli utilizzi odierni dell'IA sono spesso meno invasivi di quanto viene suggerito dai film di fantascienza, che hanno ampiamente sfruttato questo tema per creare serie e saghe di vario successo, come nel caso emblematico di HAL9000, l'IA del film "2001: Odissea nello spazio" di Stanley Kubrick.
L'IA, nel suo senso più ampio, è l'intelligenza esibita dalle macchine, in particolare dai sistemi informatici. Si tratta di un campo di ricerca nell'informatica che sviluppa e studia algoritmi e software che consentono alle macchine di percepire il loro ambiente e utilizzare l'apprendimento per compiere azioni che massimizzano le loro possibilità di raggiungere specifici obiettivi, quali la comprensione di comportamenti tipicamente umani.
L'apprendimento automatico è noto con il nome di Machine Learning (ML) ed ha registrato significativi progressi grazie allo sviluppo del Deep Learning (DL), un suo sottoinsieme che sfrutta reti neurali artificiali per affrontare problemi complessi, ispirandosi al funzionamento dei neuroni e delle reti neurali biologiche. Come suggerisce il nome, il DL adotta un approccio 'profondo', composto da più strati di reti neurali, che consente di gestire problemi altamente non lineari, migliorando la capacità dei modelli di apprendere e generalizzare dalle informazioni. Questi modelli sono progettati per affrontare problemi complessi

come il riconoscimento di immagini, l'elaborazione del linguaggio naturale, la traduzione automatica, e la previsione di serie temporali, tra gli altri. Attraverso l'addestramento su grandi quantità di dati, le reti neurali sono in grado di apprendere rappresentazioni gerarchiche delle informazioni, migliorando la capacità del modello di generalizzare e risolvere problemi che richiedono una comprensione profonda dei dati in ingresso.
L'uso delle reti neurali e di algoritmi in grado di riprodurre ragionamenti tipici degli esseri umani nelle differenti situazioni, hanno permesso ai sistemi intelligenti di migliorare sempre di più le diverse capacità di comportamento. Tali algoritmi complessi, inseriti all'interno di sistemi intelligenti, sono quindi in grado di "prendere decisioni", ossia di effettuare scelte a seconda dei contesti in cui sono inseriti. Nel caso degli algoritmi connessi ai sistemi intelligenti dei veicoli, ad esempio, un'automobile senza conducente può decidere, in caso di pericolo, se sterzare o frenare a seconda della situazione. Risulta dunque evidente che l'utilità delle reti neurali ne ha permesso un consistente sviluppo che si è espanso in vari ambiti. Ampio, infatti, è il loro utilizzo e molti sono i campi in cui esse vengono ricercate e sviluppate: dalle aziende, ai progetti universitari, dalle ricerche in campo medico, ad applicazioni su dispositivi comuni.
Le moderne IA, inoltre, stanno trovando un'applicazione concreta e crescente nel contesto industriale contemporaneo, segnando un'evoluzione significativa nelle pratiche produttive. All'interno del sistema manifatturiero attuale, i robot industriali sono ormai diventati una componente essenziale, grazie alla loro capacità di eseguire compiti ripetitivi con velocità e precisione superiori rispetto agli esseri umani. Tuttavia, il panorama industriale sta evolvendo ulteriormente verso un modello di collaborazione uomo-robot, dove le due entità operano fianco a fianco nello stesso spazio di lavoro. In questo contesto, è fondamentale che le macchine acquisiscano la capacità innata degli esseri umani di comprendere e prevedere i movimenti delle persone, al fine di garantire un'interazione fluida e sicura tra uomo e robot.
Questa nuova dinamica collaborativa è resa possibile dall'integrazione nei processi produttivi di architetture di DL avanzate, che permettono ai robot di interagire in modo sicuro ed efficiente con gli operatori umani. I robot apportano vantaggi in termini di rapidità e precisione, rendendoli ideali per operazioni che richiedono un elevato grado di ripetibilità e accuratezza. Dall'altra parte, gli esseri umani possiedono una flessibilità e una capacità di adattamento che consentono di gestire situazioni complesse e non previste, che i robot, con le loro

attuali capacità, non sono ancora in grado di affrontare autonomamente.
Questa collaborazione tra uomo e macchina non solo ottimizza i processi produttivi, ma apre anche nuove possibilità per migliorare la sicurezza sul lavoro, ridurre i tempi di fermo macchina, e aumentare la qualità del prodotto finale. In questo contesto, il ML svolge un ruolo cruciale, facilitando la comunicazione e la cooperazione tra robot e operatori umani, creando un ambiente di lavoro più efficiente e produttivo.
Nella produzione industriale collaborativa uomo-robot, i robot industriali operano fianco a fianco con i lavoratori umani ed eseguono congiuntamente i compiti assegnati in modo continuativo. Questo sistema collaborativo uomo-robot è più personalizzato e flessibile rispetto ai sistemi manifatturieri convenzionali. Nell'ambito dell'assemblaggio, ad esempio, è fondamentale che i sistemi collaborativi uomo-robot possano anticipare le azioni e le intenzioni dei lavoratori umani. L'anticipazione è cruciale in contesti collaborativi perché consente una maggiore coordinazione e fluidità nelle operazioni, migliorando l'efficienza e riducendo il rischio di errori. Quando un sistema è in grado di prevedere i movimenti e le intenzioni di un operatore umano, può adattarsi in modo costruttivo per assistere meglio durante le operazioni svolte dall'operatore. Questo approccio non solo ottimizza il processo produttivo, ma contribuisce anche a una maggiore sicurezza per il lavoratore, poiché il robot può rispondere in modo tempestivo e appropriato alle sue azioni. In questo contesto, la capacità di predire le azioni umane diventa un elemento chiave per il successo della robotica collaborativa, garantendo un'interazione armoniosa e produttiva tra uomo e macchina. In molti ambienti, quindi, risulta necessario un metodo per il riconoscimento e lo studio del comportamento umano, al fine di poter costruire algoritmi in grado di anticiparne i movimenti futuri. La Human Motion Prediction (HMP) è un campo dell'IA che si occupa della previsione dei movimenti futuri di una persona basandosi sulle osservazioni dei movimenti passati. Tuttavia, i sistemi attuali di HMP si basano unicamente sulle informazioni delle pose passate, e poiché il movimento umano è intrinsecamente imprevedibile e influenzato da numerosi fattori variabili, questa limitazione può ridurre l'accuratezza delle previsioni future.

1.1 Presentazione del problema

Questo lavoro si concentra, nello specifico, nel creare e testare un modello predittivo in cui all'informazione della posa viene aggiunta una descrizione dell'azione compiuta dal soggetto, e verificare se ciò migliora la predizione del suo comportamento.
In questa tesi, dunque, verrà presentata una possibile soluzione per la creazione di un modello per la HMP in grado di gestire insieme dataset che contengono informazioni delle pose (catturate con sistemi di motion capture) e dataset riguardanti descrizioni dei movimenti, e far sì che le informazioni tratte dai due dataset si integrino tra di loro.
Questa tesi andrà ad analizzare in primo luogo il problema della HMP che ha portato all'esecuzione del progetto qui presentato; successivamente saranno esposte le scelte attuate per la sua realizzazione, l'implementazione del modello e le prove sperimentali atte a validare il suo funzionamento.
Il modello da sviluppare, dunque, dovrà essere in grado di gestire dati derivanti da dataset contenenti le pose e da dataset contenenti le descrizioni di queste pose e di utilizzarli, attraverso una rete neurale, per generare previsioni future. Il motivo dell'implementazione di questo tipo di architettura è causa della necessità di ottenere un modello predittivo più performante rispetto agli attuali sistemi di previsione. Uno dei principali problemi nella predizione del movimento umano, infatti, è ottenere modelli predittivi che siano in grado di catturare la complessità e la variabilità dei movimenti umani in modo accurato e robusto. Questo è reso ancora più difficile dalla natura altamente dinamica e non lineare del movimento umano, e basarsi esclusivamente su dati storici di pose non permette di catturare adeguatamente le transizioni tra diverse azioni. Per tale ragione, in questa tesi, è stato scelto l'approccio di affiancare alle informazioni riguardanti le pose passate anche informazioni riguardanti la descrizione testuale dell'azione svolta in tali pose. Per mezzo di questa logica, si prevede che il modello predittivo diventi più robusto in quanto gli viene comunicata l'esatta azione svolta dall'essere umano, eliminando, così, la grande varietà di possibili soluzioni future. Ad esempio, se si pensa ad una persona che sta camminando, esistono numerose possibilità di azioni che essa possa eseguire nell'immediato futuro, come continuare a camminare, iniziare a correre, fermarsi o sedersi. Se, tuttavia, con questa posa venisse presentata la frase "Cammina, si ferma e si siede per terra", si conosce esattamente quale sarà l'azione che la persona svolgerà.

Risulta, quindi, evidente che, per riuscire a catturare al meglio la variabilità e l'incertezza del movimento umano, è importante andare a combinare i dati dei movimenti passati con informazioni aggiuntive che descrivono il movimento.
Nei sottoparagrafi che seguono verranno analizzate in dettaglio le due principali tecnologie che costituiscono l'oggetto di studio di questa tesi, evidenziandone le caratteristiche, i vantaggi e le potenziali applicazioni nel contesto specifico della ricerca.

1.1.1 Human Motion Prediction

Spesso, alle persone, capita di eseguire previsioni di come possa agire chi si trova attorno a loro. Questa capacità è essenziale affinché esse possano interagire in modo efficiente. Tramite l'esperienza di una vita, infatti, ogni azione svolta permette di stabilire degli schemi secondo le quali ad ogni azione ne corrisponde un'altra in termini di successione causa-effetto: chiunque potrebbe prevedere che se una persona indossa un cappotto, probabilmente starà uscendo di casa, oppure se prende in mano un libro, entro breve tempo lo aprirà per leggerlo. Sono numerose le possibilità di previsione che una persona ha accumulato tramite l'esperienza e per ognuna di esse è in grado di eseguire una certa previsione con un diverso grado di certezza. Risulta, infatti, meno semplice prevedere cosa farà una persona che sta camminando.
La previsione del movimento umano è un tema cruciale per numerose applicazioni che coinvolgono sistemi intelligenti che operano e collaborano con gli esseri umani in spazi di lavoro comuni. L'obiettivo della HMP è quello di comprendere questi meccanismi che governano il comportamento umano, al fine di tradurlo in modelli predittivi capaci di stimare i movimenti futuri. In qualsiasi ambiente in cui i sistemi intelligenti agiscono in presenza di esseri umani, la previsione di come questi possano agire risulta essenziale. Si pensi ai sistemi di guida autonoma [25], in cui un'azione improvvisa da parte di un pedone può causare situazioni spiacevoli. In modo simile, nei sistemi di produzione industriale, un malfunzionamento improvviso o un errore umano può portare a incidenti che mettono a rischio la sicurezza dei lavoratori. Questi sistemi possono essere anche utilizzati per migliorare tecnologie in vari ambiti, come l'animazione o le industrie video-ludiche, migliorando il realismo e la fluidità nei movimenti dei personaggi raffigurati; nello sport per analizzare e migliorare le prestazioni degli atleti; nell'assistenza sanitaria per i programmi di riabilita-

zione o di disfunzione del movimento umano. Tuttavia, risulta particolarmente complesso ottenere una buona stima del movimento umano, proprio a causa della sua naturale imprevedibilità. Sono molti i fattori che influenzano il suo comportamento e non ci si può aspettare di poterlo prevedere con facilità in quanto l'essere umano agisce spinto non solo dal compito che deve svolgere, ma anche da altri elementi, quali il suo stato fisico ed emotivo. Risulta, quindi, essenziale ricorrere a tecniche e algoritmi innovativi che siano in grado di porre rimedio a queste problematiche.
Applicando questa prospettiva nella robotica collaborativa, perfezionare e automatizzare l'integrazione tra uomo e macchina è una delle sfide principali. I cobot rappresentano una rivoluzione in questo campo, permettendo una collaborazione più stretta e sicura tra robot e operatori umani. Tuttavia, la vera svolta risiede nell'integrazione dell'intelligenza artificiale (IA) nei cobot, che li rende capaci di apprendere e adattarsi agli ambienti di lavoro in tempo reale. Questa capacità di apprendimento, supportata dal ML, è cruciale non solo nella robotica collaborativa ma anche in altre applicazioni.
Ad esempio, nella riabilitazione, la HMP è fondamentale per sviluppare sistemi che possono anticipare i movimenti del paziente, aiutando nel recupero fisico in modo più efficiente e sicuro. I dispositivi indossabili dotati di IA possono monitorare e prevedere i movimenti del paziente, personalizzando gli esercizi riabilitativi in base ai progressi e riducendo il rischio di infortuni.
Nel contesto sportivo, la HMP trova applicazione nel migliorare le prestazioni degli atleti. Attraverso l'analisi predittiva del movimento, è possibile ottimizzare la tecnica, prevenire infortuni e creare programmi di allenamento personalizzati. La capacità di prevedere il comportamento degli avversari in tempo reale può anche offrire un vantaggio competitivo significativo.
Infine, nella realtà virtuale e aumentata, la previsione del movimento umano è essenziale per creare esperienze immersive e interattive. Un sistema che anticipa i movimenti dell'utente può migliorare l'interfaccia uomo-macchina, rendendo l'interazione con gli ambienti virtuali più fluida e naturale.
In tutti questi contesti, la sfida principale consiste nel massimizzare la consapevolezza del sistema riguardo all'ambiente circostante e ai movimenti umani, un elemento che aggiunge complessità al problema della HMP e che è essenziale per una vasta gamma di applicazioni.

Gli algoritmi di DL stanno ottenendo eccellenti risultati nella previsione del movimento umano (HMP). Tuttavia, la maggior parte dei dataset attualmente

disponibili si basa su registrazioni di persone che svolgono attività generiche in ambienti aperti e privi di ostacoli. Per migliorare la capacità di prevedere il movimento umano in contesti più complessi, è cruciale includere informazioni aggiuntive, come la disposizione degli oggetti nello spazio circostante e i dettagli semantici delle azioni svolte. Questo lavoro mira a esplorare quest'ultimo aspetto, utilizzando tecniche di DL che sfruttano modelli di linguaggio per integrare informazioni semantiche, migliorando così le capacità predittive dei modelli esistenti.

1.1.2 Large Language Models

Un'ulteriore caratteristica, unica dell'essere umano, è l'abilità di comunicare attraverso un linguaggio complesso e articolato. Ogni persona, fin dai primi anni di vita, acquisisce l'abilità di comunicare con gli altri e di farsi comprendere attraverso l'uso della parola. Esistono, inoltre, migliaia di lingue e dialetti differenti nel mondo ed ognuno di essi adotta un proprio insieme di regole che ne definiscono la grammatica. Risulta quindi evidente che il linguaggio umano è estremamente complesso e denso, richiedendo una comprensione approfondita per essere gestito correttamente. Per i sistemi digitali che interagiscono con le persone, è essenziale acquisire questa comprensione per elaborare e interpretare il linguaggio in modo efficace. A tale fine è stato sviluppato il Natural Language Processing (NLP), che fornisce ai computer gli strumenti necessari per analizzare, comprendere e generare linguaggio umano in modo coerente. Il NLP [34] è la branca dell'IA che si occupa dell'elaborazione del linguaggio naturale e rappresenta la capacità di un programma informatico di comprendere il linguaggio umano parlato e scritto, noto come linguaggio naturale. Essa costituisce una componente essenziale dell'IA e rappresenta un'area chiave nello sviluppo dei modelli di linguaggio di grandi dimensioni, noti come Large Language Model (LLM). Un LLM è un modello di DL capace di riconoscere, riassumere, tradurre e persino generare contenuti testuali. Tuttavia, per eseguire queste operazioni con precisione ed efficacia, è fondamentale che disponga di enormi dataset. I grandi dataset forniscono la varietà e la ricchezza di esempi necessari per addestrare il modello su una vasta gamma di contesti e stili linguistici, migliorando così la sua capacità di generalizzare e gestire il linguaggio naturale in modo accurato. Inoltre, dataset ampi consentono al modello di apprendere le complessità e le sottigliezze del linguaggio, come le dipendenze contestuali, che

sono essenziali per eseguire compiti complessi come la traduzione e il riassunto in modo coerente e sensato.
L'obiettivo dei LLMs è quello di creare un modello linguistico in grado di apprendere le parole e le relazioni tra queste, imparando a comprendere i diversi significati di un termine identico usato in contesti diversi; inoltre, esso assimila le sue capacità a quelle di un essere umano ed è in grado di costruire frasi corrette dal punto di vista grammaticale e sintattico. Grazie a questa loro abilità, i LLMs assumono un ruolo fondamentale, poiché offrono la capacità di processare e generare il linguaggio in modo sofisticato e avanzato. Questi modelli non solo migliorano la qualità delle interazioni tra uomo e macchina, ma aprono anche la strada a nuove possibilità rivoluzionarie in vari ambiti, dall'automazione dei servizi alla creazione di contenuti intelligenti.

1.2 Stato dell'arte

In questo paragrafo viene illustrato lo stato dell'arte delle tecniche per la previsione del movimento umano (HMP) e dei modelli di linguaggio (LLM), analizzando i principali approcci e i recenti progressi nel settore. Attualmente, la previsione del moto umano è formulata come un compito sequence-to-sequence, in cui uno storico di scheletri 3D alimenta una rete neurale che prevede i movimenti futuri. Tipicamente, la previsione è valutata su due scale temporali: a breve termine, con previsioni fino a 400ms, e a lungo termine, con previsioni estese fino a 1000ms. Questo approccio rappresenta una base solida su cui si fondano molte delle tecniche moderne per la HMP, permettendo una modellazione più accurata dei movimenti complessi. Inoltre, verrà fornita una panoramica dei dataset utilizzati in ambito HMP, descrivendone le caratteristiche principali e il loro ruolo nel supportare lo sviluppo e la valutazione dei modelli proposti.

1.2.1 Modelli per la Human Motion Prediction

Come introdotto in precedenza, il problema della HMP consiste nel prevedere i movimenti futuri di un essere umano a partire dai movimenti passati forniti come input. Questo processo avviene allenando i modelli di anticipazione sfruttando una grande quantità di dati relativi ai movimenti umani. In questo modo i modelli riescono ad acquisire una grande conoscenza relativa a come il corpo di una persona si muove. Questa *conoscenza* viene acquisita dalle

reti neurali attraverso una fase di addestramento, durante la quale i parametri del modello vengono modificati per adattarsi alla specifica task. Questo processo permette alla rete neurale di ottimizzare le sue prestazioni, trovando la combinazione di parametri che garantisce i risultati migliori. In altri termini, un modello predittivo produce previsioni che possono comportare un certo errore. Questo errore può essere rappresentato da una funzione matematica, nota come funzione di perdita, che presenta dei punti di minimo. L'obiettivo della fase di addestramento è ottimizzare il modello trovando la combinazione dei parametri che minimizza questa funzione di perdita, riducendo così l'errore delle previsioni del modello.
Generalmente, la fase di addestramento consiste nell'eseguire la predizione sui dati di input e, su questa predizione, calcolarne l'errore. Questa operazione viene svolta molteplici volte, dette epoche, e la modifica dei parametri avviene tramite calcolo del gradiente, il quale rappresenta la pendenza della funzione di perdita lungo la curva di errore data dal confronto tra output e target: il metodo è quello di modificare, ad ogni epoca, i parametri affinché l'output della previsione si avvicini al target.
Nella letteratura sono stati proposti molteplici modelli di anticipazione che si basano su reti neurali con architetture differenti.

Recurrent Neural Networks (RNNs)

Le reti neurali ricorrenti [13], dette RNN, come le Long Short-Term Memory (LSTM) o le Gated Recurrent Units (GRU), sono un tipo di rete neurale artificiale che utilizza dati sequenziali (sequence-to-sequence) o dati di serie temporali (come, ad esempio, dati di movimento umano). Questi algoritmi di DL sono progettati per gestire dati sequenziali o temporali, come dimostrano le loro applicazioni in ambiti come la traduzione linguistica e il riconoscimento vocale. In particolare, il loro utilizzo è particolarmente efficace nella previsione del movimento umano, un compito che coinvolge l'analisi e la modellazione di sequenze di pose nel tempo. Pertanto, le RNN si rivelano particolarmente adatte per la HMP, poiché possono catturare e prevedere dinamiche complesse in serie temporali di dati di movimento.
In fase di addestramento, le RNN sfruttano l'algoritmo di retro-propagazione nel tempo (backpropagation through time, BPTT) [28]: tecnica che consente la riduzione dell'errore, compiuto dalla rete neurale, ricalcolando i pesi del mo-

dello partendo da quelli già noti e dall'ultimo errore calcolato per ridurre lo stesso. Tuttavia, le RNN tendono solitamente a riscontrare due problemi, noti come gradienti che esplodono e gradienti che svaniscono. Questi problemi sono determinati dalla dimensione del gradiente. Quando il gradiente diventa troppo piccolo, diminuisce progressivamente fino a rendere gli aggiornamenti dei parametri di peso quasi nulli. Quando ciò accade, l'algoritmo smette di apprendere efficacemente. Il fenomeno dei gradienti che esplodono, invece, si verifica quando il gradiente è troppo grande, portando ad avere un modello instabile e aggiornamenti dei pesi non corretti. Questo avviene principalmente quando il modello deve gestire sequenze molto lunghe, dove i gradienti possono crescere esponenzialmente attraverso le molteplici iterazioni temporali. Una possibile soluzione a questo problema può essere quella di utilizzare un limitatore ai gradienti. In alternativa, efficace è l'uso delle skip connections, che saltano uno o più strati della rete, facilitando la propagazione dei gradienti e migliorando la stabilità e l'apprendimento di reti profonde.

Convolutional neural networks (CNNs)

Le reti neurali convoluzionali [14], dette Convolutional Neural Network (CNN), sono comunemente utilizzate per le attività di classificazione e di computer vision, mostrando eccellenti prestazioni con immagini, input vocali e segnali audio. Queste reti sono composte principalmente da tre tipi di layer:

- Layer convoluzionale: estrae caratteristiche basilari come bordi e texture
- Layer di pooling: riduce la dimensione dei dati
- Layer completamente connesso: aggrega le informazioni per la classificazione finale

Il layer convoluzionale è il primo livello della rete, mentre il layer completamente connesso è il livello finale. I layer convoluzionali possono essere seguiti da altri layer convoluzionali o di pooling, e ad ogni layer la complessità della CNN aumenta, così come la porzione dell'immagine che viene identificata.
Le CNN, inoltre, si rivelano utili anche nel campo della HMP. Possono essere adattate per lavorare con sequenze di pose umane trattate come "immagini" temporali, estraendo caratteristiche da ciascun fotogramma per fare previsioni sui movimenti futuri. Inoltre, sono efficaci nell'estrazione di caratteristiche dettagliate dalle pose, come la posizione dei giunti corporei, che possono poi essere usate in modelli di predizione temporale come le RNN. Le versioni

3D delle CNN, che elaborano dati spaziali e temporali, permettono un'analisi più complessa delle sequenze di pose 3D, migliorando le previsioni del movimento umano. Infine, esse possono integrarsi con altre tecniche per combinare dati provenienti da diverse fonti, come i sensori di movimento, offrendo una rappresentazione più completa del movimento.

Esempi di applicazione in ambito HMP

In letteratura, sono state proposte diverse tecniche innovative per la HMP che migliorano la previsione dei movimenti umani. Il paper "PVRED: A Position-Velocity Recurrent Encoder-Decoder for Human Motion Prediction" [9] introduce il modello PVRED, che integra informazioni sulla velocità delle pose e le relazioni temporali tramite una Position-Velocity RNN (PVRNN). Questo approccio migliora la modellazione dinamica dei movimenti e utilizza un layer di Quaternion Transformation (QT) per una rappresentazione più robusta delle pose. Il modello ha dimostrato risultati superiori rispetto agli approcci esistenti sia nella previsione a breve che a lungo termine, offrendo pose realistiche e naturali.
Oltre a Position-Velocity Recurrent Encoder-Decoder (PVRED), altri lavori hanno esplorato vari modelli basati sulle RNNs per HMP. Tra i vari, Y.Z. Lin et al. [15] fornisce una panoramica dettagliata delle tecniche di previsione del movimento umano. Un altro studio, di X. Song et al. [27], esplora l'uso delle RNNs in combinazione con le GANs, Generative Adversarial Networks, per migliorare la qualità delle previsioni. Inoltre, il lavoro di R. Yu et al. [33] utilizza unità LSTM per catturare pattern temporali complessi nei movimenti umani, mentre H. Liu et al. [16] presenta un modello RNN dedicato alla previsione dei movimenti umani, evidenziando le dinamiche temporali e le relazioni tra pose. Ognuno di questi studi contribuisce a migliorare la comprensione e le capacità nella previsione del movimento umano, affrontando vari aspetti e sfide del campo.
Inoltre, diversi studi hanno applicato architetture di rete neurale convoluzionale (CNN) per affrontare il problema della HMP. Il paper di S. Park et al. [21] propone un modello che utilizza le CNNs per estrarre caratteristiche spaziali dalle pose umane e successivamente applica tecniche di modellazione temporale per migliorare la previsione dei movimenti. Un altro studio, di J. Tang et al. [29], utilizza le CNNs per processare immagini di pose umane e combina questi dati con modelli di attenzione per affinare le previsioni del movimento. Questi ap-

procci dimostrano che le CNNs, tradizionalmente utilizzate per l'elaborazione delle immagini, possono essere efficaci anche nella previsione dei movimenti umani quando vengono adattate per gestire dati temporali e sequenziali.
In aggiunta ai modelli che utilizzano le RNN o le CNN, il paper "A Spatio-temporal Transformer for 3D Human Motion Prediction" [3] introduce un modello innovativo chiamato Spatio-temporal Transformer (ST-Transformer) per migliorare la previsione dei movimenti umani in 3D. Il modello affronta le limitazioni dei metodi esistenti, come RNN e CNN, utilizzando meccanismi di attenzione per catturare sia le relazioni spaziali tra le articolazioni sia le dinamiche temporali dei movimenti. Questo approccio consente una modellazione più precisa e completa dei movimenti umani. I risultati mostrano che il ST-Transformer supera le prestazioni dei metodi tradizionali, offrendo previsioni più accurate e realistiche.
Infine, nell'articolo "HumanMAC: Masked Motion Completion for Human Motion Prediction" [4], gli autori introducono un innovativo framework che si basa su un'architettura di diffusione che utilizza una strategia di completamento mascherato. In particolare, il metodo impiega un modello che inizialmente genera sequenze di movimento a partire da rumore casuale. Durante il calcolo della previsione, il framework applica un processo di denoising per affinare queste sequenze e prevedere i movimenti futuri sulla base delle sequenze di movimento osservate.

1.2.2 Modelli per il Natural Language Processing

L'utilizzo dei Large Language Model (LLM) sta rivoluzionando diversi settori, spaziando dalla traduzione automatica alla generazione di contenuti, fino al miglioramento dell'assistenza clienti e allo sviluppo di software. Tra le applicazioni più note, vi sono *Google BARD* [22] e *ChatGPT* [1] di OpenAI, esempi pratici di come i LLMs possano essere utilizzati per migliorare l'interazione tra uomo e macchina, rendendo più naturale e intuitiva la comunicazione con i sistemi digitali. Tuttavia, è importante ricordare che, sebbene queste applicazioni siano diffuse e conosciute, rappresentano solo una parte dell'evoluzione tecnologica in questo campo.
Una delle più recenti e avanzate tecniche di DL che ha rivoluzionato gli algoritmi di elaborazione del linguaggio naturale (NLP) è il Transformer [31]. Esso utilizza un particolare meccanismo di attenzione per selezionare attentamente

i token in un testo, ovvero l'entità minima che ne costituisce il testo stesso. Il Transformer pesa ogni token in base alla sua rilevanza rispetto al contesto, con lo scopo di imitare la capacità umana di concentrarsi sulle parole più importanti all'interno di una determinata frase.
L'aspetto fondamentale dei modelli di Transformer è l'uso della tecnica nota come self-attention: invece di utilizzare una rete neurale sequenziale (come la RNN) per codificare le informazioni, esso elabora tutte le parole dell'input in parallelo, attraverso una matrice di attenzione. Ciò consente al modello di acquisire una migliore comprensione del contesto del testo, rispetto ad altri modelli.
Numerosi modelli basati su Transformer sono stati sviluppati, ciascuno con caratteristiche specifiche, per migliorare le performance in diversi ambiti. Tra questi, Bidirectional Encoder Representations from Transformers (BERT) è uno dei modelli più influenti [6]. BERT introduce l'addestramento bidirezionale, che consente al modello di comprendere il contesto di una parola considerando sia il testo precedente che quello successivo, superando le limitazioni dei modelli unidirezionali, che considerano solo il testo precedente. Questa capacità di catturare il contesto bidirezionale ha reso BERT estremamente efficace in vari task di NLP, come il question answering e il sentiment analysis. Inoltre, esistono diverse varianti di BERT che sono state sviluppate per rispondere a esigenze specifiche. Ad esempio, *RoBERTa* [17] è una versione ottimizzata di BERT, che aumenta la quantità di dati di addestramento e prolunga il tempo di training per migliorare ulteriormente le performance. Un'altra variante è *DistilBERT* [26], che è una versione più leggera di BERT, progettata per essere più veloce e meno esigente in termini di risorse computazionali, pur mantenendo un alto livello di accuratezza. Un altro modello rilevante è *MiniLM* [32], una versione ridotta di BERT progettata per mantenere un buon livello di performance con un numero significativamente ridotto di parametri, ideale per applicazioni in cui le risorse computazionali sono limitate.
Oltre a BERT, esistono altre architetture rilevanti nell'ambito del NLP, come *LLAMA* (Large Language Model Meta AI) di Meta [30], progettato per essere scalabile e ottimizzato per diverse dimensioni di parametri, permettendo un'efficienza superiore in applicazioni su larga scala. Anche i modelli *GPT-2* [24] e *GPT-3* [1], sviluppati da OpenAI, hanno avuto un impatto significativo nell'espansione delle capacità dei LLMs. *GPT-2* ha dimostrato la capacità di generare testo di alta qualità, gestendo contesti complessi e completando frasi in modo coerente, mentre *GPT-3* ha ulteriormente ampliato queste capacità, migliorando

la generazione del linguaggio naturale, rendendolo uno dei modelli più avanzati e versatili per compiti come la scrittura creativa, la traduzione automatica, e la risposta a domande. Infine, *Kenneth Enevoldsen* [12] ha proposto un'ulteriore variante di BERT, ottimizzata per specifici task di NLP e caratterizzata da un'elevata efficienza nell'elaborazione e nel fine-tuning su dataset ridotti.

Questi modelli e le loro varianti sono di grande importanza poiché offrono avanzate capacità di comprensione e generazione del linguaggio. La loro continua evoluzione rappresenta uno dei principali pilastri dello stato dell'arte nel campo LLM.

1.2.3 Datasets

Nell'ambito della previsione del movimento umano (HMP), sono disponibili diversi dataset di grande importanza, ognuno dei quali contribuisce in modo significativo allo sviluppo e all'addestramento dei modelli di DL. Di seguito viene esposta una sintetica descrizione dei principali dataset disponibili:

- Archive of Motion Capture As Surface Shapes (AMASS): AMASS [18] è uno dei dataset più completi e ampi disponibili per la HMP, aggregando dati da diversi piccoli dataset di dati di pose catturate con sistemi di motion capture. Con oltre 60 ore di registrazioni da 500 soggetti differenti, AMASS standardizza le rappresentazioni utilizzando il modello SMPL, che descrive le pose umane attraverso le rotazioni di 24 punti di riferimento. Le registrazioni avvengono a diverse frequenze, variabili tra i 25 e i 120 frame per secondo (fps), a seconda del dataset originario. AMASS è fondamentale per lo studio delle dinamiche del movimento umano grazie alla sua vastità e alla diversità dei dati che contiene.

- Human3.6M (H36M): Questo dataset rappresenta un pilastro nel campo della HMP, essendo stato uno dei primi ad essere ampiamente utilizzato per l'addestramento di modelli di DL. Human3.6M [10] contiene dati di pose di 11 soggetti di entrambi i generi che eseguono 15 diverse attività comuni, come camminare, discutere e mangiare, con due sotto-azioni per ogni attività, per un totale di 30 registrazioni per soggetto. I dati sono catturati a 50 fps, e ogni posa è descritta attraverso le rotazioni angolari di 32 punti di riferimento. Sebbene sia stato uno standard per anni, le dimensioni relativamente ridotte del dataset e il numero limitato di attività

rappresentate possono introdurre un difetto nei modelli che ne limita la capacità di generalizzazione.

- CMU Motion Capture Database: Pubblicato dalla Carnegie Mellon University, il CMU Motion Capture Database [2] è uno dei dataset più utilizzati e completi per la HMP. Comprende dati di pose di 144 soggetti, registrati utilizzando 41 punti di riferimento per catturare una vasta gamma di movimenti. Le registrazioni includono attività comuni come sport, camminate e azioni specifiche come dirigere il traffico. Questo dataset è particolarmente utile per la varietà di movimenti catturati, rendendolo uno strumento prezioso per l'addestramento di modelli che necessitano di comprendere una gamma diversificata di dinamiche corporee.

- Bodies, Action and Behavior with English Labels (BABEL): BABEL [23] è un dataset recente e innovativo che si distingue per la sua annotazione dettagliata delle azioni umane in combinazione con dati riguardanti le pose. BABEL si integra direttamente con AMASS, offrendo descrizioni semantiche delle azioni eseguite nelle sequenze di movimento. Questo lo rende particolarmente utile per studi che richiedono una comprensione più profonda del contesto delle azioni, oltre alla semplice previsione delle pose. Esso, quindi, favorisce lo sviluppo di modelli che richiedono una comprensione sia dei movimenti che del linguaggio, rendendolo uno strumento prezioso per la ricerca che combina dati di diverso genere. Grazie alla sua ricchezza di dati annotati, BABEL può essere considerato uno dei dataset di riferimento per chi lavora sull'interazione tra movimento e semantica nel contesto della HMP.

In sintesi, i dataset disponibili per HMP offrono una vasta gamma di risorse per migliorare la comprensione e la previsione dei movimenti umani. AMASS e BABEL, in particolare, rappresentano risorse fondamentali, offrendo dati ampi e vari che supportano avanzamenti significativi nella modellazione e nella previsione del movimento umano. La disponibilità di questi dataset, inoltre, contribuisce in modo sostanziale allo sviluppo di modelli in campo HMP.

2

Approcci Deep Learning per HMP

Questo capitolo esplora la rappresentazione dello scheletro umano in una forma compatibile con le tecniche di DL utilizzate per la previsione del movimento umano (HMP). Vengono, poi, discussi i modelli di DL impiegati negli esperimenti. In particolare, viene analizzata l'architettura di una rete neurale ricorrente (RNN), riconosciuta per la sua capacità di gestire sequenze temporali. Successivamente, viene presentato il modello Position-Velocity Recurrent Encoder-Decoder (PVRED), che si basa su una RNN e combina informazioni sulla velocità delle pose con le dinamiche temporali per ottimizzare la previsione del movimento. Questi modelli rappresentano le architetture all'avanguardia nel campo della HMP e saranno descritti nel corso del capitolo.

2.1 Rappresentazione del corpo umano

Esistono diversi dataset per la riproduzione dello scheletro umano ed ognuno di essi adotta un approccio diverso. Questa diversità risiede su due caratteristiche: il numero di keypoints ed il tipo di cinematica.
Come già anticipato nel paragrafo 1.2.3, in questo progetto si è fatto riferimento al dataset di AMASS [18]. La scelta di quest'ultimo è stata dettata dal fatto che AMASS accorpa numerosi dataset standardizzando la loro rappresentazione in un formato comune. Questo rende AMASS un dataset molto ricco, sia in termini di quantità di dati, sia in termini di variabilità dei movimenti del corpo umano. La rappresentazione utilizzata in AMASS è il modello SMPL il quale fornisce una parametrizzazione scheletrica standard che include sia la struttura dello

scheletro che la superficie del corpo.
Il progetto si concentra unicamente sulla rappresentazione scheletrica, poiché il problema di HMP riguarda la previsione del movimento dello scheletro stesso, senza la necessità di considerare la mesh 3D del corpo. Tutti i modelli contenuti in AMASS, sono rappresentati da un insieme di 24 keypoints identificanti i giunti e le terminazioni, e sono connessi tra di loro secondo il principio di cinematica diretta: la posizione e l'orientamento di ogni terna di un qualunque giunto, o end effector, sono determinati a partire dalla terna del keypoint base.
Viene riportato in Figura 2.1 un esempio di rappresentazione scheletrica, con relativo elenco dei giunti.

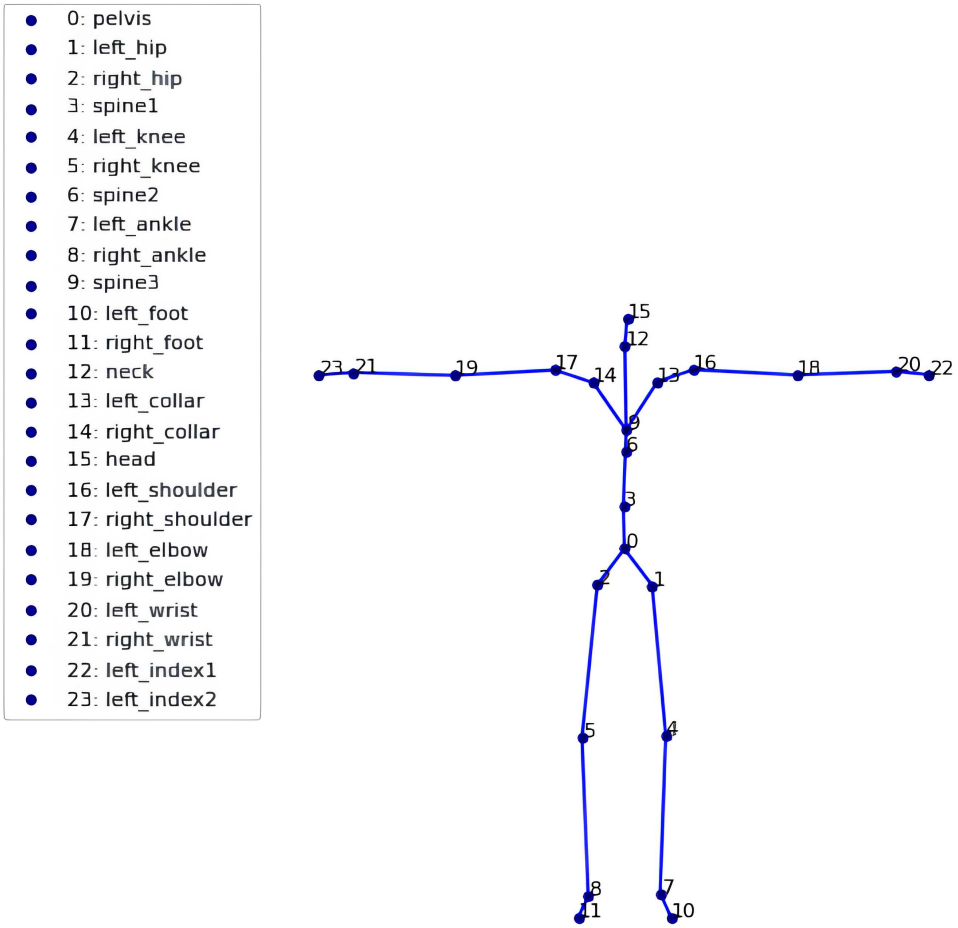

Figura 2.1: Rappresentazione scheletrica SMPL

Questa rappresentazione viene utilizzata anche per la visualizzazione delle pose, come negli esempi riportati di seguito:

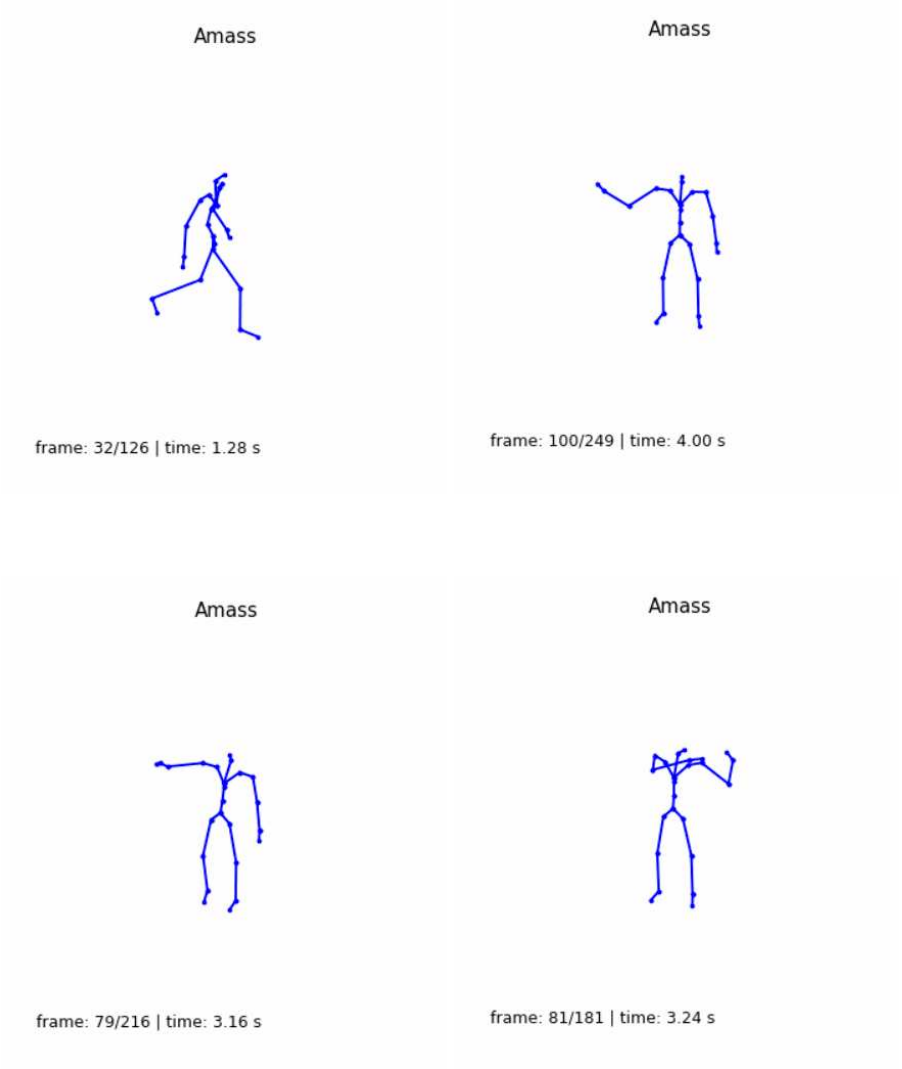

Figura 2.3: Esempi rappresentazione umana: SMPL

Le sequenze mostrate nelle Figure 2.3 sono visibili in [8].
Si può inoltre notare, dalla Figura 2.1, che il keypoint di riferimento coincide con il bacino; ciò è comune nei sistemi di motion capture in quanto tale rappresentazione presenta una serie di vantaggi:

- Stabilità e centralità: il bacino è una parte centrale e relativamente stabile del corpo umano; si trova molto vicino al suo centro di massa e non ha un'elevata possibilità di muoversi in modo indipendente e con grande ampiezza (come invece lo hanno gli arti).

- Riferimento coerente: usare il bacino come punto di riferimento principale consente di ottenere una rappresentazione uniforme e standardizzata

dei movimenti del corpo. Questo è particolarmente utile per integrare e confrontare dati provenienti da più soggetti e da diversi sistemi di motion capture, poiché il bacino funge da punto di ancoraggio comune che facilita la comparabilità delle pose.

- Rappresentazione delle pose globali: in molte tecniche di motion capture, il bacino è utilizzato per rappresentare la posizione globale del corpo nello spazio, ovvero è possibile descrivere il movimento del bacino rispetto ad un altro sistema di riferimento, e descrivere il movimento relativo alle altre parti del corpo rispetto al bacino.
- Facilità di calcolo e modellazione: in termini di onerosità computazionale, avere un singolo punto di riferimento come il bacino semplifica i calcoli per le trasformazioni e le rotazioni del corpo. Comune, infatti, è l'utilizzo di matrici di rotazione per determinare la posizione degli altri keypoints.

Ogni clip di movimento è rappresentata da una sequenza di pose, dette frame, ognuna delle quali è rappresentata da una matrice di dimensioni 24×3, in cui ciascuna delle 24 righe contiene una terna di valori che descrivono le rotazioni del rispettivo keypoint. Le terne di valori (α, β, γ) rappresentano le rotazioni attorno agli assi principali del sistema di coordinate locale del keypoint. Rotazioni, che sono espresse in termini di angoli di Eulero o quaternioni, a seconda delle specifiche del dataset e delle esigenze dell'applicazione. I 24 keypoints sono rappresentativi delle articolazioni principali, dove ogni keypoint è mappato ad una specifica riga della matrice, il cui indice corrisponde all'id assegnato nella Figura 2.1. Secondo questa rappresentazione, inoltre, deve valere il vincolo per cui i link di connessione dei keypoints hanno lunghezza fissa.
La rappresentazione matriciale facilita il processamento e l'analisi dei dati di movimento utilizzando algoritmi di ML e DL, ed è compatibile con numerosi strumenti software per l'animazione.
Una sequenza di pose è una serie temporale di frame consecutivi, ognuno rappresentato da una matrice come descritto precedentemente. In altre parole, tale sequenza è un insieme di frame, dove ognuno di essi corrisponde a una posa specifica rappresentando, quindi, un'istantanea della configurazione scheletrica del corpo in un determinato istante. Questa sequenza cattura l'evoluzione del movimento umano nel tempo e permette di modellare dinamiche e transizioni tra pose diverse. In particolare, essa è definita dalla frequenza di campionamento e dalla durata complessiva della registrazione. Per esempio, se un dataset

ha una frequenza di campionamento di 30 frame al secondo (fps) e registra un movimento per 10 secondi, ogni sequenza di pose avrà 300 frame.
Le sequenze di pose sono cruciali per la previsione del movimento, poiché consentono ai modelli di apprendere e generalizzare le transizioni temporali tra pose, migliorando la capacità di prevedere movimenti futuri basandosi su quelli passati. La rappresentazione e la gestione delle sequenze di pose richiedono algoritmi di elaborazione di dati temporali per trattare efficacemente la variabilità e la dinamicità del movimento umano.

2.2 Reti neurali ricorrenti (RNN)

Per la parte sperimentale di questa tesi è stato scelto di implementare un modello che sfrutta una RNN. Una rete neurale ricorrente (RNN) [13] è un modello di DL addestrato per elaborare e convertire un input di dati sequenziali in un output di dati sequenziali specifico. Essi sono dati come parole, frasi o serie temporali, ed i componenti sequenziali sono correlati in base a regole complesse. Nel caso in questione, tale comportamento si traduce nel convertire la sequenza di frame, detta mocap, ricevuta come input, in una sequenza di frame rappresentanti l'immediato futuro di quel mocap.
Una RNN è un modello costituito da molti componenti interconnessi. Anch'essa, infatti, è costituita da neuroni, ovvero nodi di elaborazione dati che collaborano per eseguire attività complesse.
In generale, una RNN, è costituita da vari layer, e più elevato è il numero di questi, più essa è profonda. Di questi layer, solitamente, si distinguono il primo e l'ultimo, detti rispettivamente di input e di output, da tutti gli altri, che sono layer nascosti, detti *hidden layers*. Il termine *ricorrente* si riferisce al fatto che l'output di ogni cella viene riutilizzato come input della stessa per la successiva iterazione. Questa metodologia è stata creata per conferire alla rete neurale una memoria interna, per mezzo della quale essa utilizza informazioni catturate negli step precedenti, consentendo al modello di apprendere le dipendenze lungo la sequenza e gestire input di lunghezza variabile.
Come riportato nella Figura 2.4, ogni cella riceve in input il dato allo step corrente, unito all'informazione dello stato della cella precedente nella sequenza.

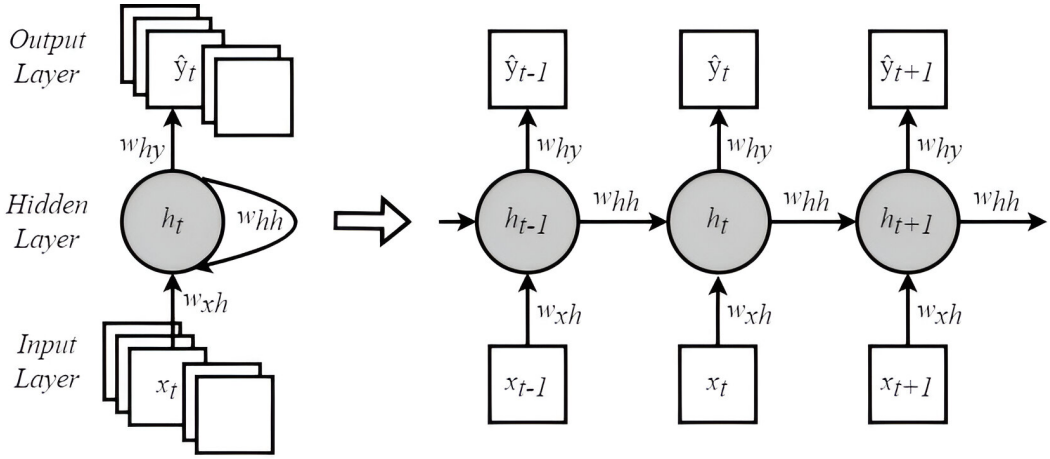

Figura 2.4: Architettura RNN

Ad ogni istante temporale t, la cella riceve in input un vettore $\mathbf{x}_t$ della sequenza $\mathbf{X}$, di dimensione N, produce un output $\hat{\mathbf{y}}_t$ ed aggiorna lo stato nascosto $\mathbf{h}_t$, detto hidden state. Per ognuno di questi step, la cella condivide gli stessi parametri; ciò significa che il set dei parametri, rappresentato dai vettori $\mathbf{w}_{xh}$, $\mathbf{w}_{hh}$ e $\mathbf{w}_{hy}$, è usato in modo consistente, dove: $\mathbf{w}_{xh}$ sono i pesi che connettono l'input $\mathbf{x}_t$ con il vettore dell'hidden state $\mathbf{h}_t$; $\mathbf{w}_{hh}$ contiene i pesi di connessione tra l'hidden state attuale e quello successivo; $\mathbf{w}_{hy}$ sono i pesi che connettono l'hidden state con l'output $\hat{\mathbf{y}}_t$. Questa condivisione dei parametri consente alla RNN di cogliere in modo efficace le dipendenze temporali e di elaborare i dati sequenziali in modo più efficiente, trattenendo le informazioni degli input precedenti nel suo hidden state corrente. Il fulcro di questa architettura, sta nell'hidden state, il quale viene aggiornato ad ogni passo con una funzione non lineare che ha per input l'hidden state allo step precedente $\mathbf{h}_{t-1}$ ed il vettore di input attuale $\mathbf{x}_t$:

$$\mathbf{h}_t = f(\mathbf{w}_{hh}\mathbf{h}_{t-1}, \mathbf{x}_t) \tag{2.1}$$

Analogamente, l'output di previsione è ottenuto tramite una seconda funzione non lineare che ha per input l'hidden state allo step corrente $\mathbf{h}_t$:

$$\hat{\mathbf{y}}_t = g(\mathbf{w}_{hy}\mathbf{h}_t + \mathbf{b}) \tag{2.2}$$

dove $\mathbf{b}$ è il vettore di bias per il vettore di output.
Ogni vettore di output spesso può passare attraverso un layer lineare al fine di variarne la sua dimensione adattandola alle esigenze del problema. L'insieme

di tutti i vettori di output $\hat{\mathbf{y}}_t$ andrà, poi, a costruire la sequenza prevista $\hat{\mathbf{Y}}$, di dimensione n.
In fase di addestramento, per determinare se la sequenza ottenuta è fedele al target previsto, detto Ground Truth (GT), si esegue il calcolo dell'errore: il calcolo spesso utilizzato è quello dell'errore quadratico medio, Mean Squared Error (MSE), definito come segue:

$$\mathrm{L} = \frac{1}{n} \sum_{i=1}^{n} \left\| \hat{\mathbf{y}}_i - \mathbf{y}_i \right\| \tag{2.3}$$

dove $\mathbf{y}$ è il vettore della GT.
Tale relazione permette di calcolare il gradiente della curva di errore per, così, eseguire la retro-propagazione dell'errore e modificare i set di parametri $\mathbf{w}_{xh}$, $\mathbf{w}_{hh}$ e $\mathbf{w}_{hy}$.

Le RNN utilizzano diverse tipologie di celle, ognuna con caratteristiche specifiche per affrontare le sfide legate alla cattura di dipendenze temporali e alla gestione del problema del gradiente che scompare o che esplode. Le principali tipologie di celle RNN, che si evolvono rispetto alle RNN standard (ovvero quella con input, hidden state e output), sono: LSTM e GRU.
La cella LSTM [20] ha una struttura più complessa in quanto possiede una memoria aggiuntiva che le permette di scegliere se mantenere o rimuovere una informazione passata. Ciò si traduce nell'affiancare all'hidden state un secondo vettore: lo stato di cella, detto cell state. A differenza del vettore di hidden state $\mathbf{h}$, che rappresenta la "memoria" attuale della rete, il vettore di cell state $\mathbf{c}$ è responsabile dell'immagazzinamento di informazioni a lungo termine durante la sequenza. Grazie al cell state, le LSTM modellano efficacemente le dipendenze a lungo termine nei dati sequenziali, sono robuste ai dati rumorosi e sono versatili per vari compiti. Tuttavia, sono computazionalmente costose da addestrare, inclini all'overfitting su dataset piccoli e richiedono un'accurata ottimizzazione degli iperparametri.
Similmente alla cella LSTM, quella GRU [5] è progettata per modellare dati sequenziali permettendo di conservare e scartare selettivamente informazioni lungo la sequenza. In particolare, essa ha un'architettura più semplice, con meno parametri, il che può renderla più facile da addestrare e più efficiente dal punto di vista computazionale.
Similmente alle altre reti neurali ricorrenti, la GRU elabora sequenze di dati considerando un elemento alla volta, aggiornando il suo hidden state in base

all'input corrente e allo stato precedente. Tuttavia, a differenza di una RNN standard, la GRU è progettata per gestire meglio le informazioni rilevanti attraverso sequenze più lunghe. Esegue tale operazione regolando automaticamente quanto delle informazioni passate e di quelle attuali deve essere conservato o aggiornato. Questo permette alla GRU di ridurre il problema della perdita di memoria a lungo termine, tipico delle RNN standard, consentendo una gestione più efficiente delle variazioni nei dati sequenziali.
Nonostante i loro vantaggi, le reti GRU potrebbero non eguagliare le LSTM in compiti che richiedono la modellazione di dipendenze a lungo termine molto complesse; possono essere più inclini all'overfitting, specialmente su dataset piccoli, e richiedono una ottimizzazione attenta degli iperparametri per ottenere buone prestazioni.
Queste architetture hanno dimostrato di essere particolarmente efficaci nel catturare dipendenze temporali grazie alle loro capacità di gestire dinamiche complesse nei dati, rappresentando una soluzione potente per molti problemi. Ciononostante, le reti neurali ricorrenti presentano generalmente una limitata interpretabilità. Questo è dovuto alla complessità dei loro meccanismi interni, che rende difficile comprendere esattamente come vengono fatte le previsioni.

L'approccio standard per applicazioni sequence-to-sequence, ad esempio la traduzione linguistica o dati di serie temporali, è quello di utilizzare l'architettura RNN con struttura encoder-decoder. Tale struttura può essere schematizzata come nella Figura 2.5:

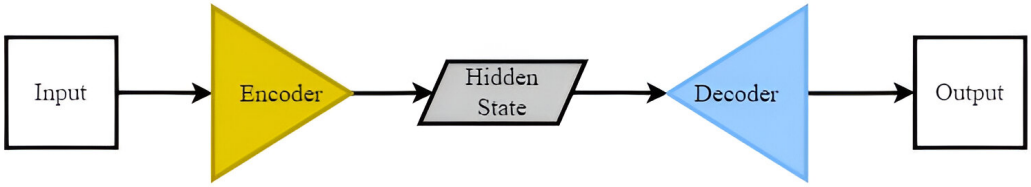

Figura 2.5: Schema encoder-decoder

L'encoder è responsabile della trasformazione delle sequenze di input in una rappresentazione interna che cattura le informazioni essenziali per la previsione futura. L'encoder utilizza i meccanismi ricorrenti, spiegati precedentemente in questo paragrafo (2.2), per elaborare le informazioni temporali. In questo caso, però, viene ignorato l'output, in quanto l'encoder è unicamente adibito ad estrapolare le informazioni contenute nella sequenza di input e non a creare la

previsione.
Il decoder si occupa di generare la sequenza di output sulla base della rappresentazione interna prodotta dall'encoder. Esso è progettato per gestire la previsione a lungo termine e per sintetizzare informazioni in modo che riflettano accuratamente le dinamiche temporali osservate.

Uno schema di questo tipo, è meglio rappresentato in Figura 2.6:

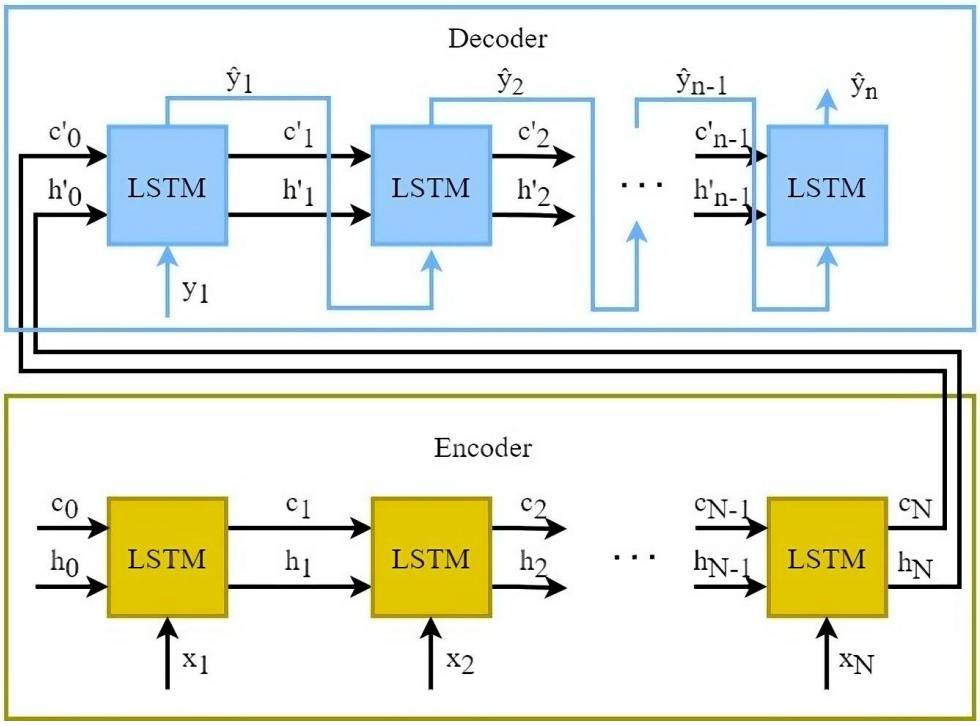

Figura 2.6: Encoder-Decoder con LSTM

In questa immagine viene rappresentata l'architettura Encoder-Decoder basata su LSTM: l'Encoder è costituito da una cella che esegue N iterazioni, dove N è la lunghezza della sequenza di input, della quale viene ignorato l'output. La cella del Decoder, invece, esegue tante iterazioni quant'è la lunghezza di output e riceve l'embedding delle informazioni catturate dall'Encoder; inoltre, generalmente, nella struttura Decoder viene mantenuto l'output dell'iterazione corrente per essere dato in input all'iterazione successiva. Infine, per semplicità di rappresentazione, questa immagine mostra una struttura a singolo strato, detto *layer*, ma è possibile aumentarne il numero di strati, ognuno dei quali riceve come input l'output dello strato inferiore (pratica molto comune nei sistemi di DL).

2.3 Position-Velocity Recurrent Encoder-Decoder

Nel caso in cui i test condotti su una RNN base diano risultati promettenti, la speranza è quella di implementare successivamente la metodologia proposta sul modello PVRED. Modello il quale è rmappresentativo degli approcci basati su RNN per la HMP, che sono tra i più utilizzati nel campo.
Gli autori di Position-Velocity Recurrent Encoder-Decoder (PVRED) [9] propongono un nuovo modello basato su una struttura RNN per il problema di anticipazione del movimento umano. Il modello PVRED si distingue per la sua capacità di integrare, oltre all'informazione delle pose passate, altre due informazioni che si esprimono sia in termini di posizione che di velocità, offrendo miglioramenti significativi rispetto agli approcci tradizionali.
Le strutture esistenti per la modellazione dei dati sequenziali, come le LSTM e le GRU, sebbene potenti, spesso mostrano limitazioni quando si tratta di gestire dinamiche complesse e variazioni nella velocità delle sequenze temporali. PVRED è progettato per affrontare questi problemi introducendo una rappresentazione delle informazioni di posizione e velocità, al fine di migliorare la capacità di previsione e la qualità della generazione dei dati. Questo avviene utilizzando un'architettura RNN basato su una struttura Encoder-Decoder.

Una delle innovazioni chiave di PVRED è l'integrazione delle informazioni della posizione della posa all'interno della sequenza, detto anche embedding di posizione, e della velocità dei movimenti attraverso una modulazione dei segnali ricorrenti. Questo approccio consente al modello di adattarsi meglio alle variazioni nella velocità dei dati temporali e di migliorare la previsione in scenari complessi.

Embedding di posizione

L'uso dell'embedding di posizione, tipicamente impiegato nell'elaborazione del linguaggio naturale, può essere adattato ai dati temporali posizionali per migliorare la capacità del modello di catturare le sfumature dei movimenti. In particolare, aiuta il modello a differenziare meglio le sequenze di pose simili, riducendo la possibilità che converga verso pose errate o meno probabili durante la previsione. L'embedding di posizione permette di codificare l'informazione relativa a quale istante temporale una certa posa fa riferimento all'interno della sequenza. Questa informazione viene rappresentata con un vettore e permette

al modello di distinguere pose simili che avvengono in momenti diversi. A tale scopo, vengono utilizzate funzioni seno e coseno per codificare le posizioni relative o assolute, come espresso nelle equazioni 2.4.1.

$$\begin{aligned} p_t(2i) &= \sin\left(\frac{t}{10000^{2i/d^p}}\right), \\ p_t(2i-1) &= \cos\left(\frac{t}{10000^{2i/d^p}}\right), \end{aligned} \tag{2.4.1}$$

dove p_t è l'embedding di posizione all'istante $t \in \{1, ..., N+n\}$, d^p indica la dimensione dell'embedding e i rappresenta l'indice del vettore dell'embedding di posizione.
Questa tecnica permette al modello di apprendere a come prestare attenzione alle posizioni relative e prevedere pose dall'aspetto naturale a vari intervalli di tempo, per sequenze di qualsiasi lunghezza. Per ottenere una buona previsione, infatti, risulta necessario riconoscere in quale modo avviene la sequenzialità degli eventi, in particolar modo quando si hanno pose apparentemente simili, come, ad esempio, camminare in avanti o indietro.

Embedding di velocità

Il terzo input previsto da PVRED, oltre alla posa umana ed all'embedding di posizione, è la velocità. Se l'embedding di posizione aiuta a differenziare le pose simili a diversi istanti temporali, la velocità delle pose è utilizzata per mantenere la continuità del movimento. Ad ogni istante di tempo t, viene calcolata la derivata temporale della posa umana $\dot{\mathbf{x}}_t$, la quale viene poi aggiunta alla posa precedente. Integrare la velocità migliora la previsione del movimento umano preservando la continuità e aumentando la precisione delle previsioni. La velocità cattura le dinamiche del movimento e aiuta a ridurre il problema della previsione di pose imprecise che potrebbe verificarsi se il modello non ha informazioni sufficienti su come queste cambiano nel tempo, permettendo, così, di modellare meglio le transizioni tra diverse azioni. Questo approccio rende le previsioni più fluide, accurate e realistiche, migliorando le prestazioni complessive del modello.

Integrazione di posizione e velocità

Ricevuto l'input della posa umana $\mathbf{x}_t$, allo step temporale t, la cella GRU dell'Encoder calcola la velocità della posa $\dot{\mathbf{x}}_t$ e l'embedding di posizione $\mathbf{p}_t$. La formulazione matematica delle operazioni compiute dalla cella, a un dato step temporale t, è espressa dalle relazioni 2.4.2.

$$\begin{aligned}
\mathbf{z}_t &= \sigma\left(\mathbf{U}_x^z\mathbf{x}_t + \mathbf{U}_v^z\dot{\mathbf{x}}_t + \mathbf{U}_p^z\mathbf{p}_t + \mathbf{W}^z\mathbf{h}_{t-1}\right), \\
\mathbf{r}_t &= \sigma\left(\mathbf{U}_x^r\mathbf{x}_t + \mathbf{U}_v^r\dot{\mathbf{x}}_t + \mathbf{U}_p^r\mathbf{p}_t + \mathbf{W}^r\mathbf{h}_{t-1}\right), \\
\mathbf{h}'_t &= \tanh\left(\mathbf{U}_x^h\mathbf{x}_t + \mathbf{U}_v^h\dot{\mathbf{x}}_t + \mathbf{U}_p^h\mathbf{p}_t + \mathbf{W}^h\left(\mathbf{r}_t \circ \mathbf{h}_{t-1}\right)\right), \\
\mathbf{h}_t &= (1 - \mathbf{z}_t) \circ \mathbf{h}_{t-1} + \mathbf{z}_t \circ \mathbf{h}'_t
\end{aligned} \tag{2.4.2}$$

dove $\mathbf{r}_t$ rappresenta il filtro ricorrente, $\mathbf{z}_t$ il filtro di aggiornamento, $\mathbf{h}_t$ l'hidden state, e $\mathbf{U}, \mathbf{W}$ sono rispettivamente le matrici delle variabili e dei pesi. La stessa formulazione matematica si applica anche all'hidden state del Decoder. Per prevedere la futura posa umana allo step temporale $t' \in \{1, ..., n\}$, si prevede prima la velocità temporale discreta e poi la si aggiunge alla posa dello step temporale precedente, come illustrato nelle equazioni 2.4.3.

$$\begin{aligned}
\dot{\mathbf{x}}_{N+t'-1} &= \mathbf{W}\mathbf{h}_{N+t'-1} + \mathbf{b}, \\
\mathbf{x}_{N+t'} &= \mathbf{x}_{N+t'-1} + \dot{\mathbf{x}}_{N+t'-1}
\end{aligned} \tag{2.4.3}$$

dove $\mathbf{W}$ e $\mathbf{b}$ rappresentano rispettivamente i pesi e i parametri di bias della rete. Questa operazione viene eseguita da un layer lineare sovrapposto ad ogni cella GRU.

Trasformazione di Quaternioni e funzione di perdita

Durante l'addestramento, il modello PVRED utilizza tecniche di ottimizzazione avanzate per garantire l'efficienza e la robustezza del processo. Per la previsione del movimento umano, le pose sono descritte principalmente dalle rotazioni articolari. Per questo motivo, PVRED integra un layer di Trasformazione di Quaternioni, la QT, che converte le rotazioni dalla notazione angolo-asse a quella dei quaternioni. Questa metodologia permette di eliminare singolarità e discontinuità, che si verificano, invece, con l'utilizzo di mappe esponenziali o degli angoli di Eulero. Inoltre, l'operatore di moltiplicazione nello spazio dei quaternioni è equivalente alla moltiplicazione di matrici di rotazione. Utilizzan-

do questa metodologia, quindi, il modello può eseguire operazioni di rotazione senza accumulare errori significativi, migliorando così la precisione delle previsioni sui movimenti. Tuttavia, i quaternioni richiedono la normalizzazione per mantenere le proprietà matematiche.
Data una rappresentazione di un vettore tridimensionale $\mathbf{e}$, la trasformazione in una rappresentazione quaternione a quattro dimensioni $\mathbf{q}$ eseguita dal layer QT è rappresentata dall'equazione 2.4.4.

$$\mathbf{q}(i) = \begin{cases} \cos\left(0.5\|\mathbf{e}\|^2\right) & i = 1 \\ \frac{\sin\left(0.5\|\mathbf{e}\|^2\right)}{\|\mathbf{e}\|^2} \cdot \mathbf{e}^{(i-1)} & i \geq 2 \end{cases} \tag{2.4.4}$$

dove i rappresenta l'i-ésimo elemento di $\mathbf{q}$.
Il calcolo della funzione di perdita utilizza la relazione MSE, con la differenza che non vi è la variazione tra gli effettivi vettori, ma tra le trasformate in quaternioni di questi. Il calcolo della loss è descritto nell'equazione 2.4.5.

$$\mathrm{L} = \frac{1}{n} \sum_{i=1}^{n} \left\| g(\hat{\mathbf{y}}_i) - g(\mathbf{y}_i) \right\| \tag{2.4.5}$$

dove g rappresenta la trasformazione dei quaternioni QT.

In Figura 2.7 è rappresentata l'architettura del modello PVRED riportata nel lavoro originale [9].

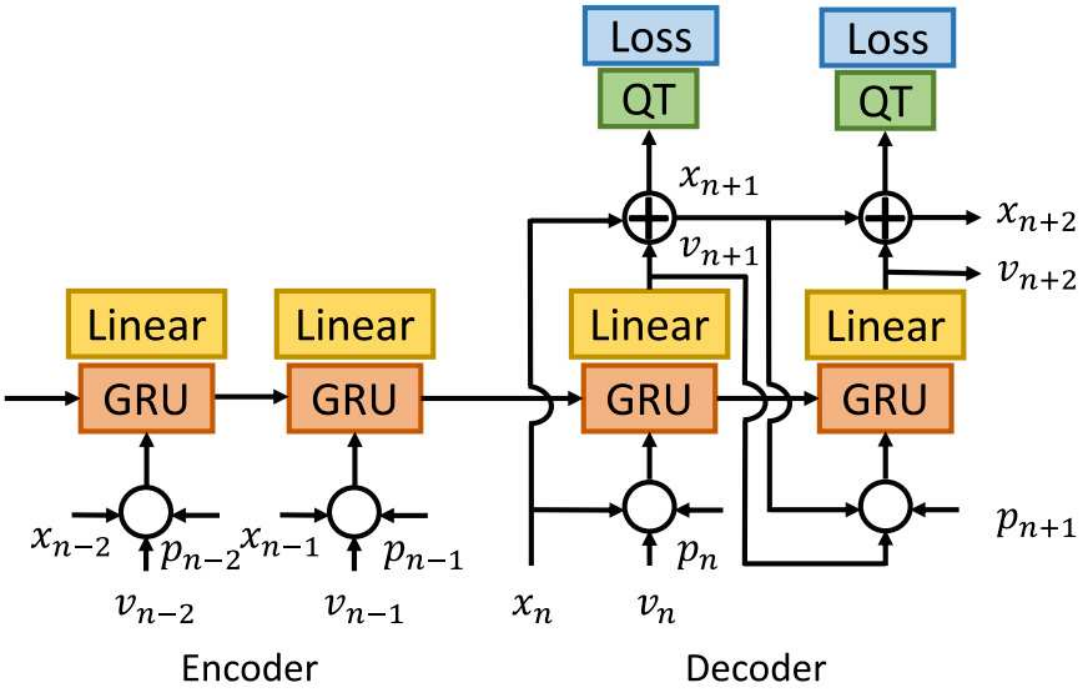

Figura 2.7: Architettura PVRED

Il modello ha dimostrato di essere efficiente dal punto di vista computazionale ed inoltre rappresenta un avanzamento significativo nella modellazione delle sequenze temporali, grazie alla sua capacità di integrare informazioni di posizione e velocità. Questo approccio migliora la qualità della previsione e la capacità di generazione dei dati, rendendolo un'opzione promettente per applicazioni avanzate nella previsione dei movimenti.

3

Approcci Deep Learning per LLM

Recentemente, i Large Language Model (LLM) hanno mostrato miglioramenti significativi nel trattamento del linguaggio naturale. Questo è dovuto alla loro capacità di modellare e apprendere dalle complesse interazioni semantiche e sintattiche nei dati testuali, diventando approcci standard per il NLP. Questa capacità deriva dalla struttura matematica complessa dei modelli,composto da numerosi layer all'interno di una rete neurale profonda, perciò detta Deep Neural Network (DNN), e un gran numero di parametri.
Tuttavia, la complessità di questi modelli comporta anche limitazioni: l'addestramento richiede enormi quantità di dati, di tempo e di potenza di calcolo. Per affrontare queste sfide, sono stati sviluppati approcci come l'uso di modelli preaddestrati, che vengono affinati con dati specifici, risparmiando tempo e risorse. Inoltre, l'ottimizzazione degli algoritmi e la creazione di strutture computazionali più efficienti possono ridurre il tempo di addestramento e aumentare sensibilmente l'uso di risorse come la memoria e la potenza di calcolo.

Questo capitolo descrive quali tecnologie LLMs sono state utilizzate per integrare la semantica nei modelli di anticipazione del movimento, evidenziando i loro principali vantaggi e critiche. In particolare, sono descritte le loro caratteristiche principali, mettendo in risalto le differenze rispetto agli approcci da loro utilizzati.
Alla base di questi LLMs vi è l'architettura dei Transformers [31], una rete neurale progettata per trasformare una sequenza di input in una sequenza di output. I Transformers apprendono il contesto e tracciano le relazioni tra i componenti della sequenza, permettendo ai modelli di linguaggio di gestire complesse rela-

zioni semantiche tra le parole e di generare output accurati e contestualizzati.
Gli iniziali modelli di ML utilizzavano tecniche dove veniva mappata la frequenza delle relazioni tra coppie o gruppi di parole nel loro set di dati di addestramento, cercando di indovinare la parola successiva. Tuttavia, queste tecnologie non erano in grado di mantenere il contesto oltre una certa lunghezza di input.
Con lo studio e l'approfondimento in ambito NLP, si sono sviluppati modelli di DL sempre più avanzati che mirano a far sì che i dispositivi intelligenti comprendessero e rispondessero al linguaggio umano naturale. Tra questi, l'utilizzo dei Transformers, che elaborano sequenze lunghe nella loro interezza con il calcolo parallelo, ha consentito la formazione di LLMs in grado di apprendere rappresentazioni linguistiche complesse, come il modello BERT.

3.1 BERT: Transformers bidirezionali preaddestrati

Bidirectional Encoder Representations from Transformers (BERT) [6] rappresenta uno dei più significativi avanzamenti in ambito NLP. Il modello è stato sviluppato da *Google AI Language* e ha rivoluzionato il modo in cui i modelli di DL comprendono e generano il linguaggio umano.
BERT è costruito sull'architettura dei Transformers, specificamente progettata per catturare il contesto bidirezionale delle parole in una sequenza. A differenza dei modelli precedenti, che elaboravano il testo in una sola direzione (da sinistra a destra o viceversa), BERT considera simultaneamente il contesto delle parole sia a sinistra che a destra del token in input, permettendo una comprensione più accurata delle sfumature linguistiche.
L'architettura di BERT è composta da una rete bidirezionale a più layer di Encoder, dove ogni layer applica meccanismi di calcolo dell'attenzione per identificare le relazioni tra le parole all'interno della frase. Questo modello utilizza un approccio di preaddestramento basato su due compiti specifici: il Masked Language Modeling (MLM) e il Next Sentence Prediction (NSP). Il MLM consiste nel mascherare una certa percentuale di token di input e nel far predire al modello questi token mascherati basandosi sul loro contesto circostante. Questo approccio impedisce la creazione di connessioni errate tra parole, evitando, così, loop infiniti, e migliora la capacità del modello di generalizzare su dati non visti. Il NSP, invece, aiuta il modello a comprendere la relazione logica tra frasi consecutive, migliorando ulteriormente la coerenza contestuale.

BERT è preaddestrato su un ampio corpus di testi, contenente miliardi di parole, e può essere ottimizzato su vari compiti specifici di NLP con un numero relativamente ridotto di dati. In questo modo, si riducono significativamente il tempo e le risorse necessarie per addestrare modelli altamente performanti. Questa flessibilità lo rende estremamente utile per applicazioni come la classificazione di testi, la sentiment analysis, e la risposta a domande.
In particolare, la dimensione del vettore di embedding utilizzata nel modello BERT base è di 768, con una lunghezza massima di input di 512 token, fattori che lo rendono capace di gestire sequenze lunghe e complesse. La versione BERT Large, più potente, ha un embedding di 1024 elementi e utilizza 24 layer di encoder, raddoppiando i 12 del modello base. In Figura 3.1 è rappresentata l'architettura semplificata del modello riportato nel lavoro originale [6].

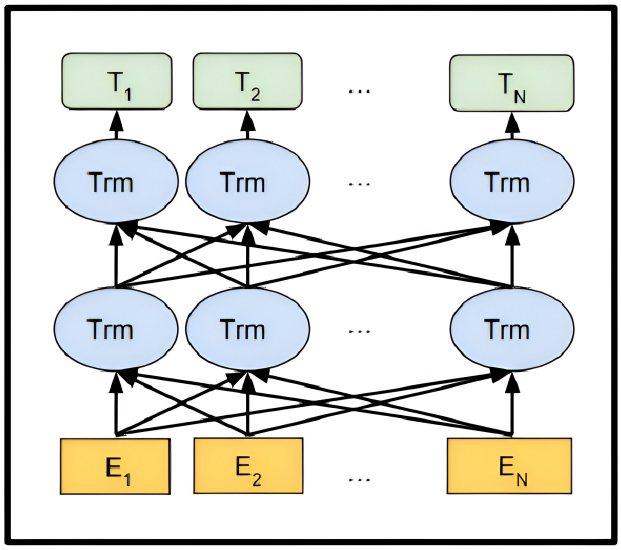

Figura 3.1: Schema della struttura di BERT

In sintesi, il modello acquisisce ogni parola avente la sua rappresentazione, denominata più propriamente embedding. Ogni layer del modello esegue un calcolo dell'attenzione sull'embedding prodotto dal layer precedente, generando così un nuovo embedding intermedio che mantiene le stesse dimensioni degli embedding precedenti. Come illustrato nella figura, E_i rappresenta l'embedding iniziale del token i-esimo, T_i rappresenta l'output finale, mentre T_{rm} denota le rappresentazioni intermedie del token attraverso i vari layer del modello. Ad esempio, in un modello BERT con 12 layer, l'embedding avrà 12 rappresentazio-

ni intermedie, ognuna corrispondente all'output di ciascun layer.
Nonostante i suoi vantaggi, BERT presenta anche delle sfide significative. Il modello è altamente complesso e richiede ingenti risorse computazionali per il preaddestramento e la sua ottimizzazione. Inoltre, la comprensione della lingua da parte di BERT, sebbene avanzata, può ancora risultare limitata in contesti ambigui o dove il significato dipende fortemente da sfumature sottili. Il meccanismo di attenzione utilizzato da BERT consente a ciascun token della sequenza di input di connettersi con qualsiasi altro token simultaneamente, e i pesi associati a queste connessioni, appresi durante l'addestramento, determinano l'influenza relativa di ciascuna parola sulle altre. In tal senso, il significato di una parola all'interno di una frase può variare in base al contesto: viene data importanza alle parole chiave che determinano il significato complessivo della frase. In una struttura unidirezionale, infatti, la finestra di contesto poteva spostarsi solo in una direzione. Di conseguenza, le parole non ancora osservate non potevano essere prese in considerazione per comprendere il significato dell'input corrente, anche se la loro presenza poteva essere determinante. Diversamente, in BERT, il meccanismo di attenzione consente a ciascun token della sequenza di input di connettersi con qualsiasi altro token nello stesso momento.
In conclusione, BERT rappresenta un passo avanti significativo nell'elaborazione del linguaggio naturale, grazie alla sua capacità di comprendere il contesto bidirezionale delle parole. La sua architettura basata sui Transformers e l'approccio pre-addestrato fine-tuned lo rendono uno strumento potente per una vasta gamma di applicazioni di NLP. In più, il codice sorgente e i modelli BERT, che coprono 103 lingue, sono facilmente accessibili grazie ai diversi framework e librerie (tensorflow, pyTorch, Keras, ecc.), in diversi linguaggi di programmazione (python, java, C++, ecc.).

3.2 Kenneth Enevoldsen: un modello ottimizzato

Kenneth Enevoldsen è un ricercatore nel campo del NLP e dell'IA, noto per il suo lavoro sulla sentiment analysis. Tra i suoi contributi più rilevanti, vi è lo sviluppo di un modello basato su un'architettura di Transformers, simile a BERT, ma ottimizzato per contesti linguistici specifici e per l'efficienza computazionale.
Il modello proposto da Enevoldsen [12] si basa su un'architettura di rete neurale simile a quella di BERT, ma con alcune modifiche significative. Il modello di

Enevoldsen presenta una semplificazione nell'architettura, che lo rende più facile da allenare e meno costoso in termini di risorse computazionali. La principale differenza risiede nella riduzione del numero di parametri e nella struttura dell'architettura, che impiega un encoder Transformer semplificato, con meno layer rispetto ai 12 di BERT. Questa semplificazione rende il modello più leggero e più facile da addestrare, mantenendo comunque un'elevata accuratezza nei compiti di NLP specifici, come la sentiment analysis. Inoltre, la sua ottimizzazione per lingue specifiche permette di ottenere performance superiori in contesti dove BERT potrebbe non essere altrettanto efficace senza un'ottimizzazione estensiva.
Gli embeddings generati sono contestuali, ossia tengono conto del significato delle parole nel loro contesto, ed hanno una dimensione del vettore di embedding ridotta a 256 dimensioni. La scelta di un approccio semplificato è motivata dalla necessità di gestire testi in lingue con risorse limitate e di ottenere risultati efficienti senza compromettere la qualità della rappresentazione semantica.
Inoltre, la versione semplificata permette tempi di addestramento significativamente ridotti e, riducendo il numero di layer e parametri, il modello richiede meno memoria e potenza di calcolo.
La scelta di adottare il modello di Enevoldsen è stata motivata dalla necessità di esplorare un'architettura meno complessa e più adatta a risorse limitate.

3.3 MiniLM: un modello compatto per il NLP

Il modello proposto da MiniLM [32] presenta un approccio innovativo per ridurre le dimensioni e il costo computazionale dei modelli di Transformers preaddestrati, come BERT, senza compromettere significativamente le loro prestazioni. Questo è particolarmente importante in applicazioni di NLP dove l'efficienza è cruciale, come in dispositivi con risorse limitate.
L'elemento centrale di MiniLM è la "Deep Self-Attention Distillation", una tecnica che permette di estrarre le informazioni critiche dal meccanismo di autoattenzione dei modelli di grandi dimensioni (i cosiddetti "teacher models") in versioni più compatte e leggere (i "student models"). Il processo di estrazione delle informazioni avviene emulando la struttura multi-layer della autoattenzione, mantenendo le informazioni più rilevanti che sono alla base della comprensione del contesto linguistico. Questo permette a MiniLM di preservare la capacità di rappresentazione contestuale dei modelli più grandi, ma con un

numero significativamente ridotto di parametri e calcoli. In particolare, MiniLM utilizza solo 6 layer e la dimensione del vettore di embedding è ridotta a 384 dimensioni. Nonostante questa diminuzione dell'architettura, questo modello riesce a raggiungere performance competitive rispetto a BERT-base, che ha 12 layer e 768 dimensioni. Inoltre, tale riduzione, rende MiniLM più leggero e più veloce da addestrare. Tutto ciò, rende il modello particolarmente utile per ambienti con risorse limitate o per applicazioni in tempo reale.
Una tecnologia di questo tipo, non solo è più efficiente in termini di memoria e velocità di esecuzione, ma può essere applicata a una varietà di compiti diversi senza la necessità di adattamenti o modifiche specifiche. A differenza di altre tecniche che richiedono la personalizzazione per specifici compiti di NLP, essa non è specificamente progettata per un compito particolare. In altre parole, un modello task-agnostic è versatile e flessibile, capace di funzionare bene in diversi contesti o applicazioni senza essere ottimizzato esclusivamente per uno specifico tipo di attività, mantenendo comunque un buon equilibrio tra efficienza e accuratezza.
Nel relativo paper, viene dimostrato che, nonostante la riduzione delle dimensioni e della complessità, MiniLM mantiene prestazioni competitive rispetto ad altri modelli su diverse attività di NLP, come il question answering e la classificazione del testo. Questo lo rende una soluzione particolarmente interessante per implementazioni che richiedono tempi di elaborazione rapidi, senza compromettere eccessivamente la qualità delle previsioni.
Nel contesto degli esperimenti condotti, MiniLM è stato selezionato per la sua efficienza su sistemi a risorse limitate, permettendo di ottenere risultati di alta qualità. Inoltre, la sua capacità di essere utilizzato senza necessità di adattamenti specifici per ogni compito facilita notevolmente la sperimentazione su diversi set di dati e task di NLP.

4

Metriche

In questo capitolo vengono descritte le metriche utilizzate per eseguire una valutazione quantitativa dei risultati ottenuti dagli esperimenti trattati successivamente nei prossimi capitoli. L'obiettivo principale è valutare le prestazioni dei modelli di Human Motion Prediction (HMP), analizzando quanto efficacemente questi modelli riescano a prevedere il movimento umano. A questo scopo, viene introdotto anche il modello Zero Velocity (ZV), cioè un modello di riferimento essenziale che funge da base per misurare i miglioramenti apportati dalle diverse tecniche implementate.
In generale, per determinare la qualità della previsione ottenuta dal modello di anticipazione, è necessario confrontare ogni frame con il corrispettivo della sequenza della Ground Truth (GT). Questo confronto viene effettuato utilizzando diverse metriche che analizzano ogni singolo frame e considerano gli angoli di rotazione o le coordinate dei giunti che descrivono le pose umane. Le metriche più rilevanti per questo compito sono: l'errore di posizione medio, MPJPE, che misura la distanza media tra la posizione predetta e quella reale dei giunti; l'errore di Eulero, MAE, il quale valuta l'accuratezza degli angoli di rotazione predetti rispetto a quelli reali; la JAD, la quale misura la differenza angolare tra le articolazioni. Queste metriche insieme offrono una visione completa della performance del modello.

4.1 Mean Per Joint Position Error (MPJPE)

Il MPJPE [11] è una delle metriche utilizzate per valutare le prestazioni dei modelli di HMP. Essa misura l'accuratezza delle pose umane previste dal modello confrontando le posizioni articolari di tali pose con le rispettive della GT. Questa metrica è essenziale per comprendere quanto bene un modello sia accurato nel prevedere i movimenti umani futuri.
La MPJPE calcola la distanza euclidea, ovvero la più breve distanza tra due punti nello spazio, tra le posizioni articolari previste e le corrispondenti posizioni articolari reali, per ogni giunto. In particolare, l'errore è calcolato tramite lo scarto quadratico medio della distanza euclidea di tutti i giunti di un frame. Questo errore è poi mediato su tutti i frame della sequenza, e per tutte le sequenze del dataset, per fornire un unico valore rappresentativo dell'accuratezza della previsione. Matematicamente, per un dato frame f dello scheletro S, la MPJPE è calcolata come:

$$\text{MPJPE}(f, S) = \frac{1}{K} \sum_{i=1}^{K} \|\mathbf{p}_{prev,f,S}(i) - \mathbf{p}_{gt,f,S}(i)\|^2 \tag{4.1}$$

dove:

- K è il numero di keypoint nello scheletro S;
- $\mathbf{p}_{prev,f,S}(i)$ è la posizione prevista della i-esima articolazione nel frame f;
- $\mathbf{p}_{gt,f,S}(i)$ è la posizione reale della i-esima articolazione nel frame f;

Tale relazione viene eseguita per ogni frame del mocap, per poi eseguire la media di tutti gli errori ottenuti. Il risultato è tipicamente riportato in millimetri per le coordinate 3D, rendendo più facile interpretare la precisione del modello. In particolare, per questa tecnica, valori di MPJPE più bassi indicano una migliore accuratezza della previsione. Tuttavia, la MPJPE presenta limitazioni, in particolare per quanto riguarda la sua specificità per il soggetto: dal momento che i corpi umani variano per dimensioni e proporzioni, l'errore posizionale può essere fuorviante se il modello non tiene conto delle differenze individuali nelle distanze tra i keypoints. Per affrontare questo problema, prima di calcolare la MPJPE lo scheletro viene normalizzato, ovvero si cerca di standardizzare le lunghezze tra i vari soggetti, fornendo una metrica di valutazione più coerente. In più, in termini di cinematica diretta, un errore di posizione può portare ad

un errore angolare, o viceversa, e tale errore è proporzionale alla lunghezza del segmento del corpo coinvolto. Infatti, se due segmenti del corpo, uno lungo e uno corto, hanno lo stesso errore angolare, il segmento più lungo produrrà un errore di posizione maggiore perché la sua lunghezza amplifica lo spostamento risultante.

4.2 Mean Angular Error (MAE)

Il MAE [19] è un'altra metrica fondamentale utilizzata in ambito HMP per misurare l'accuratezza della previsione degli angoli di rotazione delle articolazioni umane. Questa metrica misura l'errore angolare medio nelle previsioni delle rotazioni articolari, espresse in angoli di Eulero, valutando la differenza tra le rotazioni previste e quelle reali nello spazio tridimensionale. Nel contesto della previsione del movimento umano, il MAE viene impiegato per confrontare le rotazioni predette con quelle della GT, fornendo una misura dell'accuratezza della previsione degli angoli articolari. Questa metrica è particolarmente utile in ambito HMP, poiché le rotazioni articolari determinano direttamente la postura del corpo umano. Questa tecnica consente di valutare la capacità del modello di prevedere movimenti complessi come rotazioni in diverse direzioni, un aspetto cruciale nella previsione del movimento umano.
Il MAE viene calcolato confrontando i tre angoli di rotazione di ogni keypoint, tra quelli della previsione del modello e quelli effettivi della GT, per ogni frame. La metrica misura la differenza assoluta tra i valori corrispettivi e l'errore totale è ottenuto tramite lo scarto quadratico medio di questi valori su tutti i keypoint. L'errore è poi mediato per tutti i frame della sequenza, e per ogni sequenza del dataset.
Matematicamente, per un dato frame f dello scheletro S, il MAE può essere calcolato come:

$$\text{MAE}(f, S) = \frac{1}{K} \sum_{i=1}^{K} \|\boldsymbol{\theta}_{prev,f,S}(i) - \boldsymbol{\theta}_{gt,f,S}(i)\|^2 \tag{4.2}$$

dove:

- K è il numero di keypoint nello scheletro S;
- j identifica i tre angoli di Eulero per ogni articolazione;

- $\theta_{prev,f,S}(i)$ rappresenta l'angolo di rotazione previsto per l'articolazione i;
- $\theta_{gt,f,S}(i)$ rappresenta l'angolo di rotazione reale per l'articolazione i;

Tale relazione viene eseguita per ogni frame del mocap, per poi eseguire la media di tutti gli errori ottenuti. Questa metrica fornisce un'indicazione di quanto la previsione sia accurata in termini di rotazioni articolari, che influenzano direttamente la postura globale del corpo.
Nell'ambito della HMP, prevedere accuratamente le rotazioni articolari è cruciale perché le rotazioni influenzano direttamente la dinamica del movimento. Modelli predittivi che trascurano questa metrica potrebbero risultare accurati nel predire le posizioni dei giunti, ma non sarebbero in grado di catturare correttamente le variazioni nel movimento articolare. Di conseguenza, la postura e i movimenti risultanti potrebbero sembrare innaturali.
Uno dei vantaggi principali dell'uso del MAE è la sua capacità di quantificare la rotazione angolare in modo preciso, indipendentemente dalla lunghezza dei segmenti corporei. Questa caratteristica è particolarmente rilevante nei casi in cui un errore angolare può portare a un errore amplificato nella posizione, come nel caso dei segmenti corporei più lunghi. In questo contesto, il confronto tra le rotazioni effettive e quelle previste diventa essenziale per mantenere la coerenza del movimento, soprattutto per applicazioni che richiedono alta fedeltà visiva. Tuttavia, il MAE presenta una limitazione che riguarda la possibile ambiguità delle rappresentazioni angolari: diverse configurazioni angolari possono portare allo stesso orientamento nello spazio. Per ovviare a queste problematiche, in alcune applicazioni avanzate (ad esempio PVRED) si utilizzano rappresentazioni alternative come i quaternioni. Nonostante ciò, il MAE rimane una metrica comunemente utilizzata, soprattutto per la sua interpretabilità e semplicità nel contesto della previsione del movimento umano.

4.3 Joint Angle Difference (JAD)

La JAD è un'ulteriore metrica di valutazione dell'errore utilizzata per la previsione del movimento umano. Essa misura la differenza tra gli angoli articolari previsti e quelli reali per ogni giunto di ogni frame. A differenza del MAE, che misura l'errore basato sugli angoli di rotazione nello spazio, la JAD misura la differenza tra le matrici di rotazione, necessaria per allineare

l'articolazione prevista con la direzione effettiva (GT). In pratica, quantifica di quanto bisogna ruotare un'articolazione predetta affinché combaci con quella reale, valutando così la precisione angolare nella predizione dei movimenti articolari. Questa metrica è più diretta e applicabile nel contesto dell'analisi del movimento umano.
Una rappresentazione della JAD, per un dato frame f dello scheletro S, può essere espressa come:

$$\text{JAD} = \frac{1}{K}\sum_{i=1}^{K}\arccos\left(\frac{Tr(\mathbf{R}_{y_i}^{\hat{y}_i})-1}{2}\right) \tag{4.3}$$

Dove:

- K è il numero di keypoint nello scheletro S;
- $\mathbf{R}_{f,S}(i)$ è la matrice di rotazione relativa al giunto i e rappresenta la rotazione necessaria per allineare il giunto i-esimo con la GT;

La JAD risulta essere una metrica specializzata e diretta per valutare l'errore nei modelli di previsione del movimento umano, concentrandosi specificamente sulle articolazioni e sulla precisione del movimento articolare. La sua semplicità e specificità la rendono particolarmente utile nei contesti in cui è fondamentale catturare accuratamente il movimento di ciascun giunto. Rispetto ad altre metriche, essa offre una prospettiva più dettagliata e localizzata, rendendola adatta a situazioni che richiedono un'analisi precisa del movimento articolare piuttosto che una valutazione globale dell'orientamento.

4.4 Zero-Velocity: base di riferimento per la HMP

Lo ZV è un modello semplice e intuitivo per la HMP, spesso utilizzato come riferimento base per valutare l'efficacia di modelli più avanzati. Questo modello, come suggerisce il nome, assume che non ci sia movimento futuro, ovvero la posizione del corpo umano nel tempo futuro rimane costantemente ferma, replicando l'ultima posa della sequenza di input per tutta la durata della previsione.
Sebbene questa assunzione possa sembrare poco sensata, lo ZV ha dimostrato di essere efficace in molti casi, specialmente quando si tratta di sequenze di movimento a breve termine. Questo risultato è legato alla complessità del problema

di predire il movimento umano: il corpo umano è altamente articolato e capace di una grande quantità di movimenti complessi e imprevedibili. Anche piccoli errori nella previsione di un movimento possono causare errori significativi, soprattutto quando si predice un movimento nella direzione opposta a quella reale. In questi casi, si va generando un errore significativamente maggiore rispetto a quello di una previsione statica. Di conseguenza, lo ZV viene spesso utilizzato come baseline per valutare le performance dei modelli più avanzati. Se un modello più complesso non riesce a superare lo ZV, ciò indica che il modello potrebbe non essere così efficace o che la complessità aggiuntiva non stia portando ai miglioramenti attesi. Questa semplicità e robustezza rende lo ZV un valido strumento per valutare la reale efficacia delle nuove tecniche di predizione del movimento umano, evidenziando in maniera chiara quando un approccio più sofisticato non apporta miglioramenti significativi rispetto ad una previsione statica.

5

Setup sperimentale

Questo capitolo è strutturato in diverse sezioni che descrivono in dettaglio il processo sperimentale adottato. Si inizia con il preprocessing dei dataset, suddiviso in due parti: la prima dedicata al trattamento dei dati sulle pose e la seconda sulle descrizioni. In entrambi i casi, vengono illustrati due approcci distinti: uno iniziale che utilizza un dataset ridotto, per avere una visione d'insieme della struttura del modello, e uno più specifico e accurato, che utilizza il dataset completo e che permette di eseguire analisi e test più significativi grazie a un dataset più denso e strutturato. Questo metodo ha permesso di approfondire le caratteristiche specifiche di ciascun dataset, facilitando il processo di integrazione successiva. Successivamente, il capitolo si focalizza sull'integrazione dei due dataset, spiegando le tecniche impiegate per combinare le informazioni relative alle pose e alle descrizioni.
La sezione 5.4 riguarda la creazione del modello, viene analizzata la gestione degli input e si discutono le varie architetture utilizzate per generare gli embedding delle descrizioni, con l'obiettivo di ottenere la miglior rappresentazione possibile.
Infine, il capitolo si conclude con la descrizione degli esperimenti, dove è stata dedicata particolare attenzione al training dei modelli di DL, dove si sono eseguiti diversi cicli di ottimizzazione per migliorare le loro prestazioni.

5.1 Studio del dataset delle pose: analisi dei dati e preprocessing

Inizialmente, è stato studiato il dataset AMASS [18]. Come anticipato nel paragrafo 1.2.3, AMASS è uno dei dataset più completi e ampi disponibili per la HMP. Esso aggrega diversi piccoli dataset di dati di pose e standardizza le rappresentazioni dello scheletro umano utilizzando il modello SMPL, mostrato in Figura 2.1. Questa rappresentazione descrive le pose umane attraverso le rotazioni di 24 giunti.
In AMASS, i dataset contengono sequenze di movimento a diversi fps, che variano tra 30 e 120 fps a seconda della fonte specifica del dataset. Per garantire coerenza e uniformità nell'elaborazione dei dati e nei successivi esperimenti, è stato deciso di uniformare tutti i dati a un frame rate di 25 fps. Questa scelta è stata motivata anche dal fatto che molti modelli, come PVRED, sono progettati per lavorare con sequenze di 25 fps. Uniformare i frame rate semplifica la gestione ed il preprocessing dei dati, inoltre garantisce una struttura coerente nei successivi passaggi sperimentali. In particolare, per la gestione e la visualizzazione dei mocap sono state utilizzate le funzioni presenti nella libreria *hmp_utils*.

5.1.1 Preprocessing pose per dataset ridotto

I dati di AMASS selezionati, come approccio iniziale, appartengono ai dataset di "KIT" e di "ACCAD". La scelta di "KIT" è stata motivata dal suo lungo minutaggio di riprese e dalla sua ampia quantità di dati contenuti nel dataset BABEL, analizzato nel prossimo paragrafo. Anche "ACCAD" è stato considerato perché ben rappresentato in BABEL, rendendolo utile per l'integrazione di pose e descrizioni del movimento. Si procede, quindi, nel caricare i file presenti in queste cartelle: viene utilizzata la classe *Mocap_Loader*, della libreria *hmp_utils*, la quale necessita di alcune informazioni riguardanti il tipo di dataset utilizzato, come il numero di keypoints ed il framerate. Questa classe ha una funzione in grado di estrarre il contenuto dai file; si crea così una lista contenente tutti i mocap, ognuno dei quali è rappresentato da un tensore di tre dimensioni: la prima esprime il numero di frame, ed è diversa per ogni mocap; la seconda indica il numero di keypoints (24 nel caso di AMASS); la terza è pari a 3 in quanto, come già spiegato, nella cinematica diretta ogni keypoint è identificato dalle rotazioni ai tre assi rispetto al keypoint di riferimento. A questa lista ne

viene affiancata una seconda che contiene i corrispondenti percorsi file che sarà successivamente utilizzata per il processo di integrazione.
Solitamente, viene realizzato un modello predittivo il quale riceve come input due secondi di mocap per prevederne il secondo successivo. Dal momento che il framerate considerato è di 25 fps, allora l'input conterrà 50 frame ed il target 25, in totale 75 frame. In questo approccio iniziale è stato scelto di mantenere una singola sequenza di 75 frame per mocap, con inizio casuale. Tuttavia, nell'utilizzare questa scelta casuale, l'impostazione di un *random_seed* è necessaria al fine di ottenere la riproducibilità del modello, ovvero, ogni volta che esso viene eseguito, è necessario che prelevi sempre le stesse sequenze di frame.
Per eseguire questa operazione è necessario che il mocap abbia minimo 75 frame, in caso contrario tale mocap viene ignorato. Riguardo tutti gli altri mocap, invece, sebbene sia possibile selezionare più finestre da 75 frame all'interno della sequenza di movimento, per un approccio semplice ne è stata presa solo una. Al termine di questa operazione il dataset conterrà un totale di 4353 finestre.
Questa fase iniziale è stata eseguita con l'obiettivo di esplorare l'integrazione dei due dataset e per lo sviluppo del modello di anticipazione. Pertanto, il preprocessing delle pose è stato eseguito in maniera preliminare, utilizzando questo dataset ridotto per sperimentare con l'architettura del modello e individuare la configurazione ottimale. Successivamente, si è passati a un'analisi più approfondita del dataset delle pose per migliorare la precisione delle previsioni.

5.1.2 Preprocessing pose per dataset completo

Per l'architettura del modello finale, è stato scelto di utilizzare tutti i dataset appartenenti ad AMASS che contengano dati ai quali si hanno riferimenti nel dataset di BABEL. Il fine è quello di massimizzare il dataset utile. La relativa lista dei dataset è di seguito riportata:

```
['BMLrub', 'ACCAD', 'CMU', 'MPIHDM05', 'EyesJapanDataset', 'KIT',
'EKUT', 'MPImosh', 'TCDhandMocap', 'DFaust67', 'MPILimits', 'SFU',
'TotalCapture', 'HumanEva', 'SSMsynced', 'BMLmovi', 'Transitionsmocap']
```

Sono stati, quindi, selezionati tutti questi dataset e, analogamente a quanto già analizzato nel sotto-paragrafo precedente, per mezzo delle librerie di *hmp_utils* viene creata la lista di tutti i mocap, che contiene 12728 mocap. Ad

essa viene affiancata la lista con i corrispettivi percorsi file dai quali sono estratti i mocap.
Per i motivi illustrati nel sotto-paragrafo precedente, per ogni mocap si estrae una sequenza di frame con una finestra da 75 frame, purché il mocap ne abbia almeno 75. In questa versione, tuttavia, è stato scelto un diverso approccio: non si considera più un'unica finestra per mocap prelevata con un inizio casuale, ma, attraverso una finestra mobile, vengono prelevati 75 frame ogni 50.
Questo metodo consente di aumentare significativamente la copertura temporale dei movimenti all'interno del dataset, migliorando il modo in cui le sequenze sono rappresentate e fornendo al modello una quantità maggiore di informazioni utili per l'addestramento. Di conseguenza, si ottiene un dataset più ampio e denso, che fornisce una base solida per integrare in modo più efficace le descrizioni associate ai movimenti, migliorando l'intero processo di sviluppo del modello di anticipazione.

5.2 Studio del dataset delle descrizioni: analisi delle caratteristiche e preprocessing

In seconda analisi, è stato studiato il dataset BABEL [23]. Il dataset fornisce annotazioni semantiche per sequenze di motion capture 3D, associando descrizioni testuali dettagliate ai movimenti. È costruito a partire da una raccolta di dati AMASS, con annotazioni che coprono una vasta gamma di azioni umane quotidiane e complesse. Le descrizioni sono organizzate in diversi livelli e possono essere utilizzate per studi sul riconoscimento di azioni e sull'integrazione tra linguaggio e movimento umano, permettendo lo sviluppo di modelli che comprendano e generino movimenti basati su descrizioni testuali. Nel seguito, viene riportato un esempio di annotazione del dataset BABEL per illustrare meglio la sua struttura.


```
{"1833": {
    "babel_sid": 1833,
    "url": "https://babel-renders.s3.eu-central-1.amazonaws.com/001833.mp4",
    "feat_p": "BMLmovi/BMLmovi/Subject_55_F_MoSh/Subject_55_F_21_poses.npz",
    "dur": 8.3,
    "seq_ann": {
```

```
        "babel_lid": "8b112ce1-2546-4675-9efa-27db9fcfee29",
        "anntr_id": "c24c8ba1-f43d-48db-9f76-165d8264bda1",
        "mul_act": true,
        "labels": [
            {
                "raw_label": "walk back and forth",
                "proc_label": "walk back and forth",
                "seg_id": "3ca74412-f765-445b-b51b-6c2e72316010",
                "act_cat": [
                    "walk",
                    "forward movement",
                    "backwards movement"
                ]
            }
        ]
    },
    "frame_ann": {
        "babel_lid": "aabf3b2d-c8c7-47ac-aa75-4621d262434e",
        "anntr_id": "c0974846-d198-425d-846a-ce6808343841",
        "mul_act": true,
        "labels": [
            {
                "raw_label": "standing",
                "proc_label": "stand",
                "seg_id": "841150c3-030d-4369-a2e1-cf460f4a988a",
                "act_cat": [
                    "stand"
                ],
                "start_t": 0,
                "end_t": 0.341
            },
            {
                "raw_label": "standing",
                "proc_label": "stand",
                "seg_id": "947c777d-ac8f-43f1-b2bf-08578293832e",
                "act_cat": [
                    "stand"
                ],
```

```
                "start_t": 7.862,
                "end_t": 8.3
            },
            {
                "raw_label": "pacing",
                "proc_label": "pace",
                "seg_id": "59af352a-0f5e-446e-9690-18e5d1d9ebaf",
                "act_cat": [
                    "walk"
                ],
                "start_t": 0.341,
                "end_t": 7.862
            }
        ]
    }
}
```


Esempio di annotazione nel dataset BABEL

Tutti gli elementi all'interno del dataset sono contenuti in un file JSON ed ognuno di essi contiene al suo interno diversi campi. Ogni file fa riferimento ad un unico mocap ed è associato ad un valore numerico detto chiave ("1833" nell'esempio); per ogni chiave è possibile trovare come prime informazioni il percorso file (chiave "feat_p") dove si trova il file e la sua durata in secondi. Successivamente è presente una serie densa di informazioni; ogni mocap è associato a una "sequence annotation" (chiave "seq_ann"), che fornisce informazioni dettagliate su tutto il movimento catturato, includendo le etichette "raw_label" e "act_cat". La prima contiene una descrizione più dettagliata in linguaggio naturale, mentre la seconda rappresenta la sua categorizzazione rispetto ad un insieme di classi di azioni. Oltre a queste annotazioni sull'intero mocap, è presente anche una "frame annotation" (chiave "frame_ann"), che consiste in una lista di segmenti di video. Ogni elemento di questa lista contiene le stesse informazioni presenti nella "seq_ann", ma fa riferimento a intervalli di tempo specifici all'interno della sequenza, definiti dai tempi "start_t" ed "end_t".
Questo dataset presenta una serie di vantaggi e criticità che verranno analizzate e discusse nel presente e nel prossimo capitolo. In totale, il dataset contiene 10892 dati di tipo dizionario; tuttavia, non tutti contengono la stessa quantità di informazioni, i più si limitano ad avere quelle relative l'intero mocap ("seq_ann"). Si

vuole comunque specificare che, nonostante i dati siano densi di informazioni, solo quelle contenute nelle voci "raw_label" e "feat_p" sono rilevanti allo scopo dello studio condotto in questa tesi.

5.2.1 Preprocessing descrizioni per dataset ridotto

In modo analogo a quanto svolto per l'approccio iniziale del preprocessing delle pose, anche in questo caso è stata scelta una gestione semplificata delle descrizioni. Inizialmente, vengono utilizzate le informazioni contenute nelle "feat_p" e nelle "raw_label", queste ultime appartenenti alla sezione "seq_ann". In particolare, si ha che alcuni elementi sono senza informazioni semantiche, quindi vengono esclusi per non aggiungere complessità al modello. Così facendo, il dataset finale sarà ridotto a 8808 descrizioni.
Prima di procedere al preprocessing dettagliato, è stata eseguita un'analisi riguardo la complessità delle descrizioni presenti in questo dataset.

Analisi dataset delle "sequence annotations"

Dopo aver completato il preprocessing delle descrizioni, si è deciso di condurre uno studio approfondito del dataset per analizzarne le caratteristiche principali e la consistenza delle descrizioni. Ciò che si è voluto analizzare riguarda la lunghezza delle descrizioni. Il grafico in Figura 5.1, infatti, raffigura la distribuzione del numero di parole per descrizione.

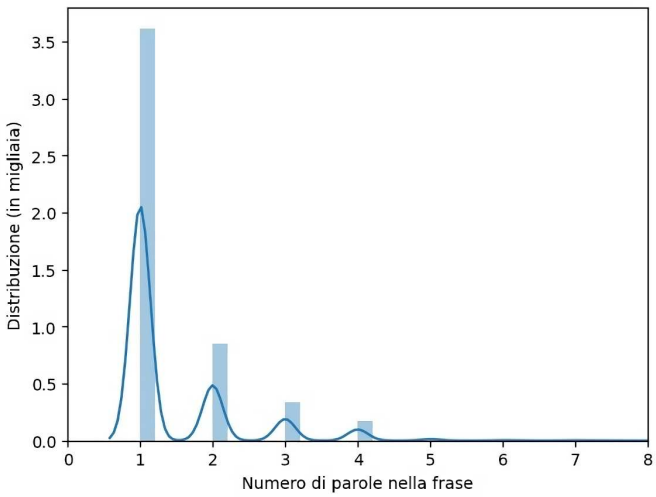

Figura 5.1: Distribuzione numero di parole per descrizione

Si può osservare che la lunghezza massima delle frasi è di circa 5 parole e, in particolar modo, si vuole far notare che la maggior parte di descrizioni si presenta con solamente una parola. Questo aspetto è particolarmente rilevante, poiché potrebbe non essere vantaggioso per l'obiettivo di questa tesi: i mocap contengono spesso pose molto variabili nel tempo ed un embedding semantico derivato da descrizioni di una parola potrebbe non essere abbastanza robusto per catturare variazioni complesse nei movimenti umani. Se ad esempio una sequenza di movimento contenesse una persona che da seduta si alza per camminare, e tale sequenza fosse definita dalla categoria "walk", si rischierebbe di perdere informazioni sulle transizioni e le dinamiche associate al movimento. Nel capitolo successivo, infatti, viene analizzato in dettaglio il risultato dell'esperimento eseguito con il dataset ridotto.

5.2.2 Preprocessing descrizioni per dataset completo

A causa dell'analisi condotta nel paragrafo precedente, si è reso necessario un preprocessing delle descrizioni che esamini in modo più dettagliato il contenuto delle sequenze di movimento. È fondamentale sviluppare un metodo con una logica più strutturata per la creazione delle finestre di movimento e per l'assegnazione di descrizioni adeguate e specifiche a ciascuna finestra, garantendo così che ogni descrizione rifletta con precisione le dinamiche della sequenza di movimento associata.
In questo caso, vengono considerate le informazioni contenute nelle "feat_p" e nelle "raw_label", dove queste ultime appartengono alla sezione "frame_ann". Come già annunciato nell'introduzione di questo paragrafo (5.2), questa sezione frammenta il video in segmenti che vanno dal tempo "start_t" al tempo "end_t", ovvero è stabilito rispetto a quali frame si riferiscono le relative descrizioni. Anche in questo caso, si mantengono solo i dati che contengono informazioni semantiche, i quali risultano essere 5341 su 10892.
Per questo preprocessing risulta necessario estrarre le informazioni contenute in "raw_label" (illustrate nel paragrafo 5.2), che includono sia la descrizione della sequenza che i frame di inizio e fine corrispondenti. Queste informazioni vengono quindi ordinate in ordine temporale. Per ogni campione del dataset, viene creata una tabella che organizza le descrizioni in base alla loro sequenza temporale, convertendo i dati temporali (tempo di inizio e fine) nei rispettivi frame.

Per una maggiore comprensione, è riportato di seguito un esempio in cui tale lista si può presentare.

```
[0, 13, 't-pose'],
[13, 24, 'transition'],
[24, 140, 'walk forwards'],
[32, 76, 'turn right'],
[93, 131, 'turn left'],
[140, 143, 'transition'],
[143, 190, 'walk backwards'],
```

Successivamente, è stata adottata una logica per la creazione delle stringhe di descrizione, costruendo una matrice con un numero di righe pari al numero di descrizioni individuate nella relativa lista (ad esempio, la lunghezza della lista illustrata nell'annotazione precedente). Il numero di colonne corrisponde invece alla lunghezza massima del mocap (in frame) a seguito della suddivisione in finestre temporali (ossia il valore massimo riportato nella seconda colonna della medesima lista dell'esempio). Per ogni riga della matrice, viene inserita la relativa descrizione in ogni colonna tra quella di inizio e quella di fine: nell'esempio, nella prima riga della matrice, tra le colonne "0" e "13", viene inserita la descrizione "t-pose". In questo modo, logicamente, la i-esima colonna di questa matrice rappresenta l'i-esimo frame del mocap, e tutte le descrizioni che si trovano nella colonna appartengono a quel frame. La stringa è stata costruita in modo tale che, quando la colonna presenta più descrizioni (in righe diverse), queste vengono unite da un "and", formando un gruppo di descrizioni; gruppi di descrizioni appartenenti a colonne diverse, invece, vengono uniti da un "then". Quest'ultimo passaggio viene gestito verificando quando si osserva una colonna diversa dalla precedente, ovvero se, e solo se, un gruppo è diverso dal precedente. Inoltre, sono state fatte due scelte fondamentali. La prima, poiché in tutti i mocap c'è una porzione consistente in cui la posa è in "t-pose", si è deciso di saltare i primi 10 frame. La seconda scelta prevede di ignorare le descrizioni che durano troppo poco tempo: nell'esempio sopra riportato, nel caso del secondo "transition" che dura solo 4 frame, si considera che non fornisca informazioni sufficientemente significative. Un ultimo particolare che può presentarsi è la mancanza di descrizioni; in tal caso si aggiunge alla stringa la parola "unknown".
Grazie a tale logica, la descrizione risultante dall'esempio riportato sopra si

presenta come: "t-pose then transition then walk forwards then walk forwards and turn right then walk forwards then walk forwards and turn left then walk forwards then walk backwards".

5.3 Integrazione dei due dataset

Segue ora l'integrazione dei due dataset precedentemente illustrati.
Per quanto riguarda l'approccio sul dataset ridotto, si basti verificare quale descrizione appartiene a quale mocap. A tal fine, le liste contenenti i passaggi delle cartelle hanno un ruolo fondamentale. Si ricorda che ogni informazione semantica si riferisce ad uno specifico mocap del dataset AMASS tramite la chiave "feat_p". Quindi, per garantire la coerenza tra i dataset, è necessario parificarli mantenendo solo le descrizioni che hanno il relativo mocap e viceversa. Questo richiede un processo in cui si verifica che ogni percorso file dell'i-esimo mocap è presente anche nel dataset delle descrizioni. Se tale corrispondenza esiste, si mantengono entrambi, altrimenti viene scartato il mocap. Questo passaggio assicura che ogni coppia mocap-descrizione rimanga allineata e valida per poter essere utilizzata nel modello. Dopo aver eseguito questo passaggio di allineamento, le sequenze di movimento valide risultano essere 2726. Tale diminuzione del dataset utile è causa del fatto che in BABEL non sono presenti dati relativi a tutti i mocap considerati nella fase iniziale.
Per quanto riguarda la parifica dei dataset completi, invece, il processo si divide in due parti. 1) Prima ancora di eseguire gli algoritmi di suddivisione in finestre dei mocap e di costruzione delle frasi, si esegue la prima parifica in modo analogo all'approccio utilizzato con il dataset ridotto: tramite le informazioni dei percorsi file si mantengono i mocap con relativa descrizione. A seguito della parifica, si ottiene una lista contenente 4991 mocap e informazioni semantiche. 2) Segue ora la suddivisione in finestre dei mocap, a seguito della quale ogni mocap viene rappresentato da una seconda lista contenente una o più finestre. A seguire, viene svolta l'operazione di costruzione della frase con la logica illustrata al paragrafo 5.2.2. In particolare, in questa fase, non viene costruita l'intera descrizione del mocap (cioè considerando l'intera matrice), ma è necessario che essa si riferisca alla specifica finestra. Perciò, se la finestra è contenuta tra il frame j e il frame k, allora la frase è costruita tra le colonne j e k.
A seguito di queste due operazioni, si ottiene un dataset contenente un totale di 32455 coppie mocap-descrizione, dove ogni mocap possiede una descrizione

dedicata e ben specifica. Per la visualizzazione di alcuni esempi, al fine di illustrare come si presentano le descrizioni associate alle pose, è possibile accedere al link [7].

5.4 Struttura del modello di anticipazione

Concluso il processo di preprocessing dei dataset, si discute ora della creazione del modello di anticipazione. In questa sezione verrà descritto lo sviluppo e l'implementazione del modello, oltre alle strategie adottate per gestire gli input. Per quanto riguarda il modello predittivo, è stata sviluppata una RNN strutturata in Encoder e Decoder, utilizzando la cella LSTM, esattamente come illustrato in Figura 2.6. A questa struttura, però, dev'essere aggiunta la componente relativa alla semantica. Per essa, viene creata una terza classe, esterna a quelle di Encoder e Decoder, che riceve in input le descrizioni e genera l'embedding come output. Tale embedding è costituito da una lista con tanti elementi quanto il numero di descrizioni ricevuto in input. Definite le classi, è stato scelto di testare due diverse modalità di immissione degli embedding delle descrizioni all'interno del modello. Innanzitutto, si vuole specificare il ruolo delle tre classi. 1) L'Encoder riceve unicamente informazioni riguardanti le pose di input (ovvero i 50 frame) ed il suo compito è quello di elaborare le informazioni ricevute per creare l'hidden state - ed il cell state nel caso dell'utilizzo della cella LSTM - da dare in input al Decoder. 2) La classe Description Embedder, che produce l'embedding delle descrizioni, la quale riceve unicamente le informazioni semantiche ed invia l'embedding al Decoder. 3) La classe Decoder, infine, riceve i due embedding e li combina, insieme alle informazioni delle pose target (ovvero i 25 frame), per generare la previsione.

Embedding delle descrizioni tra Encoder e Decoder

Una modalità di combinazione delle informazioni testata, consiste nel concatenare gli embedding generati dall'Encoder e dal Description Embedder. Si ottiene quindi un vettore concatenato, che sarà dato in input all'hidden state del Decoder. In Figura 5.2 è possibile visualizzare lo schema di tale modello.

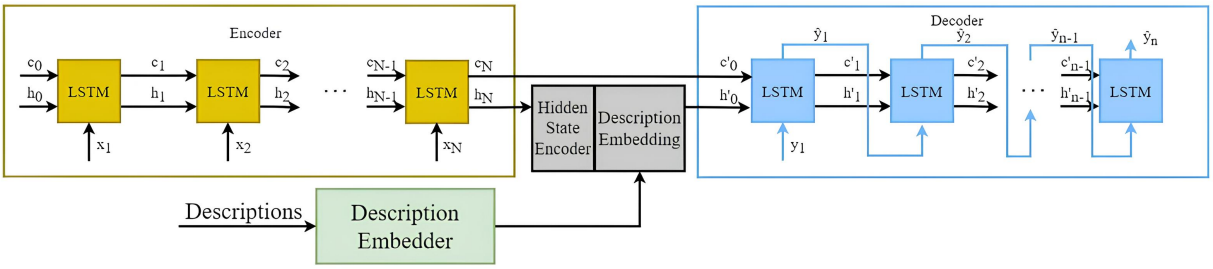

Figura 5.2: Modello RNN con integrazione delle descrizioni tra Encoder e Decoder

La scelta di concatenare l'embedding delle descrizioni all'hidden state dell'Encoder piuttosto che al cell state è stata svolta per garantire la compatibilità: come illustrato al paragrafo 2.2, la cella GRU producono unicamente l'hidden state, quindi se al posto della cella LSTM si scegliesse di usare quelle GRU, la logica del modello resterebbe la stessa.

Embedding delle descrizioni ad ogni input

In quest'aaltra modalità di combinazione delle informazioni, avviene la concatenazione dell'embedding semantico con ogni posa target, input del Decoder. Più precisamente, dal momento che il vettore di output del Description Embedder contiene l'embedding semantico di tutte le descrizioni, nel momento in cui viene calcolata la previsione dell'i-esima sequenza di movimento, ad ogni posa input del Decoder viene concatenato il corrispondente embedding semantico. In Figura 5.3 è possibile visualizzare lo schema di tale modello.

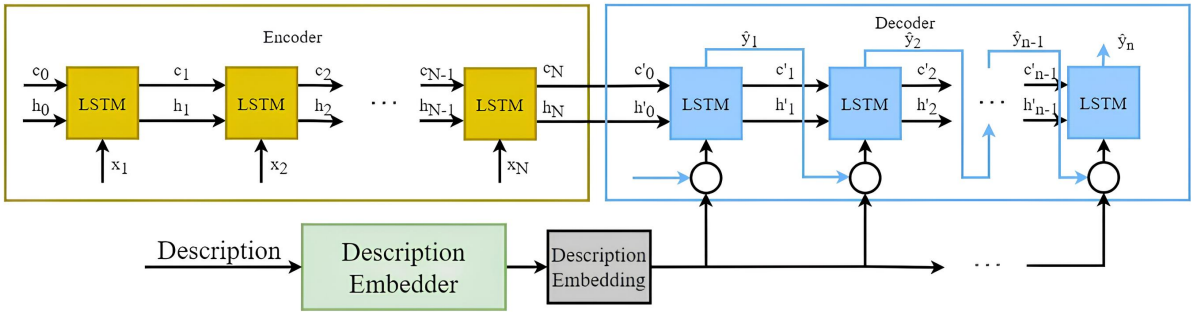

Figura 5.3: Modello RNN con integrazione delle descrizioni ad ogni input

In entrambe le modalità, la sequenza di output predetta, $\hat{\mathbf{Y}}$, viene costruita combinando le singole pose generate dalla cella del Decoder, $\hat{\mathbf{y}}_i$, per $i \in 1, ..., n$. In fase di addestramento, tale sequenza viene confrontata con la corrispettiva sequenza di target per eseguire il calcolo della funzione di perdita, detta *loss function*, tramite la funzione MSE.

Per quanto riguarda la generazione degli embedding semantici, invece, sono stati utilizzati tre diversi LLM, descritti nel capotiolo 3: BERT, il modello di Kenneth Enevoldsen e MiniLM, con l'obiettivo di valutare se i modelli che integrano queste architetture fornissero risultati coerenti, ovvero migliori rispetto allo Zero Velocity (ZV), e se migliorassero le prestazioni del modello rispetto alla versione priva di descrizioni. Per quanto riguarda il modello di Kenneth Enevoldsen e MiniLM, l'embedding viene generato a partire dalle descrizioni stesse, mentre per BERT è necessario che le descrizioni vengano prima tokenizzate. Definita la struttura del modello di anticipazione, si passa ora al processo di ottimizzazione in fase di training, con l'obiettivo di ottenere la miglior configurazione possibile per ciascuna delle configurazioni testate.

5.5 Training per ricerca di ottimizzazione degli iperparametri

Un aspetto fondamentale nella creazione di un modello predittivo è la scelta degli iperparametri, che determinano la struttura del modello e influenzano il processo di apprendimento. Nel contesto dei modelli utilizzati in questo progetto, gli iperparametri considerati includono la dimensione dell'*hidden state*, il numero di *layers* e il tasso di apprendimento, detto *learning rate*. Quest'ultimo parametro stabilisce quanto devono essere grandi le modifiche ai parametri del modello durante l'aggiornamento. Esso è cruciale perché, se impostato troppo alto, potrebbe causare variazioni troppo grandi nei parametri, impedendo al modello di avvicinarsi a un minimo locale della *loss function*, tipicamente complessa in una Deep Neural Network (DNN).
Prima di eseguire la ricerca dell'ottima configurazione, però, si vuole individuare quale dei due modelli illustrati al sotto-paragrafo 5.4 risulta essere il più performante. Perciò, sono stati eseguiti dieci cicli di addestramento per ognuna delle due strutture, utilizzando l'embedding semantico prodotto da BERT. Ciascuno dei dieci cicli è stato eseguito con un diverso random_seed, per avere più

variabilità, per poi calcolare la media, la varianza e la deviazione standard dei migliori valori di validation loss di ogni ciclo. Riconosciuta la struttura più performante, risulta, inoltre, utile verificare intorno a quale epoca di addestramento si ottiene indicativamente il minimo valore della validation loss. Perciò, è stato eseguito un test di 1000 epoche, con uno dei modelli che integrano la semantica, come esempio di allenamento per visualizzare all'interno di quale range la validation loss raggiunge il valore minimo. Il risultato grafico dell'esempio di allenamento eseguito è riportato in Figura 5.4.

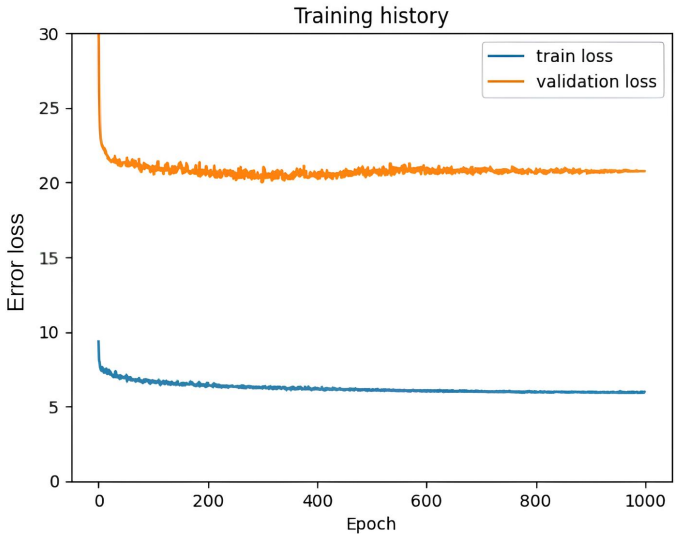

Figura 5.4: Esempio di training per ricerca del minimo su 1000 epoche

Si può osservare che il minimo è raggiunto attorno alle 250 epoche, per cui le prove eseguite di qui in avanti avranno un massimo di 300 epoche di addestramento.
Il divario della loss tra train e validation è dovuta alla tecnica di teacher forcing. Pertanto nel caso della loss sul training set, ad alcun step viene fornita la previsione corretta invece di quella precedente. Segue che, quando viene inviato il target, l'errore accumulato dalle previsioni degli step precedenti viene resettato. Mentre, se il Decoder ricevesse solo la previsione del passo precedente (come avviene nel validation set), l'errore si accumulerebbe a ogni step successivo. Questo spiega perché l'errore del training risulta essere circa un terzo di quello calcolato sul validation set.
Al fine della ricerca dell'ottima combinazione di iperparametri per le diverse configurazioni delle architetture (con e senza descrizioni), è stata eseguita un'a-

nalisi approfondita che ha esaminato tutte le possibili combinazioni di valori provenienti da tre liste distinte, ciascuna associata a uno degli iperparametri elencati precedentemente. Oltre a individuare la combinazione ottimale di iperparametri, questa ricerca ha permesso di comprendere come la curva di errore dipenda dagli stessi. In particolare, il parametro più critico è il learning rate: al suo diminuire, il rumore della curva si riduce, ma valori troppo bassi comportano un rallentamento significativo nel processo di apprendimento; diversamente, valori troppo alti portano ad avere grande rumore e causano anch'essi il rallentamento del processo di addestramento. Gli altri due parametri, ovvero hidden_dim e il numero di layer, mostrano un miglioramento del sistema con il loro aumento, sebbene la differenza diventi minima al crescere dei valori.

Questo metodo di ricerca dell'ottima combinazione dei parametri viene eseguita per ognuna delle quattro versioni del modello e per ognuna di esse si ottiene la combinazione più performante: le tre che integrano la semantica, RNN_BERT, RNN_KennethEnevoldsen e RNN_MiniML, e la versione senza semantica, RNN_senza_descrizioni. In particolare, per quest'ultima versione è necessario che l'analisi venga eseguita su un modello che utilizzi lo stesso identico dataset. Questa necessità vale per tutte le versioni, ma, dal momento che per quelle che integrano la semantica viene eseguito lo stesso preprocessing dei dataset, a variare è solo il modello che produce l'embedding delle descrizioni, perciò i dataset utilizzati da queste tre versioni sono identici. Il modello senza semantica, invece non subisce la parifica dei dataset, quindi è possibile fare l'errore di considerare l'intero dataset delle pose. Per tale motivo, è stata adottata la soluzione di eseguire la prima parifica dei dataset (ovvero quella riguardante i percorsi file) anche in questa versione, per poi mantenere solamente la lista delle pose. Grazie a questa soluzione, si mantengono le stesse sequenze di movimento delle altre versioni.
Per approfondire, si è deciso di analizzare anche come il sistema risponde alla variazione della dimensione dello hidden state del Description Embedder. A tal fine, è stato aggiunto un layer lineare che ridimensiona l'embedding delle descrizioni prima di concatenarlo con l'hidden state dell'Encoder. In Figura 5.5 è mostrato lo schema di tale modifica.

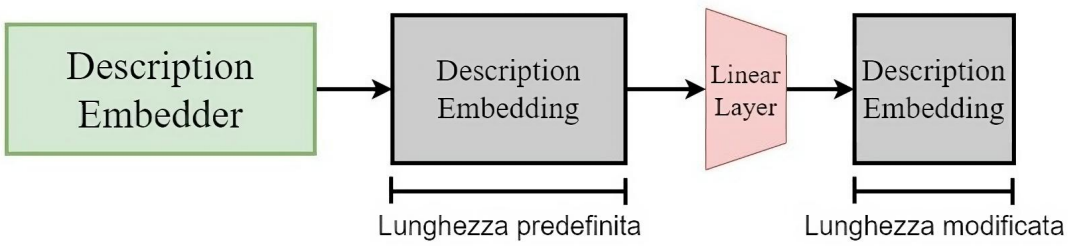

Figura 5.5: Schema modifica della lunghezza dell'embedding semantico

Compiuta l'ottimizzazione di tutte le versioni, segue quindi la ricerca condotta a valutare le performance di queste architetture e confrontarle con quella di un modello privo di descrizioni. Questo confronto offrirà una panoramica delle differenze tra i vari approcci, facilitando la scelta della configurazione più adatta alla predizione del moto umano. La valutazione viene eseguita sul test set, applicando le metriche definite nel capitolo 4, dove ogni modello ha la migliore configurazione degli iperparametri appena ottenuti, per i quali si è ottenuto il minimo errore sulla validation loss.

5.6 Approccio alternativo per dataset privi di descrizioni associate

Tutto ciò che è stato discusso e analizzato finora, vale solo ed esclusivamente con l'utilizzo dei dataset AMASS e BABEL, in quanto strettamente legati tra loro. Tuttavia, si è voluto analizzare una soluzione per i casi in cui vengono utilizzati dati di pose che non possiedono il corrispettivo dataset contenente le descrizioni di tali pose.
Nella prova di seguito proposta, è stato ancora una volta utilizzato il dataset ricavato dai file di "KIT" e "ACCAD" appartenenti ad AMASS. Tuttavia, non si fa più riferimento a BABEL. Come alternativa, è stato adottato un altro sistema di utilizzo delle descrizioni: si utilizza il nome con cui è stato salvato il file. Così facendo si ottengono descrizioni discretamente significative e, soprattutto, riduce al minimo la quantità di dati scartati dal dataset, causata significativamente dalla parifica dei dataset. Ogni nome del file ha una struttura di tipo "step_over_gap01_poses.npz" oppure "LeftTurn01_poses.npz", quindi, per ognuno di essi, si salva la stringa fino al primo valore numerico, si sostituiscono

i '_' con spazi vuoti e, nel caso di parole attaccate, si aggiungono spazi tra di esse. Con questo metodo, è possibile considerare tutti i mocap presenti nel dataset, ad esclusione di quelli il cui nome comincia per un numero - che nel caso di AMASS non presenta un problema consistente, ma se si usasse un dataset che utilizza una diversa metodologia di etichettare i mocap, allora è necessario adattare la logica allo specifico dataset. Oltre a quest'ultimo caso particolare, vengono anche tolti dal dataset i mocap che non hanno abbastanza frame per riempire la finestra di 75 frame. Terminate queste operazioni, è stato scelto di adottare il metodo semplice di ricavare un'unica finestra per mocap ad inizio casuale. Si ottiene, così, un dataset da 4095 finestre di movimento.
Vengono, dunque, eseguite le ricerche dell'ottima configurazione degli iperparametri per il modello sia con le nuove descrizioni, sia senza descrizioni, il quale si ricorda che deve contenere lo stesso identico dataset delle pose. Per il modello che utilizza questa specifica semantica, tuttavia, ci si è limitati a valutare l'efficacia solamente osservando il minimo valore di validation set, senza applicare le metriche al test set. L'obiettivo di questa variante, infatti, era solo di esplorare altre soluzioni di utilizzo delle descrizioni.

6

Risultati e discussione

Il presente capitolo discute i risultati degli esperimenti descritti nel capitolo 5, mirati a testare i diversi modelli per il calcolo degli embedding delle descrizioni, al fine di valutarne l'efficacia nel contesto della previsione del movimento umano (HMP). L'impostazione degli esperimenti e la configurazione degli iperparametri dei vari modelli si basano su quanto illustrato nel precedente capitolo.
Nei prossimi paragrafi, verranno discussi i risultati che evidenziano le scelte svolte nell'evoluzione dei metodi di preprocessing eseguiti per il dataset AMASS e il dataset BABEL, nonché per la scelta della struttura del modello. Verranno, inoltre, visualizzate le combinazioni che hanno portato ad ottenere il miglior valore della validation loss, ottenuto durante le epoche di addestramento per ogni modello. Tutto ciò, con l'obiettivo di confrontare le migliori configurazioni dei modelli sul test set, applicando le metriche di previsione sui 25 frame previsti, fornendo così una valutazione complessiva delle prestazioni ottenute.

6.1 Risultati test con il dataset ridotto

In prima analisi, vengono discussi i risultati ottenuti dal test sul dataset ridotto: questo test si basa sul dataset descritto nella sezione 5.1.1, derivante dall'integrazione delle pose contenute in "KIT" e "ACCAD", dalle quali è stata prelevata un'unica finestra da 75 frame; le descrizioni, invece, sono quelle relative alla categoria "seq_ann", come definito nel paragrafo 5.2.1. Come annunciato

nel capitolo precedente, questo test si è concentrato sull'esame delle descrizioni utilizzate, cercando di capire come si presentassero e di quanti dettagli fossero necessari per rappresentare adeguatamente i movimenti. Inoltre, il test era necessario per comprendere quale fosse la struttura più performante per l'integrazione della semantica nella predizione del moto umano. Per questa scelta è stata svolta l'analisi che calcola la media, la varianza e la deviazione standard dei migliori valori di validation loss ottenuti da dieci cicli di addestramento diversi. In Tabella 6.1 sono riportati i risultati di tali calcoli.

	Media	**Varianza**	**Deviazione standard**
Descrizioni tra Encoder e Decoder	20.6620	0.0183	0.1351
Descrizioni ad ogni input	22.5501	0.0242	0.1554

Tabella 6.1: Confronto con media, varianza e deviazione standard

Si può osservare che il modello che integra BABEL tra Encoder e Decoder (rappresentato in Figura 5.2) si dimostra il più performante tra i due della categoria che utilizza le descrizioni. Per tale motivo, nelle prossime analisi, verrà presa in considerazione solo questa struttura per l'integrazione delle descrizioni. Tuttavia, è fondamentale evidenziare che i test visualizzati sono stati eseguiti sul modello mantenendo sempre gli stessi iperparametri, senza effettuare confronti tra le migliori configurazioni per ciascuna struttura.
Determinata la struttura che adotta meglio l'integrazione delle descrizioni, si esegue il test sul dataset ridotto per confrontarlo al corrispettivo modello senza descrizioni. Da tale test si ottiene un valore della validation loss inferiore nel modello senza descrizioni: 18.179 per il modello con le descrizioni, contro 17.388 per il modello senza descrizioni. Questa circostanza, tuttavia, era prevedibile, poiché è possibile che la causa risieda nella scarsità di dettagli presenti nelle "sequence annotations" del dataset BABEL utilizzate: si ricorda che la gran maggioranza delle descrizioni utilizza una sola parola per descrivere l'intero mocap il quale contiene spesso sequenze di movimento complesse e variabili; inoltre, nel parificare i due dataset, si riduce di molto la quantità di dati utili. In aggiunta a questo, può anche verificarsi la possibilità che chi posa resta per molto tempo fermo (soprattutto a inizio o fine mocap), perciò i 75 frame estratti casualmente potrebbero contenere pose insignificanti (ovvero statiche) concatenate ad azioni. Un'altra situazione che può verificarsi, dal momento che ogni mocap può contenere diverse azioni che si susseguono, è che una descrizione così sintetica potrebbe non rispecchiare l'azione contenuta nella finestra.

O ancora finestre contenenti movimenti decisamente diversi (come una persona che cammina oppure una persona sdraiata a terra) possono essere legate alla stessa descrizione. Tutto ciò può portare il modello predittivo a "confondersi" e quindi la semantica non aggiunge alcuna informazione utile. Per affrontare questo problema, è stata adottata la metodologia per il preprocessing che, come definito nel paragrafo 5.1.2, utilizza descrizioni ricche di dettagli e più specifiche riguardo la finestra a cui sono legate. Nella successiva analisi verrà valutata l'efficacia dell'integrazione del dataset delle descrizioni, sfruttando le informazioni derivate dalle "frame annotations" del dataset BABEL.

6.2 Risultati test con il dataset completo e confronto delle prestazioni dei modelli

Il problema della qualità delle descrizioni, come discusso nel precedente paragrafo, ha portato alla conclusione che sono necessarie descrizioni precise e articolate affinché il loro contenuto fornisca informazioni significative e coerenti al modello. Pertanto, l'utilizzo delle "frame annotations" permette di risolvere questo problema. Inoltre, l'adozione di un dataset più esteso e l'applicazione di una finestra mobile per la selezione delle sequenze di frame ha consentito di ottenere un risultato più robusto e attendibile, garantendo una maggiore densità e copertura delle informazioni a disposizione del modello. Questo ha migliorato l'accuratezza e la coerenza delle analisi eseguite, consolidando ulteriormente la solidità del sistema di previsione. A seguito delle prove di ricerca per ottenere la migliore combinazione degli iperparametri, esposte al paragrafo 5.5, i rispettivi modelli hanno mostrato di essere ottimali con le serie di iperparametri riportate in Tabella 6.2. A fianco di ognuna combinazione, inoltre, è riportato il relativo valore minimo della validation loss ottenuto.

Dall'analisi dei valori di validation loss riportati in Tabella 6.2, si può notare che l'integrazione delle descrizioni semantiche nel modello ha portato a un miglioramento delle prestazioni rispetto alla versione senza descrizioni. In particolare, il modello RNN_MiniLM si distingue per aver ottenuto la validation loss più bassa (19.8216), dimostrando una maggiore accuratezza nella previsione. Questo risultato suggerisce che l'uso di modelli avanzati come MiniLM per la rappresentazione semantica delle descrizioni permette di ottenere una più efficace interpretazione delle informazioni testuali, migliorando la qualità com-

	hidden dim	**learning rate**	**n layers**	**description size**	**Migliore validation loss**
RNN_BERT	1024	0.001	4	256	20.5285
RNN_KennethEnevoldsen	1024	0.001	4	32	20.2542
RNN_MiniLM	1024	0.001	4	384	19.8216
RNN_senza_descrizioni	1024	0.001	3	/	20.6208

Tabella 6.2: Migliori combinazioni degli iperparametri e miglior validation loss per le tre configurazione del modello con la semantica integrata e per il modello senza la semantica

plessiva delle previsioni rispetto al modello base senza integrazione semantica. Anche RNN_BERT e RNN_KennethEnevoldsen, pur ottenendo risultati numerici leggermente superiori rispetto a RNN_MiniLM, si dimostrano comunque più performanti rispetto alla versione senza descrizioni, ma il miglioramento è stato più limitato. Nonostante ciò, tali risultati confermano l'importanza di un'adeguata rappresentazione semantica per il miglioramento del modello di anticipazione.

In particolare, questo risultato, dimostrando la superiorità di RNN_MiniLM nel contesto specifico, sottolinea che non tutte le tecniche di embedding delle descrizioni offrono la stessa efficacia.

Si tenga presente, però, che le osservazioni appena fatte si basano sui risultati ottenuti a seguito della fase di addestramento. Pertanto, per approfondire l'analisi, i modelli sono stati anche valutati sul test set per confrontare la performance delle architetture confrontata al modello base senza descrizioni. Questo confronto sarà effettuato utilizzando le metriche del Mean Per Joint Position Error (MPJPE), del Mean Angular Error (MAE), e della Joint Angle Difference (JAD), descritte nel capitolo 4. Questa analisi è di fondamentale importanza in quanto è necessario verificare se l'integrazione delle descrizioni possa effettivamente migliorare la qualità delle previsioni anche su dati non analizzati durante l'addestramento, ovvero quelli del test set. Questo passo è cruciale per determinare la robustezza delle architetture analizzate e la loro capacità di generalizzare oltre i dati di training (visualizzati molteplici volte), fornendo una valutazione complessiva della loro efficacia. Per questo motivo, non basta confrontare i valori della evaluation loss (calcolato tramite MSE), ma sono necessarie diverse metriche che analizzano i risultati di previsione da diverse prospettive.

I risultati delle metriche applicate in questa fase di evaluation permetteranno di confrontare in modo diretto l'impatto delle tecniche di embedding delle descrizioni e di trarre conclusioni definitive sull'utilità delle stesse nel migliorare le prestazioni dei modelli di previsione del movimento umano.
Le metriche utilizzate si basano sul calcolo cumulativo dell'errore a diversi step temporali, che vanno dai 120ms fino ai 1000ms. Questo approccio permette di valutare l'accuratezza delle previsioni in vari momenti nel tempo, offrendo una panoramica completa dell'efficacia dei modelli su differenti orizzonti temporali. Tale metodo è fondamentale per comprendere non solo la precisione immediata delle previsioni, ma anche la loro capacità di mantenere l'accuratezza su intervalli di tempo più lunghi, riflettendo meglio l'affidabilità delle tecnologie nel contesto HMP.
Un altro aspetto importante è verificare che tutte le versioni dei modelli producano risultati identici per ciascuna metrica calcolata sullo Zero Velocity (ZV). Ciò garantisce la coerenza delle valutazioni, poiché i modelli devono essere testati su dataset identici, partizionati nello stesso modo. Questo assicura che il confronto tra le diverse architetture sia equo e affidabile.
Un'altra prospettiva interessante, riguardo il modello ZV, è che questi valori servono come punto di riferimento per confrontare le performance dei modelli testati, consentendo di misurare se e quanto le tecnologie avanzate siano in grado di superare le previsioni statiche offerte da questo modello di riferimento.
Garantite queste premesse, sarà possibile effettuare una valutazione equa e precisa delle performance relative delle diverse tecnologie.
Di seguito vengono riportati in Tabella 6.3 i valori di MPJPE ottenuti per ogni modello testato. Questi dati offrono una base per valutare l'efficacia dei modelli nel ridurre l'errore di posizionamento rispetto alle previsioni statiche del modello ZV, evidenziando se e in che misura riescono a migliorare le prestazioni nei vari intervalli temporali analizzati.

Step temporali	120ms	200ms	400ms	600ms	800ms	1000ms
RNN_BERT	0.03651	0.05668	**0.09868**	**0.12479**	0.14369	0.16068
RNN_KennethEnevoldsen	0.03259	0.05821	0.10135	0.12660	0.14462	0.16063
RNN_MiniLM	**0.03168**	0.05767	0.10158	0.12587	**0.14345**	**0.15898**
RNN_senza_descrizioni	0.03191	**0.05637**	0.09885	0.12548	0.14533	0.16240
Zero-Velocity	0.04256	0.07949	0.14456	0.17872	0.19538	0.20161

Tabella 6.3: MPJPE: confronto tra modelli

Dall'analisi dei risultati riportati nella Tabella 6.3, emergono alcune osservazioni interessanti. In primo luogo, il modello RNN_MiniLM risulta essere il più performante in tre dei sei intervalli temporali, risultando il migliore nei primi 120ms, così come negli step a 800ms e 1000ms, dimostrando una capacità superiore di mantenere precisione nelle previsioni a lungo termine. Questo suggerisce che RNN_MiniLM riesce a catturare in modo più efficace le informazioni utili per le predizioni più distanti nel tempo.
Il modello RNN_senza_descrizioni, invece, risulta il migliore nei 200ms, che potrebbe indicare che per le previsioni di brevissimo termine le descrizioni semantiche non giocano un ruolo decisivo e il modello può fare affidamento su altre caratteristiche del dato di input.
Per quanto riguarda RNN_BERT, sebbene non sia sempre il migliore, eccelle particolarmente nei 400ms, dove fornisce la previsione più accurata. Questo suggerisce che RNN_BERT potrebbe avere un vantaggio nella gestione di intervalli temporali intermedi, anche se non riesce a mantenere lo stesso livello di precisione nei frame più distanti. Tuttavia, sia RNN_BERT che RNN_KennethEnevoldsen superano il modello RNN_senza_descrizioni negli ultimi due intervalli temporali, dimostrando che l'integrazione di descrizioni semantiche diventa più significativa per migliorare la qualità delle previsioni man mano che gli step temporali aumentano. Questi risultati potrebbero confermare che l'aggiunta delle descrizioni porta benefici nel lungo periodo, dove l'informazione contestuale fornita dagli embedding semantici diventa cruciale per mantenere la coerenza delle previsioni.
Inoltre, si noti che tutti i modelli con descrizioni e quello senza descrizioni ot-

tengono risultati significativamente migliori rispetto al modello ZV in tutti gli intervalli temporali. Questo risultato indica che anche con l'aggiunta di ulteriori informazioni, come quelle semantiche, i modelli sono in grado di produrre previsioni coerenti e non casuali, dimostrando che l'integrazione delle descrizioni contribuisce effettivamente a un miglioramento delle performance, mantenendo la robustezza del sistema di previsione.
Infine, per ottenere una valutazione più completa delle prestazioni dei modelli, è necessario considerare anche le altre due metriche di valutazione, così da fornire un quadro più esaustivo dell'efficacia delle descrizioni semantiche.

In aggiunta all'analisi tramite la metrica MPJPE, è stato eseguito un ulteriore confronto utilizzando la metrica del MAE, che si concentra sugli errori angolari nelle pose umane. Vengono riportati in Tabella 6.4 i valori di MAE ottenuti che mettono in confronto i tre modelli con semantica con il modello senza descrizioni.

Step temporali	120ms	200ms	400ms	600ms	800ms	1000ms
RNN_BERT	0.40876	0.64583	0.97358	**1.15544**	**1.27204**	1.39784
RNN_KennethEnevoldsen	0.37856	0.65375	0.98388	1.17546	1.29496	1.40555
RNN_MiniLM	0.36963	0.63572	0.98357	1.15818	1.27739	**1.36726**
RNN_senza_descrizioni	**0.36138**	**0.61480**	**0.95020**	1.17069	1.29302	1.41285
Zero-Velocity	0.44462	0.76947	1.25562	1.49267	1.60481	1.65907

Tabella 6.4: MAE: confronto tra modelli

Dalla lettura dei risultati riportati nella Tabella 6.4, emergono nuove osservazioni rilevanti. In questo caso, il modello RNN_senza_descrizioni si dimostra il più performante nelle prime tre finestre temporali (120ms, 200ms, e 400ms), con il valore di MAE più basso, il che indica una maggiore accuratezza nelle previsioni a breve termine. Tuttavia, con l'aumentare dello step temporale, il vantaggio del modello RNN_senza_descrizioni si riduce.
A partire dai 600ms, il modello RNN_BERT diventa il più preciso, dimostrando un vantaggio nella coerenza nelle previsioni nel medio termine (600ms e 800ms), mentre RNN_MiniLM si distingue per il miglior risultato nell'ultimo step temporale (1000ms). In particolare, si vuole far notare che RNN_MiniLM

è comunque più preciso di RNN_senza_descrizioni a partire dai 600ms, come lo sono RNN_BERT e RNN_KennethEnevoldsen a 1000ms. Anche in questo caso i risultati suggeriscono che, nonostante le prestazioni iniziali del modello RNN_senza_descrizioni siano migliori, l'integrazione delle descrizioni diventa cruciale per previsioni più distanti nel tempo, dove le informazioni semantiche contribuiscono a mantenere previsioni più accurate.
Un'ulteriore osservazione è che tutti i modelli ottengono risultati migliori rispetto al modello ZV, confermando nuovamente che l'integrazione di informazioni aggiuntive non solo migliora significativamente la qualità delle previsioni rispetto all'approccio di riferimento, ma produce previsioni plausibili.
La migliore performance della metrica MPJPE rispetto alla MAE a 120ms potrebbe derivare dal fatto che i modelli di previsione effettuano i calcoli in coordinate angolari. Valutare il movimento tramite rotazioni articolari con la metrica MAE potrebbe quindi favorire i modelli senza descrizioni, che risultano più efficienti nel catturare movimenti semplici senza l'influenza del contesto semantico. Di contro, la MPJPE, basandosi su variazioni spaziali, beneficia delle descrizioni anche nel breve termine, evidenziando meglio le differenze nelle posizioni 3D.

Al termine dell'analisi basata su MPJPE e MAE, si considera ora la metrica della JAD. Il confronto dei modelli secondo tale metrica viene riportato in Tabella 6.5.

Step temporali	120ms	200ms	400ms	600ms	800ms	1000ms
RNN_BERT	0.07437	0.11177	**0.18574**	**0.23209**	**0.26515**	0.29573
RNN_KennethEnevoldsen	0.06370	0.11264	0.19086	0.23651	0.26836	0.29644
RNN_MiniLM	0.06245	0.11198	0.19059	0.23440	0.26558	**0.29257**
RNN_senza_descrizioni	**0.06237**	**0.10899**	0.18636	0.23368	0.26845	0.29922
Zero-Velocity	0.07749	0.14337	0.25718	0.31720	0.34494	0.35594

Tabella 6.5: JAD: confronto tra modelli

Dall'analisi dei risultati riportati in Tabella 6.5, emergono alcune osservazioni simili a quelle fatte per le altre metriche. Il modello RNN_senza_descrizioni si dimostra ancora una volta il più performante a breve termine (120ms e 200ms), con il valore di JAD più basso. Questo suggerisce che, su orizzonti temporali bre-

vi, l'aggiunta di descrizioni semantiche non fornisce un contributo significativo alla predizione e, anzi, introduce una complessità aggiuntiva che può ostacolare il modello, riducendo l'efficacia rispetto a un'architettura più semplice.
A partire dai 400ms, il modello RNN_BERT risulta il più accurato, confermando la sua efficacia nella gestione delle previsioni a medio termine. Questo è coerente con i risultati delle metriche precedenti, dove BERT ha mostrato una maggiore capacità di mantenere la somiglianza del movimento umano nei range temporali intermedi.
Infine, per le previsioni a lungo termine, RNN_MiniLM si distingue con il miglior risultato all'ultimo step temporale (1000ms). Anche in questo caso, si noti che tutti e tre i modelli con descrizioni semantiche migliorano nelle performance di previsione a lungo termine. Ciò dimostra la conferma della tendenza secondo cui aggiungere l'informazione della semantica porti al superamento delle performance del modello senza descrizioni man mano che gli step temporali aumentano, evidenziando l'importanza del contesto semantico nel lungo termine. Infine, ancora una volta, tutti i modelli, indipendentemente dalla presenza o meno di descrizioni, ottengono risultati migliori rispetto al modello ZV, confermando che l'integrazione di caratteristiche avanzate produce risultati attendibili.

In conclusione, i risultati ottenuti evidenziano alcune tendenze che permettono di comprendere l'efficacia dei modelli testati.
In primo luogo, l'evaluation sul test set conferma quanto già osservato nel validation set. I modelli che includono descrizioni semantiche, come RNN_BERT e RNN_MiniLM, risultano migliori rispetto al modello RNN_senza_descrizioni per predizioni a lungo termine. Questo dimostra la loro capacità di sfruttare informazioni semantiche per migliorare la predizione del movimento umano. Tuttavia, non tutti i modelli con descrizioni sono ugualmente performanti: RNN_BERT e RNN_MiniLM si distinguono per i migliori risultati ottenuti, ma anche RNN_KennethEnevoldsen tende a migliorare nei lunghi periodi rispetto a RNN_senza_descrizioni, sebbene non sia mai il più vantaggioso. In particolare, risulta ben evidente la supremazia di RNN_BERT per periodi di media durata e quella di RNN_MiniLM per il lungo termine. Le differenze di performance tra questi due modelli possono essere spiegate dalle loro diverse architetture e modalità di gestione degli embedding semantici. BERT, con i suoi embedding di 768 dimensioni ridotti a 256 tramite un layer lineare, riesce a catturare ricche informazioni semantiche nel medio termine, ma la compressione potrebbe

causare una perdita di dettagli utili a lungo termine. MiniLM, che produce embedding più compatti da 384 dimensioni senza riduzioni ulteriori, mantiene una maggiore stabilità e continuità, risultando più performante nelle previsioni a lungo termine. Questo suggerisce che un modello più leggero come MiniLM è più adatto a mantenere la coerenza su orizzonti temporali estesi, mentre BERT eccelle nel medio periodo grazie alla sua capacità di catturare contesti complessi iniziali. In particolare, si vuole far notare che l'architettura RNN impiegata in questo lavoro è significativamente più ridotta rispetto alle strutture più complesse dei modelli LLM, in particolare rispetto a BERT. Questo fattore potrebbe spiegare perché MiniLM, con una struttura più semplice e compatta, dimostri prestazioni superiori, riuscendo a trovare un equilibrio ottimale tra semplicità e performance. Nonostante tale discussione, in generale, se nel breve termine (specialmente per MAE) i modelli senza descrizioni tendono a ottenere risultati migliori, per intervalli temporali più lunghi l'aggiunta della semantica performa di più, seppur limitatatamente. Questo comportamento può essere spiegato dalla natura delle descrizioni semantiche, che offrono un contesto più ricco e utile nel lungo periodo. Le informazioni semantiche, infatti, contribuiscono a rendere più coerente la rappresentazione delle transizioni tra diverse azioni lungo il mocap, migliorando la capacità del modello di catturare l'evoluzione del movimento umano. Descrivere l'azione tramite embedding semantici aiuta pertanto i modelli a catturare meglio la progressione temporale e l'evoluzione delle pose.

Un altro aspetto interessante da notare, riguarda il fatto che tutti i modelli testati performano meglio rispetto al modello ZV. Questo risultato dimostra che i modelli non generano previsioni casuali, ma anzi producono output sensati e utili, confermando la loro capacità di apprendere e generalizzare il movimento umano anche senza l'uso di descrizioni semantiche.

Infine, i risultati tra le diverse metriche sono stati in gran parte coerenti, dove i modelli performano meglio o peggio in modo consistente. L'eccezione è rappresentata dalla deviazione a 120ms nella metrica MPJPE, dove RNN_MiniLM ha avuto una prestazione migliore rispetto agli altri modelli. Questa discrepanza potrebbe essere causata dalla differente natura delle metriche.

In sintesi, l'analisi complessiva suggerisce che l'integrazione delle descrizioni semantiche migliora le performance nei contesti di lungo periodo, dove le informazioni contestuali diventano cruciali per mantenere previsioni più accurate e coerenti, mentre nel breve termine i modelli più semplici possono avere un

vantaggio grazie alla loro maggiore efficienza nell'elaborazione di movimenti immediati.

6.3 Risultati del modello con descrizioni derivate dai nomi dei file

In ultima analisi, sono riportati i risultati dei valori di validation loss ottenuti da un modello che integra le descrizioni derivanti dai nomi con cui i file stessi sono stati etichettati. L'architettura del modello (riportata in Figura 5.2) risulta essere la stessa utilizzata per i test di cui sopra definiti.
In questo esperimento è stato scelto di utilizzare BERT per via della sua capacità di catturare relazioni semantiche complesse anche in descrizioni brevi. Nonostante i nomi dei file utilizzati come descrizioni siano meno dettagliati rispetto alle "frame annotations" di BABEL, BERT si dimostra efficace nell'estrarre rappresentazioni ricche e utili anche da input limitati, garantendo comunque un buon livello di comprensione del contesto.
Di seguito, vengono riportati in Tabella 6.6 le rispettive combinazioni degli iperparametri ed il miglior valore di validation loss ottenuto dai due modelli (con e senza descrizioni).

	hidden dim	**learning rate**	**n layers**	**description size**	**Migliore validation loss**
RNN_BERT	512	0.001	4	32	8.3561
RNN_senza_descrizioni	1024	0.001	3	/	8.5097

Tabella 6.6: Migliori combinazioni degli iperparametri e miglior validation loss per le due configurazione del modello con la semantica integrata tramite l'approccio alternativo e per il modello senza la semantica

Dalla Tabella 6.6, si osserva che il modello RNN_BERT ha ottenuto il valore di validation loss migliore. Ciò suggerisce che l'integrazione delle descrizioni attraverso i nomi dei file ha contribuito a una maggiore efficienza nel processo di apprendimento. Sebbene il miglioramento non sia così significativo come quello delle configurazioni precedenti che utilizzano BABEL, bisogna far notare che in questo caso il dataset è di molto ridotto, quindi i due test non sono direttamente confrontabili. Nonostante ciò, questo risultato dimostra come l'utilizzo di descrizioni più dettagliate possa portare ad un miglioramento. Confrontando,

infatti, questo test con quello eseguito che integra le "sequence annotations" di BABEL (paragrafo 6.1), il quale non ha dato buoni risultati, si può affermare che la semantica aiuta a migliorare il sistema, purché sia dettagliata.

In conclusione, l'esperimento ha dimostrato che anche descrizioni derivate in modo alternativo, sebbene non ottimali come quelle fornite da BABEL, possano ancora fornire un contributo positivo alla qualità delle previsioni. Questi risultati rafforzano l'ipotesi che, anche in assenza di un dataset dedicato per le descrizioni, sia possibile adottare soluzioni come l'utilizzo dei nomi dei file per migliorare le performance dei modelli di previsione delle pose.

7

Conclusioni

In questa tesi è stato esplorato un approccio innovativo nell'ambito della Human Motion Prediction (HMP), basato sull'integrazione di informazioni semantiche. Questo metodo utilizza modelli di DL che incorporano LLMs per gestire congiuntamente i dati delle pose umane e le descrizioni semantiche delle azioni. Questo approccio è stato sviluppato poiché la semplice previsione delle pose risulta insufficiente a causa della complessità e dell'imprevedibilità dei movimenti umani. Nel corso del progetto, è stato investigato se e come l'integrazione di descrizioni semantiche possa migliorare la qualità delle previsioni rispetto ad un modello senza semantica. Questo risulta di cruciale importanza per molte applicazioni pratiche, come la guida autonoma, dove è fondamentale prevedere con anticipo i movimenti di pedoni o veicoli per evitare incidenti; la prevenzione di infortuni nello sport, prevedendo movimenti che potrebbero causare danni fisici; la robotica in ambienti dinamici, consentendo ai robot di adattarsi alle traiettorie umane per evitare collisioni; e la riabilitazione motoria, dove si può monitorare e ottimizzare il recupero dei pazienti prevedendo i loro movimenti.
Il progetto si è basato sull'utilizzo di due dataset principali: Archive of Motion Capture As Surface Shapes (AMASS) per le pose e Bodies, Action and Behavior with English Labels (BABEL), che fornisce le descrizioni semantiche dei dati presenti in AMASS. La natura diretta della relazione tra i due dataset ha facilitato la creazione di un'architettura integrata che sfrutta le informazioni semantiche per potenziare la capacità predittiva del modello. In questa fase, è stato impiegato un modello noto per la sua efficacia nella previsione di sequenze temporali,

in particolare quello della Recurrent Neural Network (RNN), con l'obiettivo di esplorare come l'aggiunta di informazioni semantiche potesse influenzare la qualità delle previsioni.
Per quanto riguarda la gestione delle descrizioni semantiche, si sono utilizzati vari modelli di linguaggio naturale basati su Transformer, tra cui BERT, KennethEnevoldsen e MiniLM. Questi modelli sono stati scelti per la loro capacità di catturare strutture grammaticali e semantiche, con l'obiettivo di individuare quale di questi fornisse la migliore rappresentazione delle descrizioni.
L'analisi condotta per definire il setup sperimentale ha esplorato diverse modalità di gestione dei dataset, con una suddivisione in step che ha permesso di studiare con maggiore accuratezza sia il dataset che la struttura del modello, per confrontare diversi approcci di integrazione delle descrizioni. Il primo test, svolto su un dataset ridotto, ha permesso di determinare la struttura del modello più efficace per integrare le informazioni semantiche: è emerso che la concatenazione dell'embedding semantico con l'hidden state dell'Encoder, e il passaggio del vettore concatenato al Decoder, è stata la soluzione più promettente. Tuttavia, a causa dell'utilizzo di descrizioni poco dettagliate, che in determinati casi possono rivelarsi incoerenti, i risultati ottenuti non sono stati migliori rispetto al modello senza semantica. Per tale motivo, sono stati eseguiti test su un dataset più ampio e dettagliato, utilizzando le "frame annotations" di BABEL, che offrono informazioni contestuali ricche e precise sulle pose del movimento umano. A seguito di questi test, sono state condotte approfondite analisi per ottimizzare gli iperparametri e valutare le performance sul test set. Queste analisi hanno mostrato chiaramente che l'integrazione delle descrizioni semantiche contribuisce a migliorare le performance del modello, in particolare nelle previsioni a lungo termine. Le metriche di valutazione, come Mean Per Joint Position Error (MPJPE), Mean Angular Error (MAE) e Joint Angle Difference (JAD), hanno evidenziato che tutte i modelli semantici (BERT, MiniLM, KennethEnevoldsen) apportano benefici, ma con differenze temporali. BERT si è rivelato il migliore a medio termine (tra i 400ms e gli 800ms), mentre MiniLM ha mostrato una superiorità nelle previsioni su orizzonti temporali più lunghi (1000ms).
Un altro risultato interessante è emerso dall'esperimento che ha esplorato l'utilizzo di descrizioni derivate dai nomi dei file. Nonostante queste descrizioni siano meno dettagliate rispetto a quelle di BABEL, i test hanno comunque mostrato un miglioramento della performance rispetto al modello privo di descrizioni.

Questo suggerisce che, anche in assenza di dataset dedicati per le descrizioni semantiche, è possibile sfruttare approcci alternativi come l'utilizzo di informazioni implicite nei nomi dei file. Questo apre nuove possibilità per l'utilizzo della semantica in contesti dove non sono disponibili dataset strutturati per la descrizione delle pose.
Dal punto di vista delle criticità, è importante notare che i miglioramenti osservati, pur significativi, restano ancora limitati. La differenza tra i modelli con e senza descrizioni è certamente apprezzabile, soprattutto nel lungo termine, ma suggerisce anche che c'è un ampio margine di miglioramento. Un aspetto rilevante emerso dai test, infatti, è la qualità delle descrizioni. I risultati hanno dimostrato che descrizioni più dettagliate portano a migliori performance predittive. Pertanto, si può ipotizzare che un dataset meglio fornito e con descrizioni più ricche e specifiche possa ottimizzare ulteriormente le performance del modello. In questo senso, BABEL rappresenta un buon punto di partenza, ma l'utilizzo di dataset ancora più completi, che includano descrizioni già dettagliate e complete dell'intero mocap, senza la necessità di ricostruire le descrizioni da porzioni di frame, potrebbe costituire un significativo passo avanti. In particolare, è importante notare che la rete RNN utilizzata è abbastanza semplice, e quindi esplorare modelli con un potenziale maggiore potrebbe rivelarsi interessante. Inoltre, future ricerche potrebbero esplorare diverse modalità di integrazione delle informazioni semantiche, andando oltre la semplice concatenazione degli embedding, per verificare se porta a migliorare ulteriormente l'efficacia delle previsioni. Dal momento che i risultati sono stati favorevoli, un lavoro futuro potrebbe essere quello di integrare questa metodologia all'interno dell'architettura Position-Velocity Recurrent Encoder-Decoder (PVRED), nota per le sue capacità avanzate in questo contesto.
In conclusione, questa tesi dimostra il potenziale dell'integrazione tra le informazioni delle pose e quelle semantiche del movimento, evidenziando come l'utilizzo di tecnologie avanzate per la gestione del linguaggio naturale possa arricchire i modelli di previsione del movimento umano, in particolare nelle fasi di transizione tra le azioni. Questo aspetto è fondamentale, poiché la capacità di prevedere accuratamente i movimenti su un orizzonte temporale esteso si basa sulla comprensione di queste transizioni, che sono cruciali per una previsione efficace a lungo termine. Le varie prove eseguite suggeriscono anche che ulteriori miglioramenti potrebbero essere ottenuti con l'uso di dataset più ricchi, ottimizzando ulteriormente le prestazioni del modello.

Riferimenti

[1] Tom B Brown et al. «Language Models are Few-Shot Learners». In: *arXiv preprint arXiv:2005.14165* (2020).

[2] Carnegie Mellon University Motion Capture Database. *CMU Graphics Lab Motion Capture Database*. Accessed: 2024-09-20. URL: http://mocap.cs.cmu.edu/.

[3] W. Chen et al. «A Spatio-temporal Transformer for 3D Human Motion Prediction». In: *arXiv* (2020). Accessed: 2024-07-29. URL: https://arxiv.org/abs/2004.08692.

[4] Yuxi Chen et al. *HumanMAC: Masked Motion Completion for Human Motion Prediction*. Accessed: 2024-07-26. 2023. URL: https://openaccess.thecvf.com/content/ICCV2023/html/Chen_HumanMAC_Masked_Motion_Completion_for_Human_Motion_Prediction_ICCV_2023_paper.html.

[5] Junyoung Chung et al. «Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling». In: *arXiv preprint arXiv:1412.3555* (2014). URL: https://arxiv.org/pdf/1412.3555.

[6] Jacob Devlin et al. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». In: *arXiv preprint arXiv:1810.04805* (2019). URL: https://arxiv.org/abs/1810.04805.

[7] Google Drive. *Cartella dei file gif su Google Drive*. Accessed: 2024-09-20. 2024. URL: https://drive.google.com/drive/folders/1beYR6xRsgkZApxOHrFOGcfzw4VEIaD0C?usp=drive_link.

[8] *Google Drive Folder Link*. https://drive.google.com/drive/folders/1VIrVVPbb2qlt4_Y8cEI21EcYHaO6XLsy?usp=drive_link. Accessed: 2024-08-30. 2024.

[9] Jianguo Gou e Shuang Zhu. «PVRED: A Position-Velocity Recurrent Encoder-Decoder for Human Motion Prediction». In: *arXiv preprint arXiv:1906.06514* (2019). URL: https://arxiv.org/abs/1906.06514.

[10] Catalin Ionescu et al. «Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1325–1339. DOI: 10.1109/TPAMI.2013.248. URL: https://ieeexplore.ieee.org/document/6682899.

[11] Catalin Ionescu et al. «Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1325–1339.

[12] Peter Schneider-Kamp Kenneth Enevoldsen Dan Saattrup Nielsen. «Encoder vs Decoder: Comparative Analysis of Encoder and Decoder Language Models on Multilingual NLU Tasks». In: *arXiv preprint arXiv:2406.13469* (2024). URL: https://arxiv.org/abs/2406.13469.

[13] Pawel Korus e Jun Zhang. *Recurrent Neural Networks: A Gentle Introduction and Overview*. Available at https://arxiv.org/pdf/1912.05911. 2019. arXiv: 1912.05911 [cs.LG].

[14] Alex Krizhevsky, Ilya Sutskever e Geoffrey E Hinton. «ImageNet classification with deep convolutional neural networks». In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[15] Y. Z. Lin et al. «A Comprehensive Review on Human Motion Prediction». In: *arXiv preprint arXiv:2301.01011* (2023). URL: https://arxiv.org/abs/2301.01011.

[16] Hao Liu et al. «Human Motion Prediction Using Recurrent Neural Networks». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 835–843. URL: https://arxiv.org/abs/1603.09520.

[17] Yinhan Liu et al. «RoBERTa: A Robustly Optimized BERT Pretraining Approach». In: *arXiv preprint arXiv:1907.11692* (2019).

[18] Naureen Mahmood et al. «AMASS: Archive of Motion Capture as Surface Shapes». In: *International Conference on Computer Vision*. Ott. 2019, pp. 5442–5451.

[19] Julieta Martinez, Michael J. Black e Javier Romero. «On Human Motion Prediction Using Recurrent Neural Networks». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2891–2900. URL: https://openaccess.thecvf.com/content_cvpr_2017/papers/Martinez_On_Human_Motion_CVPR_2017_paper.pdf.

[20] Anish Nama. *Understanding LSTM Architecture: Pros and Cons and Implementation*. Accessed: 2024-08-02. 2021. URL: https://medium.com/@anishnama20/understanding-lstm-architecture-pros-and-cons-and-implementation-3e0cca194094.

[21] Sungheon Park, Jihye Hwang e Nojun Kwak. «3D Human Pose Estimation Using Convolutional Neural Networks with 2D Pose Information». In: *arXiv preprint arXiv:1608.03075* (2016). URL: https://arxiv.org/abs/1608.03075.

[22] Sundar Pichai. *An important next step on our AI journey*. https://blog.google/intl/en-africa/products/explore-get-answers/an-important-next-step-on-our-ai-journey/. Accessed: 2024-08-18. 2023.

[23] Abhinanda R. Punnakkal et al. «BABEL: Bodies, Action and Behavior with English Labels». In: *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*. Giu. 2021, pp. 722–731.

[24] Alec Radford et al. «Language Models are Unsupervised Multitask Learners». In: *OpenAI Blog* 1.8 (2019), p. 9.

[25] Andrey Rudenko et al. «Human Motion Trajectory Prediction: A Survey». In: *International Conference on Robotics and Automation (ICRA)*. 2019.

[26] Victor Sanh et al. «DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter». In: *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (NeurIPS 2019)*. 2019. URL: https://arxiv.org/abs/1910.01108.

[27] Xiaodan Song et al. «Human Motion Prediction with Generative Adversarial Networks». In: *Proceedings of the European Conference on Computer Vision (ECCV)*. European Conference on Computer Vision, 2018, pp. 850–865. URL: https://arxiv.org/abs/1806.02525.

[28] Dom Soria. *Cosa è la back-propagation dell'errore?* Accessed: 2024-07-26. 2021. URL: https://www.domsoria.com/2021/02/cosa-e-la-back-propagation-dellerrore/.

[29] Jianwei Tang, Jieming Wang e Jian-Fang Hu. «Predicting human poses via recurrent attention network». In: *Journal of Computer Science and Technology* (2023), pp. 88–102. DOI: 10.1007/s11390-023-00020-z. URL: file:///C:/Users/pmerc/Downloads/s44267-023-00020-z.pdf.

[30] Hugo Touvron et al. «LLaMA: Open and Efficient Foundation Language Models». In: *arXiv preprint arXiv:2302.13971* (2023). URL: https://arxiv.org/abs/2302.13971.

[31] Ashish Vaswani et al. «Attention is all you need». In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[32] Wenhui Wang et al. «MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers». In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[33] Ruiqi Yu et al. «Learning Temporal Patterns in Human Motion with LSTM Networks». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 5126–5134. URL: https://arxiv.org/abs/1604.06510.

[34] Wayne Xin Zhao et al. «A Survey of Large Language Models». In: *arXiv preprint arXiv:2303.18223* (2023). URL: https://arxiv.org/abs/2303.18223.