Analisi degli Embedding ATC tramite Graph Neural Networks

Abstract

Questo report presenta un'analisi degli embedding dei codici ATC (Anatomical Therapeutic Chemical) ottenuti tramite Graph Neural Networks (GNN). A differenza dell'approccio precedente basato su embedding gerarchici concatenati, questo metodo sfrutta esplicitamente la struttura a grafo del sistema di classificazione ATC, utilizzando Graph Attention Networks (GAT) per apprendere rappresentazioni vettoriali di 128 dimensioni che rispettino le relazioni parent-child della gerarchia farmacologica. L'analisi valuta la qualità degli embedding attraverso metriche quantitative di clustering, coerenza semantica locale e preservazione della struttura gerarchica.

Contents

1 Introduzione

1.1 Contesto

Il sistema di classificazione ATC (Anatomical Therapeutic Chemical) organizza i farmaci in una struttura gerarchica a cinque livelli, dove ogni livello rappresenta un grado crescente di specificità farmacologica. Questa struttura intrinsecamente gerarchica suggerisce una rappresentazione naturale dei dati come grafo orientato, dove ogni nodo rappresenta un codice ATC e gli archi codificano le relazioni parent-child tra i livelli.

1.2 Approccio basato su grafi

Le Graph Neural Networks (GNN) offrono un paradigma più naturale per modellare dati gerarchici:

- **Rappresentazione esplicita della gerarchia**: gli archi del grafo codificano direttamente le relazioni parent-child
- **Message passing**: l'informazione fluisce attraverso la struttura del grafo, permettendo a ogni nodo di aggregare informazioni dai suoi vicini
- **Apprendimento contestuale**: le rappresentazioni dei nodi vengono apprese considerando il loro contesto strutturale all'interno della gerarchia

1.3 Obiettivo dell'analisi

L'obiettivo principale di questo lavoro è apprendere embedding vettoriali densi di 128 dimensioni per ciascun codice ATC che:

1. **Rispettino la gerarchia**: nodi parent-child devono avere embedding vicini nello spazio vettoriale
2. **Separino le categorie**: farmaci di macro-categorie anatomiche diverse devono essere rappresentati da vettori distanti
3. **Preservino il livello**: la profondità nella gerarchia deve essere codificata negli embedding

Questi obiettivi vengono perseguiti attraverso un'architettura basata su Graph Attention Networks (GAT) con tre componenti di loss complementari: classificazione della categoria anatomica, classificazione del livello gerarchico e una loss gerarchica che avvicina gli embedding di nodi parent-child.

2 Metodologia

2.1 Costruzione del grafo ATC

2.1.1 Caricamento e pre-processing dei dati

Il processo inizia con il caricamento del dataset ATC da un file CSV contenente i seguenti campi:

- **code**: codice ATC (es. `A10BA02`)
- **parent**: codice del nodo parent nella gerarchia (es. `A10BA`)
- **level**: livello gerarchico (1-5)

La fase di pre-processing include:

1. **Normalizzazione**: applicazione di `upper()` e `strip()` per uniformare i codici
2. **Gestione valori mancanti**: sostituzione dei valori `NaN` con stringhe vuote
3. **Rimozione duplicati**: eliminazione di eventuali codici duplicati mantenendo la prima occorrenza
4. **Filtraggio**: rimozione di righe con codici vuoti

2.1.2 Mapping dei nodi e creazione indici

Viene creato un dizionario che mappa ogni codice ATC a un indice numerico univoco:

$$\text{atc\_idx} : \{\text{codice ATC} \rightarrow \text{node\_idx}\}$$

Questo mapping è fondamentale per l'indicizzazione efficiente dei nodi nel grafo e per la costruzione dei tensori PyTorch. Ogni codice viene associato a un indice intero progressivo da 0 a $N - 1$, dove N è il numero totale di codici ATC unici nel dataset.

2.1.3 Costruzione degli archi

Gli archi del grafo rappresentano le relazioni gerarchiche parent-child. Per ogni codice ATC che ha un parent definito:

1. Si identifica l'indice del nodo child: `node_idx`
2. Si identifica l'indice del nodo parent: `parent_idx`
3. Si crea un arco orientato: `child` $\rightarrow$ `parent`

Il risultato è un tensore di shape $(2, E)$, dove E è il numero di archi:

$$\text{edge\_atc} = \begin{bmatrix} \text{child\_indices} \\ \text{parent\_indices} \end{bmatrix}$$

Esempio di archi Per i codici della metformina:

- `A10BA02` → `A10BA` (sostanza → gruppo chimico)
- `A10BA` → `A10B` (gruppo chimico → sottogruppo)
- `A10B` → `A10` (sottogruppo → gruppo terapeutico)
- `A10` → `A` (gruppo terapeutico → categoria anatomica)

2.1.4 Creazione grafo bidirezionale

Per permettere il message passing in entrambe le direzioni (parent → child e child → parent), vengono aggiunti anche gli archi inversi:

$$\text{edge\_index\_atc} = [\text{edge\_atc}, \text{edge\_atc}_{\text{reversed}}]$$

Questo consente alla rete neurale di propagare informazioni sia verso l'alto (aggregando informazioni dai figli) che verso il basso (ricevendo informazioni dai parent).

2.2 Grafo eterogeneo con PyTorch Geometric

Il grafo viene rappresentato utilizzando la struttura `HeteroData` di PyTorch Geometric, che supporta grafi con tipi di nodi e archi multipli. Nel nostro caso:

- **Tipo di nodo**: `"atc"` (tutti i nodi rappresentano codici ATC)
- **Tipo di arco**: `("atc", "rel_atc", "atc")` (relazioni gerarchiche tra codici ATC)
- **Numero di nodi**: 6440 codici ATC unici
- **Feature dei nodi**: matrice $X \in \mathbb{R}^{N \times d_{\text{in}}}$ (inizialmente basata sul livello), dove N indica il numero totale di nodi ATC presenti nel grafo.

2.3 Feature engineering: arricchimento delle feature dei nodi

Le feature iniziali devono catturare sia informazioni gerarchiche sia topologiche. Un semplice one-hot encoding del livello è insufficiente perché non descrive la posizione strutturale del nodo nel grafo (es. quanti figli ha un nodo, se è una foglia o un nodo interno, quanto è centrale nella topologia).

2.3.1 Componenti delle feature

Ogni nodo è descritto da un vettore $\mathbf{x}_v \in \mathbb{R}^8$ ottenuto concatenando quattro componenti complementari:

- **One-hot encoding del livello** (5 dim):

$$\mathbf{e}_{\text{level}(v)-1} \in \{0,1\}^5$$

Codifica categoriale del livello gerarchico (1–5). Permette al modello di distinguere immediatamente la profondità del nodo nella gerarchia ATC. Ad esempio, un farmaco specifico (livello 5) avrà pattern di attivazione diversi rispetto a una categoria anatomica (livello 1).

- **Grado entrante normalizzato** (1 dim):

$$\text{in\_deg\_norm}(v) = \log(1 + |\{u : (u, v) \in E\}|)$$

Numero di nodi child che puntano a v, stabilizzato logaritmicamente. La trasformazione logaritmica $\log(1 + x)$ comprime i valori elevati e previene che nodi con grado molto alto dominino numericamente le feature, garantendo stabilità durante il training. Un valore alto indica un nodo "parent" con molti figli (es. una categoria terapeutica ampia), mentre un valore nullo identifica una foglia (es. un farmaco specifico senza sottocategorie).

- **Grado uscente normalizzato** (1 dim):

$$\text{out\_deg\_norm}(v) = \log(1 + |\{u : (v, u) \in E\}|)$$

Numero di nodi parent a cui v punta, stabilizzato logaritmicamente. Nella gerarchia ATC, questo valore è tipicamente 0 (radice) o 1 (nodi con un singolo parent), ma la normalizzazione logaritmica garantisce robustezza in caso di strutture più complesse.

- **Profondità normalizzata** (1 dim):

$$\text{depth\_norm}(v) = \frac{\text{level}(v)}{5} \in [0, 1]$$

Livello del nodo scalato nell'intervallo $[0, 1]$. Questa feature ridondante rispetto al one-hot encoding fornisce al modello una rappresentazione continua della profondità, facilitando l'apprendimento di pattern graduali lungo la gerarchia.

Nota

Vettore finale:

$$\mathbf{x}_v = [\underbrace{\text{level\_onehot}(v)}_{5\text{ dim}}; \underbrace{\text{in\_deg\_norm}(v)}_{1\text{ dim}}; \underbrace{\text{out\_deg\_norm}(v)}_{1\text{ dim}}; \underbrace{\text{depth\_norm}(v)}_{1\text{ dim}}] \in \mathbb{R}^8$$

Questo vettore fornisce alla rete neurale informazioni sia categoriali (livello) sia strutturali (posizione topologica), permettendo di distinguere non solo *cosa* è un nodo (la sua classe gerarchica) ma anche *dove* si trova nel grafo (se è centrale, periferico o una foglia).

2.4 Etichette di training

Per l'addestramento supervisionato vengono estratte due tipologie di etichette per ogni nodo:

2.4.1 Etichette di categoria anatomica

La categoria anatomica corrisponde al primo carattere del codice ATC (A, B, C, ..., V). Questa classificazione identifica la macro-categoria anatomica del farmaco. Le categorie vengono mappate a indici interi sequenziali.

2.4.2 Etichette di livello gerarchico

Il livello gerarchico (1-5) viene convertito in indice 0-indexed (0-4) per compatibilità con PyTorch:

$$\text{level\_label}(v) = \text{level}(v) - 1 \in \{0, 1, 2, 3, 4\}$$

2.4.3 Cardinalità delle classi

Nel dataset utilizzato:

- **Categorie anatomiche**: 14 classi uniche
- **Livelli gerarchici**: 5 classi (livelli 1-5)

2.5 Architettura del modello: Graph Attention Network

2.5.1 Panoramica dell'architettura

Il modello `ATCGraphEncoder` implementa una Graph Neural Network basata su Graph Attention Networks (GAT). L'architettura è composta da:

1. **4 layer GAT convoluzionali**: elaborano le feature attraverso meccanismi di attention
2. **2 teste di classificazione**: predicono categoria e livello
3. **Skip connections**: preservano informazioni attraverso i layer
4. **Dropout**: regolarizzazione per prevenire overfitting

2.5.2 Graph Attention Convolution

Ogni layer GAT implementa un meccanismo di *attention* che permette ai nodi di decidere quanto "ascoltare" ciascuno dei loro vicini. L'idea chiave è semplice: non tutti i vicini sono ugualmente importanti per aggiornare la rappresentazione di un nodo.

Meccanismo di attention Immaginiamo il nodo `A10BA` (Biguanidi) che ha come vicini:

- Il parent `A10B` (Ipoglicemizzanti orali)
- I child `A10BA01`, `A10BA02` (Metformina), `A10BA03`, ...

Il meccanismo di attention calcola automaticamente quanto peso dare a ciascun vicino. Ad esempio, potrebbe decidere che:

- Il parent `A10B` è molto rilevante (alta attention) → peso 0.6
- Il child `A10BA02` è abbastanza rilevante → peso 0.3
- Altri child sono meno rilevanti → pesi bassi (0.05, 0.03, 0.02)

La rappresentazione aggiornata di `A10BA` sarà una **combinazione pesata** delle rappresentazioni dei suoi vicini, dove i pesi riflettono l'importanza relativa di ciascuno.

Tale esempio è da intendersi come puramente illustrativo: i pesi di attention non sono assegnati esplicitamente in base a ruoli semantici (parent o child), ma emergono dall'interazione tra feature dei nodi, struttura del grafo e funzione di loss durante l'addestramento.

Formalmente Per un nodo v con vicini $\mathcal{N}(v)$, il layer GAT:

1. **Calcola i coefficienti di attention** α_{vu} per ogni coppia (v, u):
   - Misura la "compatibilità" tra v e ciascun vicino u
   - Normalizza con softmax: $\sum_{u \in \mathcal{N}(v)} \alpha_{vu} = 1$
   - Risultato: quanto il nodo v "presta attenzione" al nodo u
2. **Aggrega le informazioni** pesando i vicini secondo i coefficienti:

$$\mathbf{h}'_v = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu} \mathbf{W} \mathbf{h}_u\right)$$

   dove $\mathbf{h}_u$ è la rappresentazione del vicino u, $\mathbf{W}$ trasforma le feature, e σ è una funzione di attivazione non lineare.

Multi-head attention Per aumentare l'espressività, il modello utilizza **multiple teste di attention indipendenti** ($K = 2$ nel nostro caso). Ogni testa può specializzarsi nel catturare diversi tipi di relazione:

- **Testa 1**: potrebbe imparare a dare più peso ai parent (relazioni "bottom-up")
- **Testa 2**: potrebbe imparare a dare più peso ai child (relazioni "top-down")

Le rappresentazioni prodotte dalle K teste vengono concatenate per formare l'output finale del layer:

$$\mathbf{h}'_v = [\mathbf{h}'_{v,1} \| \mathbf{h}'_{v,2} \| ... \| \mathbf{h}'_{v,K}]$$

L'utilizzo di più teste di attention consente al modello di apprendere simultaneamente diverse modalità di aggregazione delle informazioni locali. Sebbene non vi sia una corrispondenza esplicita tra una testa e uno specifico tipo di relazione (ad esempio parent o child), l'architettura permette, in linea di principio, di catturare pattern relazionali differenti all'interno del grafo.

2.5.3 Architettura dettagliata

Il modello è strutturato in quattro layer GAT convoluzionali sequenziali con le seguenti configurazioni:

Table 1: Architettura dei layer GAT

Layer	Input	Output/head	Heads	Output tot.	Attivazione
GAT 1	8	64	2	128	ELU + Dropout 20%
GAT 2	128	64	2	128	ELU + Dropout 20%
GAT 3	128	64	2	128	ELU + Dropout 20%
GAT 4	128	128	1	128	Nessuna

I primi tre layer utilizzano 2 teste di attention concatenate e attivazione ELU con dropout al 20% per regolarizzazione. Il quarto layer produce l'embedding finale di 128 dimensioni senza attivazione, che verrà utilizzato per le analisi successive e per le teste di classificazione.

2.5.4 Teste di classificazione

Dopo che i quattro layer GAT hanno prodotto gli embedding finali $\mathbf{h} \in \mathbb{R}^{128}$ per ciascun nodo, due layer lineari indipendenti utilizzano questi embedding per produrre predizioni multi-task.

Category head Predice la macro-categoria anatomica (prima lettera del codice ATC):

$$\text{logits}_{\text{cat}} = \mathbf{W}_{\text{cat}}\mathbf{h} + \mathbf{b}_{\text{cat}} \in \mathbb{R}^{14}$$

dove:

- $\mathbf{W}_{\text{cat}} \in \mathbb{R}^{14\times128}$: matrice di pesi che proietta l'embedding 128-dimensionale in 14 classi
- $\mathbf{b}_{\text{cat}} \in \mathbb{R}^{14}$: vettore di bias
- Output: 14 logit non normalizzati, uno per categoria (A, B, C, ..., V)

Durante il training, questi logit vengono passati a una funzione softmax per ottenere probabilità, poi confrontati con le etichette vere tramite Cross-Entropy loss.

Level head Predice il livello gerarchico del nodo (1–5):

$$\text{logits}_{\text{level}} = \mathbf{W}_{\text{level}}\mathbf{h} + \mathbf{b}_{\text{level}} \in \mathbb{R}^{5}$$

dove:

- $\mathbf{W}_{\text{level}} \in \mathbb{R}^{5\times128}$: matrice di pesi per proiezione in 5 classi
- $\mathbf{b}_{\text{level}} \in \mathbb{R}^{5}$: vettore di bias

- Output: 5 logit, uno per livello gerarchico

Analogamente, durante il training questi logit vengono normalizzati con softmax e ottimizzati tramite Cross-Entropy loss.

Architettura multi-task Le due teste sono completamente indipendenti ma condividono lo stesso encoder (i 4 layer GAT). Questo approccio multi-task permette al modello di:

- Apprendere rappresentazioni che codificano simultaneamente categoria e livello
- Sfruttare la correlazione tra i due task per migliorare la generalizzazione
- Produrre predizioni per entrambi gli obiettivi in un singolo forward pass

Durante l'inferenza, le predizioni finali si ottengono applicando argmax ai logit:

$$\text{categoria predetta} = \arg\max_i \text{logits}_{\text{cat}}[i], \qquad \text{livello predetto} = \arg\max_j \text{logits}_{\text{level}}[j]$$

2.5.5 Flusso di elaborazione

Il processo di forward propagation può essere rappresentato come:

$$\begin{aligned}
\mathbf{X} &\xrightarrow{\text{GAT}_1+\text{ELU}+\text{Dropout}} \mathbf{H}^{(1)} \in \mathbb{R}^{N\times 128} \\
\mathbf{H}^{(1)} &\xrightarrow{\text{GAT}_2+\text{ELU}+\text{Dropout}} \mathbf{H}^{(2)} \in \mathbb{R}^{N\times 128} \\
\mathbf{H}^{(2)} &\xrightarrow{\text{GAT}_3+\text{ELU}+\text{Dropout}} \mathbf{H}^{(3)} \in \mathbb{R}^{N\times 128} \\
\mathbf{H}^{(3)} &\xrightarrow{\text{GAT}_4} \mathbf{Z} \in \mathbb{R}^{N\times 128} \\
\mathbf{Z} &\xrightarrow{\text{Linear}} \begin{cases} \text{logits}_{\text{cat}} \in \mathbb{R}^{N\times 14} \\ \text{logits}_{\text{level}} \in \mathbb{R}^{N\times 5} \end{cases}
\end{aligned}$$

dove $\mathbf{Z}$ rappresenta la matrice degli embedding finali di tutti i nodi.

2.6 Funzioni di loss

L'addestramento del modello utilizza una loss multi-task che combina tre obiettivi complementari:

- **Loss di classificazione categoria** $\mathcal{L}_{\text{cat}}$: Cross-Entropy standard per la classificazione della macro-categoria anatomica (14 classi). Penalizza predizioni errate della prima lettera del codice ATC (A, B, C, ..., V).
- **Loss di classificazione livello** $\mathcal{L}_{\text{level}}$: Cross-Entropy per la predizione del livello gerarchico (5 classi). Penalizza predizioni errate della profondità del nodo nella gerarchia (livello 1–5).

- **Hierarchy pull loss** $\mathcal{L}_{\text{hier}}$: Componente innovativa che impone che gli embedding di nodi parent-child siano vicini nello spazio vettoriale. Per ogni arco $(c, p) \in E$ nel grafo (child $\rightarrow$ parent), la loss è definita come:

$$\mathcal{L}_{\text{hier}} = \frac{1}{|E|} \sum_{(c,p)\in E} \|\mathbf{z}_c - \mathbf{z}_p\|_2^2$$

  dove $\mathbf{z}_c$ e $\mathbf{z}_p$ sono gli embedding del nodo child e parent rispettivamente, e $\|\cdot\|_2^2$ è la distanza euclidea al quadrato. Questa loss ha l'effetto di "tirare" gli embedding dei nodi child verso i loro parent, creando cluster gerarchici nello spazio degli embedding.

Interpretazione geometrica Minimizzare $\mathcal{L}_{\text{hier}}$ significa che:

- `A10BA02` (Metformina) sarà vicina a `A10BA` (Biguanidi)
- `A10BA` sarà vicina a `A10B` (Ipoglicemizzanti orali)
- `A10B` sarà vicina a `A10` (Farmaci per diabete)
- `A10` sarà vicina a `A` (Apparato gastrointestinale)

Questo crea una struttura "nested" dove farmaci simili formano cluster compatti e la distanza riflette la distanza nella gerarchia ATC.

2.6.1 Loss totale

Le tre componenti vengono combinate con pesi differenziati:

$$\mathcal{L}_{\text{totale}} = \mathcal{L}_{\text{cat}} + w_{\text{level}} \cdot \mathcal{L}_{\text{level}} + w_{\text{hier}} \cdot \mathcal{L}_{\text{hier}}$$

dove:

- $w_{\text{level}} = 0.5$: peso per la loss del livello
- $w_{\text{hier}} = 0.1$: peso per la hierarchy pull loss

Bilanciamento dei pesi I pesi sono stati scelti per bilanciare i tre obiettivi:

- $\mathcal{L}_{\text{cat}}$ (peso 1.0): obiettivo principale, massima importanza alla separazione delle categorie
- $\mathcal{L}_{\text{level}}$ (peso 0.5): obiettivo secondario, importante ma meno critico
- $\mathcal{L}_{\text{hier}}$ (peso 0.1): vincolo geometrico, non deve dominare l'ottimizzazione ma deve guidare la struttura dello spazio

2.7 Configurazione del training

2.7.1 Parametri di ottimizzazione

Table 2: Configurazione del training

Parametro	Valore
Dataset	6440 codici ATC unici
Numero di epoche	300
Ottimizzatore	Adam
Learning rate	0.005
Weight decay	10^{-4}
Dropout	0.2 (20%)
Batch processing	Full-graph (intero grafo)

2.7.2 Strategia di ottimizzazione

A differenza del training standard su mini-batch, il modello viene addestrato sull'intero grafo contemporaneamente (full-batch training). Questo approccio presenta vantaggi:

- **Stabilità**: gradiente calcolato su tutti i nodi simultaneamente
- **Message passing globale**: ogni nodo può potenzialmente ricevere informazioni da tutto il grafo
- **Semplicità**: non richiede strategie di sampling dei vicini

2.7.3 Regolarizzazione

Due tecniche di regolarizzazione prevengono l'overfitting:

Dropout (20%) Durante il training, il 20% dei neuroni viene casualmente disattivato in ogni layer GAT. Questo:

- Previene co-adattazione dei neuroni
- Rende il modello più robusto
- Simula un ensemble di sottoreti

Ottimizzatore Adam con weight decay $\lambda = 10^{-4}$, che applica regolarizzazione L2 sui parametri del modello per prevenire overfitting e favorire soluzioni più semplici e generalizzabili.

3 Risultati del training

Durante le 300 epoche di training sono state monitorate le componenti della loss multi-task e le accuracy di classificazione, come riportato in Tabella 3.

Table 3: Evoluzione delle loss e accuracy durante il training (epoche selezionate)

Epoca	Loss tot.	L. cat	L. lvl	L. hier	Acc cat	Acc lvl
20	1.929	1.557	0.457	1.435	0.454	0.824
40	1.821	1.445	0.446	1.529	0.487	0.831
60	1.753	1.383	0.429	1.552	0.511	0.836
80	1.777	1.409	0.440	1.475	0.514	0.830
100	1.838	1.470	0.447	1.451	0.494	0.829
120	1.658	1.273	0.456	1.571	0.561	0.823
140	1.709	1.329	0.467	1.466	0.522	0.823
160	1.707	1.345	0.440	1.428	0.541	0.836
180	1.683	1.317	0.445	1.444	0.543	0.833
200	1.861	1.516	0.457	1.167	0.445	0.832
220	1.680	1.332	0.452	1.222	0.521	0.829
240	1.776	1.446	0.460	1.008	0.486	0.826
260	1.617	1.262	0.465	1.233	0.550	0.824
280	1.565	1.211	0.438	1.340	0.569	0.831
300	1.677	1.324	0.440	1.334	0.525	0.836

La loss totale presenta un andamento non monotono durante l'intero training, con una riduzione complessiva dal valore iniziale di 1.929 (epoca 20) al minimo di 1.565 (epoca 280), seguita da una stabilizzazione intorno a 1.6–1.7 nella fase finale. Tale comportamento è coerente con un regime di ottimizzazione multi-obiettivo.

Analizzando le singole componenti, la loss di categoria $\mathcal{L}_{\text{cat}}$ mostra la riduzione più marcata, pur con oscillazioni locali, indicando un progressivo miglioramento nella separazione delle macro-categorie anatomiche. La loss di livello $\mathcal{L}_{\text{level}}$ rimane invece molto stabile (range 0.429–0.467), suggerendo che la predizione del livello gerarchico costituisce un compito relativamente semplice per il modello.

La hierarchy pull loss $\mathcal{L}_{\text{hier}}$ presenta un andamento non monotono: un aumento iniziale nelle prime epoche è seguito da una diminuzione significativa fino all'epoca 240, indicando che il modello riesce progressivamente a bilanciare la separazione categoriale con il vincolo di coerenza gerarchica.

Per quanto riguarda le accuracy, la classificazione della categoria anatomica migliora con un valore finale del 52.5%. L'accuracy del livello gerarchico risulta invece elevata e molto stabile durante tutto il training (82.3%–83.6%).

3.0.1 Interpretazione complessiva

L'andamento oscillatorio della loss totale, associato al miglioramento dell'accuracy di categoria, suggerisce che il modello stia bilanciando obiettivi multipli potenzialmente in conflitto. La separazione delle categorie anatomiche rappresenta il compito più complesso e guida principalmente l'ottimizzazione, mentre la loss gerarchica introduce un vincolo geometrico che preserva la struttura del grafo.

Le oscillazioni osservate indicano l'esplorazione di diverse regioni dello spazio dei parametri e l'alternanza tra enfasi sulla separazione categoriale e rispetto della gerarchia. L'assenza di una convergenza monotona riflette quindi un compromesso tra i diversi obiettivi, piuttosto che un'instabilità del training.

3.1 Embedding finali

Al termine del training, il modello produce una matrice di embedding di dimensioni $(6440, 128)$, dove ogni riga rappresenta un codice ATC come vettore denso in $\mathbb{R}^{128}$. Questi embedding:

- Integrano informazioni da tutti i quattro layer GAT
- Riflettono la struttura del grafo attraverso il message passing
- Sono ottimizzati per separare le categorie e preservare la gerarchia
- Vengono utilizzati per tutte le analisi successive di qualità e struttura semantica

4 Metriche di valutazione

4.1 Validazione della preservazione gerarchica

4.1.1 Test distanze child-parent vs random-random

Questa metrica fondamentale verifica se la hierarchy pull loss ha effettivamente avvicinato gli embedding di nodi gerarchicamente correlati.

Metodologia Vengono confrontate due distribuzioni di distanze euclidee:

1. **Distanze child-parent**: distanze tra nodi connessi da archi nel grafo
2. **Distanze random-random**: distanze tra coppie casuali di nodi

Per efficienza computazionale, vengono campionati casualmente 5000 archi e 5000 coppie random.

Table 4: Confronto distanze child-parent vs random

Tipo di confronto	**Distanza media**
Child-Parent (gerarchico)	0.425
Random-Random (casuale)	2.082
Ratio	**0.204**

- La distanza child-parent è circa **5 volte inferiore** rispetto alla distanza random-random (ratio 0.204)
- Questo indica che la hierarchy pull loss ha effettivamente imposto una forte coerenza parent-child nello spazio embedding.
- Gli embedding rispettano chiaramente la struttura gerarchica: farmaci parent-child sono sistematicamente più vicini nello spazio vettoriale
- Il ratio basso **20%** indica una separazione tra relazioni gerarchiche e coppie casuali

4.2 K-NN Semantic Consistency

Questa metrica valuta la coerenza semantica locale dello spazio degli embedding, verificando se i k vicini più prossimi di ciascun nodo appartengono alla stessa macro-categoria anatomica.

Per ogni codice ATC:

1. Si identificano i $k = 10$ vicini più prossimi utilizzando la distanza euclidea
2. Si calcola la frazione di questi vicini che condividono la stessa macro-categoria
3. Si media questa frazione su tutti i codici del dataset

4.2.1 Risultato

Table 5: K-NN Semantic Consistency

Metrica	Valore
K-NN Consistency (k=10, macro ATC)	96.89%

Un valore del 96.89% indica che:

- In media, quasi 9.7 dei 10 vicini più prossimi di un codice appartengono alla sua stessa categoria anatomica
- Lo spazio degli embedding ha una struttura locale molto coerente
- La similarità vettoriale riflette accuratamente la similarità farmacologica

4.3 Metriche di clustering

Le metriche di clustering valutano quanto bene gli embedding si organizzano in cluster corrispondenti alle macro-categorie ATC. Le etichette ATC note fungono da ground truth per la valutazione.

4.3.1 Risultati

Table 6: Metriche di clustering basate sulle macro-categorie ATC

Metrica	Valore
Silhouette Score	0.240
Davies-Bouldin Index	2.188
Calinski-Harabasz Index	2889.4

4.3.2 Analisi dettagliata

Silhouette Score (0.240) Il Silhouette Score misura la coesione interna dei cluster vs. la separazione tra cluster diversi.

- **0.240**: Valore positivo ma moderato, indica cluster distinguibili ma con confini graduali
- I cluster non sono perfettamente separati nello spazio, ma sono chiaramente presenti

Davies-Bouldin Index (2.188) Questo indice misura il rapporto tra la dispersione interna dei cluster (quanto i punti sono sparsi al loro interno) e la separazione tra cluster diversi (quanto i centri dei cluster sono distanti).

- **2.188**: valore moderato, indica che i cluster sono distinguibili ma non perfettamente separati
- Valori intorno a 2 indicano una qualità di clustering discreta, sebbene non ottimale

Calinski-Harabasz Index (2889.4) Questo indice confronta la varianza between-cluster con la varianza within-cluster.

- **2889.4**: valore alto, indica una forte struttura di clustering
- Dimostra che le categorie anatomiche sono ben separate nello spazio degli embedding
- La varianza between-cluster è molto maggiore della varianza within-cluster, confermando cluster ben definiti

4.3.3 Confronto con l'approccio precedente

Table 7: Confronto metriche di clustering tra i due approcci

Metrica	Gerarchia	GNN	Variazione
Silhouette Score	0.050	0.240	+380%
Davies-Bouldin Index	3.739	2.188	-41%
Calinski-Harabasz Index	103.3	2889.4	+2697%

Il modello GNN mostra un miglioramento in tutte e tre le metriche di clustering, indicando che:

- Gli embedding GNN creano cluster più compatti e meglio separati
- La struttura a grafo e il message passing aiutano a organizzare meglio lo spazio vettoriale
- La separazione tra macro-categorie è significativamente più marcata rispetto all'approccio concatenato

4.3.4 Interpretazione generale

Le metriche di clustering confermano che il modello GNN ha appreso una rappresentazione che organizza i codici ATC in cluster corrispondenti alle macro-categorie anatomiche. Sebbene i cluster non siano perfettamente separati, la struttura è molto più definita rispetto all'approccio precedente.

La moderata separazione può essere attribuita a:

1. Sovrapposizioni semantiche reali tra alcune categorie farmacologiche
2. Il vincolo della hierarchy pull loss che preserva le relazioni parent-child
3. La complessità del task di classificazione (14 categorie) e l'accuracy moderata del modello (52.5%)

5 Visualizzazione dello spazio degli embedding

5.1 Proiezione t-SNE

Per ispezionare visivamente la struttura dello spazio degli embedding viene applicata la tecnica t-SNE (t-distributed Stochastic Neighbor Embedding) per ridurre le 128 dimensioni a 2 dimensioni visualizzabili. Ogni punto nel grafico rappresenta un codice ATC, colorato in base alla sua macro-categoria ATC1. L't-SNE è particolarmente adatto per:

- Preservare le relazioni di vicinanza locale
- Visualizzare cluster e strutture a livello di manifold
- Rivelare pattern che non sarebbero evidenti in dimensioni più alte

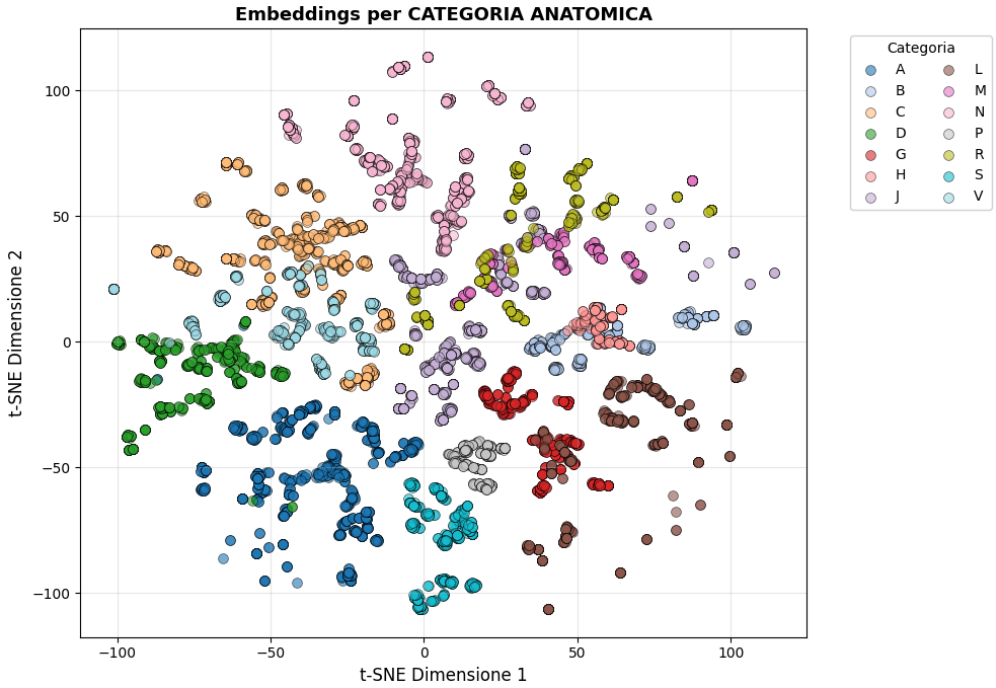

Figure 1: Proiezione t-SNE 2D degli embedding ATC.

Dal grafico t-SNE emergono diverse caratteristiche rilevanti riguardo alla struttura degli embed- ding appresi:

- **Formazione di cluster**: le macro-categorie anatomiche tendono a organizzarsi in gruppi visivamente distinguibili, con punti della stessa categoria concentrati in regioni coerenti dello spazio 2D.
- **Separazione variabile**: alcuni cluster risultano ben separati, mentre altri mostrano sovrapposizioni parziali, compatibili con similarità tra categorie e con i vincoli introdotti dall'addestramento multi-obiettivo
- **Compattezza intra-categoria**: la maggior parte delle categorie forma cluster relativamente compatti, suggerendo che farmaci della stessa macro-categoria hanno embedding simili
- **Struttura dello spazio**: la proiezione t-SNE evidenzia una struttura composta da regioni connesse e parzialmente separate; tali pattern vanno interpretati con cautela, poiché il metodo enfatizza principalmente le relazioni di vicinanza locale.

5.1.1 Analisi quantitativa delle distanze

Per supportare l'analisi visiva, vengono calcolate le distanze euclidee tra i centroidi delle categorie nello spazio t-SNE 2D. Tali distanze non hanno valore metrico assoluto, ma forniscono un'indicazione esplorativa della disposizione relativa delle categorie nella proiezione bidimensionale.

Table 8: Statistiche delle distanze tra centroidi delle categorie (spazio t-SNE)

Statistica	**Valore**
Distanza media	82.18
Distanza minima	7.84
Distanza massima	155.57
Deviazione standard	34.36
Coppie più vicine	B ↔ H (7.84)
Coppie più lontane	N ↔ S (155.57)

L'ampio range delle distanze e l'elevata deviazione standard indicano che le categorie occupano posizioni molto diverse nella proiezione t-SNE, con alcune coppie collocate in regioni prossime e altre fortemente separate. La vicinanza o lontananza tra specifiche coppie di categorie va interpretata con cautela, poiché la proiezione t-SNE enfatizza principalmente le relazioni locali e può distorcere le distanze globali.

5.2 Proiezione PCA

Per analizzare la struttura globale dello spazio degli embedding viene applicata l'Analisi delle Componenti Principali (PCA), riducendo le rappresentazioni da 128 dimensioni a due componenti principali. A differenza del t-SNE, la PCA preserva la varianza globale dei dati e consente un'interpretazione più diretta delle distanze nel piano proiettato.

In Figura 2 ogni punto rappresenta un codice ATC, colorato in base alla macro-categoria anatomica (ATC1).

Figure 2: Proiezione PCA 2D degli embedding ATC.

Dalla proiezione PCA emergono diverse caratteristiche rilevanti:

- **Separazione globale delle categorie**: molte macro-categorie occupano regioni distinte del piano PCA, indicando che una parte significativa della varianza degli embedding è correlata alla categoria anatomica.
- **Sovrapposizioni parziali**: alcune categorie presentano regioni di sovrapposizione, in particolare nelle aree centrali dello spazio, suggerendo la presenza di similarità strutturali tra specifiche macro-categorie.
- **Compattezza e orientamento dei cluster**: diverse categorie formano cluster relativamente compatti e allungati lungo direzioni specifiche, riflettendo una struttura anisotropa degli embedding nello spazio delle componenti principali.
- **Distribuzione continua**: a differenza della proiezione t-SNE, la PCA non produce isole nettamente separate, ma evidenzia una distribuzione più continua delle categorie, coerente con la natura lineare della proiezione.

Nel complesso, la visualizzazione PCA conferma che le informazioni di categoria sono in parte codificate negli embedding anche a livello globale, pur mantenendo sovrapposizioni che riflettono la complessità del dominio farmacologico.

5.2.1 Metodologia

Gli embedding 128-dimensionali vengono proiettati su 2 componenti principali che massimizzano la varianza spiegata:

$$\mathbf{Z}_{\text{PCA}} = \mathbf{Z} \cdot \mathbf{V}_{[1:2]}$$

dove $\mathbf{V}_{[1:2]}$ sono le prime due colonne della matrice degli autovettori della matrice di covarianza.

Varianza spiegata Le prime 2 componenti PCA spiegano una frazione della varianza totale, fornendo una rappresentazione bidimensionale utile per l'analisi esplorativa della struttura globale degli embedding.

5.2.2 Analisi quantitativa delle distanze

Table 9: Statistiche distanze tra centroidi delle categorie (spazio PCA)

Statistica	**Valore**
Distanza media	1.72
Distanza minima	0.04
Distanza massima	3.17
Deviazione standard	0.91
Coppie più vicine	B $\leftrightarrow$ H (0.04)
Coppie più lontane	A $\leftrightarrow$ G (3.17)

5.2.3 Confronto t-SNE vs PCA

Table 10: Confronto statistiche distanze: t-SNE vs PCA

Statistica	**t-SNE**	**PCA**
Distanza media	82.18	1.72
Distanza minima	7.84	0.04
Distanza massima	155.57	3.17
Dev. standard	34.36	0.91
Coppia più vicina	B-H	B-H
Coppia più lontana	N-S	A-G

Consistenza tra metodi

- La coppia più vicina (B-H) risulta identica in entrambi gli spazi, suggerendo una vicinanza persistente tra le due categorie anche a livello di struttura globale.
- Le coppie più lontane differiscono (N-S vs A-G), indicando che la struttura globale catturata dalla PCA non coincide completamente con la struttura locale enfatizzata dal t-SNE.

Scale diverse Le distanze PCA risultano molto più piccole di quelle t-SNE perché:

- il t-SNE amplifica le separazioni per enfatizzare le relazioni locali
- la PCA utilizza una proiezione lineare che preserva la varianza globale
- le distanze PCA sono più interpretabili in termini relativi, mentre quelle t-SNE hanno principalmente valore esplorativo.

Variabilità La deviazione standard più bassa osservata in PCA (0.91 vs 34.36) è coerente con una distribuzione delle distanze più uniforme nella proiezione globale, mentre il t-SNE, enfatizzando le relazioni locali, produce una maggiore eterogeneità nelle separazioni tra cluster.