



DIPARTIMENTO DI INGEGNERIA INFORMATICA, MODELLISTICA,  
ELETTRONICA E SISTEMISTICA

Corso di Laurea Magistrale in Ingegneria Informatica

MACHINE AND DEEP LEARNING

**Relazione Elaborato finale**

## **Improving Temporal Link Prediction via Temporal Walk Matrix Projection**

Professore:

Prof. **Fabrizio Angiulli**  
Ing. **Francesco De Luca**

Studente:

**Greco Matteo**  
matr. 252238

ANNO ACCADEMICO 2024/2025

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
1.1	Il problema della Link Prediction . . . . .	2
1.2	Un quadro unificato per la Temporal Link Prediction . . . . .	2
1.3	Il Modello TPNet . . . . .	3
1.3.1	Node Representation Maintaining (NRM) . . . . .	4
1.3.2	Link Likelihood Computing (LLC) . . . . .	5
1.3.3	Limiti del modello TPNet . . . . .	6
<b>2</b>	<b>Setting sperimentale, dataset e risultati</b>	<b>7</b>
2.1	Setting sperimentale . . . . .	7
2.2	Dataset . . . . .	8
2.3	Risultati modello <i>vanilla</i> . . . . .	9
<b>3</b>	<b>Modifiche Architetturali</b>	<b>11</b>
3.1	Modello multiscala con media semplice . . . . .	11
3.2	Modello multiscala con pesatura adattiva . . . . .	12
3.3	Prodotto con forma bilineare . . . . .	14
3.4	Time-Encoder MLP . . . . .	19
<b>4</b>	<b>Conclusioni</b>	<b>22</b>

# 1 Introduzione

## 1.1 Il problema della Link Prediction

La *link prediction* è il problema di prevedere l'esistenza di un collegamento tra due entità all'interno di una rete. L'obiettivo è determinare la probabilità che due nodi formino un legame, a partire dalle informazioni già presenti nella struttura del grafo. Si tratta di un problema centrale nello studio dei grafi, poiché consente di anticipare e comprendere i meccanismi che guidano l'evoluzione delle reti.

Questo problema ha un ruolo chiave in numerosi ambiti applicativi. Nei *social network*, la link prediction è impiegata per suggerire nuove amicizie o connessioni professionali; nei sistemi di raccomandazione, per prevedere interessi o preferenze di utenti; nelle reti biologiche, per individuare interazioni potenziali tra proteine o geni; nelle reti di comunicazione, per anticipare la formazione o l'interruzione di collegamenti. In tutti questi casi, la capacità di prevedere relazioni future permette di comprendere meglio la struttura latente e la dinamica evolutiva della rete.

Tradizionalmente, la *link prediction* è stata affrontata su grafi statici, in cui le connessioni tra nodi sono considerate fisse nel tempo. Tuttavia, nella realtà, molte reti sono dinamiche: i nodi e gli archi cambiano nel tempo, e le interazioni avvengono in momenti specifici. Da qui nasce la *temporal link prediction*, un'estensione del problema classico che integra la dimensione temporale. In questo scenario, l'obiettivo della *temporal link prediction* è prevedere le interazioni future tra le entità a partire dalle interazioni storiche osservate.

L'introduzione del tempo rende il problema più realistico, ma anche più complesso: occorre modellare l'evoluzione delle rappresentazioni dei nodi, tenere conto del decadimento temporale delle interazioni e catturare pattern di lungo periodo. Un elemento indispensabile per una *temporal link prediction* efficace è costituito dai *relative encodings*, cioè rappresentazioni che catturano la relazione tra coppie di nodi all'interno del grafo.

La sfida principale rimane quella di integrare in modo efficiente le informazioni strutturali e temporali, preservando la capacità del modello di rappresentare accuratamente le dipendenze tra le entità nel tempo.

## 1.2 Un quadro unificato per la Temporal Link Prediction

Dato un link futuro da predire  $(u, v, t)$ , negli approcci tradizionali di *temporal link prediction*, le rappresentazioni dei nodi coinvolti in una possibile futura interazione, ad esempio  $h_u(t)$  e  $h_v(t)$ , vengono apprese in modo indipendente tramite l'aggregazione dei rispettivi sottografi locali, sfruttando anche le *feature* di nodo e di archi presenti in essi. Tuttavia, come riportato dagli autori di

*TPNet*, questa strategia tende a non cogliere in modo completo le informazioni di coppia (*pairwise*) del link. Metodi più recenti inseriscono informazioni di coppia costruendo *relative encoding*  $r^{w|(u,v)}$  per ogni nodo  $w$  appartenente ad entrambi i sottografi locali dei nodi in analisi. Intuitivamente, il *relative encoding* riflette l'importanza per ogni nodo per la predizione del link  $(u, v, t)$ . In particolare, è la concatenazione di due *similarity feature*  $r^{w|u}$  e  $r^{w|v}$  il cui calcolo dipende dai diversi metodi. Gli autori mostrano che le diverse formulazioni proposte in letteratura possono essere ricondotte a una struttura comune:

$$r^{w|u} = g([A_{u,w}^{(0)}(t), A_{u,w}^{(1)}(t), \dots, A_{u,w}^{(k)}(t)]), \quad A_{u,w}^{(i)}(t) = \sum_{W \in \mathcal{M}_{u,w}^i} s(W)$$

dove:

- $A_{u,w}^{(i)}(t)$  rappresenta la somma dei punteggi associati ai cammini temporali di lunghezza  $i$  tra i nodi  $u$  e  $w$
- $s(W)$  è una funzione di scoring che assegna un peso a ciascun cammino
- $g(\cdot)$  è una funzione di aggregazione che combina tali informazioni in un vettore di rappresentazione

L'equazione riportata mostra che ogni *relative encoding* implica una costruzione di una matrice di cammini temporali (*temporal walk matrix*) basata sui pesi associati ad ogni cammino temporale (*temporal walk*) (usando  $s(\cdot)$ ). Infatti,  $\mathbf{A}^{(i)}(t)$  indica una *temporal walk matrix* in cui ogni entry  $A_{u,v}^{(i)}$  è la somma degli score di tutti i cammini temporali di lunghezza  $i$  da  $u$  a  $w$ . A partire da questa prospettiva unificata, *TPNet* propone la propria definizione della funzione di pesatura  $s(\cdot)$ , introducendo un modo alternativo di calcolare e aggiornare la *temporal walk matrix* per rappresentare in modo più accurato l'influenza temporale tra i nodi.

### 1.3 Il Modello TPNet

Il modello *TPNet* consiste in due moduli:

- *Node Representation Maintaining (NRM)*: responsabile per l'*encoding* delle informazioni di coppia.
- *Link Likelihood Computing (LLC)*: modulo di predizione.

Prima di approfondire il modello, è utile riportare dei concetti di base:

- **Temporal Graph:** un grafo temporale può essere considerato come una sequenza cronologica non decrescente di interazioni

$\mathcal{G} = [(\{u_1, v_1\}, t_1), (\{u_2, v_2\}, t_2)]$ , dove  $(\{u_i, v_i\}, t_i)$  può essere considerato come un collegamento diretto tra  $u_i$  e  $v_i$  con *timestamp*  $t_i$ . Ogni nodo  $u$  può avere associata una *feature di nodo*  $x_u \in \mathbb{R}^{d_N}$  e ogni interazione  $(\{u, v\}, t)$  può avere associata una *link feature*  $e_{u,v}^t \in \mathbb{R}^{d_E}$ , dove  $d_N$  e  $d_E$  denotano le dimensioni delle feature di nodo e delle feature di arco.

- **K-hop Subgraph:** si indica con  $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}(t))$  il grafo al tempo  $t$ , dove  $\mathcal{E}(t)$  contiene tutte le interazioni accadute prima di  $t$  e  $\mathcal{V}(t)$  include tutti i nodi che appaiono in  $\mathcal{E}(t)$ . Quindi, si definisce *sottografo a k-hop* (*k-hop subgraph*) di un nodo  $u$   $\mathcal{G}_u^k(t) = (\mathcal{V}_u^k(t), \mathcal{E}_u^k(t))$ , dove  $\mathcal{V}_u^k(t) \subset \mathcal{N}(t)$  indica l'insieme dei nodi la cui distanza minima verso  $u$  è inferiore a  $k$  su  $\mathcal{G}(t)$ , e  $\mathcal{E}_u^k(t) \subset \mathcal{E}(t)$  è l'insieme delle interazioni in  $\mathcal{V}_u^k(t)$ .
- **Temporal Walk:** un cammino temporale a  $k$  passi  $W$  (*k-step temporal walk W*) su  $\mathcal{G}(t)$  è una sequenza di nodo-istante temporale (in ordine temporale decrescente) indicato come  $W = [(w_0, t_0), \dots, (w_k, t_k)]$ , con  $(\{w_i, w_{i+1}\}, t_{i+1})$  è un arco presente in  $\mathcal{G}(t)$ . Si indica con  $\mathcal{M}_{u,v}^k(t)$  l'insieme di tutti i cammini temporali di lunghezza  $k$  dal nodo  $u$  al nodo  $v$  su  $\mathcal{G}(t)$ .

### 1.3.1 Node Representation Maintaining (NRM)

Il **Node Representation Maintaining (NRM)** costituisce uno dei due modelli del modello TPNet e ha il compito di mantenere aggiornate nel tempo le rappresentazioni dei nodi.

Il modello TPNet utilizza una funzione di scoring con l'obiettivo di considerare sia informazioni temporali che strutturali. Formalmente, considerando  $t$  l'istante corrente, e un cammino temporale  $W = [(w_0, t_0), (w_1, t_1), \dots, (w_k, t_k)]$ , il valore della funzione di score è il seguente:

$$s(W) = \prod_{i=1}^k e^{-\lambda(t-t_i)}$$

dove  $\lambda > 0$  è un iperparametro che controlla il peso del decadimento temporale. Con il passare del tempo  $t$ , per ogni interazione  $(\{w_i, w_{i+1}\}, t_{i+1})$  nel cammino temporale  $W$ , il suo peso  $e^{-\lambda(t-t_i)}$  diminuirà esponenzialmente. Quindi, più un'interazione è lontana rispetto all'istante corrente, minore è il suo peso. Questo modella un comportamento realistico, cioè l'informazione associata a interazioni cronologicamente lontane risulta meno rilevante ai fini della predizione.

In TPNet, però, non vengono calcolate direttamente le matrici di cammini temporali in quanto ciò porterebbe ad una complessità spaziale dell'ordine di  $O(n \times n)$  per ogni matrice (bisogna enumarare i cammini temporali). Implicitamente vengono mantenute le matrici di cammini temporali mantenendo una serie di rappresentazioni dei nodi  $\mathbf{H}^{(0)}(t), \mathbf{H}^{(1)}(t), \dots, \mathbf{H}^{(k)}(t) \in \mathbb{R}^{n \times d_R}$ , dove

$d_R \ll n$  è la dimensione delle rappresentazioni dei nodi e  $\mathbf{H}_u^{(l)}(t) \in \mathbb{R}^{d_R}$  codifica le informazioni relative ai cammini temporali di lunghezza  $l$  che hanno origine in  $u$ . Le rappresentazioni dei nodi vengono aggiornate quando occorre una nuova interazione:

$$\mathbf{H}_u^{(l)}(t^+) = \mathbf{H}_u^{(l)}(t) + e^{\lambda t} * H_v^{(l-1)}(t), \quad \mathbf{H}_v^{(l)}(t^+) = \mathbf{H}_v^{(l)}(t) + e^{\lambda t} * H_u^{(l-1)}(t)$$

L'inizializzazione consiste nel porre  $\mathbf{H}^{(0)}(t)$  pari ad una *random feature matrix*  $\mathbf{P} \in \mathbb{R}^{n \times d_R}$  dove ciascun elemento è campionato indipendentemente da una distribuzione Gaussiana con media 0 e varianza  $\frac{1}{d_R}$ , e le restanti rappresentazioni dei nodi pari a matrici di zero.

Questa formulazione consente di rappresentare in modo compatto e scalabile le informazioni che, nei metodi tradizionali, sarebbero esplicite attraverso le *temporal walk matrix*, riducendo la complessità da  $O(n \times n)$  a  $O(n \times d_R)$ .

### 1.3.2 Link Likelihood Computing (LLC)

Il modulo **Link Likelihood Computing (LCC)** ha il compito di stimare la probabilità che due nodi formino un collegamento in un determinato istante futuro. Dato un'interazione  $(u, v, t)$  da predire, si calcola la probabilità che si verifichi sulla base delle rappresentazioni dei nodi ottenuti e delle *feature* ausiliarie. Nello specifico, si decodificano prima le feature di coppia (*pairwise feature*)  $f_{u,v}(t)$  dalle rappresentazioni dei nodi. Dopo si calcolano gli embedding dei nodi (*node embedding*)  $h_u(t)$  e  $h_v(t)$  per il nodo  $u$  e per il nodo  $v$  rispettivamente basato sulle interazioni passate. Infine, si calcola la probabilità basato su  $h_u(t), h_v(t), f_{u,v}(t)$ .

In particolare, per ciascun nodo vengono considerate le rappresentazioni ottenute ai diversi livelli, ottenendo  $\mathbf{F}_* = [e^{-\lambda_0 t} \mathbf{H}_*^{(0)}, \dots, e^{-\lambda_k t} \mathbf{H}_*^{(k)}]$ . I vettori ottenuti per i due nodi,  $F_u$  e  $F_v$ , vengono concatenati per formare la rappresentazione congiunta  $F_{u,v}$ , da cui si ottiene una *raw pairwise feature* tramite un prodotto interno tra le loro componenti. Questa rappresentazione viene poi elaborata da una rete *MLP* che produce la *pairwise feature* finale  $f_{u,v}$ , in particolare  $\mathbf{f}_{u,v} = \text{MLP}(\log(\text{ReLU}(\tilde{\mathbf{f}}_{u,v}) + 1))$ .

Oltre alle *pairwise features*, *TPNet* impiega un meccanismo di *Auxiliary Feature Learning* per modellare in modo più accurato l'evoluzione temporale delle interazioni. Per ciascun nodo, viene considerata la sequenza delle ultime  $m$  interazioni e vengono estratte le corrispondenti *node features*  $\mathbf{X}_{u,N}$  e *edge features*  $\mathbf{X}_{u,E}$ , insieme a una codifica temporale periodica basata su funzioni coseno  $\mathbf{X}_{u,T}$ . Oltre a queste *feature* si costruisce una concatenazione di *pairwise feature*  $\mathbf{X}_{u,F}$  (dove si concatenano le diverse *pairwise feature*). Tutte queste informazioni vengono concatenate e passate attraverso un *MLP* in modo da ottenere la sequenza di *feature* finale  $\mathbf{Z}_u^{(0)} = \text{MLP}([\mathbf{X}_{u,N}, \mathbf{X}_{u,E}, \mathbf{X}_{u,T}, \mathbf{X}_{u,F}])$ . Successivamente si impilano  $l$  *layer* di *MLP-Mixer* per catturare dipendenze temporali e

strutturali all'interno della sequenza di *feature*. L'embedding del nodo, si ottiene effettuando la media  $\mathbf{h}_u = \text{MEAN}(\mathbf{Z}_u^{(l)})$

La probabilità che il link  $(u, v)$  si formi all'istante  $t$  viene calcolata come:

$$p_{u,v}^t = \text{MLP}([\mathbf{h}_u, \mathbf{h}_v, \mathbf{f}_{u,v}])$$

Combinando *pairwise decoding* e *auxiliary feature learning*, *TPNet* riesce a modellare simultaneamente le relazioni strutturali e temporali, ottenendo predizioni più stabili e accurate sulla formazione dei collegamenti futuri.

### 1.3.3 Limiti del modello TPNet

Una limitazione del modello *TPNet* riguarda la dipendenza da impostazioni manuali nella fase di costruzione implicita delle matrici. Reti con caratteristiche differenti possono richiedere strategie di costruzione diverse, comportando un ulteriore sforzo sperimentale. In particolare, il fattore di decadimento temporale controllato da  $\lambda$  incide direttamente sulla funzione di pesatura e potrebbe non essere ottimale in scenari caratterizzati da dipendenze di lungo periodo. La definizione automatica o adattiva di tale parametro e, più in generale, lo sviluppo di tecniche di costruzione dinamiche rappresentano una direzione promettente per futuri miglioramenti del modello.

## 2 Setting sperimentale, dataset e risultati

### 2.1 Setting sperimentale

Negli esperimenti condotti dagli autori di *TPNet*, sono stati considerati due differenti scenari di valutazione che verranno seguiti anche dai successivi test:

- **Transductive setting:** il modello predice collegamenti tra nodi già osservati durante la fase di addestramento.
- **Inductive setting:** il modello deve invece prevedere collegamenti che coinvolgono nodi non presenti nei dati di training.

Per il campionamento dei collegamenti negativi vengono utilizzate tre differenti strategie:

- **Random Negative Sampling:** vengono scelti casualmente due nodi non collegati al tempo  $t$ .
- **Historical Negative Sampling:** vengono scelti due nodi che in passato erano collegati ma al tempo  $t$  non lo sono più. Riflette uno scenario più realistico ma più difficile.
- **Inductive Negative Sampling:** vengono campionate coppie di nodi che non appaiono mai nei collegamenti presenti nella fase di addestramento.

I collegamenti positivi (cioè quelli realmente esistenti o osservati) sono pochi rispetto a tutte le possibili coppie di nodi. Se si usassero solo esempi positivi, il modello impararebbe molto poco in quanto non avrebbe esempi negativi. Di conseguenza, bisogna generare anche esempi negativi, cioè coppie di nodi che non sono collegati.

Le metriche adottate per valutare il modello sono l'*Average Precision (AP)* e l'*Area Under the ROC Curve (AUC-ROC)*.

L'*Average Precision* rappresenta l'area sotto la Curva *Precision-Recall* (PR Curve), indicando la capacità del modello di mantenere un'alta precisione anche quando il richiamo (la copertura dei positivi) aumenta.” Le quantità fondamentali sono definite come:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN},$$

dove  $TP$ ,  $FP$  e  $FN$  rappresentano rispettivamente veri positivi, falsi positivi e falsi negativi. Maggiore è il valore di AP più efficace è il modello.

L'*AUC-ROC* è pari all'area sotto la curva ROC e descrive il comportamento del modello confrontando il tasso di veri positivi e il tasso di falsi positivi, definiti come:

$$TPR = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN},$$

dove  $TN$  indica i veri negativi. Anche in questo caso, valori più elevati di AUC indicano una migliore capacità di predizione.

## 2.2 Dataset

Il modello *TPNet* è stato originariamente valutato su tredici dataset di natura eterogenea. Nel presente elaborato è stato selezionato un sottoinsieme di sei dataset rappresentativi, che coprono tutti i principali domini presenti nel set originale di tredici dataset (sociale, di interazione, politico, di prossimità e di trasporto), garantendo così coerenza con il lavoro originale e al tempo stesso una riduzione della complessità sperimentale.

I dataset scelti sono i seguenti:

- **UCI** — È una rete di comunicazione online in cui i nodi rappresentano studenti universitari e i collegamenti rappresentano i messaggi pubblicati.
- **LastFM** — È una rete bipartita che descrive i comportamenti di ascolto musicale degli utenti nell'arco di un mese. I nodi rappresentano gli utenti e i brani musicali, mentre i collegamenti indicano le attività di ascolto.
- **Flights** — È una rete dinamica di voli che mostra l'evoluzione del traffico aereo durante la pandemia di COVID-19. I nodi rappresentano gli aeroporti, mentre i collegamenti indicano i voli tracciati. Ogni collegamento è associato a un peso che rappresenta il numero di voli giornalieri tra due aeroporti.
- **Can. Parl.** — È una rete politica dinamica che registra le interazioni tra i membri del Parlamento canadese (MP) dal 2006 al 2019. I nodi rappresentano i parlamentari dei diversi distretti elettorali, mentre un collegamento viene creato quando due deputati votano “sì” su una proposta di legge. Il peso di ciascun collegamento rappresenta il numero di volte in cui un parlamentare ha votato “sì” a sostegno di un altro parlamentare nell'arco di un anno.
- **Contacts** — Traccia l'evoluzione della prossimità fisica tra circa 700 studenti universitari nell'arco di un mese. Ogni studente è identificato da un identificatore univoco, e i collegamenti indicano situazioni di prossimità fisica, con pesi che rappresentano il grado di vicinanza tra gli studenti.

Tabella 1: Statistiche dei dataset utilizzati negli esperimenti.

Dataset	Dominio	#Nodi	#Link	#N&L Feat	Bipartito	Durata	Step unici	Granularità
UCI	Sociale	1,899	59,835	- & -	False	196 giorni	58,911	Unix timestamps
LastFM	Interazione	1,980	1,293,103	- & 1	True	1 mese	1,283,614	Unix timestamps
Flights	Trasporti	13,169	1,927,145	- & 1	False	4 mesi	122	Giorni
Can. Parl.	Politica	734	74,478	- & 1	False	14 anni	14	Anni
Contacts	Prossimità	692	2,426,279	- & 1	False	1 mese	8,064	5 minuti

## 2.3 Risultati modello *vanilla*

In questa sezione vengono riportati i risultati ottenuti dalla riproduzione degli esperimenti originali del modello *TPNet*. Gli esperimenti sono stati condotti utilizzando, per ciascun dataset, il valore ottimale del parametro di decadimento temporale  $\lambda$  individuato dagli autori del paper tramite *grid search*.

Le differenze rispetto alla configurazione originale riguardano esclusivamente il numero di epoche di addestramento e di esecuzioni (*run*), ridotti in funzione delle risorse computazionali disponibili. In generale il numero di *run* è stato ridotto da 5 a 3. Per quanto riguarda il numero di epoche, invece:

- **UCI** e **Can. Parl.** — numero di epoche invariato;
- **LastFM**, **Flights** e **Contacts** — numero di epoche ridotto da 100 a 10.

Tutti gli altri parametri di configurazione sono rimasti coerenti con le impostazioni riportate dagli autori nelle *best configurations* originali. In particolare, i valori del parametro  $\lambda$  che controlla il decadimento temporale sono riportati di seguito.

Tabella 2: Valori ottimali del parametro di decadimento temporale  $\lambda$  per ciascun dataset, come riportato nel paper originale.

Dataset	$\lambda$
UCI	$10^{-7}$
LastFM	$10^{-7}$
Flights	$10^{-6}$
Can. Parl.	$10^{-5}$
Contacts	$10^{-4}$

Di seguito i risultati ottenuti:

Tabella 3: Confronto tra modello originale (*Paper*) e replica (*Replica*) nei due setting di *link prediction* e nei tre tipi di *negative sampling*.

Setting	Misura	Dataset	Random		Historical		Inductive	
			Paper	Replica	Paper	Replica	Paper	Replica
<b>AP</b>		UCI	0.9735	0.9732	0.8634	0.8681	0.7726	0.7787
		LastFM	0.9450	0.9404	0.8774	0.8712	0.7799	0.7704
		Flights	0.9893	0.9888	0.6910	N/A	0.6478	N/A
		Can. Parl.	0.9028	0.8360	0.8661	0.8275	0.8559	0.8099
		Contacts	0.9866	0.9860	0.9802	0.9802	0.9584	0.9583
<b>Transductive</b>		UCI	0.9679	0.9674	0.8042	0.8073	0.7085	0.7116
		LastFM	0.9439	0.9384	0.8464	0.8415	0.7248	0.7165
		Flights	0.99	0.9895	0.7182	N/A	0.6421	N/A
		Can. Parl.	0.9205	0.8498	0.8639	0.8315	0.8505	0.8169
		Contacts	0.9891	0.9887	0.9773	0.9768	0.9593	0.9579
<b>AP</b>		UCI	0.9574	0.9566	0.7848	0.7899	0.7850	0.7901
		LastFM	0.9536	0.9516	0.8227	0.8200	0.8227	0.8201
		Flights	0.9797	0.9791	0.5467	N/A	0.5463	N/A
		Can. Parl.	0.6809	0.6519	0.6897	0.6687	0.6887	0.6662
		Contacts	0.9839	0.9834	0.9356	0.9331	0.9357	0.9331
<b>Inductive</b>		UCI	0.9440	0.9426	0.7135	0.7164	0.7137	0.7165
		LastFM	0.9536	0.9501	0.7710	0.7679	0.7710	0.7679
		Flights	0.9805	0.9799	0.5308	N/A	0.5305	N/A
		Can. Parl.	0.6921	0.6484	0.6911	0.6662	0.6898	0.6636
		Contacts	0.9851	0.9847	0.9347	0.9331	0.9347	0.9331

Per il dataset **Flights** non sono riportati i risultati per le strategie di negative sampling **historical** e **inductive**, poiché l'esecuzione su Kaggle eccedeva la memoria disponibile, data la dimensione del dataset. È stato comunque mantenuto in quanto è l'unico *dataset* che copre il dominio dei trasporti.

### 3 Modifiche Architetturali

L'obiettivo di questo capitolo è presentare estensioni architetturali del modello *TPNet*.

#### 3.1 Modello multiscala con media semplice

Tra i problemi principali individuati dagli autori di *TPNet* vi è l'utilizzo di un unico parametro di decadimento  $\lambda$  che non consente di catturare sia dipendenze di breve che di lungo periodo.

Una soluzione naïve a ciò è utilizzare  $M$  valori di *lambda*. In particolare, vengono utilizzati 6 valori di  $\lambda$  (a partire da un ordine di grandezza più piccolo a finire ad un ordine di grandezza più grande dei valori utilizzati per i dataset analizzati nel paper originale):

$$\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$$

Per ciascun valore  $\lambda_m$  con  $m \in \{1, \dots, M\}$  si ottiene:

$$F_{(m)} = [F_m^{(0)}, F_m^{(1)}, \dots, F_m^{(k)}]$$

In questo caso, l'apice indica l'*hop* (lunghezza del cammino temporale) considerato per la costruzione della rappresentazione. Il pedice indica il corrispondente valore di  $\lambda$  considerato. Per pulizia di notazione l'indice del nodo è omesso; tuttavia il calcolo viene effettuato per i due nodi in analisi.

Ciascuna rappresentazione ottenuta viene combinata attraverso una **media semplice**. Quindi per ciascun *hop* si ottiene:

$$\bar{F}^{(i)} = \frac{1}{M} \sum_{m=1}^M F_m^{(i)}$$

Infine si ottiene:

$$\bar{F} = [\bar{F}^{(0)}, \bar{F}^{(1)}, \dots, \bar{F}^{(k)}]$$

dove ogni componente è la media dei corrispondenti  $F_m^{(i)}$  sui  $M$  valori di  $\lambda$  considerati. Questo è equivalente alla seguente definizione:

$$\bar{F} = \frac{1}{M} \sum_{m=1}^M F_m$$

Questa aggregazione mantiene inalterata la dimensione delle feature e preserva la compatibilità con la pipeline originale di *TPNet*. Il vantaggio di tale estensione consiste nella possibilità di modellare simultaneamente dipendenze a breve

e lungo termine, migliorando la sensibilità temporale del modello pur senza introdurre parametri addizionali.

Sono stati replicati gli esperimenti utilizzando il modello con codifica multiscala. I risultati ottenuti sono riportati nella seguente tabella:

Tabella 4: Confronto tra modello originale (*Replica*) e con modifica (*Modifica*) nei due setting di *link prediction* e nei tre tipi di *negative sampling*.

Setting	Misura	Dataset	Random		Historical		Inductive	
			Replica	Modifica	Replica	Modifica	Replica	Modifica
<b>AP</b>	<b>AUC</b>	UCI	0.9732	0.9720	0.8681	0.8682	0.7787	0.7824
		LastFM	0.9404	0.9391	0.8712	0.8694	0.7704	0.7681
		Flights	0.9888	0.9880	N/A	N/A	N/A	N/A
		Can. Parl.	0.8360	0.8006	0.8275	0.8240	0.8099	0.7912
		Contacts	0.9860	0.9944	0.9802	0.9748	0.9583	0.9598
<b>Transductive</b>	<b>AUC</b>	UCI	0.9674	0.9657	0.8073	0.8075	0.7116	0.7166
		LastFM	0.9384	0.9373	0.8415	0.8335	0.7165	0.7116
		Flights	0.9895	0.9881	N/A	N/A	N/A	N/A
		Can. Parl.	0.8498	0.8690	0.8315	0.8658	0.8169	0.8336
		Contacts	0.9887	0.9869	0.9768	0.9710	0.9579	0.9581
<b>AP</b>	<b>AUC</b>	UCI	0.9566	0.9568	0.7899	0.7887	0.7901	0.7889
		LastFM	0.9516	0.9469	0.8200	0.8143	0.8201	0.8144
		Flights	0.9791	0.9745	N/A	N/A	N/A	N/A
		Can. Parl.	0.6519	0.5871	0.6687	0.5887	0.6662	0.5893
		Contacts	0.9834	0.9802	0.9331	0.9447	0.9331	0.9447
<b>Inductive</b>	<b>AUC</b>	UCI	0.9426	0.9431	0.7164	0.7175	0.7165	0.7177
		LastFM	0.9501	0.9450	0.7679	0.7581	0.7679	0.7581
		Flights	0.9799	0.9763	N/A	N/A	N/A	N/A
		Can. Parl.	0.6484	0.6077	0.6662	0.5967	0.6636	0.5967
		Contacts	0.9847	0.9833	0.9331	0.9430	0.9331	0.9430

L'introduzione della media semplice tra proiezioni multiscala non produce variazioni significative rispetto al modello originale. Il comportamento resta pressoché invariato su tutti i dataset e schemi di negative sampling, portando alla luce che una combinazione non pesata delle scale temporali non apporta vantaggi reali.

Da notare, però, che la media semplice assegna lo stesso peso a ciascuna scala temporale, ignorando la diversa utilità predittiva delle varie componenti. Da questa considerazione nasce la seconda modifica, basata su una **pesatura adattiva**, presentata nella sezione successiva.

### 3.2 Modello multiscala con pesatura adattiva

La seconda estensione del modello *TPNet* sostituisce la media semplice tra le scale temporali con una strategia di **fusione pesata per-hop**. Idealmente si vuole consentire al modello di apprendere, quale scala temporale risulti più rilevante nel catturare le dinamiche evolutive del grafo.

Come nel modello multiscala a media semplice, si considerano diversi valori del parametro di decadimento temporale

$$\lambda_m \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}, \quad m = 1, \dots, M,$$

ognuno dei quali genera una sequenza di rappresentazioni dei nodi

$$F_m^{(i)}, \quad i = 0, \dots, k,$$

dove  $i$  rappresenta la profondità dell'hop nella propagazione.

La pesatura per-hop introduce una matrice di pesi appresi

$$W \in \mathbb{R}^{(k+1) \times M},$$

dove ogni riga  $W_i$  rappresenta gli *hop* e ogni colonna le diverse scale temporali. I pesi vengono normalizzati tramite una funzione softmax:

$$\tilde{W}_{i,m} = \frac{e^{W_{i,m}}}{\sum_{m'=1}^M e^{W_{i,m'}}},$$

in modo che, per ciascun hop  $i$ , i pesi siano positivi e la loro somma pari a uno. La fusione finale delle rappresentazioni multi-scala diventa così:

$$F^{(i)} = \sum_{m=1}^M \tilde{W}_{i,m} F_m^{(i)}.$$

Questo meccanismo consente al modello di adattare la combinazione delle scale in funzione della profondità: gli hop più bassi tendono a privilegiare scale a decadimento rapido (grandi  $\lambda$ ), più sensibili a eventi recenti, mentre gli hop più alti enfatizzano scale a decadimento lento (piccoli  $\lambda$ ), capaci di modellare relazioni di lungo periodo.

In sintesi, la pesatura per-hop introduce un numero aggiuntivo di parametri pari a  $(k+1) \times M$ , che vengono ottimizzati durante l'addestramento. Essa conferisce al modello la capacità di **adattare dinamicamente la fusione tra le scale temporali** in base alla profondità e al contenuto informativo, migliorando la rappresentazione delle dipendenze temporali complesse rispetto alla media uniforme.

Sono stati replicati gli esperimenti utilizzando il modello con la media pesata. I risultati ottenuti sono riportati nella seguente tabella:

Tabella 5: Confronto tra modello originale (*Replica*) e con modifica (*Modifica*) nei due setting di *link prediction* e nei tre tipi di *negative sampling*.

Setting	Misura	Dataset	Random		Historical		Inductive	
			Replica	Modifica	Replica	Modifica	Replica	Modifica
<b>AP</b>		UCI	0.9732	0.9717	0.8681	0.8684	0.7787	0.7829
		LastFM	0.9404	0.9372	0.8712	0.8639	0.7704	0.7580
		Flights	0.9888	0.9981	N/A	N/A	N/A	N/A
		Can. Parl.	0.8360	0.8322	0.8275	0.8472	0.8099	0.8178
		Contacts	0.9860	0.9831	0.9802	0.9756	0.9583	0.9603
<b>Transductive</b>		UCI	0.9674	0.9653	0.8073	0.8074	0.7116	0.7171
		LastFM	0.9384	0.9347	0.8415	0.8296	0.7165	0.7026
		Flights	0.9895	0.9889	N/A	N/A	N/A	N/A
		Can. Parl.	0.8498	0.8872	0.8315	0.8879	0.8169	0.8586
		Contacts	0.9887	0.9861	0.9768	0.9709	0.9579	0.9587
<b>AP</b>		UCI	0.9566	0.9562	0.7899	0.7893	0.7901	0.7894
		LastFM	0.9516	0.9460	0.8200	0.8099	0.8201	0.8099
		Flights	0.9791	0.9726	N/A	N/A	N/A	N/A
		Can. Parl.	0.6519	0.6170	0.6687	0.5999	0.6662	0.5978
		Contacts	0.9834	0.9785	0.9331	0.9485	0.9331	0.9485
<b>Inductive</b>		UCI	0.9426	0.9420	0.7164	0.7177	0.7165	0.7180
		LastFM	0.9501	0.9433	0.7679	0.7555	0.7679	0.7555
		Flights	0.9799	0.9748	N/A	N/A	N/A	N/A
		Can. Parl.	0.6484	0.6415	0.6662	0.6148	0.6636	0.6107
		Contacts	0.9847	0.9822	0.9331	0.9452	0.9331	0.9452

I risultati mostrano che l'introduzione della media pesata **non comporta miglioramenti** rispetto al modello base. Tale comportamento si può ricondurre al fatto che gli autori hanno individuato i valori ottimali di  $\lambda$  tramite *grid search* e quindi durante l'addestramento la rete potrebbe tendere naturalmente a concentrare il peso su tale valore, mentre le restanti si comportano quasi come rumore. Di conseguenza, la maggiore complessità introdotta dal meccanismo di pesatura non si traduce in un beneficio concreto nelle prestazioni.

### 3.3 Prodotto con forma bilineare

Nel modello originale di *TPNet*, la **pairwise feature grezza** tra due nodi  $u$  e  $v$  viene calcolata a partire dalle rispettive rappresentazioni  $F_u$  e  $F_v$ . Tali rappresentazioni vengono per prima cosa concatenate:

$$F_{u,v} = [F_u, F_v] \in \mathbb{R}^{2(k+1) \times d_R}$$

per poi effettuare il prodotto interno (ottenendo quindi una matrice  $\in \mathbb{R}^{2(k+1) \times 2(k+1)}$ ) e applicare un *flattening* in un vettore:

$$\tilde{f}_{u,v} = \text{flat}(F_{u,v} F_{u,v}^\top) \in \mathbb{R}^{4(k+1)^2}$$

Il vettore  $\tilde{f}_{u,v}$  così ottenuto viene poi fornito in ingresso ad una MLP per ottenere la vera e propria *pairwise feature*, in particolare:

$$f_{u,v} = \text{MLP}(\log(\text{ReLU}(\tilde{f}_{u,v}) + 1))$$

Il prodotto interno, però, è rigido. Esso consente di confrontare componenti corrispondenti dei due vettori, ma non consente interazioni tra dimensioni diverse. Nel contesto di *TPNet*, significa che non è possibile combinare informazioni provenienti da *temporal walk* di lunghezze differenti. Inoltre, il prodotto interno non permette di pesare alcune dimensioni più di altre.

Per provare a superare ciò, è stata introdotta una **forma bilineare**:

$$\tilde{f}_{u,v} = \text{flat}(F_{u,v} W F_{u,v}^\top)$$

dove  $W \in \mathbb{R}^{d_R \times d_R}$  è una **matrice di pesi learnable**, quindi, durante l'addestramento, i valori vengono ottimizzati consentendo al modello di apprendere interazioni tra le componenti di  $F_u$  e  $F_v$ .

L'operazione di *flattening* e la successiva rete MLP rimangono invariate.

Sono state considerate due diverse inizializzazione della matrice dei pesi  $W$ :

- **Inizializzazione con matrice identità**
- **Inizializzazione di Xavier**

Per valutare lo scostamento finale, dopo il training è stata presa in analisi la seguente quantità:

$$\Delta = W - I$$

In particolare è stato prodotto un istogramma che rappresenta la distribuzione degli scostamenti degli elementi di  $W$  rispetto all'identità.

### **Inizializzazione con matrice identità**

La prima configurazione della matrice  $W$  prevede un'inizializzazione pari alla matrice identità in modo da partire dal comportamento di default di *TPNet* e replicare quindi il prodotto interno.

$$W_0 = I$$

Si è voluto verificare se la forma bilineare introduce effettivamente modifiche oppure la matrice  $W$  rimane "simile" alla matrice identità.

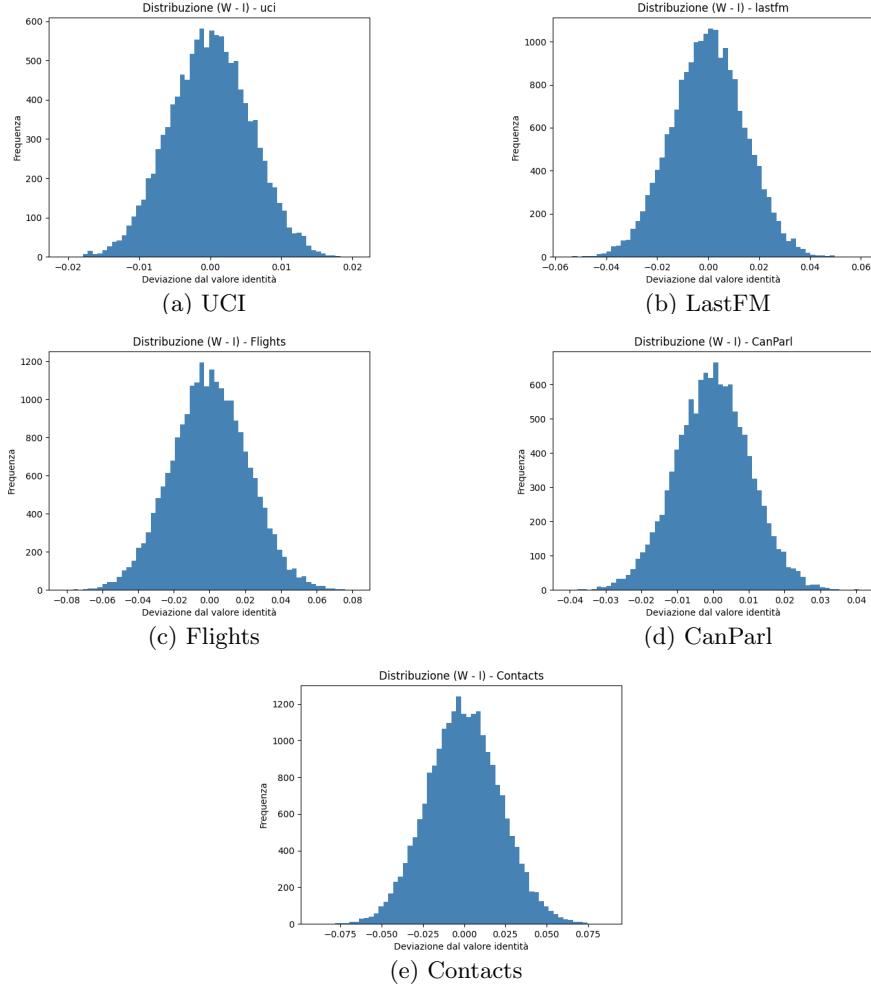


Figura 1: Distribuzione degli scostamenti  $\Delta = W - I$  con inizializzazione pari alla matrice identità

La figura mostra che la deviazione rispetto alla matrice identità, a fine dell'addestramento, risulta estremamente piccola, con valori compresi entro  $\pm 0.04$ . Questo comportamento indica che, nonostante  $W$  sia una matrice *learnable*, il processo di ottimizzazione la mantiene vicino l'originale.

Bisogna osservare che in seguito il prodotto viene passato attraverso una MLP e quindi essa potrebbe "assorbire" qualsiasi trasformazione introdotta da  $W$ .

## Inizializzazione di Xavier

Per evitare che l'inizializzazione identitaria ancorasse il modello alla soluzione banale  $W \approx I_{d_R}$ , è stata testata una seconda strategia basata sull'inizializzazione uniforme di Xavier:

$$W_0 \sim \mathcal{U}(-\alpha, \alpha), \quad \alpha = \sqrt{\frac{6}{\text{fan\_in} + \text{fan\_out}}}.$$

In questo caso *fan-in* e *fan-out* coincidono e sono pari alla dimensione dello spazio delle proiezioni casuali:

$$\text{fan\_in} = \text{fan\_out} = d_R$$

Anche in questo caso è stata esaminata la deviazione rispetto alla matrice identità analizzando la distribuzione degli elementi della differenza  $W - I$  dopo l'inizializzazione di Xavier.

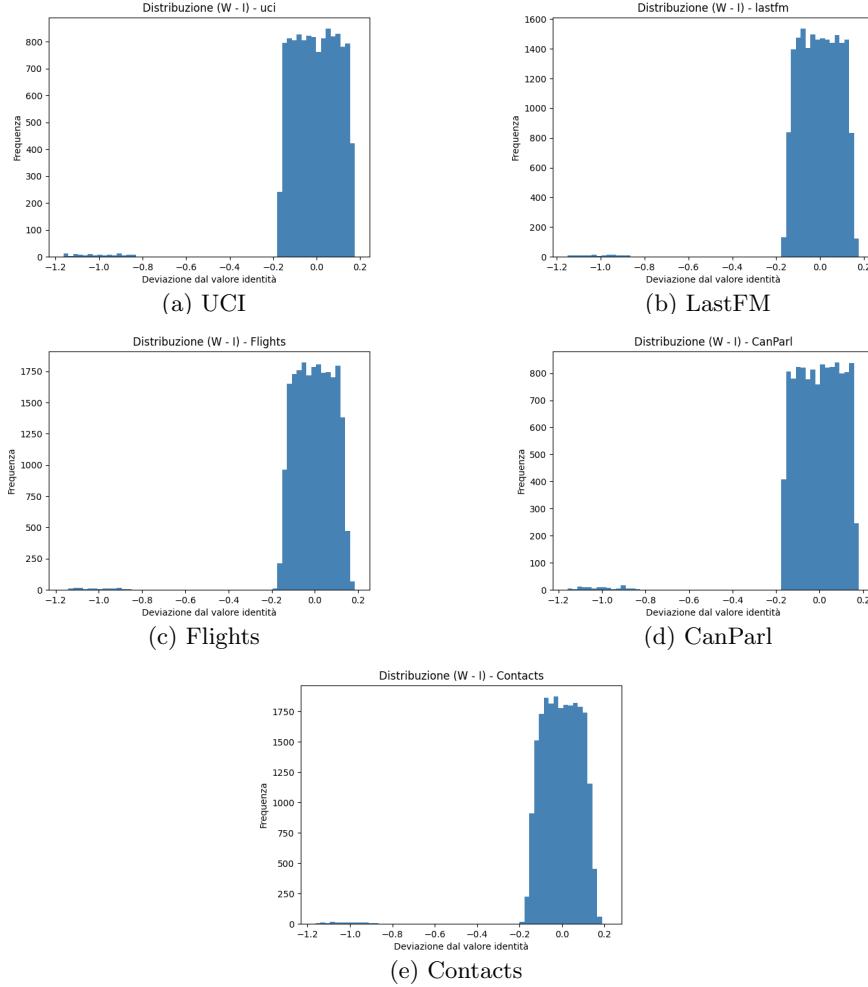


Figura 2: Distribuzione degli scostamenti  $\Delta = W - I$  con inizializzazione di Xavier

Gli istogrammi mostrano che la distribuzione degli scostamenti è sostanzialmente sovrapponibile per tutti i dataset analizzati.

In ogni caso, la matrice  $W$ , seppur resa learnbale, rimane vicina alla condizione di inizializzazione. Questo comportamento è coerente con quanto osservato anche nel caso di inizializzazione con la matrice identità. L'addestramento non porta ad una trasformazione significativa della matrice. Come indicato anche nel caso precedente, considerato che successivamente il prodotto viene dato in pasto ad una MLP, la capacità adattiva è demandata ad essa.

In questa configurazione l'intero set di test è stato rieseguito; la tabella seguente

contiene i risultati ottenuti.

Tabella 6: Confronto tra modello originale (*Replica*) e con modifica (*Modifica*) nei due setting di *link prediction* e nei tre tipi di *negative sampling*.

Setting	Misura	Dataset	Random		Historical		Inductive	
			Replica	Modifica	Replica	Modifica	Replica	Modifica
<b>AP</b>	<b>AUC</b>	UCI	0.9732	0.9626	0.8681	0.8627	0.7787	0.7579
		LastFM	0.9404	0.8911	0.8712	0.8160	0.7704	0.7005
		Flights	0.9888	0.9837	N/A	N/A	N/A	N/A
		Can. Parl.	0.8360	0.8007	0.8275	0.8174	0.8099	0.7865
		Contacts	0.9860	0.9830	0.9802	0.9790	0.9583	0.9533
<b>Transductive</b>	<b>AUC</b>	UCI	0.9674	0.9542	0.8073	0.8108	0.7116	0.7051
		LastFM	0.9384	0.8906	0.8415	0.7956	0.7165	0.6598
		Flights	0.9895	0.9837	N/A	N/A	N/A	N/A
		Can. Parl.	0.8498	0.8663	0.8315	0.8728	0.8169	0.8184
		Contacts	0.9887	0.9861	0.9768	0.9756	0.9579	0.9545
<b>AP</b>	<b>AUC</b>	UCI	0.9566	0.9465	0.7899	0.7674	0.7901	0.7676
		LastFM	0.9516	0.9205	0.8200	0.7511	0.8201	0.7511
		Flights	0.9791	0.9689	N/A	N/A	N/A	N/A
		Can. Parl.	0.6519	0.5790	0.6687	0.5980	0.6662	0.5996
		Contacts	0.9834	0.9793	0.9331	0.9381	0.9331	0.9382
<b>Inductive</b>	<b>AUC</b>	UCI	0.9426	0.9296	0.7164	0.7082	0.7165	0.7083
		LastFM	0.9501	0.9199	0.7679	0.7156	0.7679	0.7156
		Flights	0.9799	0.9687	N/A	N/A	N/A	N/A
		Can. Parl.	0.6484	0.5991	0.6662	0.6098	0.6636	0.6102
		Contacts	0.9847	0.9820	0.9331	0.9399	0.9331	0.9399

I risultati mostrano che l'introduzione della forma bilineare con inizializzazione di Xavier non produce miglioramenti rispetto al modello originale, anzi, porta a prestazioni peggiori.

### 3.4 Time-Encoder MLP

Nel modello originale di *TPNet*, il tempo è codificato mediante l'applicazione di funzione sinusoidale ai *timesteps*. In particolare, per ogni nodo in analisi  $u$  si effettua il *mapping* tra *timesteps* in *feature temporali*:

$$X_{u,T} = [\phi(t - t_1), \dots, \phi(t - t_n)]$$

con la funzione di encoding  $\phi$  definita come segue:

$$\phi(\Delta t) = [\cos(w_1 \Delta t), \dots, \cos(w_{d_T} \Delta t)]$$

I parametri  $w$  sono inizializzati su una scala logaritmica e ottimizzati durante il training.

Per rendere la rappresentazione temporale più espressiva è stata introdotta una nuova classe **TimeEncoderV2**, che sostituisce la codifica sinusoidale con una rete

neurale *MLP* a due layer. Il modello risultante è:

$$\phi(\Delta t) = \text{LayerNorm}\left(W_2 \text{ReLU}(W_1 \Delta t + b_1) + b_2\right)$$

dove  $W_1 \in \mathbb{R}^{d_T \times 1}$  e  $W_2 \in \mathbb{R}^{d_T \times d_T}$  sono parametri apprendibili. La funzione di attivazione ReLU introduce non linearità, mentre la LayerNorm stabilizza la distribuzione.

Questa variante consente al modello di apprendere una rappresentazione del tempo più flessibile e adattiva rispetto a quella sinusoidale, potenzialmente migliorando la capacità di catturare relazioni temporali irregolari presenti nei dati.

I risultati ottenuti con il nuovo *Time-Encoder* sono riportati nella tabella che segue:

Tabella 7: Confronto tra modello originale (*Replica*) e con modifica (*Modifica*) nei due setting di *link prediction* e nei tre tipi di *negative sampling*.

Setting	Misura	Dataset	Random		Historical		Inductive	
			Replica	Modifica	Replica	Modifica	Replica	Modifica
<b>AP</b>		UCI	0.9732	0.9734	0.8681	0.8604	0.7787	0.7622
		LastFM	0.9404	0.0903	0.8712	0.8732	0.7704	0.7763
		Flights	0.9888	0.9888	N/A	N/A	N/A	N/A
		Can. Parl.	0.8360	0.8990	0.8275	0.8257	0.8099	0.8488
		Contacts	0.9860	0.9859	0.9802	0.9810	0.9583	0.9588
<b>Transductive</b>		UCI	0.9674	0.9670	0.8073	0.8017	0.7116	0.6996
		LastFM	0.9384	0.9386	0.8415	0.8450	0.7165	0.7222
		Flights	0.9895	0.9895	N/A	N/A	N/A	N/A
		Can. Parl.	0.8498	0.9151	0.8315	0.8156	0.8169	0.8448
		Contacts	0.9887	0.9886	0.9768	0.9778	0.9579	0.9587
<b>AP</b>		UCI	0.9566	0.9580	0.7899	0.7678	0.7901	0.7679
		LastFM	0.9516	0.9510	0.8200	0.8218	0.8201	0.8218
		Flights	0.9791	0.9794	N/A	N/A	N/A	N/A
		Can. Parl.	0.6519	0.6903	0.6687	0.7055	0.6662	0.7060
		Contacts	0.9834	0.9823	0.9331	0.9724	0.9331	0.9275
<b>Inductive</b>		UCI	0.9426	0.9440	0.7164	0.7010	0.7165	0.7013
		LastFM	0.9501	0.9503	0.7679	0.7692	0.7679	0.7692
		Flights	0.9799	0.9801	N/A	N/A	N/A	N/A
		Can. Parl.	0.6484	0.7022	0.6662	0.7052	0.6636	0.7048
		Contacts	0.9847	0.9838	0.9331	0.9302	0.9331	0.9303

I risultati mostrano che il *Time-Encoder* basato su *MLP* non altera in modo significativo le prestazioni complessive: nella maggior parte dei dataset le differenze rispetto alla codifica sinusoidale sono minime. L’unico miglioramento evidente riguarda il dataset **Canadian Parliament**, dove la versione con *MLP* ottiene risultati superiori in tutti i setting.

Questo comportamento si può ricondurre alla natura del dataset: le interazioni presentano dinamiche temporali irregolari che la codifica sinusoidale non riesce a rappresentare in modo ottimale. La presenza di una componente non lineare

apprendibile consente invece al modello di adattarsi a tali variazioni, catturando evoluzioni temporali più complesse.

Negli altri dataset, l'encoder sinusoidale risulta già sufficiente; di conseguenza, la maggiore espressività dell'MLP non si traduce in un miglioramento delle prestazioni, né produce vantaggi tangibili.

## 4 Conclusioni

Il presente elaborato ha esplorato il modello **TPNet**, approfondendone il funzionamento, replicandone gli esperimenti e introducendo una serie di modifiche architettoniche con l’obiettivo di migliorarne la capacità.

Una prima estensione ha riguardato l’introduzione di **più valori di decadimento temporale**  $\lambda$ . Sono stati sperimentati due approcci distinti: la *media semplice* delle scale e la *media pesata*. In entrambi i casi, nonostante l’utilizzo di un solo valore di  $\lambda$  fosse un limite portato alla luce dagli autori stessi, gli esperimenti hanno mostrato risultati simili al modello originale.

Un secondo tentativo di estensione ha riguardato l’introduzione di una **forma bilineare** al posto di prodotto interno per la costruzione delle pairwise features. L’obiettivo era consentire al modello di apprendere una combinazione delle rappresentazioni dei nodi più flessibile e più significativa. Tuttavia, gli esperimenti indicano che la matrice dei pesi rimane vicina ai valori con i quali viene inizializzata.

Diversamente dalle modifiche precedenti, l’introduzione di un **Time-Encoder basato su MLP** in sostituzione ad un *Time-Encoder sinusoidale* ha portato a un miglioramento concreto. La versione *learnable* permette di apprendere direttamente una rappresentazione più flessibile del tempo.

Nel complesso, i risultati indicano che **TPNet è un modello solido**. Le modifiche introdotte, pur concettualmente promettenti, non hanno portato ai miglioramenti attesi. L’unica estensione in grado di incrementare la qualità predittiva è quella relativa alla codifica temporale.

In conclusione, il lavoro mostra che tecniche di introduzione e aggregazione di più valori del parametro di decadimento temporale *semplici* (come media e media pesata) non sono sufficienti per ottenere miglioramenti. Meccanismi più complessi potrebbero effettivamente superare i limiti individuati dagli autori del modello. Un risultato positivo è invece emerso nella componente di codifica temporale: rendere totalmente *learnable* il *Time-Encoder* ha permesso di ottenere l’unico miglioramento consistente.