Leveraging Discrete Time Dynamic Graph with global attention.

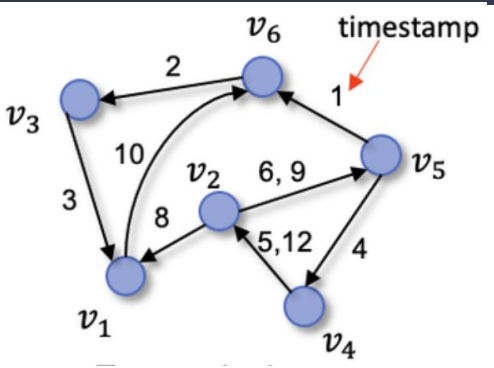
Y.Karmim, R. Fournier, N. Thome

le cnam

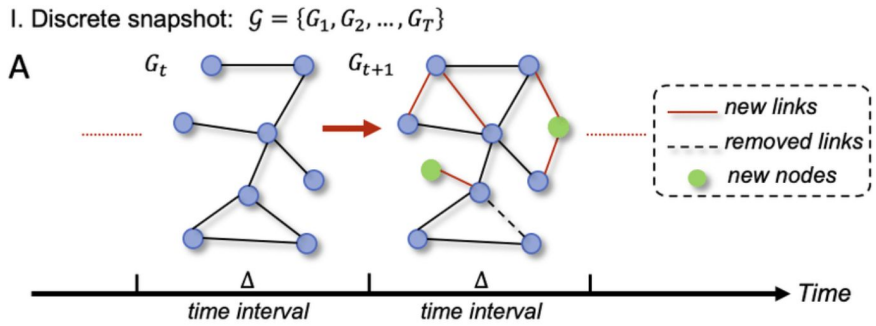
4/04/2024

Discrete-Time Dynamic Graphs

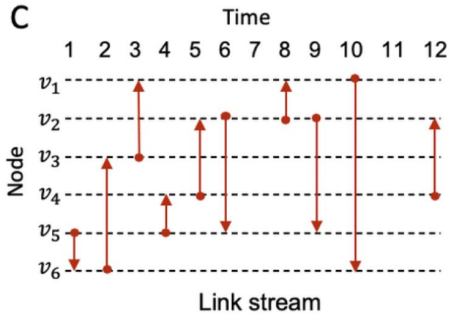
How to represent a dynamic graph



Edge-weighted graph. TGN slides Rossi



Discrete Graph (DTDG). TGN slides Rossi



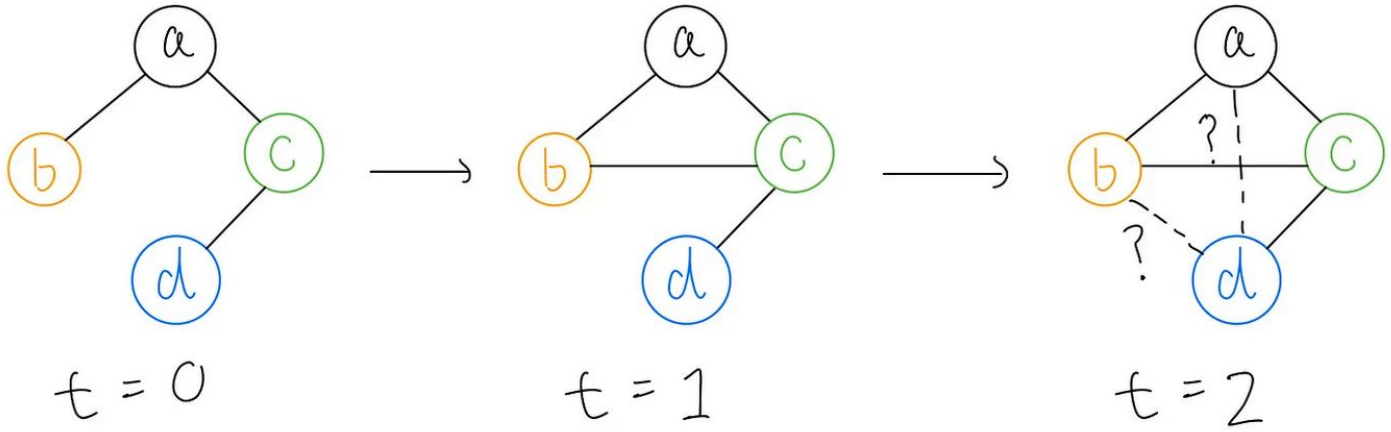
Continuous graph (CTDG). TGN slides Rossi

Focus on DTDG

Some tasks

Dynamic Link Prediction : From the history of the graphs → Predict the future connections

Close to a classification task: Classify a negative link to 0 and a positive link to 1.



Classic Training Framework on DTDG

Classical Training of Dynamic Graph Neural Networks (DGNNs) on DTDG;
combine a GNN (spatial) and a time-serie model (dynamic)

Spatial Learning

$$Z^t = GNN(G_t)$$

$$H^t = f(H^{t-1}, Z^t)$$

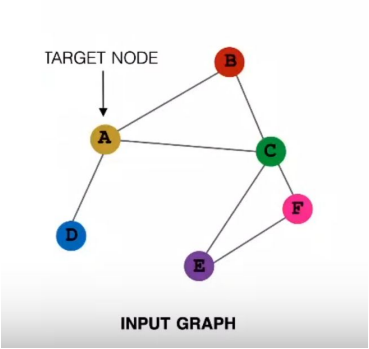
Dynamic Learning

- Main difference: choice of **spatial** model and a **time-serie** function *f*.
  GNN learn the structure of current snapshot.
  f update dynamically node embedding.

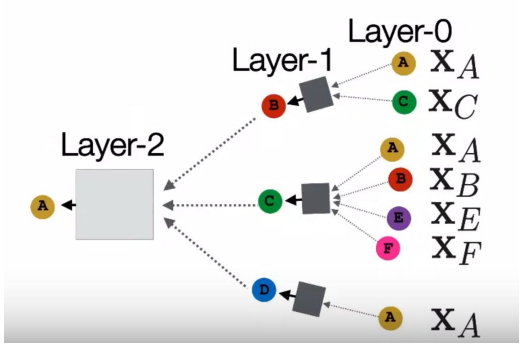
- **Examples from state of the art methods.**

Model	Spatial (GNN)	Dynamic (f)
LSTM-GCN (2017)	GCN (Kipf16)	LSTM
EvolveGCN (2020)	GCN (Kipf16)	GRU
DySat (2019)	GAT (Velikcovic17)	Transformer1D (AttentionNeed17)
ROLAND (2022)	Generalization to all GNN	GRU / MLP / Moving Average

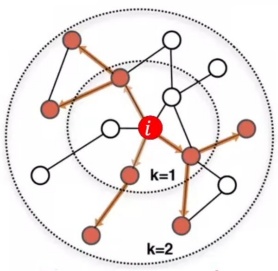
Message Passing (MP-GNNs)

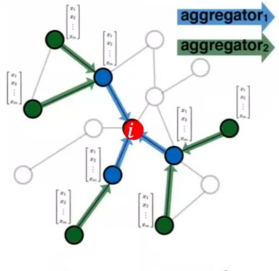


Cours CS224w (Lescovec 2019)



Idea: Propagate and aggregate messages (i.e embeddings) from neighborhood *(GCN Kipf 2016)*





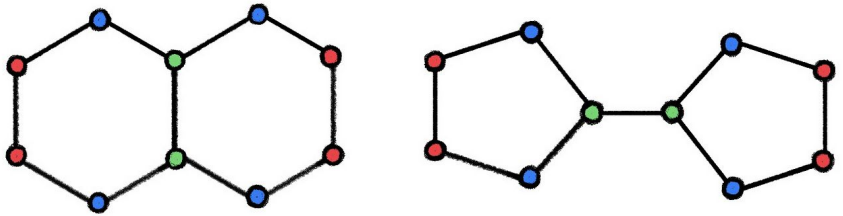
GraphSage (Hamilton 2017)

$$\mathbf{h}_u^{(k)} = \sigma\left(\mathbf{W}_{\text{self}}^{(k)}\mathbf{h}_u^{(k-1)} + \mathbf{W}_{\text{neigh}}^{(k)}\sum_{v\in\mathcal{N}(u)}\mathbf{h}_v^{(k-1)} + \mathbf{b}^{(k)}\right)$$

Original Message-Passing (*Merwirth 2005)*

Some identified limitations of MP-GNNs

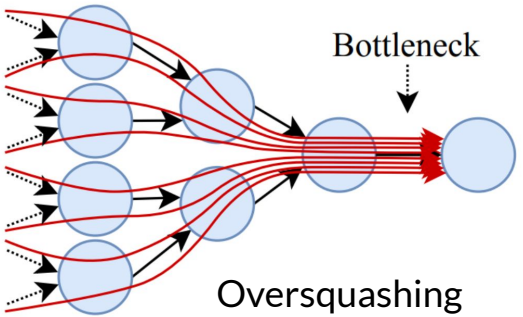
- Limited Expressivity
  MP-GNNs can't differentiate some structures. *(Weisfeiler-Lehman 1997)*



As the WL test, MP-GNNs can't differentiate those two structures (How powerful are GNN, Xu 2018)

- Oversmoothing
  Adding to many layers → all nodes will share same representations (*Oversmoothing Li 2018*)

- Oversquashing
  Receptive field grow exponentially by aggregating neighbor of neighbor (*Bottleneck Alon 2020)*



Limitations of existing DGNNs architectures

Recently WL test has been extended to DTDG with an analysis of the expressive power of existing DGNNs architectures*.

**Beddar-Wiesing, S. et al. (2022). Weisfeiler--Lehman goes Dynamic: An Analysis of the Expressive Power of Graph Neural Networks for Attributed and Dynamic Graphs. Neural Networks.*

Important statements

Def: "If the two graphs are dynamic, they are called to be isomorphic if and only if the static graph snapshots of each timestep are isomorphic."

Theorem : Let G a DTDG, N is the maximal number of nodes in a snapshot. Then there exists a DGNN composed by GNN with 2N-1 layers a hidden dimension r = 1 and a RNN with a state dimension of 1 that can approximate the dynamic system.

Limitations of existing DGNNs architectures

- **To be an universal approximator of dynamic system, a DGNN must stack 2N-1 layers.**

- **Stacking to many layers can lead to oversmoothing (Oversmoothing Li 2018) and oversquashing (Bottleneck Alon 2020).**

- **Difficult to train deep GNNs and hard to capture dynamic dependencies between nodes (1 to 1 memory update).**

Contributions and motivations

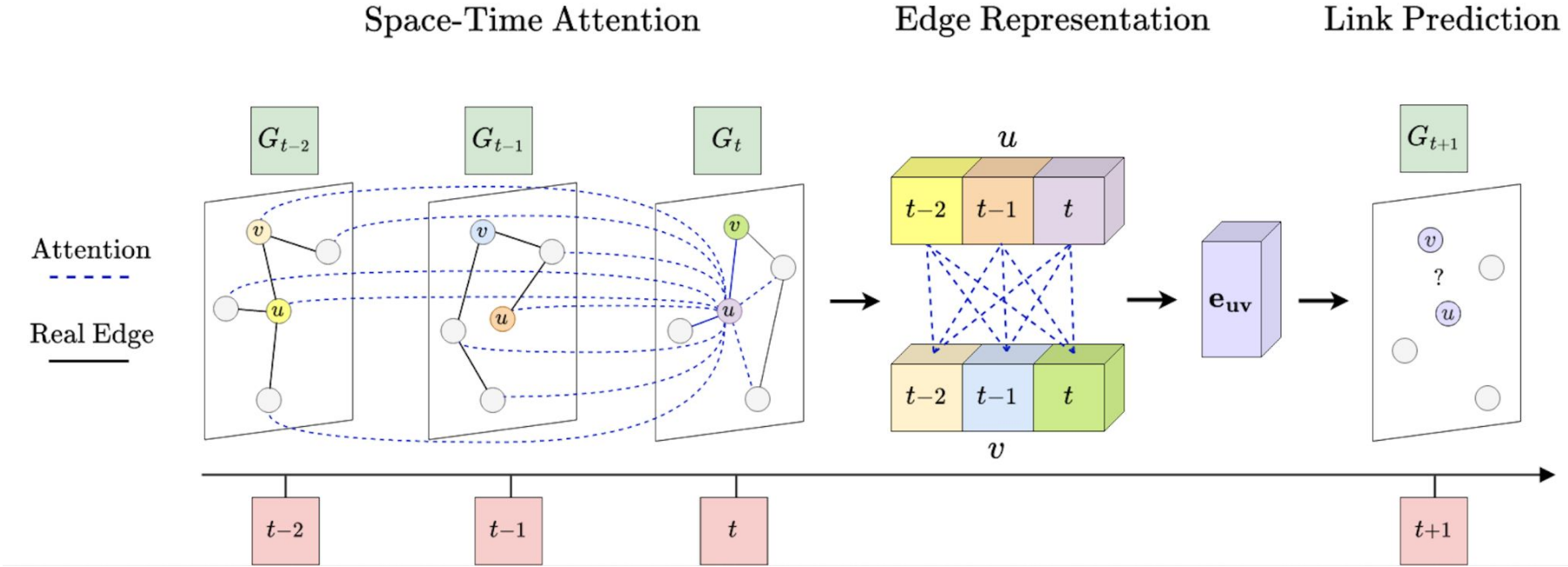
Ideas and motivations

- 1 layer of global attention can resolve the above-mentioned limitations.

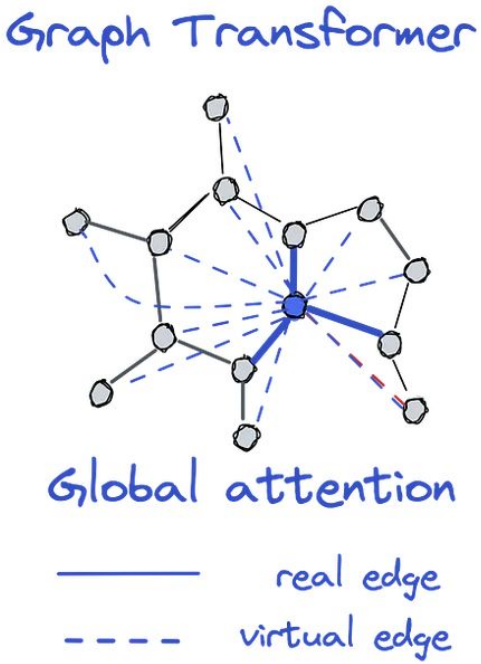
- Interconnect all nodes at any time-steps to model Spatio-Temporal dependencies.

- Show that attention models scenarios are more powerful than DGNNs (RNN-GNN) on real-world DTDG.

- Take benefits of the Full Attention Transformer to construct pairwise representation for dynamic link prediction


Space-Time Attention
Edge Representation
Link Prediction
Attention
Real Edge
G_{t-2}
G_{t-1}
G_t
G_{t+1}
u
v
$t-2$
$t-1$
t
$\mathbf{e_{uv}}$
?
$t-2$
$t-1$
t
$t+1$

Beyond MP-GNNs: Graph Transformer (GT)



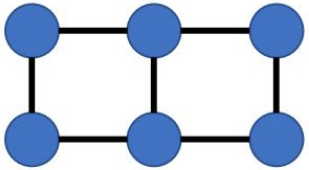
GraphGPS (Rampasek 2023)

- Adapted from Transformers in NLP and Vision *(Generalization Dwivedi 2020)*
- Interconnect all nodes → Lost of the structure.
- Positional encoding (PE) and structural encoding (SE) inform the transformer about the graph structure
- Universal approximator with the right sets of PE (Kreuzer 2021). More expressive than any k-WL test.

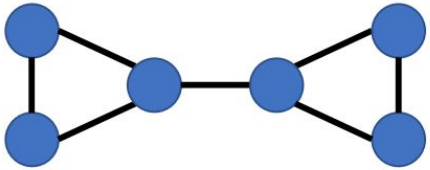
Intuition : Learn an augmented graph through attention factor (bridge with Graph Structure Learning)

Beyond MP-GNNs: Graph Transformer (GT)

Graph 1



Graph 2



Eigenvalues of Graph 1	Eigenvalues of Graph 2
0	0
1	0.438
2	3
3	3
3	3
5	4.562

Beyond MP-GNNs: Graph Transformer

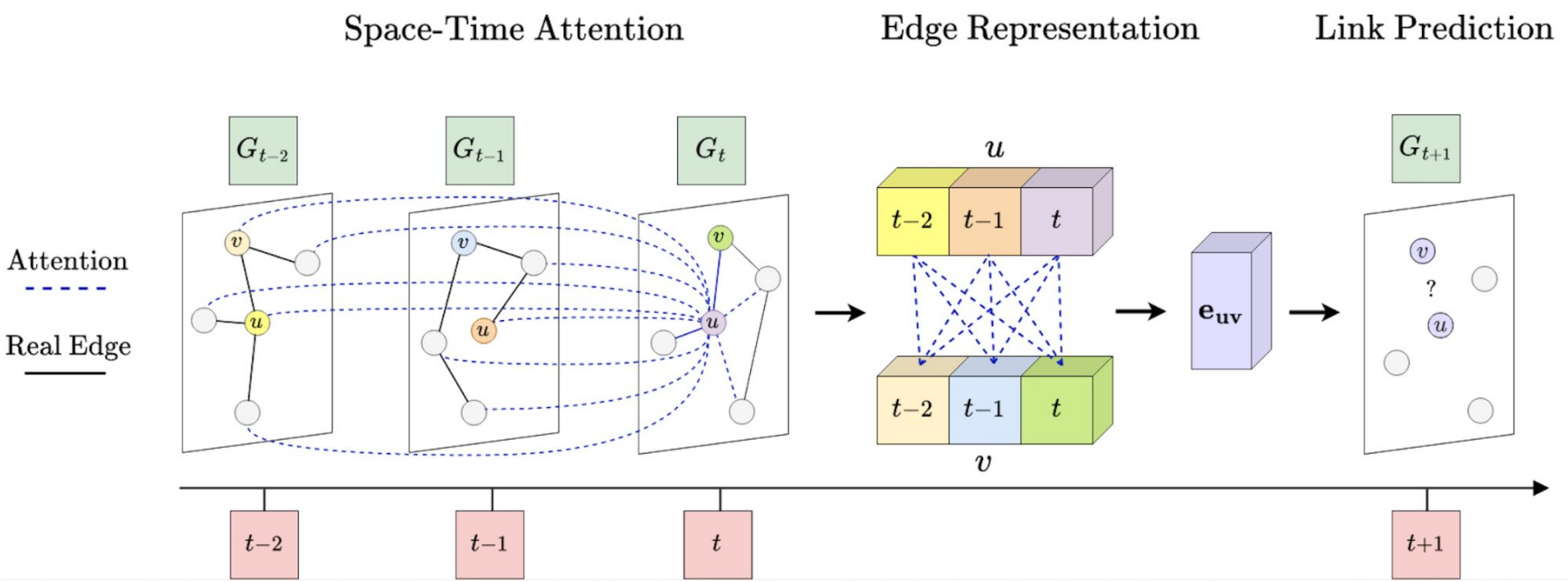
Table 1: The proposed categorization of positional encodings (PE) and structural encodings (SE). Some encodings are assigned to multiple categories in order to show their multiple expected roles.

Encoding type	Description	Examples
Local PE *node features*	Allow a node to know its position and role within a local cluster of nodes. *Within a **cluster**, the closer two nodes are to each other, the closer their local PE will be, such as the position of a word in a sentence (not in the text).*	• Sum each column of non-diagonal elements of the m-steps random walk matrix. • Distance between a node and the centroid of a cluster containing the node.
Global PE *node features*	Allow a node to know its global position within the graph. *Within a **graph**, the closer two nodes are, the closer their global PE will be, such as the position of a word in a text.*	• Eigenvectors of the Adjacency, Laplacian [15, 36] or distance matrices. • SignNet [39] (includes aspects of relative PE and local SE). • Distance from the graph's centroid. • Unique identifier for each connected component of the graph.
Relative PE *edge features*	Allow two nodes to understand their distances or directional relationships. *Edge embedding that is correlated to the distance given by any global or local PE, such as the distance between two words.*	• Pair-wise node distances [38, 3, 36, 63, 44] based on shortest-paths, heat kernels, random-walks, Green's function, graph geodesic, or any local/global PE. • Gradient of eigenvectors [3, 36] or any local/global PE. • PEG layer [57] with specific node-wise distances. • Boolean indicating if two nodes are in the same cluster.
Local SE *node features*	Allow a node to understand what substructures it is a part of. *Given an SE of radius m, the more similar the m-hop subgraphs around two nodes are, the closer their local SE will be.*	• Degree of a node [63]. • Diagonal of the m-steps random-walk matrix [16]. • Time-derivative of the heat-kernel diagonal (gives the degree at $t = 0$). • Enumerate or count predefined structures such as triangles, rings, etc. [6, 68]. • Ricci curvature [54].
Global SE *graph features*	Provide the network with information about the global structure of the graph. *The more similar two graphs are, the closer their global SE will be.*	• Eigenvalues of the Adjacency or Laplacian matrices [36]. • Graph properties: diameter, girth, number of connected components, # of nodes, # of edges, nodes-to-edges ratio.
Relative SE *edge features*	Allow two nodes to understand how much their structures differ. *Edge embedding that is correlated to the difference between any local SE.*	• Pair-wise distance, encoding, or gradient of any local SE. • Boolean indicating if two nodes are in the same substructure [5] (similar to the gradient of sub-structure enumeration).

Exhaustive list of different structural and positional encoding. (Rampasek 2023)

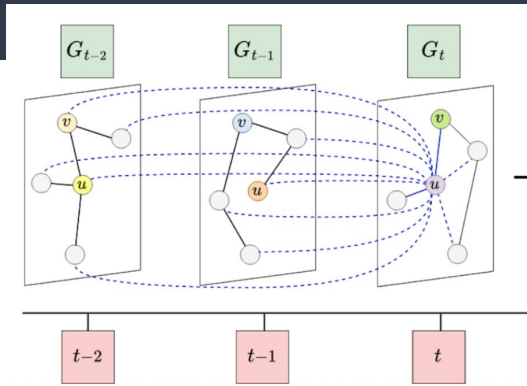
- Node degree
- Random Walk
- Distances (SPD)
- ...

Full Attention on Dynamic Graph



Two Main contributions; Space-Time Attention and Edge Representation

Full Space-Time Attention

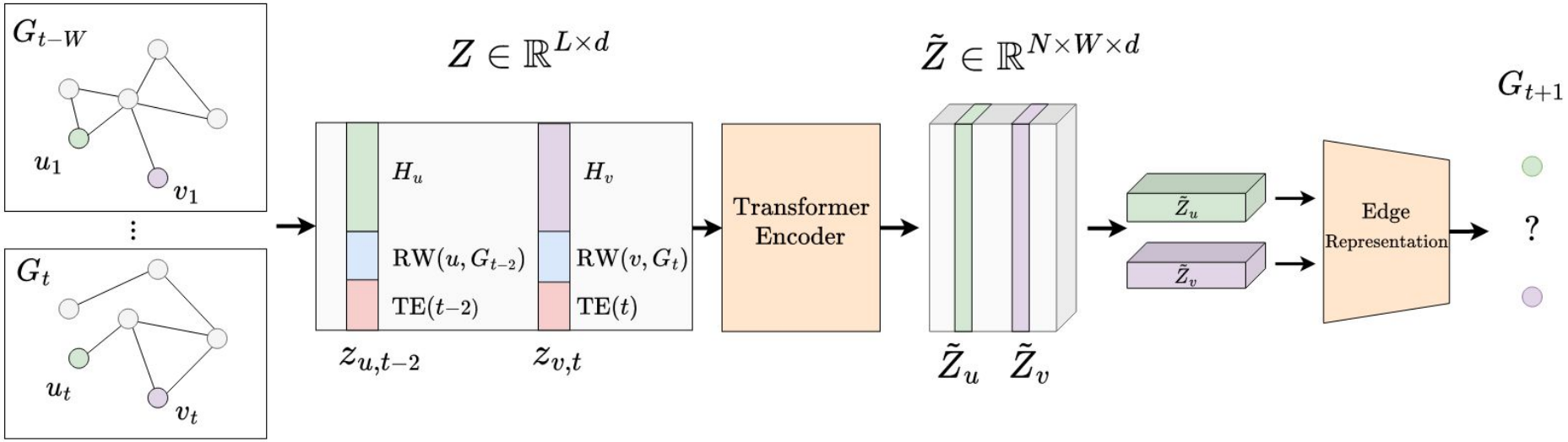


- **Idea: Apply a global attention on DTDG with Transformer**

- Each node at each time-steps is an independent token.

- **We lose both the structural / positional and temporal information of the DTDG.**

- Transformer must be informed about :
  The position / structure of the graph around the token
  The time step in which the token is located.



For each token z_u we inject temporal (TE) and structural information (RW) before the Full Attention

Incorporate Structural and Temporal Encoding

Structural Encoding

Aim : Encoding the structure of the *t-th* snapshot

1. *Self - Random Walk :*

$$\mathrm{rwPE}_i^t = (RW_{ii}, RW_{ii}^2, ..., RW_{ii}^{d_{\mathrm{pos}}})$$

2. *Laplacian*

$$\Delta^t = I - D^{t^{-1/2}} A^t D^{t^{-1/2}} = U^{t^T} \Lambda U^t$$

$$\mathrm{lapPE}_i^t = (U_{i,1}^t, U_{i,2}^t, ..., U_{i,d_{\mathrm{pos}}}^t)$$

3. *GNN*

$$\mathrm{gnnPE}_i^t = Z_i = GNN(G^t)$$

Temporal Encoding

Aim: Encoding the position of the snapshot in the DTDG

$$\mathrm{timePE}(t) = H_{\mathrm{time}}[t].$$

Similar to BERT we learn the embedding position of the snapshot in the sequence.

Token construction

Token = Concatenation of the node, structural and time embedding

g = MLP

$$z_{i,t} = g(H_i \oplus \text{posPE}_i^t \oplus \text{timePE}_t),$$

This, compose our token Matrix Z containing all of our nodes at each timestep

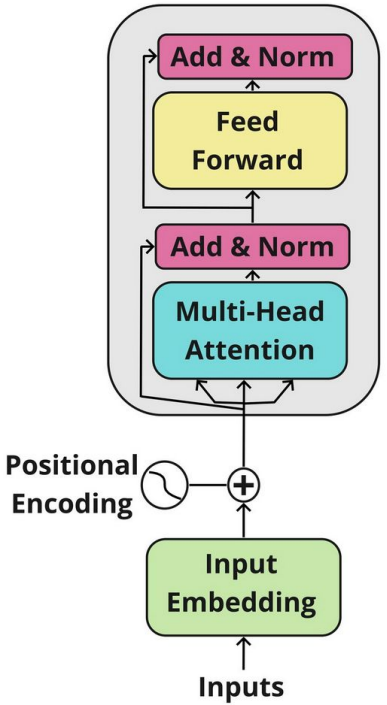
Self Attention in Z.

Attention computing

The attention between all nodes at each time, is computed as follow

$$\mathrm{SA}(Z) = \mathrm{softmax}\left(\frac{ZQZ^T K^T}{\sqrt{d}}\right) ZV,$$

One layer of a Transformer Encoder is enough *(Simplifying Wu 2023)*



Efficient Implementation

Time Window

In most of real DTDG, a very long term dependency is not crucial.

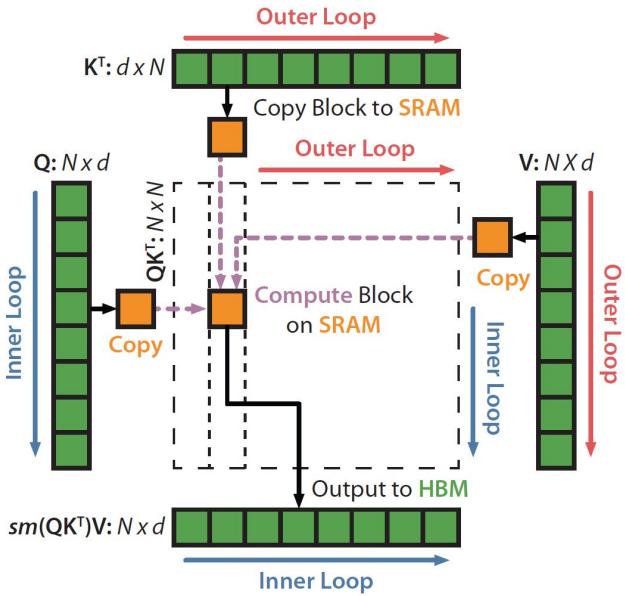
We use a modulable time-window to capture the necessary time context to perform

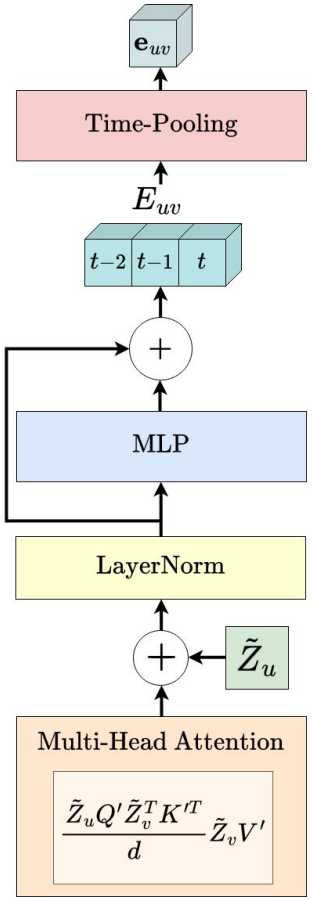
W is a hyperparameter.

We reduce the number of token:
T x N to W x N, with W << T.

Flash Attention 2.0

Reduce the memory complexity





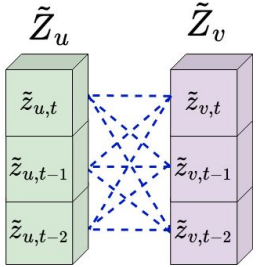
Training

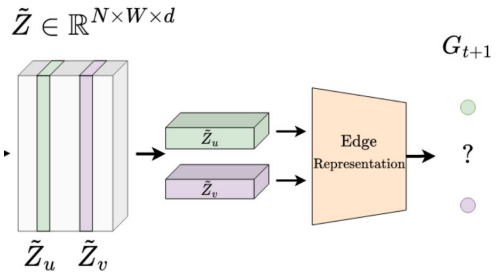
$$s_{uv_{\text{pos}}} = g(\mathbf{e}_{uv_{\text{pos}}})$$

$$s_{uv_{\text{neg}}} = g(\mathbf{e}_{uv_{\text{neg}}})$$

(Eq.8)

$$\mathcal{L} = \text{BCE}(s_{uv}, \text{label})$$





- After the space-time attention on DTDG $\rightarrow \tilde{Z}$ of dimension N (nodes) W (time steps) and d (emb dim).
- For each node u and v: W representations.
- Aim : Capture dependencies between those two representations with cross attention

Edge Representation Module

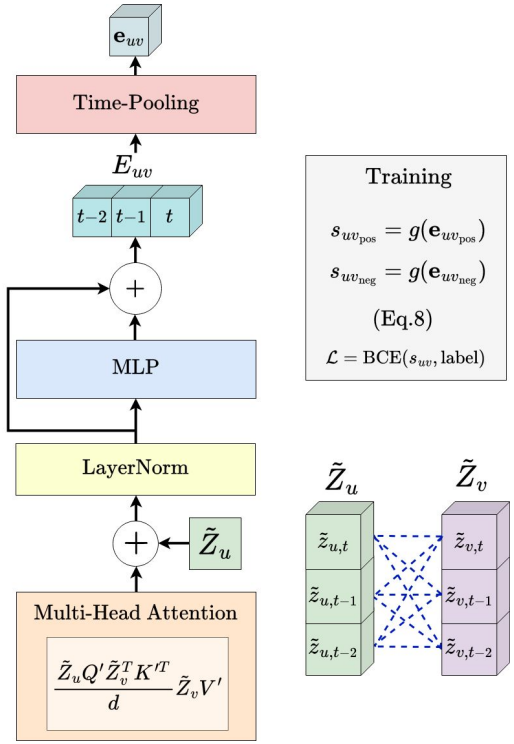
Edge Representation with cross-attention

The aim is to predict from the snapshots $\{G^i\}_{i=1}^{t-1}$ the links E^t, of G^t.

For u, v we have W representations in $\{G^i\}_{i=1}^{t-1}$ respectively

$$E_{uv} = \text{LayerNorm}\big(\text{MHCA}(\tilde{Z}_u, \tilde{Z}_v) + \tilde{Z}_u\big)$$
$$E_{uv} = f(E_{uv}) + E_{uv}$$
$$\mathbf{e}_{uv} = p(E_{uv}), \quad s_{uv} = g(E_{uv})$$

$$\text{CA}(\tilde{Z}_u, \tilde{Z}_v) = \text{softmax}\Big(\frac{\tilde{Z}_u Q' \tilde{Z}_v^T K'^T}{d}\Big)\tilde{Z}_v V'$$



Results

Custom baseline

Spatial Learning

$$Z^t = GNN(G_t)$$

$$H^t = f(H^{t-1}, Z^t)$$

Dynamic Learning

Baseline: Keep the classical learning framework

Replace GNN by a Full Attention Transformer

Take f = LSTM or GRU

Aim : Show the benefits of a full spatio-temporal attention against a full spatial attention updated by f.

LSTM-GT

Datasets

Datasets	Domains	Nodes	Links	Snapshots
CanParl	Politics	734	74,478	14
USLegis	Politics	225	60,396	12
Flights	Transports	13,169	1,927,145	122
Trade	Economics	255	507,497	32
UNVote	Politics	201	1,035,742	72
Contact	Proximity	692	2,426,279	8064
HepPh	Citations	15,330	976,097	36
AS733	Router	6,628	13,512	30
Enron	Mail	184	790	11
Colab	Citations	315	943	10

Different datasets we used in experiments.

Medium-size datasets in term of nodes.

Impact of FAST components

Table 2: Performance Comparison of LSTM-GT, FAST without `Edge`, and FAST Models. Results in ROC-AUC.

Datasets	LSTM-GT	FAST w/o `Edge`	FAST
CanParl	86.29 ± 1.10	89.45 ± 0.38	**92.37** ± 0.51
USLegis	92.64 ± 0.70	93.30 ± 0.29	**95.80** ± 0.11
Flights	95.17 ± 0.34	**99.04** ± 0.61	**99.07** ± 0.41
Trade	91.81 ± 0.11	94.01 ± 0.73	**96.73** ± 0.29
UNVote	91.38 ± 0.74	93.56 ± 0.68	**99.94** ± 0.05
Contact	92.49 ± 0.97	97.41 ± 0.10	**98.12** ± 0.37
HepPh	81.40 ± 0.45	90.44 ± 1.07	**93.21** ± 0.37
AS733	94.75 ± 0.87	96.84 ± 0.26	**97.46** ± 0.45
Enron	90.20 ± 1.12	90.57 ± 0.27	**96.39** ± 0.18
COLAB	82.95 ± 0.45	86.34 ± 0.34	**90.84** ± 0.41

LSTM-GT follow the conventional framework

FAST w/o edge is only the space-time attention on DTDG with dot product.

FAST is equipped with the edge representation module

Comparison to DTDG models

Table 3: Comparison to DTDG models on discrete data using (Yang et al., 2021) protocol.

Method	HepPh		AS733		Enron		Colab		**Avg.**
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC
GAE	69.44 ± 0.56	73.61 ± 0.58	93.21 ± 1.53	94.75 ± 0.90	92.50 ± 0.68	93.48 ± 0.64	84.57 ± 0.64	87.69 ± 0.44	84.93 ± 0.85
VGAE	72.39 ± 0.11	75.78 ± 0.06	95.76 ± 0.91	96.42 ± 0.55	91.93 ± 0.34	93.45 ± 0.49	85.16 ± 0.74	88.70 ± 0.35	86.31 ± 0.52
EvolveGCN	76.82 ± 1.46	81.18 ± 0.89	92.47 ± 0.04	95.28 ± 0.01	90.12 ± 0.69	92.71 ± 0.34	83.88 ± 0.53	87.53 ± 0.22	85.82 ± 0.68
GRUGCN	82.86 ± 0.53	85.87 ± 0.23	94.96 ± 0.35	96.64 ± 0.22	92.47 ± 0.36	93.38 ± 0.24	84.60 ± 0.92	87.87 ± 0.58	88.72 ± 0.54
DySat	81.02 ± 0.25	84.47 ± 0.23	95.06 ± 0.21	96.72 ± 0.12	93.06 ± 0.97	93.06 ± 1.05	87.25 ± 1.70	90.40 ± 1 .47	89.10 ± 0.78
VGRNN	77.65 ± 0.99	80.95 ± 0.94	95.17 ± 0.62	96.69 ± 0.31	93.10 ± 0.57	93.29 ± 0.69	85.95 ± 0.49	87.77 ± 0.79	87.97 ± 0.67
HTGN	91.13 ± 0.14	89.52 ± 0.28	**98.75** ± 0.03	**98.41** ± 0.03	94.17 ± 0.17	94.31 ± 0.26	89.26 ± 0.17	91.91 ± 0.07	93.33 ± 0.13
FAST	**93.21** ± 0.37	**90.74** ± 0.51	97.46 ± 0.45	98.16 ± 0.36	**96.39** ± 0.17	**95.40** ± 0.29	**90.84** ± 0.41	**92.15** ± 0.28	**94.48** ± 0.35

Comparison to CTDG models

Table 4: Comparison to CTDG models on discrete data using (Yu et al., 2023) protocol (AUC).

Method	CanParl	USLegis	Flights	Trade	UNVote	Contact	**Avg.**
JODIE	78.21 ± 0.23	82.85 ± 1.07	96.21 ± 1.42	69.62 ± 0.44	68.53 ± 0.95	96.66 ± 0.89	82.01 ± 0.83
DyREP	73.35 ± 3.67	82.28 ± 0.32	95.95 ± 0.62	67.44 ± 0.83	67.18 ± 1.04	96.48 ± 0.14	80.45 ± 1.10
TGAT	75.69 ± 0.78	75.84 ± 1.99	94.13 ± 0.17	64.01 ± 0.12	52.83 ± 1.12	96.95 ± 0.08	76.58 ± 0.71
TGN	76.99 ± 1.80	83.34 ± 0.43	98.22 ± 0.13	69.10 ± 1.67	69.71 ± 2.65	97.54 ± 0.35	82.48 ± 1.17
CAWN	75.70 ± 3.27	77.16 ± 0.39	98.45 ± 0.01	68.54 ± 0.18	53.09 ± 0.22	89.99 ± 0.34	77.16 ± 0.74
EdgeBank	64.14 ± 0.00	62.57 ± 0.00	90.23 ± 0.00	66.75 ± 0.00	62.97 ± 0.00	94.34 ± 0.00	73.50 ± 0.00
TCL	72.46 ± 3.23	76.27 ± 0.63	91.21 ± 0.02	64.72 ± 0.05	51.88 ± 0.36	94.15 ± 0.09	75.11 ± 0.73
GraphMixer	83.17 ± 0.53	76.96 ± 0.79	91.13 ± 0.01	65.52 ± 0.51	52.46 ± 0.27	93.94 ± 0.02	77.20 ± 0.36
DyGformer	**97.76** ± 0.41	77.90 ± 0.58	98.93 ± 0.01	70.20 ± 1.44	57.12 ± 0.62	**98.53** ± 0.01	83.41 ± 0.51
FAST	92.37 ± 0.51	**95.80** ± 0.11	**99.07** ± 0.41	**96.73** ± 0.29	**99.94** ± 0.05	98.12 ± 0.37	**96.88** ± 0.26

+ **13pt average against the last CTDG models**.

Ablations

Temporal and Structural Encoding

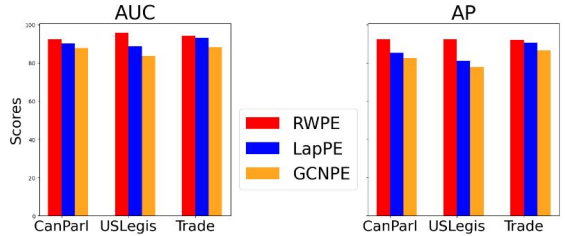


Figure 4: Comparison of Laplacian (LapPE), Random Walk (RWPE) encodings and GCN encodings.

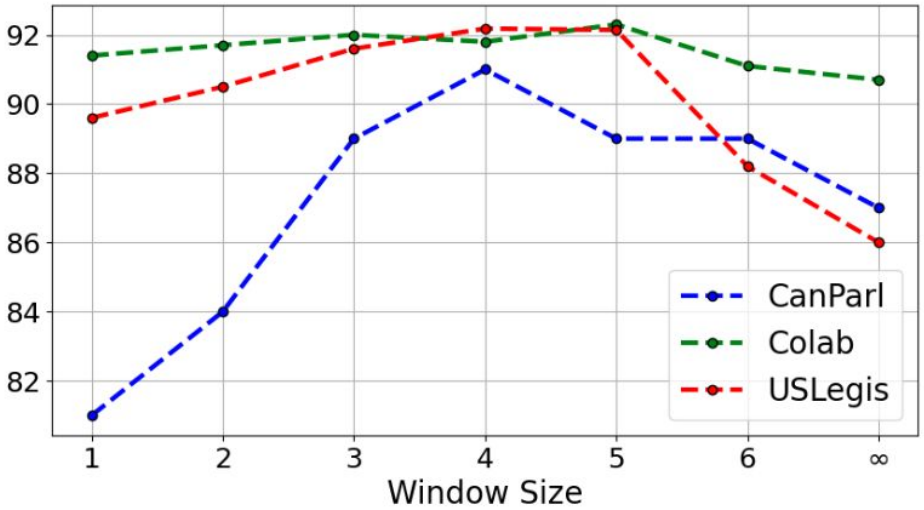
Table 5: Comparison of FAST RWPE with and without temporal encoding (TE).

	Method	AUC	AP
CanParl	FAST RWPE	**92.37** ± 0.51	**92.44** ± 0.53
	FAST RWPE w/o TE	91.67 ± 0.38	88.01 ± 0.61
USLegis	FAST RWPE	**95.80** ± 0.11	**92.44** ± 1.27
	FAST RWPE w/o TE	93.65 ± 0.73	89.70 ± 0.56
Trade	FAST RWPE	**94.01** ± 0.73	**92.06** ± 0.66
	FAST RWPE w/o TE	92.81 ± 0.19	90.94 ± 0.37

We found RWPE is an appropriate PE for our framework.

The choice of PE can have a major impact on results

Time Window



Time Window = the number of snapshots we consider.

CanParl and USLegis spans more than 40 years, considering distant snapshots can have a negative impact on results.

W= 1 is equivalent to only a spatial full attention.

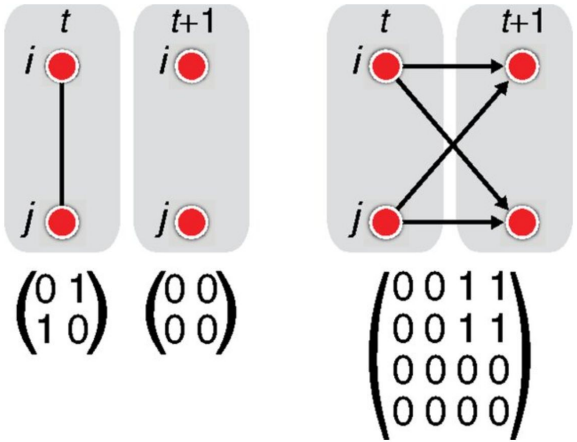
Future works and perspectives

Unified spatio-temporal encoding

- FAST (as existing works for dynamic graph transformers) combines positional encoding and temporal encoding

- **Idea:** Consider the supra-adjacency matrix to compute an encoding that better models the spatio-temporal position of a node.

- Existing works compute metrics and spectral properties on multilayers graphs. *Cozzo, E., et al.. Multilayer networks: metrics and spectral properties.*



Supra adjacency matrix of a DTDG.

Spatio-temporal motif classification

- In static graphs to evaluate expressivity of differents models → Classify structure like cycle, clique, triangle…

- What spatio-temporal motif can we try to detect to evaluate experimentally the expressive power of dynamic graphs model ? (ex: Blinking motifs such as cycles or triangles in graphs.)

Merci !