Temporal Energy Transformer for Long Range Propagation in Continuous Time Dynamic Graphs

Anonymous authors
Paper under double-blind review

Abstract

Representation learning on temporal graphs is crucial for understanding dynamically varying real-world systems such as social media platforms, financial transactions, transportation networks, and communication systems. Existing self-attention based models encounter limitations in capturing long-range dependencies and lack clear theoretical foundations. Energy-based models offer a promising alternative, with a well-established theoretical foundation that avoids reliance on pseudo-losses. However, their application in this domain remains largely unexplored, primarily due to the challenge of designing energy functionals. In this work, we introduce the Temporal Energy Transformer (TET), a novel energy-based architecture that integrates with the Temporal Graph Network (TGN) framework. Our approach centres on a novel energy-based graph propagation module that leverages a specially designed energy functional to capture and preserve spatio-temporal information. This is achieved by modelling the temporal dynamics of irregular data streams with a continuous-time differential equation. Our temporal energy transformer (TET) layer employs a series of temporal energy attention layers and a dense associative memory model or a modern Hopfield network. This design demonstrably minimizes the energy functional that is tailored, enabling efficient retention of historical context while assimilating the incoming data. The efficacy of the model is comprehensively validated across a diverse range of temporal graph datasets, including those with long-range dependencies, demonstrating superior performance in both transductive and inductive scenarios for dynamic link prediction.

1 Introduction

Dynamical systems characterize the temporal evolution of complex phenomena across diverse domains from celestial mechanics to population dynamics (Strogatz, 2024). Traditionally, the modelling of dynamical systems relied on physics-based formulations derived from first principles, such as Newton's laws and conservation equations, where governing equations were explicitly obtained from fundamental physical insights. In modern settings, however, many real-world systems exhibit complex, nonlinear, and high-dimensional behaviours for which the exact dynamics are unknown or analytically intractable. In such cases, the governing function is derived directly from observational or experimental data (Brunton & Kutz, 2019). This data-driven approach faces significant challenges, including nonlinearities, chaotic regimes, transient phenomena, noise, stochastic effects, multi-scale interactions, and inherent uncertainties. Machine learning (ML) and deep learning (DL) methods have emerged as powerful tools to infer these dynamics from data, offering flexibility in capturing intricate patterns. Yet, to enhance interpretability, generalizability, and physical consistency, it is often beneficial to integrate domain knowledge—embedding known physics, symmetries, and constraints—into the learning framework, thereby combining the predictive strengths of data-driven models with the robustness of physics-based reasoning. Among such hybrid approaches, energy-based models have emerged as a path to synergize data-driven learning with principled notions of energy in dynamical systems.

Parallel to these developments, the field of Graph Neural Networks (GNNs) has advanced rapidly, enabling effective representation learning on relational data (Graziani et al., 2024). While majority of existing research pertains to static graphs, many real-world graphs are dynamic in nature. This has motivated the emergence of temporal graph learning methodologies Trivedi et al. (2018); Kumar et al. (2019); Rossi et al. (2020), (Xu et al., 2020; Cong et al., 2023). Despite these advances, effectively managing information propagation across temporal and spatial dimensions remains a persistent challenge (Longa et al., 2023). Recent studies, such as that by Yu et al. (2023), have explored Transformer-based architectures (Vaswani et al., 2017) to better preserve long-range temporal information. However, such approaches often entail quadratic complexity inherent to attention mechanisms and lack strong theoretical foundations. More recently, Gravina et al. (2024) presented deep graph networks specifically tailored for CTDGs, offering theoretical guarantees to mitigate information loss. Nonetheless, despite their theoretical rigor, these models exhibit inconsistent empirical performance across standard temporal graph benchmarks. In this scenario, energy-based models could be seen as an alternative approach to the modelling of temporal graphs.

Concurrently, Associative Memory models, commonly referred to as Hopfield Networks (Hopfield, 1982; 1984), have seen a resurgence due to new theoretical insights into their memory capacity and novel architectural extensions (Chaudhry et al., 2023). Modern variants, termed Dense Associative Memories or Modern Hopfield Networks, leverage sharpened activation functions to amplify their memory storage capacities, achieving super-linear (Krotov & Hopfield, 2016) and even exponential growth (Demircigil et al., 2017). These advances position them as powerful mechanisms for structured information retrieval in machine learning tasks. Additionally, research presented in Demircigil et al. (2017) highlights a compelling connection between these networks and the attention mechanism employed in transformers, identifying the transformer attention as a special instance of Modern Hopfield Networks characterized by a softmax activation function. Despite these developments, the potential of Hopfield-inspired memory mechanisms for temporal graph machine learning remains largely underexplored.

Motivated by the aforementioned challenges and observations, our work proposes a novel architecture rooted in the principles of energy-based models — the **Temporal Energy Transformer (`TET`)**— for temporal graph machine learning. Our framework builds upon the foundational components of the Temporal Graph Network (TGN) (Rossi et al., 2020), incorporating key modules such as memory, message module and embedding module. The primary innovation is a novel graph embedding module designed as a sequence of energy-based attention layers. ,wherein the model parameters are optimized through a newly designed energy functional tailored specifically for temporal graph learning. Such an energy-based formulation provides a principled way to enforce long-term information retention and enhances model interpretability, which are key limitations of previous approaches. Designing an appropriate energy functional is a non-trivial task, giving rise to a fundamental challenge: ***How can energy-based models be extended to effectively propagate information across both temporal and spatial dimensions in dynamically varying graphs?*** To address this, we design the energy functional by appropriately incorporating the information from a current stream of events with the information from the past events retained in the model's so-called 'memory'. The contributions of this paper are outlined as follows:

1. We present the first energy-based model for temporal graph machine learning `TETN` operating on continuous-time dynamic graphs represented as a sequence of events.
2. *Empirical findings:* We demonstrate that `TETN` achieves state-of-the-art performance across multiple temporal graph datasets, consistently outperforming prior methods in both transductive and inductive link prediction settings.

2 Preliminaries

Dynamic graphs. A dynamic graph (also called a temporal graph) is defined as a tuple of the form $\mathcal{G}(t) = \{\mathcal{V}(t), \mathcal{E}(t), \mathbf{X}(t), \mathbf{E}(t)\}, t \geq 0$ where $\mathcal{V}(t)$ is the set of all nodes of the graph and $\mathcal{E}(t)$ is the set of all edges in the graph at time t. The matrices $\mathbf{X}(t)$ and $\mathbf{E}(t)$ are the node and edge feature matrices at time t. The way in which we observe a system of interacting entities (as a dynamic graph) distinguishes the type of dynamic graph as *discrete-time dynamic graph* (DTDG) or *continuous-time dynamic graph* (CTDG).

A DTDG, $\mathcal{G} = \{\mathcal{G}_t | t \in [t_0, t_n]\}$, consists of a sequence of static graphs, known as snapshots, observed at periodic time intervals. Here, $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}, \mathbf{X}_t, \mathcal{E}_t)$ represents the graph observed at a particular timestamp $t = T$ and $\eta_u(T) = \{v : (u, v) \in \mathcal{E}(T)\}$ is the temporal neighborhood of node u and we denote $\eta_u^K(T)$ as the K-hop temporal neighborhood of node u.

In contrast, CTDG is a continuous stream of events observed over time, that is, $\mathcal{G} = \{o_t | t \in [t_0, t_n]\}$, with events occurring at any time (irregular timestamps). Events are of three types: *node-wise* events wherein a node is created; *edge-wise* events wherein a temporal edge is created; *deletion* events when a node/edge is deleted. And, we define the temporal neighbourhood of node u as $\eta_u([0,t]) = \{v : (u,v) \in o_t \ , \ t \in [0,t]\}$.

Problem Definition. Let $\{o_{t_1}, o_{t_2}, \ldots, o_{t_n}\}$ denote a sequence of timestamped events occurring at nodes $u \in \mathcal{V}(t)$ within a continuous-time dynamic graph (CTDG). We seek to develop a representation learning framework with a theoretically grounded embedding module that ensures long-term retention and propagation of event information across the graph. Concretely, for any node u, if an event o_{t_i} occurs at time t_i, the embedding $h_u(t)$ at any subsequent time $t > t_i$ must satisfy: $h_u(t)$ retains information about event o_{t_i}, $\forall \ t > t_i$.

3 Temporal Energy Transformer (TET)

In this work, we propose a **Temporal Energy Transformer (TET)**, building upon the principles of conventional Energy Transformers (Hoover et al., 2024), to address the challenges of temporal graph learning. While existing models for static graph machine learning like ET (Hoover et al., 2024) minimize an energy function to align node representations with their immediate neighbors, our TET model is specifically tailored to incorporate both current and historical graph dynamics into the energy function. This is essential for integration into existing frameworks for temporal graph machine learning, such as Temporal graph Networks (TGN) (Rossi et al., 2020). Existing attention-based models, such as DyGFormer (Yu et al., 2023), often struggle to retain long-term historical information in dynamic graph tasks (Gravina et al., 2024). Our proposed TET model directly addresses this limitation by appropriately fusing past information into the learning process. Building upon the Energy Transformer framework (Hoover et al., 2024), our approach integrates two energy components: Energy attention and Hopfield-inspired energy, which we describe in detail in Section 5. The Energy attention component computes attention coefficients for a node's neighbours, akin to the Graph Attention Network (GAT) framework. The Hopfield energy, whose equivalence to the Transformer architecture (Vaswani et al., 2017) has been demonstrated (Ramsauer et al., 2020), is central to our model's capacity for long-range information retention.

Let us consider a CTDG graph $\mathcal{G}$. For a given node u at time t, the model processes $x_u(t)$, which is the information of node u from the current batch of events (e.g., recent edge additions), and $\eta_u^K([0,t])$, which captures its K-hop temporal neighborhood spanning all past interactions up to time t.

The core of our model's learning process is to minimize the energy functional $E(t)$, defined for a stream of events at time t as:

$$E(t) = \sum_{u=1}^{|\mathcal{V}(t)|} \sum_{v \in \eta_u^K([0,t])} f(x_u(t), s_u(t), e_{uv}(t)). \quad (1)$$

Here, $f(\cdot)$ is a learnable function that quantifies the interaction potential or compatibility between the current representation of node u, $x_u(t)$, its compressed representation in memory, denoted by $s_u(t)$ and the representation of edges $e_{uv}(t)$ in its K-hop temporal neighborhood $\eta_u^K([0,t])$. $s_u(t)$ denotes a concise, aggregated representation of node u's historical interactions. The total energy of the the CTDG system is then $E = \sum_{t<T} E(t)$. As a consequence of the learning, minimizing $E(t) \, \forall \, t$ realizes a gradual decrease in E over time thereby ensuring that the whole system moves towards a low energy configuration, though it may not attain the global minima (or a fixed attractor state).

A salient property of such a model is its superior interpretability, as the learning is guided by the minimization of the function f and consequently the functional E which encapsulates the system's configuration. In contrast to a general loss-based learning framework, which quantifies the discrepancy between predicted and true class probabilities for discrete outcomes; the energy-based minimization ensures that a continuous measure of the inherent dynamics of the system evolves smoothly towards a low-energy configuration (Lecun et al., 2006). To the best of our knowledge, our model is the first to introduce an energy-based approach for temporal graph machine learning.

4 Energy-based framework for Temporal Graph Machine Learning

We describe below, the proposed **TETN** framework, incorporating an energy-based graph propagation module into the TGN framework for CTDGs (Rossi et al., 2020). This enhanced framework is structured around its fundamental components: **(1) the Message module**, **(2) Memory**, and the **(3) Graph Propagation module**. Figure 1 provides a detailed depiction of the complete architecture. *The core contribution of this work lies in the design of our novel energy-based graph propagation module depicted in Figure 2, with detailed training and inference procedures outlined in Algorithm 1 and Algorithm 2 in Appendix A.9. For simplicity, we describe $E(t)$ as E in the subsequent discussions.*

To formally define the operations in our framework, we first establish the notation for this setting. Consider a CTDG graph $\mathcal{G}$ described by a stream of events occurring in the time interval $[0, T]$. Let the nodes in our graph be indexed by $A, B, C, \ldots$ and edges be represented as $AB, BC, \ldots$. We denote the corresponding node embeddings at a time instant t as $x_A(t), x_B(t), \ldots, x_N(t)$ and edge embeddings as $e_{AB}(t), e_{BC}(t), \ldots$. Furthermore, the memory states of these nodes are represented by $s_A(t), s_B(t), s_C(t), \ldots s_N(t)$. Then $\mathcal{G}_t = (V(t), \mathcal{E}(t))$, where $V(t) = \{k : \exists x_k(t_i), t_i \in [0, t]\}, \mathcal{E}(t) = \{(i, j) : \exists e_{ij}(t_i), t_i \in [0, t]\}$. With these established notations, we describe the main modules of the framework below.

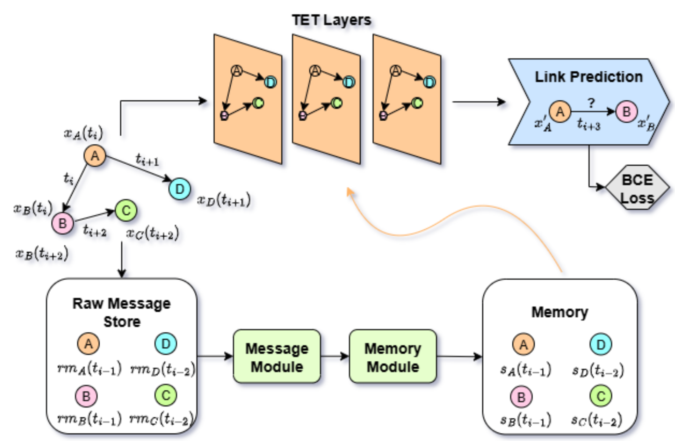

Figure 1: **TETN** pipeline consisting of the message module, memory and the graph propagation module (TET).

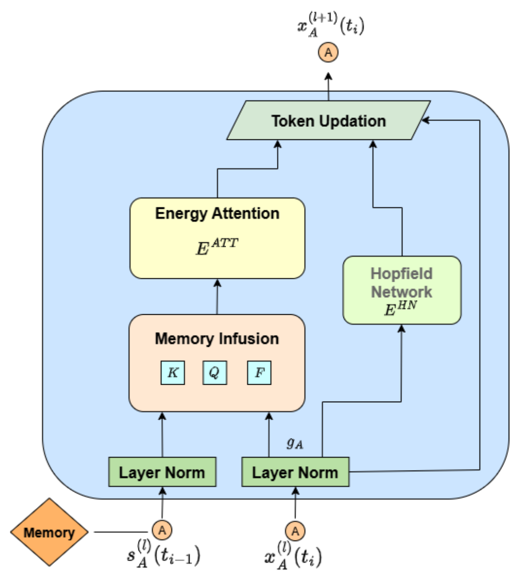

Figure 2: The block diagram represents the **Temporal Energy Transformer (TET)** layer in the architecture of Figure 1, showcasing how the embedding of a node A gets updated.

4.1 Message Module.

The message module is responsible for generating messages that facilitate the update of nodes' memory states within the temporal graph network. Specifically, whenever a new interaction event occurs at time t, messages are computed for both the source node A and the target node B. These messages encapsulate information pertinent to the current interaction and the historical state of the nodes involved. For a source node A and a target node B, their respective messages, $m_A(t)$ and $m_B(t)$, are formulated as follows:

$$m_A(t) = \text{msg}(rm_A(t)), \; m_B(t) = \text{msg}(rm_B(t)), \tag{2}$$

where the raw messages, as stored in the raw message store as $rm_A(t)$ and $rm_B(t)$, are given as:

$$rm_A(t) = s_A(t^-) \,\|\, s_B(t^-) \,\|\, \theta(t - t_A) \,\|\, e_{AB}(t),$$
$$rm_B(t) = s_B(t^-) \,\|\, s_A(t^-) \,\|\, \theta(t - t_B) \,\|\, e_{AB}(t).$$

In these equations, $s_A(t^-)$ and $s_B(t^-)$ represent the memory vectors of nodes A and B, respectively, computed during their last update, prior to the current event. The terms t_A and t_B denote the timestamps of the most recent interactions involving nodes A and B. The function $\theta(\cdot)$ processes the time difference between the current event and the node's last interaction and provides a functional time encoding (Xu et al., 2020). $\mathbf{e}_{AB}(t)$ represents features associated with the edge between A and B at time t. The operator $\|$ signifies vector concatenation. The function $\text{msg}(\cdot)$ is a learnable function. In scenarios where multiple events involving the same node occur within the same batch, the corresponding individual messages are aggregated into a single comprehensive message. This aggregation is performed using an aggregation function, denoted as agg:

$$\overline{m}_w(t) = \text{agg}(m_w(t_1), \ldots, m_w(t_b)),$$

where w refers to the node (either A or B), and $\text{agg}(\cdot)$ can be implemented using various strategies, such as mean, sum, or simply taking the last message.

4.2 Memory.

The memory module serves as a crucial component for maintaining and evolving the historical representations of individual nodes. Following each interaction event, a node's memory state is updated by integrating its prior memory with incoming messages generated by the Message Module. Specifically, given the aggregated incoming message $\bar{m}_A(t)$ for node A at time t and its memory state $s_A(t^-)$ immediately preceding this event, the updated memory representation, $s_A(t)$, is computed as:

$$s_A(t) = \text{mem}(\overline{m}_A(t), s_A(t^-)).$$

The function $\text{mem}(\cdot)$ typically employs an RNN-based architecture, such as a Gated Recurrent Unit (GRU) (Cho et al., 2014).

4.3 Graph Propagation Module.

The graph propagation module generates the temporal embeddings for the nodes involved in the current input batch of events. Once a node's memory is updated with new event information from the memory updater module, its representation is enriched through this module, that involves multiple (`TET`) layers. The input involves the node embeddings in the current batch of CTDG events between times t_i and t_{i+k}, the temporal K-hop neighbourhood ($\mathcal{N}_A^K([0, t_{i+k}])$) of the involved nodes and their compressed representation $s_A(t)$ from the memory module. Note that only the nodes involved in the current batch of input events are updated during a forward pass through the `TET` layer.

In the following section, we describe our graph propagation approach built upon modifications of the energy transformer block of Hoover et al. (2024) (described in Appendix A.10). The general architectural framework of our `TET` block is visually represented in Figure 2. It's core components are described below.

4.3.1 Layer Norm.

Given a node embedding x_A, a layer-normalized representation g_A is obtained using this layer. $g_{iA} = \gamma \dfrac{(x_{iA} - \bar{x}_A)}{\sqrt{\frac{1}{D}(x_{iA} - \bar{x}_A)^2 + \epsilon}}$ where $\bar{x}_A = \sum_{i=1}^{D} x_{iA}$.

4.3.2 Memory Infusion.

The layer-normalized embeddings g_A of the nodes and their corresponding memory states s_A from the memory module are used to compute the key, query and edge tensors in `TETN`. They are operated upon by summation or concatenation like in Eqn. (6).

4.3.3 Energy Attention.

The energy attention E^{ATT} is designed by modifying the conventional transformer-based attention mechanism, and is used to align the representation of a particular node with respect to the information of its neighbours. The memory-infused keys and queries (and/or edge tensors) for each node are used here to evaluate the attention. The model leverages memory-infused tensors for cross-temporal attention, thereby aligning node representations across time. The exact formulation for the attention based energy is provided in Eqn. (8).

4.3.4 Hopfield Network.

The Hopfield Network (HN) aligns the node representations to be consistent with the information learnt from the evolving graph across batches. This is in contrast to the HN block in ET which learns the patterns over the entire static graph. In `TETN`, the Hopfield energy is defined similar to as the one in `ET`:

$$E^{HN} = -\sum_{B=1}^{N}\sum_{\mu=1}^{K} G\left(\sum_{j} \xi_{\mu j} g_{jB}\right), \xi \in \mathbf{R}^{K\times D}, \tag{3}$$

where, $\xi_{\mu j}$ is a set of learnable weights, also called memories in HN, and $G(.)$ is an integral of the activation function $r(.)$. The activation function that we use is $r(.) = ReLU$.

4.3.5 Token Updation.

The inference pass of the `TET` block is designed as an ODE update:

$$\tau \frac{dx_{iA}}{dt} = \mathcal{F}\left(\frac{\partial E}{\partial g}, \frac{\partial E}{\partial s}, \frac{\partial E}{\partial e}\right), \tag{4}$$

where, $E = E^{ATT} + E^{HN}$. Here, E is total energy, E^{ATT} is the attention based energy term and E^{HN} is the contribution to energy from Hopfield network. The exact formulation of the update function $\mathcal{F}$ for `TETN` is described in Eqn. (10).

5 Energy in Temporal Energy Transformer (`TET`)

In this section, we describe in detail the energy formulation in Temporal Energy Transformer (`TET`). The energy formulation used in ET (Hoover et al., 2024) does not explicitly use the edge features and is limited to learning from static graphs. Our `TETN` model, designed for temporal learning, involves a modified energy attention with cross-temporal attention terms to align the node representations across the temporal neighbourhood. Moreover, the edge features are added to the computation of the key tensor with an additional learnable matrix $W^F \in \mathbf{R}^{Y\times H\times D}$. Further to this, the memory infusion module involves the consideration of the memory states of the nodes $s_A(t)$ in the computation of the new key, query and edge tensors as:

$$K_{\alpha hB} = \sum_j W^K_{\alpha hj}(g_{jB} + s_{jB}), K \in \mathbf{R}^{Y\times H\times N}, \tag{5}$$

$$Q_{\alpha hC} = \sum_j W^Q_{\alpha hj}(g_{jC} + s_{jC}), Q \in \mathbf{R}^{Y\times H\times N}, \tag{6}$$

$$F_{\alpha hBC} = \sum_j W^F_{\alpha hj} e_{jBC}, F \in \mathbf{R}^{Y\times H\times N}. \tag{7}$$

Incorporating the above modifications into the classical energy attention equation of ET (Hoover et al., 2024), we have the modified energy attention equation given as:

$$E^{ATT} = -\frac{1}{\beta}\sum_{h=1}^{H}\sum_{C=1}^{N} log\left(\sum_{B\in\mathcal{N}_C} \exp A_{hBC}\right), \tag{8}$$

where, $A_{hBC} = \beta \sum_{\alpha}(K_{\alpha hB} + F_{\alpha hBC})Q_{\alpha hC}$. Combining E^{ATT} with the Hopfield Energy from Eqn. (3), we have the total energy as:

$$E = E^{ATT} + E^{HN} \tag{9}$$

While in the ET, the energy function depends solely on the node embeddings g of the current input stream of events, the proposed energy functional in **TETN** depend not only on the current node embeddings g, but also their corresponding memory states s and the features e of the involved edges. Consequently, the token update for each forward pass must take into consideration the variations of energy E with respect to each of these variables involved. Specifically, the ODE-based update for x is guided by the gradient $\frac{\partial E}{\partial g_A}$, ensuring descent along the energy landscape towards a minimum. While the descent direction is definite, we drive the descent by defining an adaptive step length which incorporates a linear combination of the energy variation with respect to s and e as given below. This integration preserves the core principle that token updates should reduce energy (see proof of Theorem 1 in Appendix A.1), while simultaneously supporting stable and consistent learning dynamics. Thus, the token update, following Eqn. (4), is defined as:

$$\tau \frac{dx_{iA}}{dt} = -\frac{\partial E}{\partial g_{iA}} \left(m + \gamma_1 \left\| \frac{\partial E}{\partial s_{iA}} \right\| + \gamma_2 \sum_{C \in \mathcal{N}_A} \left\| \frac{\partial E}{\partial e_{iCA}} \right\| \right). \tag{10}$$

where γ_1 and γ_2 are hyperparameters and m is a margin whose default value is 0 and it takes the value 1, if $\gamma_1, \gamma_2 = 0$. The token update process defined by Eqn. (10) integrates three terms: $\frac{\partial E}{\partial g_{iA}}$ incorporates information from the **current batch**, $\frac{\partial E}{\partial s_{iA}}$ leverages data from **memory**, and $\frac{\partial E}{\partial e_{iCA}}$ captures **edge-specific information, as explained above**. Since we update only the current node embeddings in the **TETN** layers, we do not consider layer-wise update equations for s or e. Here, the energy gradients (assuming only one head of attention) are given as:

$$-\frac{\partial E^{ATT}}{\partial g_{iA}} = -\frac{\partial E^{ATT}}{\partial s_{iA}} = \sum_{C \in \mathcal{N}_A} \sum_{\alpha} W^Q_{\alpha i}(K_{\alpha C} + F_{\alpha AC})\omega_{CA} + W^K_{\alpha i} Q_{\alpha C}\omega_{AC}$$

$$-\frac{\partial E^{ATT}}{\partial e_{iCA}} = \sum_{\alpha} W^F_{\alpha i} Q_{\alpha A}\, \omega_{CA}$$

where,

$$\omega_{CA} = \operatorname*{softmax}_{C} \left(\beta \sum_{\gamma} (K_{\gamma C} + F_{\gamma AC}) Q_{\gamma A} \right) \text{and}\, \omega_{AC} = \operatorname*{softmax}_{A} \left(\beta \sum_{\gamma} (K_{\gamma A} + F_{\gamma AA}) Q_{\gamma C} \right).$$

6 Experiments

We assess our model's capability on temporal representation learning by evaluating it on the CTDG benchmark datasets. We explain the experimental setup in more detail in Section 6.1 and discuss the results in Section 6.2. Additionally, in Appendix A.1, we demonstrate the provable, monotonic decrease of our energy functional over time. We present brief descriptions and statistics of the datasets in Appendix A.7.

6.1 Experimental Setup

Datasets: We test the performance of **TETN** on CTDG learning task by comparing it with other baselines methods on dynamic link prediction task on CTDG datasets. We consider six real world datasets collected by Poursafaei et al. (2022), namely, Wikipedia, UCI, Reddit, MOOC, Enron, LastFM (see Table 10). These datasets are very widely used in some recent papers like Ding et al. (2024) and Yu et al. (2023). Out of these datasets MOOC, Enron and LastFM are considered long range temporal dependent datasets as established by Yu et al. (2023).

Baselines: We compare **TETN** with eight recent CTDG models, viz., DyRep (Trivedi et al., 2018), JODIE (Kumar et al., 2019), TGAT (Xu et al., 2020), TGN (Rossi et al., 2020), GraphMixer (Cong et al.,

Table 1: Results of the future link prediction task - transductive setting. We report the mean test set average precision (AP) and standard deviation in percent averaged over random weight initializations. Methods are ranked by average rank across all datasets (lower is better).

Dataset → / Method ↓	Wikipedia	UCI	Reddit	MOOC	Enron	LastFM	Avg. Rank
TETN (ours)	**99.20**$_{\pm 0.05}$	**98.98**$_{\pm 0.03}$	**99.42**$_{\pm 0.69}$	85.07$_{\pm 1.10}$	**92.67**$_{\pm 0.04}$	**96.68**$_{\pm 0.14}$	**1.67**
DyGMamba	99.15$_{\pm 0.02}$	95.91$_{\pm 0.15}$	99.32$_{\pm 0.01}$	**89.21**$_{\pm 0.08}$	92.65$_{\pm 0.12}$	93.35$_{\pm 0.20}$	1.83
DyGFormer	99.03$_{\pm 0.03}$	95.74$_{\pm 0.17}$	99.22$_{\pm 0.01}$	87.66$_{\pm 0.48}$	92.42$_{\pm 0.11}$	92.95$_{\pm 0.14}$	3.00
TGN	98.45$_{\pm 0.01}$	92.33$_{\pm 0.64}$	98.65$_{\pm 0.04}$	89.15$_{\pm 1.69}$	86.98$_{\pm 1.05}$	75.31$_{\pm 5.62}$	4.33
GraphMixer	97.22$_{\pm 0.02}$	93.15$_{\pm 0.41}$	97.31$_{\pm 0.01}$	82.80$_{\pm 0.15}$	82.13$_{\pm 0.30}$	75.56$_{\pm 0.19}$	5.50
CTAN	96.61$_{\pm 0.79}$	76.64$_{\pm 4.11}$	97.21$_{\pm 0.84}$	84.71$_{\pm 2.85}$	92.52$_{\pm 1.20}$	86.44$_{\pm 0.80}$	6.17
TGAT	96.88$_{\pm 0.06}$	79.40$_{\pm 0.61}$	98.57$_{\pm 0.01}$	85.71$_{\pm 0.20}$	70.76$_{\pm 1.05}$	73.30$_{\pm 0.18}$	6.50
JODIE	96.51$_{\pm 0.22}$	89.28$_{\pm 1.02}$	98.31$_{\pm 0.06}$	81.04$_{\pm 0.83}$	84.85$_{\pm 3.13}$	70.95$_{\pm 2.94}$	7.83
DyRep	94.88$_{\pm 0.29}$	66.11$_{\pm 2.75}$	98.18$_{\pm 0.03}$	81.50$_{\pm 0.77}$	79.80$_{\pm 2.28}$	71.85$_{\pm 2.44}$	8.67

Table 2: Results of the future link prediction task - inductive setting. We report the mean test set average precision (AP) and standard deviation in percent averaged over random weight initializations. Methods are ranked by average rank across all datasets (lower is better).

Dataset → / Method ↓	Wikipedia	UCI	Reddit	MOOC	Enron	LastFM	Avg. Rank
DyGMamba	98.77$_{\pm 0.03}$	94.76$_{\pm 0.19}$	98.97$_{\pm 0.01}$	**88.64**$_{\pm 0.08}$	**89.67**$_{\pm 0.27}$	94.42$_{\pm 0.21}$	**1.67**
TETN (ours)	**98.83**$_{\pm 0.24}$	**95.44**$_{\pm 0.15}$	**99.01**$_{\pm 0.24}$	80.87$_{\pm 0.14}$	89.65$_{\pm 0.22}$	**97.69**$_{\pm 0.21}$	2.00
DyGFormer	98.58$_{\pm 0.01}$	94.45$_{\pm 0.13}$	98.83$_{\pm 0.02}$	87.05$_{\pm 0.51}$	89.62$_{\pm 0.27}$	94.17$_{\pm 0.10}$	3.00
TGN	97.81$_{\pm 0.18}$	87.81$_{\pm 1.32}$	97.41$_{\pm 0.12}$	88.01$_{\pm 1.48}$	78.76$_{\pm 1.69}$	81.18$_{\pm 3.27}$	4.67
GraphMixer	96.61$_{\pm 0.04}$	91.17$_{\pm 0.29}$	95.24$_{\pm 0.08}$	81.38$_{\pm 0.17}$	75.55$_{\pm 0.81}$	82.07$_{\pm 0.31}$	6.00
JODIE	94.91$_{\pm 0.32}$	79.73$_{\pm 1.48}$	96.43$_{\pm 0.16}$	80.57$_{\pm 0.52}$	78.97$_{\pm 1.59}$	83.13$_{\pm 1.19}$	6.00
TGAT	96.26$_{\pm 0.12}$	79.10$_{\pm 0.49}$	97.13$_{\pm 0.04}$	85.28$_{\pm 0.30}$	66.67$_{\pm 1.07}$	78.40$_{\pm 0.30}$	6.50
DyRep	92.21$_{\pm 0.29}$	58.39$_{\pm 2.38}$	95.89$_{\pm 0.26}$	80.50$_{\pm 0.68}$	73.97$_{\pm 3.00}$	83.47$_{\pm 1.06}$	7.33
CTAN	93.58$_{\pm 0.65}$	49.78$_{\pm 5.02}$	80.07$_{\pm 2.53}$	64.99$_{\pm 2.24}$	74.61$_{\pm 1.64}$	60.40$_{\pm 3.01}$	8.50

2023), CTAN (Gravina et al., 2024), DyGFormer (Yu et al., 2023) and DyGMamba (Ding et al., 2024). Among these models, CTAN, DyGFormer and DyGMamba are designed to capture long-range temporal information propagation and we compare them with TETN and highlight the advantage of using energy based models.

Implementation Details: We follow the experimental protocol established by Ding et al. (2024) to ensure fair comparison with existing baselines. Our evaluation employs the same datasets, metrics (average precision (AP) and area under the Receiver Operating Characteristic curve (AUC-ROC)), and experimental configurations as the comparative methods, maintaining consistency across all benchmarks to eliminate potential evaluation bias. The code for model and experiments are present at `https://anonymous.4open.science/r/tetn_log-C119`.

Evaluation: We employ two evaluation settings following the previous works: the transductive and inductive settings. We use a random negative sampling strategy for inductive and transductive setting. Model performance is evaluated using the metrics, AP and AUC-ROC. All experiments are implemented in PyTorch and on a server with NVIDIA A30 GPUs with 24GB of RAM. To ensure statistical reliability, all experiments were conducted across five independent runs with different random initializations. Tables 1, 2, 3 and 4 report the mean performance along with standard deviations. We also do hyper-parameter tuning for TETN and the details are mentioned in Appendix A.8.

6.2 Performance of TETN

We present the future link prediction results for transductive and inductive settings in Tables 1 and 2, respectively. Across both these evaluations, TETN demonstrates consistently strong performance, establishing itself as a robust and generalizable approach for future link prediction in dynamic graphs. In the **transduc-**

tive setting, `TETN` attains the best performance on majority of the datasets, achieving an average rank of **1.67**. It secures top performance on UCI (98.98% AP) and LastFM (96.68% AP) datasets showcasing an improvement in the metric with high margins. And it maintains highly competitive results on remaining datasets with performance improvement within a narrow margin of the leading methods. The performance on the long-range datasets (Yu et al., 2023), particularly *Enron* and *LastFM*, further pronounces `TETN`'s robustness. *LastFM* is a dense spanning over a month with around 1.3 million events and *Enron* is a dataset spanning over 3 years with over $125k$ events (see Table 10 in Appendix A.7). The ability of `TETN` to maintain stable and expressive temporal embeddings under such diverse temporal settings indicates that its temporal encoding strategy preserves long-range temporal structure more effectively than attention-based or memory-based baselines. Importantly, `TETN` avoids common pitfalls such as non-dissipativeness over time and space (Gravina et al., 2024), which degrades performance in continuous-time architectures. This robustness stems from `TETN`'s energy-based design.

In the **inductive setting**, where the model must generalize to previously unseen nodes, `TETN` achieves an average rank of **2.0**, closely following DyGMamba while outperforming transformer-based, recurrent, and structural baselines by a comfortable margin. Notably, `TETN` secures the best performance on datasets such as *Wikipedia*, *UCI*, *Reddit*, and *LastFM*, demonstrating that its learned temporal representations remain transferable even when the structural context shifts at test time. This capability is crucial for real-world dynamic graph applications—including recommender systems, communication platforms, and evolving knowledge graphs—where new entities frequently appear.

Performance on irregular datasets such as *MOOC* and *Enron* shows a noticeable sensitivity in the inductive setting. On *Enron*, the scores remain close to those of DyGFormer and DyGMamba, whereas on *MOOC* the gap widens. Unlike LastFM, which exhibits fine-grained and regular temporal evolution, Enron and MOOC suffer from limited temporal resolution, with many interactions collapsing onto the same timestamps, which might be diminishing the performance on these datasets.

Further to the AP results, we report the AUC-ROC results on the CTDG datasets in Tables 3 (transductive) and 4 (inductive) in Appendix A.2. We observe that the performance of the `TETN` model is in alignment with it's performance based on the AP metric.

Additionally, we conducted auxiliary experiments on the tgbl-review dataset, which has nearly 5 million events and is long-range in both time and space. We refer the reader to Table 5 in Appendix A.3 for the results. `TETN` achieved an MRR (Mean Reciprocal Rank) of 0.375, outperforming DyGFormer (0.224 MRR) and TGN (0.349 MRR). This represents a substantial relative improvement and further highlights the superior long-range temporal propagation capability of `TETN`.

We also provide the results for ablation of various module of our architecture in Tables 6 and 7 in Appendix A.4. The results demonstrate that each component, viz. Energy Attention, HN and Edge-features, contributes significantly to the observed performance of our model. Further to this, Table 8 in A.5 reports the model's performance under different choices of the hyperparameters γ_1 and γ_2 in Eq. 10. This illustrates the significance of the various energy gradient terms in Eqn. 10. Further details and discussion are provided in the respective appendix sections.

7 Discussion

The performance of `TETN` across both transductive and inductive settings highlight an important strength: **`TETN` does not overfit to the observed graph structure**. Instead, it learns temporal representations that encode both short-range interaction signals and long-horizon dynamics in a manner that generalizes across graph topologies. This balance is notably difficult to achieve, as many existing temporal graph neural networks rely heavily on localized structural patterns or explicit memory modules that fail to transfer well to unseen nodes. `TETN` achieves top performance on UCI (both settings), Wikipedia (both settings), Reddit (both settings), LastFM (both settings) and Enron (transductive) with only a minor decrease in the performance metrics for Enron (inductive). These results confirm that our energy-based formulation excels at the long-range propagation problem that motivates this work. That an energy-based approach ranks as the top performing method (or a close second rank) among 9 established methods, including several designed explicitly for long-range modeling (like CTAN and DyGMamba), represents a significant advance in establishing a new research direction.

`TETN`'s exceptional performance stems from its energy-based architecture for temporal graph learning, demonstrating that this previously unexplored paradigm can compete with highly optimized Transformer or SSM-based methods (viz. DyGFormer, DyGMamba), while providing theoretical guarantees (Theorem 1 on monotonic energy decrease). The model compresses each node's historical information into compact representations through the energy functional, enabling effective modeling of temporal evolution without computational bottlenecks. Further, the learning proceeds through minimization of the energy functional, which depends explicitly on node representations rather than a loss based on predictions alone. We consider this energy-based minimization as a more effective approach since the energy values themselves are an indication of the system's state at each update, making it more interpretable. This decomposition allows practitioners to interpret low-energy configurations as states where nodes are temporally and structurally consistent with their neighborhoods, beyond mere prediction accuracy. Moreover, the energy plots in Appendix A.1 demonstrate that our formulated energy functional exhibits smooth convergence behaviour throughout training on the Wikipedia dataset, ensuring stable and optimal temporal node embedding updates

The energy-based approach proves particularly effective on challenging datasets like LastFM and tgbl-review, with extensive edge counts, where conventional methods struggle with scalability and long-term memory requirements. We attribute the performance gain of `TETN` to three major building blocks of it's architecture: (i) the ODE-based update rules, (ii) the dense associative memory inherited from Hopfield-style dynamics, and (iii) the energy-based attention. By embedding past information directly into the energy functional (memory infusion), the model achieves more reliable long-range propagation than purely attention-driven mechanisms. Compared to contemporary architectures, `TETN` achieves these improvements without introducing additional features (as in DyGMamba or DyGFormer) or stabilizing terms (as in CTAN). Moreover, `TETN` can handle concurrent edges, while DyGMamba can handle edges (or events) only sequentially. We acknowledge limitations in performance on the MOOC dataset. Concurrently, we emphasize that opening this methodological paradigm, achieving state-of-the-art results on long-range tasks, and providing theoretical foundations absent in competitors constitute substantial contributions beyond point-wise empirical comparisons.

Overall, the results demonstrate that **`TETN` achieves a rare combination of accuracy, training stability, and generalization**. Its consistently strong performance across datasets of varying scale, sparsity, and temporal characteristics underscores its suitability as a general-purpose framework for temporal graph learning. These findings position `TETN` as a compelling model for advancing dynamic graph representation learning, particularly for AI applications requiring reliable forecasting over evolving relational structures.

8 Related Work

Long range Propagation for Temporal Graph Machine Learning: Despite recent efforts, long-range information retention in temporal graph machine learning remains a significant challenge. The Continuous-time Graph Anti-symmetric Network (CTAN) (Gravina et al., 2024) pioneered an ODE-based approach that captures long-term temporal dependencies through non-dissipative information propagation across time and space. Building on prior work that incorporated asymmetric terms to mitigate over-smoothing in static graphs (Gravina et al., 2022), CTAN demonstrates superior performance over existing state-of-the-art methods including TGN (Rossi et al., 2020) and DyGFormer (Yu et al., 2023). However, CTAN shows limited improvements on certain temporal datasets and lacks interpretability due to its reliance on self-attention mechanisms. Existing approaches for dynamic graph modeling, such as DyGMamba (Ding et al., 2024) (based on SSMs) and DyGFormer (Yu et al., 2023) (based on Transformers), effectively model long-range dependencies, but are often constrained by quadratic computational complexity at high patch sizes and lack interpretability. To overcome these limitations, we introduce a novel energy-based method. Our approach combines an **Energy attention block** and a **Hopfield Network** (Section 4.3), a fusion that offers a compelling alternative. Specifically, we leverage the Hopfield Network for its proven equivalence to the self-attention mechanism used in DyGFormer, as demonstrated in this work (Ramsauer et al., 2020). Critically, unlike the black-box nature of many Transformer-based models, our network's parameters are optimized using an interpretable energy function, thereby enhancing model transparency and providing a path toward more explainable dynamic graph models.

Temporal Graph Machine Learning: The challenge of modelling evolving network structures has led to diverse methodological innovations in temporal graph analysis. Contemporary approaches utilize different architectural principles and temporal modeling strategies. Recurrent-based methods like JODIE (Kumar et al., 2019) employ sequential neural architectures to maintain evolving node representations through interaction history processing. GraphMixer (Cong et al., 2023) introduces a streamlined architecture with dedicated encoding components for nodes and edges, integrated with MLP-Mixer modules for feature synthesis. Hybrid approaches address both spatial and temporal complexities by integrating multiple modeling paradigms. TGAT (Xu et al., 2020) incorporates temporal encoding directly into node feature representations, while TGN (Rossi et al., 2020) combines recurrent processing with attention-based spatial modeling for comprehensive spatio-temporal learning. Our framework builds upon several core modules from TGN, with a key distinction: our graph propagation module incorporates an energy-based model for enhanced interpretability and performance.

Energy-based methods for deep learning: Energy-based models (EBMs) provide a unified framework for learning by associating scalar energy values to variable configurations, where inference involves finding minimum energy states without probabilistic normalization (Lecun et al., 2006). Recent advances have leveraged EBMs for neural reasoning by parameterizing energy landscapes over output spaces, enabling adaptive computational allocation where harder problems receive more optimization steps (Du et al., 2022).

The Energy Transformer (Hoover et al., 2024) minimize a global energy function E, with minima corresponding to fixed attractor states. While promising for static graphs, adapting energy functionals to temporal graphs and addressing long-range propagation challenges remain non-trivial. Our energy-based framework models interacting entities as a dynamical system of ODEs driven by an energy function that represents entity relationships, with local minima corresponding to fixed-point attractor states.

Oversquashing in static and dynamic graphs: To mitigate the oversquashing problem, wherein deep graph neural networks struggle to capture long-range dependencies, several ODE-based methods have been proposed for static graphs (Chamberlain et al., 2021; Eliasof et al., 2021). Inspired by these developments, our **TETN** token update rule is also formulated as an ODE. For discrete-time dynamic graphs (DTDGs), existing approaches such as Pareja et al. (2020) model graph evolution through recurrent architectures, while Ceni et al. (2025) employs graph state-space models to address information propagation loss. By contrast, our energy-based formulation naturally generalises to continuous-time dynamic graphs (CTDGs), which are a realistic representation of real-world networks.

9 Conclusion

In this work, we present Temporal Energy Transformer Network (**TETN**), as a first attempt of introducing energy-based optimization principles to continuous-time dynamic graph learning. Our framework fundamentally reconceptualizes temporal representation learning by formulating node evolution as an energy minimization problem, providing theoretical grounding for capturing complex temporal dynamics in networked systems. The proposed energy functional establishes a unified mathematical framework that naturally balances temporal continuity with structural coherence, addressing key limitations in existing temporal graph neural networks. Beyond demonstrating strong empirical performance, this work establishes energy-based methods as a viable paradigm for dynamic graph analysis. The theoretical foundation provided by our energy formulation opens avenues for incorporating first principles into graph learning, potentially enabling more interpretable and theoretically valid approaches to modeling real-world dynamic systems. Moreover, as this work establishes the framework to use energy-based models for temporal graph learning, further improved designs of energy functionals while retaining the current framework could be potentially explored in future. We believe our work represents a significant step toward bridging physics-inspired optimization with modern graph neural network architectures.

References

Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.

Andrea Ceni, Alessio Gravina, Claudio Gallicchio, Davide Bacciu, Carola-Bibiane Schonlieb, and Moshe Eliasof. Message-passing state-space models: Improving graph learning with modern sequence modeling. *arXiv preprint arXiv:2505.18728*, 2025.

Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In *International conference on machine learning*, pp. 1407–1418. PMLR, 2021.

Hamza Chaudhry, Jacob Zavatone-Veth, Dmitry Krotov, and Cengiz Pehlevan. Long sequence hopfield memory. *Advances in Neural Information Processing Systems*, 36:54300–54340, 2023.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? *arXiv preprint arXiv:2302.11636*, 2023.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017.

Zifeng Ding, Yifeng Li, Yuan He, Antonio Norelli, Jingcheng Wu, Volker Tresp, Michael Bronstein, and Yunpu Ma. Dygmamba: Efficiently modeling long-term temporal dependency on continuous-time dynamic graphs with state space models. *arXiv preprint arXiv:2408.04713*, 2024.

Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Learning iterative reasoning through energy minimization. In *International Conference on Machine Learning*, pp. 5570–5582. PMLR, 2022.

Moshe Eliasof, Eldad Haber, and Eran Treister. Pde-gcn: Novel architectures for graph neural networks motivated by partial differential equations. *Advances in neural information processing systems*, 34:3836–3849, 2021.

Alessio Gravina, Davide Bacciu, and Claudio Gallicchio. Anti-symmetric dgn: a stable architecture for deep graph networks. *arXiv preprint arXiv:2210.09789*, 2022.

Alessio Gravina, Giulio Lovisotto, Claudio Gallicchio, Davide Bacciu, and Claas Grohnfeldt. Long range propagation on continuous-time dynamic graphs. *arXiv preprint arXiv:2406.02740*, 2024.

Caterina Graziani, Tamara Drucks, Fabian Jogl, Monica Bianchini, Franco Scarselli, and Thomas Gärtner. The expressive power of path-based graph neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Benjamin Hoover, Yuchen Liang, Bao Pham, Rameswar Panda, Hendrik Strobelt, Duen Horng Chau, Mohammed Zaki, and Dmitry Krotov. Energy transformer. *Advances in Neural Information Processing Systems*, 36, 2024.

J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. doi: 10.1073/pnas.79.8.2554.

J J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10):3088–3092, 1984. doi: 10.1073/pnas.81.10.3088.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *Advances in Neural Information Processing Systems*, 36:2056–2073, 2023.

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1269–1278, 2019.

Yann Lecun, Sumit Chopra, and Raia Hadsell. *A tutorial on energy-based learning.* MIT Press, 01 2006.

Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Lio, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *arXiv preprint arXiv:2302.01018*, 2023.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5363–5370, 2020.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 32928–32941. Curran Associates, Inc., 2022.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering.* Chapman and Hall/CRC, New York, 3rd edition, 2024. ISBN 9780429398490. doi: 10.1201/9780429398490.

Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Representation learning over dynamic graphs. *arXiv preprint arXiv:1803.04051*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *CoRR*, abs/2002.07962, 2020.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–67700, 2023.

A Appendix

A.1 Analysis of Energy

One of the major considerations while designing an energy function whose gradient controls the dynamics of the neural ODE is that the energy function must be a monotonically decreasing function.

Theorem 1. *The energy functional E in Eqn. (9) of* **TETN** *is a decreasing function under the dynamics defined by the ODE-update equation Eqn. (10), for $\beta, \gamma_1, \gamma_2 \in \mathbb{R}^+$ and $m = 0$.*

Proof. Consider energy E of `TETN`. We have $E(t) = E(g_A, s_A, e_{AC} | A \in \mathcal{V}(t) \, \text{and} \, (A, C) \in \mathcal{E}(t))$ and the update from one layer to the next is given by $\frac{dE}{dt}$ as:

$$\frac{dE}{dt} = \sum_{i,j,A} \frac{\partial E}{\partial g_{iA}} \frac{\partial g_{iA}}{\partial x_{jA}} \frac{dx_{jA}}{dt} + \sum_{i,A} \frac{\partial E}{\partial s_{iA}} \frac{ds_{iA}}{dt} + \sum_{i,A,C} \frac{\partial E}{\partial e_{iAC}} \frac{de_{iAC}}{dt}. \tag{11}$$

Since, for a given pass through the `TETN` layer, the ODE system only updates the node features, while the memory states and the edge features remain unchanged, we get $\frac{ds_{iA}}{dt} = 0$ and $\frac{de_{iAC}}{dt} = 0$. Additionally, we have,

$$\mathbf{M}^A_{ij} = \frac{\partial g_{iA}}{x_{jA}} = \frac{\partial^2 L}{\partial x_{iA} \partial x_{jA}} \, ,$$

where L is the Lagrangian given as

$$L = D\gamma \sqrt{\frac{1}{D}(x_{iA} - \bar{x}_A)^2 + \epsilon} + \sum_j \delta_j x_j .$$

So, $\mathbf{M}$ is a positive semi-definite matrix. Hence, using Eqn. (10) with $m = 0$, we get,

$$\frac{dE}{dt} = - \sum_{i,j,A} \left(\frac{\partial E}{\partial g_{iA}} \mathbf{M}^A_{ij} \frac{\partial E}{\partial g_{jA}} P_{jA} \right) \leq 0 \, , \tag{12}$$

since $P_{jA} = \left(\left\| \frac{\partial E}{\partial s_{jA}} \right\| + \sum_{C \in \mathcal{N}_A} \left\| \frac{\partial E}{\partial e_{jCA}} \right\| \right) \geq 0$. □

Remark: For $m = 1$ and $\gamma_1, \gamma_2 = 0$, Eqn. 10 reduces to

$$\tau \frac{dx_{iA}}{dt} = - \frac{\partial E}{\partial g_{iA}}$$

and E can be proved to be a decreasing functional (proof similar as above).

Our experiments empirically demonstrate that energy is a decreasing function across the `TETN` layers. As shown in Figures 3 and 4, the layer-wise energy decreases as training progresses. Figure 3 plots the average energy per layer across training epochs. Initially, the energy values decrease and enter a stable, low-energy region (indicated by the colored area in the figures). As the training continues, the model begins to overfit the training data, causing the energy values to increase, a trend visible in Figure 3 after epoch 125. For a more fine-grained analysis, Figure 4 displays the energy at every gradient update (i.e., for each batch). This plot clearly shows that as the model processes more batches, the loss decreases, and the model's energy also decreases, demonstrating a synergistic relationship between learning and energy. We note that even with a dynamically evolving dataset, our energy-based model directs the system toward a basin of attraction and a local minimum, rather than continuously decreasing. This supports our hypothesis that the model learns a few global patterns in a dynamic dataset, allowing it to predict accurately even when some data properties change.

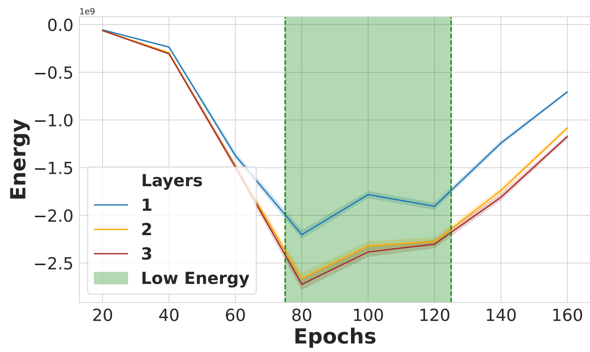

Figure 3: Energy over epochs across **TETN** layers

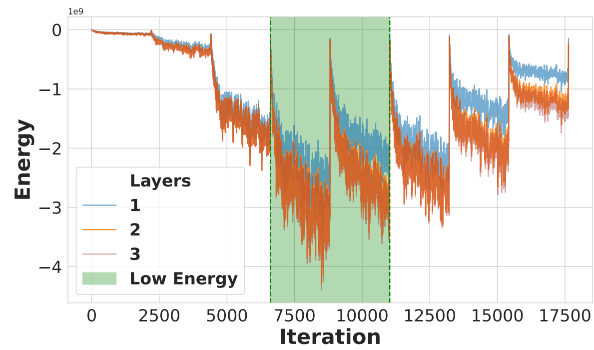

Figure 4: Energy over iterations across **TETN** layers.

We further explain the behavior of the energy curves in Figure 4. The oscillatory behavior in the energy curves during later iterations can be attributed to the non-i.i.d. nature of our training batches. In dynamic graph learning pipelines, batches are typically constructed by sorting edges chronologically based on their addition timestamps. This temporal ordering means that consecutive batches can exhibit significantly different graph structures and patterns. As the energy function parameters are updated batch-wise, each update optimizes for the current batch's characteristics, which may slightly deviate from the optimal parameters for earlier temporal segments of the graph. Consequently, at the beginning of each new epoch, when the model re-encounters the initial batches, the energy values start relatively high due to this parameter misalignment. However, as training progresses through the epoch and the model processes a few batches, the parameters quickly realign to better fit the early temporal patterns, causing the energy to decrease rapidly. This creates the oscillatory pattern observed in Figure 4. It is worth noting that Figure 3 demonstrates the overall decreasing trend in energy across epochs, capturing the global convergence behavior of our model. We emphasize that the local oscillations in Figure 4, while present, do not detract from the fundamental learning progress of the system.

A.2 AUC-ROC Results

We report the Area under the Receiver Operating Characteristic curve (AUC-ROC) values for the experiments on CTDG datasets in Tables 3 and 4. These results are consistent with the results in Tables 1 and 2.

Table 3: Results of the future link prediction task - transductive setting. We report the mean test set area under the ROC curve (AUC-ROC) and standard deviation in percent averaged over random weight initializations. Methods are ranked by average rank across all datasets (lower is better).

Dataset → / Method ↓	Wikipedia	UCI	Reddit	MOOC	Enron	LastFM	Avg. Rank
TETN (ours)	**99.15** ± 0.13	**98.95** ± 0.10	**99.32** ± 0.02	84.70 ± 0.40	**93.56** ± 1.05	**96.28** ± 0.57	**1.83**
DyGMamba	99.08 ± 0.02	94.77 ± 0.18	99.27 ± 0.01	89.58 ± 0.12	93.34 ± 0.23	93.31 ± 0.18	**2.00**
DyGFormer	98.92 ± 0.03	94.45 ± 0.22	99.15 ± 0.01	88.08 ± 0.50	93.20 ± 0.12	93.03 ± 0.11	**3.00**
TGN	98.37 ± 0.10	92.03 ± 0.69	98.61 ± 0.05	**91.91** ± 0.82	88.72 ± 0.95	76.64 ± 4.66	**3.67**
CTAN	97.00 ± 0.21	76.25 ± 2.83	97.24 ± 0.75	85.40 ± 2.67	87.09 ± 1.51	85.12 ± 0.77	6.00
TGAT	96.60 ± 0.07	78.76 ± 1.10	98.50 ± 0.01	87.01 ± 0.16	68.57 ± 1.46	71.47 ± 0.14	6.50
JODIE	96.36 ± 0.14	90.35 ± 0.51	98.29 ± 0.05	84.50 ± 0.87	87.77 ± 2.43	70.89 ± 1.97	6.83
GraphMixer	96.89 ± 0.04	91.62 ± 0.52	97.17 ± 0.02	84.04 ± 0.12	84.16 ± 0.34	73.51 ± 0.14	7.00
DyRep	94.43 ± 0.32	69.46 ± 2.66	98.13 ± 0.04	84.50 ± 0.60	83.09 ± 2.20	71.40 ± 2.12	8.00

Table 4: Results of the future link prediction task - inductive setting. We report the mean test set area under the ROC curve (AUC-ROC) and standard deviation in percent averaged over random weight initializations. Methods are ranked by average rank across all datasets (lower is better).

Method ↓ / Dataset →	Wikipedia	UCI	Reddit	MOOC	Enron	LastFM	Avg. Rank
DyGMamba	$98.72_{\pm 0.03}$	$92.70_{\pm 0.19}$	$98.88_{\pm 0.01}$	$89.34_{\pm 0.12}$	$\mathbf{89.76}_{\pm 0.21}$	$94.36_{\pm 0.13}$	**1.83**
TETN (ours)	$\mathbf{98.81}_{\pm 0.05}$	$\mathbf{95.93}_{\pm 0.21}$	$\mathbf{98.92}_{\pm 0.10}$	$80.47_{\pm 0.50}$	$89.61_{\pm 1.10}$	$\mathbf{97.35}_{\pm 0.35}$	**2.33**
DyGFormer	$98.49_{\pm 0.02}$	$92.43_{\pm 0.20}$	$98.70_{\pm 0.02}$	$87.75_{\pm 0.42}$	$89.59_{\pm 0.10}$	$94.10_{\pm 0.09}$	**3.00**
TGN	$97.71_{\pm 0.19}$	$86.27_{\pm 1.49}$	$97.30_{\pm 0.12}$	$\mathbf{91.58}_{\pm 0.74}$	$79.40_{\pm 1.77}$	$82.61_{\pm 2.62}$	4.00
JODIE	$94.43_{\pm 0.28}$	$78.78_{\pm 1.11}$	$96.42_{\pm 0.13}$	$83.82_{\pm 0.30}$	$80.16_{\pm 1.50}$	$82.49_{\pm 0.94}$	5.67
GraphMixer	$96.26_{\pm 0.04}$	$89.26_{\pm 0.42}$	$94.95_{\pm 0.08}$	$82.76_{\pm 0.13}$	$76.08_{\pm 0.92}$	$80.34_{\pm 0.14}$	6.17
TGAT	$95.93_{\pm 0.19}$	$77.41_{\pm 0.65}$	$97.02_{\pm 0.04}$	$86.67_{\pm 0.24}$	$64.25_{\pm 1.29}$	$76.76_{\pm 0.22}$	6.33
DyRep	$91.31_{\pm 0.40}$	$58.84_{\pm 2.54}$	$95.87_{\pm 0.21}$	$83.42_{\pm 0.77}$	$75.82_{\pm 3.14}$	$82.82_{\pm 1.17}$	6.83
CTAN	$92.59_{\pm 0.70}$	$48.58_{\pm 6.02}$	$82.35_{\pm 4.03}$	$66.38_{\pm 1.59}$	$61.49_{\pm 2.78}$	$75.23_{\pm 2.24}$	8.83

A.3 TGBL Dataset

Method ↓ / Dataset →	tgbl-wiki	tgbl-review
DyGFormer	$\mathbf{0.79}_{\pm \mathbf{0.004}}$	$0.224_{\pm 0.015}$
TETN (ours)	$0.61_{\pm 0.02}$	$\mathbf{0.375}_{\pm \mathbf{0.003}}$
TGN	$0.39_{\pm 0.06}$	$0.349_{\pm 0.020}$

Table 5: MRR values for tgbl-v2 datasets.

In Table 5, we present the MRR scores obtained for the link prediction task on tgbl-wiki and tgbl-review datasets discussed in Huang et al. (2023). We observe that on the large-scale tgbl-review dataset (with $4,873,540$ edges), `TETN` performs better than TGN and DyGFormer. This again emphasizes our model's ability to deal with large-scale, long-range datasets. This is in alignment with our results on CTDG datasets such as, LastFM.

A.4 Ablation Studies

In this section, we present the ablation results for different parts of the model. Tables 6 and Table 7 show the impact of Hopfield layers and the use of edges in TET. We see that each component is essential for our model. Switching off any of the components leads to a drop in the average precision metric.

We also tried ablation experiments *without using the energy attention* component, i.e., by considering $E = E^{HN}$, without E^{ATT}. This led to a significant drop in precision and made the model behave as a random predictor on the current dataset.

Table 6: Average Precision scores for different model configurations showing the impact of Hopfield Network, and edge features.

Dataset	No HN	No Edges	Full Model
UCI (trans.)	$94.05_{\pm 0.20}$	$93.72_{\pm 0.01}$	$\mathbf{98.98}_{\pm \mathbf{0.03}}$
UCI (ind.)	$91.36_{\pm 0.60}$	$80.82_{\pm 0.12}$	$\mathbf{95.44}_{\pm \mathbf{0.15}}$
Wikipedia (trans.)	$97.91_{\pm 0.30}$	$92.15_{\pm 0.03}$	$\mathbf{99.20}_{\pm \mathbf{0.01}}$
Wikipedia (ind.)	$97.54_{\pm 0.30}$	$91.43_{\pm 0.05}$	$\mathbf{98.83}_{\pm \mathbf{0.13}}$

Table 7: ROC-AUC scores for different model configurations showing the impact of Hopfield Network, and edge features

Dataset	No HN	No Edges	Full Model
UCI (trans.)	$94.94_{\pm 0.02}$	$94.56_{\pm 0.01}$	$\mathbf{98.95_{\pm 0.10}}$
UCI (ind.)	$91.90_{\pm 0.07}$	$84.96_{\pm 0.01}$	$\mathbf{95.93_{\pm 0.21}}$
Wikipedia (trans.)	$97.83_{\pm 0.03}$	$93.58_{\pm 0.02}$	$\mathbf{99.15_{\pm 0.05}}$
Wikipedia (ind.)	$97.40_{\pm 0.03}$	$92.67_{\pm 0.03}$	$\mathbf{98.81_{\pm 0.01}}$

A.5 Hyperparameter Studies

We evaluated our model by varying the hyperparameters γ_1 and γ_2 and report the results in Table 8. The experiment is conducted on tgbl-wiki v2 dataset. This highlights the effect of edge term (controlled by γ_2) in the ODE update equation (Eq. 10). We clearly see that the additional gradients of energy with respect to the memory states, $\frac{\partial E}{\partial s}$, and with respect to the edges, $\frac{\partial E}{\partial e}$, both play an essential role in the update equations.

γ_1	γ_2	MRR
0.0	0.0	0.31
0.0	0.5	0.61
0.5	0.0	0.39
0.5	0.5	0.58

Table 8: Ablation on γ_1, γ_2 on tgbl-wiki

A.6 Complexity Analysis

	Peak memory (GB)			Training time per epoch (min)		
Method	UCI	Enron	LastFM	UCI	Enron	LastFM
DyGMamba	1.93	2.74	4.17	0.60	2.05	28.45
DyGFormer	2.30	3.23	7.57	0.62	2.73	47.00
TETN (ours)	2.05	1.46	3.99	1.265	2.10	36.34

Table 9: Comparison of peak memory utilization and training time per epoch across datasets for TET with DyGFormer (Yu et al., 2023), and DyGMamba (Ding et al., 2024).

We analyze the computational complexity of TET layer forward pass, where D is the feature dimension, Y is the hidden dimension of attention, K is the number of neighbours over k-hop. Let $n, \mathbf{e}$ denotes the number of nodes, edges of the graph, and H is the number of attention heads.

Energy Attention (Eq. 8) For a single head, the energy attention involves: (1) query-key projections $\mathcal{O}(nDY)$, (2) A_{BC} term involves $\mathcal{O}(nKY)$, and (3) aggregation of the energy attention term involves $\mathcal{O}(nD)$ computations.

Hopfield network (Eq. 3) We use the L-layered MLP-based Hopfield network used in the implementation of energy transformer (Hoover et al., 2024) which has a complexity of $\mathcal{O}(nD^2L)$.
Overall run time Complexity for forward pass: $\mathcal{O}(nDY) + O(nYK) + O(nD^2L)$.

Considering a single CTDG event and a single layer, the forward-pass complexity of the `TETN`, DyGFormer, and DyGMamba models are as below.

$$\begin{aligned} \texttt{TETN}: &\quad \mathcal{O}(DY + YK + D^2), \\ \text{DyGFormer}: &\quad \mathcal{O}(K^2D + KD^2), \\ \text{DyGMamba}: &\quad \mathcal{O}(KD\, d_{\text{ssm}}). \end{aligned}$$

We note that d_{ssm} of the state-space model DyGMamba is equivalent to the hidden dimension, Y of the `TETN` model.

Memory complexity for forward pass of `TETN`: $\mathcal{O}(n + \mathbf{e})$, which is identical to that of Graph Attention Networks (Veličković et al., 2017).

Empirical results on the run-time and memory usage of the TET graph embedding module during training are presented in Table 9. `TETN` achieves a favourable trade-off between memory efficiency and computation time. On the Enron and LastFM datasets, `TETN` exhibits notably lower peak memory consumption compared to DyGFormer (1.46 GB vs. 3.23 GB on Enron, and 3.99 GB vs. 7.57 GB on LastFM), highlighting its ability to handle large dynamic graphs more efficiently. While DyGMamba maintains slightly lower runtime on smaller datasets such as UCI, the difference diminishes as dataset size and temporal density increase, indicating that `TETN` scales competitively for real-world temporal graphs.

A.7 Dataset Statistics

The statistics of the CTDG datasets used in our experiments are given below in Table 10. Each dataset is split chronologically, in the ratio of $70\% - 15\% - 15\%$, for training, validation, and testing, respectively.

	# Nodes	# Edges	# Edge ft.	# Node ft.	# Duration	# Time Stamps
Wikipedia	9,227	157,474	172	0	1 month	152,757
UCI	1,899	59,835	172	0	196 days	58,911
Reddit	10,984	672,447	172	0	1 month	669,065
Enron	184	125,235	0	0	3 years	22,632
MOOC	7,144	411,749	4	0	17 months	345,600
LastFM	1,980	1,293,103	0	0	1 month	1,283,614

Table 10: Statistics of the datasets used in our experiments

A.8 Hyper-parameter Space

The grid of hyper-parameters used in our experiments is listed in Table 11.

Hyper-parameter	Values
# layers	{1, 2, 3}
learning rate	{1e-3, 1e-4}
dropout	{0.1}
Time encoding dim	{100}
Node embedding dim, D	{172}
Edge embedding dim, D	{172}
Hidden dim (in Attention), Y	{172}
# Heads, H	{2}
ϵ	{ 0.5, 1.0}
γ_1, γ_2	{1.0}

Table 11: Hyper-parameter Space

A.9 Algorithm for Training and Inference

Algorithm 1 $\mathbf{emb}_t(x, \hat{s}, e, u, v, w)$

Layer parameters $W^K, W^Q \in \mathbf{R}^{Y \times D}$
$x^{(1)} \leftarrow x$
$\mathcal{V} \leftarrow \{u, v, w\}$ {w represents the node from the negative edge sample}
for all l = 1,2. . . L **do**
  TET Block Starts
  $\mathcal{V} \leftarrow \mathcal{V} \cup \eta_u^{(l)}(0,t) \cup \eta_v^{(l)}(0,t) \cup \eta_w^{(l)}(0,t)$ { $\eta_u^{(l)}(0,t)$ is the set of l-hop neighbors of node u till time t.}
  $g^{(l)} \leftarrow \text{LayerNorm}(x^{(l)})$
  $s \leftarrow \text{LayerNorm}(\hat{s})$ {s is a local variable}
  $\left.\begin{array}{l} K_{\alpha B} \leftarrow \sum_j W^K_{\alpha j}(g^{(l)}_{Bj} + s_{Bj}) \\ Q_{\alpha B} \leftarrow \sum_j W^Q_{\alpha j}(g^{(l)}_{Bj} + s_{Bj}) \end{array}\right\}$ {**Memory infusion** $(\forall B \in \mathcal{V}, \alpha \in D)$}
  $E \leftarrow E^{ATT} + E^{HN}$ {**Energy attention** and **Hopfield Energy** are computed.}
  $\left.\begin{array}{l} x_u^{(l+1)} \leftarrow g_u^{(l)} - \eta \nabla_{g_u, s_u} E \\ x_v^{(l+1)} \leftarrow g_v^{(l)} - \eta \nabla_{g_v, s_v} E \\ x_w^{(l+1)} \leftarrow g_w^{(l)} - \eta \nabla_{g_w, s_w} E \end{array}\right\}$ {**Token updation** using energy gradients}
  TET Block Ends
end for
return $x_u^{(L)}, x_v^{(L)}, x_w^{(L)}$

Algorithm 2 Link Prediction using TET

$\mathbf{s} \leftarrow \mathbf{0}$ {Initialize memory to zeros}
$\mathbf{rm} \leftarrow [\,]$ {Initialize raw messages}
for all batch $(i, j, x, e, t) \in$ training data **do**
  $\mathbf{n} \leftarrow$ sample negatives
  $\mathbf{m} \leftarrow \text{msg}(\mathbf{rm})$ {Compute messages from raw features}
  $\tilde{\mathbf{m}} \leftarrow \text{agg}(\mathbf{m})$ {Aggregate messages for the same nodes}
  $\hat{\mathbf{s}} \leftarrow \text{memory\_updater}(\tilde{\mathbf{m}}, \mathbf{s})$ {Get updated memory}
  $z_i, z_j, z_n \leftarrow emb_t(x, \hat{s}, e, i, j, n)$ {Compute node embeddings} (Algorithm 1).
  $\mathbf{p}_{pos}, \mathbf{p}_{neg} \leftarrow \text{dec}(\mathbf{z}_i, \mathbf{z}_j), \text{dec}(\mathbf{z}_i, \mathbf{z}_n)$ {Compute interactions probs}
  $l = \text{BCE}(\mathbf{p}_{pos}, \mathbf{p}_{neg})$ {Compute BCE loss}
  $\mathbf{rm}_i, \mathbf{rm}_j \leftarrow (\hat{\mathbf{s}}_i, \hat{\mathbf{s}}_j, t, e), (\hat{\mathbf{s}}_j, \hat{\mathbf{s}}_i, t, e)$ {Compute raw messages}
  $\mathbf{rm} \leftarrow \text{store\_raw\_messages}(\mathbf{rm}, \mathbf{rm}_i, \mathbf{rm}_j)$ {Store raw messages}
  $\mathbf{s}_i, \mathbf{s}_j \leftarrow \hat{\mathbf{s}}_i, \hat{\mathbf{s}}_j$ {Store updated memory for sources and destinations}
end for

A.10 Energy equations for the static graph as described in

The components of the energy E for static ET are as follows:

$$E^{ATT} = -\frac{1}{\beta} \sum_{h=1}^{N} \sum_{C=1}^{N} log \left(\sum_{B \in \mathcal{N}_C} \exp \left(\beta \sum_{\alpha} K_{\alpha h B} Q_{\alpha h C} \right) \right), \tag{13}$$

where key and query tensors are as follows :

$$K_{\alpha h B} = \sum_j W^K_{\alpha h j} g_{jB}\,, \mathbf{K} \in \mathbf{R}^{Y \times H \times N}\,,$$
$$Q_{\alpha h C} = \sum_j W^Q_{\alpha h j} g_{jC}\,, \mathbf{Q} \in \mathbf{R}^{Y \times H \times N}\,,$$

with $W^K \in \mathbf{R}^{Y\times H\times D}$ and $W^Q \in \mathbf{R}^{Y\times H\times D}$ being learnable parameters, and

$$E^{HN} = -\sum_{B=1}^{N}\sum_{\mu=1}^{K} G\left(\sum_{j} \xi_{\mu j} g_{jB}\right), \xi \in \mathbf{R}^{K\times D}. \tag{14}$$

A.11 Notations

Notation	Description
E	Total Energy
E^{ATT}	Energy Attention
E^{HN}	Hopfield Energy
K	Key Tensor
Q	Query Tensor
F	Edge Tensor
W^K, W^Q, W^F	Weight Matrices
A, B, C	Node Indices
N	Number of Nodes
D	Node/Edge embedding dim
Y	Hidden dim (in Attention)
H	Number of Heads
ϵ	Time step in ODE
γ_1, γ_2	Token update parameters
m	Margin

Table 12: Notations