Conflict Event Actor Prediction using Spatial Graph Neural Networks

Niamat Zawad
Department of Computer Science
The University of Texas at Dallas
Dallas, USA
nxz190009@utdallas.edu

Patrick T. Brandt
Economic, Political and Policy Sciences
The University of Texas at Dallas
Dallas, USA
pbrandt@utdallas.edu

Latifur Khan
Department of Computer Science
The University of Texas at Dallas
Dallas, USA
lkhan@utdallas.edu

Vito J. D'Orazio
Department of Political Science
West Virginia University
Morgantown, USA
vito.dorazio@mail.wvu.edu

Javier Osorio
School of Government and Public Policy
The University of Arizona
Tucscon, USA
josorio1@arizona.edu

2025 IEEE International Conference on Big Data (BigData) | 979-8-3315-9447-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/BIGDATA66926.2025.11400998

***Abstract*—Predicting actors involved in future conflict events is a critical task in political forecasting, especially for identifying lesser-known actors who are underrepresented in mainstream media coverage. This paper presents a comprehensive benchmarking study of graph-based machine learning models for this task, formulating it as both a link prediction and a node classification problem. Given the complex and heterogeneous nature of conflict data—where entities such as actors, locations and event types interact along multiple semantic paths—we evaluate a range of models, including Graph Attention Networks (GAT), Metapath2Vec and other baselines—across transductive, semi-inductive, and inductive experimental settings. We quantify how well both homogeneous and heterogeneous models generalize to unseen nodes and edges, and analyze the sources of these gaps to offer actionable guidance on model selection across data regimes and deployment scenarios.**

***Index Terms*—Domain-specific social and collaborative applications, Social networks and services, Conflict Forecasting, Link Prediction, Metapath Aggregation, Graph Neural Networks**

I. Introduction

Conflict data can naturally be represented as graph-structured data, as it captures the relational nature of events through shared entities. Network analysis allows quantitative characterization of complex structures that are often too complicated to comprehend visually. As a result networks have been widely applied to various tasks within the conflict analysis domain. Events within a conflict setting often involve common actors, occur in overlapping geographical regions, or exhibit similar temporal or tactical patterns. Modeling such data as a graph enables the representation of these connections, where nodes may represent actors, events, or locations, and edges capture their interactions or co-occurrence, facilitating structured analysis of conflict dynamics.

Early research on conflict event prediction primarily utilized logistic regression models and simple baseline classifiers trained on data from news streams and Twitter [1]–[4]. Butts [5] developed a framework to model actions or behaviors within social contexts by parametrizing continuous-time hazard models with statistics such as recency, popularity, triadic closure, preferential attachment, and transitive/cyclic interactions to forecast relational events (who acts upon whom and when). Reciprocity patterns within actor networks [6] have been analyzed over multiple years to better understand temporal dynamics [7], while other work highlights how intangible factors such as thoughts and feelings contribute to the emergence of stable, self-organizing conflict networks [8]. Stoll [9] tackles the conflict prediction problem by grouping countries after grouping *negative-negative* hubs and authorities [10] into components and then calculating their distribution to determine the regional scope of conflicts. Graphs representing relationships between governments and anti-government actors have also been constructed to demonstrate how structural network properties influence conflict behavior [11]. More recently, spatio-temporal graph convolutional neural networks have been employed to forecast fatalities in conflict hotspots, capturing complex spatial and temporal dependencies [12]. Additionally, structural iterative representation learning combined with anomaly detection has been applied to vectorized country-level representations to predict potential future state-initiated attacks [13], [14]. Behavioral simulation models of network evolution adopt rule-based heuristics instead of cost-benefit optimization to explain tie formation [15] while adaptive evolutionary frameworks model how agents update link strengths over time in response to payoff feedback [16]. These models assume the network is non-dynamic, do not account for multiple simultaneous influences/covariates (e.g., homophily, reciprocity, triadic closure) and offer no statistical framework to test the model on real longitudinal network data. To address this gap Stadtfeld [17] proposes Dynamic Network Actor Model, a maximum likelihood estimation framework that estimates the coefficients to the aforementioned covariates so that the likelihood of observing the actual sequence of events is maximized. DyNAMs are actor-oriented and simulate

how individual actors decide to form or drop ties over time based on individual preferences in contrast to REM [5] and Exponential Random Graph Models [18], tie-oriented models that assess the likelihood of the entire network structure based on local structural patterns (e.g., edges, triangles).

Latent-variable frameworks tackle well-known weaknesses of classical exponential random graph models (ERGMs)—such as heavy reliance on hand-specified motifs, degeneracy, and poor scalability. Hoff's latent-space approach [19] assigns every actor an unobserved position in a "social space." A probability distribution over these latent positions makes the chance of a tie between two actors depend on how close they sit in that space—so transitive patterns (friends of friends connecting) arise naturally without being hard-coded as separate terms. Building on this latent representation, prior work has leveraged graph-based models to analyze, understand, and predict various aspects of conflict. Deng [20] applied GCN architecture to identify event subgraphs to predict future events by training on events occuring during a set number of preceding days. Additionally, deep learning models—including LSTM [21], BiLSTM [22], CNN [23], and hybrid CNN-LSTM architectures—have been proposed to model associations among terrorist attacks, capturing their functional similarities and contextual condition in order to identify the most at-risk districts likely to be targeted in future attacks.

One of the key goals of analyzing conflict networks is to infer potential collaborations between individuals, even when direct evidence of their interactions is unavailable. Essentially, this involves predicting hidden or unobserved connections within a social network composed of terrorist actors by estimating the likelihood of a connection between two nodes in a social structure. As such it can be framed as a binary classification problem, where the objective is to accurately identify whether a link exists by distinguishing positive examples from the rest of the dataset. Traditional approaches in conflict prediction have primarily relied on statistical modeling, rule-based systems or basic network analysis techniques that fail to capture the complex, multi-relational nature of conflict data. While some recent studies have experimented with node embeddings or basic GNNs for actor-event prediction, these efforts typically assume a homogeneous graph structure and overlook the rich semantics embedded in the heterogeneous and temporal relationships among actors, events, and geolocations. By introducing advanced heterogeneous graph models into the conflict prediction pipeline, we aim to bridge a significant methodological gap in the field. In addition to predictive tasks, graph-based representations of conflict data also enable unsupervised analyses such as node clustering, which can reveal latent groupings among actors or events. Identifying such clusters is particularly valuable for uncovering communities of coordinated actors, regional patterns of violence, or tactical similarities across conflict zones. These insights can support early warning systems, strategic intervention planning and a deeper understanding of conflict dynamics beyond individual events.

Our study introduces two key innovations. First, we curate the ICEWS conflict graphs into four progressively harder evaluation splits:

1) Transductive – In the transductive setting, we mimic the routine task of inferring new interactions among established actors whose histories are well documented. The train, validation, and test edges all lie inside one shared node set.
2) Semi-inductive – Test edges are unseen but end-point nodes are visible during training. This is analogous to asking the model to anticipate first-time relationships between familiar actors, such as a veteran peacekeeping force negotiating with an insurgent group it has never engaged before.
3) Inductive (50%) – Half of the test-set nodes are completely removed from the training graph. This partition reflects a more dynamic environment in which roughly half of the actors do not appear in the training graph at all, forcing the model to generalize from known entities to those only encountered at test time.
4) Inductive (100%) – All test-set nodes are withheld, forcing the model to generalise to entirely new actors. This simulates a scenario involving entirely unseen actors, akin to anticipating the role of a nascent coalition or a previously inactive state actor stepping onto the geopolitical stage for the first time.

These four evaluation regimes correspond to increasingly difficult, yet realistic, forecasting challenges. By increasing the share of previously unseen nodes and incident edges in the test set from half*(Inductive 50%)* to all*(Inductive 100%)*, we can quantify how each model's performance degrades as structural priors disappear; smaller declines across these settings indicate greater robustness and a stronger ability to exploit type information, attributes and heterogeneous context rather than memorized neighborhoods. We then apply a suite of GNNs to quantify how well methods originally designed for social/citation graph networks transfer to actor–actor networks within the conflict-event domain. Together, the novel data splits and the first large-scale benchmarking of classical GNN architectures on conflict data make our contribution distinctive within the link-prediction literature.

II. Related Work

Traditional link prediction methods can be broadly categorized into heuristic-based and latent feature-based approaches [24]. Heuristic methods estimate the likelihood of a link between two nodes by computing predefined similarity scores. Some common heuristic methods include common neighbors [25] which counts the number of shared neighbors between two nodes, Adamic-Adar Index [26] which assigns higher importance to less-connected common neighbors to downweight hubs, preferential attachment [27] which predicts links based on the product of the degrees of the two nodes assuming that highly connected nodes are more likely to gain new links and Katz Index [28], which sums over all paths between two nodes exponentially dampened by path length to favor shorter,

stronger connections. These heuristics are computationally efficient and interpretable but are often limited in capturing complex structural patterns.

In contrast to the traditional approaches, latent feature-based methods learn heuristics for each graph by learning from the graph topology and node/edge features. We group prior work along two axes—**graph type (homogeneous vs. heterogeneous)** and **model family (shallow latent embeddings vs. deep message-passing)**. For homogeneous–shallow methods, DeepWalk [29], Node2Vec [30], and LINE [31] learn node embeddings from random-walk co-occurrence or proximity preservation but are transductive and type-agnostic. Homogeneous–deep models such as GraphSAGE [32] learn inductive aggregators over local neighborhoods, yet still assume a single node/edge type. In heterogeneous–shallow settings, Metapath2Vec [33] leverages typed metapaths for constrained walks but treats sequences as flat text. Heterogeneous–deep approaches include relation-aware GNNs (R-GCN [34], CompGCN [35]), content and type-aware aggregation (HetGNN [36]), transformer-style attention over typed edges (HGT [37]), and metapath-aware GNNs (MAGNN [38]). These advances better capture multi-relational structure, yet largely underutilize spatial signals central to conflict dynamics. This paper concentrates on evaluating a range of latent feature–based methods which have consistently outperformed traditional heuristics by learning task-specific similarity functions directly from graph topology and attributes [39]. By systematically comparing these learned representations, we seek to identify which approaches best adapt to the complexity and heterogeneity inherent in conflict-event networks.

III. Methodology

While basic neural architectures can achieve strong link-prediction scores these results can be misinterpreted as in many interaction graphs a small set of edges occurs disproportionately often. In such cases a model may learn to exploit these frequency patterns rather than capturing the underlying structural or semantic relationships within the graph. This can lead to inflated performance metrics that reflect memorization of recurring edges rather than genuine predictive capability. To address this limitation, we construct semi-inductive and inductive dataset splits. In the semi-inductive setting, some nodes from the training set are withheld from the validation and test sets, ensuring that the model must generalize to partially unseen node combinations. In the inductive setting, all nodes in the validation and test sets are entirely absent from the training set. These strategies deliberately reduce the overlap and frequency bias between training and non-training edges, providing a more rigorous assessment of the model's ability to generalize to truly unseen data. By minimizing the extent to which edge frequency alone can drive predictions, these inductive settings enable a clearer evaluation of whether a model captures transferable patterns in the graph rather than overfitting to dataset-specific repetition.

Graph neural networks (GNNs) are particularly well-suited to such inductive scenarios. GNNs explicitly incorporate the topology of the graph into their learning process. Through iterative neighborhood aggregation, they combine node features, edge attributes and structural patterns to produce context-aware node representations. This allows GNNs to infer relationships between nodes that have never been observed together during training, leveraging connectivity patterns, shared neighbors and higher-order graph motifs.

Next, we describe the graph neural network (GNN) models evaluated in this study.

A. Evaluated models

To benchmark the effectiveness of different graph learning architectures for conflict actor prediction, we evaluate a suite of models that share the ability to propagate information across connected nodes to capture the structural, relational and spatial dependencies present in conflict event graphs. This shared design enables them to learn latent interaction patterns between actors, making them well suited for comparison under a unified framework. Evaluating both homogeneous and heterogeneous GNN variants allows us to assess how different levels of relational expressiveness and type awareness influence predictive performance in multi-relational conflict networks.

1) Graph Convolution Network: Graph Convolutional Networks (GCN) extend the concept of convolution from grid-structured data to arbitrary graph structures. In GCNs, each node updates its representation by aggregating and transforming the feature information of its neighbors through a shared, learnable weight matrix. This allows GCNs to capture local structural dependencies and feature smoothness across connected nodes. However, the aggregation process in standard GCNs treats all neighboring nodes as equally important, which can limit expressiveness in graphs where some neighbors carry more informative signals than others.

2) Graph Attention Network: Graph Attention Networks (GAT) build upon GCNs by introducing an attention mechanism to make neighborhood aggregation adaptive and learnable. Instead of averaging neighbor features uniformly, GAT computes attention coefficients that measure the relative importance of each neighbor based on feature similarity. These coefficients are used to weight neighbor contributions during message passing, enabling the model to focus on more informative connections. Multiple attention heads are employed to stabilize training and capture diverse relational patterns. Unlike GCNs, which rely on fixed normalization, GAT dynamically learns how much influence each neighbor should have, offering greater flexibility—especially when node features are expressive.

3) R-GCN: The Relational Graph Convolutional Network (R-GCN) is an extension of the traditional Graph Convolutional Network designed to operate on multi-relational graphs, where edges represent different types of relations between entities. Unlike standard GCNs that use a single shared weight matrix for all edges, R-GCN introduces relation-specific transformation parameters to capture the semantics of diverse edge types. This allows the model to learn more expressive

(a)

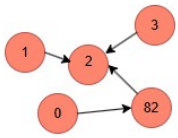
Event ID	Event Date	Source Name	Source Sectors	Source Country	Event Text	CAMEO Code	Intensity	Target Name	Target Sectors
33420423	2018-07-01	Business (United Kingdom)	Social,Business	United Kingdom	Make pessimistic comment	12	-0.4	Citizen (United Kingdom)	General Population / Civilian / Social,Social
33428831	2018-07-01	Head of Government (Germany)	Government,Executive,Executive Office	Germany	Consult	40	1.0	Horst Seehofer	Government,Executive, (National) Minor Party,Lo...
33424707	2018-07-01	Angela Merkel	Government,Executive,Executive Office,Parties,...	Germany	Engage in negotiation	46	7.0	Horst Seehofer	Government,Executive, (National) Minor Party,Lo...

Source Name	Target Name	Source ID	Target ID
Business (United Kingdom)	Citizen (United Kingdom)	0	82
Head of Government (Germany)	Horst Seehofer	1	2
Angela Merkel	Horst Seehofer	3	2

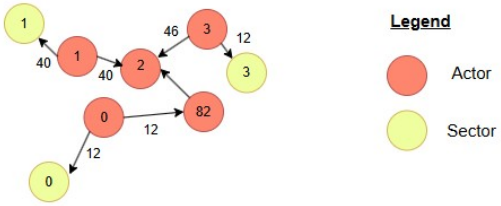
Source ID	Source Sectors	Source Sectors ID
0	Social,Business	0
1	Government,Executive,Executive Office	1
3	Government,Executive,Executive Office,Parties,...	3

(b)

Source ID	CAMEO Code
0	12
1	40
3	46



(c)



(d)

Fig. 1. (a) A snapshot from the ICEWS18 dataset, where nodes and edges are derived from recorded event interactions between entities. The red and yellow nodes stand for the actor and sector type nodes respectively while the edge label denotes the *CAMEO code*/type of action. (b) From this data, we extract *actor-actor*, *actor–CAMEO/action* code, and *actor–sector* subgraphs to create separate dataframes. (c) The *actor-actor* graph is used for training homogeneous graph models, while (d) the *actor–CAMEO code–sector* graph structure is used for heterogeneous graph models, where edge labels represent the associated *CAMEO* action codes.

node embeddings by aggregating information not only from neighboring nodes but also by considering the type of relation through which the information is propagated.

4) GraphSAGE: GraphSAGE (Graph Sample and Aggregate) is a general inductive framework for learning node representations in large-scale and dynamic graphs. Unlike transductive models that learn embeddings for fixed node IDs, GraphSAGE generates embeddings by sampling and aggregating features from a node's local neighborhood. This makes it suitable for inductive settings, where new nodes may appear during inference. The key idea is to use a learnable aggregation function—such as mean, LSTM, or max pooling—that combines the features of sampled neighbors at each layer. These aggregated messages are then concatenated with the node's own features and passed through a neural network layer. By stacking multiple such layers, the model can capture multi-hop neighborhood information. The training parameters are learnt by optimizing the following loss function.

$$J_G(z_u) = -\log \sigma(z_u^\top z_v) - Q\,\mathbb{E}_{v_n \sim P_n(v)}[\log \sigma(-z_u^\top z_{v_n})] \quad (1)$$

Here v denotes a node encountered in a fixed-length random walk starting from u, σ is the sigmoid activation, P_n specifies the distribution used for drawing negative samples, and Q indicates how many negative samples are taken per positive example. This corresponds to the skip-gram with negative sampling (SGNS) objective: pulling random-walk co-occurring pairs (u, v) together by making $z_u^T z_v$ large and pushing u away from Q negatives $v_n \sim P_n$ by making $z_u^T z_{v_n}$ small. GraphSAGE's sampling mechanism also enables efficient mini-batch training on large graphs, reducing memory usage and computation compared to full-graph methods. This flexibility and scalability make GraphSAGE a popular choice for tasks like node classification, link prediction and recommendation in evolving graph environments.

5) HGT: The Heterogeneous Graph Transformer (HGT) is a graph neural network architecture specifically designed to model complex, heterogeneous graphs that contain multiple types of nodes and edges. Unlike traditional GNNs that assume a homogeneous structure, HGT incorporates type-specific parameters and attention mechanisms to capture the diverse semantics of different node and relation types. It uses a multi-head attention framework, where each head learns to focus on different aspects of the neighborhood,

and relation-specific transformations are applied to encode the structural and semantic differences among edge types. Additionally, HGT uses separate projection matrices for each node and relation type to preserve type-aware information during message passing. This allows the model to learn highly expressive representations that are sensitive to both the content and structure of heterogeneous graphs.

6) Node2Vec: Node2Vec is an unsupervised algorithm designed to learn continuous feature representations for nodes in a network. It operates on homogeneous graphs where all nodes and edges are treated as being of the same type. Node2Vec combines the principles of random walks with the skip-gram model [40] to capture network structure. The algorithm performs multiple random walks from each node to generate sequences that reflect local and global connectivity patterns. These sequences are treated analogously to sentences and node embeddings are trained to maximize the likelihood of observing a node's neighbors in these walks using the skip-gram objective. The resulting embeddings encode both homophily (nodes that are neighbors) and structural equivalence (nodes playing similar roles). Node2Vec is fundamentally designed for homogeneous graphs, where all nodes and edges are considered indistinguishable in terms of type.

7) Metapath2Vec: Metapath2Vec is an unsupervised representation learning model designed specifically for heterogeneous graphs, where nodes and edges can belong to multiple types. Unlike traditional methods like Node2Vec that perform uniform or biased random walks on homogeneous graphs, Metapath2Vec introduces the concept of metapath-guided random walks. A metapath defines a schema-based sequence of node types that guides the random walker to traverse the graph in a semantically meaningful manner. The resulting node sequences are used to train a skip-gram model, similar to Word2Vec [41], which learns embeddings that preserve not only topological proximity but also semantic relatedness governed by the metapath structure. By incorporating type-aware walk strategies, it captures both the local and global semantic relationships that arise from the interplay between different node and edge types. This enables the model to produce embeddings that are significantly more informative than those learned via structure-only methods in complex networks like social graphs, knowledge bases or event data.

8) MAGNN: MAGNN is a graph neural network model specifically designed for heterogeneous graphs, where nodes and edges can belong to different types. MAGNN introduces a two-level aggregation strategy. In the intra-metapath stage, each concrete path instance $P_{i(v,u_j)}$ that matches schema P_i and starts at node v is encoded into a vector $\mathbf{h}_{P_i(v,u_j)}$ [42]. A concrete example of a metapath instance is shown in Figure 2, where the metapath schema is *[Actor 1, Sector, Actor 2, Sector, Actor 1]* and one possible instance of this schema is *[Jose Maria Sison, Dissident, Roger Duterte, Government, Jose Maria Sison]*, where each placeholder in the schema is instantiated with a specific entity. An attention layer assigns a weight $\beta_{P_i(v,u_j)}$ to every instance, and the weighted sum, passed through a non-linearity σ, yields the schema-specific embedding $\mathbf{h}^v_{P_i}$. Continuing the metapath instance example above, an alternative instance such as *[Jose Maria Sison, Dissident, Maria Consuelo, Government, Jose Maria Sison]*—where *Maria Consuelo* is a relatively minor political figure in the dataset—may receive a lower attention weight due to her limited co-occurrence and weaker structural relevance in the graph. During inter-metapath aggregation, these embeddings are first averaged over all nodes participating in a metapath schema to obtain the summary vector $\mathbf{s}_i$. A second attention mechanism produces schema weights β_{P_i}, and the final node representation is computed as $\mathbf{h}^v = \sigma\big(\sum_i \beta_{P_i}\mathbf{h}^v_{P_i}\big)$. In this way, MAGNN first pinpoints the most relevant path instances within each schema and then adapts the relative importance of whole schemas, producing node embeddings that are simultaneously instance-aware and schema-aware.

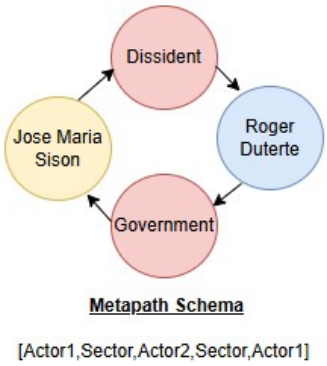


Fig. 2. An example of a metapath schema and the corresponding metapath instance. MAGNN enumerates and aggregates over every such instance in the heterograph.

This hierarchical aggregation enables MAGNN to encode both structural patterns and semantic contexts more effectively than earlier metapath-based models. Compared to Metapath2Vec which uses metapaths to guide random walks, MAGNN learns over metapath instances directly and performs end-to-end optimization. This allows it to adaptively learn which paths and substructures are most predictive for a given task, rather than relying on static sampling and unsupervised objectives.

IV. Experimental Setup

A. Implementation Details

We implement and evaluate all models using the PyTorch Geometric framework on a single NVIDIA RTX 2080, following the hyperparameter configurations summarized in Table III. Each model is trained with the Adam optimizer using a learning rate selected to ensure stable convergence without overfitting. We adopt a binary link prediction setup across all graph-based architectures, where the training objective is to distinguish positive from sampled negative edges. For GNN-based approaches, the final link prediction score is computed by a *predictor network*—a lightweight feed-forward module that takes the embeddings of two nodes $(\mathbf{h}_u, \mathbf{h}_v)$ and

predicts the likelihood of an edge existing between them. Unless otherwise noted, the predictor is implemented as a two-layer multilayer perceptron (MLP) with ReLU activation and a sigmoid output, operating on the element-wise product $\mathbf{h}_u \odot \mathbf{h}_v$. This separation between the encoder (which produces node embeddings) and the predictor (which models pairwise interaction) allows consistent evaluation across architectures and facilitates the comparison between homogeneous and heterogeneous GNNs.

Homogeneous GNNs: The baseline graph convolutional models (GCN, GAT, and GraphSAGE) operate on homogeneous *actor-actor* graphs derived from ICEWS/ACLED event interactions. For GCN, we use two layers with ReLU activations and a hidden dimension of 64, optimized with a learning rate of 0.01. The choice of two layers balances depth and oversmoothing, as deeper variants showed marginal gain. GAT extends this with relation-aware aggregation through multi-head attention (two heads in the first layer, one in the second). Feature and attention dropout of 0.2 mitigate overfitting. GraphSAGE employs mean aggregation with an MLP dropout of 0.5. It uses a structured tail-corruption negative sampling strategy with 1 negative per positive edge.

Relational and Heterogeneous GNNs: To model typed or multi-relational edges, we include R-GCN, HGT, and MAGNN. R-GCN uses two convolutional layers with basis decomposition (30 bases) and a hidden dimension of 64, trained on CPU in a full-batch fashion following the original paper. A dropout rate of 0.2 is applied to node embeddings, and tail-corruption sampling (1 negative per positive) is used to maintain relational consistency. HGT employs two transformer layers with two attention heads per layer, enabling relation-specific message passing and dynamic importance weighting across node types. It uses a dropout of 0.3 and the same tail-corruption negative sampling. MAGNN leverages metapath-based contextual aggregation with recurrent RotatE-type composition (RotatE0). A hidden dimension of 64, attention vector dimension of 128, and dropout rate of 0.5 yielded the best trade-off between expressiveness and generalization. The model is trained with early stopping (patience of 5) and uniform 1:1 negative sampling.

Random Walk Models: For unsupervised baselines, Metapath2Vec and Node2Vec are trained using the Skip-Gram with Negative Sampling (SGNS) objective. Both use 128-dimensional embeddings and a learning rate of 0.01 with the SparseAdam optimizer. Node2Vec performs homogeneous random walks with $(p, q) = (1.0, 1.0)$, walk length 20, and context size 10. Metapath2Vec instead performs metapath-guided walks (*actor–interacts–actor–interacted_by–actor*) to capture heterogeneous semantic context. Each model samples 20 negatives per positive pair and trains over 50 epochs with batch size 128 (walk pairs). These parameters ensure sufficient walk diversity while maintaining computational tractability.

Across all models, evaluation is performed using Area Under Curve(AUC) and Average Precision (AP). Early stopping based on validation loss prevents overfitting, and gradient clipping (when applicable) stabilizes training in the higher-capacity architectures such as HGT and MAGNN.

B. Dataset

To ensure comparability with prior link-prediction studies, we employ the ICEWS14 and ICEWS18 datasets for training and evaluation. The results on ICEWS14 are omitted, as they exhibit trends nearly identical to those of ICEWS18. Although simplified versions containing only actor IDs and timestamps are publicly available, metapath-based GNNs require richer node and relation types. Accordingly, we obtained the full ICEWS archive and then filtered it to include only events from 2014 and 2018, yielding our ICEWS14 and ICEWS18 subsets. From each of these we then extract edges *actor-actor*, *actor-cameo code*, *actor-sector*. We extract only the actor, sector, and CAMEO code features, omitting the rest. These particular features are selected based on the hypothesis that they are the most informative for predicting actor links. These features are then used to construct graph representations suitable for both homogeneous and heterogeneous GNNs. In the heterogeneous graphs, we define three relation types—*actor–actor*, *actor–CAMEO code*, and *actor–sector*—each forming an edge set that captures a distinct aspect of the conflict dynamics. The *actor–actor* edges represent direct interactions between entities participating in the same event, the *actor–CAMEO code* edges connect actors to the event type or action category, and the *actor–sector* edges associate actors with their organizational or functional sectors. These relation-specific edge sets are combined into a unified heterogeneous graph that preserves the multi-relational structure of the data. For homogeneous GNNs which assume a single node and edge type, we construct a simplified graph by retaining only the *actor–actor* relations. This focuses the model purely on the co-occurrence structure among actors. The overall graph creation process, including feature extraction and relation mapping, is illustrated in Figure 1.

In addition to ICEWS14 and ICEWS18, we also evaluate our models on the Armed Conflict Location and Event Data (ACLED) [43] dataset to test their generalizability across independently curated conflict corpora. To maintain computational feasibility we truncate the ACLED data to include only events occurring after January 1, 2020, which substantially reduces the number of nodes and edges while preserving contemporary conflict dynamics. Similar to ICEWS, we extract the *actor*, *sector*, and *subevent* type fields, where the subevent field represents the event's action category. These attributes are used to construct the *actor–actor*, *actor–sector*, and *actor–subevent* relations that form the heterogeneous conflict graph, while homogeneous baselines are trained solely on actor–actor connections.

C. Sampling

1) Evaluation Splits: To support transductive, semi-inductive and inductive link prediction experiments, we performed customized dataset splits for each graph. In the transductive setting, the full set of nodes is preserved across training and testing, while only a subset of edges is used for training.

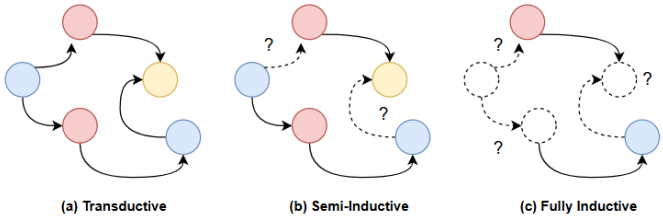


Fig. 3. (a) *Transductive:* Nodes and edges are seen in train, validation and test splits. (b) *Semi-inductive:* Edges in validation and test splits are not observed in train split. (c) *Inductive:* None of the nodes and edges in validation and test splits are observed in train split.

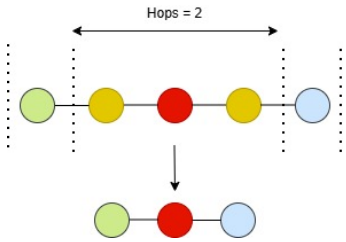


Fig. 4. Illustration of negative neighborhood sampling for link prediction. The red node represents the source node, and nodes located within a 2-hop radius in this example are considered for sampling. The green and blue nodes, which lie exactly two hops away, are treated as hard negatives—they are not directly connected to the source node but are structurally close and thus plausible candidates for link formation. This approach yields more informative negative samples that better evaluate the model's ability to distinguish between true and false links.

This setup assumes access to the entire node set during training and is commonly used to evaluate a model's ability to recover missing links within a fixed graph structure. In contrast, the semi-inductive setting involves removing all edges connected nodes in the validation and test graphs to those in the training graph, simulating scenarios where the model must generalize to unseen edges. Finally in the inductive setting in addition to no common edges, overlapping nodes between validation, test and train splits are removed in grades of 50 and 100%. This strict separation allows us to evaluate how well different models can generalize to new entities in unseen regions of the graph. These link predictions settings are illustrated in Figure 3. For each setting, we partition the edges into training, validation and test sets in an 80:10:10 ratio.

In link prediction, the training data naturally contains only positive edges—pairs of nodes (u, v) that are observed to be connected. If we train only on these, the model can trivially predict all pairs as connected and still achieve high recall, without actually learning to distinguish true links from non-links. Negative sampling addresses this by generating artificial negative examples—pairs of nodes that are not connected in the observed graph. During training, the model is encouraged to assign high scores to positive pairs and low scores to negative pairs, effectively learning a decision boundary between likely and unlikely links.

If negatives are sampled uniformly from the entire remaining node set, most pairs $(u, \tilde{v})$ would be topologically distant, meaning there is no short path or shared neighborhood between them. Such negatives are trivially easy for the model because distant node pairs tend to have embeddings that are already very dissimilar: in dot-product or bilinear decoders like DistMult, this naturally results in very low scores without the model needing to learn meaningful relational patterns. As a result, metrics such as AUC/AP can become artificially inflated, since the model appears to "reject" these negatives correctly without actually demonstrating nuanced link-discrimination ability. To address this, we use neighborhood-aware negative sampling, in which the corrupted tail node $\tilde{v}$ is drawn from nodes within a small k-hop neighborhood of u (excluding v and other known positives), as illustrated in Figure 4. This forces the model to distinguish between structurally plausible but false edges and true edges, making the evaluation more challenging and realistic. By drawing $\tilde{v}$ from the local neighborhood of u, negatives are *hard* (topologically or semantically close to v) yet remain non-edges. Table I reports the proportion of negative edges sampled from the local neighborhood across all experimental settings for the ICEWS18 dataset. In the transductive and semi-inductive configurations, roughly 75% of negatives originate from within each node's local vicinity, indicating that most sampled non-edges are structurally plausible and therefore harder to distinguish. In contrast, the inductive setups exhibit substantially higher locality rates of up to 98%, as the disjoint training graphs constrain sampling to smaller, more interconnected neighborhoods. This pattern confirms that the inductive splits maintain strong structural coherence while still offering sufficient diversity in negative examples, resulting in a balanced and challenging evaluation environment for link-prediction models.

D. Results

1) Link Prediction: Table IV presents the updated link prediction results across the transductive, semi-inductive, and inductive evaluation regimes on both the ICEWS18 and ACLED datasets. A clear trend persists: performance degrades monotonically as the task becomes more inductive, confirming that unseen nodes substantially weaken relational reasoning. Across all models, R-GCN continues to achieve the highest AUC in the transductive and semi-inductive settings, validating the importance of relation-specific transformations. However, in the fully inductive splits—where no node overlap exists

TABLE I
NEGATIVE SAMPLING BREAKDOWN FOR ICEWS18 ACROSS ALL EXPERIMENTAL SETTINGS. EACH CELL SHOWS THE PERCENTAGE OF NEGATIVE EDGES SAMPLED FROM THE LOCAL .

Dataset	Split	Transductive/Semi-Inductive	Inductive (50%)	Inductive (100%)
ICEWS18	Train	74.8	75.6	98.3
	Val	75.6	72.6	98.2
	Test	74.6	67.8	97.3
ACLED	Train	64.5	75.6	98.3
	Val	64.1	72.6	98.2
	Test	65.7	67.8	97.3

TABLE II

PERFORMANCE OF MAGNN ON DIFFERENT METAPATH CONFIGURATIONS ACROSS EVALUATION SETTINGS FOR ICEWS18 AND ACLED DATASETS. HERE, NODE TYPE INDICES ARE DEFINED AS FOLLOWS: 0 — ACTOR$_1$, 1 — ACTOR$_2$, 2 — ACTION, AND 3 — SECTOR. EACH METAPATH PATTERN REPRESENTS A SPECIFIC RELATIONAL SCHEMA CONNECTING SOURCE AND TARGET NODES, WHERE LONGER AND MORE DIVERSE METAPATHS CAPTURE RICHER SEMANTIC DEPENDENCIES.

Dataset	**Metapath Configuration**	**Transductive**		**Semi-Inductive**		**Inductive (50%)**		**Inductive (100%)**	
		AUC	AP	AUC	AP	AUC	AP	AUC	AP
ICEWS18	(0,1,0)(1,0,1)	0.726	0.754	0.728	0.756	0.524	0.539	0.331	0.386
	(0,1,0)(0,2,0)(1,0,1)(1,2,1)	0.722	0.746	0.729	0.756	0.500	0.570	0.369	0.400
	(0,1,0)(0,2,0)(0,3,0)(1,0,1)(1,2,1)(1,3,1)	0.722	0.753	0.724	0.757	0.495	0.567	0.334	0.389
	(0,1,0,2,0)(0,2,0,1,0)(1,0,1,2,1)	0.721	0.754	0.730	0.762	0.512	0.594	0.459	0.456

TABLE III

HYPERPARAMETER CONFIGURATIONS OF ALL MODELS USED FOR LINK PREDICTION.

Model	**LR**	**Hidden Dim**	**Heads**	**Dropout**	**Epochs**	**Batch Size**
GCN	0.01	64	–	–	50	Full-batch
GAT	0.01	64	(2,1)	0.2 (feat/attn)	50	Full-batch
GraphSAGE	0.01	64	–	0.5 (predictor)	50	Full-batch
R-GCN	0.01	64	–	0.2	50	Full-batch
HGT	0.002	128 (hidden) / 64 (out)	2 per layer	0.3 (predictor)	50	Full-batch
Metapath2Vec	0.01	128	–	–	50	128 (walk pairs)
Node2Vec	0.01	128	–	–	50	128 (random walks)
MAGNN	0.005	64	2	0.5	10	8

TABLE IV

PERFORMANCE COMPARISON ACROSS GNN MODELS ON ICEWS18 AND ACLED DATASETS UNDER DIFFERENT SETTINGS.

Model	**ICEWS18**								**ACLED**							
	Transductive		Semi-Inductive		Inductive (50%)		Inductive (100%)		Transductive		Semi-Inductive		Inductive (50%)		Inductive (100%)	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
GCN	0.730	0.758	0.698	0.721	**0.711**	**0.761**	**0.640**	**0.674**	0.725	0.734	0.724	0.734	0.742	0.787	0.725	0.798
GAT	0.730	0.758	0.676	0.681	0.374	0.482	0.557	0.620	0.584	0.644	0.588	0.644	0.575	0.709	0.675	**0.816**
GraphSAGE	0.875	0.525	0.867	0.500	0.541	0.344	0.507	0.303	0.831	0.248	0.822	0.236	0.741	0.638	**0.836**	0.754
R-GCN	**0.954**	0.710	**0.926**	0.627	0.593	0.685	0.512	0.466	**0.956**	0.558	**0.954**	0.606	**0.879**	**0.804**	0.702	0.596
HGT	0.794	0.154	0.790	0.144	0.476	0.056	0.131	0.011	0.782	0.190	0.783	0.194	0.414	0.080	0.782	0.190
Metapath2Vec	0.936	0.820	0.886	0.707	–	–	–	–	0.822	0.631	0.825	0.623	–	–	–	–
Node2Vec	0.939	**0.965**	0.889	**0.933**	–	–	–	–	0.808	**0.891**	0.805	**0.889**	–	–	–	–
MAGNN	0.723	0.753	0.725	0.757	0.495	0.567	0.334	0.389	0.755	0.784	0.732	0.741	0.481	0.547	0.357	0.394

between training and test graphs—GraphSAGE consistently demonstrates the strongest robustness, with higher AP values than GCN and GAT. This behavior aligns with its design: learned neighborhood aggregators can generalize to new structural contexts even when node embeddings are unseen during training. By contrast, HGT exhibits noticeable degradation in inductive settings, likely due to the model's reliance on dense attention patterns that are harder to estimate in sparse relational graphs such as ICEWS and ACLED. Shallow proximity baselines such as Node2Vec and Metapath2Vec perform well only in transductive scenarios—when all nodes are observed at training time—since they cannot infer embeddings for unseen nodes. Among metapath-based methods random-walk–based Metapath2Vec generally achieves higher AUC, indicating better global separation between positive and negative edges, while MAGNN is competitive and in some settings yields higher AP, reflecting stronger precision among the top-ranked predictions. This is consistent with its architecture: MAGNN aggregates over multiple metapath schemas simultaneously and learns to weigh heterogeneous contexts via attention at both the instance (intra-metapath) and schema (inter-metapath) levels, which helps down-weight noisy or irrelevant paths, but this added expressiveness does not always translate into uniformly better overall ranking performance.

2) Statistical Significance Analysis: To verify the observed differences, we conducted paired two-sided *t*-tests between all model pairs (Table V). Most comparisons yield $p < 10^{-3}$, confirming that the performance gaps are statistically significant rather than random fluctuations. In particular, MAGNN's improvements over GraphSAGE, R-GCN, and Metapath2Vec are highly significant ($p < 10^{-10}$), validating that metapath-guided aggregation provides a measurable and reliable advantage in relational inference. Similarly, the large *t*-statistics between MAGNN and Node2Vec/Metapath2Vec ($|t| > 100$) indicate that semantic message passing, rather than proximity-only context, drives the observed performance gains.

TABLE V
PAIRED TWO-SIDED T-TESTS ON TRANSDUCTIVE LINK-PREDICTION SCORES.

Model A	Model B	t-statistic	p-value	Significant?
GCN	GraphSAGE	-3.17	6.8×10^{-3}	Yes
GCN	HGT	45.19	$< 10^{-15}$	Yes
GCN	R-GCN	-44.02	$< 10^{-15}$	Yes
GCN	Node2Vec	-1.50	0.15	No
GCN	Metapath2vec	-61.88	$< 10^{-15}$	Yes
GCN	MAGNN	-26.07	$< 10^{-12}$	Yes
GraphSAGE	HGT	21.04	5.4×10^{-12}	Yes
GraphSAGE	R-GCN	-12.10	8.4×10^{-9}	Yes
GraphSAGE	Node2Vec	2.37	3.3×10^{-2}	Yes
GraphSAGE	Metapath2vec	-16.58	3.3×10^{-10}	Yes
GraphSAGE	MAGNN	-13.72	1.6×10^{-9}	Yes
HGT	R-GCN	-191.82	$< 10^{-15}$	Yes
HGT	Node2Vec	-103.10	$< 10^{-15}$	Yes
HGT	Metapath2vec	-263.96	$< 10^{-15}$	Yes
HGT	MAGNN	-57.16	$< 10^{-15}$	Yes
R-GCN	Node2Vec	89.60	$< 10^{-15}$	Yes
R-GCN	Metapath2vec	36.88	$< 10^{-15}$	Yes
R-GCN	MAGNN	2.90	1.2×10^{-2}	Yes
Node2Vec	Metapath2vec	-142.16	$< 10^{-15}$	Yes
Node2Vec	MAGNN	-29.42	$< 10^{-14}$	Yes
Metapath2vec	MAGNN	5.41	3.1×10^{-3}	Yes

3) Effect of Metapath Configurations: Table II explores the impact of different metapath combinations on MAGNN's performance. Each tuple represents a distinct metapath schema connecting heterogeneous node types in the ICEWS18 graph. For instance $(0, 1, 0)$ denotes an *Actor* $\rightarrow$ *Action* $\rightarrow$ *Actor* metapath, capturing co-participation of actors in common events. Similarly, $(0, 1, 0, 2, 0)$ extends this with an intermediate *Sector* node (*Actor* $\rightarrow$ *Action* $\rightarrow$ *Sector* $\rightarrow$ *Action* $\rightarrow$ *Actor*), allowing the model to infer similarity through both shared actions and organizational sectors. Combinations such as $(0, 1, 0)(0, 2, 0)$ or $(0, 1, 0)(1, 2, 1)$ represent multi-schema aggregation, where **MAGNN** jointly reasons over multiple relational channels. As the *length and diversity* of metapath schemas increase, the model gains access to richer contextual information and captures higher-order semantic dependencies—reflected in gradual improvements in both AUC and AP. However, this comes at a steep computational cost: the number of metapath instances grows multiplicatively with the degrees of intermediate nodes, as shown in (2). Due to hardware constraints, only four metapath combinations were evaluated, but even within this limited configuration, a consistent upward trend is observed in the transductive and semi-inductive settings.

$$\begin{aligned}\text{Total Metapath Instances} &= \prod_{i=1}^{L} \text{Degree}(v_i) \\ &= \text{Degree}(v_1) \times \text{Degree}(v_2) \times \cdots \times \text{Degree}(v_L) \quad (2)\end{aligned}$$

E. Challenges

One of the key challenges encountered during MAGNN training is its high memory consumption, particularly when increasing the number of shortest paths collected for each metapath. Table II compares the performance of MAGNN across progressively richer metapath configurations. As more metapaths are introduced—such as incorporating both action- and sector-level contexts—the model generally benefits from increased structural and semantic expressiveness, yielding modest gains in both AUC and AP. However, this improvement comes at a steep computational cost: the number of instantiated metapath edges grows exponentially with the degrees of the intermediate nodes, since each hop in the metapath multiplies the branching factor of possible connections. Equation (2) shows the calculation for total metapath instances where L denotes the length of the metapath schema (the number of node types or hops), v_i represents the i^{th} node type encountered along the metapath, and Degree(v_i) denotes the number of neighbors associated with node v_i. Consequently, each additional hop multiplicatively increases the number of reachable path instances, leading to exponential growth in the total number of metapath instances. To mitigate this combinatorial explosion we constrained the neighborhood expansion to a maximum branch factor of five, meaning that only up to five neighboring edges were explored per node during metapath generation. While this truncation may discard informative relationships—potentially leading to slightly suboptimal downstream performance—it was necessary to maintain tractable memory and runtime requirements given hardware limitations. In future work a more adaptive neighbor selection strategy that prioritizes informative or high-attention nodes rather than sampling neighbors uniformly at random could better preserve structural diversity while still controlling computational overhead.

V. FUTURE WORK

Several directions can be explored to address the scalability limitations of MAGNN and further improve its performance in downstream tasks. One promising direction is to design adaptive sampling techniques that prioritize informative or high-weight paths instead of enumerating all possible combinations. Beyond MAGNN, this paper focuses exclusively on spatial (static) graph models; extending the study to temporal (dynamic) graph models—where both features and topology evolve over time—is left for future work. This includes evaluating time-aware architectures and training protocols that respect temporal ordering and address leakage and negative-sampling design under temporal constraints.

ACKNOWLEDGMENT

This research was supported by NSF award 2311142; used Delta at NCSA / University of Illinois through allocation CIS220162 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by NSF grants 2138259, 2138286, 2138307, 2137603, and 2138296.

REFERENCES

[1] Jordan Bakerman, Karl Pazdernik, Gizem Korkmaz, and Alyson G Wilson. Dynamic logistic regression and variable selection: Forecasting and contextualizing civil unrest. *International Journal of Forecasting*, 38(2):648–661, 2022.

[2] Yue Ning, Sathappan Muthiah, Huzefa Rangwala, and Naren Ramakrishnan. Modeling precursors for event forecasting via nested multi-instance learning. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1095–1104, 2016.

[3] Gizem Korkmaz, Jose Cadena, Chris J Kuhlman, Achla Marathe, Anil Vullikanti, and Naren Ramakrishnan. Multi-source models for civil unrest forecasting. *Social network analysis and mining*, 6:1–25, 2016.

[4] Naren Ramakrishnan, Patrick Butler, Sathappan Muthiah, Nathan Self, Rupinder Khandpur, Parang Saraf, Wei Wang, Jose Cadena, Anil Vullikanti, Gizem Korkmaz, et al. 'beating the news' with embers: forecasting civil unrest using open source indicators. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1799–1808, 2014.

[5] Carter T Butts. A relational event framework for social action. *Sociological methodology*, 38(1):155–200, 2008.

[6] Doug Bond, Joe Bond, Churl Oh, J Craig Jenkins, and Charles Lewis Taylor. Integrated data for events analysis (idea): An event typology for automated events data development. *Journal of Peace Research*, 40(6):733–745, 2003.

[7] Patrick Meier and EA Leicht. Conflict events-data and network analysis: A case study of afghanistan. In *Online unter: http://www. santafe. edu/events/workshops/images/f/f1/Sf_ csss06_meier_et_al. pdf [Zugegriffen 30*, 2008.

[8] Peter T Coleman, Robin R Vallacher, Andrzej Nowak, and Lan Bui-Wrzosinska. Intractable conflict as an attractor: A dynamical systems approach to conflict escalation and intractability. *American Behavioral Scientist*, 50(11):1454–1475, 2007.

[9] Richard J Stoll and Devika Subramanian. Hubs, authorities, and networks: Predicting conflict using events data. In *Annual Meeting of International Studies Association*. Citeseer, 2006.

[10] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[11] Nils W Metternich, Cassy Dorff, Max Gallop, Simon Weschle, and Michael D Ward. Antigovernment networks in civil conflicts: How network structures affect conflictual behavior. *American Journal of Political Science*, 57(4):892–911, 2013.

[12] Patrick T Brandt, Vito D'Orazio, Latifur Khan, Yi-Fan Li, Javier Osorio, and Marcus Sianan. Conflict forecasting with event data and spatio-temporal graph convolutional networks. *International Interactions*, 48(4):800–822, 2022.

[13] Mikel Joaristi and Edoardo Serra. Sir-gn: A fast structural iterative representation learning approach for graph nodes. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(6):1–39, 2021.

[14] Namitha Nayak, Manasa Rayachoti, Ananya M Gupta, GP Prerna, Sreenath MV, and D Annapurna. Learning future terrorist targets using attention based hybrid cnn and bilstm model. In *2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS)*, pages 1–5. IEEE, 2023.

[15] Norman P Hummon. Utility and dynamic social networks. *Social networks*, 22(3):221–249, 2000.

[16] Brian Skyrms and Robin Pemantle. A dynamic model of social network formation. In *Adaptive Networks: Theory, Models and Applications*, pages 231–251. Springer, 2009.

[17] Christoph Stadtfeld, James Hollway, and Per Block. Dynamic network actor models: Investigating coordination ties through time. *Sociological Methodology*, 47(1):1–40, 2017.

[18] Skyler J Cranmer and Bruce A Desmarais. Inferential network analysis with exponential random graph models. *Political analysis*, 19(1):66–86, 2011.

[19] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[20] Songgaojun Deng, Huzefa Rangwala, and Yue Ning. Learning dynamic context graphs for predicting social events. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1007–1016, 2019.

[21] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[22] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

[24] Muhan Zhang. Graph neural networks: link prediction. In *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 195–223. Springer, 2022.

[25] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, 2003.

[26] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[27] Michael Mitzenmacher. A brief history of lognormal and power law distributions. In *Proceedings of the Allerton conference on communication, control, and computing*, pages 182–191, 2001.

[28] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[29] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[30] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[31] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015.

[32] Jielun Liu, Ghim Ping Ong, and Xiqun Chen. Graphsage-based traffic speed forecasting for segment network with sparse data. *IEEE Transactions on Intelligent Transportation Systems*, 23(3):1755–1766, 2020.

[33] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 135–144, 2017.

[34] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15*, pages 593–607. Springer, 2018.

[35] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*, 2019.

[36] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 793–803, 2019.

[37] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pages 2704–2710, 2020.

[38] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020*, pages 2331–2341, 2020.

[39] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

[40] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[41] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[42] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019.

[43] Clionadh Raleigh, Rew Linke, Håvard Hegre, and Joakim Karlsen. Introducing acled: An armed conflict location and event dataset. *Journal of peace research*, 47(5):651–660, 2010.