Appendix A
Experiments

A. Datasets

We use six dynamic graph datasets including five publicly available graphs (Emails-DNC (Email), Bitcoin-Alpha (Bitcoin), Wiki-Vote (Wiki), Brain, and GDELT) and a guaranteed-loan network (Guarantee) collected from a major commercial bank. For Email, Bitcoin, Wiki, and Guarantee, we follow a similar setting in CoEvoGNN [62] to obtain the input node attributes. In the **Email** [50], each node represents a person, and the edges indicate email communication between individuals. We use the number of received emails and the number of sent emails for each person as node attributes. **Bitcoin** [27] is a trust network of people using Bitcoin on the Bitcoin Alpha platform, where each node denotes a Bitcoin user, and each weighted edge is associated with a credibility rating assigned by the source node to the target node. We use the average credibility rating as the node attribute for each user. In the **Wiki** [30] network, nodes represent users in the Wikipedia community, and each edge signifies a vote for selecting a Wikipedia administrator. We consider the number of votes received by each person as the node attribute. In the **Guarantee** dataset, nodes correspond to small and medium enterprises, and edges indicate the guarantee relationships. Each edge is associated with a guaranteed loan amount. The node attributes we consider are the total guarantee amount and loan amount for each enterprise. In the **Brain** [67] dataset, the nodes represent the tidy cubes of brain tissue and the edges indicate the connectivity. The attributes are functional magnetic resonance imaging (fMRI) data processed by PCA. **GDELT** [77] is a temporal knowledge graph, where the nodes represent actors and edges represent point-time events. We apply PCA to the CAMEO codes of each actor to generate node attributes.

B. Compared Methods

As discussed in Section I, it is not easy for us to directly extend existing dynamic graph generators (e.g., TagGen [76], Dymond [73], TGGAN [75], and TIGGER [14]) to attributed graph generative models as baselines. Therefore, we primarily focus on comparing the structure generation performance of VRDAG with those methods in experiments. Additionally, we compare our model with several powerful static graph generators including GRAN [33] and GenCAT [36]. GenCAT is the latest state-of-the-art attributed graph generator. Following a similar setting in TagGen, we apply these static methods at each timestep and report the average results in the evaluation. To assess the effectiveness of VRDAG's submodules in the ablation study, we further compare our model against its variants including VRDAG-noBE, VRDAG-noMB, and VRDAG-noGAD, which denotes the VRDAG removes the bi-flow graph encoder, the MixBernoulli sampler, and the graph-based attentional attribute decoder, respectively. The capabilities of different generators are summarized in TABLE V.

TABLE V: The capabilities of different graph generators. ✓ and ✕ indicate whether the generators fulfill or lack the respective characteristics. For example, the **Attribute** indicates whether the generator can simulate node attributes. VRDAG is the only model that can simulate dynamic attributed graphs

Generator	Structure	Attribute	Dynamic
GRAN	✓	✕	✕
GenCAT	✓	✓	✕
TagGen	✓	✕	✓
TGGAN	✓	✕	✓
Dymond	✓	✕	✓
TIGGER	✓	✕	✓
VRDAG	✓	✓	✓

C. Evaluation Metrics

Node attribute metrics. To quantify the attribute generation performance, we first calculate the average attribute distribution discrepancy between the original attribute distribution and the synthetic attribute distribution using Jensen–Shannon divergence (JSD) [11] and Earth Mover's Distance (EMD) [4] across the snapshots in graph sequence. In our experiments, we aggregate the discrepancies of multiple node attributes (if they exist) and report the average value. Furthermore, since node attributes are known to exhibit interdependence, our synthetic graph data should consider the correlation between different attributes in the original snapshots. For simplicity, we focus on evaluating the generated graph with no more than two node attributes. As the commonly used Pearson correlation coefficient is only suitable for linearly correlated data that follows a normal distribution, it is not suitable for evaluating more complex attribute data. Instead, we employ Spearman's correlation coefficient [38] to measure the monotonic relationship between two attributes. The error with respect to Spearman's correlation coefficient R_p is quantified as the mean absolute error, that is, $Mean_{t=1:T}|R_p(X_t^1, X_t^2) - R_p(\tilde{X}_t^1, \tilde{X}_t^2)|$, where X_t^i denotes i-th node attribute value at timestep t.

D. Implementation details

For the proposed VRDAG, we set the layer number of the bi-flow graph encoder to 3. The prior network and posterior network are implemented using two MLPs with LeakyReLU as the activation function, respectively. d_h, d_ε are set to 16. We choose the number of Bernoulli mixture K and dimensionality of latent variable d_z based on parameter analysis experiments for each dataset. We implement our attribute decoder with a 3-layer GAT and an MLP with two hidden layers. The residual technique is applied to mitigate over-smoothing. We adopt the incremental learning strategy [71] that borrows the idea of a truncated version of back-propagation-through-time [64] for scalable and efficient training. Scaling factor α is tuned over $\{1,2,3,4\}$. Our method is trained using the Adam optimizer with an initial learning rate of 5×10^{-2}, which is gradually decreased adaptively during the optimization process. All of our experiments are implemented with PyTorch and trained on a server with two NVIDIA Tesla V100 32GB GPUs. We release our code at https://github.com/Coco-Hut/VRDAG.git.

TABLE VI: Performance evaluation of different variants of VRDAG

Generator	Email							Guarantee						
	In-deg dist	Out-deg dist	Clus dist	NC	JSD	EMD	R_p	In-deg dist	Out-deg dist	Clus dist	NC	JSD	EMD	R_p
VRDAG-noBE	0.0084	0.0081	0.0579	0.6486	0.1335	0.0225	0.3439	0.0087	0.0028	0.0079	0.6495	0.1293	**0.0213**	0.2972
VRDAG-noMB	0.0152	0.0113	0.0598	0.7842	0.3132	0.0279	0.4136	0.0105	0.0094	0.0116	0.8378	0.2738	0.0286	0.4895
VRDAG-noGAD	0.0048	0.0078	0.0414	0.3846	0.2193	0.0154	0.3713	0.0024	0.0041	0.0017	0.1779	0.1193	0.0315	0.4972
VRDAG	**0.0026**	**0.0038**	**0.0312**	**0.1520**	**0.1162**	**0.0138**	**0.1389**	**0.0013**	**0.0015**	**0.0009**	**0.1604**	**0.0874**	0.0237	**0.2440**

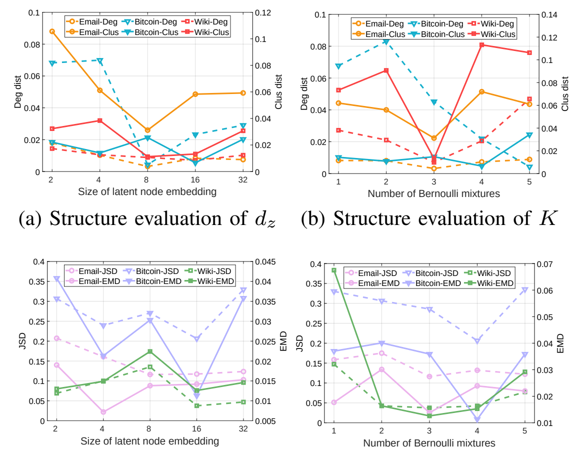
E. Ablation Study

To evaluate the effectiveness of the devised submodules in VRDAG, we conduct a comprehensive ablation study on Email and Guarantee. The results presented in TABLE VI demonstrate that our proposed method outperforms all the variants in terms of both structure and attribute metrics. In particular, removing the bi-flow graph encoder resulted in a significant performance drop in matching in/out-degree distribution and a larger error in Spearman's correlation coefficients on Email. This highlights the importance of the bi-directional message-passing layer in reconstructing the pattern of information flow on the synthetic graph. Meanwhile, we find the degradation in directed edge generation also negatively affects generating correlated attributes, as errors on correlation coefficients increase on both datasets. Notably, VRDAG-noMB has the worst performance among the three variants across all quality measures. This indicates that our Mixture Bernoulli sampling strategy, which is conditioned on the global view of dynamic node embeddings, can effectively model the probability of edge generation. Furthermore, the larger discrepancy in node attribute distribution and the larger error in attribute correlation modeling both suggest a strong correlation between the graph structure and node attributes. We also remove the graph-based attribute decoder and replace it with a vanilla MLP, the variant performs worse than VRDAG in terms of different metrics. This is also because the generation process is not independent. Topological knowledge incorporated by the graph-based decoder helps generate the most likely node attributes, while node attributes serve as a driving force in creating hidden links between node entities.

F. Parameter Analysis

To gain further insights into our framework, we study the impact of two key hyperparameters: latent embedding dimensionality d_z and the number of Bernoulli mixtures K. Fig. 11. presents the experimental results on three benchmark datasets. For structure generation evaluation, we report Deg dist and Clus dist, while for attribute generation evaluation, we use JSD and EMD as quality measures. Specifically, Deg dist represents the averaged discrepancy value of in-degree and out-degree distributions in our experiments.

Dimensionality of latent node representation d_z. The latent node embedding, which preserves the co-evolution pattern of structure and node attributes, is a crucial component in generating new snapshots. We vary the dimensionality of latent variables from 2 to 32. As shown in Fig. 11(a), the structure generation performance initially improves as the embedding size increases, indicating that a stronger node state representation ability leads to better synthesis results. However, the performance becomes stable or even worse as d_z further increases. This is because the embedding space becomes sparse when d_z gets larger. Consequently, there is limited meaningful information available in each dimension of the embedding, with many dimensions being irrelevant or redundant. The attribute generation performance, as shown in Fig. 11(c), follows a similar trend to that of structure generation. In addition, we observe that the EMD metric is more sensitive than the JSD to changes in latent representation size. This difference can also be found in another parameter experiment.



(a) Structure evaluation of d_z (b) Structure evaluation of K

(c) Attribute evaluation of d_z (d) Attribute evaluation of K

Fig. 11: Hyperparameter analysis of VRDAG. (a) and (b) present the changes of Deg dist (left y-axis) and Clus dist (right y-axis) with the influence of latent embedding dimensionality and the number of mixture components. (c) and (d) show the test results in terms of JSD (left y-axis) and EMD (right y-axis) under different settings of two parameters. Deg dist denotes the average value of in/out Deg dist metrics

Number of Bernoulli Mixtures K. According to Fig. 11(b), when $K = 1$, the model performs the worst across different datasets, as the distribution degenerates to Bernoulli, assuming the independence of each potentially generated edge. However, when $K > 1$, the generation of individual edges is not independent due to the latent mixture components, bringing a performance gain in structure generation. As shown in Fig. 11(d), the attribute distribution discrepancy also gradually decreases as the attribute generation is influenced by the generated edges. Additionally, we find that on the large and dense Wiki dataset, the number of Bernoulli mixtures has a more significant impact on the model's performance. Based on the sensitivity experiment, we can select the best hyperparameters for our framework.

APPENDIX B
RELATED WORKS

A. Variational Recurrent Model

Variational recurrent models have been proposed in recent years to model the joint probability distribution over sequence data. Chung et al. [10] first propose VRNN for speech generation and achieve superior performance than traditional dynamic Bayesian networks (DBNs). Inspired by VRNN, Purushotham et al. [46] devise VRADA, which is the first to capture and transfer temporal latent dependencies of multivariate time-series data. After that, Habibie et al. [15] employ a variational recurrent framework to generate high-quality motion without providing frames from an existing sequence. The deep variational recurrent model has been applied to many real-world applications, such as machine translation [55], anomaly detection [42], and dynamic link prediction [17]. We are the first to investigate the problem of dynamic attributed graph generation with the variational recurrent model.