SEMANTIC SCORE ALIGNMENT FOR GRAPH DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph diffusion models have achieved promising performance in graph generation by learning reverse-time denoising processes. Existing approaches typically optimize denoising objectives on noisy node features and edge structures, where the predicted clean graphs implicitly characterize the underlying data distribution. However, conventional denoising supervision mainly focuses on individual reconstruction accuracy and does not explicitly regularize the distributional consistency between real graphs and model-predicted graphs. In this work, we propose semantic score alignment, a distribution-aware regularization framework for graph diffusion models. Motivated by the connection between denoising prediction and data distribution estimation, we introduce an additional score alignment objective to enforce consistency between real graph distributions and predicted graph distributions. Specifically, we first learn a graph-level semantic representation space that captures global structural and attribute information from graphs. The real graphs and predicted graphs are then mapped into this semantic representation space, where their density-induced score functions are estimated and aligned through Fisher divergence. This semantic score alignment provides global semantic guidance for diffusion models beyond conventional reconstruction supervision. Extensive experiments on benchmark graph generation tasks demonstrate that our proposed semantic score alignment consistently improves generation quality across diverse datasets, showing the effectiveness of distribution-level regularization for diffusion-based graph generative modeling.

1 INTRODUCTION

Diffusion models (Ho et al., 2020; Song et al., 2021) have recently emerged as powerful generative frameworks and have been extended to graph generation for modeling complex structures (Jo et al., 2022; 2024; Vignac et al., 2023; Qin et al., 2025).

Existing graph diffusion models mainly follow the denoising score matching paradigm, where noisy node and edge states are progressively recovered through reconstruction objectives (Jo et al., 2022; 2024). While effective, such objectives primarily provide sample-level supervision along individual denoising trajectories, without explicitly constraining whether the predicted clean graphs follow the same underlying distribution and semantic geometry as real graphs.

Existing graph diffusion models are usually trained with denoising objectives (Jo et al., 2022; 2024; Vignac et al., 2023; Qin et al., 2025), where the model learns to predict clean graph states from their noisy counterparts. Although such objectives have been shown effective, they mainly optimize the reconstruction accuracy of individual predictions. From the perspective of generative modeling, however, the ultimate goal of diffusion models is not merely recovering each noisy sample, but learning a generative distribution that approaches the underlying data distribution. Through the Tweedie relationship (Song et al., 2021), the predicted clean sample from a denoising model can be regarded as an implicit estimation of the original data. Thus, improving the quality of denoised predictions requires not only reducing reconstruction errors but also ensuring that the distribution of predicted clean graphs is consistent with that of real graphs.

However, conventional objectives provide limited supervision on such distribution-level consistency. The point-wise regression loss tends to learn conditional expectations of clean samples.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

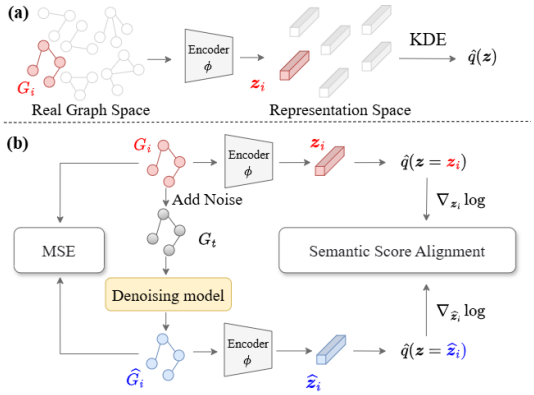


Figure 1: Overview of Semantic Score Alignment (SSA). (a) Given a mini-batch of real graphs, graph-level semantic representations are extracted to construct an empirical latent distribution $\widehat{q}(\boldsymbol{z})$ via KDE, from which the corresponding semantic score field is obtained. (b) For each training sample, both the real graph G_i and its denoised prediction $\widehat{G}_i$ are projected into the same semantic space. Their latent representations are evaluated using the empirical score field constructed in (a), and the resulting semantic scores are aligned to regularize denoising learning.

Consequently, although the predicted graph components can achieve low reconstruction errors, the generated graphs may still deviate from the underlying data manifold due to the absence of distributional constraints.

A natural way to address this issue is to introduce an additional distribution regularization term between real graphs and denoised predictions. Since score functions characterize the local geometry of probability distributions, aligning score fields provides an effective way to measure the discrepancy between two distributions. Specifically, given real graph samples and model-predicted graph samples, one can estimate their empirical distributions and enforce consistency between their corresponding scores through Fisher divergence. Such a distribution-level regularization complements the conventional denoising objective by providing geometric guidance toward the real data manifold.

However, directly applying score alignment in the original graph space is challenging. Graphs are discrete combinatorial objects consisting of node attributes and edge structures, where defining meaningful distances and estimating local densities remain difficult. Although node and edge representations can be compared, such component-wise kernel similarities cannot fully capture the semantic relationships among complete graphs. In contrast, a learned graph representation space can encode high-order structural and semantic information, providing a more suitable metric space for density estimation and manifold modeling Simonovsky & Komodakis (2018); Vahdat et al. (2021).

Motivated by this observation, we propose Semantic Score Alignment (SSA), which performs distribution-level score alignment in a learned semantic graph representation space (Figure 1). Specifically, a graph encoder maps both real graphs and denoised predictions into a unified latent space, where graph-level semantic representations are obtained. Parzen kernel density estimation (KDE) (Parzen, 1962; Davis et al., 2011) is then performed on these latent representations to approximate the local probability distributions, and the corresponding score fields are aligned through Fisher divergence. In this way, SSA introduces a semantic manifold regularization over graph diffusion models, guiding denoising trajectories toward the distribution geometry of real graphs.

Since mini-batch samples provide only limited observations for density estimation, directly estimating KDE over all batch samples may introduce inaccurate density gradients. To obtain more reliable local probability estimation, SSA further adopts a neighborhood-aware KDE strategy by constructing density estimates from nearest neighbors in the latent representation space. This local estimation better captures the underlying manifold geometry and provides more stable score alignment.

Our main contributions are summarized as follows:

- We reveal that most denoising objectives in graph diffusion models provide point-wise supervision, while manifold-level semantic consistency of predicted graphs remains unexplored well.
- We propose Semantic Score Alignment, which introduces a graph semantic manifold regularization by aligning score fields of real and predicted graph manifolds in a learned representation space.
- We develop a local neighborhood-aware density estimation strategy to obtain stable empirical score fields for high-dimensional semantic graph representations.

Extensive experiments on general and molecular graph benchmarks demonstrate that SSA improves structural validity, generation quality, and manifold consistency compared with existing diffusion-based graph generation methods.

2 Related Work

Graph generative models aim to learn complex distributions over discrete structural spaces and generate novel graph instances with reasonable topological properties. With the development of generative learning techniques, graph generation approaches have evolved from autoregressive models You et al. (2018); Reiser et al. (2022), latent variable models Simonovsky & Komodakis (2018); Cao & Kipf (2018); Jin et al. (2018), and flow-based generative models Shi et al. (2020); Eijkelboom et al. (2024) to diffusion-based frameworks Jo et al. (2022); Vignac et al. (2023); Jo et al. (2024); Qin et al. (2025). Among these approaches, diffusion models have demonstrated strong capability in modeling and generating complex graph structures.

Beyond conventional denoising-based diffusion models, recent studies have explored more flexible formulations for graph generation. Diffusion Schrödinger Bridge (DSB) (De Bortoli et al., 2021; Chen et al., 2021) formulates the generation process as a stochastic optimal control problem, where an optimal probability flow is learned under predefined boundary constraints. Diffusion mixture models (Peluchetti, 2023; Wu et al., 2022; Jo & Hwang, 2024) explicitly model intermediate mixture distributions, providing an alternative to conventional reverse denoising prediction. GruM Jo et al. (2024) extends diffusion bridge mechanisms to graph generation through endpoint-conditioned modeling, where structural transitions are represented by weighted graph mixtures, enabling more flexible graph evolution. However, these approaches mainly focus on improving the formulation of the generation process, while the geometric consistency between generated samples and the underlying data manifold remains insufficiently explored.

Score matching provides a principled framework for distribution modeling by directly estimating the gradient of the log-density function without requiring explicit computation of intractable normalization constants. Its extension, denoising score matching (Vincent, 2011), introduces stochastic perturbations to improve the accuracy of score estimation. From early Noise Conditional Score Networks (Song & Ermon, 2019) to continuous-time stochastic differential equation formulations (Song et al., 2021), score-based generative modeling has established a unified framework for learning distribution-dependent vector fields. However, existing score-based graph diffusion models typically learn score functions implicitly in the original discrete graph space, making it difficult for the models to effectively capture the underlying manifold structure of complex graph distributions.

To characterize geometric relationships between data distributions, non-parametric distribution matching methods have been widely studied. Maximum Mean Discrepancy (MMD) (Li et al., 2015; Binkowski et al., 2018) measures distribution discrepancy through kernel-based embeddings, while Wasserstein distance based on Optimal Transport (OT) (Arjovsky et al., 2017; Patrini et al., 2020) aligns distributions by minimizing geometric transport costs. Integral Probability Metrics (IPMs) (Mroueh & Sercu, 2017) further provide a general framework for measuring distribution differences through function-based discrepancy measures. Due to the lack of intrinsic continuous geometric structures and effective distance metrics in discrete graph spaces, directly extending these methods to complex graph generation tasks remains challenging.

To address these limitations, we propose Semantic Score Alignment (SSA), which projects discrete graph structures into a continuous latent representation space (Simonovsky & Komodakis, 2018; Vahdat et al., 2021) through a graph-level representation encoder and performs score field estimation based on this space. The continuous latent space provides a geometric coordinate system, where structurally similar graphs are mapped to nearby positions, enabling score direction estimation.

3 Semantic Score Alignment

3.1 Preliminaries and Motivation

Given samples $\boldsymbol{x}$ drawn from the data distribution $q(\boldsymbol{x})$, diffusion models (Ho et al., 2020; Song et al., 2021) aim to learn a parameterized generative distribution $p_\theta(\boldsymbol{x})$ that approximates $q(\boldsymbol{x})$. From the perspective of information geometry, the generalized de Bruijn identity establishes a fundamental connection between the Kullback-Leibler divergence and the accumulated Fisher divergence along the diffusion trajectory, allowing the discrepancy between $p_\theta(\boldsymbol{x})$ and $q(\boldsymbol{x})$ to be expressed as an integral of score mismatching over all diffusion timesteps. However, directly optimizing this continuous-time distribution matching objective is generally intractable. Instead, diffusion models are trained via

denoising score matching (DSM) (Vincent, 2011), which provides a tractable surrogate for score estimation. Under Gaussian perturbations, the Tweedie formula (Song et al., 2021) further shows that score estimation is equivalent to predicting the clean sample from its noisy observation, leading to the widely adopted reconstruction objective:

$$\mathcal{L}_{\text{kl}} = D_{\text{kl}}(q(\boldsymbol{x})\|p_\theta(\boldsymbol{x})) = \frac{1}{2}\int_0^T g_t^2 D_f\big(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)\big)\mathrm{d}t \equiv \int_0^T w_t \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{x}_t)}[\|\boldsymbol{x}-\widehat{\boldsymbol{x}}\|_2^2]\mathrm{d}t \quad (1)$$

where D_{kl} denotes the KL divergence, g_t is the diffusion coefficient, D_f represents the Fisher divergence, and w_t is the time-dependent weighting function.

Under the reconstruction objective, the optimal prediction is the conditional expectation, $\widehat{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x}|\boldsymbol{x}_t]$, which provides only point-wise supervision for individual samples. Although such a regression objective effectively improves reconstruction accuracy, it does not explicitly constrain whether the collection of predicted samples follows the underlying data manifold. To introduce distribution-level supervision, we consider the reverse transition at an arbitrary diffusion step. Given the current noisy state $\boldsymbol{x}_{t+1}$, the next reverse state is generated through the same conditional transition kernel, $q(\boldsymbol{x}_t|\boldsymbol{x}_{t+1},\boldsymbol{x})$, where $\boldsymbol{x}$ denotes either a real clean sample or the model prediction $\widehat{\boldsymbol{x}}$. Consequently, both the real trajectory and the generated trajectory evolve under the same reverse kernel: $\boldsymbol{x} \to q(\boldsymbol{x}_t|\boldsymbol{x}_{t+1},\boldsymbol{x})$, $\widehat{\boldsymbol{x}} \to q(\boldsymbol{x}_t|\boldsymbol{x}_{t+1},\widehat{\boldsymbol{x}})$. Since the reverse conditional kernel is shared, the marginal distribution at time t is completely determined by the distribution of the clean endpoint. In other words, propagating the predicted clean-data distribution $p_\theta(\boldsymbol{x})$ through the reverse kernel produces the same marginal distribution $p_\theta(\boldsymbol{x}_t)$ as directly marginalizing over the predicted samples. Thus, reducing the discrepancy between the clean distributions, $p_\theta(\boldsymbol{x}) \approx q(\boldsymbol{x})$, naturally regularizes the generated marginals $p_\theta(\boldsymbol{x}_t) \approx q(\boldsymbol{x}_t)$ throughout the reverse diffusion process. The derivation of the above theory can be found in Appendix A.

Motivated by this observation, instead of explicitly matching distributions at every diffusion step, we regularize the empirical clean distributions. Specifically, we estimate the empirical distributions of real samples $\widehat{q}(\boldsymbol{x})$ and predicted samples $\widehat{p}_\theta(\boldsymbol{x})$, and align their score fields via Fisher divergence. This distribution-level regularization complements the conventional denoising objective by encouraging the predicted clean samples to better align with the underlying manifold of the real data.

3.2 Graph Semantic Representation

As discussed in §3.1, Semantic Score Alignment requires estimating the empirical distributions of real samples and model predictions. A straightforward solution is to perform KDE directly on graphs. However, unlike Euclidean data, graphs are discrete combinatorial structures consisting of node attributes and edge connectivity, making it inherently difficult to define meaningful distances and local neighborhoods. Consequently, KDE in the original graph space cannot reliably characterize the underlying semantic distribution. For graph data, however, defining such similarities in the original discrete graph space is highly nontrivial.

To obtain a meaningful metric space for density estimation, we first map graphs into a continuous semantic representation space. Specifically, given a graph $G = (X, A)$, where X denotes node features and A represents edge connectivity and types. We learn its graph-level representation through a graph encoder $\boldsymbol{z} = \phi(G)$, where $\phi(\cdot)$ jointly encodes node features and graph topology into a continuous semantic embedding. Specifically, we adopt the graph convolutional network (GCN) (Kipf & Welling, 2017) as the graph encoder. GCN is naturally suited for modeling molecular graphs, as molecular properties are inherently governed by structural connectivity. Likewise, the denoised prediction $\widehat{G} = (\widehat{X}, \widehat{A})$ is mapped into the same latent space, $\widehat{\boldsymbol{z}} = \phi(\widehat{G})$. The learned semantic space preserves graph-level structural similarity while providing a continuous metric space, making it suitable for subsequent KDE and semantic score alignment.

3.3 Semantic Score Estimation

Since the analytical form of the semantic distribution $q(\boldsymbol{z})$ is unavailable, we estimate it non-parametrically from each mini-batch. Let $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{n_b}\}$ denote the graph semantic representations of all real graphs in the current mini-batch. The empirical semantic distribution is estimated

by KDE:

$$\widehat{q}(\boldsymbol{z}) = \frac{1}{|\mathcal{Z}|} \sum_{\boldsymbol{z}_i \in \mathcal{Z}} k(\boldsymbol{z}, \boldsymbol{z}_i), \tag{2}$$

where $k(\cdot, \cdot)$ denotes a Gaussian kernel.

The corresponding score field is obtained from the gradient of the log-density, $\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}) = \frac{\nabla_{\boldsymbol{z}} \widehat{q}(\boldsymbol{z})}{\widehat{q}(\boldsymbol{z})}$. For Gaussian kernels, this score can be written as

$$\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}) \propto \boldsymbol{\mu}(\boldsymbol{z}) - \boldsymbol{z}, \tag{3}$$

where $\boldsymbol{\mu}(\boldsymbol{z}) = \frac{\sum_{\boldsymbol{z}_i \in \mathcal{Z}} k(\boldsymbol{z}, \boldsymbol{z}_i) \boldsymbol{z}_i}{\sum_{\boldsymbol{z}_i \in \mathcal{Z}} k(\boldsymbol{z}, \boldsymbol{z}_i)}$. The vector $\boldsymbol{\mu}(\boldsymbol{z}) - \boldsymbol{z}$ therefore points toward the local high-density region of the semantic manifold and provides an empirical approximation of the underlying score field. Proofs are given in Appendix B.

Given a real graph representation $\boldsymbol{z}_i$ and the corresponding predicted representation $\widehat{\boldsymbol{z}}_i$, both are evaluated on the same empirical semantic distribution $\widehat{q}(\boldsymbol{z})$ to obtain their semantic scores, $\boldsymbol{s}_j = \nabla_{\boldsymbol{z}_j} \log \widehat{q}(\boldsymbol{z} = \boldsymbol{z}_j), \widehat{\boldsymbol{s}}_j = \nabla_{\widehat{\boldsymbol{z}}_j} \log \widehat{q}(\boldsymbol{z} = \widehat{\boldsymbol{z}}_j)$. The proposed Semantic Score Alignment regularizes the denoising model by minimizing the discrepancy between the two empirical score fields, $\|\boldsymbol{s}_j - \widehat{\boldsymbol{s}}_j\|_2^2$. In practice, estimating KDE from all batch samples may introduce noisy density estimates in high-dimensional semantic spaces. Therefore, we construct the local KDE using only the K-nearest neighbors of each query point, yielding a more reliable approximation of the local semantic manifold while preserving the same formulation above. Accordingly, the support set $\mathcal{Z}$ in Eq. 2 is replaced by its local neighborhood $\mathcal{Z}_K(\boldsymbol{z}) = N_K(\boldsymbol{z}) \cap \mathcal{Z}$, where $N_K(\boldsymbol{z})$ denotes the K-nearest neighbors of the query $\boldsymbol{z}$ in the current mini-batch.

3.4 Overall SSA Training

The training procedure is summarized in Algorithm 1. Based on the Fisher divergence objective, we define the proposed Semantic Score Alignment (SSA) loss by matching the empirical semantic scores of real graphs and their corresponding denoised predictions. Specifically, the semantic score of each sample is estimated from the empirical KDE distribution in the shared latent space, and the diffusion model is regularized by minimizing:

$$\mathcal{L}_{\text{ssa}} = \frac{1}{n_b} \sum_{j=1}^{n_b} \Big\| \text{sg}\left(\boldsymbol{\mu}(\widehat{\boldsymbol{z}}_j)\right) - \widehat{\boldsymbol{z}}_j - \left(\text{sg}\left(\boldsymbol{\mu}(\boldsymbol{z}_j)\right) - \boldsymbol{z}_j\right) \Big\|_2^2, \tag{4}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator.

The proposed Semantic Score Alignment serves as a plug-in regularization for graph diffusion models. Combined with the conventional denoising reconstruction objective, the overall training loss is

$$\mathcal{L} = \mathcal{L}_{\text{mse}} + \lambda \mathcal{L}_{\text{ssa}}, \tag{5}$$

where $\mathcal{L}_{\text{mse}}$ denotes the original denoising reconstruction loss, $\mathcal{L}_{\text{ssa}}$ regularizes the semantic score fields of real and predicted graphs, and λ balances $\mathcal{L}_{\text{mse}}$ and $\mathcal{L}_{\text{ssa}}$.

During training, both the real graph and its denoised prediction are projected into the shared semantic representation space. Their semantic scores are estimated from the empirical KDE distribution constructed on the current mini-batch, and the resulting score discrepancy is minimized jointly with the conventional reconstruction objective.

Algorithm 1: SSA Training Algorithm.

Input: Training set $\mathcal{D}$ and parameter λ.
Initialize: Diffusion model parameters $\{\theta, \phi\}$.
for $t \sim U(0, T)$ **do**
  Sample $\{G_j\}_{j=1}^{n_b} \sim \mathcal{D}$ where $G_j = (X_j, A_j)$;
  for $j \in [1, n_b]$ **do**
    $X_t \sim q(X_t | X_j; \boldsymbol{\epsilon}_x), \boldsymbol{\epsilon}_x \sim \mathcal{N}(0, I)$;
    $A_t \sim q(A_t | A_j; \boldsymbol{\epsilon}_a), \boldsymbol{\epsilon}_a \sim \mathcal{N}(0, I)$;
    $\widehat{G}_j = f_\theta(G_t, t)$ where $G_t = (X_t, A_t)$;
    $\boldsymbol{z}_j = \phi(G_j), \quad \widehat{\boldsymbol{z}}_j = \phi(\widehat{G}_j)$;
  end for
  $\mathcal{L}_{\text{mse}} = \frac{1}{n_b} \sum_{j=1}^{n_b} \left(\|X_j - \widehat{X}_j\|_2^2 + \|A_j - \widehat{A}_j\|_2^2\right)$;
  Calculate SSA loss by Eq. 4;
  Calculate overall loss: $\mathcal{L} = \mathcal{L}_{\text{mse}} + \lambda \mathcal{L}_{\text{ssa}}$;
  Update $\{\theta, \phi\}$ by minimizing $\mathcal{L}$;
end for
Output: Optimized parameters $\theta^\star = \theta$.

4 EXPERIMENTS

To evaluate the generality of SSA, we integrate it into three representative graph diffusion models, including GDSS Jo et al. (2022), DiGress Vignac et al. (2023), and GruM Jo et al. (2024), and evaluate them on the QM9 benchmark. As reported in Table 1, SSA consistently improves all three backbones, demonstrating that it serves as a general distribution-level regularization rather than being tailored to a specific diffusion architecture. Since GruM equipped with SSA achieves the best overall performance, we adopt it as the default backbone in all subsequent experiments.

Table 1: SSA performance of three different models.

Model	Method	Valid (%) ↑	FCD ↓	NSPDK ↓	Scaf. ↑
GDSS	Baseline	95.72	2.9000	0.0033	0.6983
	SSA	97.15	2.5572	0.0026	0.7246
DiGress	Baseline	99.59	0.3000	0.0006	0.9267
	SSA	99.63	0.2317	0.0004	0.9329
GruM	Baseline	99.79	0.1238	0.0003	0.9357
	SSA	99.79	0.1090	0.0002	0.9412

4.1 EXPERIMENTAL SETTING

To perform Semantic Score Alignment, we introduce a graph encoder that maps graphs into a continuous semantic representation space. Specifically, the encoder consists of a GCN-based backbone (Kipf & Welling, 2017) that aggregates node and edge information through message passing, followed by masked global average pooling to obtain a permutation-invariant graph-level representation. A lightweight MLP projection head further maps the graph representation into a compact low-dimensional latent space, providing a suitable and effective representation space for semantic score estimation. The parameters ϕ of the graph encoder are jointly optimized with the diffusion model parameters θ in an end-to-end manner during training.

For graph denoising, we adopt the Graph Transformer architecture of GruM Jo et al. (2024) as the backbone diffusion model. Unless otherwise specified, all optimization and sampling settings follow the original implementation, including the AdamW optimizer, exponential moving average (EMA), and the Euler-Maruyama predictor-corrector sampler.

Additional details, including the graph encoder architecture, optimization settings, hyperparameter analysis, and dataset-specific configurations that consist of the neighborhood size K and the SSA weight λ are provided in Appendix D.

4.2 MOLECULAR GRAPH GENERATION

Datasets and Metrics. We evaluate SSA on two benchmarks: QM9 Ramakrishnan et al. (2014) and ZINC250k Irwin et al. (2012). Detailed descriptions are provided in Appendix F. To evaluate the quality of generated molecules, we adopt four complementary metrics. Validity measures the proportion of generated molecules that satisfy chemical valence constraints without any post-processing correction. Fréchet ChemNet Distance (FCD) (Preuer et al., 2018) evaluates the distributional discrepancy between generated and reference molecules in the learned chemical feature space. NSPDK MMD (Costa & Grave, 2010) assesses the structural similarity between generated and real molecular graphs through neighborhood subgraph kernel statistics. Scaffold Similarity (Scaf.) quantifies retention of core molecular scaffolds and functional substructures in generated molecules.

Baselines. To comprehensively evaluate our approach, we compare it against a diverse set of representative molecular graph generation models, including MoFlow Zang & Wang (2020), GraphAF Shi et al. (2020), GraphDF Luo et al. (2021), EDP-GNN Niu et al. (2020), GDSS, DiGress, GraphARM Kong et al. (2023), GruM, DeFoG Qin et al. (2025), and HOG-Diff Huang & Birdal (2026). For GruM, we obtain the reported results through our own implementations using the same experimental settings and evaluation protocols as our method, ensuring a fair comparison.

Table 2: Comparison of SSA with baselines on QM9 and ZINC250k.

Method	QM9				ZINC250k			
	Real, $\|V\| \leq 9$				Synthetic, $\|V\| \leq 38$			
	Valid (%)↑	FCD ↓	NSPDK ↓	Scaf. ↑	Valid (%)↑	FCD ↓	NSPDK ↓	Scaf. ↑
Training set	100.0	0.0398	0.0001	0.9719	100.0	0.0615	0.0001	0.8395
MoFlow	91.36	4.4670	0.0169	0.1447	63.11	20.931	0.0455	0.0133
GraphAF	74.43	5.6250	0.0207	0.3046	68.47	16.023	0.0442	0.0672
GraphDF	93.88	10.928	0.0636	0.0978	90.61	33.546	0.1770	0.0000
EDP-GNN	47.52	2.6800	0.0046	0.3270	82.97	16.737	0.0485	0.0000
GDSS	95.72	2.9000	0.0033	0.6983	97.01	14.656	0.0195	0.0467
DiGress	99.59	0.3000	0.0006	0.9267	94.99	3.4820	0.0021	0.4163
GraphARM	90.25	1.2200	0.0020	-	88.23	16.260	0.0550	-
GruM	99.79	0.1238	0.0003	0.9357	98.65	2.2570	0.0015	0.5299
DeFoG	97.12	0.1135	0.0004	0.9408	99.22	2.1230	0.0022	0.4245
HOG-Diff	98.74	0.1720	0.0003	-	98.56	1.6330	0.0010	-
SSA	99.79	0.1090	0.0002	0.9412	99.40	1.5660	0.0009	0.5629

Results. The experimental results are summarized in Table 2. The results demonstrate that SSA achieves superior molecular generation performance on QM9. Specifically, SSA obtains a Validity score of 99.79%, which is comparable to the best-performing method, indicating its capability to consistently generate chemically valid molecules. Meanwhile, SSA achieves the best results in terms of FCD (0.1090) and NSPDK (0.0002), suggesting that the generated molecules exhibit closer chemical distribution and topological characteristics to the real data. SSA also achieves the highest Scaf. score (0.9412), demonstrating the effectiveness of real-semantic score alignment in preserving essential molecular scaffolds. On ZINC250k, SSA maintains consistent advantages, achieving a Validity score of 99.40% and obtaining the best performance in FCD (1.566), NSPDK (0.0009), and Scaf. (0.5629). The generated molecular graph visualizations are provided in Appendix G.

4.3 General Graph Generation

Datasets and Metrics. We evaluate SSA on six benchmark datasets: Planar Dobson & Doig (2003); Vignac et al. (2023), Stochastic Block Model (SBM) Dobson & Doig (2003); Vignac et al. (2023), Ego-small Sen et al. (2008), Community-small, Enzymes Schomburg et al. (2004), and Grid.

Planar consists of planar graphs satisfying non-crossing edge constraints. SBM contains graphs generated by stochastic block models with community structures. Ego-small contains ego networks extracted from the Citeseer citation graph. Community-small contains graphs with community-based topologies. Enzymes contains protein structure graphs from the BRENDA database, Grid contains regular two-dimensional lattice graphs. Detailed dataset statistics are provided in Appendix F.

To evaluate the quality of generated graphs, we compute the Maximum Mean Discrepancy (MMD) (You et al., 2018) between the distributions of graph statistics from generated and test graphs. For Planar and SBM, we measure degree distribution (Deg.), clustering coefficient (Clus.), 4-node orbit counts (Orbit), and graph Laplacian spectrum (Spec.). For Ego-small, Community-small, Enzymes, and Grid, we follow previous works and evaluate generation performance using degree distribution (Deg.) and 4-node orbit counts (Orbit).

Baselines. For the Planar and SBM benchmarks, we compare against a diverse and widely used set of representative graph generative models, including GraphRNN You et al. (2018), GRAN Liao et al. (2019), SPECTRE Martinkus et al. (2022), EDP-GNN, GDSS, ConGress Vignac et al. (2023), DiGress, GruM, and DeFoG. For GruM and DeFoG, we conduct our own reproduction under the same experimental environment and evaluation protocol as our method, using identical evaluation scripts to ensure a fair and consistent comparison.

For the Ego-small, Community-small, Enzymes, and Grid benchmarks, we compare with DeepGMG Li et al. (2018), GraphRNN, GraphAF Shi et al. (2020), GraphDF Luo et al. (2021),

Table 3: Comparison of SSA with baselines on Planar and SBM.

Method	Planar				SBM			
	Synthetic, $\|V\| = 64$				Synthetic, $44 \leq \|V\| \leq 187$			
	Deg. ↓	Clus. ↓	Orbit ↓	Spec. ↓	Deg. ↓	Clus. ↓	Orbit ↓	Spec. ↓
Training set	0.0002	0.0310	0.0005	0.0052	0.0008	0.0332	0.0255	0.0063
GraphRNN	0.0049	0.2779	1.2543	0.0459	0.0055	0.0584	0.0785	0.0065
GRAN	0.0007	0.0426	0.0009	0.0075	0.0113	0.0553	0.0540	0.0054
SPECTRE	0.0005	0.0785	0.0012	0.0112	0.0015	0.0521	0.0412	0.0056
EDP-GNN	0.0044	0.3187	1.4986	0.0813	0.0011	0.0552	0.0520	0.0070
GDSS	0.0041	0.2676	0.1720	0.0370	0.0212	0.0646	0.0894	0.0128
ConGress	0.0048	0.2728	1.2950	0.0418	0.0273	0.1029	0.1148	–
DiGress	0.0003	0.0372	0.0009	0.0106	0.0013	0.0498	0.0434	0.0400
GruM	0.0007	0.0489	0.0048	0.0075	0.0007	0.0492	0.0448	0.0050
DeFoG	0.0005	0.0501	0.0006	0.0072	0.0006	0.0517	0.0556	0.0054
SSA	0.0003	0.0241	0.0040	0.0061	0.0003	0.0488	0.0369	0.0036

GraphVAE Simonovsky & Komodakis (2018), GNF Liu et al. (2019), EDP-GNN, GDSS, GruM, GraphARM Kong et al. (2023), DiGress and HOG-Diff. For GDSS and GruM, we report results from our own reproduction under the same experimental environment and evaluation scripts as our method to ensure a fair and consistent comparison.

Results. The experimental results are summarized in Tables 3 and 4. SSA achieves competitive performance across structural statistics on both synthetic and real graph datasets. On Planar, SSA achieves the best clustering coefficient distribution while maintaining competitive degree and spectral statistics. On SBM, SSA achieves the best degree and spectral distribution matching, demonstrating its ability to provide structural guidance during generation. For Ego-small, Community-small, Enzymes, and Grid, SSA consistently achieves lower degree distribution discrepancies than existing methods, indicating its ability to preserve key connectivity patterns and structural regularities. Generated graph visualizations are provided in Appendix G.

4.4 Ablation Studies

We investigate whether the empirical semantic distribution should be estimated from the entire mini-batch or a local neighborhood. Specifically, SSA (KDE) constructs the KDE using all samples in the current mini-batch, whereas SSA (KDE + $N_K(\boldsymbol{z})$) restricts the support set to the K-nearest neighbors of each query embedding. The two variants differ only in the support set used for density estimation. As shown in Table 5, local KDE estimation consistently outperforms global KDE on both QM9 and Planar. This indicates that semantic score estimation benefits from local neighborhoods, which provide a more faithful approximation of the underlying manifold while reducing the influence of irrelevant samples in high-dimensional latent spaces.

We further investigate which empirical distribution should be used to estimate the semantic score of generated samples. In SSA, both the real embeddings and the generated embeddings query the same empirical distribution constructed from real graph embeddings. In contrast, the ablation test estimates the semantic score of generated embeddings from a separate KDE constructed on the generated graph embeddings, while the real embeddings are still evaluated on the real empirical distribution. All other training settings remain identical.

As shown in Table 6, using the real empirical distribution to estimate the semantic scores of generated samples consistently achieves better performance than estimating them from the generated distribution itself. This suggests that the real data manifold provides a more reliable and stable semantic score field, whereas the generated manifold may introduce inaccurate score estimates due to imperfect samples, especially during the early stages of training. Consequently, anchoring generated samples to the real semantic score field provides more stable distributional supervision and leads to consistently better generation quality. These results indicate that generated samples should be aligned to the

Table 4: Comparison of SSA with baselines on Ego-small, Community-small, Enzymes, and Grid.

Method	Ego-small		Community-small		Enzymes		Grid	
	Real		Synthetic		Real		Synthetic	
	Deg. ↓	Orbit ↓	Deg. ↓	Orbit ↓	Deg. ↓	Orbit ↓	Deg. ↓	Orbit ↓
DeepGMG	0.0400	0.0200	0.2200	0.4000	–	–	–	–
GraphRNN	0.0900	0.0030	0.0800	0.0400	0.0170	0.0460	0.0640	0.0210
GraphAF	0.0300	0.0010	0.0180	0.0200	1.6690	0.2660	–	–
GraphDF	0.0400	0.0100	0.0600	0.0300	1.5030	0.2020	–	–
GraphVAE	0.1300	0.0500	0.3500	0.5400	1.3690	0.1910	1.6190	0.9190
GNF	0.0300	0.0010	0.2000	0.1100	–	–	–	–
EDP-GNN	0.0520	0.0070	0.0530	0.0260	0.0230	0.0820	0.4550	0.3280
GDSS	0.0259	0.0133	0.1035	0.0365	0.0387	0.0135	0.2071	0.2737
DiGress	0.0210	0.0240	0.0470	0.0260	0.0130	0.0110	-	-
GraphARM	0.0190	0.0100	0.0340	0.0040	0.0290	0.0150	-	-
GruM	0.0081	0.0415	0.0175	0.0431	0.0117	0.0122	0.0063	0.0612
HOG-Diff	0.0150	0.0040	0.0060	0.0020	0.0110	0.0070	-	-
SSA	0.0011	0.0217	0.0041	0.0170	0.0007	0.0035	0.0001	0.0558

Table 5: Ablation study of KDE on QM9 and Planar.

Method	QM9				Planar			
	Valid (%)↑	FCD↓	NSPDK↓	Scaf.↑	Deg.↓	Clus.↓	Orbit↓	Spec.↓
w/o SSA	99.79	0.1238	0.0003	0.9357	0.0007	0.0489	0.0048	0.0075
SSA (KDE)	99.68	0.1205	0.0003	0.9351	0.0009	0.0589	0.0080	0.0068
SSA (KDE + $N_K(\boldsymbol{z})$)	99.79	0.1090	0.0002	0.9412	0.0003	0.0241	0.0040	0.0061

Table 6: Effect of shared and separate KDE for semantic score estimation on QM9 and Planar.

Method	QM9				Planar			
	Valid (%)↑	FCD↓	NSPDK↓	Scaf.↑	Deg.↓	Clus.↓	Orbit↓	Spec.↓
Separate KDE	99.63	0.1347	0.0004	0.9370	0.0007	0.0428	0.0055	0.0074
Shared KDE (SSA)	99.79	0.1090	0.0002	0.9412	0.0003	0.0241	0.0040	0.0061

semantic score field of the real data manifold rather than their own evolving distribution, which serves as a more stable teacher throughout training.

5 Conclusion

In this paper, we presented SSA, a distribution-level regularization framework for graph diffusion models. Unlike existing methods that rely solely on point-wise denoising supervision, SSA introduces semantic score alignment to explicitly regularize the empirical graph distribution during training. The key idea is to estimate the semantic score field from the empirical distribution of graph-level representations and align the score directions of real and generated graphs through a Fisher-divergence-inspired objective. By performing score estimation in a continuous semantic representation space, SSA effectively overcomes the difficulty of measuring similarities directly in the discrete graph domain while preserving the global structural semantics of graphs.

From a broader perspective, our study suggests that graph diffusion models can benefit not only from improving denoising accuracy, but also from explicitly constraining the underlying geometry of the learned data distribution. We hope this work provides a simple yet effective perspective for incorporating explicit distribution-level semantic regularization into diffusion-based graph generation, and inspires future research on further combining representation learning, non-parametric distribution estimation, and score-based generative modeling.

AI USE STATEMENT

During the preparation of this manuscript, generative AI tools were used primarily as aids for language polishing. All AI-assisted content was thoroughly reviewed and appropriately revised by the authors, who take full responsibility for the accuracy, completeness, and academic integrity of the final manuscript.

ETHICS STATEMENT

All experiments in this study were conducted using publicly available datasets and established standard evaluation protocols. No human participants, personally identifiable information, or other sensitive personal data were involved. All datasets and evaluation resources were used in accordance with their respective licenses, attribution requirements, and applicable regulations. This work strictly follows the applicable ethical guidelines and requirements of the conference.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, we provide key implementation details, including training configurations, hyperparameter settings, datasets, and evaluation procedures. To further facilitate reproducibility, the source code, relevant configuration files, and evaluation scripts will be released upon acceptance or with the camera-ready submission.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, volume 70, pp. 214–223, 2017.

Mikolaj Binkowski, Danica J. Sutherland, Michal Arbel, and Arthur Gretton. Demystifying MMD GANs. In *ICLR*, 2018.

Nicola De Cao and Thomas Kipf. MolGAN: An implicit generative model for small molecular graphs. *ArXiv*, abs/1805.11973, 2018.

Tian Qi Chen, Guan-Horng Liu, and Evangelos A. Theodorou. Likelihood training of Schrödinger bridge using forward-backward SDEs theory. In *ICLR*, 2021.

Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *ICML*, 2010.

Richard A Davis, Keh-Shin Lii, and Dimitris N Politis. Remarks on some nonparametric estimates of a density function. In *Selected Works of Murray Rosenblatt*, pp. 95–100, 2011.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion Schrödinger bridge with applications to score-based generative modeling. In *NeurIPS*, volume 34, pp. 17695–17709, 2021.

Paul D. Dobson and Andrew J. Doig. Distinguishing Enzyme structures from non-enzymes without alignments. *Journal of Molecular Biology*, 330 4:771–783, 2003.

Floor Eijkelboom, Grigory Bartosh, Christian A. Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. In *NeurIPS*, volume 37, pp. 11735–11764, 2024. doi: 10.52202/079017-0374.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pp. 6840–6851, 2020.

Yiming Huang and Tolga Birdal. HOG-Diff: Higher-order guided diffusion for graph generation. In *ICLR*, volume 14, 2026.

John J. Irwin, T. Sterling, Michael M. Mysinger, Erin S. Bolstad, and Ryan G. Coleman. ZINC: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52: 1757 – 1768, 2012.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *ICML*, volume 80, pp. 2323–2332, 2018.

Jaehyeong Jo and Sung Ju Hwang. Generative modeling on manifolds through mixture of Riemannian diffusion processes. In *ICML*, volume 235, pp. 22348–22370, 2024.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *ICML*, volume 162, pp. 10362–10383, 2022.

Jaehyeong Jo, Dongki Kim, and Sung Ju Hwang. Graph generation with diffusion mixture. In *ICML*, volume 235, pp. 22371–22405, 2024.

Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Lingkai Kong, Jiaming Cui, Haotian Sun, Yuchen Zhuang, B. Aditya Prakash, and Chao Zhang. Autoregressive diffusion model for graph generation. In *ICML*, volume 202, pp. 17391–17408, 2023.

Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *ICML*, volume 37, pp. 1718–1727, 2015.

Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter W. Battaglia. Learning deep generative models of graphs. In *ICLR*, 2018.

Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. In *NeurIPS*, volume 32, 2019.

Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, and Kevin Swersky. Graph normalizing flows. In *NeurIPS*, volume 32, 2019.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. Graphdf: A discrete flow model for molecular graph generation. In *ICML*, volume 139, pp. 7192–7203, 2021.

Karolis Martinkus, Andreas Loukas, Nathanaël Perraudin, and Roger Wattenhofer. SPECTRE: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In *ICML*, volume 162, pp. 15159–15179, 2022.

Youssef Mroueh and Tom Sercu. Fisher GAN. In *NeurIPS*, volume 30, 2017.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *AISTATS*, volume 108, pp. 4474–4484, 2020.

Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

Giorgio Patrini, Rianne van den Berg, Patrick Forré, Marcello Carioni, Samarth Bhargav, Max Welling, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. In *UAI*, volume 115, pp. 733–743, 2020.

Stefano Peluchetti. Non-denoising forward-time diffusions. *ArXiv*, abs/2312.14589, 2023.

Kristina Preuer, Philipp Renz, Thomas Unterthiner, Sepp Hochreiter, and Günter Klambauer. Fréchet ChemNet distance: A metric for generative models for molecules in drug discovery. *Journal of Chemical Information and Modeling*, 58 9:1736–1741, 2018.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *ICML*, volume 267, pp. 50269–50326, 2025.

Raghunathan Ramakrishnan, Pavlo O. Dral, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

Patrick Reiser, Marlen Neubert, Andr'e Eberhard, et al. Graph neural networks for materials science and chemistry. *Communications Materials*, 3, 2022.

Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. Brenda, the enzyme database: updates and major new developments. *Nucleic Acids Research*, 32:D431–3, 2004.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *The AI Magazine*, 2008.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. GraphAF: a flow-based autoregressive model for molecular graph generation. In *ICLR*, 2020.

Martin Simonovsky and Nikos Komodakis. GraphVAE: Towards generation of small graphs using variational autoencoders. In *ICANN*, pp. 412–422, 2018.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, volume 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021.

Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *NeurIPS*, volume 34, pp. 11287–11302, 2021.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *ICLR*, volume 11, pp. 1–13, 2023.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23:1661–1674, 2011.

Lemeng Wu, Chengyue Gong, Xingchao Liu, Mao Ye, and Qiang Liu. Diffusion-based molecule generation with informative prior bridges. In *NeurIPS*, volume 35, pp. 36533–36545, 2022.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. GraphRNN: Generating realistic graphs with deep auto-regressive models. In *ICML*, volume 80, pp. 5708–5717, 2018.

Chengxi Zang and Fei Wang. Moflow: An invertible flow model for generating molecular graphs. In *KDD*, pp. 617–626, 2020.

Appendix

This supplementary material provides additional information to support the main text. The contents are organized as follows:

§A Theoretical Foundations and Mathematical Derivation of SSA

§B Derivation of Empirical Score Function via Kernel Density Estimation

§C Detailed network architecture

§D Detailed experimental setting

§E Hyperparameter and Stability Analysis

§F Datasets Description

§G Graph Generation Visualization

A Theoretical Foundations and Mathematical Derivation of SSA

A.1 Preliminaries and Mathematical Formulation

Score-based diffusion generative models aim to learn a parameterized generative distribution $p_\theta(\boldsymbol{x}_0)$ that approximates the real data distribution $q(\boldsymbol{x}_0)$. Ideally, the generation process can be formulated as minimizing the Kullback-Leibler (KL) divergence between the two distributions:

$$\mathcal{L}_{\text{kl}} = D_{\text{kl}}(q(\boldsymbol{x}_0)\|p_\theta(\boldsymbol{x}_0)) = \int q(\boldsymbol{x}_0)\log\frac{q(\boldsymbol{x}_0)}{p_\theta(\boldsymbol{x}_0)}\mathrm{d}\boldsymbol{x}_0. \tag{6}$$

However, since both the real data distribution $q(\boldsymbol{x}_0)$ and the model distribution $p_\theta(\boldsymbol{x}_0)$ are not explicitly available, this objective is generally difficult to optimize directly. Score-based diffusion models introduce a continuous-time diffusion process and transform the distribution matching problem into a score field estimation problem along the diffusion trajectory.

Consider the following continuous-time stochastic differential equation (SDE):

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}, \tag{7}$$

where $\boldsymbol{f}(\boldsymbol{x}, t)$ denotes the drift term, $g(t)$ represents the scalar diffusion coefficient, and $\boldsymbol{w}$ is a standard Wiener process vector.

Although the KL divergence provides a principled criterion for measuring the discrepancy between the real and generated distributions, directly optimizing this objective is infeasible due to the unknown probability densities. The key insight of score-based diffusion models is that the evolution of distributions under a continuous diffusion process can be characterized by their corresponding score fields. Therefore, instead of explicitly estimating the data distributions, we can measure their discrepancy through the difference between score functions along the diffusion trajectory. In the following, we establish the theoretical relationship between the original KL distribution matching objective and the trajectory-wise Fisher score alignment induced by the diffusion dynamics.

A.2 Generalized De Bruijn Identity from the Perspective of Diffusion Dynamics

To establish this connection, we consider the evolution of the real and generated distributions under the same forward diffusion process. Let $q(\boldsymbol{x}_t)$ and $p_\theta(\boldsymbol{x}_t)$ denote the corresponding marginal distributions at diffusion time t. Since the temporal evolution of probability densities governed by the SDE can be described by the Fokker-Planck equation, the dynamics of these two distributions satisfy:

$$\frac{\partial q(\boldsymbol{x}_t)}{\partial t} = -\nabla_{\boldsymbol{x}}\cdot\big(\boldsymbol{f}(\boldsymbol{x}, t)q(\boldsymbol{x}_t)\big) + \frac{1}{2}g(t)^2\Delta q(\boldsymbol{x}_t), \tag{8}$$

$$\frac{\partial p_\theta(\boldsymbol{x}_t)}{\partial t} = -\nabla_{\boldsymbol{x}}\cdot\big(\boldsymbol{f}(\boldsymbol{x}, t)p_\theta(\boldsymbol{x}_t)\big) + \frac{1}{2}g(t)^2\Delta p_\theta(\boldsymbol{x}_t). \tag{9}$$

Consider the time-dependent KL divergence:

$$D_{\mathrm{kl}}(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)) = \int q(\boldsymbol{x}_t)\log\frac{q(\boldsymbol{x}_t)}{p_\theta(\boldsymbol{x}_t)}\mathrm{d}\boldsymbol{x}_t. \tag{10}$$

Taking the derivative with respect to time gives:

$$\frac{\mathrm{d}}{\mathrm{d}t}D_{\mathrm{kl}}(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)) = \int \frac{\partial q(\boldsymbol{x}_t)}{\partial t}\log\frac{q(\boldsymbol{x}_t)}{p_\theta(\boldsymbol{x}_t)}\mathrm{d}\boldsymbol{x}_t - \int \frac{q(\boldsymbol{x}_t)}{p_\theta(\boldsymbol{x}_t)}\frac{\partial p_\theta(\boldsymbol{x}_t)}{\partial t}\mathrm{d}\boldsymbol{x}_t. \tag{11}$$

Substituting the above Fokker-Planck equations and applying integration by parts. Under the probability density boundary condition $\lim_{\|\boldsymbol{x}\|\to\infty} q(\boldsymbol{x}_t) = 0$, the drift terms cancel each other, leaving only the diffusion term:

$$\frac{\mathrm{d}}{\mathrm{d}t}D_{\mathrm{kl}}(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)) = -\frac{1}{2}g(t)^2\int q(\boldsymbol{x}_t)\|\nabla_{\boldsymbol{x}}\log q(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}}\log p_\theta(\boldsymbol{x}_t)\|_2^2\mathrm{d}\boldsymbol{x}_t. \tag{12}$$

Define the Fisher divergence as:

$$D_f(q\|p) = \mathbb{E}_q[\|\nabla_{\boldsymbol{x}}\log q(\boldsymbol{x}) - \nabla_{\boldsymbol{x}}\log p(\boldsymbol{x})\|_2^2], \tag{13}$$

then the generalized De Bruijn identity is obtained:

$$\frac{d}{dt}D_{\mathrm{kl}}(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)) = -\frac{1}{2}g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t)). \tag{14}$$

This equation demonstrates that the temporal evolution rate of the KL distribution discrepancy is determined by the difference between the score fields at the current diffusion time step.

Furthermore, integrating over the entire diffusion trajectory $t \in [0, T]$:

$$\int_0^T \frac{\mathrm{d}}{\mathrm{d}t}D_{\mathrm{kl}}(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t = -\frac{1}{2}\int_0^T g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t. \tag{15}$$

According to the fundamental theorem of calculus:

$$D_{\mathrm{kl}}(q(\boldsymbol{x}_T)\|p_\theta(\boldsymbol{x}_T)) - D_{\mathrm{kl}}(q(\boldsymbol{x}_0)\|p_\theta(\boldsymbol{x}_0)) = -\frac{1}{2}\int_0^T g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t. \tag{16}$$

When the diffusion time becomes sufficiently large, the forward perturbation gradually removes the original data structural information, and both the real distribution and the generated distribution converge to the same prior distribution:

$$q(\boldsymbol{x}_T) \approx p_\theta(\boldsymbol{x}_T), \tag{17}$$

therefore:

$$D_{\mathrm{kl}}(q(\boldsymbol{x}_T)\|p_\theta(\boldsymbol{x}_T)) \approx 0. \tag{18}$$

Finally, we obtain:

$$D_{\mathrm{kl}}(q(\boldsymbol{x}_0)\|p_\theta(\boldsymbol{x}_0)) = \frac{1}{2}\int_0^T g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t, \tag{19}$$

This result establishes that minimizing the original data-space KL divergence is equivalent to aligning the score fields of the real and generated distributions throughout the diffusion trajectory. Therefore, the distribution matching problem can be transformed into a score alignment problem, which provides the theoretical foundation for subsequent denoising score matching objectives and the proposed semantic score alignment regularization.

A.3 Relationship between Denoising Score Matching and MSE Optimization

Since the real data marginal distribution $q(\boldsymbol{x}_t)$ is unknown, its score function $\nabla_{\boldsymbol{x}_t}\log q(\boldsymbol{x}_t)$ is generally intractable to compute directly. Therefore, diffusion models employ Denoising Score Matching (DSM) to transform the marginal score estimation problem into a conditional score learning problem.

Equivalent Transformation of Fisher Divergence via DSM

Consider the forward diffusion process:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I}), \tag{20}$$

where:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{21}$$

where $\boldsymbol{x}_0$ denotes the original clean graph representation sampled from the data distribution, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ represents the Gaussian noise injected during the forward diffusion process. Here, α_t and σ_t control the signal and noise scales at diffusion timestep t, respectively.

The marginal distribution is given by:

$$q(\boldsymbol{x}_t) = \int q(\boldsymbol{x}_t|\boldsymbol{x}_0) q(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0. \tag{22}$$

According to DSM theory (Vincent, 2011), minimizing the Fisher divergence between the real marginal score and the model score:

$$D_f(q(\boldsymbol{x}_t) \| p_\theta(\boldsymbol{x}_t)) = \mathbb{E}_{q(\boldsymbol{x}_t)} \left[\| \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t) - \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \|_2^2 \right] \tag{23}$$

is equivalent to minimizing the conditional score matching objective:

$$\mathcal{L}_{\mathrm{DSM}} = \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{x}_t)} \left[\| \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t|\boldsymbol{x}_0) \|_2^2 \right]. \tag{24}$$

The two objectives differ only by a constant independent of the model parameters:

$$D_f(q(\boldsymbol{x}_t) \| p_\theta(\boldsymbol{x}_t)) = \mathcal{L}_{\mathrm{DSM}} + C(t). \tag{25}$$

Therefore, optimizing the Fisher divergence can be transformed into optimizing the DSM loss.

From Conditional Score to Noise Prediction Objective

For Gaussian perturbation:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I}), \tag{26}$$

the conditional score can be directly computed as:

$$\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t|\boldsymbol{x}_0) = -\frac{\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0}{\sigma_t^2}. \tag{27}$$

Let:

$$\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0 = \sigma_t \boldsymbol{\epsilon}, \tag{28}$$

then:

$$\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t|\boldsymbol{x}_0) = -\frac{\boldsymbol{\epsilon}}{\sigma_t}. \tag{29}$$

Therefore, the DSM objective can be written as:

$$\mathcal{L}_{\mathrm{DSM}} = \mathbb{E}\left[\left\| \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) + \frac{\boldsymbol{\epsilon}}{\sigma_t} \right\|_2^2 \right]. \tag{30}$$

Furthermore, defining the noise prediction network as:

$$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) = -\sigma_t \boldsymbol{s}_\theta(\boldsymbol{x}_t, t), \tag{31}$$

then:

$$\mathcal{L}_{\mathrm{DSM}} = \frac{1}{\sigma_t^2} \mathbb{E}[\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \|_2^2]. \tag{32}$$

This demonstrates that score matching can be equivalently transformed into a noise prediction MSE objective.

Denoising Reconstruction Objective and Tweedie's Formula

In the previous subsection, we established that Denoising Score Matching (DSM) can be implemented through the noise prediction objective, to analyze the optimization objective from the perspective of the original data space $\boldsymbol{x}_0$, we rewrite the true noise and the predicted noise according to the forward diffusion process $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$. Therefore, the true noise and the predicted noise can be expressed as:

$$\boldsymbol{\epsilon} = \frac{\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0}{\sigma_t}, \quad \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{x}_t - \alpha_t \widehat{\boldsymbol{x}}}{\sigma_t}, \tag{33}$$

where $\widehat{\boldsymbol{x}}$ denotes the denoised data. Substituting these two expressions into the DSM objective, we obtain:

$$\mathcal{L}_{\text{DSM}} = \frac{1}{\sigma_t^2} \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{x}_t)} \left[\left\| \frac{\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0}{\sigma_t} - \frac{\boldsymbol{x}_t - \alpha_t \widehat{\boldsymbol{x}}}{\sigma_t} \right\|_2^2 \right] = \frac{\alpha_t^2}{\sigma_t^4} \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{x}_t)} \left[\| \boldsymbol{x}_0 - \widehat{\boldsymbol{x}} \|_2^2 \right]. \tag{34}$$

By integrating this equivalence along the entire diffusion trajectory $t \in [0, T]$, the Fisher divergence objective can be reformulated as a weighted reconstruction objective:

$$\begin{aligned} \frac{1}{2} \int_0^T g(t)^2 D_f \left(q(\boldsymbol{x}_t) \| p_\theta(\boldsymbol{x}_t) \right) \mathrm{d}t &\equiv \int_0^T \underbrace{\left(\frac{1}{2} g(t)^2 \frac{\alpha_t^2}{\sigma_t^4} \right)}_{w(t)} \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{x}_t)} \left[\| \boldsymbol{x}_0 - \widehat{\boldsymbol{x}} \|_2^2 \right] \mathrm{d}t \\ &= \int_0^T w(t) \mathbb{E}_{q(\boldsymbol{x}_0, \boldsymbol{x}_t)} \left[\| \boldsymbol{x}_0 - \widehat{\boldsymbol{x}} \|_2^2 \right] \mathrm{d}t, \end{aligned} \tag{35}$$

where $w(t)$ represents the time-dependent weighting function determined by the diffusion schedule.

Furthermore, the above reconstruction objective can be rigorously derived from the perspective of Tweedie's formula (Song et al., 2021). For the forward Gaussian diffusion process: $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$. Tweedie's formula indicates that the original data $\boldsymbol{x}_0$ can be recovered as:

$$\boldsymbol{x}_0 = \frac{1}{\alpha_t} \left(\boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t | \boldsymbol{x}_0) \right), \tag{36}$$

where $\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t | \boldsymbol{x}_0)$ denotes the conditional score function given the clean sample $\boldsymbol{x}_0$. By learning the score function of the marginal distribution $q(\boldsymbol{x}_t)$:

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \approx \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t), \tag{37}$$

the denoised estimation produced by the model can be formulated as:

$$\widehat{\boldsymbol{x}} = \frac{1}{\alpha_t} \left(\boldsymbol{x}_t + \sigma_t^2 \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \right), \tag{38}$$

which further gives the following parameterization of the model score:

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) = \frac{\alpha_t \widehat{\boldsymbol{x}} - \boldsymbol{x}_t}{\sigma_t^2}. \tag{39}$$

On the other hand, for a given clean sample $\boldsymbol{x}_0$, the conditional perturbation distribution is defined as:

$$q(\boldsymbol{x}_t | \boldsymbol{x}_0) = \mathcal{N}(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I}), \tag{40}$$

and its conditional score function can be analytically derived as:

$$\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t | \boldsymbol{x}_0) = \frac{\alpha_t \boldsymbol{x}_0 - \boldsymbol{x}_t}{\sigma_t^2}. \tag{41}$$

Subtracting the conditional score from the estimated score yields the direct correspondence between the score matching residual and the reconstruction error:

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t | \boldsymbol{x}_0) = \frac{\alpha_t \widehat{\boldsymbol{x}} - \boldsymbol{x}_t}{\sigma_t^2} - \frac{\alpha_t \boldsymbol{x}_0 - \boldsymbol{x}_t}{\sigma_t^2} = \frac{\alpha_t}{\sigma_t^2} (\widehat{\boldsymbol{x}} - \boldsymbol{x}_0). \tag{42}$$

Taking the squared L_2 norm on both sides and computing the expectation gives:

$$\mathbb{E}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\log q(\boldsymbol{x}_t|\boldsymbol{x}_0)\|_2^2\right]=\frac{\alpha_t^2}{\sigma_t^4}\mathbb{E}_{q(\boldsymbol{x}_0,\boldsymbol{x}_t)}\left[\|\boldsymbol{x}_0-\widehat{\boldsymbol{x}}\|_2^2\right]. \tag{43}$$

According to the equivalence between marginal score matching and DSM established previously, the Fisher divergence of the marginal distributions is equivalent to the conditional DSM objective up to a constant term independent of model parameters:

$$D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\equiv\mathbb{E}_{q(\boldsymbol{x}_0,\boldsymbol{x}_t)}\left[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\log q(\boldsymbol{x}_t|\boldsymbol{x}_0)\|_2^2\right]. \tag{44}$$

Substituting the above equivalence between score matching error and reconstruction error into the trajectory-wise Fisher divergence objective yields:

$$\frac{1}{2}\int_0^T g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t\equiv\frac{1}{2}\int_0^T g(t)^2\frac{\alpha_t^2}{\sigma_t^4}\mathbb{E}_{q(\boldsymbol{x}_0,\boldsymbol{x}_t)}\left[\|\boldsymbol{x}_0-\widehat{\boldsymbol{x}}\|_2^2\right]\mathrm{d}t. \tag{45}$$

By defining the time-dependent weighting function: $w(t)=\frac{1}{2}g(t)^2\frac{\alpha_t^2}{\sigma_t^4}$, the Fisher divergence can be finally rewritten as the weighted reconstruction objective over the entire diffusion trajectory:

$$\frac{1}{2}\int_0^T g(t)^2 D_f(q(\boldsymbol{x}_t)\|p_\theta(\boldsymbol{x}_t))\mathrm{d}t\equiv\int_0^T w(t)\mathbb{E}_{q(\boldsymbol{x}_0,\boldsymbol{x}_t)}\left[\|\boldsymbol{x}_0-\widehat{\boldsymbol{x}}\|_2^2\right]\mathrm{d}t. \tag{46}$$

However, under the MSE reconstruction objective, the optimal estimator is strictly given by:

$$\widehat{\boldsymbol{x}}=\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]. \tag{47}$$

According to the minimum mean squared error estimation theory, the analytical solution of MSE optimization is:

$$\widehat{\boldsymbol{x}}=\arg\min_{\hat{\boldsymbol{x}}}\mathbb{E}_{q(\boldsymbol{x}_0|\boldsymbol{x}_t)}\left[\|\boldsymbol{x}_0-\hat{\boldsymbol{x}}\|_2^2\right]=\mathbb{E}_{q(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\boldsymbol{x}_0|\boldsymbol{x}_t]. \tag{48}$$

This property indicates that MSE optimization essentially performs posterior mean estimation rather than explicitly learning the complete conditional posterior distribution $q(\boldsymbol{x}_0|\boldsymbol{x}_t)$.

For graph generation tasks, due to the discrete, combinatorial, and topologically constrained nature of graph structures, multiple plausible structural modes may exist for a given noisy state $\boldsymbol{x}_t$. Therefore, the conditional distribution of clean graphs can be represented as a mixture of latent structural modes:

$$q(\boldsymbol{x}_0|\boldsymbol{x}_t)=\sum_{k=1}^K\pi_k q_k(\boldsymbol{x}_0|\boldsymbol{x}_t), \tag{49}$$

where $q_k(\boldsymbol{x}_0|\boldsymbol{x}_t)$ denotes the k-th latent structural mode and π_k represents its corresponding probability weight. In this case, the conditional mean learned by MSE becomes:

$$\widehat{\boldsymbol{x}}=\sum_{k=1}^K\pi_k\boldsymbol{\mu}_k, \tag{50}$$

where $\boldsymbol{\mu}_k$ denotes the center of the k-th structural mode.

However, unlike continuous Euclidean spaces, graph spaces are discrete and non-convex. The linear interpolation among different valid graph structural modes generally does not correspond to a valid graph structure, but instead falls into a low-density region outside the true data manifold. Consequently, MSE-based reconstruction networks implicitly collapse the complex multimodal posterior distribution into a deterministic point estimate:

$$p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t)\approx\delta(\boldsymbol{x}_0-\widehat{\boldsymbol{x}}), \tag{51}$$

thereby ignoring the structural diversity and generation uncertainty contained in the true posterior distribution.

Such posterior degeneration further affects the reverse diffusion process. Ideally, the single-step reverse transition can be expressed as:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \int q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t) \mathrm{d}\boldsymbol{x}_0. \tag{52}$$

When the posterior distribution is approximated by a deterministic estimation:

$$p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t) \approx \delta(\boldsymbol{x}_0 - \widehat{\boldsymbol{x}}), \tag{53}$$

the reverse transition degenerates into:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) \approx q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \widehat{\boldsymbol{x}}). \tag{54}$$

At this point, the reverse diffusion process is no longer sampled according to the complete posterior distribution, but is instead guided by a single posterior mean estimate $\widehat{\boldsymbol{x}}$. When $\widehat{\boldsymbol{x}}$ deviates from the true data manifold, local reconstruction errors accumulate through successive reverse diffusion steps, eventually resulting in a distribution shift between the generated diffusion distribution $p_\theta(\boldsymbol{x}_t)$ and the real diffusion distribution $q(\boldsymbol{x}_t)$. Therefore, relying solely on local MSE reconstruction constraints is insufficient to guarantee global distributional consistency throughout the entire generation trajectory.

A.4 From Reconstruction Consistency to Semantic Score Alignment

The above analysis reveals that the conventional denoising reconstruction objective, although effective in learning local denoising capability, primarily optimizes point-wise reconstruction errors and implicitly estimates the posterior mean of the conditional distribution. For complex structured data such as graphs, where the data distribution is inherently discrete, non-convex, and multi-modal, such point-wise optimization may cause posterior degeneration and fail to preserve the global structural diversity of the underlying data manifold. Consequently, minimizing reconstruction errors alone cannot guarantee that the generated samples remain consistent with the true data distribution throughout the reverse diffusion trajectory.

Motivated by this observation, we propose Semantic Score Alignment (SSA), a distribution-aware regularization framework that complements the conventional denoising objective by explicitly enforcing distributional consistency in the semantic representation space. Instead of matching noisy distributions at every diffusion timestep, SSA focuses on aligning the empirical distributions of clean real samples and model-reconstructed samples. Specifically, we first project discrete data structures into a continuous semantic space, where the empirical distributions of real and predicted samples are estimated. Then, their corresponding score fields are aligned through Fisher divergence minimization. By constraining the semantic score consistency between generated and real distributions, SSA encourages reconstructed samples to remain close to the underlying data manifold and preserves the intrinsic structural diversity beyond local reconstruction accuracy. Therefore, SSA provides a principled distribution-level regularization mechanism for diffusion models, bridging the gap between point-wise denoising optimization and global manifold consistency.

B Derivation of Empirical Score Function via Kernel Density Estimation

B.1 KDE Density Estimation

Assume that n_b independent and identically distributed (i.i.d.) samples are drawn from an unknown data distribution $q(\boldsymbol{z})$:

$$\mathcal{Z} = \{\boldsymbol{z}_i\}_{i=1}^{n_b}, \quad \boldsymbol{z}_i \in \mathbb{R}^d.$$

Using Kernel Density Estimation (KDE), the continuous empirical density function is estimated as:

$$\widehat{q}(\boldsymbol{z}) = \frac{1}{n_b} \sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i), \tag{55}$$

where $k_h(\cdot, \cdot)$ denotes an isotropic Gaussian kernel with bandwidth parameter h:

$$k_h(\boldsymbol{z}, \boldsymbol{z}_i) = \frac{1}{(2\pi h^2)^{d/2}} \exp\left(-\frac{\|\boldsymbol{z} - \boldsymbol{z}_i\|^2}{2h^2}\right). \tag{56}$$

Our goal is to derive the score function corresponding to the empirical density $\widehat{q}(\boldsymbol{z})$:

$$\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}). \tag{57}$$

B.2 Derivation of KDE-based Score Function

According to the definition of the logarithmic gradient:

$$\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}) = \frac{\nabla_{\boldsymbol{z}} \widehat{q}(\boldsymbol{z})}{\widehat{q}(\boldsymbol{z})}. \tag{58}$$

First, we compute the gradient of the Gaussian kernel with respect to the query point $\boldsymbol{z}$:

$$\nabla_{\boldsymbol{z}} k_h(\boldsymbol{z}, \boldsymbol{z}_i) = k_h(\boldsymbol{z}, \boldsymbol{z}_i) \nabla_{\boldsymbol{z}} \left(-\frac{\|\boldsymbol{z} - \boldsymbol{z}_i\|^2}{2h^2} \right) = \frac{1}{h^2} k_h(\boldsymbol{z}, \boldsymbol{z}_i)(\boldsymbol{z}_i - \boldsymbol{z}). \tag{59}$$

Therefore, based on the linearity of differentiation, the gradient of the KDE density is given by:

$$\nabla_{\boldsymbol{z}} \widehat{q}(\boldsymbol{z}) = \frac{1}{n_b h^2} \sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i)(\boldsymbol{z}_i - \boldsymbol{z}). \tag{60}$$

Substituting Eq. 60 and Eq. 55 into Eq. 58, we obtain:

$$\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}) = \frac{1}{h^2} \frac{\sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i)(\boldsymbol{z}_i - \boldsymbol{z})}{\sum_{j=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_j)} = \frac{1}{h^2} \left(\frac{\sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i) \boldsymbol{z}_i}{\sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i)} - \boldsymbol{z} \right). \tag{61}$$

We further define the weighted local center as:

$$\boldsymbol{\mu}(\boldsymbol{z}) = \frac{\sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i) \boldsymbol{z}_i}{\sum_{i=1}^{n_b} k_h(\boldsymbol{z}, \boldsymbol{z}_i)}. \tag{62}$$

B.3 Geometric Interpretation

Since the Gaussian kernel bandwidth satisfies $h^2 > 0$, it only acts as a positive scaling coefficient of the score magnitude. Therefore, ignoring the constant coefficient, we have:

$$\nabla_{\boldsymbol{z}} \log \widehat{q}(\boldsymbol{z}) \propto \boldsymbol{\mu}(\boldsymbol{z}) - \boldsymbol{z}. \tag{63}$$

This result indicates that the KDE-based score direction geometrically corresponds to the displacement vector from the current query point $\boldsymbol{z}$ toward its local density center $\boldsymbol{\mu}(\boldsymbol{z})$. In other words, the score function describes the local direction of increasing probability density, which can be characterized through the weighted neighborhood geometry of surrounding samples.

This derivation provides the theoretical foundation for estimating score fields using local neighborhood structures. Instead of explicitly evaluating the probability density function, the local geometric relationship among neighboring samples can be exploited to approximate the underlying distribution gradient direction.

C Detailed network architecture

C.1 Backbone denoising network

The proposed graph generation model adopts a graph Transformer architecture to jointly capture node-level, edge-level, and graph-level information. The overall network is built upon a stack of XEy Transformer layers, where node features, edge features, and global graph representations are updated simultaneously through feature interaction and attention-based message passing.

Given an input graph, the node attributes, edge attributes, and global graph features are first projected into latent feature spaces using three independent multilayer perceptrons (MLPs). The node and edge

representations are embedded into hidden dimensions for subsequent Transformer processing, while the global graph representation is mapped into a compact latent space. To enhance the structural representation capability, the edge features are augmented with higher-order adjacency information before feature projection, enabling the model to incorporate multi-hop topological dependencies.

The core component of the network is the XEy Transformer layer, which consists of a graph-aware self-attention module and three independent feed-forward networks for updating node, edge, and global representations. Different from conventional Transformer architectures that only consider interactions between node features, the proposed layer explicitly incorporates edge information into the attention mechanism. Specifically, edge representations are transformed into learnable modulation factors that adaptively adjust the attention coefficients between node pairs. Therefore, the attention operation is simultaneously conditioned on node similarity and pairwise graph connectivity, allowing the model to capture complex structural patterns in graph data.

The edge representations are updated based on the learned attention relationships between node pairs. In addition, global graph features are introduced through feature-wise linear modulation (FiLM), where graph-level information dynamically controls the update process of both node and edge representations. This design enables effective information exchange between local structural features and global graph semantics.

The global graph representation is updated by aggregating information from both node and edge features. Specifically, permutation-invariant statistical pooling operations, including mean, maximum, minimum, and standard deviation aggregation, are applied to node and edge representations to extract graph-level structural descriptors. These descriptors are combined with the existing global features and further transformed through an MLP, allowing the network to model global molecular properties and long-range dependencies.

Each XEy Transformer layer employs residual connections and layer normalization to improve training stability. After the attention-based feature interaction stage, node, edge, and global features are independently refined through feed-forward networks. Multiple Transformer layers are sequentially stacked to progressively enhance the expressive capability of the graph representation.

After the final Transformer layer, the latent node and edge representations are mapped back to their original feature spaces through output MLPs. For molecular graph generation, node predictions are converted into categorical distributions using softmax normalization, while edge predictions are transformed into bond probabilities using sigmoid activation. The generated edge representations are further symmetrized to guarantee the undirected property of molecular graphs. Node and edge masks are applied throughout the network to support variable-size graphs and avoid the influence of padded nodes during mini-batch training.

C.2 GCN-based Graph Latent Representation Encoder

To map discrete molecular graphs into a continuous latent space, we design a graph representation encoder based on Graph Convolutional Networks (GCNs). The proposed module jointly models node attributes and graph structural information to learn graph-level latent representations with rich structural semantics, which are further used for distribution estimation.

Given the node features and adjacency relationships of molecular graphs, multiple GCN layers are employed to progressively update node representations. Each GCN layer aggregates neighborhood information through a normalized adjacency matrix, enabling node features to incorporate local topological patterns. The aggregated features are further transformed through linear projection, batch normalization, and nonlinear activation to enhance representation capability. For molecular graphs with multiple edge types, the edge features are first aggregated into a unified adjacency representation, allowing the GCN encoder to effectively handle different bond structures.

After several graph convolution operations, the resulting node representations capture higher-order structural information. A masked global average pooling operation is then applied to aggregate node-level features into a fixed-dimensional graph-level representation. This operation ensures permutation invariance while preserving the overall structural characteristics of molecular graphs.

Finally, a lightweight MLP mapping network is applied to further transform the graph-level representation into a continuous latent embedding. The obtained latent representation encodes both

local topological patterns and global structural semantics of molecular graphs, providing an effective feature space for subsequent distribution estimation.

Table 7: Training configurations and computational costs of SSA on different datasets

Dataset	Batch size	Epoch	K	λ	Time (h)	Mem. (GB)
Planar	128	30000	64	0.5	5.0 ± 1.0	30.0 ± 3.0
SBM	32	30000	16	0.5	32.0 ± 3.0	60.0 ± 5.0
Ego-small	128	5000	64	0.5	1.0 ± 0.5	2.0 ± 1.0
Community-small	128	5000	64	0.5	0.25 ± 0.1	2.0 ± 1.0
Enzymes	32	5000	16	0.5	35.0 ± 3.0	14.0 ± 2.0
Grid	8	5000	4	0.5	35.0 ± 5.0	50.0 ± 5.0
QM9	2048	1000	1024	0.05	15.0 ± 2.0	15.0 ± 2.0
ZINC250k	512	1000	256	0.05	65.0 ± 10.0	55.0 ± 5.0

D Detailed experimental setting

D.1 Experimental environment

To ensure the reproducibility of our results, all experiments were conducted on a high-performance computing node equipped with an NVIDIA GeForce RTX 3090 GPU (24GB VRAM). The experimental environment was built on a Linux system with Python 3.9.23, and CUDA 12.5 was utilized for GPU acceleration to enable efficient deep learning computation.

All models were implemented based on PyTorch 1.13.1 and PyTorch Geometric (PyG) 2.3.1. For graph processing and molecular property evaluation, NetworkX and RDKit were employed for graph analysis and chemical property computation, respectively. In addition, commonly used scientific computing libraries, including NumPy and SciPy, were adopted for data preprocessing and auxiliary numerical operations.

D.2 Training Details and Hyperparameter Settings

To ensure the reproducibility of our results, we report the detailed training configurations and computational costs of the proposed SSA framework in Table 7. Since different datasets exhibit significant variations in graph size, number of nodes, and structural complexity, we adopt dataset-specific training strategies and configure the batch size, training epochs, latent-space neighborhood size K, and distribution alignment weight λ accordingly. Meanwhile, the training time and GPU memory consumption are also recorded.

In all experiments, the model is trained using the AdamW optimizer with an initial learning rate of 2.0×10^{-4} and a weight decay coefficient of 1.0×10^{-12}. To improve training stability and prevent gradient explosion, gradient clipping is applied during training, with the maximum gradient norm limited to 1.0. In addition, an Exponential Moving Average (EMA) mechanism is introduced to smoothly update the model parameters, with a decay rate of 0.999. The EMA parameters are used during evaluation to obtain more stable generation results.

For the optimization objective, the model jointly optimizes the original graph generation loss and the proposed distribution consistency constraint. The weighting coefficient λ is used to balance the contribution between the generation objective and the latent-space distribution alignment term.

During the diffusion sampling process, we employ the Euler predictor as the numerical solver for the reverse process and do not introduce an additional corrector step to reduce the computational complexity of sampling. The Signal-to-Noise Ratio (SNR) is set to 0, the sampling scale parameter is set to 1.0, and one prediction update is performed at each sampling time step.

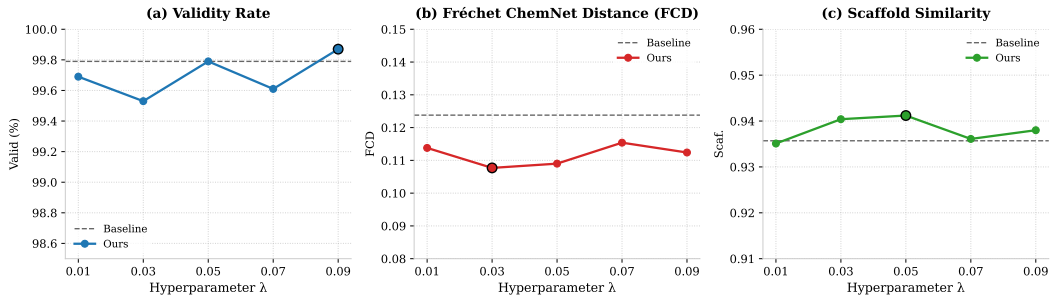


Figure 2: Parameter sensitivity analysis with respect to the hyperparameter λ across three performance metrics: (a) Validity Rate, (b) Fréchet ChemNet Distance (FCD), and (c) Scaffold Similarity. The gray dashed lines denote the baseline performance.

E Hyperparameter and Stability Analysis

E.1 Sensitivity Analysis of λ

The hyperparameter λ controls the contribution of the proposed distribution alignment constraint. To investigate its influence on the generation performance, we evaluate SSA with different λ values, including 0.01, 0.03, 0.05, 0.07, and 0.09, while keeping other configurations unchanged. The results are presented in Fig. 2. As shown in Fig. 2, SSA achieves stable performance under different λ settings. The Valid, FCD, and Scaf. metrics exhibit only minor fluctuations, indicating that the proposed framework is insensitive to the selection of λ.

E.2 Analysis of Different Values of K

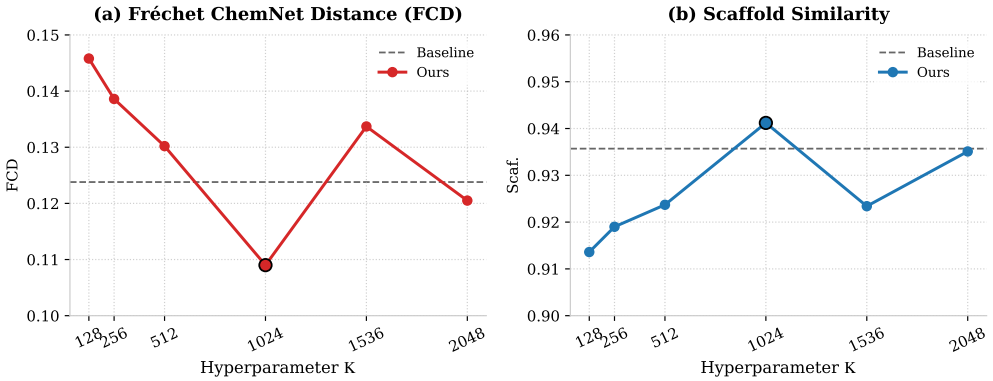


Figure 3: Performance comparison under different neighborhood sizes K. The gray dashed lines denote the baseline performance.

The neighborhood size K determines the number of real latent representations used for local KDE-based score estimation, which directly affects the accuracy of the estimated score field. As shown in Fig. 3, the performance exhibits a clear sensitivity to the choice of K. When K is relatively small (e.g., $K = 128$ and $K = 256$), the limited neighborhood information leads to insufficient characterization of local semantic structures, resulting in inaccurate score directions and degraded generation performance. As K increases, more informative manifold neighbors are incorporated, leading to continuous improvements in both FCD and scaffold similarity.

The best performance is achieved at $K = 1024$, where the model obtains an FCD of 0.1090 and a Scaf. score of 0.9412. This indicates that an appropriate neighborhood size provides a favorable balance between local geometric preservation and stable density estimation. However, further increasing K

introduces more distant samples from different semantic regions, which weakens the locality of the estimated score field and reduces the effectiveness of manifold guidance.

E.3 Statistical Analysis with Different Random Seeds

Table 8: Performance of SSA under different random seeds on the QM9 dataset.

Seed	Valid (%) ↑	FCD ↓	NSPDK ↓	Scaf. ↑
40	99.84	0.1124	0.0002	0.9390
41	99.72	0.1106	0.0003	0.9435
42	99.79	0.1090	0.0002	0.9412
43	99.70	0.1192	0.0002	0.9427
44	99.81	0.1087	0.0003	0.9381
Mean	99.77	0.1120	0.0002	0.9410

Table 9: Performance of SSA under different random seeds on the ZINC250k dataset.

Seed	Valid (%) ↑	FCD ↓	NSPDK ↓	Scaf. ↑
40	98.76	1.5324	0.0011	0.5598
41	99.63	1.5487	0.0010	0.5603
42	99.40	1.5660	0.0009	0.5629
43	99.32	1.6015	0.0008	0.5707
44	98.95	1.7235	0.0008	0.5770
Mean	99.21	1.5944	0.0009	0.5661

To verify that the performance improvements of SSA are not caused by random initialization, we further evaluate SSA with five different random seeds on both the QM9 and ZINC250k datasets. The results are reported in Tables 8 and 9, respectively. It can be observed that SSA maintains stable generation performance across different random seeds on both datasets. Specifically, on the QM9 dataset, the FCD score remains within a narrow range, while the Scaffold similarity consistently maintains a high level, demonstrating the robustness of SSA in preserving molecular structural characteristics. Similarly, on the larger-scale ZINC250k dataset, SSA achieves consistently competitive performance under different random seeds, despite the increased structural complexity and larger molecular space.

Although slight variations exist due to the stochastic nature of diffusion training and sampling, SSA consistently provides reliable generation quality without depending on a specific random seed. The average results over multiple runs further validate the effectiveness of the proposed Semantic Score Alignment strategy, indicating that the observed improvements mainly originate from the introduced distribution-level score constraint rather than random fluctuations.

F Datasets Description

In this section, we describe the datasets used for training the SSA model in detail, covering their overall scale, the distribution of molecular sizes in terms of node numbers, and the generation process of the molecules.

Planar. The Planar Graph dataset consists of 200 planar graphs, each containing 64 nodes. These graphs are generated by randomly sampling points in a two-dimensional space and constructing edges through Delaunay triangulation, which guarantees planar structures without edge crossings. Node attributes can represent spatial coordinates or node identifiers, while edges encode the triangulation-based connectivity. This dataset provides a controlled benchmark for evaluating graph generative models in terms of their ability to preserve planar constraints, capture diverse topological patterns, and maintain structural validity in the generated graphs.

SBM. The SBM (Stochastic Block Model) dataset comprises 200 synthetic graphs, each containing 2–5 communities with 20–40 nodes per community. Edges are generated according to predefined connection probabilities, with an intra-community connection probability of 0.3 and an inter-community connection probability of 0.05. All graphs are undirected with a single edge type, while node features can optionally incorporate community labels or additional attributes. This dataset serves as a benchmark for evaluating graph generative models in terms of community structure preservation, topological diversity, and latent space consistency.

Ego-small. The Ego-small dataset contains 200 ego graphs extracted from the Citeseer citation network. Citeseer is a widely used academic citation graph, where nodes represent research papers and edges indicate citation relationships. Each ego graph is constructed by selecting a central node and its neighboring nodes, resulting in a local subgraph with 4–18 nodes. Node attributes can be associated with paper categories or citation-related information, while edges are undirected and of a single type. This dataset provides a benchmark for evaluating graph generative models on their ability to capture sparse local structures and reproduce realistic network patterns.

Community-small. The Community-small dataset contains 100 randomly generated community graphs, with each graph consisting of 12–20 nodes. Node attributes can optionally incorporate community labels to distinguish different groups. The graphs contain undirected edges with a single edge type, where connections are more likely to occur within communities than across communities, resulting in clear modular structures. This dataset serves as a benchmark for evaluating graph generative models in preserving community-level organization and capturing characteristic topological patterns.

Enzymes. ENZYMES consists of 587 protein graphs, each representing the tertiary structure of an individual enzyme. Nodes correspond to amino acid residues and are categorized into three types according to their chemical properties or functional roles. Edges represent spatial or chemical interactions between residues and are undirected with a single edge type. The number of nodes in each graph ranges from 10 to 125. This dataset provides a benchmark for evaluating graph generative models in capturing biochemical structural patterns and learning informative protein representations.

Grid. The Grid dataset consists of 100 regular 2D grid graphs, with each graph containing 100–400 nodes. All nodes belong to a single homogeneous category, and edges represent adjacency relationships between neighboring nodes with an undirected single edge type. This dataset provides a controlled benchmark for evaluating graph generative models on their ability to reproduce regular grid structures and preserve underlying structural organization in the latent space.

QM9. We conduct experiments on the 2D version of QM9 (QM9-2D), containing approximately 134,000 small organic molecules with atom types $\{\mathrm{C}, \mathrm{O}, \mathrm{N}, \mathrm{F}, \mathrm{H}\}$. Each molecule is represented as a graph, retaining atomic connectivity and bond types (single, double, triple, aromatic), with hydrogen atoms inferred from valence rules. Node features encode atom categories and discrete valence attributes, while edge features encode bond types. Molecules are encoded as adjacency and node feature matrices, normalized to a maximum of nine atoms and four bond types. The 2D representation emphasizes molecular topology, enabling graph-level generative modeling while avoiding geometric degrees of freedom.

ZINC250k. ZINC250k contains approximately 250,000 drug-like molecules, each with up to 38 atoms. The elements primarily include C, N, O, F, S, and Cl, with a small number of molecules containing Br or P. Each molecule is represented using both SMILES and SELFIES notations, and the bond types include single, double, and aromatic bonds. The dataset provides rich structural diversity and serves as an important benchmark for molecular generation and representation learning.

G Graph Generation Visualization

In this section, we present the molecular visualization results of our SSA model on different datasets. For the visualization results, the left column shows the molecular graphs generated from the training set, while the right column presents the visualizations produced by our SSA model.

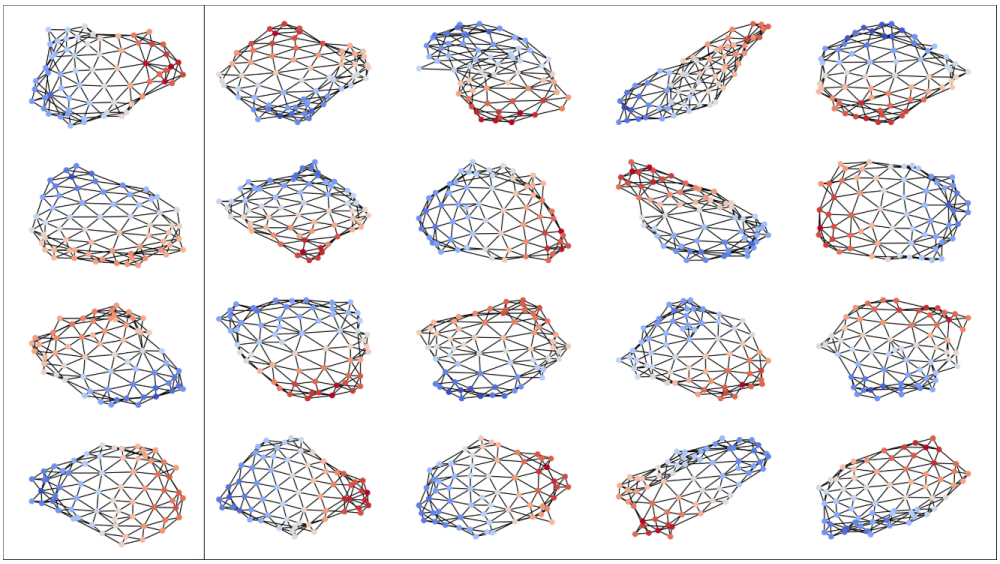


Figure 4: Visualization of graphs from the Planar dataset and the graphs generated by SSA.

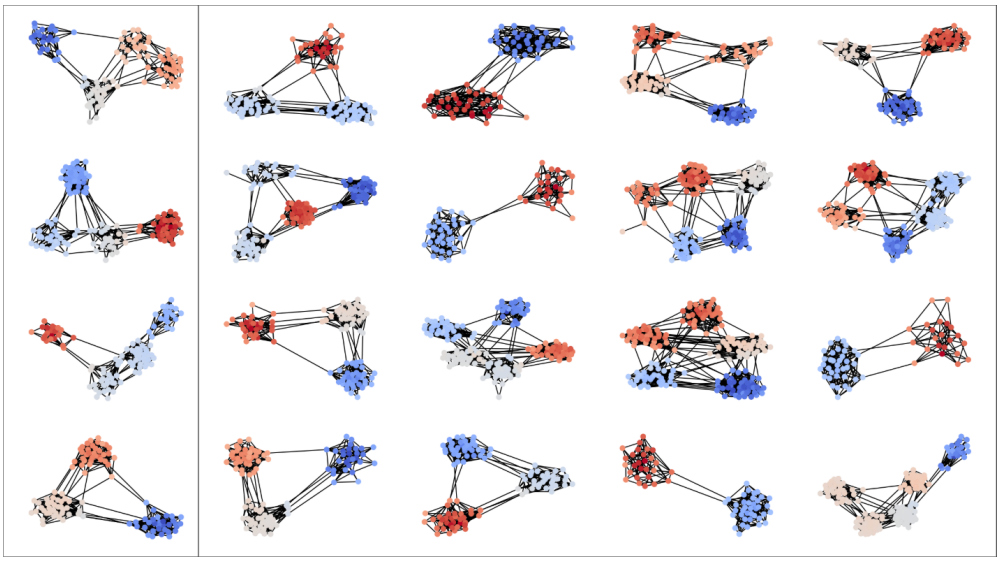


Figure 5: Visualization of graphs from SBM and the graphs generated by SSA.

G.1 Visualization of general graph

Figures 4, 5, 6, 7, 8, and 9 present the visualization results of the training sets and our SSA on the Planar, SBM, Ego-small and ENZYMES datasets.

G.2 Visualization of molecular graphs

Figures 10 and 11 show the visualization results for both the training sets and ourSSAmodel on the QM9 and ZINC250k molecular datasets.

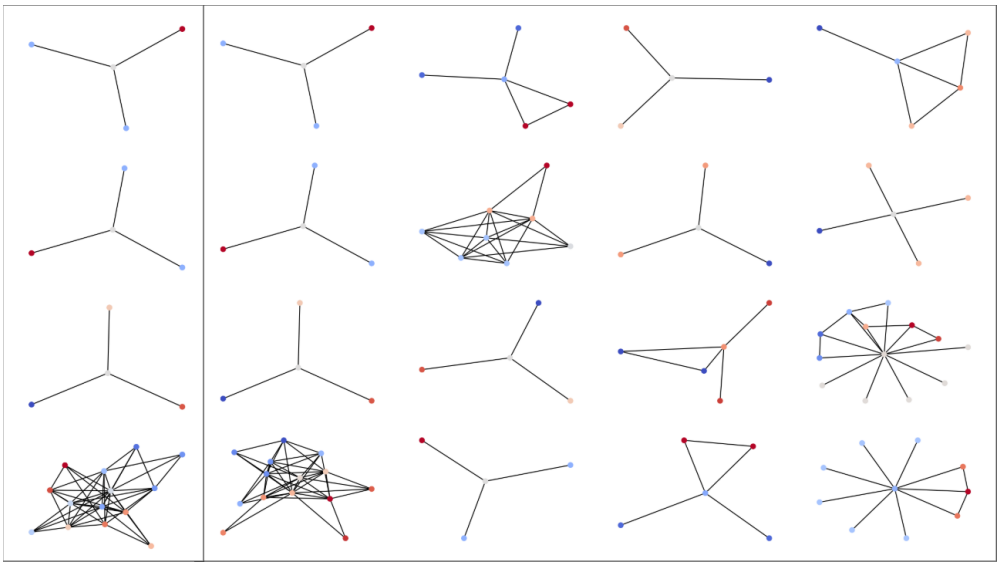


Figure 6: Visualization of graphs from the Ego-small and the graphs generated by SSA.

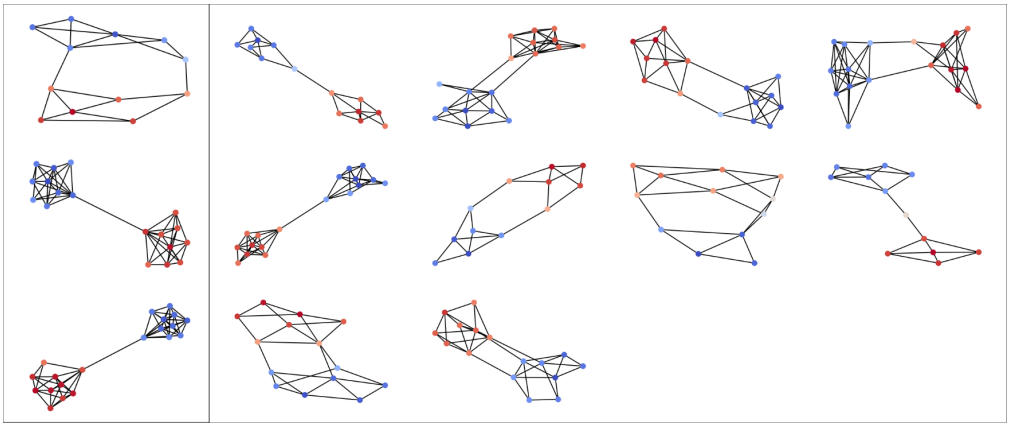


Figure 7: Visualization of graphs from the Community-small and the graphs generated by SSA.

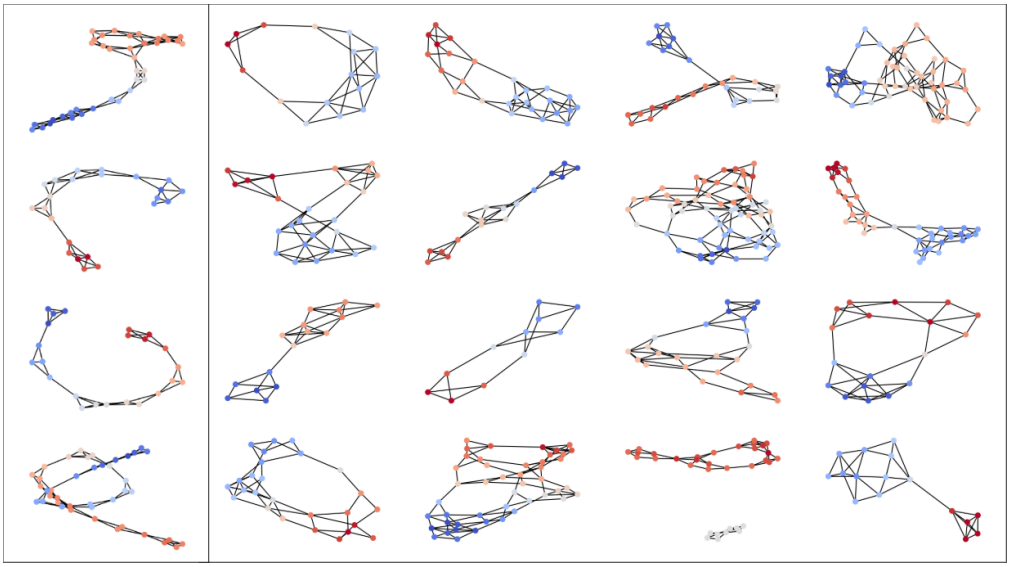


Figure 8: Visualization of graphs from ENZYMES and the graphs generated by SSA.

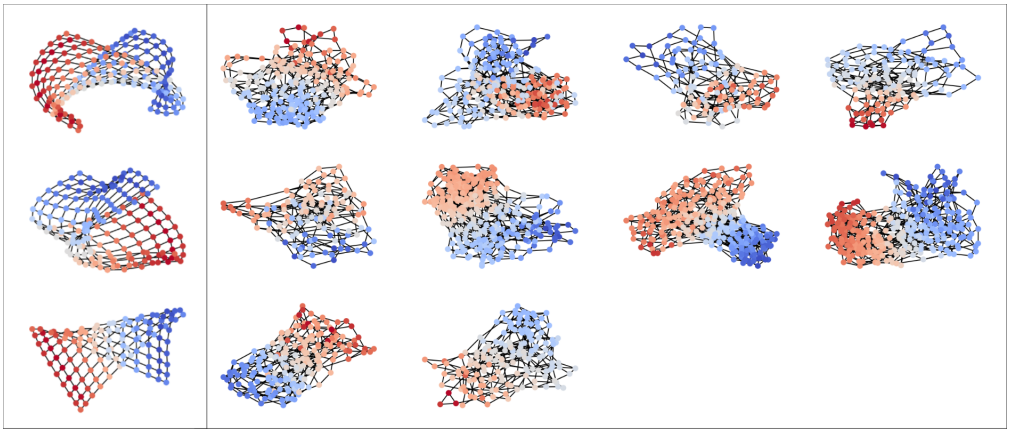


Figure 9: Visualization of graphs from the Grid dataset and the graphs generated by SSA.

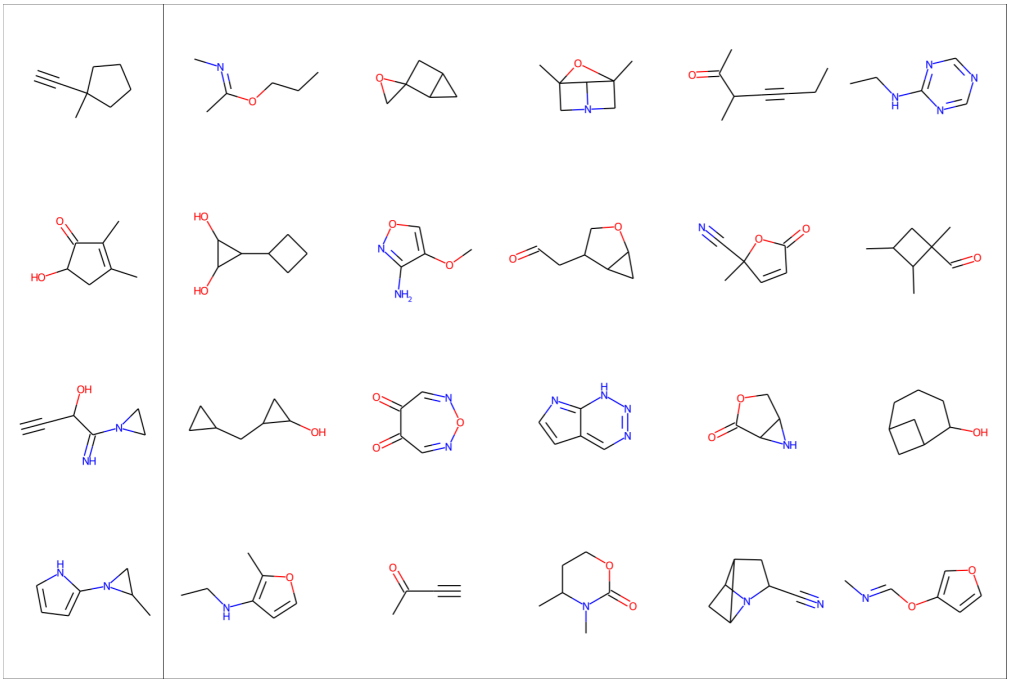


Figure 10: Visualization of graphs from QM9 and the graphs generated by SSA.

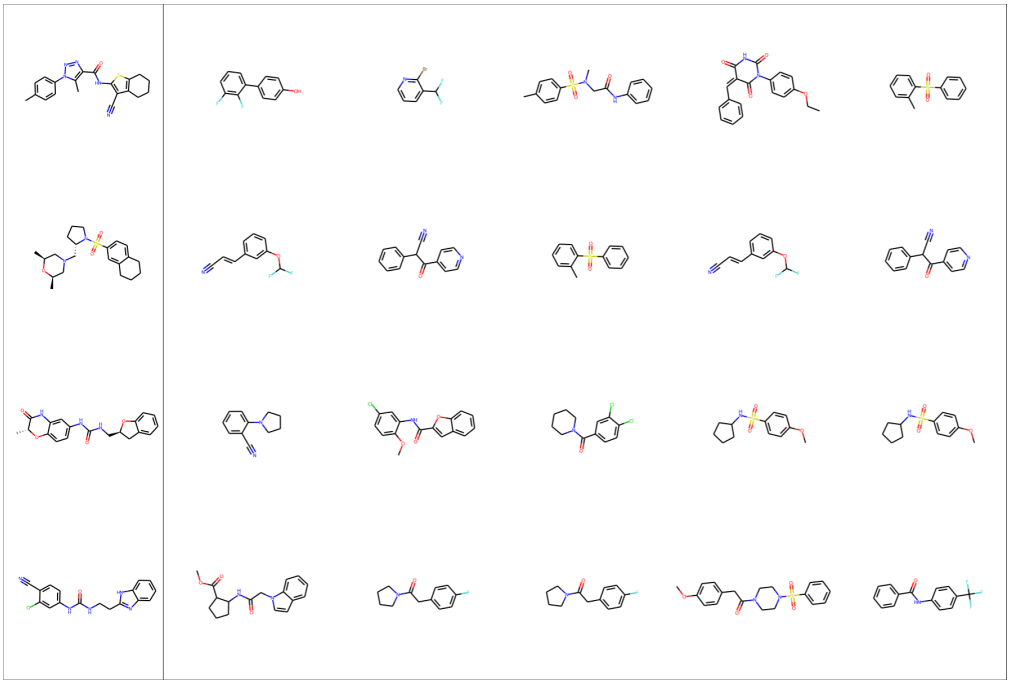


Figure 11: Visualization of graphs from ZINC250k and the graphs generated by SSA.