Efficient Retrosynthesis Prediction with Integral Flow Matching and Latent Inversion

Tao Yin[1] **Xiaohong Zhang**[1*] **Yinjie Zhu**[1] **Jiacheng Zhang**[1] **Haotian Zou**[1]
Li Huang[1] **Jiajun Cai**[1] **Zhibin Zhang**[1] **Meng Yan**[1*]

[1]Chongqing University

Abstract

Fast and accurate retrosynthesis prediction is desired for downstream drug discovery and synthetic planning tasks. Currently, training and sampling state-of-the-art discrete diffusion or flow-based models for retrosynthesis requires significant computational resources. In this work, we propose FlashRetro, an Integral Flow Matching (IFM) framework with Latent Inversion for efficient retrosynthesis prediction. FlashRetro learns finite-interval latent transport rather than local instantaneous velocity, directly matching the transport quantity required for one-step latent inversion and avoiding test-time numerical integration. For inference, FlashRetro uses a principled 1-NFE Latent Inversion rule that maps a noise latent conditioned on the product latent to the reactant latent with a single learned integral-transport update. Experimental results on standard benchmarks (USPTO-50K) show that FlashRetro achieves superior top@1 accuracy, outperforming prior discrete flow-matching methods by 13.5% and diffusion-based baselines by 14.2%, while its average inference time is only 5.1 ms per reactant—38$\times$ faster than multi-step diffusion-based models and nearly 22$\times$ faster than recent discrete flow models. These results show that FlashRetro provides an effective path toward highly efficient 1-NFE retrosynthesis prediction with flow-based models.

1 Introduction

Retrosynthesis prediction is the task to predict the set of reactant molecules that can synthesize a given target product molecule through a single chemical transformation Strieth-Kalthoff et al. [2020], Corey and Cheng. Generating accurate, diverse, and chemically feasible reactant sets that align with real-world synthesis pathways is a crucial task for computational chemistry because many drug discovery and material design processes rely on efficient retrosynthetic analysis to identify viable precursors Gaiński et al. [2025], Yadav et al. [2025]. In the domain of drug discovery, retrosynthesis prediction is a prerequisite for both retrosynthetic planning Segler et al. [2018], Tu et al. [2023] and automated synthesis platforms Coley et al. [2019].

For established computational chemistry retrosynthesis tools, there is a trade-off between prediction speed and the quality or diversity of generated reactants Somnath et al. [2021], Torren-Peraire et al. [2024]. For example, template-based methods Segler and Waller [2017], Dai et al. [2019] can generate plausible reactants by matching predefined reaction patterns, but this paradigm is limited by the coverage of expert-curated templates and scales poorly with novel chemistries Han et al. [2024]. Template-free methods are more flexible but may generate invalid or low-diversity reactants, and their performance deteriorates on complex molecules with discontinuous graph structures Yao et al. [2024].

*Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Thus, deep generative models are being sought as potential solutions to improve retrosynthesis prediction quality, while efficient inference remains crucial for practical deployment.

Many earlier works study deep generative formulations for retrosynthesis Shi et al. [2020], Wang et al. [2021], Chen et al. [2023], Zhong et al. [2023], Yao et al. [2024]. However, established cheminformatics tools such as ASKCOS Coley et al. [2019] still have better prediction quality and faster inference compared to early deep learning-based methods. RetroBridge Igashov et al. [2024] is the first Markov bridge model to achieve better generation quality than template-based baselines. More recent works, such as RetroSynFlow Yadav et al. [2025], RetroDiff Wang et al. [2023], and GDiffRetro Sun et al. [2025], perform diffusion or flow-matching directly on molecular graphs or with additional priors. With more scalable graph architectures, they have achieved the state-of-the-art retrosynthesis quality. However, iterative ODE or SDE solving with large graph models to generate every reactant set can still be computationally infeasible when scaling to multi-step planning or high-throughput virtual screening Omar et al. [2021] involving millions of compounds Zhong et al. [2024], Torren-Peraire et al. [2024].

In this work, we target a central limitation of recent probabilistic retrosynthesis prediction models: while diffusion and flow-matching generators achieve strong accuracy and diversity, their iterative sampling makes them computationally expensive and difficult to scale in high-throughput settings and multi-step planning. In retrosynthesis, the generated latent must be decoded into discrete molecular graph attributes, including atom types, bond types, charges, and aromaticity. Therefore, a small endpoint error in latent space can cross a molecular decoding boundary and change the predicted reactant graph. This makes large-step local velocity integration fragile for one-step retrosynthesis. We propose FlashRetro, an Integral Flow Matching (IFM) framework with Latent Inversion for efficient one-step retrosynthesis prediction. Instead of learning only a time-local velocity field, FlashRetrolearns the interval-normalized integral velocity used by the inference rule. At test time, FlashRetro applies Latent Inversion to use this learned integral transport once and obtain the reactant latent, thereby avoiding iterative ODE/SDE integration while preserving the high-confidence predictions needed for retrosynthetic planning.

Our main contributions are summarized as follows: **(i)** We introduce FlashRetro, an IFM-based retrosynthesis framework that learns finite-interval integral transport in latent space and aligns the training objective with one-step retrosynthesis inference. **(ii)** We propose Latent Inversion, a 1-NFE inference rule that infers the predicted reactant latent from noise under product conditioning, without test-time numerical integration. **(iii)** The IFM-based finite-interval transport objective and Latent Inversion jointly enable fast retrosynthesis prediction with superior top-1 accuracy, improving the practicality of generative retrosynthesis by effectively balancing high accuracy with the efficiency required for high-throughput applications.

2 Related Works

Retrosynthesis Modeling Retrosynthesis prediction is commonly categorized into template-based, semi-template, and template-free approaches. Template-based methods retrieve reaction templates from large databases using hand-crafted rules or learned neural scorers Coley et al. [2017], Dai et al. [2019], Somnath et al. [2021], achieving high validity but suffering from limited template coverage and diminished generalization to rare reactions. Template-free methods instead model retrosynthesis as a direct generation process from products to reactants, using sequence-based transformers Wan et al. [2022], Tu and Coley [2022] or graph neural networks Shi et al. [2020], Somnath et al. [2021]. While more flexible, these models often lack explicit chemical priors, resulting in invalid or low-plausibility outputs. Semi-template decouple reaction center identification from synthon completion Yan et al. [2020], Wang et al. [2021], Chen et al. [2023], improving interpretability and validity through a staged design. Recently, probabilistic generative models based on Markov bridges, diffusion, and flow matching have achieved state-of-the-art performance in template-free retrosynthesis Igashov et al. [2024], Yadav et al. [2025], Wang et al. [2023], Sun et al. [2025], but their reliance on iterative sampling incurs substantial computational cost, limiting scalability.

Flow Matching Flow Matching [Lipman et al., 2023, Albergo and Vanden-Eijnden, 2023] provides a continuous-time generative framework that transforms samples from a tractable base distribution p_0 (e.g., Gaussian noise) to a target data distribution p_1 by specifying a probability path $\{p_t(\mathbf{x}_t)\}_{t\in[0,1]}$. The path is induced by a flow map ψ_t such that $p_t = (\psi_t)_* p_0$, where $(\psi_t)_*$ denotes the pushforward

operator. In practice, ψ_t is parameterized via an ODE with a learnable velocity field, $\frac{d\mathbf{x}_t}{dt} = \mathbf{v}_\theta(\mathbf{x}_t, t)$. Sampling requires numerically integrating this ODE from $\mathbf{x}_0 \sim p_0$ to $t = 1$, which can impose a high computational cost when many samples are needed. A key advantage of Flow Matching is simulation-free training via a conditional probability path. A key advantage of Flow Matching is simulation-free training via a conditional probability path, which can be formalized using conditional distributions $p_t(\mathbf{x} \mid \mathbf{x}_1)$ and conditional vector fields $\mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1)$, while the unconditional quantities follow by marginalizing over $\mathbf{x}_1$:

$$p_t(\mathbf{x}) = \int p_t(\mathbf{x} \mid \mathbf{x}_1)\, q(\mathbf{x}_1)\, d\mathbf{x}_1, \tag{1}$$

$$\mathbf{u}_t(\mathbf{x}) = \int \mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1)\, \frac{p_t(\mathbf{x} \mid \mathbf{x}_1)\, q(\mathbf{x}_1)}{p_t(\mathbf{x})}\, d\mathbf{x}_1, \tag{2}$$

where q denotes the data distribution. A common instantiation adopts Gaussian conditional paths,

$$p_t(\mathbf{x} \mid \mathbf{x}_1) = \mathcal{N}\big(\mathbf{x};\, \boldsymbol{\mu}_t(\mathbf{x}_1),\, \sigma_t^2(\mathbf{x}_1)\mathbf{I}\big), \tag{3}$$

which can be driven by an affine map $\psi_t(\mathbf{x}) = \sigma_t(\mathbf{x}_1)\mathbf{x} + \boldsymbol{\mu}_t(\mathbf{x}_1)$. Under this choice, the conditional vector field has a closed form:

$$\mathbf{u}_t(\mathbf{x} \mid \mathbf{x}_1) = \frac{\sigma_t'(\mathbf{x}_1)}{\sigma_t(\mathbf{x}_1)}\big(\mathbf{x} - \boldsymbol{\mu}_t(\mathbf{x}_1)\big) + \boldsymbol{\mu}_t'(\mathbf{x}_1). \tag{4}$$

In practice, a common choice is the *linear interpolant* between the base and data latents (often interpreted as an *optimal transport*–style construction; Lipman et al. [2023]), as it is simple and numerically stable.

Efficiency in probabilistic retrosynthesis. Probabilistic retrosynthesis models based on diffusion or flow matching typically rely on iterative ODE/SDE sampling, leading to many function evaluations (NFE), high latency, and sensitivity to solver choices. This cost becomes a bottleneck in high-throughput screening and multi-step synthesis planning. Recent diffusion- and flow-based retrosynthesis models, including RetroDiff Wang et al. [2023], GDiffRetro Sun et al. [2025], and RetroSynFlow Yadav et al. [2025], improve generation quality but still require iterative sampling. FlashRetro instead formulates Integral Flow Matching in continuous latent space, where the model learns finite-interval integral transport rather than local instantaneous velocity. This objective aligns with Latent Inversion, enabling 1-NFE retrosynthesis prediction without constructing or aligning discrete-time intermediate molecular states along the sampling path. Since the generated latent must decode into atom types, bond types, charges, and aromaticity, endpoint-aligned integral transport is especially important for stable discrete molecular graph decoding.

3 Method

3.1 Problem Setup

A chemical reaction can be represented as $\{\boldsymbol{G}_R^i\}_{i=1}^{|R|} \rightarrow \{\boldsymbol{G}_P^j\}_{j=1}^{|P|}$, where $\boldsymbol{G}_R^i$ and $\boldsymbol{G}_P^j$ denote reactant and product molecular graphs, respectively. Following retrosynthesis benchmarks, we assume a single product ($|P| = 1$) and focus on the inverse prediction task: given a product graph $\mathbf{x} \in \mathcal{X}$, the goal is to predict the corresponding reactant set $\{\boldsymbol{G}_R^i\}_{i=1}^{|R|}$. Since multiple reactants can be represented as a disjoint union, we treat the output as a single disconnected reactant graph $\mathbf{y} \in \mathcal{Y}$ and model the conditional distribution $P(\mathbf{y} \mid \mathbf{x})$. We denote the product and reactant random variables by $\mathbf{x} \sim P_\mathcal{X}$ and $\mathbf{y} \sim P_\mathcal{Y}$. Let $\mathbf{z}_x \in \mathcal{Z}$ be the product latent, $\mathbf{z}_1 \in \mathcal{Z}$ the reactant latent, and $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ a Gaussian noise latent.

FlashRetro formulates retrosynthesis as a generative process with two components, as illustrated in Figure 1. First, discrete molecular graphs are projected into a continuous latent space via two independent graph encoders, producing the product latent $\mathbf{z}_x = f_\xi(\mathbf{x})$ and the supervised reactant latent $\mathbf{z}_1 = f_\phi(\mathbf{y})$. Conditioned on $\mathbf{z}_x$ and initialized from noise $\mathbf{z}_0$, we learn an interval-normalized integral transport field $u_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x)$ that transports $\mathbf{z}_0$ to $\mathbf{z}_1$ along a linear path, enabling efficient 1-NFE latent inversion. The resulting latent $\hat{\mathbf{z}}_1$ is then decoded into discrete reactant graphs via a molecule decoder $d_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x)$ with validity-aware post-processing. The following subsections describe Integral Flow Matching, Latent Inversion, and the discrete molecule decoder.

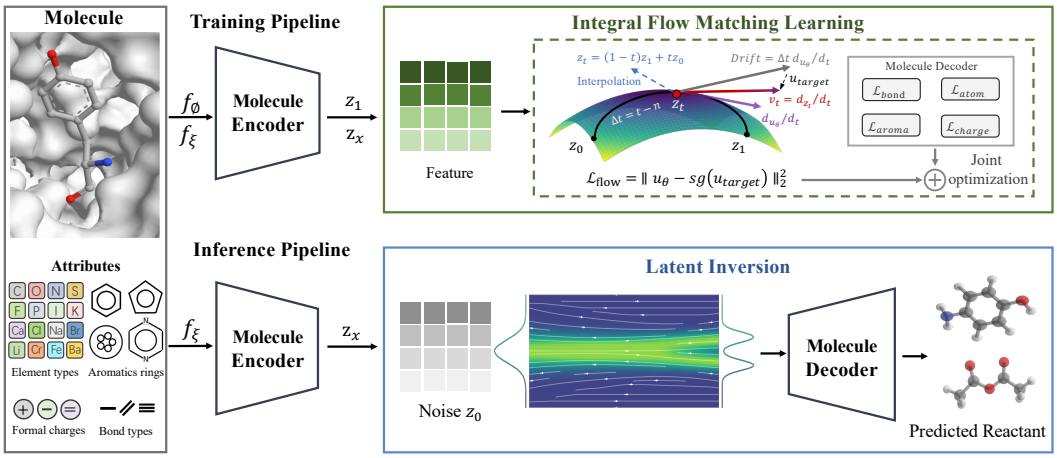


Figure 1: Overview of the FlashRetro pipeline, illustrating the stages of latent encoding, Integral Flow Matching, Latent Inversion, and discrete molecule decoding. IFM learns finite-interval latent transport for 1-NFE retrosynthesis prediction conditioned on the product latent.

3.2 Integral Flow Matching

We first embed discrete molecular graphs into a continuous latent space. Each molecular graph is represented by discrete attributes, including element types, bond types, charges, and aromaticity, with a validity mask M indicating valid reactant nodes. We employ two graph transformer encoders with the same architecture but separate parameters: a reactant encoder f_ϕ and a product encoder f_ξ. The encoders map the reactant graph $\mathbf{y}$ and product graph $\mathbf{x}$ to latent embeddings:

$$\mathbf{z}_1 = f_\phi(\mathbf{y}), \quad \mathbf{z}_x = f_\xi(\mathbf{x}), \quad \text{with } \mathbf{z} \in \mathbb{R}^{L\times D}. \tag{5}$$

The product latent $\mathbf{z}_x$ provides the conditioning signal for latent-space transport. We then learn a conditional interval-normalized integral velocity field to enable 1-NFE latent inversion without ODE/SDE solvers. Let $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and let $\mathbf{z}_1 = f_\phi(\mathbf{y})$ be the supervised reactant latent. Conditioned on the product latent $\mathbf{z}_x$, we define a linear probability path

$$\mathbf{z}_t = (1-t)\mathbf{z}_1 + t\mathbf{z}_0, \quad t \in [0,1]. \tag{6}$$

The instantaneous velocity is given by:

$$\mathbf{v}_t \triangleq \frac{d}{dt}\mathbf{z}_t = \mathbf{z}_0 - \mathbf{z}_1. \tag{7}$$

For any interval $[r,t]$ with $0 \le r \le t \le 1$, define the interval-normalized integral velocity as the finite-interval transport quantity:

$$u(\mathbf{z}_t, r, t) \triangleq \frac{1}{t-r}\int_r^t v(\mathbf{z}_\tau, \tau)\, d\tau, \tag{8}$$

where $v(\cdot,\cdot)$ denotes the instantaneous velocity field. We refer to $u(\mathbf{z}_t, r, t)$ as the interval-normalized integral velocity. Multiplying it by $(t-r)$ gives the integral displacement over the interval $[r,t]$. FlashRetro parameterizes this integral transport field directly by a neural network $u_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x)$, which is conditioned on the product latent $\mathbf{z}_x$.

Equation (8) defines the finite-interval transport quantity used by FlashRetro. Under our linear latent path, the oracle instantaneous velocity and the full-interval normalized transport coincide along a fixed sample path. The IFM objective aligns training with the transport quantity directly queried by the 1-NFE Latent Inversion rule.

Theorem 3.1 (Integral Transport Identity)**.** *Under the linear latent interpolation* $\mathbf{z}_t = (1-t)\mathbf{z}_1 + t\mathbf{z}_0$ *from the reactant latent* $\mathbf{z}_1$ *to the noise latent* $\mathbf{z}_0$*, the oracle full-interval normalized transport coincides with the endpoint displacement required by the one-step inversion from* $\mathbf{z}_0$ *back to* $\mathbf{z}_1$*:*

$$^\star(\mathbf{z}_0, 0, 1) = \mathbf{z}_0 - \mathbf{z}_1, \quad \mathbf{z}_0 -^\star (\mathbf{z}_0, 0, 1) = \mathbf{z}_1. \tag{9}$$

Theorem 3.1 shows that the oracle IFM field provides the finite-interval transport used by 1-NFE Latent Inversion. The proof is provided in Appendix A.1.

Directly regressing the integral definition is impractical. Instead, we apply the local identity obtained by differentiating the integral definition with respect to t:

$$u(\mathbf{z}_t, r, t) = v(\mathbf{z}_t, t) - (t - r)\frac{d}{dt}u(\mathbf{z}_t, r, t). \tag{10}$$

To train u_θ to satisfy Eq. (10) along the chosen path, we regress the identity. We sample (r, t) with $0 \le r \le t \le 1$, draw latent pairs $(\mathbf{z}_0, \mathbf{z}_1)$, and construct the point $\mathbf{z}_t$. We then compute the total derivative of u_θ with respect to time at $(\mathbf{z}_t, r, t \mid \mathbf{z}_x)$ using automatic differentiation:

$$g_\theta \triangleq \mathbf{v}_t \cdot \partial_{\mathbf{z}} u_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x) + \partial_t u_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x), \tag{11}$$

where $\mathbf{v}_t$ denotes the instantaneous velocity $\frac{d\mathbf{z}_t}{dt}$. The regression target is subsequently constructed as

$$u_{\text{tgt}} = \mathbf{v}_t - (t - r)g_\theta. \tag{12}$$

We omit the explicit dependence on $(\mathbf{z}_t, r, t, \mathbf{z}_x)$ in the partial derivatives. We apply stop-gradient to stabilize the bootstrap target and minimize a masked mean-squared error:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t,r,\mathbf{z}_{0,1},\mathbf{z}_x}\left[\|u_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x) - \text{sg}(u_{\text{tgt}})\|_{\boldsymbol{M}}^2\right]. \tag{13}$$

3.3 Latent Inversion

In this section, we propose Latent Inversion, a robust 1-NFE inference and decoding mechanism that maps the terminal state of the latent flow back to the molecular graph manifold $\mathcal{Y}$. Given a predicted reactant latent $\hat{\mathbf{z}}_1$ derived from a single integral-transport update and the product conditioning $\mathbf{z}_x$, the molecule decoder d_ψ reconstructs the discrete graph attributes of $\mathbf{y}$, including element types, bond types, charges, and aromaticity, all under the guidance of the validity mask $\boldsymbol{M}$.

A key design choice of FlashRetro is to use the reactant latent $\mathbf{z}_1 = f_\phi(\mathbf{y})$ as the supervised endpoint for both latent transport learning and molecular reconstruction. At inference, since the ground-truth reactant graph is unavailable, Latent Inversion estimates the corresponding reactant latent from $\mathbf{z}_0$ and $\mathbf{z}_x$ using the learned integral transport field. The Latent Inversion rule used at inference is

$$\hat{\mathbf{u}} = u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x), \qquad \hat{\mathbf{z}}_1 = \mathbf{z}_0 - \hat{\mathbf{u}}. \tag{14}$$

Equation (14) provides a way to measure how the learned finite-interval transport affects the predicted reactant latent. We next analyze how errors in the learned integral transport field translate into endpoint errors in the predicted reactant latent.

Theorem 3.2 (Latent Inversion Error)**.** *For a product condition $\mathbf{z}_x$, suppose the estimated reactant latent is obtained by one-step inversion with the learned IFM field, while the oracle reactant latent is obtained by the exact full-interval transport. Then the endpoint deviation is exactly the IFM field approximation error:*

$$\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2^2 = \|u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)\|_2^2. \tag{15}$$

Thus, Latent Inversion avoids solver-induced discretization error; the remaining test-time errors are governed by integral-field approximation and decoder sensitivity. The proof is provided in Appendix A.2.

The reconstruction loss is formulated as the sum of masked cross-entropies:

$$\mathcal{L}_{\text{recon}} = \mathbb{E}_{(\mathbf{x},\mathbf{y})}\left[\sum_{a\in\mathcal{A}} \text{CE}_{\boldsymbol{M}}\left(d_\psi^{(a)}(\mathbf{z}_1, \mathbf{z}_x), \mathbf{y}^{(a)}\right)\right]. \tag{16}$$

The reconstruction loss in Eq. (16) trains the decoder to map the supervised reactant latent $\mathbf{z}_1$ and product conditioning $\mathbf{z}_x$ to the corresponding molecular attributes. We therefore analyze when the latent approximation error preserves the same discrete molecular attributes after decoding.

Theorem 3.3 (Molecular Attribute Stability)**.** *For each molecular attribute head $a \in \mathcal{A}$, let $\gamma^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})$ denote the logit margin between the correct attribute and the strongest incorrect attribute. If the decoder logits for head a are L_a-Lipschitz in $\mathbf{z}$, then the attribute-wise argmax predictions are unchanged whenever*

$$\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2 < \min_{a \in \mathcal{A}} \frac{\gamma^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})}{2L_a}. \tag{17}$$

Theorem 3.3 characterizes the molecular decoding basin around the reactant latent. It applies to atom types, bond types, charges, aromaticity, and other discrete attribute heads decoded from the latent representation. The proof is provided in Appendix A.3.

3.4 Overall Training Objective

FlashRetro jointly optimizes the molecule encoders, latent transport, and discrete molecule decoder:

$$\mathcal{L}_{\text{total}}(\phi, \xi, \theta, \psi) = \mathcal{L}_{\text{flow}}(\phi, \xi, \theta) + \mathcal{L}_{\text{recon}}(\phi, \xi, \psi). \tag{18}$$

The total objective in Eq. (18) combines decoder reconstruction with finite-interval latent transport learning. We next show that these two terms jointly control the inference-time decoding risk of Latent Inversion.

Theorem 3.4 (Risk Bound for Latent Inversion)**.** *Let $\ell_\psi(\mathbf{z}, \mathbf{z}_x; \mathbf{y}) = -\log p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{z}_x)$ be the decoder negative log-likelihood. If ℓ_ψ is locally Lipschitz in $\mathbf{z}$, then the expected decoding risk of the inferred latent is bounded by the reconstruction risk at the reactant latent plus a term proportional to the latent inversion error:*

$$\mathbb{E}[\ell_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x; \mathbf{y})] \leq \mathbb{E}[\ell_\psi(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})] + C\,\mathbb{E}\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{19}$$

Theorem 3.4 explains the role of the two training losses. The reconstruction term controls the decoder risk at the reactant latent, while the IFM loss controls the latent inversion error through Theorem 3.2. Thus, the joint objective aligns finite-interval transport learning with the decoding objective used for 1-NFE retrosynthesis prediction. The proof is provided in Appendix A.4.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct all experiments on the USPTO-50K benchmark [Lowe, 2017], a standard retrosynthesis prediction dataset of approximately 50K atom-mapped reactions extracted from US patents. To ensure comparability with prior retrosynthesis studies, we adopt the same data split protocol as commonly used in earlier work [Yadav et al., 2025, Igashov et al., 2024, Dai et al., 2019].

Baselines. Baselines are selected as follows: (i)Template-based methods. We include MHNreact [Seidl et al., 2022], GLN [Dai et al., 2019], LocalRetro [Chen and Jung, 2021], GraphRetro [Somnath et al., 2021], RetroComposer [Yan et al., 2022], and Dual-TB [Sun et al., 2020]. (ii)Semi-template and graph-edit methods. We compare with MEGAN [Sacha et al., 2021], G2Gs [Shi et al., 2020], RetroXpert [Yan et al., 2020], G2Retro [Chen et al., 2023], RetroPrime [Wang et al., 2021], and RetroDiff [Wang et al., 2023]. (iii)Template-free methods. These methods serve as our primary baselines, as they also perform end-to-end, template-free reactant generation. We consider Transformer [Vaswani et al., 2017], SCROP [Zheng et al., 2019], Retroformer [Wan et al., 2022], GTA [Seo et al., 2021], Graph2SMILES [Tu and Coley, 2022], Transformer (Aug.) [Tetko et al., 2020], Dual-TF [Sun et al., 2020] Chemformer [Irwin et al., 2022], RetroBridge [Igashov et al., 2024] and RetroSynFlow [Yadav et al., 2025].

Evaluation Metrics. We use top-k exact match accuracy as the evaluation metric, defined as the fraction of test reactions where the ground-truth reactant set is among the model's top-k predicted sets [Yadav et al., 2025]. As our flow-based method does not natively produce beam-ranked outputs, we follow RetroBridge [Igashov et al., 2024] by generating candidate reactant sets per test reaction and selecting the top-k predictions based on model scores.

Implementation. We train FlashRetro on USPTO-50K using AdamW (lr=1×10^{-4}, effective batch size 256). The model trains for up to 200,000 iterations with cosine decay scheduling (2,000-step warmup, minimum lr 1×10^{-6}) and gradient clipping (norm 1.0). The backbone uses hidden dimension $d_{\text{model}} = 256$ and 8 attention heads. The encoder has 6 layers, the decoder-only molecular decoder 12 layers, and the IFM integral-transport network 6 layers. The latent flow uses linear $\kappa(t) = t$ and square-root $\sigma(t) = \sqrt{t(1-t)} \cdot \sigma_{\max}$ schedules. The IFM component follows a linear latent interpolation path with logit-normal time sampling ($\mu = -0.4$, $\sigma = 1.0$) and positive-span ($\Delta = t - r > 0$) interval sampling probability $p_{\text{span}} = 0.25$.

4.2 Main Results

Accuracy. We further evaluate the performance of FlashRetro against a broad range of retrosynthesis baselines on the standard USPTO-50K test set. Table 1 reports Top-k exact-match accuracy across template-based, semi-template, and template-free methods. Our approach achieves the highest Top-1 accuracy (69.3%) and Top-3 accuracy (78.4%), substantially outperforming all prior template-free methods and surpassing strong semi-template baselines such as GDiffRetro and the semi-template variant of RetroSynFlow on Top-1/Top-3. On higher-k metrics, FlashRetro remains competitive, attaining 79.8% Top-5 and 81.1% Top-10 accuracy, while the best reported Top-5 and Top-10 scores among template-free approaches are achieved by the template-free variant of RetroSynFlow (81.2% and 85.8%, respectively). Overall, FlashRetro is strongest in the low-latency Top-1/Top-3 regime, while its gains at higher k are smaller than the best high-k template-free results.

Table 1: Main results on USPTO-50k benchmark. Comparison of Top-k ($k = 1, 3, 5$) exact match accuracy across different retrosynthesis paradigms. Evaluation covers both unknown and known reaction class settings. Best results within each category are highlighted in **bold**.

	Baselines	Top-k accuracy (unknown class)			Top-k accuracy (known class)		
		k=1	k=3	k=5	k=1	k=3	k=5
Template-Based	MHNreact [Seidl et al., 2022]	51.8	74.6	81.2	-	-	-
	GLN [Dai et al., 2019]	52.5	69.0	75.6	64.2	79.1	85.2
	LocalRetro [Chen and Jung, 2021]	53.4	**77.5**	**85.9**	63.9	**86.8**	**92.4**
	GraphRetro [Somnath et al., 2021]	53.7	68.3	72.2	63.9	81.5	85.2
	RetroComposer [Yan et al., 2022]	54.5	77.2	83.2	65.9	85.8	89.5
	Dual-TB [Sun et al., 2020]	**55.2**	74.6	80.5	**67.7**	84.8	88.9
Semi-Template	MEGAN [Sacha et al., 2021]	48.1	70.7	78.4	60.7	82.0	**87.5**
	G2Gs [Shi et al., 2020]	48.9	67.6	72.5	61.0	81.3	86.0
	RetroXpert [Yan et al., 2020]	50.4	61.1	62.3	62.1	75.8	78.5
	G^2Retro [Chen et al., 2023]	51.4	72.1	78.2	59.4	79.4	84.2
	RetroPrime [Wang et al., 2021]	51.4	70.8	74.0	64.8	82.7	85.0
	RetroDiff [Wang et al., 2023]	52.6	71.2	81.0	-	-	-
	GDiffRetro [Sun et al., 2025]	58.9	**79.1**	81.9	**67.6**	**84.1**	86.1
	RetroSynFlow [Yadav et al., 2025]	**60.0**	77.9	**82.7**	-	-	-
Template-Free	Transformer [Vaswani et al., 2017]	43.7	59.7	65.1	-	-	-
	SCROP [Zheng et al., 2019]	43.7	60.0	65.2	59.0	74.8	78.1
	Retroformer [Wan et al., 2022]	52.9	68.2	72.5	64.0	**82.5**	**86.7**
	Graph2SMILES [Tu and Coley, 2022]	52.9	66.5	70.0	-	-	-
	Dual-TF [Sun et al., 2020]	53.6	70.7	74.6	65.7	81.9	84.7
	RetroBridge [Igashov et al., 2024]	50.8	74.1	80.6	-	-	-
	RetroProdFlow [Yadav et al., 2025]	50.2	74.3	**81.2**	-	-	-
	FlashRetro	**69.3**	**78.4**	79.8	**71.8**	80.5	82.1

Faster Convergence and Data-Efficient Learning. To demonstrate the advantage of IFM over prior retrosynthesis objectives, we evaluate models on the standard USPTO-50K test set against RetroSynFlow and GDiffRetro. We report accuracy by epoch in the left column and by percentage of training data used in the right column. Fig. 2 shows that FlashRetro consistently outperforms both baselines, with the largest gain in Top-1 accuracy. On the primary Top-1 metric, FlashRetro after 40 epochs surpasses RetroSynFlow and GDiffRetro trained for 100 epochs. For data efficiency, FlashRetro exceeds RetroSynFlow's final Top-1 performance using less than 20% of the full data. For higher-k metrics, such as Top-10, FlashRetro maintains a clear lead over GDiffRetro, reaches its peak performance with roughly 30% of the data, and consistently outperforms RetroSynFlow at convergence. We provide additional wall-clock training time and cumulative-compute comparisons in

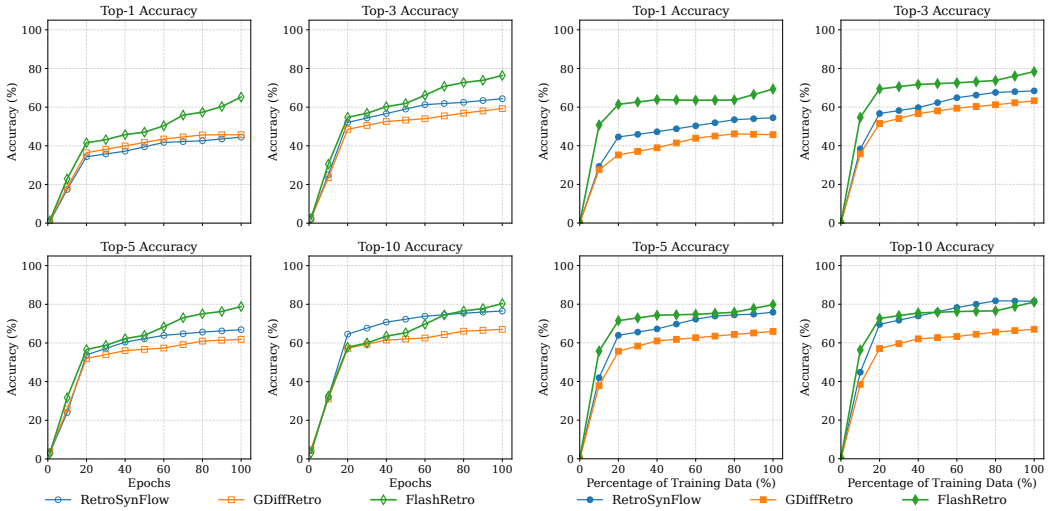


Figure 2: Comparison of retrosynthesis accuracy across training progress. Both RetroSynFlow, GDiffRetro, and FlashRetro are evaluated on Top-k accuracy metrics. FlashRetro converges to superior performance with faster convergence speed across all Top-1, Top-3, and Top-5 metrics.

Appendix D.2. Overall, IFM training enables faster convergence and superior retrosynthesis accuracy, especially for challenging Top-1 predictions.

Sampling Time. To evaluate FlashRetro's efficiency, we compare its 1-NFE Latent Inversion latency with strong retrosynthesis baselines on USPTO-50K. All timings are measured on a single NVIDIA A800 GPU with batch size 1. Figure 3 shows that FlashRetro achieves higher Top-1 accuracy than the compared baselines while maintaining low inference latency. With 55.2M parameters and 0.51 seconds per 100 reactants, FlashRetro reaches ∼69% Top-1 accuracy, offering a superior trade-off among capacity, accuracy, and cost. Its efficiency excels against iterative flow- and diffusion-based methods requiring multiple ODE/SDE integrations. Compared with RetroSynFlow (3.4M parameters, 11.54s/100 reactants, ∼63% accuracy), FlashRetro is 22.6× faster and gains 7 percentage points in Top-1 accuracy. Against GDiffRetro (16.0M parameters, 19.48s/100 reactants, ∼60% accuracy), it is ∼38.2× faster with a ∼10 percentage point Top-1 lead. This speedup stems from IFM learning finite-interval transport and Latent Inversion performing one-step inference, bypassing iterative sampling. For high-throughput screening, such gains allow evaluating more candidates per unit time. Overall, FlashRetro boosts sampling efficiency while maintaining strong accuracy.

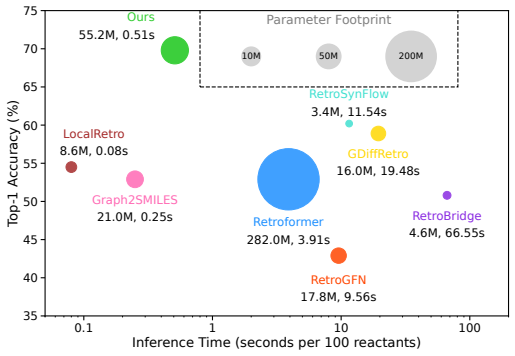


Figure 3: Accuracy–latency Pareto comparison for retrosynthesis prediction.

4.3 Analysis

Ablation Study. As shown in Table 2, we ablate FlashRetro components under unified USPTO-50K evaluation. Local Flow Matching with multi-step ODE sampling reaches 49.5% Top-1 (row 1). Applying the 1-NFE Latent Inversion rule to a Local-FM field drops this to roughly 25% (row 2), as inference requires an interval-integrated transport quantity unavailable from local instantaneous velocity. Integral Flow Matching using only zero-span intervals (Δ=0, row 3) reduces the target to instantaneous velocity, recovering 60.51% Top-1; thus, IFM without positive-span interval sampling is training-equivalent to local FM. Enabling positive-span interval sampling (p_{span}=0.25) without

stop-gradient on the bootstrap target (row 4) destabilizes the auto-differentiated identity in Eq. (10), degrading Top-1 by 10 points. A two-stage frozen-latent pipeline (row 5) yields 52.4%, proving end-to-end co-adaptation ensures decoder stability. The final configuration (row 7) attains 69.30% Top-1. Thus, IFM, positive-span interval stabilization, and 1-NFE Latent Inversion are jointly necessary for FlashRetro's endpoint-alignment guarantee.

Table 2: FlashRetro USPTO-50K ablation. Vel. Field: local instantaneous (Local) vs. finite-interval integral velocity (Integral). Inference: N-step ODE (ODE-N) vs. one-step Latent Inversion (1-NFE LI). p_{span}: positive-span interval ($\Delta = t - r > 0$) sampling probability. BS-sg: bootstrap target stop-gradient. Train: end-to-end (E2E) vs. two-stage frozen-latent (TS).

Framework	Vel. Field	Inference	p_{span}	BS-sg	Train	Top-1	Top-3	Top-5	Top-10
Local-FM	Local	ODE-100	–	–	E2E	49.50	72.30	80.30	85.60
Local-FM + 1-NFE	Local	1-NFE LI	–	–	E2E	25.10	39.80	46.70	52.40
IFM (Δ=0, no bootstrap)	Integral	1-NFE LI	0	✓	E2E	60.51	69.26	71.37	72.97
IFM w/o stop-gradient	Integral	1-NFE LI	0.25	×	E2E	58.20	68.10	70.40	72.30
IFM, two-stage frozen latent	Integral	1-NFE LI	0.25	✓	TS	52.40	66.50	70.20	73.80
FlashRetro (ODE inference)	Integral	ODE-100	0.25	✓	E2E	**69.44**	78.21	79.74	81.06
FlashRetro (*final*)	**Integral**	**1-NFE LI**	**0.25**	✓	**E2E**	69.30	**78.40**	**79.80**	**81.10**

Effect of ODE/SDE steps on performance. Fig. 4 analyzes FlashRetro with ODE and SDE samplers using $N_{\text{step}} \in \{1, 2, 5, 10, 30, 50, 100\}$. Once IFM learns finite-interval transport, additional ODE/SDE solver steps do not provide a consistent accuracy gain while increasing inference cost. For example, the ODE sampler achieves 69.46% Top-1 accuracy at $N_{\text{step}} = 2$, which is comparable to our 1-NFE Latent Inversion result. This supports the alignment between IFM training and the 1-NFE inference rule: FlashRetro is not merely a fast approximation of a multi-step sampler, but is trained to predict the finite-interval transport needed by Latent Inversion. Although the ODE sampler is stable in the low-step regime ($N_{\text{step}} < 10$), its Top-5 accuracy slightly degrades when N_{step} exceeds 10. SDE sampling can maintain higher validity on larger candidate sets, with Top-5 validity above 99.0% across most step counts, whereas ODE Top-5 validity fluctuates between 88% and 90%. This suggests that stochasticity may help preserve structural validity for diverse candidates. Overall, extra solver steps do not yield stable accuracy improvements, and 1-NFE Latent Inversion provides the best accuracy–latency trade-off for high-confidence retrosynthesis prediction.

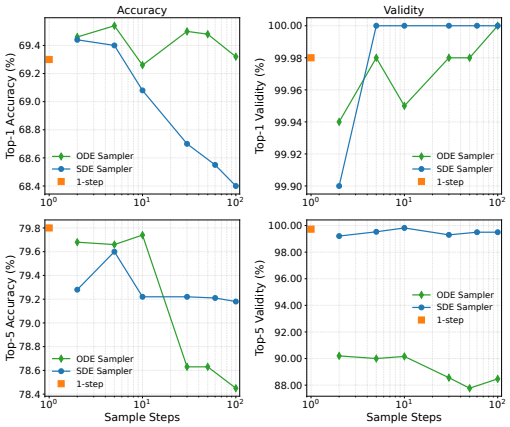


Figure 4: Effect of sampling strategy and inference steps. Comparison of ODE, SDE, and 1-NFE Latent Inversion across Top-k accuracy and validity.

Effect of positive-span interval sampling probability. We examine how positive-span interval sampling probability p_{span} affects 1-step Flow Matching performance; positive spans satisfy $\Delta = t - r > 0$. Setting $p_{\text{span}} = 0$ samples only zero-span intervals ($\Delta = 0$), recovering the standard Flow Matching limit. Tab. 3 shows moderate positive-span sampling significantly improves performance. $p_{\text{span}} = 0.25$ is best across metrics, raising Top-1 accuracy from 60.51% to 69.30% (+8.79 points) and Top-10 from 72.97% to 81.10% (+8.13 points). Metric-wide gains indicate that a small finite-span interval training probability improves 1-step retrieval quality and candidate-set robustness. Larger probabilities are harmful: at $p_{\text{span}} = 0.75$, Top-1 accuracy drops sharply to 36.00% (24.51 points below the standard Flow Matching baseline), then collapses at $p_{\text{span}} = 1.00$ (20.74% Top-1). Thus, overusing these samples substantially degrades 1-step performance. Overall, moderate sampling improves accuracy, whereas excessive sampling causes severe deterioration, highlighting the importance of controlling p_{span} for 1-step inference.

Table 3: Effect of positive-span interval sampling probability p_{span} in 1-step Flow Matching. The setting $p_{\text{span}} = 0$ uses only zero-span intervals ($\Delta = 0$) and reduces to the standard Flow Matching limit. Results report top-k exact match accuracy (%).

p_{span}	Top-1 (%)	Top-3 (%)	Top-5 (%)	Top-10 (%)
0 ($\Delta = 0$)	60.51	69.26	71.37	72.97
0.25	**69.30**	**78.40**	**79.80**	**81.10**
0.75	36.00	58.41	64.70	67.60
1.00	20.74	31.38	34.16	36.04

NFE Analysis. We evaluate how predictive performance changes with the number of function evaluations (NFE). This comparison isolates the effect of test-time solver discretization rather than conflating it with model capacity. As shown in Table 4, diffusion- and bridge-based baselines strongly depend on iterative sampling. RetroBridge collapses under 1–2 steps, and RetroSynFlow drops from 49% to 28% Top-1 when reducing steps from 50 to 5. In contrast, FlashRetro remains stable across NFEs: its 1-step Top-1 accuracy of 69.3% nearly matches the 100-step result, indicating that the single integral update already captures the endpoint transport needed for decoding. This supports the alignment between IFM training and 1-NFE Latent Inversion. Once FlashRetro learns finite-interval transport, additional ODE/SDE solver steps do not provide a consistent accuracy gain while increasing inference cost. This stability also shows that FlashRetro is not merely a fast approximation of a multi-step sampler; rather, its training objective is designed for the one-step inference rule.

Table 4: Top-k exact match accuracy under different sampling steps (n) on USPTO-50K.

Method	Steps (n)	Top-1	Top-3	Top-5	Top-10
Multi-step Iterative Sampling					
RetroBridge [Igashov et al., 2024]	500	26.83	43.40	50.26	59.68
GDiffRetro [Sun et al., 2025]	100	58.90	79.10	81.90	-
FlashRetro	100	69.44	78.21	79.74	81.06
RetroSynFlow [Yadav et al., 2025]	50	49.45	72.30	80.25	85.63
FlashRetro	50	68.84	77.49	78.63	79.74
RetroSynFlow	10	46.20	61.04	66.25	69.86
FlashRetro	10	68.84	77.49	78.63	79.74
RetroSynFlow	5	28.00	42.00	48.00	52.00
FlashRetro	5	69.52	78.30	79.54	80.82
Two-step Sampling					
RetroBridge	2	0.04	0.04	0.04	0.18
FlashRetro	2	69.46	78.23	79.68	80.86
One-step Sampling					
RetroBridge	1	0.02	0.02	0.02	0.02
FlashRetro	1	69.30	78.40	79.80	81.10

5 Conclusion

We present FlashRetro, an Integral Flow Matching framework with Latent Inversion for efficient retrosynthesis prediction. FlashRetro learns finite-interval integral transport in a continuous latent space, aligning training with the 1-NFE inference rule. This design suits retrosynthesis, where endpoint errors in latent space can alter discrete molecular graph attributes. At inference time, Latent Inversion infers the reactant latent from noise under product conditioning with one learned integral-transport update, avoiding iterative ODE/SDE sampling and solver-induced discretization. Through extensive experiments, we demonstrate that our approach improves data efficiency, matching or surpassing baseline performance using only 60% of the training data. Finally, FlashRetro provides a 22×–38× speedup in wall-clock inference time compared to iterative flow and diffusion-based baselines. Overall, our work improves the practicality of generative retrosynthesis by balancing high accuracy with the efficiency required for high-throughput applications.

6 Acknowledgments

This work is supported by Science and Technology Innovation Key R&D Program of Chongqing (CSTB2024TIAD-STX0032).

References

Michael Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *ICLR 2023 Conference*, 2023.

Shuan Chen and Yousung Jung. Deep retrosynthetic reaction prediction using local reactivity and global attention. *JACS Au*, 1(10):1612–1620, 2021.

Ziqi Chen, Oluwatosin R Ayinde, James R Fuchs, Huan Sun, and Xia Ning. G 2 retro as a two-step graph generative models for retrosynthesis prediction. *Communications Chemistry*, 6(1):102, 2023.

Connor W Coley, Luke Rogers, William H Green, and Klavs F Jensen. Computer-assisted retrosynthesis based on molecular similarity. *ACS central science*, 3(12):1237–1245, 2017.

Connor W Coley, Dale A Thomas III, Justin AM Lummiss, Jonathan N Jaworski, Christopher P Breen, Victor Schultz, Travis Hart, Joshua S Fishman, Luke Rogers, Hanyu Gao, et al. A robotic platform for flow synthesis of organic compounds informed by ai planning. *Science*, 365(6453): eaax1566, 2019.

EJ Corey and Xue-Min Cheng. The logic of chemical synthesis.

Hanjun Dai, Chengtao Li, Connor Coley, Bo Dai, and Le Song. Retrosynthesis prediction with conditional graph logic network. *Advances in Neural Information Processing Systems*, 32, 2019.

Piotr Gaiński, Michał Koziarski, Krzysztof Maziarz, Marwin Segler, Jacek Tabor, and Marek Śmieja. Diverse and feasible retrosynthesis using gflownets. *Information Sciences*, 714:122194, 2025.

Yuqiang Han, Xiaoyang Xu, Chang-Yu Hsieh, Keyan Ding, Hongxia Xu, Renjun Xu, Tingjun Hou, Qiang Zhang, and Huajun Chen. Retrosynthesis prediction with an iterative string editing model. *Nature Communications*, 15(1):6404, 2024.

Ilia Igashov, Arne Schneuing, Marwin H. S. Segler, Michael M. Bronstein, and Bruno E. Correia. Retrobridge: Modeling retrosynthesis with markov bridges. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=770DetV8He`.

Ross Irwin, Spyridon Dimitriadis, Jiazhen He, and Esben Jannik Bjerrum. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1): 015022, 2022.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Daniel Lowe. Chemical reactions from us patents, 2017.

Ömer H Omar, Marcos Del Cueto, Tahereh Nematiaram, and Alessandro Troisi. High-throughput virtual screening for organic electronics: a comparative study of alternative strategies. *Journal of Materials Chemistry C*, 9(39):13557–13583, 2021.

Mikołaj Sacha, Mikołaj Błaz, Piotr Byrski, Paweł Dabrowski-Tumanski, Mikołaj Chrominski, Rafał Loska, Paweł Włodarczyk-Pruszynski, and Stanisław Jastrzebski. Molecule edit graph attention network: modeling chemical reactions as sequences of graph edits. *Journal of Chemical Information and Modeling*, 61(7):3273–3284, 2021.

Marwin HS Segler and Mark P Waller. Neural-symbolic machine learning for retrosynthesis and reaction prediction. *Chemistry–A European Journal*, 23(25):5966–5971, 2017.

Marwin HS Segler, Mike Preuss, and Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic ai. *Nature*, 555(7698):604–610, 2018.

Philipp Seidl, Philipp Renz, Natalia Dyubankova, Paulo Neves, Jonas Verhoeven, Jorg K Wegner, Marwin Segler, Sepp Hochreiter, and Gunter Klambauer. Improving few-and zero-shot reaction template prediction using modern hopfield networks. *Journal of chemical information and modeling*, 62(9):2111–2120, 2022.

Seung-Woo Seo, You Young Song, June Yong Yang, Seohui Bae, Hankook Lee, Jinwoo Shin, Sung Ju Hwang, and Eunho Yang. Gta: Graph truncated attention for retrosynthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 531–539, 2021.

Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. A graph to graphs framework for retrosynthesis prediction. In *International conference on machine learning*, pages 8818–8827. PMLR, 2020.

Vignesh Ram Somnath, Charlotte Bunne, Connor Coley, Andreas Krause, and Regina Barzilay. Learning graph models for retrosynthesis prediction. *Advances in Neural Information Processing Systems*, 34:9405–9415, 2021.

Felix Strieth-Kalthoff, Frederik Sandfort, Marwin HS Segler, and Frank Glorius. Machine learning the ropes: principles, applications and directions in synthetic chemistry. *Chemical Society Reviews*, 49(17):6154–6168, 2020.

Ruoxi Sun, Hanjun Dai, Li Li, Steven Kearnes, and Bo Dai. Energy-based view of retrosynthesis. *arXiv preprint arXiv:2007.13437*, 2020.

Shengyin Sun, Wenhao Yu, Yuxiang Ren, Weitao Du, Liwei Liu, Xuecang Zhang, Ying Hu, and Chen Ma. Gdiffretro: Retrosynthesis prediction with dual graph enhanced molecular representation and diffusion generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12595–12603, 2025.

Igor V Tetko, Pavel Karpov, Ruud Van Deursen, and Guillaume Godin. State-of-the-art augmented nlp transformer models for direct and single-step retrosynthesis. *Nature communications*, 11(1): 5575, 2020.

Paula Torren-Peraire, Alan Kai Hassen, Samuel Genheden, Jonas Verhoeven, Djork-Arné Clevert, Mike Preuss, and Igor V Tetko. Models matter: the impact of single-step retrosynthesis on synthesis planning. *Digital Discovery*, 3(3):558–572, 2024.

Zhengkai Tu and Connor W Coley. Permutation invariant graph-to-sequence model for template-free retrosynthesis and reaction prediction. *Journal of chemical information and modeling*, 62(15): 3503–3513, 2022.

Zhengkai Tu, Thijs Stuyver, and Connor W Coley. Predictive chemistry: machine learning for reaction deployment, reaction development, and reaction discovery. *Chemical science*, 14(2): 226–244, 2023.

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

Yue Wan, Chang-Yu Hsieh, Ben Liao, and Shengyu Zhang. Retroformer: Pushing the limits of end-to-end retrosynthesis transformer. In *International Conference on Machine Learning*, pages 22475–22490. PMLR, 2022.

Xiaorui Wang, Yuquan Li, Jiezhong Qiu, Guangyong Chen, Huanxiang Liu, Benben Liao, Chang-Yu Hsieh, and Xiaojun Yao. Retroprime: A diverse, plausible and transformer-based method for single-step retrosynthesis predictions. *Chemical Engineering Journal*, 420:129845, 2021.

Yiming Wang, Yuxuan Song, Yiqun Wang, Minkai Xu, Rui Wang, Hao Zhou, and Wei-Ying Ma. Retrodiff: Retrosynthesis as multi-stage distribution interpolation. *arXiv preprint arXiv:2311.14077*, 2023.

Robin Yadav, Qi Yan, Guy Wolf, Avishek Joey Bose, and Renjie Liao. Retro synflow: Discrete flow matching for accurate and diverse single-step retrosynthesis. *arXiv preprint arXiv:2506.04439*, 2025.

Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, Jinyu Yang, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like a chemist. *Advances in Neural Information Processing Systems*, 33:11248–11258, 2020.

Chaochao Yan, Peilin Zhao, Chan Lu, Yang Yu, and Junzhou Huang. Retrocomposer: composing templates for template-based retrosynthesis prediction. *Biomolecules*, 12(9):1325, 2022.

Lin Yao, Wentao Guo, Zhen Wang, Shang Xiang, Wentan Liu, and Guolin Ke. Node-aligned graph-to-graph: elevating template-free deep learning approaches in single-step retrosynthesis. *JACS Au*, 4(3):992–1003, 2024.

Shuangjia Zheng, Jiahua Rao, Zhongyue Zhang, Jun Xu, and Yuedong Yang. Predicting retrosynthetic reactions using self-corrected transformer neural networks. *Journal of chemical information and modeling*, 60(1):47–55, 2019.

Weihe Zhong, Ziduo Yang, and Calvin Yu-Chian Chen. Retrosynthesis prediction using an end-to-end graph generative architecture for molecular graph editing. *Nature Communications*, 14(1):3009, 2023.

Zipeng Zhong, Jie Song, Zunlei Feng, Tiantao Liu, Lingxiang Jia, Shaolun Yao, Tingjun Hou, and Mingli Song. Recent advances in deep learning for retrosynthesis. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 14(1):e1694, 2024.

A Proofs of Theoretical Results

Remark on the linear latent path. Under the linear latent path used in this work, the oracle interval-normalized transport coincides with the endpoint displacement along any non-degenerate interval. Therefore, the following analysis should be interpreted as an endpoint-alignment argument for the proposed 1-NFE Latent Inversion rule, rather than as a claim that the oracle integral target is fundamentally different from the oracle local velocity target under this particular path. The empirical benefit of IFM comes from its finite-interval self-consistency objective and its direct alignment with the full-interval inversion rule used at inference. For the degenerate case $r = t$, we define the interval-normalized integral velocity by continuity:

$$u^\star(\mathbf{z}_t, t, t) \triangleq \lim_{\epsilon \to 0^+} \frac{1}{\epsilon} \int_t^{t+\epsilon} v^\star(\mathbf{z}_\tau, \tau)\, d\tau = v^\star(\mathbf{z}_t, t), \tag{20}$$

whenever the limit exists.

A.1 Proof of Theorem 3.1

Proof. According to the linear latent interpolation used in Integral Flow Matching, we have

$$\mathbf{z}_\tau = (1-\tau)\mathbf{z}_1 + \tau \mathbf{z}_0, \qquad \tau \in [0,1]. \tag{21}$$

Taking the derivative with respect to τ, the oracle instantaneous velocity along this path is

$$v^\star(\mathbf{z}_\tau, \tau) = \frac{d\mathbf{z}_\tau}{d\tau} = \mathbf{z}_0 - \mathbf{z}_1. \tag{22}$$

Therefore, for any interval $0 \le r < t \le 1$, the oracle interval-normalized integral velocity is

$$u^\star(\mathbf{z}_t, r, t) = \frac{1}{t-r} \int_r^t v^\star(\mathbf{z}_\tau, \tau)\, d\tau \tag{23}$$

$$= \frac{1}{t-r} \int_r^t (\mathbf{z}_0 - \mathbf{z}_1)\, d\tau \tag{24}$$

$$= \mathbf{z}_0 - \mathbf{z}_1. \tag{25}$$

In particular, over the full interval $[0, 1]$, we have

$$u^\star(\mathbf{z}_0, 0, 1) = \mathbf{z}_0 - \mathbf{z}_1. \tag{26}$$

Applying the oracle one-step inversion rule gives

$$\mathbf{z}_0 - u^\star(\mathbf{z}_0, 0, 1) = \mathbf{z}_0 - (\mathbf{z}_0 - \mathbf{z}_1) = \mathbf{z}_1. \tag{27}$$

Thus, under the chosen linear latent path, the full-interval oracle IFM field equals the endpoint displacement required for recovering the target reactant latent. This completes the proof. □

A.2 Proof of Theorem 3.2

Proof. According to the 1-NFE Latent Inversion rule, the estimated reactant latent is

$$\hat{\mathbf{z}}_1 = \mathbf{z}_0 - u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x). \tag{28}$$

Similarly, the oracle reactant latent obtained by the exact full-interval transport field is

$$\mathbf{z}_1 = \mathbf{z}_0 - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x). \tag{29}$$

Subtracting the second equation from the first one, we have

$$\begin{aligned} \hat{\mathbf{z}}_1 - \mathbf{z}_1 &= [\mathbf{z}_0 - u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)] - [\mathbf{z}_0 - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)] && (30)\\ &= u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x). && (31) \end{aligned}$$

Taking the squared Euclidean norm on both sides yields

$$\begin{aligned} \|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2^2 &= \|u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)\|_2^2 && (32)\\ &= \|u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)\|_2^2. && (33) \end{aligned}$$

The last equality follows from the symmetry of the Euclidean norm. Therefore, the endpoint error induced by one-step latent inversion is exactly the full-interval IFM approximation error. Since this inference rule uses a single direct update and does not numerically integrate an ODE/SDE trajectory, no accumulated solver-discretization term appears in this decomposition. This completes the proof. □

A.3 Proof of Theorem 3.3

Proof. For each molecular attribute head $a \in \mathcal{A}$, let $\mathcal{I}_a(M)$ denote the set of valid node or edge positions for that attribute under the validity mask M. For example, atom attributes are indexed by valid node positions, while bond attributes are indexed by valid edge positions. Let $\Lambda^{(a)}_{\psi,i}(\mathbf{z}, \mathbf{z}_x)$ be the decoder logits for attribute head a at position $i \in \mathcal{I}_a(M)$, and let $y_i^{(a)}$ be the corresponding ground-truth class.

We define the logit margin at the reactant latent $\mathbf{z}_1$ as

$$\gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y}) = \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_{y_i^{(a)}} - \max_{k \neq y_i^{(a)}} \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_k. \tag{34}$$

We assume the target prediction has a positive margin,

$$\gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y}) > 0, \qquad \forall a \in \mathcal{A},\ i \in \mathcal{I}_a(M). \tag{35}$$

Thus, for any incorrect class $k \neq y_i^{(a)}$, we have

$$\Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_{y_i^{(a)}} - \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_k \geq \gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y}). \tag{36}$$

Assume that each logit coordinate is locally $L_{a,i}$-Lipschitz with respect to $\mathbf{z}$ in a neighborhood containing both $\mathbf{z}_1$ and $\hat{\mathbf{z}}_1$. That is, for every class index j,

$$\left| \Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_j - \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_j \right| \leq L_{a,i} \|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{37}$$

Then for any incorrect class $k \neq y_i^{(a)}$, the perturbed logit gap satisfies

$$\Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_{y_i^{(a)}} - \Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_k \tag{38}$$

$$= \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_{y_i^{(a)}} - \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_k \tag{39}$$

$$+ \left[\Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_{y_i^{(a)}} - \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_{y_i^{(a)}}\right] \tag{40}$$

$$- \left[\Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_k - \Lambda^{(a)}_{\psi,i}(\mathbf{z}_1, \mathbf{z}_x)_k\right] \tag{41}$$

$$\geq \gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y}) - 2L_{a,i}\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{42}$$

Therefore, if

$$\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2 < \frac{\gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})}{2L_{a,i}}, \tag{43}$$

then

$$\Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_{y_i^{(a)}} > \Lambda^{(a)}_{\psi,i}(\hat{\mathbf{z}}_1, \mathbf{z}_x)_k, \qquad \forall k \neq y_i^{(a)}. \tag{44}$$

Thus, the argmax prediction for attribute head a at position i remains unchanged when replacing $\mathbf{z}_1$ with $\hat{\mathbf{z}}_1$.

Applying the same argument to all attribute heads and all valid node or edge positions yields the sufficient condition

$$\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2 < \min_{a \in \mathcal{A}} \min_{i \in \mathcal{I}_a(M)} \frac{\gamma_i^{(a)}(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})}{2L_{a,i}}. \tag{45}$$

Under this condition, all valid discrete molecular attribute predictions are preserved, including atom types, bond types, charges, aromaticity, and other decoded attributes. This completes the proof. □

A.4 Proof of Theorem 3.4

Proof. Let the decoder negative log-likelihood be

$$\ell_\psi(\mathbf{z}, \mathbf{z}_x; \mathbf{y}) = -\log p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{z}_x). \tag{46}$$

Assume that, for the latent pairs considered in expectation, there exists an integrable local Lipschitz constant $C(\mathbf{z}_1, \hat{\mathbf{z}}_1, \mathbf{z}_x, \mathbf{y}) < \infty$ such that

$$|\ell_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x; \mathbf{y}) - \ell_\psi(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})| \leq C(\mathbf{z}_1, \hat{\mathbf{z}}_1, \mathbf{z}_x, \mathbf{y})\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{47}$$

If this local constant is uniformly bounded by C almost surely, or if the product term is integrable and upper-bounded in expectation by a finite constant C, then we obtain the one-sided upper bound

$$\ell_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x; \mathbf{y}) \leq \ell_\psi(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y}) + C\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{48}$$

Taking expectation over the data distribution and the sampled latent noise gives

$$\mathbb{E}[\ell_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x; \mathbf{y})] \leq \mathbb{E}[\ell_\psi(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})] + C\,\mathbb{E}\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2. \tag{49}$$

According to Theorem 3.2, the latent discrepancy can be written as

$$\|\hat{\mathbf{z}}_1 - \mathbf{z}_1\|_2 = \|u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)\|_2. \tag{50}$$

Substituting this identity into the previous inequality, we obtain

$$\mathbb{E}[\ell_\psi(\hat{\mathbf{z}}_1, \mathbf{z}_x; \mathbf{y})] \leq \mathbb{E}[\ell_\psi(\mathbf{z}_1, \mathbf{z}_x; \mathbf{y})] + C\,\mathbb{E}\|u_\theta(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x) - u^\star(\mathbf{z}_0, 0, 1 \mid \mathbf{z}_x)\|_2. \tag{51}$$

Thus, the inference-time decoding risk at the inverted latent is bounded by the decoder risk at the reactant latent plus a term proportional to the full-interval IFM approximation error. This result is an error decomposition under the stated local regularity assumptions; it does not by itself imply that the training objective globally minimizes the full-interval approximation error. This completes the proof. □

B Details of Model Architectures

FlashRetro consists of three neural components for continuous latent flow matching in retrosynthesis: a molecular encoder that maps graphs to latents, a velocity-field network that models latent transport, and a molecular decoder that reconstructs discrete reactant structures.

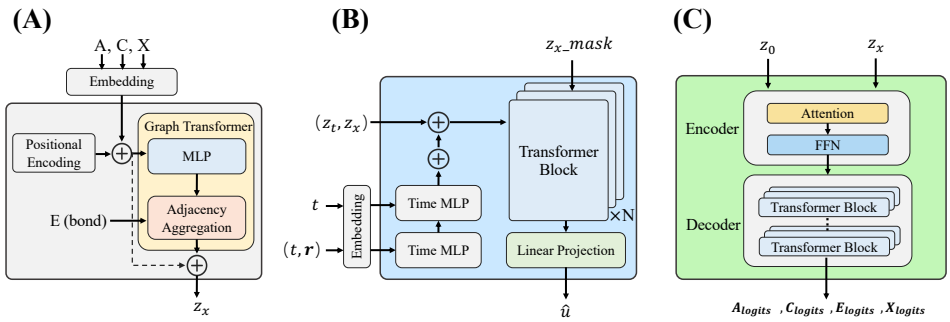


Figure 5: **Overview of the FlashRetro Architecture.** (Left) The Molecular Encoder processes discrete graphs into continuous latents. (Middle) The Velocity Field Network (Latent Transformer) models the flow between noise and reactant latents. (Right) The Molecular Decoder reconstructs the discrete reactant graph from the flowed latent.

Molecular Encoder. The molecular encoder maps discrete molecular graphs into a continuous latent space. It takes graph attributes (e.g., element types, bond types, aromaticity, and charges), along with masks, and outputs a sequence of latent vectors. Architecturally, we use a graph transformer encoder that aggregates neighborhood information through message passing and adjacency-based attention, so the resulting embeddings reflect the input molecular structure.

Velocity Field Neural Network. The velocity-field network g_θ parameterizes the averaged flow-matching model. We implement g_θ as a multi-layer latent transformer that predicts a time-dependent velocity used to transport a noise latent toward the target reactant latent. The network input is formed by combining the current latent $\mathbf{z}_t$ with the conditioning product latent $\mathbf{z}_x$, along with sinusoidal time embeddings $\tau(t)$ and $\tau(t-r)$. The output $g_\theta(\mathbf{z}_t, r, t \mid \mathbf{z}_x)$ has the same dimensionality as the latent space and defines the update direction for one-step generation.

Molecular Decoder. The Molecular Decoder is tasked with reconstructing chemically valid discrete graphs from the matched latent vector $\mathbf{z}_1$. This process operates in two stages to ensure high-fidelity generation. First, a Merge Encoder f_m fuses the generative latent $\mathbf{z}_1$ with the product condition $\mathbf{z}_x$ using cross-attention to align structural contexts. Second, a Transformer Decoder f_d predicts the specific logits $\mathbf{\Lambda}$ for discrete attributes, including bonds ($\mathbf{\Lambda}_E$), aromaticity ($\mathbf{\Lambda}_A$), charges ($\mathbf{\Lambda}_C$), and element types ($\mathbf{\Lambda}_X$). The final graph is derived by taking the argmax of these logits, utilizing symmetry enforcement for bond predictions.

C Details of Experiments

C.1 Details of Dataset

We evaluate FlashRetro on USPTO-50K, a standard single-step retrosynthesis benchmark derived from the United States Patent and Trademark Office dataset [Lowe, 2017]. USPTO-50K contains roughly 50,000 reactions grouped into 10 reaction classes. Following prior work [Yadav et al., 2025, Igashov et al., 2024], we preprocess the data as follows: (1) reactions with multiple products are split into single-product instances while keeping all original reactants; and (2) trivial products (e.g., inorganic ions and solvents) are removed. We use the standard 80%/10%/10% train/validation/test split. Class-wise counts are reported in Table 5.

C.2 Topological Characteristics

Chemical graphs in USPTO-50K exhibit diverse structural properties. For example, 13.61% of reaction centers involve ring bonds, and 40.09% have exactly one node on a ring. Non-planar graphs are rare (0.448%), but we retain them in all experiments. To better capture such cases, FlashRetro uses both the original graph and a dual-graph representation.

Table 5: Distribution of reaction classes in the USPTO-50k dataset.

Class Name	# Examples
Heteroatom alkylation and arylation	15,204
Acylation and related processes	11,972
Deprotections	8,405
C-C bond formation	5,667
Reductions	4,642
Functional group interconversion (FGI)	1,858
Heterocycle formation	909
Oxidations	822
Protections	672
Functional group addition (FGA)	231
Total	**50,402**

D Extended Analysis and Discussion

D.1 Performance Across Diverse Reaction Categories

Table 6 reports the per-class performance of FlashRetro across different reaction categories. FlashRetro achieves strong Top-1 accuracy on frequent and localized transformations, including *Acylation* (83.54%), *Protections* (85.45%), and *Functional group addition* (90.48%). These results show that the model performs effectively on reaction classes characterized by clear functional-group patterns. We also observe that some categories, including *Heterocycle formation*, exhibit relatively stable performance as the candidate set increases, indicating that the main predictions are concentrated among the top-ranked candidates.

Table 6: Detailed Top-k performance breakdown by reaction class on the USPTO-50k test set. Reaction fractions are based on the standard dataset distribution.

Reaction Class	Fraction (%)	Top-k accuracy			
		$k=1$	$k=3$	$k=5$	$k=10$
Heteroatom alkylation/arylation	30.3	66.58	82.14	84.79	85.98
Acylation and related processes	23.8	83.54	90.97	91.15	91.77
C-C bond formation	11.3	41.53	50.12	52.27	54.42
Heterocycle formation	1.8	42.59	46.30	46.30	46.30
Protections	1.3	85.45	87.27	87.27	87.27
Deprotections	16.5	77.66	80.67	81.74	82.09
Reductions	9.2	77.01	81.16	81.44	82.27
Oxidations	1.6	54.69	64.06	65.62	65.62
Functional group interconversion	3.7	66.42	75.37	76.87	79.10
Functional group addition	0.5	90.48	95.24	95.24	95.24

D.2 Wall-Clock Training Time

We compare FlashRetro with two recent flow-based retrosynthesis baselines, RetroSynFlow and RetroBridge, using wall-clock time and cumulative training compute at matched Top-1 accuracy milestones. Rather than comparing convergence by epoch, which can be misleading for models with different sizes and training costs, we report the time required to reach 30% Top-1 accuracy and approximately 50% Top-1 accuracy, together with the cumulative PFLOPs required to reach the latter milestone. As shown in Table 7, FlashRetro reaches both accuracy milestones substantially faster than the baselines. In particular, FlashRetro reaches 30% Top-1 accuracy in 1.8 hours and approximately 50% Top-1 accuracy in 3.9 hours, compared with 2.2/48.0 hours for RetroSynFlow and 12.2/376.0 hours for RetroBridge. FlashRetro also requires lower cumulative compute to reach approximately 50% accuracy, using 41.59 PFLOPs compared with 60.5 PFLOPs for RetroSynFlow and 58.9 PFLOPs for RetroBridge. These results indicate that FlashRetro improves training efficiency in both wall-clock time and total compute.

Table 7: Training efficiency comparison on USPTO-50K. Wall-clock time and cumulative PFLOPs at matched Top-1 accuracy milestones.

Model	Time to 30% Acc.	Time to ∼50% Acc.	PFLOPs to ∼50% Acc.
RetroSynFlow	2.2 h	48.0 h	60.5
RetroBridge	12.2 h	376.0 h	58.9
FlashRetro	1.8 h	3.9 h	41.59

D.3 Top-k Round-Trip Coverage and Accuracy

We further report Top-k round-trip coverage and accuracy on USPTO-50K to evaluate chemical validity and forward consistency of the generated reactants. Following prior work, we measure validity (Val@1) and round-trip consistency (RTC@1 / RTA@1). As shown in Table 8, FlashRetro achieves competitive validity and round-trip.

Table 8: Top-k round-trip coverage and accuracy on USPTO-50K. Comparisons with baselines regarding chemical validity and consistency.

Method	Top-1 (%)	Val@1 (%)	RTC@1 (%)	RTA@1 (%)
RetroSynFlow	60.0	99.2	88.2	88.2
RetroProdFlow	50.2	99.8	**90.7**	**90.7**
RetroBridge	50.8	99.2	85.1	85.1
FlashRetro	**69.3**	**99.9**	<u>88.5</u>	<u>88.5</u>

D.4 Decoder Robustness and Transport Quality

To further examine whether the strong 1-step performance is caused by an overly robust decoder, we inject Gaussian noise into two types of latent representations: the ground-truth reactant latent z_1 and the predicted reactant latent produced by the 1-step Latent Inversion rule. By perturbing these latents separately, we decouple decoder tolerance from the accuracy of the learned transport endpoint. As shown in Tables 9 and 10, FlashRetro is stable under small perturbations, but performance drops sharply when the noise becomes large. This suggests that the decoder has bounded local tolerance, while accurate 1-step transport is still necessary to keep the predicted latent within a decodable neighborhood.

Table 9: Robustness when perturbing the ground-truth reactant latent z_1.

Noise Std. (σ)	Top-1	Val@1	Top-3	Val@3	Top-5	Val@5	Top-10	Val@10
0.00	69.30	99.98	78.40	96.82	79.80	90.82	81.10	70.20
0.05	69.48	100.0	78.15	96.61	79.86	90.36	81.18	69.71
0.10	69.44	99.96	78.17	97.19	79.60	91.82	81.00	72.40
0.20	69.56	99.98	78.09	98.88	79.62	96.63	80.98	85.93
0.25	69.28	100.0	78.25	99.70	79.78	98.86	81.14	94.16
0.30	69.02	100.0	77.57	99.95	78.97	99.84	80.16	98.60
0.50	15.12	100.0	17.50	100.0	18.08	100.0	19.08	99.96

D.5 Sensitivity to Flow Loss Weight

We further study the sensitivity of FlashRetro to the flow loss weight λ_{flow}. As shown in Table 11, the default value $\lambda_{\text{flow}} = 0.01$ achieves the best performance across all Top-k metrics. Smaller weights provide insufficient supervision for latent transport, while larger weights may over-regularize the latent space and hurt reconstruction quality.

Table 10: Robustness when perturbing the predicted latent produced by the 1-step flow.

Noise Std. (σ)	**Top-1**	**Val@1**	**Top-3**	**Val@3**	**Top-5**	**Val@5**	**Top-10**	**Val@10**
0.00	69.30	99.98	78.40	96.82	79.80	90.82	81.10	70.20
0.02	69.54	99.96	78.73	97.66	80.06	92.73	81.36	74.46
0.05	69.34	100.0	79.07	99.41	80.52	97.92	82.22	91.21
0.10	53.56	100.0	61.37	100.0	62.90	100.0	64.22	99.40
0.15	3.12	100.0	3.96	100.0	4.23	99.90	4.61	99.95

Table 11: Sensitivity analysis of the flow loss weight λ_{flow}.

Weight (λ_{flow})	**Top-1 (%)**	**Top-3 (%)**	**Top-5 (%)**	**Top-10 (%)**
0.001	32.10	34.50	35.60	36.54
0.01	**69.30**	**78.40**	**79.80**	**81.10**
0.05	54.51	55.24	58.91	61.01
0.10	34.22	38.14	41.59	44.32

D.6 Case Study via Visualization

In this section, we provide a qualitative comparison between FlashRetro and state-of-the-art baselines, including *RetroSynFlow*, *GDiffRetro*, *RetroBridge*, *Graph2SMILES*, and *RetroFormer*. As illustrated in Figures 6 – 7, FlashRetro demonstrates superior precision in strategic disconnection, consistently identifying the ground-truth reactants in the Top-1 prediction. In contrast, diffusion-based baselines like *GDiffRetro* and sequence-based models like *Graph2SMILES* frequently fail to recover the correct disconnection site (indicated by red boxes), often generating chemically implausible or over-fragmented intermediates. Furthermore, FlashRetro exhibits remarkable structural robustness across its Top-k candidates; even when alternative precursor sets are proposed (Predict #2–#5), they predominantly remain chemically valid and structurally coherent. This contrasts with *RetroSynFlow* and *RetroFormer*, where lower-ranked predictions often degrade into invalid scaffolds, highlighting our model's robust grasp of the chemical reaction manifold.

D.7 Code Availability and Reproducibility

The code is available at :`https://github.com/jk-sounds/FlashRetro`

E Limitations and Future Work

Despite its strong accuracy and efficient inference, FlashRetro still has room for improvement. Although the template-free design provides flexibility in reactant generation, many chemists may favor template-based methods because they are grounded in familiar reaction patterns, reagents, and reactant classes. A promising future direction is to introduce template guidance or reaction-type priors into the latent inversion process, which could improve interpretability while preserving generative flexibility. Moreover, FlashRetro currently emphasizes fast and high-confidence prediction, and generating more diverse candidate reactant sets remains an important challenge. Looking ahead, we aim to combine FlashRetro with richer chemical knowledge, verification modules, and planning feedback to build more robust and interpretable retrosynthesis systems.



Figure 6: Visual comparison of retrosynthetic predictions (Example 1). **FlashRetro** (top row) correctly identifies the ground truth in the first prediction (green box) and provides valid alternatives. Baselines such as *RetroSynFlow* and *Graph2SMILES* show prediction failures (red boxes) in top ranks.

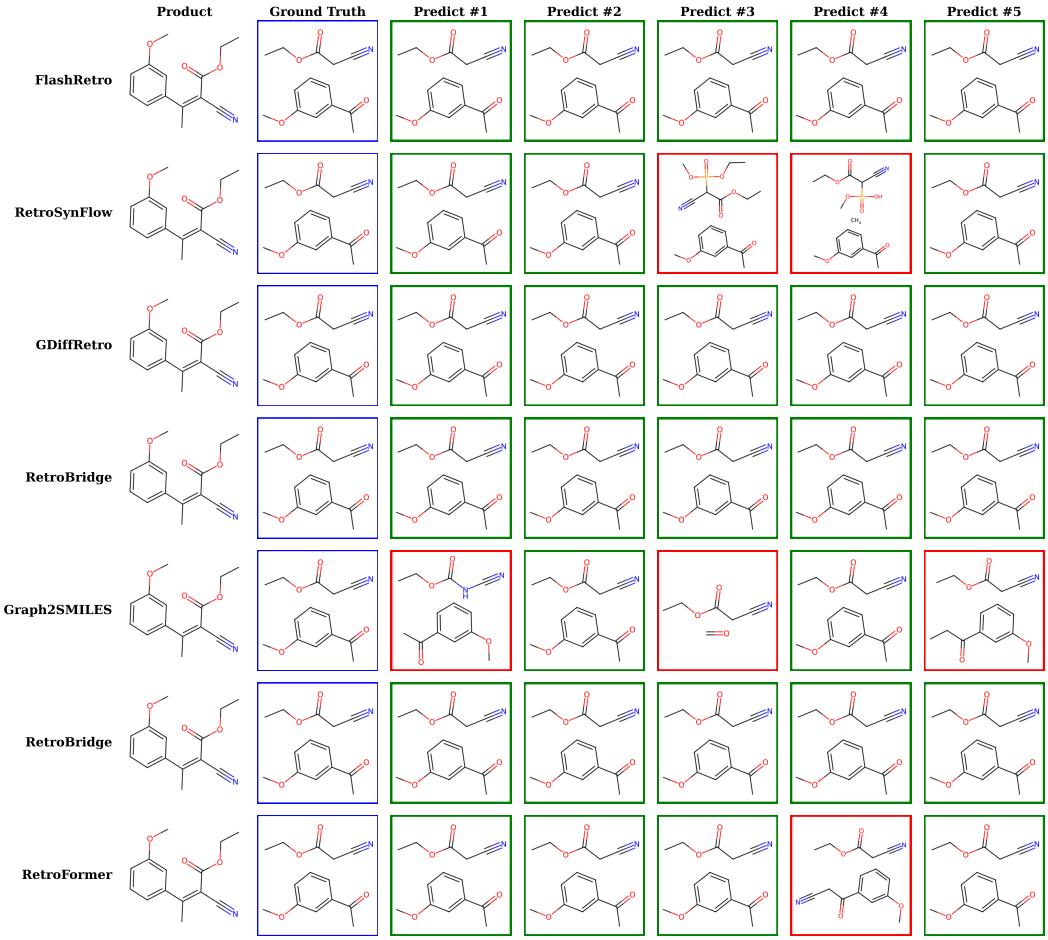


Figure 7: Visual comparison of retrosynthetic predictions (Example 2). **FlashRetro** demonstrates high structural robustness with consistently valid Top-5 predictions, whereas baselines like *Graph2SMILES* generate chemically invalid structures.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction state that FlashRetro targets efficient retrosynthesis prediction through Integral Flow Matching and 1-NFE Latent Inversion. These claims are supported by the method in Section 3 and by the accuracy, latency, NFE, ablation, and robustness experiments in Section 4.2 and Appendix D.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: Appendix E discusses the main limitations of FlashRetro, including weaker high-k diversity of deterministic Latent Inversion and the dependence on latent encoder–decoder reconstruction fidelity. The paper also reports reaction-class breakdowns and higher-k results to clarify where the method is strongest and where it is more limited.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

   Justification: The paper states the assumptions for the integral transport identity, latent inversion error, molecular attribute stability, and inference-time risk bound in Section 3. Complete proofs are provided in Appendix A, including the linear latent path assumption, decoder margin/Lipschitz assumptions, and the local regularity assumptions used in the risk bound.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: Section 4.1 specifies the dataset, split protocol, baselines, evaluation metrics, optimizer, model dimensions, training schedule, and inference settings. Appendix B provides architecture details, Appendix D provides additional evaluation details, and Appendix D.7 provides an anonymized implementation link for reproducing the main results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The paper uses the public USPTO-50K benchmark and provides an anonymized code repository in Appendix D.7. The repository is intended to include the implementation, preprocessing, training, and evaluation scripts required to reproduce the FlashRetro experiments.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: Section 4.1 reports the USPTO-50K split protocol, baselines, top-k exact-match evaluation, optimizer, learning rate, batch size, training iterations, scheduler, gradient clipping, model depth, hidden dimension, attention heads, time sampling, finite-interval sampling probability, and loss weights. Appendix B further describes the encoder, transport network, and decoder architectures.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [N/A].

   Justification: All main FlashRetro results in the paper are averages under three random seeds to eliminate randomness.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: Section 4.2 reports inference latency measured on a single NVIDIA A800 GPU with batch size 1. Appendix D.2 further reports training wall-clock time and cumulative PFLOPs required to reach matched Top-1 accuracy milestones, enabling comparison of training efficiency against iterative baselines.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The work uses public chemical reaction benchmarks and does not involve private data, human subjects, or deceptive data collection. The submission is anonymized, and the reported experiments and code release are designed to support reproducibility and transparent evaluation.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: The Impact Statement discusses positive impacts such as more efficient retrosynthesis prediction for drug discovery, sustainable molecular design, high-throughput screening, and multi-step synthesis planning. It also notes that the method should be used as a computational aid rather than a replacement for expert chemical validation, since incorrect predictions could mislead downstream synthesis planning.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: This work does not release a high-risk generative model such as a pretrained language model, image generator, surveillance system, or scraped dataset. The released implementation targets single-step retrosynthesis prediction on a public benchmark and does not contain sensitive personal data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: The paper cites the USPTO-50K dataset and all baseline methods used for comparison, and the implementation is based on publicly available scientific benchmarks and prior work. Dataset access, baseline attribution, and applicable license or usage terms are described in the reproducibility material and repository documentation.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: The paper introduces the FlashRetro implementation as a new research asset and provides it through an anonymized repository. The repository documentation describes how to preprocess data, train the model, run inference, and reproduce the main experimental results.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

Justification: This work does not involve crowdsourcing, user studies, human-subject experiments, or participant compensation.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A].

Justification: This work does not involve human subjects, crowdsourcing, or participant data, and therefore does not require IRB approval or equivalent human-subjects review.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A].

Justification: LLMs were used only for manuscript polishing and did not contribute to the core methodology, experiments, scientific claims, or originality of this work.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.