A Related Work

Autoregressive Molecule Generation. Autoregressive models provide control over the generative process by enabling direct conditioning on prior information, allowing for a more precise and targeted generation of output. Autoregressive generation has shown success in 2D molecule tasks using SMILE-based methods, as seen in MolMIM (Reidenbach et al., 2023), as well as graph-based atom-wise and subgraph-level techniques, as shown in GraphAF (Shi et al., 2020) and HierVAE (Jin et al., 2020). Similarly, 3DLinker (Huang et al., 2022) and SQUID (Adams and Coley, 2023) showcase the usefulness of 3D autoregressive molecule generation and their ability to leverage conditional information in both atom-wise and subgraph-level settings for 3D linkage and shape-conditioned generative tasks respectively. We note that, unlike prior methods (Adams and Coley, 2023), CoarsenConf does not require a predefined fragment vocabulary. HERN (Jin et al., 2022b) further demonstrates the power of hierarchical equivariant autoregressive methods in the task of computational 3D antibody design. Similarly, Pocket2Mol (Peng et al., 2022) uses autoregressive sampling for structure-based drug design.

Protein Docking and Structure-based Drug Design. Protein docking is a key downstream use case for generating optimal 3D molecule structures. Recent research has prominently explored two distinct directions within this field. The first is blind docking, where the goal is to locate the pocket and generate the optimal ligand to bind (Corso et al., 2022). The second is structure-based drug design (SBDD), where optimal 3D ligands are generated by conditioning on a specific protein pocket. Specifically, the SBDD task focuses on the ability to generate ligands that achieve a low AutoDock Vina score for the CrossDocked2020 (Francoeur et al., 2020) dataset. AutoDock Vina (Eberhardt et al., 2021) is a widely used molecular docking software that predicts the binding affinity of ligands (drug-like molecules) to target proteins. Autodock Vina takes in the 3D structures of the ligand, target protein, and binding pocket and considers various factors such as van der Waals interactions, electrostatic interactions, and hydrogen bonding between the ligand and target protein to predict the binding affinity. We demonstrate how SBDD can be adapted to construct comprehensive MCG benchmarks. In this framework, we evaluate the generative abilities of MCG models by measuring the binding affinities of generated comforters and comparing them to the provided ground truth ligand conformers for a wide array of protein-ligand complexes.

SE(3)-Equivariance. Let $\mathcal{X}$ and $\mathcal{Y}$ be the input and output vector spaces, respectively, which possess a set of transformations G: $G \times \mathcal{X} \to \mathcal{X}$ and $G \times \mathcal{Y} \to \mathcal{Y}$. The function $\phi : \mathcal{X} \to \mathcal{Y}$ is called equivariant with respect to G if, when we apply any transformation to the input, the output also changes via the same transformation or under a certain predictable behavior, *i.e.*,

Definition 1 *The function $\phi : \mathcal{X} \mapsto \mathcal{Y}$ is G-equivariant if it commutes with any transformation in G,*

$$\phi(\rho_{\mathcal{X}}(g)x) = \rho_{\mathcal{Y}}(g)\phi(x), \forall g \in G, \tag{4}$$

where $\rho_{\mathcal{X}}$ and $\rho_{\mathcal{Y}}$ are the group representations in the input and output space, respectively. Specifically, ϕ is called invariant if $\rho_{\mathcal{Y}}$ is the identity.

By enforcing SE(3)-equivariance in our probabilistic model, $p(\boldsymbol{X}|\mathcal{R})$ remains unchanged for any rototranslation of the approximate conformer $\mathcal{R}$. CoarsenConf's architecture is inspired by recent equivariant graph neural network architectures, such as EGNN (Satorras et al., 2021) and PaiNN (Schütt et al., 2021), as well as Vector Neuron multi-layer perceptron (VN-MLP) (Deng et al., 2021).

B Loss function

As described in §3, CoarsenConf optimizes the following loss function:

$$\text{MSE}(\mathcal{A}(X, X_{true})) + \beta_1 D_{KL}(q_\phi(z|X, \mathcal{R}) \parallel p_\psi(z|\mathcal{R})) + \beta_2 \frac{1}{|\mathcal{E}^*|} \sum_{(i,j)\in\mathcal{E}^*} ||r_{ij} - r_{ij}^{true}||^2, \tag{5}$$

where $\mathcal{A}$ is the Kabsch alignment function (Kabsch, 1993), $\mathcal{E}^*$ are all the 1 and 2-hop edges in the molecular graph, with r_{ij} corresponding to the distance between atoms i and j. We note that both β_1 and β_2 play a crucial role in the optimization. β_1 has to be set low enough ($1e{-}3$) to allow the optimization to focus on the MSE when the differences between the model-based X and the ground truth are very close, due to the RDKit distortion parameterization.

For the QM9 experiments, β_1 is annealed starting from $1e{-}6$ to $1e{-}1$, increasing by a factor of 10 each epoch. β_2 controls the distance auxiliary loss and also had to be similarly annealed. We found that when $\beta_2 = 0$, CoarsenConf still learned to improve upon the aligned MSE loss by 50%, as compared to RDKit. Our error analysis showed that the resulting molecules either had extremely low distance error with high MSE, or vice-versa. Therefore, when the learning objective is unconstrained, our model learns to violate distance constraints by placing atoms in low-error but unphysical positions.

For QM9, by slowly annealing the distance loss, we allow our model to reach a metaphysical unstable transition state where distances are violated, but the aligned coordinate error is better. We then force the model to respect distance constraints. In the case of DRUGS, we found that this transition state was too difficult for the model to escape from, and we report the results using $\beta_2 = 0.5$ in Tab. 1. In Appendix §J, we further explore this idea and experiment with different annealing schedules for DRUGS. We note that as CoarsenConf learns the torsion angles in an unsupervised manner because of the chosen CG strategy, we leave explicit angle optimization to future work.

C COARSE-GRAINING

We elaborate on the coarse-graining procedure introduced in §3. Following Wang et al. (2022), we represent fine-grained (FG) molecular conformers as $x = \{x_i\}_{i=1}^{n} \in \mathbb{R}^{n\times 3}$. Similarly, the coarse-grained (CG) conformers are represented by $X = \{X_I\}_{I=1}^{N} \in \mathbb{R}^{N\times 3}$ where $N < n$. Let $[n]$ and $[N]$ denote the set $\{1, 2, ..., n\}$ and $\{1, 2, ..., N\}$ respectively. The CG operation can be defined as an assignment $m : [n] \to [N]$, which maps each FG atom i in $[n]$ to CG bead $I \in [N]$, i.e., bead I is composed of the set of atoms $C_I = (k \in n \mid m(k) = I)$. X_I is initialized at the center of mass = $\frac{1}{|C_I|}\sum_{j\in C_I} x_j$.

We note that CoarsenConf coarsens input molecules by first severing all torsion angles τ_{abcd}, with k torsion angles resulting in $k + 1$ connected components or CG beads. This allows us, on average, to represent QM9 molecules with three beads and large drug molecules ($n > 100$) with 29 beads. We opted for a torsion angle-based strategy as it allows for unsupervised control over torsion angles, as well as the ability to rotate each subgraph independently. The CG strategy can be altered for various applications going forward.

D ENCODER EQUATIONS

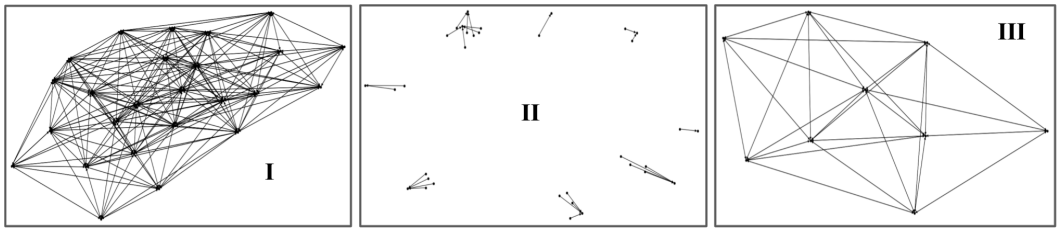


Figure 6: **Encoder module message passing structure. (I)** Fine-grained graph with auxiliary 4Å distance cut off. **(II)** Pooling graph with nodes for each atom and coarse-grained bead. Each group of nodes represents the formation of a CG bead. There is a single directional edge from each atom to its corresponding bead. **(III)** Coarse-grained graph with auxiliary 4Å distance cut off, using the learned representation from the pooling graph. CoarsenConf reduces the input from (I) to (III), drastically reducing the complexity of the problem.

D.1 FINE-GRAIN MODULE

We describe the encoder, shown in Fig. 2(I). The model operates over SE(3)-invariant atom features $h \in R^{n \times D}$, and SE(3)-equivariant atomistic coordinates $x \in R^{n \times 3}$. A single encoder layer is composed of three modules: fine-grained, pooling, and coarse-grained. Full equations for each module can be found in Appendix §D.1, §D.2, §D.3, respectively.

The fine-grained module is a graph-matching message-passing architecture. It differs from Stärk et al. (2022) by not having internal closed-form distance regularization and exclusively using unidirectional attention. It aims to effectively match the approximate conformer and ground truth by updating attention from the former to the latter.

The FG module is responsible for processing the FG atom coordinates and invariant features. More formally, the FG is defined as follows:

$$
\begin{aligned}
\boldsymbol{m}_{j \to i} &= \phi^e(\boldsymbol{h}_i^{(t)}, \boldsymbol{h}_j^{(t)}, \|\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t)}\|^2, \boldsymbol{f}_{j \to i}), \forall (I, J) \in \mathcal{E} \cup \mathcal{E}', \\
\boldsymbol{u}_{j' \to i} &= a_{j' \to i} \boldsymbol{W} \boldsymbol{h}_{j'}^{(t)}, \forall i \in \mathcal{V}, j' \in \mathcal{V}', \\
\boldsymbol{m}_i &= \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{j \to i}, \forall i \in \mathcal{V} \cup \mathcal{V}', \\
\boldsymbol{u}_i &= \sum_{j' \in \mathcal{V}'} \boldsymbol{u}_{j' \to i}, \forall i \in \mathcal{V}, \quad \text{and} \quad \boldsymbol{u}'_i = 0, \\
\boldsymbol{x}_i^{(t+1)} &= \eta_x \cdot \boldsymbol{x}_i^{(0)} + (1 - \eta_x) \cdot \boldsymbol{x}_i^{(t)} + \sum_{j \in \mathcal{N}(i)} (\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t)}) \phi^x(\boldsymbol{m}_{j \to i}), \\
\boldsymbol{h}_i^{(t+1)} &= (1 - \eta_h) \cdot \boldsymbol{h}_i^{(t)} + \eta_h \cdot \phi^h(\boldsymbol{h}_i^{(t)}, \boldsymbol{m}_i, \boldsymbol{u}_i, \boldsymbol{f}_i), \forall i \in \mathcal{V} \cup \mathcal{V}',
\end{aligned}
\tag{6}
$$

where f represents the original invariant node features $h^{t=0}$, $a_{j \to i}$ are SE(3)-invariant attention coefficients derived from h embeddings, $\mathcal{N}(i)$ are the graph neighbors of node i, and W is a parameter matrix. $(\mathcal{V}, \mathcal{E})$ and $(\mathcal{V}', \mathcal{E}')$ refer to the low-energy and RDKit approximation molecular graphs, respectively. The various ϕ functions are modeled using shallow MLPs, with ϕ^x outputting a scalar and ϕ^e and ϕ^h returning a D-dimensional vector. η_x and η_h are weighted update parameters for the FG coordinates x and invariant features h respectively. We note that attention flows in a single direction from the RDKit approximation to the ground truth to prevent leakage in the parameterization of the learned prior distribution.

D.2 POOLING MODULE

The pooling module takes in the updated representations (h and x) of both the ground truth molecule and the RDKit reference from the FG module. The pooling module is similar to the FG module, except it no longer uses attention and operates over a pooling graph. Given a molecule with n atoms

and N CG beads, the pooling graph consists of $n + N$ nodes. There is a single directional edge from all atoms to their respective beads. This allows message passing to propagate information through the predefined coarsening strategy.

The pooling module is responsible for learning the coordinates and invariant features of each coarse-grained bead by pooling FG information in a graph-matching framework. More formally, the pooling module is defined as follows:

$$\begin{aligned}
\boldsymbol{m}_{j\to I} &= \phi^e(\boldsymbol{H}_I^{(t)}, \boldsymbol{h}_j^{(t)}, \|\boldsymbol{X}_I^{(t)} - \boldsymbol{x}_j^{(t)}\|^2, \boldsymbol{f}_{j\to I}), \forall (I,J) \in \mathcal{E} \cup \mathcal{E}', \\
\boldsymbol{m}_I &= \frac{1}{|\mathcal{N}(I)|} \sum_{j \in \mathcal{N}(I)} \boldsymbol{m}_{j\to I}, \forall I \in \mathcal{V} \cup \mathcal{V}', \\
\boldsymbol{X}_I^{(t+1)} &= \eta_X \cdot \boldsymbol{X}_I^{(0)} + (1-\eta_X) \cdot \boldsymbol{X}_I^{(t)} + \sum_{j \in \mathcal{N}(I)} (\boldsymbol{X}_I^{(t)} - \boldsymbol{x}_j^{(t)}) \phi^x(\boldsymbol{m}_{j\to I}), \\
\boldsymbol{H}_I^{(t+1)} &= (1-\eta_H) \cdot \boldsymbol{H}_I^{(t)} + \eta_H \cdot \phi^h(\boldsymbol{H}_I^{(t)}, \boldsymbol{m}_I, \boldsymbol{f}_I), \forall I \in \mathcal{V} \cup \mathcal{V}',
\end{aligned} \tag{7}$$

where capital letters refer to the CG representation of the pooling graph. The pooling module mimics the FG module without attention on a pooling graph, as seen in Fig. 6(II). The pooling graph contains a single node for each atom and CG bead, with a single edge from each FG atom to its corresponding bead. It is used to learn the appropriate representations of the CG information. As the pooling graph only contains edges from fine-to-coarse nodes, the fine-grain coordinates and features remain unchanged. The pooling graph at layer t uses the invariant feature H from the CG module of layer $t-1$ to propagate information forward through the neural network. The main function of the pooling module is to act as a buffer between the FG and CG spaces. As a result, we found integrating the updated CG representation useful for building a better transition from FG to CG space.

D.3 Coarse-grain Module

The coarse-grained module uses the updated CG representations ($H \in R^{N\times D}$ and $X \in R^{N\times 3}$) from the pooling module to learn equivariant CG features (Z and $\tilde{Z} \in R^{N\times F\times 3}$) for the ground truth molecule and the RDKit reference. F is fixed as a hyperparameter for latent space size. N is allowed to be variable-length to handle molecules resulting from any coarsening procedure. The CG features are learned using a graph-matching point convolution (Thomas et al., 2018) with similar unidirectional attention as the FG module. Prior to the main message-passing operations, the input features undergo equivariant mixing (Huang et al., 2022) to further distill geometric information into the learned CG representation.

The CG module is responsible for taking the pooled CG representation from the pooling module and learning a node-level equivariant latent representation. We note that we use simple scalar and vector operations to mix equivariant and invariant features without relying on computationally expensive higher-order tensor products. In the first step, invariant CG features H and equivariant features $\boldsymbol{v} \in \mathbb{R}^{F\times 3}$ are transformed and mixed to construct new expressive intermediate features $H', H'', \boldsymbol{v}'$ by,

$$H_I' = \phi_1(h_I^{(t)}, \|\text{VN-MLP}_1(\boldsymbol{v_I^{(t)}})\|) \in \mathbb{R}^D, \tag{8a}$$

$$H_I'' = \phi_2(h_I^{(t)}, \|\text{VN-MLP}_2(\boldsymbol{v_I^{(t)}})\|) \in \mathbb{R}^F, \tag{8b}$$

$$\boldsymbol{v}_I' = \text{diag}\{\phi_3(H_I^{(t)})\} \cdot \text{VN-MLP}_3(\boldsymbol{v_I^{(t)}}) \in \mathbb{R}^{F\times 3}. \tag{8c}$$

Next, a point convolution (Thomas et al., 2018; Schütt et al., 2021; Huang et al., 2022) is applied to linearly transform the mixed features $H', H'', \boldsymbol{v}'$ into messages:

$$\boldsymbol{m}^{\boldsymbol{H}}_{I\leftarrow J} = \text{Ker}_1(\|\boldsymbol{r}_{I,J}\|) \odot H'_J, \tag{9a}$$

$$\boldsymbol{m}^{\boldsymbol{v}}_{I\leftarrow J} = \text{diag}\left\{\text{Ker}_2(\|\boldsymbol{r}_{I,J}\|)\right\} \cdot \boldsymbol{v}'_J + \left(\text{Ker}_3(\|\boldsymbol{r}_{I,J}\|) \odot H''_J\right) \cdot \boldsymbol{r}^{\top}_{I,J}, \tag{9b}$$

$$\boldsymbol{u}_{J'\to I} = a_{J'\to I}\boldsymbol{W}\boldsymbol{H}^{(t)}_{J'}, \forall I \in \mathcal{V}, J' \in \mathcal{V}', \tag{9c}$$

$$\boldsymbol{u}_I = \sum_{J'\in\mathcal{V}'} \boldsymbol{u}_{J'\to I}, \forall I \in \mathcal{V}, \quad \text{and} \quad \boldsymbol{u}'_I = 0, \tag{9d}$$

$$\boldsymbol{H}^{t+1}_I = (1-\eta_H)\cdot H^{\ell}_I + \eta_H \cdot \text{MLP}(H^{\ell}_I, \sum_{J\in N(I)} m^{H}_{I\leftarrow J}, u_I), \forall I \in \mathcal{V}\cup\mathcal{V}', \tag{9e}$$

$$\boldsymbol{v}^{t+1}_I = (1-\eta_v)\cdot v^{\ell}_I + \eta_v \cdot \text{VN-MLP}_4(\boldsymbol{v}^{\ell}_I, \sum_{J\in N(I)} \boldsymbol{m}^{\boldsymbol{v}}_{I\leftarrow J}), \forall I \in \mathcal{V}\cup\mathcal{V}', \tag{9f}$$

where each Ker refers to a learned RBF kernel, r_{IJ} is the difference between X_I and X_J, and $a_{J\to I}$ are SE(3)-invariant attention coefficients derived from the learned invariant features H. η_H and η_v control the mixing of the learned invariant and equivariant representations.

We note that for $t > 0$, the H_I from the CG module are used in the next layer's pooling module, creating a cyclic dependency to learn an information-rich CG representation. This is shown by the dashed lines in Fig. 2(I). The cyclic flow of information grounds the learned CG representation to the innate FG structure. All equivariant CG features v are initialized as zero and are slowly built up through each message passing layer. As point convolutions and VN operations are strictly SO(3)-equivariant, we subtract the molecule's centroid from the atomic coordinates prior to encoding, making it effectively SE(3)-equivariant.

The modules in each encoder layer communicate with the respective module of the previous layer. This hierarchical message-passing scheme results in an informative and geometrically grounded final CG latent representation. We note that the pooling module of layer ℓ uses the updated invariant features H from the CG module of layer $\ell - 1$, as shown by the dashed lines in Fig. 2(I).

E Decoder Architecture

We sample from the learned posterior (training) and learned prior (inference) to get $Z = \mu + \epsilon\sigma$, where ϵ is noise sampled from a standard Gaussian distribution as the input to the decoder. We note the role of the decoder is two-fold. The first is to convert the latent coarsened representation back into FG space through a process we call channel selection. The second is to refine the fine-grain representation autoregressively to generate the final low-energy coordinates.

Channel Selection. To explicitly handle all choices of coarse-graining techniques, our model performs variable-length backmapping. This aspect is crucial because every molecule can be coarsened into a different number of beads, and there is no explicit limit to the number of atoms a single bead can represent. Unlike CGVAE (Wang et al., 2022), which requires training a separate model for each choice in granularity N, CoarsenConf is capable of reconstructing FG coordinates from any N (illustrated in Fig. 2(III)).

CGVAE defines the process of channel selection as selecting the top k latent channels, where k is the number of atoms in a CG bead of interest. Instead of discarding all learned information in the remaining $F-k$ channels in the latent representation, we use a novel aggregated attention mechanism. This mechanism learns the optimal mixing of channels to reconstruct the FG coordinates and is illustrated in Fig. 3. The attention operation allows us to actively query our latent representation for the number of atoms we need, and draw upon similarities to the learned RDKit approximation that has been distilled into the latent space through the encoding process. Channel selection translates the CG latent tensor $Z \in R^{N\times F\times 3}$ into FG coordinates $x_{cs} \in R^{n\times 3}$.

Coordinate Refinement. Once channel selection is complete, we have effectively translated the variable-length CG representation back into the desired FG form. From here, x_{cs} is grouped into its corresponding CG beads but left in FG coordinates to do a bead-wise autoregressive generation of final low-energy coordinates (Fig. 2(IV)). As there is no intrinsic ordering of subgraphs, we use

a breadth-first search that prioritizes larger subgraphs with large out-degrees. In other words, we generate a linear order that focuses on the largest, most connected subgraphs and works outward. We believe that by focusing on the most central component first, which occupies the most 3D volume, we can reduce the propagation of error that is typically observed in autoregressive approaches. We stress that by coarse-graining by torsion angle connectivity, our model learns the optimal torsion angles in an unsupervised manner, as the conditional input to the decoder is not aligned. CoarsenConf ensures each next generated subgraph is rotated properly to achieve a low coordinate and distance error.

Learning the Optimal Distortion. The decoder architecture is similar to the EGNN-based FG layer in the encoder. However, it differs in two important ways. First, we mix the conditional coordinates with the invariant atom features using a similar procedure as in the CG layer instead of typical graph matching. Second, we learn to predict the difference between the RDKit reference and ground truth conformations. This provides an upper error bound and enables us to leverage easy-to-obtain approximations more effectively.

More formally, a single decoder layer is defined as follows:

$$\boldsymbol{\mu}^{(t)} = \frac{1}{|\mathcal{V}_{prev}|} \sum_{k \in \mathcal{V}_{prev}} x_k, \tag{10a}$$

$$\tilde{\boldsymbol{h}}_i = \phi^m(\boldsymbol{h}_i^{(t)}, \boldsymbol{x}_i^{(t)}, \boldsymbol{\mu}^{(t)}, \|\boldsymbol{x}_i^{(t)} - \boldsymbol{\mu}^{(t)}\|^2), \forall i \in \mathcal{V}_{cur}, \tag{10b}$$

$$\boldsymbol{m}_{j \to i} = \phi^e(\tilde{\boldsymbol{h}}_i^{(t)}, \tilde{\boldsymbol{h}}_j^{(t)}, \|\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t)}\|^2, \|\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_{ref,j}^{(t)}\|^2, \|\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_{ref,i}^{(t)}\|^2), \forall (i,j) \in \mathcal{E}_{cur}, \tag{10c}$$

$$\boldsymbol{m}_i = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{j \to i}, \forall i \in \mathcal{V}_{cur}, \tag{10d}$$

$$\boldsymbol{u}_{j' \to i} = a_{j' \to i} \boldsymbol{W} \boldsymbol{h}_{j'}^{(t)}, \forall i \in \mathcal{V}_{cur}, j' \in \mathcal{V}_{prev}, \tag{10e}$$

$$\boldsymbol{u}_i = \sum_{j' \in \mathcal{V}_{prev}} \boldsymbol{u}_{j' \to i}, \forall i \in \mathcal{V}_{cur}, \tag{10f}$$

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_{ref,i}^{(t)} + \sum_{j \in \mathcal{N}(i)} (\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t)}) \phi^x(\boldsymbol{m}_{j \to i}), \forall i \in \mathcal{V}_{cur}, \tag{10g}$$

$$\boldsymbol{h}_i^{(t+1)} = (1 - \beta) \cdot \boldsymbol{h}_i^{(t)} + \beta \cdot \phi^h(\tilde{\boldsymbol{h}}_i^{(t)}, \boldsymbol{m}_i, \boldsymbol{u}_i, \boldsymbol{f}_i), \forall i \in \mathcal{V}_{cur}, \tag{10h}$$

where ($\mathcal{V}_{cur}$, $\mathcal{E}_{cur}$) and ($\mathcal{V}_{prev}$, $\mathcal{E}_{prev}$) refer to the subgraph currently being generated and the set of all previously generated subgraphs, *i.e.*, the current state of the molecule. ϕ^m, ϕ^e, ϕ^x, and ϕ^h refer to separate shallow MLPs for the feature mixing, edge message calculation, coordinate update, and invariant feature update, respectively. Eq. 10(a-b) creates a mixed feature for each atom consisting of the current FG invariant feature and 3D position vectors (h and x), and the previous centroid μ and respective centroid distances. Eq. 10(c-d) defines the message passing operation that uses the aforementioned mixed features $\tilde{h}$ and a series of important distances between the model-based conformer and RDKit reference.Eq. 10(e-f) apply the same unidirectional attention updates seen in the encoder architecture. Eq. 10(g-h) update the position and feature vector for each atom using the above messages and attention coefficients, with f representing the original invariant node features $h^{\ell=0}$ and β a weighted update parameter. We emphasize that Eq. 10(g) formulates the overall objective as learning the optimal distortion of the RDKit reference to achieve the low-energy position *i.e.*, $x^* = x_{ref} + \Delta x$. The CG autoregressive strategy allows CoarsenConf to handle extremely large molecules efficiently, as the max number of time steps is equal to the max number of CG beads. CoarsenConf is trained using teacher forcing (Williams and Zipser, 1989), which enables an explicit mixing of low-energy coordinates with the current FG positions from channel selection Eq. 10(a-b).

F Model Configuration

Model. We present the model configuration that was used to generate the results in §4.1 - §4.4. Overall, the default model has 1.9M parameters: 1.6M for the encoder and 300K for the decoder. We note that as CoarsenConf uses graph matching, half the encoder parameters are used for each of the two inputs representing the same molecule in different spatial orientations. For both the encoder and decoder, we use five message-passing layers, a learning rate of $1e{-}3$ with an 80% step reduction after

each epoch, and a latent space channel dimension (F) of 32. All other architectural parameters, such as feature mixing ratios or nonlinearities, were set following similar architectures (Huang et al., 2022; Deng et al., 2021; Stärk et al., 2022). We present further ablations in Appendix §J. We note that the ability to share weights between the inputs as well as between each layer in the encoder is left as a hyperparameter. This could allow the encoder to see a 2x or 5x reduction in model size, respectively.

Compute. The QM9 model was trained and validated for five epochs in 15 hours using a single 40GB A100 GPU. We used a batch size of 600, where a single input refers to two graphs: the ground truth and RDKit approximate conformer. The DRUGs model was trained and validated for five epochs in 50 hours using distributed data-parallel (DDP) with 4 40GB A100 GPUs with a batch size of 300 on each GPU. For DRUGs, the GPU utilization was, on average, 66% as few batches contain very large molecules. In the future, lower run times can be achieved if the large molecules are more intuitively spaced out in each batch.

We note DDP has a negative effect on overall model benchmark performance due to the gradient synchronization but was used due to compute constraints. Without DDP, we expect the training time to take around 7 days, which is on par with Torsional Diffusion (4-11 days). We demonstrated that CoarsenConf achieves as good or better results than prior methods with less data and time, and these results can be further optimized in future work. We provide evidence of the negative effects of DDP in Appendix §J.

Optimal Transport reduces compute requirements. The optimal transport (OT) models were trained on 2 epochs on a single A6000 GPU for 8 and 15 hours total for QM9 and DRUGS, respectively. For OT details, see Appendix Eq. 11. Here, both models use the first 5 ground truth conformers. In real-world applications like polymer design, the availability of data is frequently limited and accompanied by a scarcity of conformers for each molecule. The current datasets, QM9 and DRUGS, do not mimic this setting very well. For example, on average, QM9 has 15 conformers per molecule, and DRUGS has 104 per molecule—both datasets have significantly more conformers than in an experimental drug design setting. Given this, rather than training on the first 30 conformers as done in Torsional Diffusion, we train on the first five (typically those with the largest Boltzmann weight) for QM9 and DRUGS, respectively.

G RDKIT APPROXIMATE CONFORMER

Generating Approximate Conformers. For CoarsenConf's initial conditional approximations, we only use RDKit + MMFF when it can converge (~90% and ~40% convergence for QM9 and DRUGS, respectively). We emphasize that RDKit only throws an error when MMFF is not possible but often returns structures with a non-zero return code, which signifies incomplete and potentially inaccurate optimizations. Therefore, in generating the RDKit structures for training and evaluation, we filter for MMFF converged structures. We default to the base EKTDG-produced structures when either the optimization cannot converge, or MMFF does not yield enough unique conformers. CoarsenConf ultimately offers a solution that can effectively learn from traditional cheminformatics methods. This aspect of MMFF convergence has not been discussed in prior ML for MCG methods, and we leave it to future cheminformatics research to learn the causes and implications of incomplete optimizations.

Eliminating distribution shift with explicit conditioning. Both CoarsenConf and TD optimize $p(X|\mathcal{R})$ but utilize the RDKit approximations $\mathcal{R}$ in different ways. TD learns to update the torsion angles of $\mathcal{R}$, while CoarsenConf leverages CG information to inform geometric updates (coordinates, distances, and torsion angles) to translate $\mathcal{R}$ to X. Unlike TD, which uses a preprocessing optimization procedure to generate substitute ground truth conformers that mimic $p(\mathcal{R})$, CoarsenConf directly learns from both X and $\mathcal{R}$ through its hierarchical graph matching procedure. This directly addresses the distributional shift problem. We hypothesize that this, along with our angle-based CG strategy, leads to our observed improvements. Overall, CoarsenConf provides a comprehensive framework for accurate conformer generation that can be directly applied for downstream tasks such as oracle-based protein docking.

H GEOM BENCHMARK DISCUSSION

xTB energy and property prediction. We note the issues surrounding the RMSD metrics have always existed, and prior MCG methods have introduced energy-based benchmarks that we describe and report in Tab. 2. We note these energies are calculated with xTB, and thus are not very accurate compared to density functional theory (DFT), as it is limited by the level of theory used to produce the energies further discussed in Axelrod and Gómez-Bombarelli (2022). Therefore, since current benchmarks mainly focus on gauging the effectiveness of the machine learning objective and less on the chemical feasibility and downstream use of the generated conformers, we use oracle-based protein docking-based to evaluate conformer quality on downstream tasks. These evaluations are highly informative, as molecular docking is a crucial step in the drug discovery process, as it helps researchers identify potential drug candidates and understand how they interact with their target proteins. The combination of RMSD, xTB energy, and downstream docking tasks presents a more comprehensive evaluation of generated conformers.

I QM9 EXPERIMENTAL DETAILS

Both CoarsenConf and CoarsenConf-OT were trained on 5 conformers per ground truth molecule, compared to Torsional Diffusion's 30. We hypothesize that since CoarsenConf uses a one-to-one loss function, we are able to maintain high recall, whereas the OT model finds an optimal matching that focuses on precision. By adding more ground truth conformers, we hypothesize our model can better cover the true conformer space, improving recall, as the OT setting would not be as biased toward precision.

Table 5: Quality of generated conformer ensembles for the GEOM-QM9 test set ($\delta = 0.5$Å) in terms of Coverage (%) and Average RMSD (Å). Torsional Diffusion (TD) was benchmarked using its evaluation code and available generated molecules, per their public instructions. Note that CoarsenConf (5 epochs) was restricted to using 41% of the data used by TD (250 epochs) to exemplify a low-compute and data-constrained setting. OMEGA results were taken from Jing et al. (2022) (we were unable to run the coverage normalization).

	Recall				Precision			
	Coverage ↑		AR ↓		Coverage ↑		AR ↓	
Method	Mean	Med	Mean	Med	Mean	Med	Mean	Med
OMEGA	85.5	100.0	0.177	0.126	82.9	100.0	0.224	0.186
RDKit + MMFF	75.2	100.0	0.219	0.173	82.1	100.0	0.157	0.119
GeoMol	79.4	100.0	0.219	0.191	75.9	100.0	0.262	0.233
Torsional Diffusion	82.2	100.0	0.179	0.148	78.4	100.0	0.222	0.197
CoarsenConf	76.9	100.0	0.246	0.211	80.2	100.0	0.227	0.186
CoarsenConf-OT	56.1	50.0	0.361	0.345	80.2	100.0	0.149	0.108

J DRUGS EXTENDED BENCHMARKS

Evaluation Details. All models in Tab. 1 were benchmarked with Torsional Diffusion's (TD) evaluation code and retrained if generated molecules were not public (using their public instructions). We note that TD uses higher-order tensor products to maintain equivariance ($\ell = 2$). In contrast, GeoMol, GeoDiff, and CoarsenConf use scalar-vector operations that are theoretically analogous to $\ell = 1$. CoarsenConf-OT uses an optimal transport (OT) loss with the same decoder architecture as in Fig. 2, but is no longer autoregressive. GeoDiff's code would not load, so we were able to evaluate the GeoDiff generated DRUGS molecules from the Torsional Diffusion authors' evaluation on the same test set.

Table 6: DRUGS-Precision equivariance ablations. OMEGA (Hawkins et al., 2010) results were taken from Jing et al. (2022). All others were re-benchmarked using Torsional Diffusion's code with an error normalized Coverage score to prevent the masking out of method failures. This enforces that each method is fairly evaluated on the entire test set as now Coverage truly represents the percentage of the test set that meets the threshold criteria. OMEGA requires a commercial license, so we were unable to test the results ourselves, thus taking results from TD. As a non-ML method, we also assume OMEGA has no failures as each molecule in the test set is valid, which could artificially inflate the observed coverage scores.

	Coverage ↑		AMR ↓	
Method	Mean	Med	Mean	Med
RDKit	37.9	29.9	0.988	0.878
RDKit + MMFF	52.3	52.1	0.840	0.715
OMEGA	53.4	54.6	0.841	0.762
GeoDiff	23.7	13.0	1.131	1.083
GeoMol	40.5	33.5	0.919	0.842
Torsional Diffusion ($\ell = 1$)	48.9	50.0	0.804	0.758
Torsional Diffusion ($\ell = 2$)	52.1	53.7	0.770	0.720
CoarsenConf	43.8	35.5	0.914	0.829
CoarsenConf-OT	52.0	52.1	0.836	0.694

We copy the results from Tab. 1 and provide additional results, including TD for rotation order $\ell = 1$, OMEGA (Hawkins et al., 2010), and RDKit. This allows for a closer comparison to the scalar and vector operations that CoarsenConf employs to maintain equivariance. Using a lower rotation order results in slightly worse results in nearly all categories. We further discuss the implications of the choice in equivariant representation in Appendix §L.

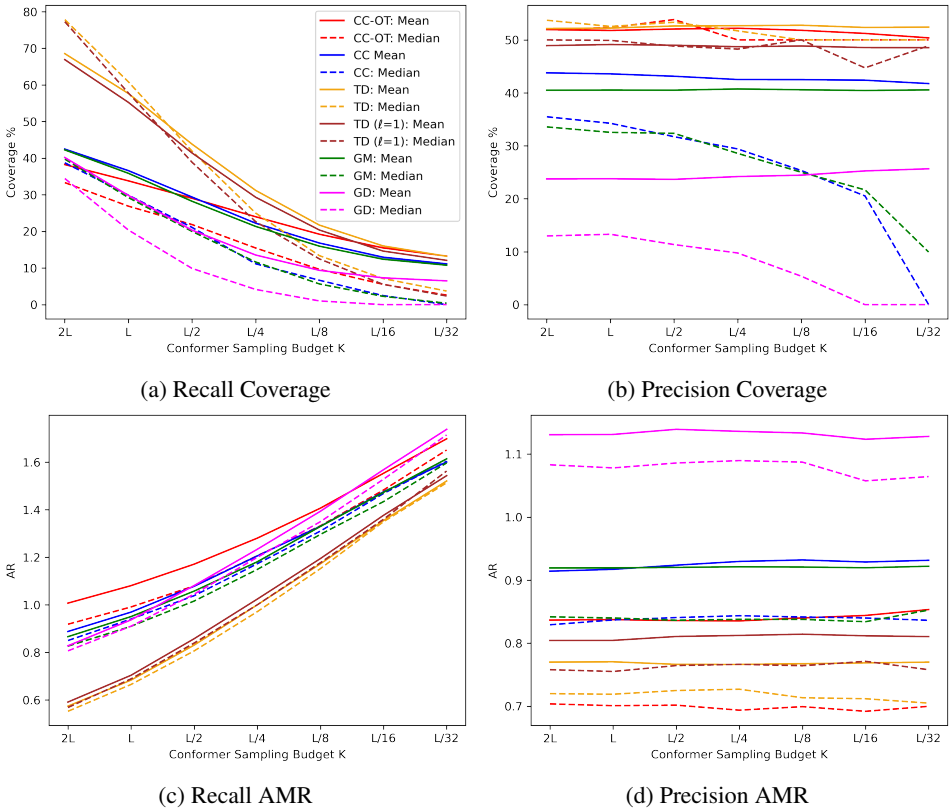
Optimal Transport. In practice, our model generates a set of conformers, $\{\mathcal{C}_k\}_{k\in[1..K]}$, that needs to match a variable-length set of low-energy ground truth conformers, $\{\mathcal{C}_l^*\}_{l\in[1..L]}$. In our case, the number L of true conformers, or the matching between generated and true conformers is not known upfront. For these reasons, we introduce an optimal transport-based, minimization-only, loss function (Ganea et al., 2021):

$$\begin{aligned} \mathcal{L}_{OT} &= \min_{\mathbf{T}\in\mathcal{Q}_{K,L}} \sum_{k,l} T_{kl}\mathcal{L}(\mathcal{C}_k, \mathcal{C}_l^*), \\ \mathcal{L}(\mathcal{C}_k, \mathcal{C}_l^*) &= \text{MSE}(\mathcal{C}_k, \mathcal{C}_l^*) + \text{distance error}(\mathcal{C}_k, \mathcal{C}_l^*), \end{aligned} \tag{11}$$

where $\mathbf{T}$ is the ***transport plan*** satisfying $\mathcal{Q}_{K,L} = \{\mathbf{T} \in \mathbb{R}_+^{K\times L} : \mathbf{T}\mathbf{1}_L = \frac{1}{K}\mathbf{1}_K, \mathbf{T}^T\mathbf{1}_K = \frac{1}{L}\mathbf{1}_L\}$. The minimization w.r.t. T is computed quickly using the Earth Mover Distance and the POT library (Flamary et al., 2021). As the OT loss focuses more on finding the optimal mapping from generated conformers to ground truth reference, we removed the autoregressive decoding path of CoarsenConf and replaced it with a single pass with the same decoder architecture. The underlying loss function, which is tasked to minimize MSE coordinate error, and interatomic distance error is the same in both (Eq. 5), the autoregressive (AR) and non-AR OT-based loss functions. The OT version additionally finds the optimal mapping between the generated and ground truth structures, which better aligns with the AMR and Coverage benchmarks.

Hyperparameter Ablations. We experimented with increasing the latent channels (F) from 32 to 64 and 128, and introducing a step-wise distance loss and KL regularization annealing schedule, as done in the QM9 experiments. Both these experiments resulted in slightly worse performance when limited to 2 conformers per training molecule. We hypothesize that due to the DRUGs molecules being much larger than those in QM9, more training may be necessary, and a more sensitive annealing schedule may be required.

GEOM-DRUGS Recall Results. Fig. 7 demonstrates extensive Precision and Recall results for a wide range of tested sampling budgets for GEOM-DRUGS. We see that only Precision is stable across nearly all values. Due to the extreme sensitivity of the Recall metric and little difference in model performance for reasonable sampling budgets, we focus on Precision for QM9 and DRUGS.



(a) Recall Coverage

(b) Precision Coverage

(c) Recall AMR

(d) Precision AMR

Figure 7: GEOM-DRUGS evaluation as a function of number of generated conformers. GEOM-DRUGS has 104 conformers per molecule on average. Recall is heavily dependent on the sampling budget. Precision is mostly stable. Lower AMR and higher coverage is better, but coverage is set by an arbitrary threshold, which in this case is 0.75Å. Results show CoarsenConf (CC), Torsional Diffusion (TD), GeoMol (GM), and GeoDiff (GD).

We also note that while CoarsenConf-OT saw worse recall results for QM9, this was not the case for DRUGS. In the case of DRUGS, CoarsenConf-OT achieves the learning objective of instilling force field optimizations as the lower error bound and does so with very little training and inference time.

K Oracle-based Protein Docking

We utilize the oracle-based protein docking task as molecules with higher affinity (more negative) have more potential for higher bioactivity, which is significant for real-world drug discovery. We use the CrossDocked2020 trainset consisting of 166000 protein-ligand interactions (2,358 unique proteins and 11,735 unique ligands) and its associated benchmarks, as it has been heavily used in Structure-based drug discovery as defined by Peng et al. (2022); Guan et al. (2023).

The CrossDocked2020 dataset is derived from PDBBind but uses smina (Koes et al., 2013), a derivative of AutoDock Vina with more explicit scoring control, to generate the protein-conditioned ligand structures to yield ground truth data. We note that based on the raw data, 2.2 billion conformer-protein interactions are possible, but we filtered out any ground-truth example that AutoDock Vina failed to score. Furthermore, in the TDC oracle-based task, each ligand is known to fit well in the given protein. CrossDocked2020, on the other hand, consists of various ligand-protein interactions, not all of which are optimal, making the overall task more difficult.

We note that while it takes on the orders of hours to generate 1.2M conformers (100 conformers per molecule), it takes on the orders of ~weeks to months to score each conformer for up to the 2,358 unique proteins for each evaluated method (evaluation time is 100x the time to score the ground truth

data as we generate 100 conformers per molecule). As a result, we report the results for the first 100,000 conformer-protein interactions.

L LIMITATIONS

As demonstrated in §4.1-§4.4, CoarsenConf significantly improves the accuracy and reduces the overall data usage and runtime for conformer generation. However, CoarsenConf also has some limitations that we will discuss in this section.

Autoregressive generation. While CoarsenConf improves accuracy with reduced training time and overall data, autoregressive generation is the main bottleneck in inference time. We linearize the input molecule based on spatially significant subgraphs and then process each one autoregressively. For a model with k torsion angles, we need $k + 1$ passes through our decoder. Coarse-graining is an effective strategy to reduce the number of decoder passes compared to traditional atom-wise autoregressive modeling. For example, for a given molecule, the number of torsion angles (which we use to coarse-grain) is significantly less than the number of atoms. Our choice of coarse-graining strategy allows us to break the problem into more manageable subunits, making autoregressive modeling a useful strategy, as it provides greater flexibility and control by allowing conditional dependence. CoarsenConf is a good example of the trade-offs that exist between generative flexibility and speed. We target this limitation by introducing a non-autoregressive version with an optimal transport loss. We see this improves the overall GEOM results, at the slight cost of a higher right tail of the error distribution.

Optimal Transport. While CoarsenConf-OT trained with a non-autoregressive decoder with an optimal transport loss significantly outperforms prior methods and accomplishes the goal of effectively learning from traditional cheminformatics methods, the recall results still have room for improvement, especially for QM9. While CoarsenConf (no OT) achieves competitive results, we believe that continuing to focus on how to better integrate physics and cheminformatics into machine learning will be crucial for improving downstream performance. Due to the above concerns, we chose to evaluate our non-OT model on the property prediction and protein docking tasks, as wanted to use our best model denoted by the lowest overall RMSD error distribution.

Approximate structure error. The success of learning the optimal distortion between low-energy and RDKit approximate structure depends on having reasonable approximations. While CoarsenConf relaxes the rigid local structure assumption of Torsional Diffusion in a way that leverages the torsional flexibility in molecular structures, it still depends on an approximate structure. This is a non-issue in some instances, as RDKit does well. In more experimental cases for larger systems, the RDKit errors may be too significant to overcome. We emphasize that the underlying framework of CoarsenConf is adjustable and can learn from scratch, not only the distortion from approximate RDKit structures. In some cases, this may be more appropriate if the approximations have particularly high error. We leave to future work to explore the balance between the approximation error and the inductive bias of learning from approximate structures, as well as methods to maintain flexibility while avoiding the issues of conditioning on out-of-distribution poor approximations.

Equivariance. As CoarsenConf uses the EGNN (Satorras et al., 2021) framework as its equivariant backbone and thus only scalar and vector operations, there is no simple way to incorporate higher-order tensors. As the value of using higher-order tensors is still actively being explored, and in some cases, the costs outweigh the benefits, we used simple scalar and vector operations and avoided expensive tensor products. We leave exploring the use of higher-order equivariant representations to future work, as it is still an ongoing research effort (Han et al., 2022).