PEPTUNE: *De Novo* GENERATION OF THERAPEUTIC PEPTIDES WITH MULTI-OBJECTIVE-GUIDED DISCRETE DIFFUSION

Sophia Tang,[1,2,*] **Yinuo Zhang,**[1,3,*] **Pranam Chatterjee**[1,4,5,†]

[1]Department of Biomedical Engineering, Duke University
[2]Management and Technology Program, University of Pennsylvania
[3]Center of Computational Biology, Duke-NUS Medical School
[4]Department of Computer Science, Duke University
[5]Department of Biostatistics and Bioinformatics, Duke University

[*]These authors contributed equally
[†]Corresponding author: pranam.chatterjee@duke.edu

ABSTRACT

We present **PepTune**, a multi-objective discrete diffusion model for the simultaneous generation and optimization of therapeutic peptide SMILES. Built on the Masked Discrete Language Model (MDLM) framework, PepTune ensures valid peptide structures with bond-dependent masking schedules and penalty-based objectives. To guide the diffusion process, we propose a Monte Carlo Tree Search (MCTS)-based strategy that balances exploration and exploitation to iteratively refine Pareto-optimal sequences. MCTS integrates classifier-based rewards with search-tree expansion, overcoming gradient estimation challenges and data sparsity. Using PepTune, we generate diverse, chemically modified peptides simultaneously optimized for multiple therapeutic properties, including target binding affinity, membrane permeability, solubility, hemolysis, and non-fouling for various disease-relevant targets. In total, our results demonstrate that MCTS-guided masked discrete diffusion is a powerful and modular approach for multi-objective sequence design in discrete state spaces.

1 INTRODUCTION

Peptides possess unique advantages as a therapeutic modality, including their low cytotoxicity and structural flexibility to bind to a diverse set of binding motifs without requiring stable binding pockets, making them ideal for targeting structurally diverse protein surfaces [14, 79]. However, peptides containing only the 20 wild-type amino acids have limitations including susceptibility to enzymatic degradation and low membrane permeability [79]. To overcome these limitations, non-natural amino acids (nAAs) containing diverse chemical modifications to the peptide backbone and side chains have been integrated into peptides to enhance their therapeutic properties. Despite this progress in peptide drug development, searching for the vast space of chemically modified peptides remains a major limitation [56, 77]. This motivates the development of generative deep learning models that can effectively learn the space of clinically relevant peptides and sample *de novo* peptides conditioned with various therapeutic properties.

Generative diffusion models are considered state-of-the-art for *de novo* binder design, but they often rely on stable tertiary structures of target proteins [65, 7, 60, 44], precluding the design of peptide binders to disordered and dynamic targets, which drive a sizable portion of diseases [75]. Generative peptide design language models that depend only on the target sequence ([4, 10]) have demonstrated robust success on disordered and structurally diverse targets, but their use of only 20 wild-type amino acids limits these models from sampling from the space of chemically modified or cyclic peptides. Furthermore, discrete generative models still face significant limitations in multi-objective-guided generation and optimization [3, 51, 70, 68, 21, 64, 15]. Classifier-based and classifier-free

guidance strategies have been explored to steer discrete diffusion objectives toward specific properties [81, 64, 73, 58], yet these approaches often struggle with conflicting objectives, gradient estimation, and the sparsity of quality data.

In this work, we introduce **PepTune**, the first multi-objective-guided discrete diffusion model for *de novo* peptide SMILES generation. Our key contributions include:

1. **Masked Diffusion Language Model for Peptide SMILES.** We introduce the first discrete diffusion model for generating peptide SMILES with non-canonical amino acids and cyclic modifications.
2. **Bond-Dependent Masking Schedule.** We derive the NELBO and reverse-posterior for a bond-dependent masking schedule which increases the structural validity of our generated peptide SMILES.
3. **Global Sequence Invalid Loss.** We introduce a novel invalid loss based on our peptide SMILES filter that efficiently propagates penalties from a discrete sequence to continuous probability distributions.
4. **Monte Carlo Tree Search (MCTS) for Classifier-Based Multi-Objective Guidance.** We develop a robust framework for multi-objective guidance for discrete diffusion by iteratively refining Pareto-optimal peptide sequences across therapeutic properties without gradient estimation or re-training.
5. **Property Prediction Toolkit.** We train a set of classifiers and regressors for binding affinity, membrane permeability, solubility, hemolysis, and non-fouling.

2 RESULTS

2.1 MASKED DISCRETE DIFFUSION FOR PEPTIDE SMILES GENERATION

We based our unconditional generator on the Masked Diffusion Language Model (MDLM) framework which learns to reconstruct clean sequences from sequences corrupted with [MASK] tokens (Figure 11) [68, 70, 59, 94].

Extending previous work in state-dependent masking [70], we devised a bond-dependent masking schedule $\alpha_t(\mathbf{x}_0)$ such that peptide bond tokens are masked at a slower rate than non-peptide-bond tokens during the forward diffusion process. We derive the parameterized reverse posterior $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$ and NELBO loss demonstrating that our bond-dependent masking scheme encourages the model to first unmask peptide bond tokens before filling in the side-chain tokens, effectively reducing generation of invalid peptides due to slight inaccuracies in the fundamental peptide backbone (Methods B.2, Appendix D.2 for reverse posterior derivation, Appendix D.3 for NELBO derivation).

$$\alpha_t(\mathbf{x}_0) = \begin{cases} 1 - t^w & \mathbf{x}_0 = \mathbf{b} \\ 1 - t & \mathbf{x}_0 \neq \mathbf{b} \end{cases} \tag{1}$$

In addition to enforcing peptide bond structure, we find that small syntactical errors during the sampling can result in completely invalid SMILES. Therefore, we introduce an additional invalid loss term $\mathcal{L}_{\text{invalid}}$ (Equation 10) for when the sequence sampled from taking the `argmax` of the predicted probabilities for each training step corresponds to an invalid peptide SMILES based on our peptide SMILES validity filter (Methods B.6).

Our optimized unconditional MDLM (PepMDLM) shows increased uniqueness and diversity with lower SNN compared to the autoregressive generator of macrocyclic peptides, HELM-GPT [86], demonstrating our capability to comprehensively search the sub-space of valid peptide SMILES (Table 5). Furthermore, PepMDLM generates valid peptides with a higher average nAA frequency than experimentally-validated peptide SMILES for membrane permeability and binding affinity (Figure 12), demonstrating our unique ability to design *de novo* peptides with cyclic and nAA modifications and expanding the search space of therapeutic peptides well beyond any generative model trained on canonical amino acid representations.

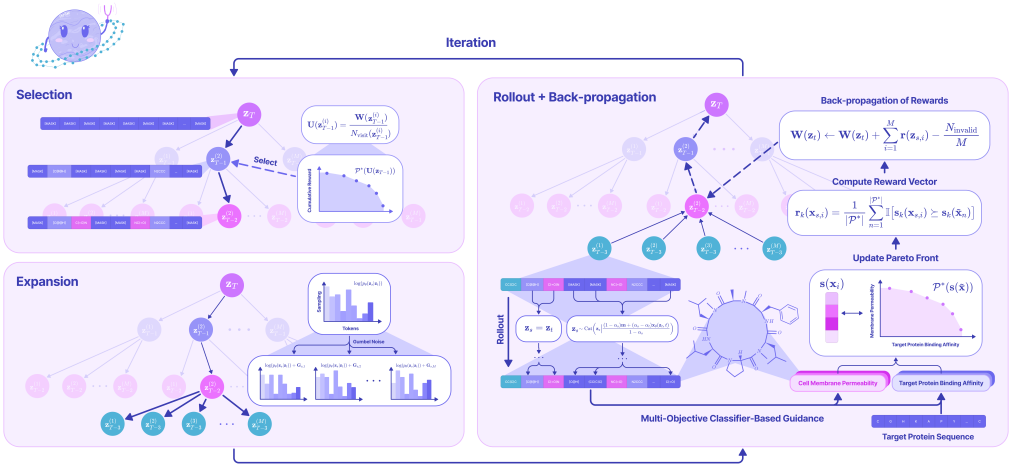


Figure 1: **PepTune**. PepTune is a multi-objective discrete diffusion model guided by Monte-Carlo Tree Search (MCTS). The full algorithm is detailed in Algorithm 3.

2.2 Multi-Objective Guidance on Therapeutic Properties for Discrete Diffusion

We introduce a classifier-based multi-objective guidance strategy to generate a set of non-dominated peptide SMILES $\mathcal{P}^*$ using Monte Carlo Tree Search (MCTS) (Figure 1). A sequence is non-dominated when no other sequence has a strictly larger score in one or more objectives while maintaining equal scores for the remaining objectives. We leverage the unique capability of our unconditional MDLM model to sample from the diverse space of peptide SMILES and our trained property classifiers to search the constrained space of peptides with therapeutically optimal properties through an iterative selection, expansion, rollout, and backpropagation loop (Methods B.4).

The **PepTune** multi-objective guidance framework is defined as follows. We start each iteration from a fully masked sequence $\mathbf{z}_{t(T)}$, defining the root node of the MCTS tree. During the selection step, we traverse an optimal path through the MCTS tree defined as a series of partial unmasking steps that have been previously traversed based on the selection score vector $\mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i})$ (defined in Equation 16), indicating the likelihood that the given step generates a Pareto non-dominated sequence. Upon reaching a terminal leaf node $\mathbf{z}_t$ defined as a partially masked sequence at time $t > 0$ that has yet to be further unmasked, we expand the leaf node to explore M different possible unmasking steps by batched Gumbel unmasking (Equation 18), which applies independently sampled Gumbel noise vectors to the token probabilities predicted by the MDLM backbone for sampling each child node $\mathbf{z}_{s,i}$ for $i = 1 \dots M$, enforcing diverse unmasking schemes while remaining consistent with the predicted token distribution.

Each expanded node is unmasked with greedy Gumbel-max sampling to obtain a fully unmasked sequence $\mathbf{x}_{s,i}$. For all i, we compute a K-dimensional score vector for K objectives and compare it with the scores of the current pool of Pareto non-dominated sequences $\mathcal{P}^*$ to generate a reward vector $\mathbf{r}(\mathbf{x}_{s,i}) \in \mathbb{R}^K$, where the entry $r_k(\mathbf{x}_{s,i})$ at index k is a value in $[0, 1]$ indicating the Pareto-optimality of the sequence for the kth objective compared to the current pool of Pareto non-dominated $\mathcal{P}^*$. After computing the reward vector for all valid sequences generated during rollout, we take the sum of the reward vectors subtracted by a penalty score proportional to the fraction of sampled SMILES that are not peptides and add the resulting K-dimensional vector to the cumulative reward vectors $\mathbf{W}(\mathbf{z}_t)$ of all predecessor nodes on the path to the root node $\mathbf{z}_{t(T)}$, which determines the selection steps of proceeding iterations.

To generate the property scores given an input peptide SMILES, we train regression models for target-binding affinity and cell membrane permeability and binary classification models for solubility, hemolysis, and non-fouling specifically on peptide SMILES data (Appendix B.5).

Even though the selection process favors high-reward unmasking steps, we show that the resulting pool of generated peptides retains similar uniqueness and diversity scores to the peptides generated by our unconditional MDLM and in the training dataset (Table 1). In addition, the fraction of valid peptides consistently reaches 100% after only 20 iterations of the MCTS search algorithm, demonstrating the effectiveness of backpropagating the classifier-based rewards and invalidity penalty.

Table 1: Evaluation metrics for generative quality of peptide SMILES sequences of max token length set to 200.[1]

Model	**Validity** (↑)	**Uniqueness** (↑)	**Diversity** (↑)	**SNN** (↓)	**Randomness** (↑)	**KL-Divergence** (↑)
Data	1.000	1.000	0.885	1.000	4.55	0 (Reference)
PepMDLM	0.450	1.000	0.705	0.513	4.11	0.174
PepTune	1.000	1.000	0.677	0.486	4.12	0.173

2.3 Case Study for Multi-Objective Generation of Peptide Binders

Given the significant development of glucagon-like peptide-1 (GLP-1R) peptide agonists for the treatment of type-2 diabetes and obesity [2], we compared GLP-1R binding affinity-conditioned peptides generated using PepTune with recent blockbuster GLP-1R agonists: semaglutide and liraglutide. Both semaglutide and liraglutide are over 30 amino acids in length and act by mimicking the binding of natural GLP-1 by binding to the activation pocket of GLP-1R with high precision (Figure 2) [52, 57].

Shorter agonists or antagonists to GLP-1R would serve several benefits to the treatment of insulin-related disorders, including reduced cost and complexity of synthesis, lower immunogenicity, and faster tissue penetration. Therefore, we sought to generate shorter-chain peptides that are capable of binding to GLP-1R with comparable affinity to the existing agonists. We first generated a pool of peptide binders conditioned on binding affinity with the GLP-1R sequence, solubility, hemolysis, and non-fouling. After selecting the peptides with the highest predicted binding affinity scores from the Pareto non-dominated set, we performed docking and determined docking scores of -7.4 kcal/mol and -7.0 kcal/mol for the two best candidates. Our peptides show superior docking affinity to GLP-1R while interacting at overlapping binding motifs to semaglutide and liraglutide derived from the natural hormone ligand, GLP-1 (Figure 2). These results suggest that our PepTune-derived peptides can serve as potent agonists or antagonists of GLP-1R signaling.

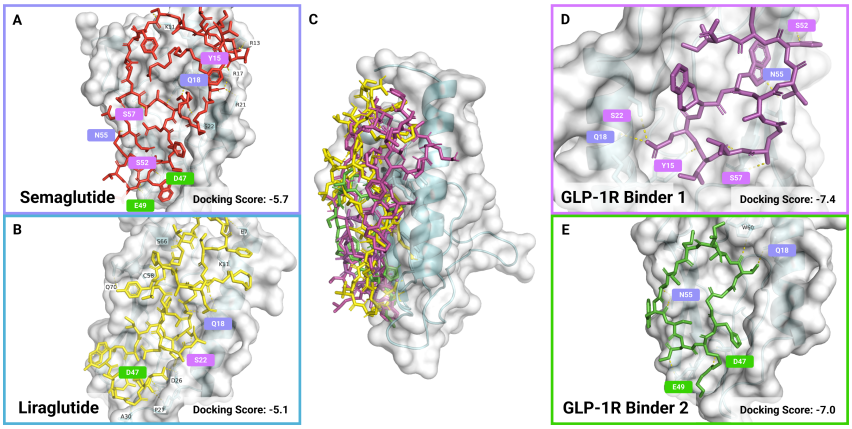


Figure 2: **Comparison of docked PepTune-generated peptides to existing GLP-1R agonists.** (**A, B**) Docking images of semaglutide (score: -5.7 kcal/mol) and liraglutide (score: -5.1 kcal/mol) binding to GLP-1R. **(C)** Full view of the positive control GLP-1R agonists and the PepTune-generated binders on GLP-1R. (**D, E**) Docking images of binder 1 (score: -7.4 kcal/mol) and 2 (score: -7.0 kcal/mol) were generated using PepTune conditioned on predicted affinity to GLP-1R, solubility, hemolysis, and non-fouling. Shared polar contacts between binder 1 and either controls are highlighted in pink, shared polar contacts between binder 2 and either controls are highlighted in green, and the shared contacts across both binders are highlighted in purple.

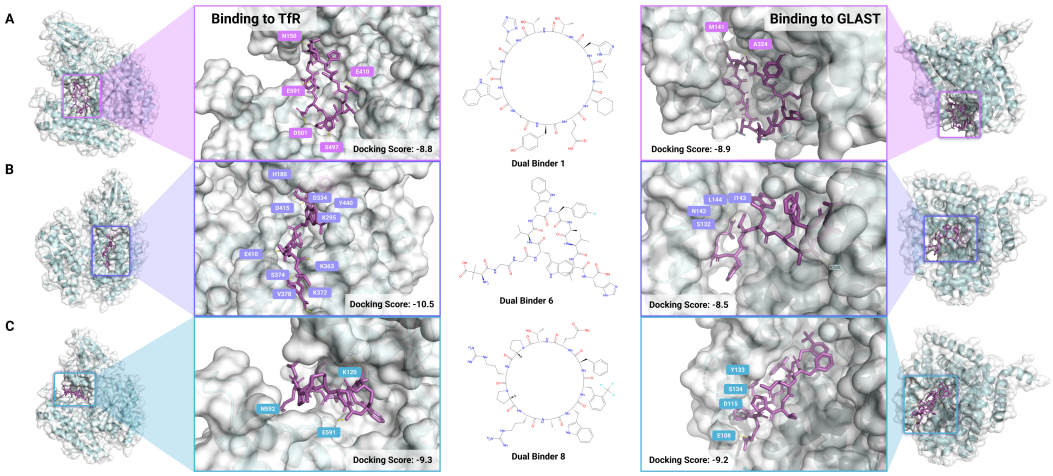


Figure 3: **PepTune-generated bi-specific peptides to TfR and GLAST.** Full protein binding location and close-up binding position for (**A**) dual binder 1, (**B**) dual binder 6, and (**C**) dual binder 8 with TfR (left) and GLAST (right). Polar contacts within 3.5 Å are highlighted.

2.4 Case Study for Dual-Target Binding Peptides

Multi-target drug discovery is of significant interest in various fields including cancer therapeutics and drug delivery for neurological disorders given their ability to perform multiple different functions such as binding to biological barriers like the blood-brain barrier, penetrating target cells, and inhibiting protein-protein interactions [45], [9].

To evaluate PepTune's capabilities in multi-target guidance, we generate bi-specific peptide binders to TfR and glutamate-aspartate transporter (GLAST) protein abundant on the surface of astrocytes, a type of glial cell in the brain. Successfully generating these peptides can facilitate BBB-crossing via TfR binding and uptake in astrocytes via GLAST binding for intravenous delivery of therapeutics for a multitude of neurological disorders where astrocytes are involved, including Alexander disease [43], Alzheimer's disease [28], Parkinson's disease [89], Huntington's disease [35], multiple sclerosis [85], and several psychiatric disorders [53].

We generated 100 peptide binders conditioned on five properties: predicted binding affinity to TfR, binding affinity to GLAST, solubility, hemolysis, and non-fouling. Remarkably, all property scores improved over iterations, with final solubility, hemolysis, and non-fouling scores surpassing binders conditioned solely on TfR binding affinity (Figure 9). This highlights that our multi-target guidance strategy avoids significant property trade-offs. To validate binding to both TfR and GLAST, we selected seven binders and docked them against each target. All seven achieved docking scores $\leq$ -7.5 kcal/mol, with the top binder scoring -10.5 kcal/mol for TfR and -9.2 kcal/mol for GLAST (Table 2). These top binders exhibited diverse secondary structures (Figure 3), positive solubility, and low hemolysis probabilities (Table 2). The binding positions and polar interactions varied across candidates, demonstrating PepTune's ability to discover diverse, optimal peptides with strong dual-target binding independent of specific motifs.

3 Conclusion

In this work, we introduce PepTune, a generative framework for multi-objective optimization directly in discrete sequence space. PepTune introduces a novel application of Monte-Carlo Tree Search (MCTS) to diffusion for the generation of Pareto-optimal peptide SMILES sequences optimized for therapeutic properties such as binding affinity, solubility, membrane permeability, and hemolysis. PepTune performs guidance in the discrete state space overcoming limitations of gradient estimation [58] and projections between continuous and discrete spaces [26]. Crucially, unlike recent binder design methods [65, 60, 44, 83], PepTune requires no 3D target structures, enabling the design of peptides for conformationally diverse proteins optimized for properties beyond local geometric interactions.

4 DECLARATIONS

Acknowledgments. We thank the Duke Compute Cluster, Pratt School of Engineering IT department, and Mark III Systems, for providing database and hardware support that has contributed to the research reported within this manuscript. We thank Alexander Tong for reviewing the theoretical formulations of PepTune. We also thank Sophia Vincoff and Lauren Hong for their assistance with figure generation.

Author Contributions. S.T. devised and developed PepTune architecture and theoretical formulations, and trained and benchmarked generation, prediction, and sampling models. Y.Z. advised on model design and theoretical framework, trained classifier models, and performed molecular docking. S.T. drafted the manuscript and S.T. and Y.Z. designed the figures. P.C. conceived, designed, supervised, and directed the study, and reviewed and finalized the manuscript.

Data and Materials Availability. Our peptide filtering, analysis, and visualization tool, SMILES2PEPTIDE, is freely available on HuggingFace: `https://huggingface.co/spaces/ChatterjeeLab/SMILES2PEPTIDE`. The PepTune codebase is freely accessible to the academic community via a non-commercial license at `https://huggingface.co/ChatterjeeLab/PepTune`.

Funding Statement. This research was supported by NIH grant R35GM155282 as well as a gift from the EndAxD Foundation to the lab of P.C.

Competing Interests. P.C. is a co-founder of Gameto, Inc. and UbiquiTx, Inc. and advises companies involved in peptide therapeutics development. P.C., S.T., and Y.Z. have and are currently filing patent applications related to this work. P.C.'s interests are reviewed and managed by Duke University in accordance with their conflict-of-interest policies.

REFERENCES

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *International Conference on Knowledge Discovery and Data Mining*, pp. 2623–2631, 2019.

[2] Nasreen Alfaris, Stephanie Waldrop, Veronica Johnson, Brunna Boaventura, Karla Kendrick, and Fatima Cody Stanford. Glp-1 single, dual, and triple receptor agonists for treating type 2 diabetes and obesity: a narrative review. *eClinicalMedicine*, 75:102782, 2024.

[3] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 2021.

[4] Suhaas Bhat, Kalyan Palepu, Lauren Hong, Joey Mao, Tianzheng Ye, Rema Iyer, Lin Zhao, Tianlai Chen, Sophia Vincoff, and Rio et al. Watson. De novo design of peptide binders to conformationally diverse targets with contrastive language modeling. *Science Advances*, 11(4), 2025.

[5] Yunke Bi, Lisha Liu, Yifei Lu, Tao Sun, Chen Shen, Xinli Chen, Qinjun Chen, Sai An, Xi He, and Chunhui et al. Ruan. T7 peptide-functionalized peg-plga micelles loaded with carmustine for targeting therapy of glioma. *ACS Applied Materials & Interfaces*, 8(41):27465–27473, 2016.

[6] Michael Brenner, Anne B. Johnson, Odile Boespflug-Tanguy, Diana Rodriguez, James E. Goldman, and Albee Messing. Mutations in gfap, encoding glial fibrillary acidic protein, are associated with alexander disease. *Nature Genetics*, 27(1):117–120, 2001.

[7] Patrick Bryant and Arne Elofsson. Peptide binder design with inverse folding and protein structure prediction. *Communications Chemistry*, 6(1), 2023.

[8] Lulu Cai, Chuanyao Yang, Wenfeng Jia, Yuwei Liu, Rou Xie, Ting Lei, Zhihang Yang, Xueqin He, Rongsheng Tong, and Huile Gao. Endo/lysosome-escapable delivery depot for improving bbb transcytosis and neuron targeted therapy of alzheimer's disease. *Advanced Functional Materials*, 30(27), 2020.

[9] Lai Yue Chan, David J. Craik, and Norelle L. Daly. Dual-targeting anti-angiogenic cyclic peptides as potential drug leads for cancer therapy. *Scientific Reports*, 6(1), 2016.

[10] Tianlai Chen, Madeleine Dumas, Rio Watson, Sophia Vincoff, Christina Peng, Lin Zhao, Lauren Hong, Sarah Pertsemlidis, Mayumi Shaepers-Cheu, and Tian Zi et al. Wang. Pepmlm: Target sequence-conditioned generation of therapeutic peptide binders via span masked language modeling. *arXiv preprint arXiv:2310.03842*, 2023.

[11] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

[12] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *arXiv preprint arXiv:1603.02754*, 2016.

[13] Tong Chen, Yinuo Zhang, and Pranam Chatterjee. moppit: De novo generation of motif-specific binders with protein language models. *bioRxiv*, 2024.

[14] Chi V. Dang, E. Premkumar Reddy, Kevan M. Shokat, and Laura Soucek. Drugging the "undruggable" cancer targets. *Nature Reviews Cancer*, 17(8):502–508, 2017.

[15] Oscar Davis, Samuel Kessler, Mircea Petrache, Ismail Ilkan Ceylan, Michael Bronstein, and Avishek Joey Bose. Fisher flow matching for generative modeling over discrete data. *Advances in Neural Information Processing Systems*, 2024.

[16] Nathalie di Clemente, Chrystèle Racine, and Rodolfo A. Rey. Anti-müllerian hormone and polycystic ovary syndrome in women and its male equivalent. *Biomedicines*, 10(10):2506, 2022.

[17] Fergal J. Duffy, Mélanie Verniere, Marc Devocelle, Elise Bernard, Denis C. Shields, and Anthony J. Chubb. Cyclops: Generating virtual libraries of cyclized and constrained peptides including nonnatural amino acids. *Journal of Chemical Information and Modeling*, 51(4): 829–836, 2011.

[18] Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. Autodock vina 1.2. 0: New docking methods, expanded force field, and python bindings. *Journal of chemical information and modeling*, 61(8):3891–3898, 2021.

[19] Lawrence F. Eng. Glial fibrillary acidic protein (gfap): the major protein of glial intermediate filaments in differentiated astrocytes. *Journal of Neuroimmunology*, 8:203–214, 1985.

[20] Aaron L Feller and Claus O Wilke. Peptide-aware chemical language model successfully predicts membrane diffusion of cyclic peptides. *bioRxiv*, 2024.

[21] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 2024.

[22] David Gfeller, Olivier Michielin, and Vincent Zoete. Swisssidechain: a molecular and structural database of non-natural sidechains. *Nucleic Acids Research*, 41(D1):D327–D332, 2012.

[23] Sara Gosk, Charlotte Vermehren, Gert Storm, and Torben Moos. Targeting anti—transferrin receptor antibody (ox26) and ox26-conjugated liposomes to brain capillary endothelial cells using in situ perfusion. *Journal of Cerebral Blood Flow & Metabolism*, 24(11), 2004.

[24] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? *Advances in Neural Information Processing Systems*, 2022.

[25] Alice Grossi, Francesca Rosamilia, Silvia Carestiato, Ettore Salsano, Isabella Ceccherini, and Tiziana Bachetti. A systematic review and meta-analysis of gfap gene variants in alexander disease. *Scientific Reports*, 14(1), 2024.

[26] Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. *Advances in Neural Information Processing Systems*, 2023.

[27] Chakradhar Guntuboina, Adrita Das, Parisa Mollaei, Seongwon Kim, and Amir Barati Farimani. Peptidebert: A language model based on transformers for peptide property prediction. *The Journal of Physical Chemistry Letters*, 14(46):10427–10434, 2023.

[28] Naomi Habib, Cristin McCabe, Sedi Medina, Miriam Varshavsky, Daniel Kitsberg, Raz Dvir-Szternfeld, Gilad Green, Danielle Dionne, Lan Nguyen, and Jamie L. et al. Marshall. Disease-associated astrocytes in alzheimer's disease and aging. *Nature Neuroscience*, 23(6):701–706, 2020. ISSN 1546-1726.

[29] Kaitlin N. Hart, William A. Stocker, Nicholas G. Nagykery, Kelly L. Walton, Craig A. Harrison, Patricia K. Donahoe, David Pépin, and Thomas B. Thompson. Structure of amh bound to amhr2 provides insight into a unique signaling pair in the tgf-β family. *Proceedings of the National Academy of Sciences*, 118(26), 2021.

[30] William M. Hewitt, Siegfried S. F. Leung, Cameron R. Pye, Alexandra R. Ponkey, Maria Bednarek, Matthew P. Jacobson, and R. Scott Lokey. Cell-permeable cyclic peptides from synthetic libraries inspired by natural products. *Journal of the American Chemical Society*, 137 (2):715–721, 2015.

[31] Elly M Hol and Milos Pekny. Glial fibrillary acidic protein (gfap) and the astrocyte intermediate filament system in diseases of the central nervous system. *Current Opinion in Cell Biology*, 32, 2015.

[32] Sandrine Imbeaud, Corinne Belville, Liza Messika-Zeitoun, Rodolfo Rey, Nathalie di Clemente, Nathalie Josso, and Jean-Yves Picard. A 27 base-pair deletion of the anti-müllerian type ii receptor gene is the most common cause of the persistent müllerian duct syndrome. *Human molecular genetics*, 5(9):1269–1277, 1996.

[33] Wilfred A. Jefferies, Malcolm R. Brandon, Simon V. Hunt, Alan F. Williams, Kevin C. Gatter, and David Y. Mason. Transferrin receptor on endothelium of brain capillaries. *Nature*, 312 (5990), 1984.

[34] Kasper Bendix Johnsen, Annette Burkhart, Fredrik Melander, Paul Joseph Kempen, Jonas Bruun Vejlebo, Piotr Siupka, Morten Schallburg Nielsen, Thomas Lars Andresen, and Torben Moos. Targeting transferrin receptors at the blood-brain barrier improves the uptake of immunoliposomes and subsequent cargo transport into the brain parenchyma. *Scientific Reports*, 7(1), 2017.

[35] Baljit S. Khakh, Vahri Beaumont, Roger Cachope, Ignacio Munoz-Sanjuan, Steven A. Goldman, and Rosemarie Grantyn. Unravelling and exploiting astrocyte dysfunction in huntington's disease. *Trends in Neurosciences*, 40(7):422–437, 2017.

[36] Gyeungyun Kim, Minkyung Kim, Youngki Lee, Jung Woo Byun, Do Won Hwang, and Minhyung Lee. Systemic delivery of microrna-21 antisense oligonucleotides to the brain using t7-peptide decorated exosomes. *Journal of Controlled Release*, 317:273–281, 2020.

[37] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in Neural Information Processing Systems*, 2021.

[38] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, 2020.

[39] Yuyang Kuang, Sai An, Yubo Guo, Shixian Huang, Kun Shao, Yang Liu, Jianfeng Li, Haojun Ma, and Chen Jiang. T7 peptide-functionalized nanoparticles utilizing rna interference for glioma dual targeting. *International Journal of Pharmaceutics*, 454(1):11–20, 2013.

[40] Leandros Lazaros, Anthi Fotaki, Christina Pamporaki, Elissavet Hatzi, Chrysoula Kitsou, Athanasios Zikopoulos, Christina Virgiliou, Ioannis Kosmas, Ioanna Bouba, Theodoros Stefos, et al. The ovarian response to standard gonadotropin stimulation is influenced by amhrii genotypes. *Gynecological Endocrinology*, 32(8):641–645, 2016.

[41] Jae H. Lee, Jeffrey A. Engler, James F. Collawn, and Bryan A. Moore. Receptor mediated uptake of peptides that bind the human transferrin receptor. *European Journal of Biochemistry*, 268(7), 2001.

[42] Jia-Ning Li, Guang Yang, Peng-Cheng Zhao, Xue-Xin Wei, and Jian-Yu Shi. Cpromg: controllable protein-oriented molecule generation with desired binding affinity and drug-like properties. *Bioinformatics*, 39:i326–i336, 2023.

[43] Li Li, E Tian, Xianwei Chen, Jianfei Chao, Jeremy Klein, Qiuhao Qu, Guihua Sun, Guoqiang Sun, Yanzhou Huang, Charles D. Warden, and Peng et al. Ye. Gfap mutations in astrocytes impair oligodendrocyte progenitor proliferation and myelination in an hipsc model of alexander disease. *Cell Stem Cell*, 23(2):239–251.e6, 2018.

[44] Qiuzhen Li, Efstathios Nikolaos Vlachos, and Patrick Bryant. Design of linear and cyclic peptide binders of different lengths from protein sequence information. *bioRxiv preprint bioRxiv:10.1101/2024.06.20.599739*, 2024.

[45] Xiaoxu Li, Ximing Pu, Xingming Wang, Juan Wang, Xiaoming Liao, Zhongbin Huang, and Guangfu Yin. A dual-targeting peptide for glioblastoma screened by phage display peptide library biopanning combined with affinity-adaptability analysis. *International Journal of Pharmaceutics*, 644:123306, 2023.

[46] Yanyan Li, Honghong Zhou, Xiaomin Chen, Yu Zheng, Quan Kang, Di Hao, Lili Zhang, Tingrui Song, Huaxia Luo, Yajing Hao, Runsheng Chen, Peng Zhang, and Shunmin He. Smprot: A reliable repository with comprehensive annotation of small proteins identified from ribosome profiling. *Genomics, Proteomics & Bioinformatics*, 19(4):602–610, 2021.

[47] Meng Liang, Chunhong Gao, Yuli Wang, Wei Gong, Shiyao Fu, Lin Cui, Zhenhan Zhou, Xiaoyang Chu, Yue Zhang, and Qianqian et al. Liu. Enhanced blood–brain barrier penetration and glioma therapy mediated by t7 peptide-modified low-density lipoprotein particles. *Drug Delivery*, 25(1):1652–1663, 2018.

[48] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

[49] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, and Yaniv et al. Shmueli. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[50] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

[51] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *International Conference on Machine Learning*, 2024.

[52] Manoj Kumar Mahapatra, Muthukumar Karuppasamy, and Biswa Mohan Sahoo. Semaglutide, a glucagon like peptide-1 receptor agonist with cardiovascular benefits for management of type 2 diabetes. *Reviews in Endocrine and Metabolic Disorders*, 23(3):521–539, 2022.

[53] Mario Martin-Fernandez, Stephanie Jamison, Laurie M Robin, Zhe Zhao, Eduardo D Martin, Juan Aguilar, Michael A Benneyworth, Giovanni Marsicano, and Alfonso Araque. Synapse-specific astrocyte gating of amygdala-related behavior. *Nature Neuroscience*, 20(11):1540–1548, 2017.

[54] David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, and María et al. Gordillo-Marañón. Chembl: towards direct deposition of bioassay data. *Nucleic Acids Research*, 47(D1):D930–D940, 2018.

[55] Garrett M Morris, Ruth Huey, William Lindstrom, Michel F Sanner, Richard K Belew, David S Goodsell, and Arthur J Olson. Autodock4 and autodocktools4: Automated docking with selective receptor flexibility. *Journal of computational chemistry*, 30(16):2785–2791, 2009.

[56] Markus Muttenthaler, Glenn F. King, David J. Adams, and Paul F. Alewood. Trends in peptide drug discovery. *Nature Reviews Drug Discovery*, 20(4):309–325, 2021.

[57] Michael A. Nauck and David A. D'Alessio. Tirzepatide, a dual gip/glp-1 receptor co-agonist for the treatment of type 2 diabetes with unmatched effectiveness regrading glycaemic control and body weight reduction. *Cardiovascular Diabetology*, 21(1), 2022.

[58] Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

[59] Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024.

[60] Martin Pacesa, Lennart Nickel, Christian Schellhaas, Joseph Schmidt, and Pyatova et al. Bindcraft: one-shot design of functional protein binders. *bioRxiv*, 2024.

[61] Gustavo Paratcha, Fernanda Ledda, and Carlos F. Ibáñez. The neural cell adhesion molecule ncam is an alternative signaling receptor for gdnf family ligands. *Cell*, 113(7):867–879, 2003.

[62] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, and Mark et al. Veselov. Molecular sets (moses): A benchmarking platform for molecular generation models. *Frontiers in Pharmacology*, 11, 2020.

[63] RDKit, online. RDKit: Open-source cheminformatics.

[64] Jarrid Rector-Brooks, Mohsin Hasan, Zhangzhi Peng, Zachary Quinn, Chenghao Liu, Sarthak Mittal, Nouha Dziri, Michael Bronstein, Yoshua Bengio, Pranam Chatterjee, Alexander Tong, and Avishek Joey Bose. Steering masked discrete diffusion models via discrete denoising posterior prediction. *arXiv preprint arXiv:2410.08134*, 2024.

[65] Stephen A. Rettie, David Juergens, Victor Adebomi, Yensi Flores Bueso, Qinqin Zhao, Alexandria N. Leveille, Andi Liu, Asim K. Bera, Joana A. Wilms, and Alina et al. Üffing. Accurate de novo design of high-affinity protein binding macrocycles using deep learning. *bioRxiv*, 2024.

[66] Chiara Rigon, Alessandra Andrisani, Monica Forzan, Donato D'Antona, Alice Bruson, Erich Cosmi, Guido Ambrosini, Gian Mario Tiboni, and Maurizio Clementi. Association study of amh and amhrii polymorphisms with unexplained infertility. *Fertility and sterility*, 94(4):1244–1248, 2010.

[67] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010.

[68] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 2024.

[69] Schrödinger, LLC. The PyMOL molecular graphics system, version 1.8. November 2015.

[70] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in Neural Information Processing Systems*, 2024.

[71] Samradhi Singh, Namrata Pal, Swasti Shubham, Devojit Kumar Sarma, Vinod Verma, Francesco Marotta, and Manoj Kumar. Polycystic ovary syndrome: Etiology, current management, and future therapeutics. *Journal of Clinical Medicine*, 12(4):1454, 2023.

[72] Alexander A. Sosunov, Guy M. McKhann, and James E. Goldman. The origin of rosenthal fibers and their contributions to astrocyte pathology in alexander disease. *Acta Neuropathologica Communications*, 5(1), 2017.

[73] Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to dna sequence design. *International Conference on Machine Learning*, 2024.

[74] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[75] Vladimir N. Uversky, Christopher J. Oldfield, and A. Keith Dunker. Intrinsically disordered proteins in human diseases: Introducing the d2concept. *Annual Review of Biophysics*, 37(1): 215–246, 2008.

[76] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *International Conference on Learning Representations*, 2022.

[77] Alexander A. Vinogradov, Yizhen Yin, and Hiroaki Suga. Macrocyclic peptides as drug candidates: Recent progress and remaining challenges. *Journal of the American Chemical Society*, 141(10):4167–4181, 2019.

[78] Vanja Vukojevic, Pavlina Mastrandreas, Andreas Arnold, Fabian Peter, Iris-T Kolassa, Sarah Wilker, Thomas Elbert, Dominique J-F de Quervain, Andreas Papassotiropoulos, and Attila Stetak. Evolutionary conserved role of neural cell adhesion molecule-1 in memory. *Translational psychiatry*, 10(1):217, 2020.

[79] Lei Wang, Nanxi Wang, Wenping Zhang, Xurui Cheng, Zhibin Yan, Gang Shao, Xi Wang, Rui Wang, and Caiyun Fu. Therapeutic peptides: current applications and future directions. *Signal Transduction and Targeted Therapy*, 7(1), 2022.

[80] Shuzhe Wang, Jagna Witek, Gregory A Landrum, and Sereina Riniker. Improving conformer generation for small rings and macrocycles based on distance geometry and experimental torsional-angle preferences. *Journal of chemical information and modeling*, 60(4):2044–2058, 2020.

[81] Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. *International Conference on Machine Learning*, 2024.

[82] Zhongyuan Wang, Yue Zhao, Yan Jiang, Wei Lv, Lin Wu, Baoyan Wang, Lingyan Lv, Qunwei Xu, and Hongliang Xin. Enhanced anti-ischemic stroke of zl006 by t7-conjugated pegylated liposomes drug delivery system. *Scientific Reports*, 5(1), 2015.

[83] Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, and Lukas F. et al. Milles. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[84] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28 (1):31–36, 1988.

[85] Michael A. Wheeler, Iain C. Clark, Emily C. Tjon, Zhaorong Li, Stephanie E. J. Zandee, Charles P. Couturier, Brianna R. Watson, Giulia Scalisi, Sarah Alkwai, and Veit et al. Rothhammer. Mafg-driven astrocytes promote cns inflammation. *Nature*, 578(7796):593–599, 2020.

[86] Xiaopeng Xu, Chencheng Xu, Wenjia He, Lesong Wei, Haoyang Li, Juexiao Zhou, Ruochi Zhang, Yu Wang, Yuanpeng Xiong, and Xin Gao. Helm-gpt: de novo macrocyclic peptide design using generative pre-trained transformer. *Bioinformatics*, 40(6), 2024.

[87] Quan Yang, Jinyao Zhao, Dan Chen, and Yang Wang. E3 ubiquitin ligases: styles, structures and functions. *Molecular Biomedicine*, 2(1), 2021.

[88] Meinan Yu, Dunyan Su, Yuanyuan Yang, Lin Qin, Chuan Hu, Rui Liu, Yang Zhou, Chuanyao Yang, Xiaotong Yang, Guanlin Wang, and Huile Gao. D-t7 peptide-modified pegylated bilirubin nanoparticles loaded with cediranib and paclitaxel for antiangiogenesis and chemotherapy of glioma. *ACS Applied Materials & Interfaces*, 11(1):176–186, 2018.

[89] Seung Pil Yun, Tae-In Kam, Nikhil Panicker, SangMin Kim, Yumin Oh, Jong-Sung Park, Seung-Hwan Kwon, Yong Joo Park, Senthilkumar S. Karuppagounder, and Hyejin et al. Park. Block of a1 astrocyte conversion by microglia is neuroprotective in models of parkinson's disease. *Nature Medicine*, 24(7):931–938, 2018.

[90] Dingpeng Zhang, Jhoely Duque-Jimenez, Francesco Facchinetti, Garyk Brixi, Kaitlin Rhee, William W. Feng, Pasi A. Jänne, and Xin Zhou. Transferrin receptor targeting chimeras for membrane protein degradation. *Nature*, 2024.

[91] Ruochi Zhang, Haoran Wu, Yuting Xiu, Kewei Li, Ningning Chen, Yu Wang, Yan Wang, Xin Gao, and Fengfeng Zhou. Pepland: a large-scale pre-trained peptide representation model for a comprehensive landscape of both canonical and non-canonical amino acids. *arXiv preprint arXiv:2311.04419*, 2023.

[92] Tianhong Zhang, Hongli Li, Hualin Xi, Robert V. Stanton, and Sergio H. Rotstein. Helm: A hierarchical notation language for complex biomolecule structure representation. *Journal of Chemical Information and Modeling*, 52(10):2796–2806, 2012.

[93] Yue Zhao, Yan Jiang, Wei Lv, Zhongyuan Wang, Lingyan Lv, Baoyan Wang, Xin Liu, Yang Liu, Quanyin Hu, Wujin Sun, Qunwei Xu, Hongliang Xin, and Zhen Gu. Dual targeted nanocarrier for brain ischemic stroke treatment. *Journal of Controlled Release*, 233:64–71, 2016.

[94] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. 2024.

A Additional Case Studies

With our trained property classifiers, we conduct experiments for diverse, therapeutically relevant protein targets to evaluate our multi-objective MCTS guidance strategy. To demonstrate generalizability, we include targets with known peptide binders such as TfR, and proteins with no known binders, including GFAP, NCAM1, and AMHR2. We also design bi-specific binders for GFAP and RBX1 as a case study for Alexander disease therapeutics. These targets include both receptor proteins involved with active transport pathways as well as intracellular targets where cell membrane permeability is crucial to achieving therapeutic effects. For each target, we condition the generation on the binding affinity score given the target protein sequence along with solubility, hemolysis, non-fouling, and cell membrane permeability for intracellular targets. For external testing and validation, we use Autodock Vina [18] to compute *in silico* binding affinities of our generated binders (See Methods B.6).

Targeting Receptors on the Blood-Brain Barrier. The Transferrin receptor (TfR) is a receptor protein abundant on the selectively permeable blood-brain barrier (BBB) that is responsible for transporting iron-binding transferrin (Tf) proteins into the brain parenchyma [34]. Given its selective expression on brain endothelial cells and glioma cells and its ability to recycle back to the luminal surface after facilitating the internalization of cargo through the BBB [33], TfR has been extensively studied as a target for the intravenous delivery of various therapeutics and therapeutic nanocarriers through the BBB [23, 90].

To generate relevant binders for TfR, we condition PepTune on binding affinity with the TfR sequence, in addition to solubility, hemolysis, and non-fouling. At each iteration, we measured the mean of the properties scores across all rolled-out sequences from the selected node to evaluate the effectiveness of the optimization strategy. We show that all properties, except solubility, exhibited an upward trend over iterations, with the average score of the binding affinity classifier exhibiting a significant increase in score to over 9.0 (Figure 4B). After plotting the distribution of 100 peptides generated from a single run of PepTune with the minimum number of sequences set to 100, we confirm that our multi-objective MCTS algorithm shifted the distribution to a higher predicted binding affinity than the unconditionally generated peptides (PepMDLM) and the data used to train the binding regression model (Figure 4A). Despite being conditioned on four distinct properties, PepTune is capable of generating higher affinity binders than the unconditional model, supporting the effectiveness of our multi-objective guidance strategy.

Encouraged by these results, we sampled the Pareto-optimal sequences from the generated peptides and used Vina docking to compute their optimized docking score. Notably, we observed that all of the generated binders that were selected for docking produced affinity scores below -6.0 kcal/mol, with our top-performing binder achieving a -8.4 kcal/mol binding affinity (Figure 4C). From the docking scores, we took the two binders with the best docking scores and visualized their binding conformation with TfR, showing that they bind to distinct motifs on the protein surface (Figure 4B, F, G).

To further confirm binding affinity to TfR, we compared our peptides to the well-established 7-amino acid peptide T7 (sequence: HAIYPRH) that selectively binds to an alternative site as compared to endogenous Tf on TfR [41]. T7 has been extensively explored for targeted delivery of nanoparticles to the brain [39, 36, 5, 8, 82, 93, 47], and has demonstrated 7.89-fold enhanced brain penetration in *in vivo* mice models [88]. After docking T7 with TfR, we obtained a docking score of -8.4 kcal/mol. Notably, our peptides optimized on all four therapeutic properties including TfR binding affinity show competitive docking scores to T7 (Figure 5A, C, E), suggesting that PepTune is capable of generating promising candidates for *in vivo* targeting and delivery across the BBB. Furthermore, after annotating polar contacts within 3.5 Å we determine that both of the generated peptides with the best binding affinity scores have shared residue contacts when binding with TfR as T7 (Figure 5B, D, F), indicating that our generated peptides have similar binding properties to T7, enabling it to bind strongly to a shared binding site. Furthermore, our generated binders have diverse structural features, such as cycles in binder 1 and side-chain modifications in binder 2. Since T7 is known to bind to an alternative site than endogenous Tf [41], we show that PepTune can generate viable candidates for non-competitive binding to TfR for BBB-targeting applications.

Targeting Intracellular Proteins. Glial fibrillary acidic protein (GFAP) is an intracellular protein differentially expressed in astrocytes, a family of glial cells in the brain [31]. Dysregulation of GFAP expression has been found to cause Rosenthal fibers, astrocytic cytoplasmic inclusions that

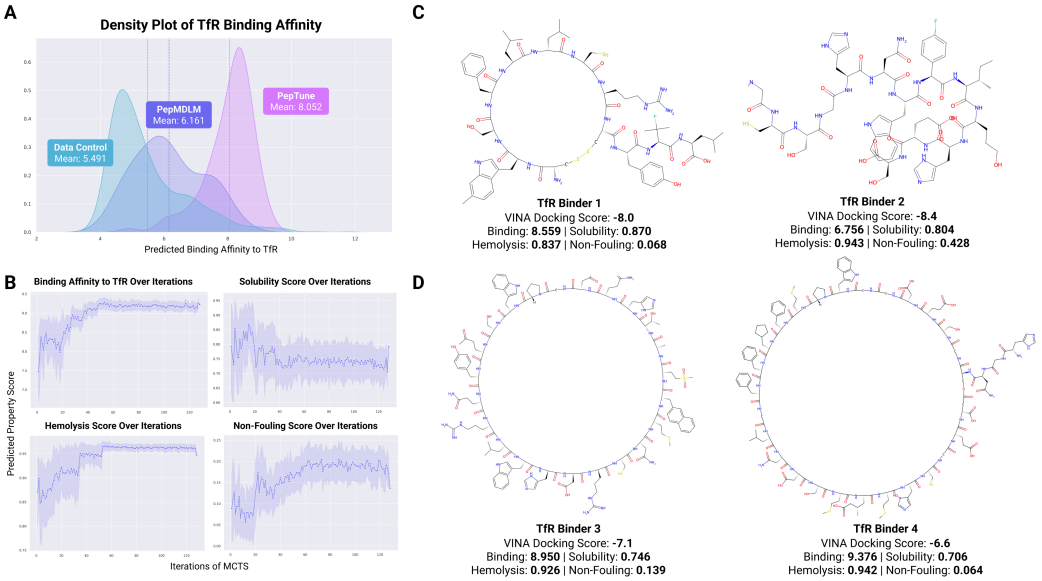


Figure 4: **PepTune-generated peptide binders to TfR.** (**A**) Density plot depicting the frequency of predicted binding affinity scores from our trained regression model for the sequences in the data used to train the regression model, the generated peptides from our unconditional PepMDLM model, and our PepTune model conditioned on TfR binding affinity, solubility, hemolysis, and non-fouling. (**B**) Plots depicting the mean scores for each property over the number of iterations or traversals of the MCTS algorithm for 128 iterations and a maximum token length of 200. The shaded region represents the standard deviation. (**C**) Two-dimensional visualization of generated binders with token length 100, their corresponding docking scores (↓) computed using Vina docking, and predicted classifier scores (↑) from the trained classifiers. (**D**) Visualizations of generated binders with token length 200, their docking scores, and predicted classifier scores.

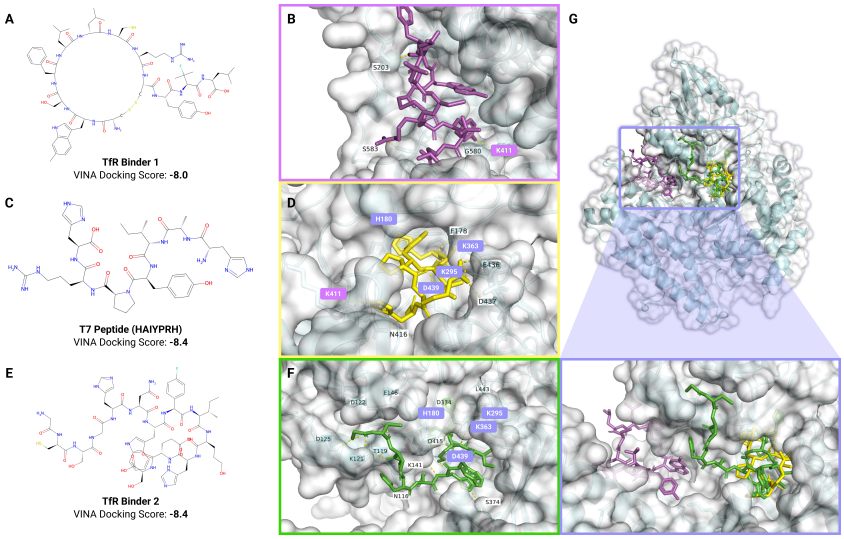


Figure 5: **Comparison of PepTune-generated peptides and established T7-peptide to TfR.** Two-dimensional chemical structure of (**A**) PepTune-generated binder 1, (**C**) established T7 peptide, and (**E**) PepTune-generated TfR binder 2 and their Vina docking scores to TfR (↓). Zoomed-in visualization of the docked binding positions of (**A**) binder 1, (**B**) T7, and (**C**) binder 2 with TfR. Polar contacts within 3.5 Å are annotated, and shared contacts between T7 and binder 1 (purple) and between T7 and binder 2 (blue) are highlighted. (**C**) Overlay of peptide binders on full TfR protein

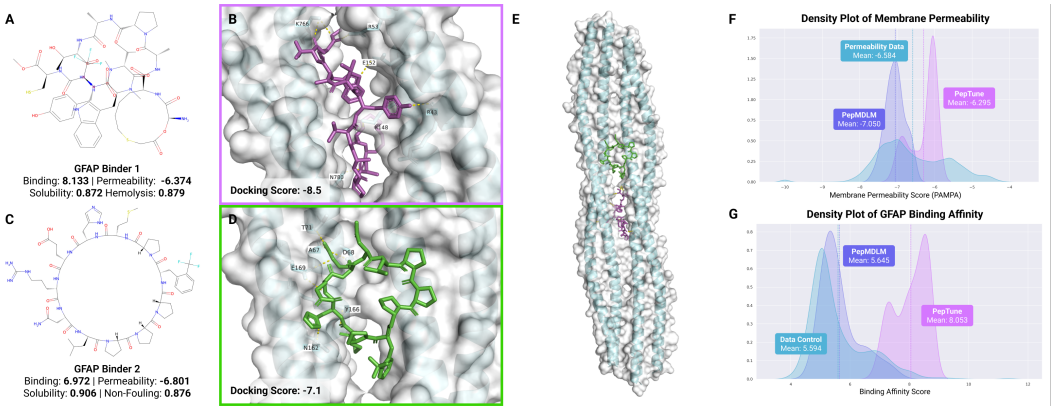


Figure 6: **PepTune-generated peptide binders to intracellular protein GFAP.** (**A**, **C**) Two-dimensional structures of GFAP binder 1 and 2 with predicted property scores, including cell membrane permeability. (**B**, **D**) GFAP binders 1 and 2 docked to GFAP with score of -8.5 kcal/mol and -7.1 kcal/mol respectively. (**E**) Full GFAP protein structure with docked binders 1 and 2. (**F**) The distribution of PAMPA membrane permeability scores from 34,853 experimentally-validated peptides compared to 100 peptides generated using our unconditional PepMDLM model, and 100 peptides generated with PepTune conditioned on both cell membrane permeability and affinity to GFAP. The permeability curve shifted towards higher permeability with a mean of -6.295. **(G)** Simultaneously, the distribution of predicted binding affinity scores to GFAP for the PepTune-generated peptides is shifted to higher scores with a mean of 8.053 compared to a set of experimentally-tested peptides and unconditional PepMDLM-generated peptides.

are responsible for Alexander disease, a fatal neurodegenerative disease affecting infants [6, 25]. Discovering potent binders that inhibit or degrade GFAP proteins can have significant therapeutic implications. However, no established peptide binders exist to GFAP, which motivates their *de novo* design. In addition to achieving high binding affinity with GFAP, we posit that an optimal peptide binder must also cross the astrocyte cell membrane into the cytosol to access GFAP. Therefore, we condition the generation of GFAP binders on five properties: binding affinity to GFAP, solubility, hemolysis, non-fouling, and cell membrane permeability using our permeability regression model, demonstrating optimization across all of these properties (Figure 6). To confirm GFAP engagement, our docking peptides demonstrate strong affinities below -7 kcal/mol, motivating downstream experimental validation in astrocyte cultures (Figure 6B and D).

Targeting Extracellular Proteins Without Existing Binders. To test the ability of our model to generate binders to challenging extracellular targets without existing binders, we evaluate PepTune-generated peptides for NCAM1 and AMHR2, two therapeutically relevant receptor proteins. Neural cell adhesion molecule 1 (NCAM1), is a transmembrane protein expressed on the surface of neurons and glial cells [61]. Beyond its roles in neuronal migration and synaptogenesis, NCAM1 is also crucial for memory formation, highlighting its significance in brain development [78]. As NCAM1 is an extracellular protein, we generated a library of peptides with PepTune-optimized NCAM1 binding affinity, solubility, hemolysis, and non-fouling (Figure 7F, G). All properties exhibited an upward trend across optimization iterations.

We selected two binders with the highest Vina docking scores for visualization (Figure 7A-E). Notably, *in silico* docking analysis revealed that binder 1 exhibits markedly high affinity binding (-8.6 kcal/mol) while binder 2 wraps around the NCAM1 structure via numerous polar contacts, suggesting extensive and specific interactions (Figure 7B and D).

Table 2: PepTune-generated dual-target binders to TfR and GLAST.

Binder ID	**TfR Docking Score** (kcal/mol) (↓)	**GLAST Docking Score** (kcal/mol) (↓)	**Solubility** (↑)	**Hemolysis** (↑)	**Non-fouling** (↑)
Binder 1	-8.8 (8.800)	-8.9 (7.775)	**0.975**	0.743	0.118
Binder 2	-8.0 (7.599)	-7.9 (6.751)	0.938	0.835	**0.309**
Binder 3	-8.3 (7.537)	-8.2 (6.662)	0.972	**0.914**	0.214
Binder 4	-7.6 (7.748)	-7.5 (6.946)	0.959	0.902	0.290
Binder 5	**-10.5** (8.714)	-8.5 (7.398)	0.811	0.748	0.202
Binder 6	-8.4 (8.197)	-7.5 (7.076)	0.971	0.855	0.165
Binder 7	-9.3 (8.321)	**-9.2** (7.190)	0.881	0.860	0.212

[1] The predicted binding affinity scores by our trained classifier are placed in brackets beside the docking score. Larger scores indicate stronger binding for our classifier.

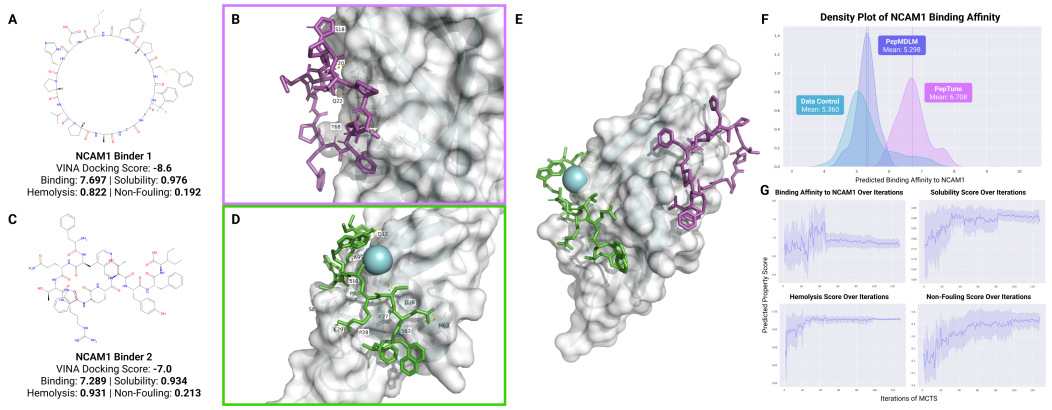


Figure 7: **PepTune-generated peptide binders to NCAM1.** Two-dimensional structures of (**A**) binder 1 and (**C**) binder 2 genered with PepTune. Docking positions of (**B**) binder 1 and (**C**) binder 2 on NCAM1 with annotated polar contacts within 3.5 Å(**G**) Full NCAM1 protein structure with docked peptide binders 1 and 2. (**H**) (Top) Density plot of NCAM1 binding affinity scores for PepTune (mean: 6.708), PepMDLM (mean: 5.298), and peptides from a control set of experimentally-tested peptide SMILES (mean: 5.360). (Bottom) Plots depicting the average predicted score for NCAM1 binding affinity, solubility, hemolysis, and non-fouling over iterations of MCTS.

Anti-Müllerian hormone type-2 receptor (AMHR2) is a transmembrane receptor involved in sex differentiation. Mutations in the AMHR2 gene are a leading cause of Persistent Müllerian duct syndrome (PMDS) in males, resulting in the retention of female gonads alongside male reproductive structures [32]. In females, polymorphisms of AMHR2 have been associated with infertility [66, 40]. Most interestingly, antagonism of AMHR2 with therapeutic peptides can potentially serve as a specific therapy for polycystic ovarian syndrome (PCOS), which affects an estimated 4% to 10% of women globally [71], as AMHR2 signaling has been implicated in follicular arrest and dysregulated ovarian function [16].

Following similar computational set-ups as described previously, we generated *in silico* binders with high Vina predicted binding affinities (<-6 kcal/mol), despite observing a decrease in the predicted solubility along iterations (Figure 8). However, our observation of reduced solubility upon binder docking can be attributed to the presence of hydrophobic patches within the AMHR2 extracellular domain, particularly near the binding site to its ligand AMH [29]. This phenomenon highlights the importance of balancing solubility and binding affinity in binder development. With further optimization of their therapeutic properties, we hope to demonstrate the potential of these binders for applications in fertility treatment in the future.

The examples above demonstrate the versatility of our method, which can be effectively applied to discover peptide binders for single target proteins lacking known ligands, thereby unlocking their therapeutic potential.

Dual-Targeting of TfR and GLAST for Drug Delivery to Astrocytes. As described in the main text, we used PepTune to generate peptides with high binding affinity to TfR and GLAST (Results 2.4). Here, we provide the supplementary details for this case study (Figure 9, Table 2).

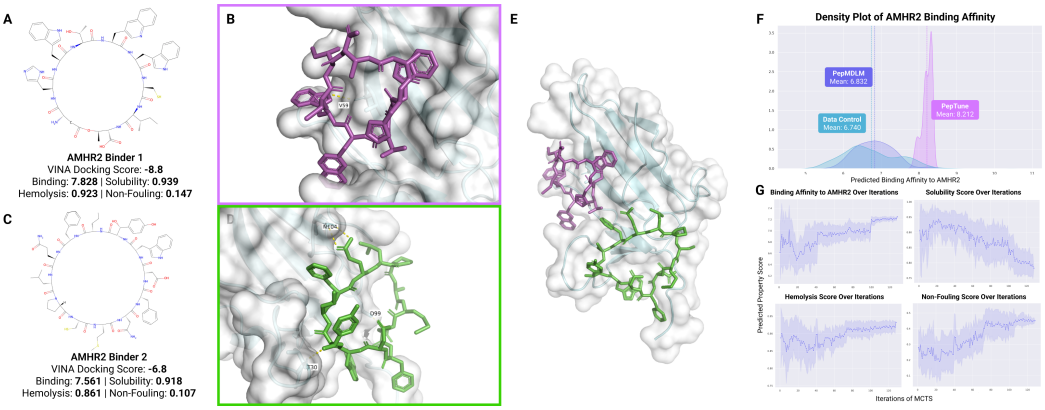


Figure 8: **PepTune-generated peptides to AMHR2.** Two-dimensional structures of (**A**) binder 1 and (**B**) binder 2 generated with PepTune. Docking positions of (**A**) binder 1 and (**B**) binder 2 on NCAM1 with annotated polar contacts. (**G**) Full AMHR2 protein structure with docked peptide binders 1 and 2. (**H**) (Top) Density plot of AMHR2 binding affinity scores for PepTune (mean: 8.212), PepMDLM (mean: 6.832), and peptides from a control set of experimentally-tested peptide SMILES (mean: 6.740). (Bottom) Plots depicting the average predicted score for AMHR2 binding affinity, solubility, hemolysis, and non-fouling over iterations of MCTS.

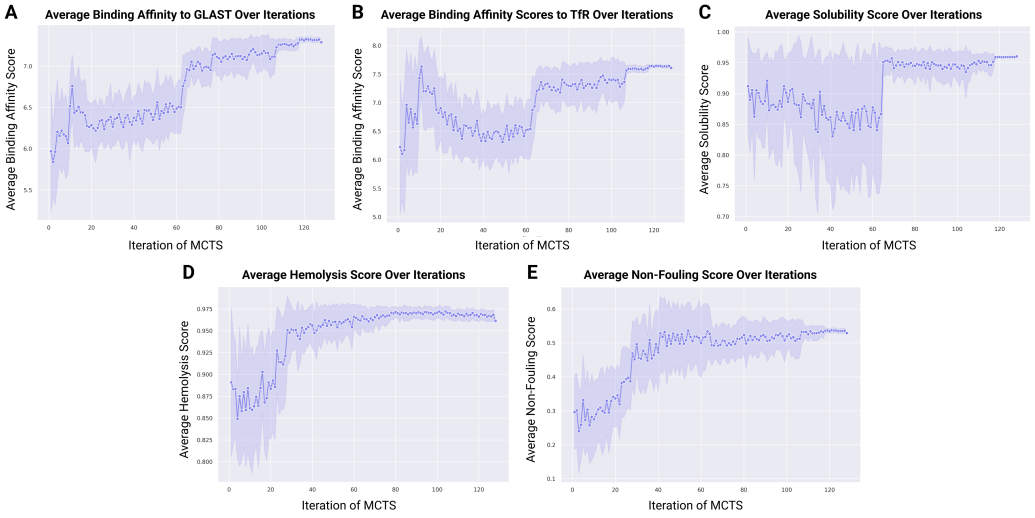


Figure 9: **Property Scores Over Iteration for Dual-Target Conditioning on TfR and GLAST.** (**A**) Plot of average predicted binding affinity score to GLAST over iterations. (**B**) Plot of average predicted binding affinity score to TfR over iterations. (**C**, **D**, **E**) Plot of average predicted solubility, hemolysis, and non-fouling scores over iterations.

Table 3: PepTune-generated dual-target binders to GFAP and RBX1.

Binder ID	**GFAP Docking Score** (kcal/mol) (↓)	**E3 Docking Score** (kcal/mol) (↓)	**Solubility** (↑)	**Hemolysis** (↑)	**Non-fouling** (↑)
Binder 1	-8.0 (8.384)	-8.4 (7.468)	0.730	**0.894**	0.111
Binder 2	-8.3 (7.395)	**-9.3** (7.089)	**0.972**	0.869	0.134
Binder 3	-7.3 (7.925)	-8.7 (7.158)	0.935	0.812	0.143
Binder 4	**-8.8** (7.144)	-8.7 (7.000)	0.897	0.807	**0.158**

[1] The predicted binding affinity scores by our trained classifier are placed in brackets beside the docking score. Larger scores indicate stronger binding for our classifier.

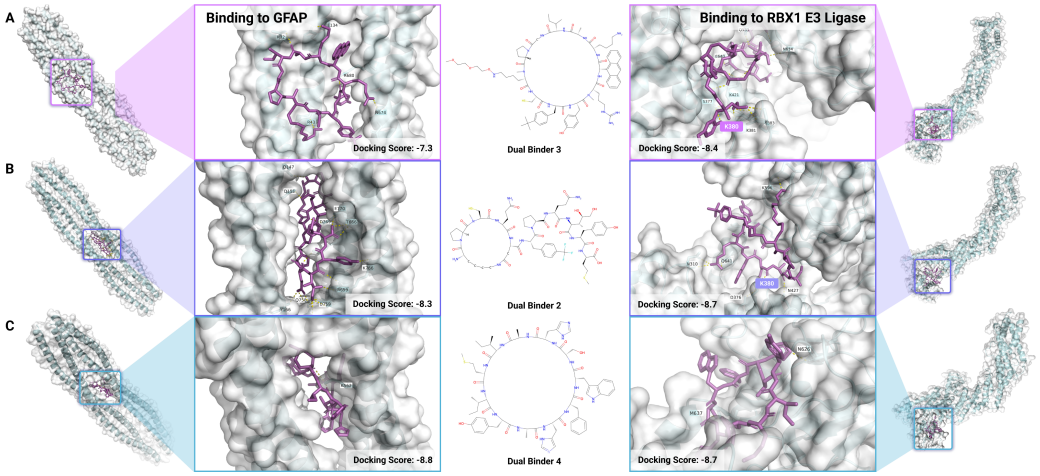


Figure 10: **PepTune-generated peptides with dual GFAP and RBX1 affinity.** Full protein binding location and close-up binding position for (**A**) dual binder 3, (**B**) dual binder 2, and (**C**) dual binder 4 with GFAP (left) and RBX1 (right). Polar contacts within 3.5 Å are annotated and shared polar contacts between binders are highlighted.

Dual-Targeting of GFAP and an E3 Ubiquitin Ligase for Target Protein Degradation. As another dual-target case study, we used PepTune to generate peptides with high binding affinity to GFAP protein and an E3 ubiquitin ligase protein RBX1, a protein in the Skp1/Cullin-1/F-box (SCF) E3 ubiquitin ligase complex that recruits E2 to catalyze ubiquitination and subsequent degradation [87]. A peptide generated for this task would be capable of binding to GFAP proteins overexpressed in Alexander disease and mediate their proteasomal degradation, which could alleviate the production of disease-causing Rosenthal fibers in astrocytes [72]. After conditioning PepTune generation on binding affinity to GFAP, binding affinity to RBX1, solubility, hemolysis, and non-fouling (Table 3), we selected three non-dominated binders with predicted affinities greater than 7.0 for docking experiments. For these Pareto-optimal peptides, we indeed observed strong binding affinities for both GFAP and RBX1 post-docking, indicating their unique potential for multi-target interaction (Figure 10). GFAP is an intermediate filament protein [19] and thus forms a unique rod-like structure with a head domain and a tail domain. The docking positions of all three candidates were along the rod domain, binding in the gap between adjacent rods in the filament. Contrarily, docked candidates to RBX1 consistently bound close to its interaction site of Cullin, rather than at the Skp2 F-box adaptor site (Figure 10), indicating that further motif conditioning, as done with recent peptide design language models [13], would benefit PepTune's clinical potential.

B Methods

B.1 Data Curation and Tokenization.

MDLM Training Data. To train the unconditional masked diffusion language model generator, we collected 11 million peptide SMILES consisting of 7451 sequences from the CycPeptMPDB database [42], 825,632 unique peptides from SmProt [46], and approximately 10 million modified peptides generated from CycloPs [17, 20], which consists of 90% canonical amino acids, 10% unnatural amino acids from SwissSidechain [22], 10% dextro-chiral alpha carbons, 20% N-methylated amine backbone

atoms, and 10% PEGylated peptides. All possible cyclization conformations were attempted on the peptides generated with CycloPs. We used SELFIES [38] to check the integrity of the SMILES sequences.

We split our data by k-means clustering into 1000 groups of sequences with similar chemical properties based on their Morgan fingerprint [67], which is a bit-vector representation of the full peptide sequence where each bit encodes a feature relating to the SMILES atom types, connectivity, and bonding environment. The final dataset was a 0.8 to 0.2 split based on the clusters, maintaining similar diversities of the SMILES strings. Since the degree of masking is evenly spread between $t = 0$ to $t = 1$ within each training batch, grouping similar SMILES in the same batch ensures the model learns to reconstruct a diverse set of peptide SMILES from various degrees of masking.

Dynamic Batching. We applied dynamic batching to handle variable-length token sequences and increase computational efficiency. Inspired by ESM-2's dynamic batching technique [49], input SMILES are sorted by length to maximize the utility of GPU memory. The maximum tokens per GPU was set to 16,000.

SMILES Tokenization. To enable the novel generation of non-natural amino acids containing cyclizations and diverse backbone and side-chain modifications, we trained our generative diffusion model on Simplified Molecular-Input Line-Entry System (SMILES) [84] representations of peptides. We experimented with several tokenization schemes that capture common motifs in the training data to enhance the generation of valid peptide SMILES. We find that the SMILES Pair Encoding (SPE) tokenization scheme [46] with the PeptideCLM [20] vocabulary of 581 SMILES tokens and 5 special tokens with an average length of four characters per token, demonstrated superior performance, generating precise but valid peptides (Appendix F.2).

Classifier Data and Training. We trained our membrane permeability XGBoost regression model using experimentally-validated peptide SMILES consisting of 22,040 SMILES sequences obtained from the ChEMBL database [54] and 7451 sequences from the CycPeptMPDB database [42].

We collected binding affinity, solubility, hemolysis, and non-fouling data for classifier training from the PepLand and PeptideBERT datasets [91, 27]. The binding affinity data was split within groups of peptides with weak, medium, and strong binding scores with a 0.8/0.2 ratio. The classifier was trained with cross-attention between ESM-2 protein embeddings and PeptideCLM SMILES representations [20]. The hyperparameters were chosen with 50 trials of OPTUNA search [1]. For other classifiers, data were randomly shuffled and split into 0.8/0.1/0.1 ratio for train, validation, and test. XGBoost classifiers [11, 24] were applied on PeptideCLM embeddings with 50 trials of OPTUNA search for the optimal boosted tree hyperparameters.

B.2 Unconditional Masked Discrete Diffusion Model

Notation. Let $\mathbf{x}_0 \in \{0,1\}^{|\mathcal{V}|}$ represent the one-hot vector of a token in a sequence in the training data and $\mathbf{x}_\theta(\mathbf{z}_t, t) \in \Delta^{|\mathcal{V}|}$ be the vector of predicted token probabilities across the vocabulary $\mathcal{V}$ given the current state $\mathbf{z}_t$ at time t. In most contexts, $\mathbf{x}_0$ will be used to denote a single token, but when discussing the full sequence, $\mathbf{x}_0^{(\ell)}$ is used to denote the token at position ℓ in the sequence. Let T denote the total number of time steps in the discrete forward and reverse diffusion processes. In the context of all time steps, we expand t to $t(n) \in (0, 1]$ when denoting a single time step in the forward and backward diffusion process with $n \in [1 \dots T-1]$. Let $s(n) = t(n) - \frac{1}{T}$ denote the previous time step in the forward process. Then, let $\mathbf{z}_{t(n)}$ and $\mathbf{z}_{s(n)}$ denote the state of a specific token at time $t(n)$ and $s(n)$ in the diffusion process, respectively. Let $\alpha_t(\mathbf{x}_0) : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}$ denote a function that takes the unmasked token $\mathbf{x}_0$ and outputs a value in [0,1] representing the probability of remaining unmasked at time t in the forward diffusion process. Let $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$ denote a vector with ones at indices of peptide bond tokens and zeroes at all remaining indices and let $\mathbf{x}_0 = \mathbf{b}$ indicate that $\mathbf{x}_0$ is a peptide-bond token.

Bond-Dependent Masking Schedule. Since all peptides follow a distinct SMILES structure consisting of un-modified or modified peptide bonds before and after each central carbon atom with an amino acid side chain, we hypothesized that applying bond-dependent masking and unmasking schedules would allow the reverse diffusion process to learn to unmask the crucial structural components of a

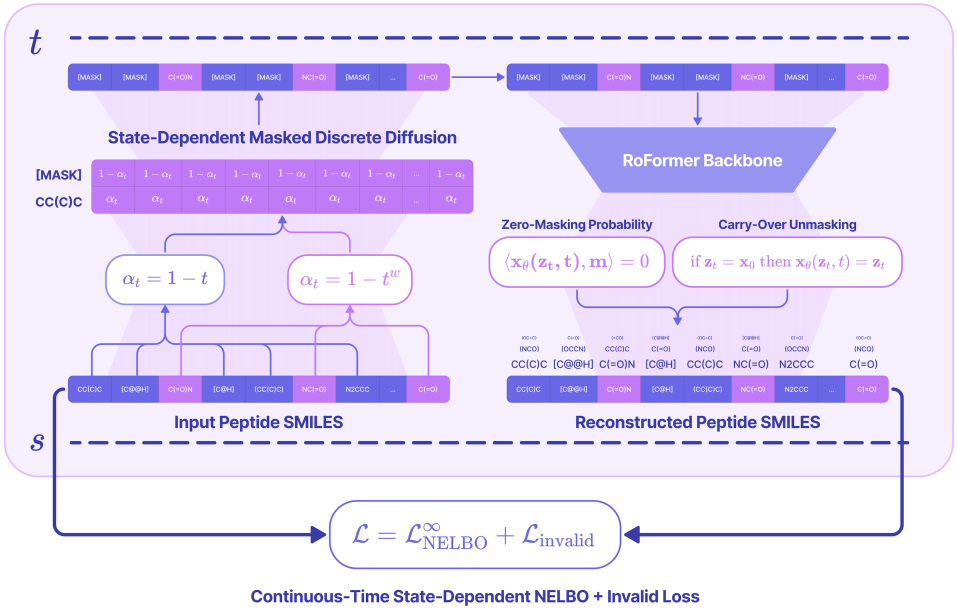


Figure 11: **PepMDLM**. PepMDLM is a discrete masked diffusion model for unconditional *de novo* generation of peptide SMILES representations.

peptide SMILES that are common across all peptides before filling in the segments in-between with diverse amino acid side-chains.

Extending previous work in state-dependent masking [70], we devised a masking schedule where the probability of masking a token within a peptide bond increases at a slower rate in earlier times t in the masking process compared to non-peptide bond tokens. To achieve this, we define the discrete-time log-linear masking schedule $\sigma(t) = -\log(1-t)$ for non-peptide bond tokens and the log-polynomial masking schedule $\sigma(t) = -\log(1-t^w)$ for peptide-bond tokens. We show in Appendix D.1 that the continuous-time probability of remaining unmasked at time t in the forward diffusion process is given by the function $\alpha_t(\mathbf{x}_0) : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}$ that takes the vector encoding the token $\mathbf{x}_0$ and returns a probability.

$$\alpha_t(\mathbf{x}_0) = \begin{cases} 1 - t^w & \mathbf{x}_0 = \mathbf{b} \\ 1 - t & \mathbf{x}_0 \neq \mathbf{b} \end{cases} \tag{2}$$

where $\mathbf{b}$ is the vector with ones at indices of peptide bond tokens and zeroes in remaining indices, including tokens within modified peptide-bonds identified with our BONDMASK function (Appendix 7). Since the probability of transitioning to a [MASK] token at time t is given by $1 - \alpha_t(\mathbf{x}_0)$, there is a lower probability t^w for $t \in (0, 1]$ of masking a peptide bond token than the probability t of masking a non-peptide bond token, especially in earlier time steps for smaller t (Fig. 15A). As $t \to 1$, $\alpha_t(\mathbf{x}_0) \to 0$ and the probability of masking for both peptide and non-peptide bond tokens approaches 1, ensuring that the model can learn to reconstruct the full token sequence during the reverse diffusion process.

With our bond-dependent masking rate $\alpha_t(\mathbf{x}_0)$, we define the forward transition matrix as

$$q(\mathbf{z}_t|\mathbf{x}_0) = \text{Cat}(\mathbf{z}_t; \alpha_t(\mathbf{x}_0)\mathbf{x}_0 + (1 - \alpha_t(\mathbf{x}_0))\mathbf{m}) \tag{3}$$

The reverse transition from state $t \to s$ given the bond-dependent forward masking schedule is derived in Appendix D.2 as

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0) = \begin{cases} \left\langle \left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}, \mathbf{x}_0 \right\rangle \mathbf{x}_0 + \left\langle \left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}, \mathbf{x}_0 \right\rangle \mathbf{m} & \mathbf{z}_t = \mathbf{m} \\ \mathbf{z}_t & \mathbf{z}_t \neq \mathbf{m} \end{cases} \tag{4}$$

To estimate the reverse posterior, we define a parameterized RoFormer model $\mathbf{x}_\theta(\mathbf{z}_t, t) : \mathcal{V}^L \times [0,1] \to \Delta^{|\mathcal{V}|}$ that takes the partially masked sequence at time t and predicts a vector of token probabilities over the $|\mathcal{V}|$-dimensional simplex for each position in the sequence. By substituting $\mathbf{x}_0 = \mathbf{x}_\theta(\mathbf{z}_t, t)$ into the true reverse transition, we get the predicted reverse transition distribution.

$$p_\theta(\mathbf{z}_s|\mathbf{z}_t) = \begin{cases} \left\langle \left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}, \mathbf{x}_\theta(\mathbf{z}_t, t)\right\rangle \mathbf{z}_s + \left\langle \left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}, \mathbf{x}_\theta(\mathbf{z}_t, t)\right\rangle \mathbf{m} & \mathbf{z}_t = \mathbf{m} \\ \mathbf{z}_t & \mathbf{z}_t \neq \mathbf{m} \end{cases} \tag{5}$$

For larger w, peptide bonds are masked at later timesteps, encouraging earlier unmasking in the reverse diffusion process. However, setting w too large can result in the model over-fitting to the dataset [70]. Empirically, we found that $w = 3$ increased peptide validity while maintaining diversity across generated samples.

SUBS Parametrization. Following Sahoo et al. [68], we parameterize the reverse diffusion model using SUBS parametrization, which enforces zero-masking probability and carry-over unmasking. This strategy enforces the constraints applied in the forward diffusion process and has been shown to minimize perplexity [68].

1. **Zero Masking Probability**. The forward process operates under the assumption that a token can only be masked once across times $t = 0 \to 1$. It follows that the probability of a sequence being masked in the reverse diffusion process is zero. To enforce this property, we explicitly set the log probability of the sequence being masked in the reverse transition distribution to zero such that if a token is unmasked at time t, it remains unmasked for all timesteps from $t \to 0$.

$$\langle \mathbf{x}_\theta(\mathbf{z_t}, \mathbf{t}), \mathbf{m}\rangle = 0 \tag{6}$$

2. **Carry-Over Unmasking**. For each transition in the forward pass, all tokens either remain unchanged or are masked. Therefore, in the reverse process, when a token is unmasked at time t, the unmasked token is copied over all time steps from $t \to 0$. We enforce this by setting the logits vector equal to the one-hot encoding for the unmasked token.

$$\mathbf{x}_\theta(\mathbf{z}_t, t) = \mathbf{z}_t \;\text{ if }\; \mathbf{z}_t \neq \mathbf{m} \tag{7}$$

Since slight modifications to the peptide sequence can result in significant alterations to its properties [30], SUBS parametrization allows us to backpropagate rewards from unmasked sequences to earlier unmasking steps that accurately reflect the favorability of the particular unmasking step in generating optimal peptide SMILES sequences.

RoFormer Backbone. The backbone model used to generate the predicted probabilities $\mathbf{x}_\theta(\mathbf{z}_t, t)$ of transitioning from a masked state to any token is predicted by a backbone RoFormer architecture (See Appendix E.1). RoFormer leverages rotary positional embeddings (RoPE) [74], which effectively captures the relative inter-token interactions in peptide SMILES, especially for cyclic peptides.

B.3 Loss Functions

Bond-Dependent Continuous-Time Diffusion Loss. To optimize the parameters θ of the reverse diffusion model, we maximize the evidence lower bound (ELBO) of the distribution $\log p(\mathbf{x}_0)$, which is the log-probability distribution of generating the peptide sequences $\mathbf{x}_0$ present in the dataset. Therefore, we define our loss function as the negative ELBO (NELBO) given by

$$\begin{aligned}\mathcal{L}_{\text{NELBO}} = &\underbrace{\mathbb{E}_{q(\mathbf{z}_{t(1)}|\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)})\right]}_{\text{reconstruction loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_{t(T)},\mathbf{z}_{s(T)}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\right]}_{\text{prior loss}} \\ &+ \underbrace{\sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{s(n)},\mathbf{z}_{t(n)},\mathbf{z}_{t(n+1)}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\right]}_{\text{diffusion loss}}\end{aligned} \tag{8}$$

Training on samples masked for continuous values of $t \sim \text{Uniform}(0,1)$ yields a tighter lower bound compared to discrete values of t [37]. When the predicted probability distribution $\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t)$ is

exactly the one-hot encoding vector $\mathbf{x}_0^{(\ell)}$ for each token at position ℓ in the true sequence, the loss reduces to 0, which supports our objective.

By our derivation in Appendix D.3, our continuous-time bond-dependent NELBO separates into the sum of the negative log-losses (NLLs) for all non-peptide bond tokens that follow a log-linear masking schedule and the sum of the NLLs for all peptide bond tokens that follow a log-polynomial schedule.

$$\mathcal{L}_{\text{NELBO}}^{\infty} = \mathbb{E}_{t\sim\mathcal{U}(0,1)}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)}\Bigg[-\sum_{\ell:\mathbf{x}_0^{(\ell)}=\mathbf{b}} \frac{w}{t}\log\langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\rangle - \sum_{\ell:\mathbf{x}_0^{(\ell)}\neq\mathbf{b}} \frac{1}{t}\log\langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\rangle\Bigg] \tag{9}$$

Since the NLL term is minimized when the predicted probability of the ground truth token is close to 1, we show that applying the log-polynomial masking schedule for an exponent $w > 1$ scales the diffusion loss NELBO by a factor of w from the log-linear schedule. However, for earlier timesteps as $t \to 0$, both NLL weights increase to ∞, ensuring high precision in the final unmasking steps.

Given that peptide bonds form the fundamental backbone structure of a peptide, our bond-dependent masking strategy for peptide bonds acts as a peptide bond loss that introduces a higher penalty when the token predictions at positions of peptide bonds are inconsistent from the ground truth tokens during training, forcing the model to learn the specific structure of peptide SMILES strings in a vast space of SMILES strings that are not valid peptides.

Invalid Peptide Loss. To further discourage the generation from predicting token logits that produce invalid peptide SMILES, we incorporate a loss to penalize sampling of invalid peptide SMILES during training by taking the `argmax` of the predicted logits and assigning a penalty based on our peptide validity filter (Appendix 8). Given the peptide sequence $\tilde{\mathbf{x}}^{(\ell)} \in \{0,1\}^K$ generated from the `argmax` tokens with the highest probability from the predicted logits $\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)$, we minimize a penalty determined by our validity filter $\mathbf{1}[\tilde{\mathbf{x}}_0 \text{ is Invalid}]$ which returns 0 when the sequence is a valid peptide SMILES and 1 when the sequence either is not a valid SMILES or does not correspond to a peptide sequence. Since the `argmax` function is not differentiable, we use the `softmax` probability of the sampled tokens to scale the penalty score for each token which acts as a scalar multiplier in the loss function.

$$\begin{aligned}\mathcal{L}_{\text{invalid}} &= \sum_{\ell=1}^{L} \tilde{\mathbf{x}}_0^{(\ell)\top}\text{SM}\big(\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\big)\cdot \mathbf{1}[\tilde{\mathbf{x}}_0 \text{ is Invalid}] \\ &= \sum_{\ell=1}^{L} \frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{K}\exp(x_{\theta,j}^{(\ell)})}\cdot \mathbf{1}[\tilde{\mathbf{x}}_0 \text{ is Invalid}]\end{aligned} \tag{10}$$

where $k = \arg\max_j(\mathbf{x}_\theta^{(\ell)}\big(\mathbf{z}_t,t)\big)$ is the token with the highest predicted probability at position ℓ of the sequence.

Differentiating the invalid loss with respect to the probability vector $\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)$ for position ℓ, we derive the gradient with respect to the predicted probability of the sampled token $j = k$ and all other tokens in the vocabulary $j \neq k$ in Appendix D.1 as

$$\nabla\mathcal{L}_{\text{invalid}} = \begin{cases}\text{SM}(x_{\theta,k}^{(\ell)})\left(1-\text{SM}(x_{\theta.k}^{(\ell)})\right) & j = k \\ -\text{SM}(x_{\theta,j}^{(\ell)})\text{SM}(x_{\theta,k}^{(\ell)}) & j \neq k\end{cases} \tag{11}$$

Minimizing this objective function updates the parameters to lower the predicted probabilities for tokens that result in invalid peptide SMILES and increase the probabilities of the remaining tokens proportional to their original distribution such that the relative probability distribution of all other tokens $j \neq k$ is maintained.

Training. To train the MDLM to accurately approximate the true reverse transition distribution $q(\mathbf{z}_s|\mathbf{z}_t,\mathbf{x}_0)$ of a training sample $\mathbf{x}_0$ for all continuous timesteps $t = 1 \to 0$, we train a parameterized model $\mathbf{x}_\theta(\mathbf{z}_t,t)$ that takes the current sequence $\mathbf{z}_t$ and returns a $|\mathcal{V}|$-dimensional vector of predicted probabilities of transitioning to each token at time $s < t$ (Algorithm 1). For each dynamic training batch B, we randomly sample $|B|$ values $t \in \text{Uniform}(0,1)$ and off-set each time t by

Table 4: Training and validation loss after convergence on 11 million peptide SMILES with bond-dependent masking and invalid loss.

Model	**Train Loss** (↑)	**Train PPL** (↓)	**Val Loss** (↑)	**Val PPL** (↓)
PepMDLM	0.832	2.460	0.880	2.277

Table 5: Benchmark of PepMDLM unconditional model against HELM-GPT.

Model	**Validity** (↑)	**Uniqueness** (↑)	**Diversity** (↑)	**SNN** (↓)
HELM-GPT	**0.839**	0.913	0.595	0.975
PepMDLM	0.450	**1.000**	**0.705**	**0.513**

[1] The best scores are bolded. HELM-GPT was trained on HELM notation, where each token is a monomer encoding natural and modified residues. Since there are no existing peptide SMILES generative models, we chose HELM-GPT as the closest comparison. The validity is assessed differently since we use our in-house peptide SMILES filter.

$\vec{\delta} = \left[0, \frac{1}{|B|}, \frac{2}{|B|}, \ldots, \frac{|B|-1}{|B|}, 1\right]$ to get a vector $\vec{t} = (\vec{t} + \vec{\delta}) \bmod \mathbf{1}$ of evenly distributed time steps to ensure the model learns to regenerate the clean sample $\mathbf{z}_t$ for a continuous range of time steps. After applying bond-dependent masking to each training sequence, obtaining the predicted probabilities $\mathbf{x}_\theta(\mathbf{z}_t, t)$ and sampling the discrete sequence $\tilde{\mathbf{x}}_0$ from greedy `argmax` sampling, we minimize the total loss function $\mathcal{L}$ given by

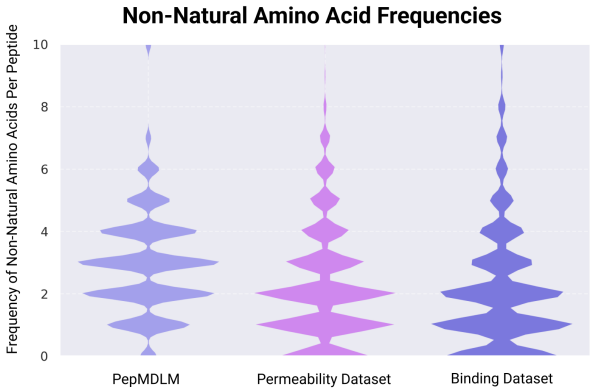
$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{\text{NELBO}}^{\infty} + \mathcal{L}_{\text{invalid}} \\ &= \frac{1}{|B|} \sum_{\mathbf{x}_0 \in B} \Bigg(- \sum_{\ell : \mathbf{x}_0^{(\ell)} = \mathbf{b}} \frac{w}{t} \log \langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t) \rangle - \sum_{\ell : \mathbf{x}_0^{(\ell)} \neq \mathbf{b}} \frac{1}{t} \log \langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t) \rangle \\ &\quad + \sum_{\ell=1}^{L} \tilde{\mathbf{x}}_0^{(\ell)\top} \mathrm{SM}\big(\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t)\big) \cdot \mathbf{1}[\tilde{\mathbf{x}}_0 \text{ is Invalid}] \Bigg)\end{aligned} \tag{12}$$

By increasing batch size and applying dynamic batching (Methods B.1), we obtain a tighter ELBO of the true distribution $\log p(\mathbf{x}_0)$. The model used to generate the validation results in this manuscript is trained on our in-house 8×A6000 Nvidia GPUs (50G memory) for 1600 GPU hours using the AdamW optimizer with a learning rate of 0.0003 and weight decay of 0.075.

Sampling. To sample from the unconditional PepMDLM model, we start with a sequence of length L of only [MASK] tokens. We first compute the diffusion time steps $t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}$ where $T = 128$. From the predicted token probabilities $\mathbf{x}_\theta(\mathbf{z}_t, t)$ generated by feeding $\mathbf{z}_t$ through the trained RoFormer backbone, we compute the reverse transition token distribution $p_\theta(\mathbf{z}_s | \mathbf{z}_t)$ following Equation (5) and perform Gumbel-max sampling to get the next token $\mathbf{z}_s$.

$$\begin{aligned}\mathbf{z}_s^{(\ell)} &\sim \arg\max \left(\log p_\theta(\mathbf{z}_s^{(\ell)} | \mathbf{z}_t^{(\ell)}) + \mathbf{G}^{(\ell)} \right) && (\mathbf{G}^{(\ell)} \in \mathbb{R}^{|\mathcal{V}|}) \\ G_i^{(\ell)} &= -\log(-\log(u_i^{(\ell)} + \epsilon) + \epsilon) && (u_i^{(\ell)} \sim \text{Uniform}(0, 1))\end{aligned}$$

where G_i is the Gumbel noise applied to the token probability at index i for position ℓ and $\epsilon = 1e{-}10$. Then, we return the newly sampled tokens only when $\mathbf{z}_t^{(\ell)} = \mathbf{m}$, while keeping all unmasked tokens unchanged. After T timesteps, we obtain a fully unmasked sequence $\mathbf{x}$.



	Permeability Data	Binding Data	PepMDLM
Mean nAAs Per Peptide	2.215	2.150	2.940
Cyclic Peptides (%)	0.467	0.027	0.100

Figure 12: **PepMDLM generates cyclic and modified peptides.** (Above) Distribution comparison of non-natural amino acid frequency for 100 unconditionally-generated peptide SMILES with the peptide SMILES dataset of experimentally-validated peptides for membrane permeability (PAMPA) and binding affinity (Methods B.1). (Bottom) Per peptide frequency of non-natural amino acids (nAAs) and percentage of cyclic peptides in PepMDLM-generated sequences and experimentally-validated membrane-permeable peptides.

B.4 Multi-Objective Guidance for Discrete Diffusion

In this section, we introduce the concept of Pareto dominance and non-dominance for multiple objectives and describe the Monte Carlo Tree Search (MCTS)-based algorithm for generating a set of Pareto-optimal sequences using our trained masked discrete diffusion model.

Pareto Optimization. When optimizing sequences for multiple objectives (e.g. affinity to multiple protein targets, membrane permeability, solubility, etc.), there is likely no single best sequence that achieves the highest score across all objectives. Optimizing one objective often leads to sacrificing performance on another objective.

Therefore, we focus on finding a set of Pareto optimal sequences that minimize the trade-offs between objectives to achieve overall optimal performance across all objectives. Formally, Pareto-optimal sequences (or non-dominated sequences) cannot be further optimized in any single objective without sacrificing performance in another objective.

Let $\mathbf{s}(\mathbf{x}) = [s_1(\mathbf{x}), \ldots, s_K(\mathbf{x})] \in \mathbb{R}^K$ be a vector of scores that measures the performance of a sequence $\mathbf{x}$ in K different objectives, with higher scores indicating better performance. A sequence $\mathbf{x}^*$ is said to dominate another sequence $\mathbf{x}$ (denoted as $\mathbf{x}^* \succ \mathbf{x}$) if and only if it satisfies the following property. For all objectives $k \in [1 \ldots K]$, the score for the kth objective for $\mathbf{x}^*$ is greater than or equal to the score for the kth objective for $\mathbf{x}$, and for at least one objective k', the score for $\mathbf{x}^*$ is strictly greater than the score for $\mathbf{x}$.

$$\underbrace{\mathbf{s}(\mathbf{x}^*) \succ \mathbf{s}(\mathbf{x})}_{\mathbf{x}^* \text{ dominates } \mathbf{x}} \quad \text{iff} \quad \underbrace{\forall k \in [1, K] \;\; s_k(\mathbf{x}^*) \geq s_k(\mathbf{x})}_{\mathbf{x}^* \text{ is no worse than } \mathbf{x} \text{ in any objective}} \wedge \quad \underbrace{\exists k' \in [1, K] \;\; s_{k'}(\mathbf{x}^*) \geq s_{k'}(\mathbf{x})}_{\mathbf{x}^* \text{ is strictly better than } \mathbf{x} \text{ in at least one objective}} \tag{13}$$

A Pareto-optimal sequence $\mathbf{x}$ is a sequence where there does not exist another sequence $\mathbf{x}^*$ in the current Pareto-optimal set $\mathcal{P}^*$ that dominates it. Since there are trade-offs between objectives, this does not mean that $\mathbf{x}$ is dominant to all other sequences.

$$\underbrace{\nexists \mathbf{x}^* \in \mathcal{P}^* \;\; \text{s.t.} \;\; \mathbf{s}(\mathbf{x}^*) \succ \mathbf{s}(\mathbf{x})}_{\mathbf{x} \text{ is non-dominated}} \tag{14}$$

We define the Pareto front as the set of non-dominated sequences $\mathbf{x}$ and their K-dimensional objective score vectors.

$$\mathcal{P}^* = \left\{ \big(\mathbf{x}, \mathbf{s}(\mathbf{x})\big) \mid \nexists \mathbf{x}^* \in \mathcal{P}^* \;\; \text{s.t.} \;\; \mathbf{s}(\mathbf{x}^*) \succ \mathbf{s}(\mathbf{x}) \right\} \tag{15}$$

Since infinitely many trade-offs can exist between the K objectives, there can be an infinite number of Pareto-optimal sequences. Therefore, multi-objective optimization aims to approximate a finite set of Pareto-optimal sequences with a reasonable number of iterations.

Notation. Let $\mathbf{z}_t$ denote the partially unmasked sequence at time t. $\mathbf{z}_t$ also corresponds to a node in the MCTS tree with a set of M children nodes denoted as $\text{children}(\mathbf{z}_t) = \{\mathbf{z}_{s,1}, \ldots, \mathbf{z}_{s,M}\}$. Each child node is itself a partially unmasked sequence at time s derived from sampling the MDLM reverse posterior $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$. The children nodes at each iteration of MCTS are rolled out into a set of clean sequences denoted as $\{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,M}\}$, for each of which we compute a score vector $\mathbf{s}(\mathbf{x}_{s,i}) \in \mathbb{R}^K$ and a rewards vector $\mathbf{r}(\mathbf{x}_{s,i}) \in \mathbb{R}^K$, where K is the number of objectives guiding the MCTS search.

Let $\mathcal{P}^* = \{\mathbf{x}_n^*\}$ be the set of $|\mathcal{P}^*|$ Pareto non-dominated sequences indexed $n \in [1 \ldots |\mathcal{P}^*|]$, which is updated at each iteration. At each node $\mathbf{z}_t$, we store a cumulative rewards vector $\mathbf{W}(\mathbf{z}_t)$ and a counter for the number of times the node has been visited across all iterations $N_{\text{visit}}(\mathbf{z}_t)$. Finally, we denote the iteration index as $i \in [1 \ldots N_{\text{iter}}]$, where N_{iter} is the total number of search iterations.

Initialization. We initialize a sequence of length L with only [MASK] tokens as the root node of the MCTS tree at $t = 0$ and an empty set $\mathcal{P}^*$ that will maintain clean sequences with Pareto-optimal score vectors. We initialize a set of scoring functions $\mathbf{s} : \mathcal{V}^L \to \mathbb{R}^K$ that takes a clean sequence $\mathbf{x}_{s,i} \in \mathcal{V}^L$ generated from the partially masked sequence $\mathbf{z}_{s,i}$ and outputs a vector of real values $\mathbf{s}(\mathbf{x}_{s,i}) \in \mathbb{R}^K$ that measures its performance in each of the K objectives. We also set the hyperparameters, including the number of iterations N_{iter}, the number of children M, and the length of the token sequence L.

At each iteration, four steps are performed to update the set of Pareto optimal solutions: traversing the tree by selecting the best child node until reaching a leaf node (selection), expanding the leaf node into M distinct partially unmasked sequences (expansion), fully unmasking each child node into a clean sequence and computing multi-objective score and reward vector (rollout), and finally backpropagating the total rewards to the predecessor nodes to guide the selection process at the next iteration (backpropagation).

Selection. At each iteration, we traverse the tree starting at the root node (fully masked sequence) $\mathbf{z}_T$ and selecting a child node based on the selection score vector $\mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i})$ that balances child nodes that generate high reward sequences from previous iterations and unexplored unmasking actions that could lead to a larger pool of diverse sequences.

$$\mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i}) = \frac{\mathbf{W}(\mathbf{z}_{s,i})}{N_{\text{visit}}(\mathbf{z}_{s,i})} + c \cdot p_\theta(\mathbf{z}_{s,i}|\mathbf{z}_t) \frac{\sqrt{N_{\text{visit}}(\mathbf{z}_t)}}{1 + N_{\text{visit}}(\mathbf{z}_{s,i})} \tag{16}$$

The first term is the cumulative reward vector $\mathbf{W}(\mathbf{z}_{s,i})$ normalized by the number of times the node was previously visited. This guides the selection process towards the unmasking step that has resulted in fully unmasked sequences with optimal properties without biasing towards highly visited nodes. The second term is a scalar added element-wise to the normalized rewards. The scalar probability of the unmasking step based on the unconditional reverse posterior $p_\theta(\mathbf{z}_{s,i}|\mathbf{z}_t)$ guides the selection towards the unmasking step with the highest probability to generate a valid peptide based on the pre-trained MDLM. When the number of times the parent node has been explored is high and the number of visits to a child node is low, the $\frac{\sqrt{N_{\text{visit}}(\mathbf{z}_t)}}{1+N_{\text{visit}}(\mathbf{z}_{s,i})}$ term encourages exploration of the unexplored unmasking scheme given that $p_\theta(\mathbf{z}_{s,i}|\mathbf{z}_t)$ is sufficiently high. However, as the number of visits to a child node increases, the impact of the second term decreases and the cumulative rewards dominate the selection score vector. c is a scalar hyperparameter that determines the degree of exploration compared to exploiting high-reward nodes, which is selected to be $c = 0.1$.

Then, we select uniformly at random from the pool of children nodes $\mathbf{z}_{s,i} \in \mathcal{P}^*_{\text{select}}$ whose selection score vectors are non-dominated, such that there doesn't exist another child $\mathbf{z}_{s,j}$ where the selection score across each of the K-objectives is equal to the score of $\mathbf{z}_{s,i}$ and there exists a score strictly greater than $\mathbf{z}_{s,i}$.

$$\mathcal{P}^*_{\text{select}} = \{\mathbf{z}_{s,i} \mid \nexists \mathbf{z}_{s,j} \in \text{children}(\mathbf{z}_t) \;\; \text{s.t.} \;\; \mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,j}) \succ \mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i})\} \tag{17}$$

If the selected node is a non-leaf node, the loop repeats with the selected node $\mathbf{z}_{s,i}$ as the new parent node. If a fully unmasked node with $t = 0$ is reached, we restart the selection process from the root node. Once a leaf node is reached, the loop ends and the next step executes.

Expansion. At the iteration at time t, we sample M sequences from the reverse posterior $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$ defined in Equation (5) to get a set of partially masked sequences which form the set of children nodes of $\mathbf{z}_t$: children$(\mathbf{z}_t) = \{\mathbf{z}_{s,1}, \ldots, \mathbf{z}_{s,M}\}$. All the children nodes are added to the tree.

To ensure that the expansion step results in M distinct unmasking steps, we experimented with two different batched unmasking techniques from the single partially masked sequence at a parent node. For the first method, we repeated the array corresponding to the parent node tokens over M dimensions and added independently sampled Gumbel noise values $G_{i,j}^{(\ell)}$, where i denotes the sequence in the batch, ℓ denotes the position in the sequence, and j denotes the token index.

$$\log \tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) = \log p_\theta\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) + \mathbf{G}_i^{(\ell)} \tag{18}$$

$$G_{i,j}^{(\ell)} = -\log\Big(-\log(u_{i,j}^{(\ell)} + \epsilon) + \epsilon\Big) \qquad u_{i,j}^{(\ell)} \sim \text{Uniform}(0,1) \tag{19}$$

where $\tilde{p}_{\theta,i}$ denotes the ith modified reverse transition distribution after applying Gumbel noise independently sampled for each token at each position for each sequence i in the batch, where $i = [1 \ldots M]$. Then, for each position ℓ in the sequence, we sample M distinct child sequences from the M distinct distributions $\mathbf{z}_{s,i}^{(\ell)} \sim \tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big)$.

The second method involves taking the `softmax` (denoted as SM) across the top k probabilities after applying Gumbel noise and drawing random samples from the re-normalized softmax distribution over only the top k most probable tokens.

$$\log \tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) = \text{SM}\Big(\text{top}k\big\{\log p_\theta\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) + \mathbf{G}_i^{(\ell)}\big\}\Big) \tag{20}$$

After empirical experimentation, we found that the first method results in higher diversity across sequences whereas the second method prevents unlikely tokens. Since the reward generated by a sequence ultimately determines whether it is selected in subsequent iterations, we chose the first method to allow for greater exploration during the expansion step.

Rollout. From each child node generated at time s, we completely unmask the sequence by greedily sampling the `argmax` tokens from the predicted reverse transition distribution $p_{\theta,i}^{(\ell)}(\mathbf{z}_{s'}|\mathbf{z}_s)$ for all remaining time steps $s \to 0$ and $s' = s - \frac{1}{T}$ to get a set of clean sequences $\{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,M}\}$ of SMILES tokens. Then, we feed each clean sequence $\mathbf{x}_{s,i}$ as input to the scoring functions for all of the K objectives to generate the score vector $\mathbf{s}(\mathbf{x}_{s,i}) = \big[s_1(\mathbf{x}_{s,i}), \ldots, s_K(\mathbf{x}_{s,i})\big] \in \mathbb{R}^K$. Then, we use the score vector to compute a vector of rewards $\mathbf{r}(\mathbf{x}_{s,i}) = \big[r_1(\mathbf{x}_{s,i}), \ldots, r_K(\mathbf{x}_{s,i})\big] \in \mathbb{R}^K$.

The reward of a child node sequence for the kth objective is the fraction of the sequences $\mathbf{x}_n^*$ in the current set of Pareto-optimal sequences $\mathcal{P}^*$ where the child node has a higher classifier score in that objective. Specifically, the reward for the ith child node $\mathbf{z}_{s,i}$ and the resulting unmasked sequence $\mathbf{x}_{s,i}$ for the kth objective is given by

$$r_k(\mathbf{x}_{s,i}) = \frac{1}{|\mathcal{P}^*|}\sum_{n=1}^{|\mathcal{P}^*|} \mathbf{1}\big[s_k(\mathbf{x}_{s,i}) \geq s_k(\mathbf{x}_n^*)\big] \tag{21}$$

where $\mathbf{1}$ is an indicator function that returns 1 if the score for the kth objective of the ith child node is greater than or equal to the score of the nth sequence in the Pareto-optimal set $\mathcal{P}^*$. In parallel to computing the reward, we add all non-dominated children sequences to the set of Pareto optimal sequences $\mathcal{P}^*$ and remove all dominated sequences (Algorithm 6).

$$\mathcal{P}'^* = \mathcal{P}^* \cup \big\{(\mathbf{z}_{s,i}, \mathbf{s}(\mathbf{x}_{s,i})) \mid \forall \tilde{\mathbf{x}} \in \mathcal{P}^* \;\; \mathbf{s}(\mathbf{x}_{s,i}) \succeq \mathbf{s}(\tilde{\mathbf{x}})\big\} \tag{22}$$

$$\mathcal{P}'^* = \mathcal{P}^* \setminus \big\{\tilde{\mathbf{x}} \mid \exists \mathbf{x}_{s,i} \in \text{children}(\mathbf{z}_t) \text{ s.t. } \mathbf{s}(\mathbf{x}_{s,i}) \succ \mathbf{s}(\tilde{\mathbf{x}})\big\} \tag{23}$$

In Appendix F.1, we show a proof-of-concept for a time-dependent multi-objective guidance strategy where the update to the Pareto-optimal set $\mathcal{P}^*$ depends on the rewards for only a subset of the K objectives that varies depending on the current iteration, enabling the prioritization of properties with larger influence on peptide structure and function in earlier iterations and fine-tuning on additional properties in later iterations.

Back-propagation. At each child node $\mathbf{z}_{s,i}$, the reward vector $\mathbf{r}(\mathbf{x}_{s,i})$ is used to initialize the cumulative reward vector $\mathbf{W}(\mathbf{z}_{s,i})$, and the number of visits $N_{\text{visits}}(\mathbf{z}_{s,i})$ is initialized to 1.

$$\mathbf{W}(\mathbf{z}_{s,i}) \leftarrow \mathbf{r}(\mathbf{x}_{s,i}) \tag{24}$$

$$N_{\text{visit}}(\mathbf{z}_{s,i}) \leftarrow 1 \tag{25}$$

Then, we backtrack through the predecessor nodes of $\mathbf{z}_{s,i}$ up to the root node $\mathbf{z}_T$, adding the child reward vector to the cumulative reward vector and incrementing the number of visits for each node in the path. For all nodes from $\mathbf{z}_t = \text{parent}(\mathbf{z}_{s,i})$ to $\mathbf{z}_t = \mathbf{z}_T$ we apply the following update

$$\mathbf{W}(\mathbf{z}_t) \leftarrow \mathbf{W}(\mathbf{z}_t) + \sum_{i=1}^{M} \mathbf{r}(\mathbf{x}_{s,i}) \tag{26}$$

$$N_{\text{visit}}(\mathbf{z}_t) \leftarrow N(\mathbf{z}_t) + 1 \tag{27}$$

These updated scores are used to guide the selection process in the next iteration such that the unmasking paths that result in the highest reward sequences have a greater chance of being selected and explored further.

Penalizing Invalid Peptides. To discourage the selection process from choosing unmasking steps that result in invalid or unsynthesizable peptide SMILES, we subtract a penalty score calculated as the fraction of invalid SMILES sequences rolled out from the expanded children of a parent node determined based on our peptide SMILES validity filter. Since properties are irrelevant if the peptide SMILES is invalid, we subtract the penalty scaled by a constant c element-wise from the cumulative reward vectors. This penalty-adjusted reward vector is backpropagated to all parent nodes to avoid paths leading to high invalid rates in the next iteration. After empirical experimentation, we determined that setting the constant $c = 0.5$ consistently results in 100% peptide validity rate after only $\sim$20 iterations of the MCTS search.

$$\mathbf{W}(\mathbf{z}_t) \leftarrow \mathbf{W}(\mathbf{z}_t) + \sum_{i=1}^{M} \mathbf{r}(\mathbf{z}_{s,i}) - c \cdot \left(\frac{N_{\text{invalid}}}{M} \right) \tag{28}$$

Output. The output after N_{iter} iterations is the set $\mathcal{P}^*$ of Pareto-optimal sequences across the K objectives. Our strategy simultaneously guides the unmasking process towards optimality across multiple objectives directly in the discrete state space while exploring the diverse space of peptide sequences using the trained unconditional MDLM generator. Furthermore, we generate an set of Pareto-optimal sequences from a single run through the MCTS-search algorithm which are non-dominated from the total of $N_{\text{iter}} \cdot M$ total sequences sampled across all iterations.

B.5 Therapeutic Property Prediction for Peptide SMILES

While several classifiers exist for predicting properties of small-molecule SMILES sequences and amino-acid representations of peptides, there exists a gap in high-quality property models trained specifically on peptide SMILES data. To fill this gap, we train regression models for target-binding affinity and cell membrane permeability and binary classification models for solubility, hemolysis, and non-fouling specifically on peptide SMILES data (Table 6).

Table 6: Benchmarks of solubility, hemolysis, and non-fouling prediction for PeptideCLM and PeptideBERT embeddings. Each was trained using XGBoost for classification.

	Solubility		**Hemolysis**		**Non-fouling**	
Metric	Ours	PeptideBERT	Ours	PeptideBERT	Ours	PeptideBERT
F1	**0.660**	0.597	**0.846**	0.483	**0.768**	0.699
Accuracy	**0.661**	0.651	**0.846**	0.823	0.766	**0.873**

[1] The optimal thresholds for the positive class were determined to be 0.500 for solubility, 0.800 for hemolysis (non-hemolysis), and 0.450 for non-fouling.

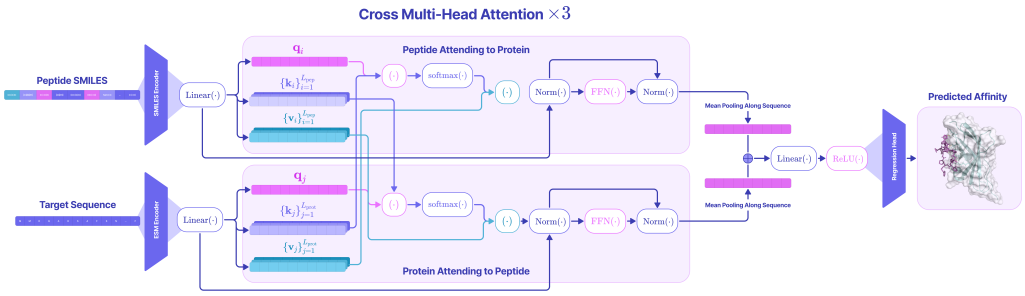


Figure 13: **Architecture of binding affinity regression model.** Embeddings for the target protein sequence are generated with ESM-2 and embeddings for the peptide SMILES are generated using PeptideCLM. Cross multi-head attention layers combine the embeddings and predict a binding affinity score.

To guide the generation of peptides with high binding affinity to a given protein target, we trained a Transformer-based model with cross multi-head attention layers that learn the joint latent space of ESM-2-650M [49] embeddings of the protein amino acid sequence and PeptideCLM [20] embeddings of the peptide SMILES sequence (Figure 13. See full architectural details in Appendix E.3). Given a peptide SMILES sequence and a protein amino acid sequence, the model was trained on natural and non-natural peptide SMILES $K_d/K_i/IC50$ binding affinity to predict a score that indicates weak binding (< 6.0), medium binding (6.0-7.5), and high binding ($\geq$ 7.5). Our regression model achieved a strong Spearman correlation coefficient of 0.869 on the training data and 0.633 on the held-out validation data.

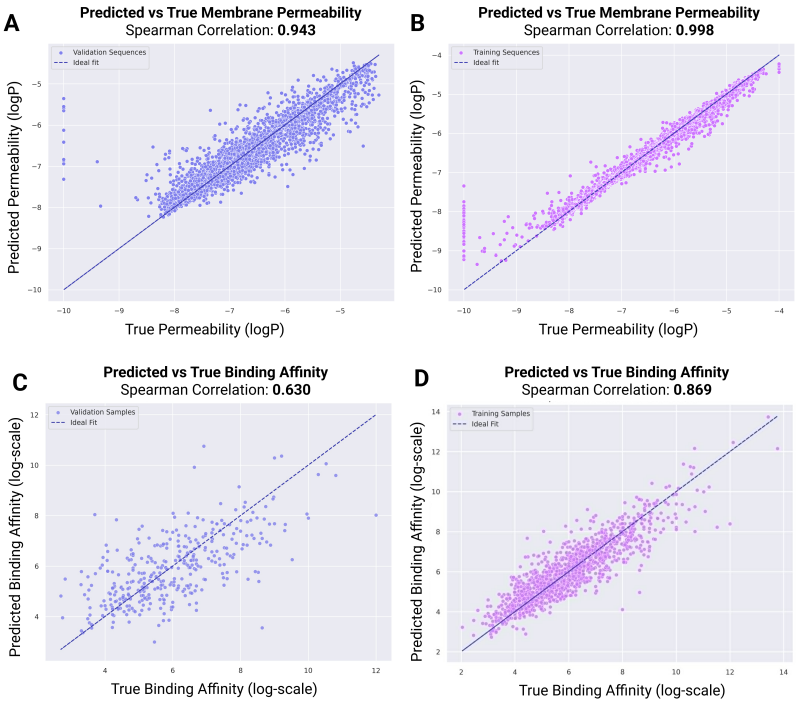


Figure 14: **Correlation plots for binding affinity and membrane permeability classifiers.** Plot of true permeability (logP) on the x-axis and predicted permeability on the y-axis for the (**A**) validation set and (**B**) training set. Plot of true binding affinity (log-scale) on the x-axis and predicted permeability on the y-axis for the (**C**) validation set and (**D**) training set.

For cell membrane permeability, we trained an XGBoost [12] boosted tree regression model on PeptideCLM [20] embeddings which returns the predicted PAMPA lipophilicity score ($\log P$) given a peptide SMILES sequence, where values ≥ -6.0 indicate strong permeability and values < 6.0 indicate weak permeability. We trained our model on 34,853 experimentally validated peptide SMILES (See Methods B.1) and achieved a strong Spearman correlation coefficient of 0.998 on the training dataset and 0.943 on the test dataset (Figure 14, Table 7).

Table 7: Held-out validation performance of binding affinity and membrane permeability regression models trained on peptide SMILES.

Metric	**Binding Affinity**	**Membrane Permeability**
Spearman Rank Correlation	0.633	0.943
MSE	0.566	0.088

[1] Spearman rank correlation and MSE were calculated on the 20 percent held-out validation set

B.6 Evaluation

Peptide Validity Filter. Among the sequential representations of peptides including amino acid sequences, HELM [92], and SMILES [84], SMILES is the most intricate representation of peptide sequences. Although this enables the representation of non-natural amino acids, diverse side-chain, and backbone modifications, and cyclic peptides, it also means that the vast majority of SMILES strings are not synthesizable peptides. Therefore, we devised an algorithm that determines whether a SMILES string is a valid peptide, characterized by peptide bonds and central carbon atoms. The filter first checks if the SMILES sequence is a valid molecule using RDKit [63].

Then, we use regular expressions to detect bond patterns for peptide bonds, N-methylated peptide bonds, reversed peptide bonds, and ester bonds, along the sequence to split the sequence into several segments with a bond before and after each segment. The filter checks each segment for chemical modifications based on their bond type, including N-methylation (N-Me) and O-linked glycosylation. The remaining segment is matched to the corresponding natural or non-natural amino acid side chains (Algorithm 8). Our filter is capable of detecting a library of over 200 nAAs from SwissSidechain [22] and can classify a peptide SMILES as cyclic or non-cyclic. The tool is freely available on HuggingFace: `https://huggingface.co/spaces/ChatterjeeLab/SMILES2PEPTIDE`.

Generation Quality Metrics. To evaluate the generation quality of our unconditional MDLM, PepMDLM, and our MCTS-guided MDLM, PepTune, we leverage the Moses metrics, including validity, uniqueness, diversity, similarity to nearest neighbor (SNN), and novelty [62]. Validity is the percentage of generated sequences that are valid peptide SMILES based on our peptide validity filter. Uniqueness is the fraction of distinct peptide SMILES sequences among the valid peptide SMILES. Diversity is calculated as one minus the average Tanimoto similarity between the Morgan fingerprints for every pair of generated sequences. Similarity to nearest neighbor (SNN) takes the maximum Tanimoto similarity score for each generated sequence with the sequences in the training dataset and averages across all generated sequences to measure the average similarity of a generated sequence with its most similar neighbor.

Due to the limit of memory and CPU time required to load all the training dataset of 11 million peptide SMILES, we chose to sample a subset of 1000 batches randomly ($\sim$100k sequences) for novelty and SNN calculation. To assess the novelty of generated sequences, we employed Shannon entropy [48] to quantify the SMILES token randomness between 100 PepTune-generated and 100 PepMDLM-generated sequences and the same randomly sampled 1000 subsets from the training set. Then Kullback-Leibler (KL) divergence was also used to evaluate divergence across token distributions from the generated peptides compared to the training data. The equations for all metrics are provided in Appendix E.5.

Docking. For valid generated peptide SMILES with non-dominated scores across objectives, we used Autodock Vina [18] (v 1.1.2) for *in silico* docking of the peptide binders to their target proteins (Appendix 9) to confirm binding affinity. Targets were preprocessed with MGITools [55] (v 1.5.7) and the conformations of the SMILES were optimized by ETKDG from RDKit [18, 80]. The final

results were visualized in PyMol [69] (v 3.1), where the residues in the protein targets with polar contacts to the peptide binder with distances closer than 3.5 Å are annotated.

C EXTENDED BACKGROUND

C.1 CONTINUOUS-TIME DISCRETE DIFFUSION

Discrete diffusion models [3] are a subset of diffusion models where the forward corruption and reverse denoising processes operate in the discrete latent space via categorical probability distributions for discrete variables.

We denote a token in a sequence from the dataset as a one-hot vector $\mathbf{x}_0^{(\ell)} \in \{0,1\}^{|\mathcal{V}|}$. The discrete-time forward diffusion process involves applying categorical noise to $\mathbf{x}_0$ at varying degrees based on a noise schedule $\sigma(t)$ for a total of T time steps ranging from no noise at $t = 0$ to maximum noise at $t = 1$. To clearly distinguish each step, we denote the nth transition in the forward pass as the transition from $s(n)$ to $t(n)$, where $s(n) = \frac{n-1}{T}$ and $t(n) = \frac{n}{T}$. The marginal noise that transforms the sequence $\mathbf{z}_{s(n)}$ at time $s(n)$ to a progressively noisier sequence $\mathbf{z}_{t(n)}$ at the next time step $t(n) = s(n) + \frac{1}{T}$ is given by a $|\mathcal{V}| \times |\mathcal{V}|$ marginal transition matrix $\mathbf{Q}_{t|s} = \sigma(t)\mathbf{I}_{|\mathcal{V}|} + (1-\sigma(t))\mathbf{1}\pi^\top$, where the (i,j)th entry denotes the probability of transitioning from token i to token j at each position in the sequence.

Therefore, the marginal categorical distribution of $\mathbf{z}_{t(n)}$ in the discrete-time forward-pass diffusion process can be derived as

$$\begin{aligned} q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)}) &= \text{Cat}(\mathbf{z}_{t(n)}; \mathbf{Q}_{t|s}^\top \mathbf{z}_{s(n)}) \\ &= \text{Cat}(\mathbf{z}_{t(n)}; \sigma(t(n))\mathbf{z}_{s(n)} + (1-\sigma(t(n)))\pi) \end{aligned} \tag{29}$$

where $\sigma(t(n))$ the marginal probability of a single position in the sequence remaining the same token during the transition $s(n) \to t(n)$ and $\big(1-\sigma(t(n))\big)$ is the marginal probability of transitioning to a different token based on the token probability distribution $\pi \in \mathbf{\Delta}^{|\mathcal{V}|}$. For simplicity, we denote $\sigma(t(n)) = \sigma(n)$.

The cumulative transition from time 0 to time $t(t)$ is denoted as the product of the marginals $\mathbf{Q}_t = \prod_{n=0}^{t} \mathbf{Q}_{t|s}$, which is the product of marginal transitions $s(n) \to t(n)$ for n ranging from 0 to t applied to the clean input sequence $\mathbf{x}_0$.

$$\mathbf{Q}_t = \left(\prod_{n=0}^{t}(1-\sigma(n))\right)\mathbf{I} + \left(1 - \prod_{n=0}^{t}(1-\sigma(n))\right)\mathbf{1}\pi^\top \tag{30}$$

For the continuous-time forward pass diffusion process where $T \to \infty$ and $\frac{1}{T} \to 0$, we can take the limit as $T \to \infty$ to derive an expression for the continuous-time transition probability, α_t.

$$\begin{aligned} \lim_{T\to\infty} \prod_{n=0}^{t}(1-\sigma(n)) &= \lim_{T\to\infty} \exp\left(\log \prod_{n=0}^{t}(1-\sigma(n))\right) \\ &= \lim_{T\to\infty} \exp\left(\sum_{n=0}^{t} \log(1-\sigma(n))\right) \\ &\approx \lim_{T\to\infty} \exp\left(\sum_{n=0}^{t} -\sigma(n)\right) \qquad (\log(1-x) \approx -x \text{ for small } x) \\ &= \exp\left(-\int_{n=0}^{t} \sigma(n)dn\right) \\ &= \exp\left(-\int_{s=0}^{t(t)} \sigma(s)ds\right) \end{aligned} \tag{31}$$

We have shown that the continuous-time forward transition probability from $t = 0$ to $t = t(t)$ is $\alpha_t = \exp\left(-\int_{s=0}^{t(t)} \sigma(s)ds\right) = \exp(-\bar{\sigma}(t))$ where $\bar{\sigma}(t) = \int_{s=0}^{t(t)} \sigma(s)ds$. Letting $t = t(t)$, we can

define the continuous-time cumulative transition matrix $\mathbf{Q}_t$ at the limit where $T \to \infty$ and the continuous-time distribution $q(\mathbf{z}_t|\mathbf{x}_0)$ as

$$\mathbf{Q}_t = \alpha_t \mathbf{I} + \alpha_t \mathbf{1}\pi^\top \tag{32}$$

$$\begin{aligned} q(\mathbf{z}_t|\mathbf{x}_0) &= \text{Cat}(\mathbf{z}_t; \mathbf{Q}_t \mathbf{x}_0) \\ &= \text{Cat}(\mathbf{z}_t; \alpha_t \mathbf{x}_0 + (1-\alpha_t)\pi) \end{aligned} \tag{33}$$

It follows that the marginal transition $\mathbf{Q}_{s|t}$ is the cumulative transition $\mathbf{Q}_t$ divided by all previous transition probabilities, denoted as $\mathbf{Q}_s = \prod_{r=0}^{s} \mathbf{Q}_{s|r}$, so $\alpha_{s|t} = \frac{\alpha_t}{\alpha_s}$.

Following Austin et al. [3] and substituting the marginal and cumulative probability distributions, we derive the true reverse transition from $t \to s$ conditioned on a clean sequence $\mathbf{x}_0$ as

$$\begin{aligned} q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0) &= \frac{q(\mathbf{z}_t|\mathbf{z}_s, \mathbf{x}_0) q(\mathbf{z}_s|\mathbf{x}_0)}{q(\mathbf{z}_t|\mathbf{x}_0)} \\ &= \text{Cat}\left(\mathbf{z}_s; \frac{\mathbf{Q}_{t|s}\mathbf{z}_t \odot \mathbf{Q}_s^\top \mathbf{x}_0}{\mathbf{z}_t^\top \mathbf{Q}_t^\top \mathbf{x}_0}\right) \\ &= \text{Cat}\left(\mathbf{z}_s; \frac{[\alpha_{t|s}\mathbf{z}_t + (1-\alpha_{t|s})\mathbf{1}\pi^\top \mathbf{z}_t] \odot [\alpha_s \mathbf{x}_0 + (1-\alpha_s)\pi]}{\alpha_t \mathbf{z}_t^\top \mathbf{x}_0 + (1-\alpha_t)\mathbf{z}_t^\top \pi}\right) \end{aligned} \tag{34}$$

where the numerator is the element-wise product of $|\mathcal{V}|$-dimensional vectors representing the marginal probability distribution of sampling $\mathbf{z}_t$ given $\mathbf{z}_s$ and the cumulative probability distribution for $\mathbf{z}_s$ from the original clean sequence $\mathbf{x}_0$. The denominator is a scalar probability of the specific token $\mathbf{z}_t$ being drawn from the noisy probability distribution at time t.

C.2 Continuous-Time Negative Evidence Lower Bound (NELBO)

The objective of denoising diffusion probabilistic models (DDPMs) [31] is to iteratively sample slightly less noisy intermediate sequences $\mathbf{z}_t$ until obtaining a clean sequence $\mathbf{x}$ that has a high probability of being drawn from the data distribution $p(\mathbf{x}_0)$. To train a model that accurately samples from $p(\mathbf{x}_0)$, we maximize the Evidence Lower Bound (ELBO) of $\log p_\theta(\mathbf{x}_0)$ which measures how accurately the model parameterized by θ generates true samples $\mathbf{x}_0$ given a corrupted sequence $\mathbf{z}_T$ by iterative sampling from the reverse posterior $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$. The ELBO is maximized when $p_\theta(\mathbf{x}_0) = 1$ and $\log(p_\theta(\mathbf{x}_0)) = 0$ for all sequences $\mathbf{x}_0$ in the dataset, which supports the objective of accurately generating sequences from the data distribution. To convert this into a loss minimization objective, we define the loss function as the negative ELBO (NELBO). First, we compute $\log p_\theta(\mathbf{x}_0)$ by integrating over the joint probability of all possible paths of intermediate states from the noisy state $\mathbf{z}_T$ at $t = T$ to the clean sample x_0 at $t = 0$, denoted by $p_\theta(\mathbf{x}_{0:T})$. Since our goal is to reverse the forward masking of the clean sample x_0 from all time steps, we introduce an encoder term $q(\mathbf{z}_{1:T}|\mathbf{x}_0)$ denoting the combined distribution of obtaining any noisy sequence between times $t = 1$ to $t = T$ from the clean sequence $\mathbf{x}_0$.

$$\begin{aligned} \log p_\theta(\mathbf{x}_0) &= \log \int p_\theta(\mathbf{z}_{0:T}) d\mathbf{z}_{1:T} \\ &= \log \int q(\mathbf{z}_{1:T}|\mathbf{x}_0) \left[\frac{p_\theta(\mathbf{z}_{0:T})}{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\right] d\mathbf{z}_{1:T} \\ &= \log \left(\mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)} \left[\frac{p_\theta(\mathbf{z}_{0:T})}{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\right]\right) \end{aligned} \tag{35}$$

where $\mathbf{z}_{0:T}$ includes $\mathbf{x}_0$.

Using Jenson's inequality, we move the logarithm inside the expectation and reverse the sign to get the NELBO for $\log p_\theta(\mathbf{x}_0)$. We split the terms associated with the forward noising process $q(\mathbf{z}_{1:T}|\mathbf{x}_0)$ and the reverse denoising model $p_\theta(\mathbf{z}_{0:T})$ into reconstruction term, the prior term, and

the intermediate diffusion term.

$$
\begin{aligned}
\mathcal{L}_{\text{NELBO}} &= \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{z}_{0:T})}{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\right] \\
&= \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)})p_\theta(\mathbf{z}_{t(T)})\prod_{n=1}^{T-1} p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{t(T-1)})\prod_{n=1}^{T-1} q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\right] \\
&= \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)}) - \log \frac{p_\theta(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})} - \log \frac{\prod_{t=1}^{T-1} p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{\prod_{n=1}^{T-1} q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\right] \\
&= \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)}) - \log \frac{p_\theta(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})} - \sum_{n=1}^{T-1} \log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{s(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\right] \\
&= \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\Big[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)})\Big] + \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\right] \\
&\quad + \sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{1:T}|\mathbf{x}_0)}\Big[-\log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\Big] \\
&= \underbrace{\mathbb{E}_{q(\mathbf{z}_{t(1)}|\mathbf{x}_0)}\Big[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)})\Big]}_{\text{reconstruction loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_{t(T)},\mathbf{z}_{s(T)}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\right]}_{\text{prior loss}} \\
&\quad + \underbrace{\sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{s(n)},\mathbf{z}_{t(n)},\mathbf{z}_{t(n+1)}|\mathbf{x}_0)}\Big[-\log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})}\Big]}_{\text{diffusion loss}}
\end{aligned} \tag{36}
$$

Now, we can take the limit for each of the loss terms as $T \to \infty$ to derive the continuous-time MDLM objective.

Reconstruction Loss $\mathcal{L}_{\text{reconst}}$. The reconstruction loss evaluates the final step of the reverse diffusion process that denoises the sequence at time $t(1)$ to the clean sequence at time $t = 0$. Since $t(0) = \frac{1}{T}$, the distribution that the sequence $\mathbf{z}_{t(1)}$ is drawn from in the forward pass diffusion is given by

$$
p(\mathbf{z}_{t(1)}|\mathbf{x}_0) = \text{Cat}(\mathbf{z}_{t(1)}; \alpha_{t(1)}(\mathbf{x}_0)\mathbf{x}_0 + (1 - \alpha_{t(1)}(\mathbf{x}_0))\mathbf{m}) \tag{37}
$$

Since we have $\alpha_t(\mathbf{x}_0) = 1 - t^w$ for $\mathbf{x}_0 = \mathbf{b}$ and $\alpha_t(\mathbf{x}_0) = 1 - t$ for $\mathbf{x}_0 \neq \mathbf{b}$, we can write

$$
\alpha_{t(1)}(\mathbf{x}_0)\mathbf{x}_0 + (1 - \alpha_{t(1)}(\mathbf{x}_0))\mathbf{m} = \begin{cases} \left(1 - \frac{1}{T^w}\right)\mathbf{x}_0 + \frac{1}{T^w}\mathbf{m} & \mathbf{x}_0 = \mathbf{b} \\ \left(1 - \frac{1}{T}\right)\mathbf{x}_0 + \frac{1}{T}\mathbf{m} & \mathbf{x}_0 \neq \mathbf{b} \end{cases} \tag{38}
$$

In the limit as $T \to \infty$, both cases converge to $\mathbf{x}_0$, so we have $\mathbf{z}_{t(1)} \sim \text{Cat}(\mathbf{z}_{t(1)}; \mathbf{x}_0)$ and $\mathbf{z}_{t(1)} = \mathbf{x}_0$. Since $q(\mathbf{z}_{t(1)}|\mathbf{x}_0) = \mathbf{x}_0$ in the forward pass, by parameterizing the reverse posterior to copy-over unmasked tokens, we get $p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)}) = \mathbf{x}_0$. Therefore, the reconstruction loss reduces to 0.

$$
\begin{aligned}
\mathbb{E}_{q(\mathbf{z}_{t(1)}|\mathbf{x}_0)}\Big[-\log p_\theta(\mathbf{x}_0|\mathbf{z}_{t(1)})\Big] &= \mathbb{E}_{q(\mathbf{z}_{t(1)}|\mathbf{x}_0)}\Big[-\log p_\theta(\mathbf{x}_0|\mathbf{x}_0)\Big] \\
&= 0
\end{aligned}
$$

Prior Loss $\mathcal{L}_{\text{prior}}$. The prior loss measures the first reverse transition from the fully masked sequence $\mathbf{z}_{t(T)}$ to a slightly unmasked sequence $\mathbf{z}_{s(T)}$.

$$
\begin{aligned}
\mathbb{E}_{q(\mathbf{z}_{t(T)},\mathbf{z}_{s(T)}|\mathbf{x}_0)}\left[-\log \frac{p(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\right] &= -\mathbb{E}_{q(\mathbf{z}_{s(T)}|\mathbf{x}_0)} \underbrace{\mathbb{E}_{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\left[\log \frac{p(\mathbf{z}_{t(T)})}{q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})}\right]}_{\text{KL Divergence}} \\
&= -\mathbb{E}_{q(\mathbf{z}_{s(T)}|\mathbf{x}_0)}\Big[\text{KL}\Big(q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)})||p_\theta(\mathbf{z}_{t(T)})\Big)\Big]
\end{aligned} \tag{39}
$$

Since $t(T) = 1$, we have $\alpha_{t(T)}(\mathbf{x}_0) = 1 - 1 = 0$. Therefore, we derive

$$\begin{aligned} q(\mathbf{z}_{t(T)}|\mathbf{x}_0) &= \text{Cat}\big(\mathbf{z}_{t(T)}\ ;\ \alpha_{t(T)}(\mathbf{x}_0)\mathbf{x}_0 + (1 - \alpha_{t(T)}(\mathbf{x}_0))\mathbf{m}\big) \\ &= \text{Cat}\big(\mathbf{z}_{t(T)}\ ;\ 0\mathbf{x}_0 + (1-0)\mathbf{m}\big) \\ &= \text{Cat}\big(\mathbf{z}_{t(T)}; \mathbf{m}\big) \end{aligned} \tag{40}$$

Since all sequences are completely masked at time T, it follows that the marginal distribution $q(\mathbf{z}_{t(T)}|\mathbf{z}_{s(T)}) = \text{Cat}\big(\mathbf{z}_{t(T)}; \mathbf{m}\big)$ and the prior distribution $p_\theta(\mathbf{z}_{t(T)}) = \text{Cat}\big(\mathbf{z}_{t(T)}; \mathbf{m}\big)$, so the KL divergence reduces to 0.

Diffusion Loss $\mathcal{L}_T$. The diffusion loss measures the consistency of each predicted reverse transition with the forward marginal transition for all T time steps.

$$\begin{aligned} &\sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{s(n)}, \mathbf{z}_{t(n)}, \mathbf{z}_{t(n+1)}|\mathbf{x}_0)} \left[-\log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})} \right] \\ &= -\sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{s(n)}, \mathbf{z}_{t(n+1)}|\mathbf{x}_0)} \underbrace{\mathbb{E}_{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})} \left[\log \frac{p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})}{q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})} \right]}_{\text{KL divergence}} \\ &= -\sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{s(n)}, \mathbf{z}_{t(n+1)}|\mathbf{x}_0)} \left[\text{KL}\Big(q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)}) || p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}) \Big) \right] \end{aligned} \tag{41}$$

Since the objective is to accurate predict $\mathbf{z}_{s(n)}$ given $\mathbf{z}_{t(n)}$, we cannot condition on the term $\mathbf{z}_{s(n)}$. Instead, we can condition $q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)})$ on $\mathbf{x}_0$ and use Baye's theorem to derive

$$q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)}, \mathbf{x}_0) = \frac{q(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}, \mathbf{x}_0) q(\mathbf{z}_{t(n)}|\mathbf{x}_0)}{q(\mathbf{z}_{s(n)}|\mathbf{x}_0)}$$

Rearranging the terms we get an expression for the true reverse transition $q(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}, \mathbf{x}_0)$ conditioned on the clean data $\mathbf{x}_0$. By minimizing the KL divergence between the learned reverse transition $p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})$ and $q(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}, \mathbf{x}_0)$, we can rewrite the diffusion loss as

$$\mathcal{L}_T = \sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{t(n)}|\mathbf{x}_0)} \left[\text{KL}\Big(q(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}, \mathbf{x}_0) || p_\theta(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}) \Big) \right] \tag{42}$$

In Appendix D.3, we derive the bond-dependent continuous-time NELBO loss from its general form above.

C.3 Guided Diffusion Models

Guided diffusion aims to sample from the data distribution conditioned on some property y, $\mathbf{x} \sim q(\mathbf{x}_0, y)$, such that $q(y|\mathbf{x})$ is maximized. Therefore, the marginal reverse transition aims to sample $\mathbf{z}_s$ from a distribution $q(\mathbf{z}_s|\mathbf{z}_t, y)$ conditioned on the current sequence $\mathbf{z}_t$ and a property y. Using Baye's theorem, we can decompose the guided conditional distribution as

$$q(\mathbf{z}_s|\mathbf{z}_t, y) = \frac{q(y|\mathbf{z}_s, \mathbf{z}_t) q(\mathbf{z}_s|\mathbf{z}_t)}{q(y|\mathbf{z}_t)} \tag{43}$$

There are two strategies to generate samples from this conditional distribution: classifier-free and classifier-based guidance.

Classifier-Free Guidance. Classifier-free guidance strategies aim to model the conditional distribution $q(\mathbf{z}_s|\mathbf{z}_t, y)$ by directly training the diffusion model on a subset of the unconditional data with property y, such that after training, the model samples from a learned distribution $p_\theta(\mathbf{z}_s|\mathbf{z}_t, y)$. However, classifier-free guidance fails at tasks where quality annotated data is scarce, including peptide sequences. Furthermore, this strategy cannot scale to multiple objectives which would require data conditioned on more than one property.

Classifier-Based Guidance. Classifier-based guidance trains an unconditional diffusion model $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$ and a classifier model $p_\phi(y|\mathbf{z}_s)$ with learned parameters ϕ that generates a score measuring

the probability that the intermediate sequence $\mathbf{z}_s$ has property y. By Bayes' theorem, we can model the conditional distribution as

$$p_{\theta,\phi}(\mathbf{z}_s|\mathbf{z}_t, y) = \frac{p_\phi(y|\mathbf{z}_s)p_\theta(\mathbf{z}_s|\mathbf{z}_t)}{p_\phi(y|\mathbf{z}_t)} \tag{44}$$

Since the model parameters implicitly learn the normalized distribution, we can drop the $p_\phi(y|\mathbf{z}_t)$ term. Then, at each iteration, we update the parameters θ, ϕ in the direction of the gradient of $\log p_{\theta,\phi}(\mathbf{z}_s|\mathbf{z}_t, y)$ obtained as the sum of the gradients of the unconditional distribution $\log p_\theta(\mathbf{z}_s|\mathbf{z}_t)$ and classifier probability $p_\phi(y|\mathbf{z}_s)$ with respect to the sampled sequence $\mathbf{z}_s$.

$$\nabla_{\mathbf{z}_s} \log p_{\theta,\phi}(\mathbf{z}_s|\mathbf{z}_t, y) = \nabla_{\mathbf{z}_s} \log p_\phi(y|\mathbf{z}_s) + \nabla_{\mathbf{z}_s} \log p_\theta(\mathbf{z}_s|\mathbf{z}_t) \tag{45}$$

After joint training with the classifier and unconditional data distribution, we can sample from the learned conditional distribution $p_{\theta,\phi}(\mathbf{z}_s|\mathbf{z}_t, y)$.

Unlike classifier-free guidance, classifier-based guidance does not require training a generative model from a conditioned dataset. However, the gradient-based strategy for classifier-based guidance is not directly applicable to discrete state spaces due to the lack of a defined gradient. To mimic gradient-based updates to each sampling step, Gruver et al. [26] leveraged iterative guidance steps on continuous latent embeddings for each token before decoding back to a discrete sequence at each time step. However, projecting to and from the continuous and discrete spaces results in inconsistencies in the guided diffusion process, where optimized hidden embeddings do not always map to optimal tokens.

Guidance in the Discrete State Space. To directly guide the diffusion objective in the discrete space, we must compute the optimality of a single discrete reverse transition $\mathbf{z}_s$ against all other possible transitions to maximize the conditional probability $p(y|\mathbf{z}_s, \mathbf{z}_t)$. That is, we need to compute Equation (43) with the denominator expanded to represent all possible transitions from $\mathbf{z}_t$.

$$p_{\theta,\phi}(\mathbf{z}_s|\mathbf{z}_t, y) = \frac{p_\phi(y|\mathbf{z}_s)p_\theta(\mathbf{z}_s|\mathbf{z}_t)}{\sum_{\mathbf{z}'_s} p_\phi(y|\mathbf{z}'_s)p_\theta(\mathbf{z}'_s|\mathbf{z}_t)} \tag{46}$$

However, computing $p_\phi(y|\mathbf{z}'_s)$ for all the possible transitions from state $\mathbf{z}_t$ is computationally intractable. Previous work has bypassed this limitation by approximation. For continuous and differentiable classifier functions $p(y|\mathbf{x}) : \mathbb{R}^{L\times|\mathcal{V}|} \to \mathbb{R}$, we can approximate the denominator using the first-order Taylor expansion given by

$$\log p_\phi(y|\mathbf{z}_s, \mathbf{z}_t) \approx \log p_\phi(y|\mathbf{z}_t) + (\mathbf{z}_s - \mathbf{z}_t)^\top \nabla_{\mathbf{z}} \log p(y|\mathbf{z})|_{\mathbf{z}=\mathbf{z}_t} \tag{47}$$

which approximates the likelihood of observing property y at the slightly denoised state $\mathbf{z}_s = \mathbf{z}_{t-\frac{1}{T}}$ given the known log-probability of observing y for state $\mathbf{z}_t$. This eliminates the need to explicitly sample $\mathbf{z}_s$ for all possible state transitions and compute $\log p_\phi(y|\mathbf{z}_s, \mathbf{z}_t)$.

Digress [76] has used this approximation strategy for guided discrete diffusion on categorical graph generation. Furthermore, Nisonoff et al. [58] uses the Taylor-approximated conditional distribution $\log p_\phi(y|\mathbf{z}_s)$ to adjust the unconditional transition rates $R_t(\mathbf{z}_t, \mathbf{z}_s|y)$ given the unconditional rates $R_t(\mathbf{z}_t, \mathbf{z}_s)$ for predictor-guidance of Continuous-Time Markov Chains (CTMCs) in the discrete state space.

$$R_t(\mathbf{z}_t, \mathbf{z}_s|y) = R_t(\mathbf{z}_t, \mathbf{z}_s)\frac{\log p_\phi(y|\mathbf{z}_s, \mathbf{z}_t)}{\log p_\phi(y|\mathbf{z}_t)} \tag{48}$$

where $R_t(\mathbf{z}_t, \mathbf{z}_s|y)$ is the predictor-adjusted rate of transitioning from state $\mathbf{z}_t$ to state $\mathbf{z}_s$

However, this strategy fails to scale to multiple objective guidance since it would require computing the joint probability over K objectives $p_\phi(y_1, y_2, \ldots, y_K|\mathbf{z}_s, \mathbf{z}_t)$ for some $K > 1$. If all properties are mutually independent, we can factorize the distribution and compute the estimated probability of each objective and take their product $p_\phi(y_1, y_2, \ldots, y_K|\mathbf{z}_s, \mathbf{z}_t) = \prod_{k=1}^{K} p_\phi(y_k|\mathbf{z}_s, \mathbf{z}_t)$. For the majority of multi-objective tasks including therapeutic peptide generation, independence across properties is not a reasonable assumption, and computing the joint distribution is required. Moreover, for objectives that guide toward contradictory optimal rates or transitions, training a model conditioned on these objectives could prevent the model from generating optimal sequences for either objective. Given these limitations, there remains a gap for efficient classifier-based conditioning for discrete diffusion that is robust to multi-objective tasks, which we address in this work.

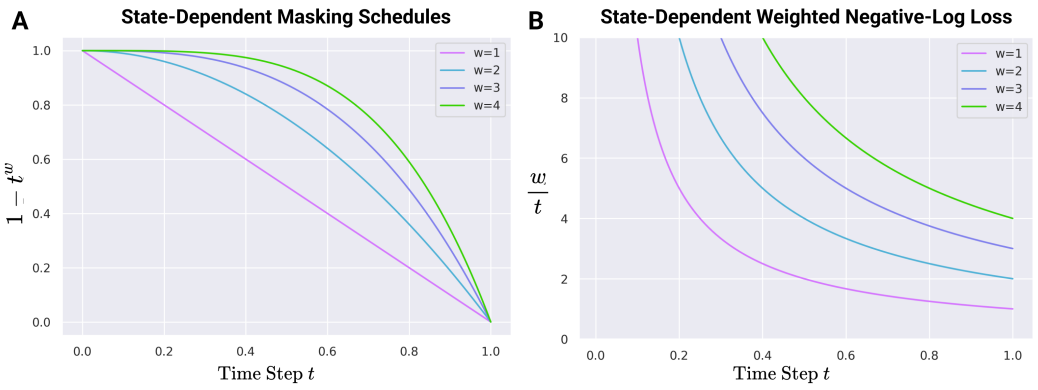


Figure 15: **Plots of bond-dependent masking schedules.** (**A**) The probability of remaining unmasked during the continuous-time forward diffusion process over time t given different values of w as the exponent of the masking schedule $\alpha_t = 1 - t^w$. We use $w = 1$ for non-peptide bond tokens and $w = 3$ for peptide bond tokens, resulting in slower masking of peptide-bond tokens. (**B**) The weight of the negative log-loss for different exponents w in the log-polynomial masking schedule. The weight of the loss is higher for larger w in earlier time steps, which results in a higher penalty for inaccurate predictions of peptide bond tokens compared to other tokens.

D Theoretical Details

D.1 Bond-Dependent Masking Schedule

From Equation (31), we define the continuous-time forward masking probability $1 - \alpha_t$ at time t with $\alpha_t = \exp(-\bar{\sigma}(t))$, where $\bar{\sigma} : [0, 1] \to \mathbb{R}^+$ is the cumulative discrete-time masking schedule. Following Lou et al. [50], we apply a log-linear masking schedule $\bar{\sigma}(t) = -\log(1 - t)$ for the forward diffusion process which has shown to result in the lowest variance in the NELBO loss [68]. Therefore, the continuous-time probability of remaining unmasked at time t is equal to $\alpha_t = \exp\big(-(-\log(1-t))\big) = 1 - t$ and the weight that scales the negative log loss (NLL) is given by $\frac{1}{t}$ by our derivation in Appendix D.3.

For peptide-bond tokens, we alter the masking schedule such that peptide-bonds are masked at a slower rate at earlier time steps by defining a log-polynomial masking schedule $\bar{\sigma}(t) = -\log(1-t^w)$, for some constant exponent $w > 1$. Note that when $w = 1$, the log-polynomial schedule reduces to the log-linear schedule. Therefore, the probability of remaining unmasked becomes $\alpha_t = \big(-(-\log(1 - t^w))\big) = 1 - t^w$ and the weight that scales the negative log loss (NLL) is given by $\frac{w}{t}$ by our derivation in Appendix D.3.

Since $t \in (0, 1]$, the probability that a peptide-bond token remains unmasked at time t is equal to $\alpha_t = 1 - t^w$ which is larger than the log-linear schedule for $w > 1$. Conversely, the probability that a peptide-bond token is masked before t is $1 - \alpha_t = t^w$ which is smaller than the log-linear schedule for $w > 1$. As $t \to 1$, $\alpha_t \to 0$ for both the log-linear and log-polynomial time schedules, which means that both peptide-bond and non-peptide bond tokens will have a high probability of being masked in later times in the forward pass diffusion process.

The NLL of the peptide-bond tokens is weighted more heavily than non-peptide bond tokens for t close to 1. As $t \to 0$, the NLL weight approaches ∞ for all tokens. This biases the reverse diffusion process to unmask peptide bond tokens earlier since it was trained to minimize the loss associated with each unmasking step. As $t \to 0$, the large NLL weight ensures that the final unmasking steps during the reverse diffusion process result in an unmasked sequence that lies within the space of valid peptide SMILES.

D.2 Derivation of Bond-Dependent Reverse Posterior

For a single token, the bond-dependent forward diffusion process is defined by the probability distribution $q(\mathbf{z}_t|\mathbf{x}_0)$ which transforms the clean inputs to sequences with varying degrees of masking

based on a probability distribution $\alpha_t(\mathbf{x}_0)$. We define $\alpha_t(\mathbf{x}_0) : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}$ as a function that takes the clean token encoding $\mathbf{x}_0$ and outputs the probability of remaining unmasked at time t depending on whether $\mathbf{x}_0$ encodes a peptide bond token.

$$q(\mathbf{z}_t|\mathbf{x}_0) = \text{Cat}(\mathbf{z}_t; (\alpha_t(\mathbf{x}_0))\mathbf{x}_0 + (1 - \alpha_t(\mathbf{x}_0))\mathbf{m}) \tag{49}$$

Then, the marginal forward transition from time $s(n) \to t(n)$ is defined as

$$q(\mathbf{z}_{t(n)}|\mathbf{z}_{s(n)}) = \text{Cat}\left(\mathbf{z}_{t(n)}; \left(\frac{\alpha_t(\mathbf{x}_0)}{\alpha_s(\mathbf{x}_0)}\right)\mathbf{x}_0 + \left(\mathbf{1} - \frac{\alpha_t(\mathbf{x}_0)}{\alpha_s(\mathbf{x}_0)}\right)\mathbf{m}\right) \tag{50}$$

In this work, we classify each token into one of two states: peptide-bond tokens and non-peptide-bond tokens, which represent amino acid side chains and modifications. We define a function that generates a mask with values of 1 indicating tokens containing or contained within a peptide bond, and 0 otherwise (Algorithm 7). Let $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$ denote a vector with ones at indices of peptide-bond tokens. For derivation purposes, we let $\mathbf{b}^\top \mathbf{x}_0^{(\ell)} = 1$ and $\mathbf{x}_0^{(\ell)} = \mathbf{b}$ when a token at position ℓ is a peptide bond token. Note that $\mathbf{b}$ is defined differently depending on the context of the token in the full sequence which is handled by the BONDMASK function. Then, we have $\alpha_t(\mathbf{x}_0) = \big(\mathbf{1} - \mathbf{b}^\top \mathbf{x}_0\big)(1 - t) + \mathbf{b}^\top \mathbf{x}_0(1 - t^w)$ or equivalently we can write

$$\alpha_t(\mathbf{x}_0) = \begin{cases} 1 - t^w & \mathbf{x}_0 = \mathbf{b} \\ 1 - t & \mathbf{x}_0 \neq \mathbf{b} \end{cases} \tag{51}$$

By Bayes' rule, the general state-independent form of the true reverse posterior is given by

$$\begin{aligned} q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0) &= \frac{q(\mathbf{z}_t|\mathbf{z}_s)q(\mathbf{z}_s|\mathbf{x}_0)}{q(\mathbf{z}_t|\mathbf{x}_0)} \\ &= \frac{\left[\left(\frac{\alpha_t}{\alpha_s}\right)\mathbf{z}_s^\top \mathbf{z}_t + \left(1 - \frac{\alpha_t}{\alpha_s}\right)\mathbf{m}^\top \mathbf{z}_t\right]\left[\alpha_s \mathbf{x}_0^\top \mathbf{z}_t + (1 - \alpha_s)\,\mathbf{m}^\top \mathbf{z}_t\right]}{\left[\alpha_t \mathbf{x}_0^\top \mathbf{z}_t + (1 - \alpha_t)\,\mathbf{m}^\top \mathbf{z}_t\right]} \end{aligned} \tag{52}$$

With bond-dependent masking, the value of $\alpha_t(\mathbf{x}_0)$ and $\alpha_s(\mathbf{x}_0)$ depend on the state of $\mathbf{x}_0$, so the bond-dependent reverse posterior becomes

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0) = \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{z}_t + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{m}^\top \mathbf{z}_t\right]\left[\alpha_s(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{z}_t + (1 - \alpha_s(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{z}_t\right]}{\left[\alpha_t(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{z}_t + (1 - \alpha_t(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{z}_t\right]} \tag{53}$$

When $\mathbf{z}_t = \mathbf{x}_0$, the true reverse posterior simplifies to

$$\begin{aligned} q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{x}_0, \mathbf{x}_0) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{x}_0 + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{m}^\top \mathbf{x}_0\right]\left[\alpha_s(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{x}_0 + (1 - \alpha_s(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{x}_0\right]}{\left[\alpha_t(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{x}_0 + (1 - \alpha_t(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{x}_0\right]} \\ &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{x}_0\right]\left[\alpha_s(\mathbf{x}_0)\right]}{\alpha_t(\mathbf{x}_0)} \end{aligned} \tag{54}$$

When $\mathbf{z}_s \neq \mathbf{x}_0$, $\mathbf{z}_s^\top \mathbf{x}_0 = 0$ so $q(\mathbf{z}_s \neq \mathbf{x}_0|\mathbf{z}_t = \mathbf{x}_0, \mathbf{x}_0) = 0$. When $\mathbf{z}_s = \mathbf{x}_0$, we have

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{x}_0|\mathbf{z}_t = \mathbf{x}_0, \mathbf{x}_0) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{x}_0)}{\alpha_s(\mathbf{x}_0)}\right)\mathbf{x}_0^\top \mathbf{x}_0\right]\left[\alpha_s(\mathbf{x}_0)\right]}{\alpha_t(\mathbf{x}_0)} \\ &= \left(\frac{\alpha_t(\mathbf{x}_0)}{\alpha_s(\mathbf{x}_0)}\right)\left(\frac{\alpha_s(\mathbf{x}_0)}{\alpha_t(\mathbf{x}_0)}\right) \\ &= 1 \end{aligned}$$

which means that $\mathbf{z}_t$ remains unchanged after unmasking. This supports the carry-over unmasking scheme which explicitly sets the probability of changing an unmasked token equal to $-\infty$.

In the forward diffusion process, a token either remains unchanged or is masked, so the only other case we need to consider is $\mathbf{z}_t = \mathbf{m}$. Since the masking schedule differs only when the ground truth

token is a peptide bond token, or $\mathbf{x}_0 = \mathbf{b}$, we can consider two cases: first, when $\mathbf{x}_0 = \mathbf{b}$ and second, when $\mathbf{x}_0 \neq \mathbf{b}$.

Case 1. Consider the case when $\mathbf{x}_0 = \mathbf{b}$ or the ground truth token $\mathbf{x}_0$ is a peptide-bond token. From our modified masking schedule, we have $\alpha_t(\mathbf{b}) = 1 - t^w$. Therefore, we can write the probability distribution for unmasking a peptide-bond token as

$$\begin{aligned} q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 = \mathbf{b}) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{m}^\top \mathbf{m}\right]\left[\alpha_s(\mathbf{b})\mathbf{b}^\top \mathbf{z}_s + (1 - \alpha_s(\mathbf{b}))\,\mathbf{m}^\top \mathbf{z}_s\right]}{\left[\alpha_t(\mathbf{b})\mathbf{b}^\top \mathbf{m} + (1 - \alpha_t(\mathbf{b}))\,\mathbf{m}^\top \mathbf{m}\right]} \\ &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\right]\left[\alpha_s(\mathbf{b})\mathbf{b}^\top \mathbf{z}_s + (1 - \alpha_s(\mathbf{b}))\,\mathbf{m}^\top \mathbf{z}_s\right]}{(1 - \alpha_t(\mathbf{b}))} \end{aligned} \tag{55}$$

The probability of transitioning from a masked state to a peptide-bond token is simplified to

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{b}|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 = \mathbf{b}) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{b})}{\alpha_s(\mathbf{b})}\right)\mathbf{b}^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{b})}{\alpha_s(\mathbf{b})}\right)\right]\left[\alpha_s(\mathbf{b})\mathbf{b}^\top \mathbf{b} + (1 - \alpha_s(\mathbf{b}))\,\mathbf{m}^\top \mathbf{b}\right]}{(1 - \alpha_t(\mathbf{b}))} \\ &= \frac{\left(1 - \frac{1-t^w}{1-s^w}\right)(1 - s^w)}{(1 - (1 - t^w))} \\ &= \frac{\left(\frac{1-s^w-1+t^w}{1-s^w}\right)(1 - s^w)}{t^w} \\ &= \frac{t^w - s^w}{t^w} \\ &= 1 - \frac{s^w}{t^w} \end{aligned} \tag{56}$$

The probability of remaining in a masked state is

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 = \mathbf{b}) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{m})}{\alpha_s(\mathbf{m})}\right)\mathbf{m}^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{m})}{\alpha_s(\mathbf{m})}\right)\right]\left[\alpha_s(\mathbf{b})\mathbf{b}^\top \mathbf{m} + (1 - \alpha_s(\mathbf{b}))\,\mathbf{m}^\top \mathbf{m}\right]}{(1 - \alpha_t(\mathbf{b}))} \\ &= \frac{\left[\left(\frac{\alpha_t(\mathbf{m})}{\alpha_s(\mathbf{m})}\right) + \left(1 - \frac{\alpha_t(\mathbf{m})}{\alpha_s(\mathbf{m})}\right)\right](1 - \alpha_s(\mathbf{b}))}{(1 - \alpha_t(\mathbf{b}))} \\ &= \frac{1 - \alpha_s(\mathbf{b})}{1 - \alpha_t(\mathbf{m})} \\ &= \frac{1 - (1 - s^w)}{1 - (1 - t^w)} \\ &= \frac{s^w}{t^w} \end{aligned} \tag{57}$$

which aligns with the constraint that $\mathbf{z}_t \in \{\mathbf{m}, \mathbf{x}_0\}$ in the forward diffusion process.

Case 2: Consider the case when $\mathbf{x}_0 \neq \mathbf{b}$ or the ground truth token $\mathbf{x}_0$ is not a peptide-bond token. From the baseline log-linear masking schedule, we have $\vec{\alpha}_t^\top \mathbf{x}_0 = 1 - t$. Therefore, we can write the probability distribution for unmasking a peptide-bond token as

$$\begin{aligned} q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 \neq \mathbf{b}) &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{m}^\top \mathbf{m}\right]\left[\alpha_s(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{m} + (1 - \alpha_s(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{m}\right]}{\left[\alpha_t(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{m} + (1 - \alpha_t(\mathbf{x}_0))\,\mathbf{m}^\top \mathbf{m}\right]} \\ &= \frac{\left[\left(\frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\mathbf{z}_s^\top \mathbf{m} + \left(1 - \frac{\alpha_t(\mathbf{z}_s)}{\alpha_s(\mathbf{z}_s)}\right)\right]\left[\alpha_s(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{m} + (1 - \alpha_s(\mathbf{x}_0))\right]}{\left[\alpha_t(\mathbf{x}_0)\mathbf{x}_0^\top \mathbf{m} + (1 - \alpha_t(\mathbf{x}_0))\right]} \end{aligned} \tag{58}$$

With similar steps to Case 1, the probability of transitioning from a masked state to a non-peptide-bond token is given by

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{x}_0 | \mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 \neq \mathbf{b}) &= \frac{\left(1 - \frac{\alpha_t(\mathbf{x}_0)}{\alpha_s(\mathbf{x}_0)}\right)(1 - \alpha_s(\mathbf{x}_0))}{(1 - \alpha_t(\mathbf{x}_0))} \\ &= \frac{\left(1 - \frac{1-t}{1-s}\right)(1 - (1-s))}{(1 - (1-t))} \\ &= \frac{t-s}{t} \\ &= 1 - \frac{s}{t} \end{aligned} \tag{59}$$

It follows that the probability of remaining in a masked state in the reverse process is

$$q(\mathbf{z}_s = \mathbf{m} | \mathbf{z}_t = \mathbf{m}, \mathbf{x}_0 \neq \mathbf{b}) = \frac{s}{t} \tag{60}$$

This demonstrates that the probability of remaining in a masked state when $\mathbf{x}_0 = \mathbf{b}$ is smaller than when $\mathbf{x}_0 \neq \mathbf{b}$ since taking the exponent of a fraction results in a smaller value. So we have $\frac{s^w}{t^w} < \frac{s}{t}$ for $w > 1$.

Combining Equations (57) and (60) we get the following distribution for the case when $\mathbf{z}_t = \mathbf{m}$ and $\mathbf{z}_s = \mathbf{m}$

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{m} | \mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) &= \left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b}^\top \mathbf{x}_0 + \frac{s}{t} \\ &= \left(\frac{s^w}{t^w}\mathbf{b} - \frac{s}{t}\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \\ &= \left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \end{aligned} \tag{61}$$

Similarly, combining (56) and (59) we get the following distribution for the case when $\mathbf{z}_t = \mathbf{m}$ and $\mathbf{z}_s \neq \mathbf{m}$ or equivalently $\mathbf{z}_s = \mathbf{x}_0$.

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{x}_0 | \mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) &= \left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b}^\top \mathbf{x}_0 + \left(1 - \frac{s}{t}\right) \\ &= \left(\frac{s}{t}\mathbf{b} - \frac{s^w}{t^w}\mathbf{b} + \mathbf{1} - \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \\ &= \left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \end{aligned} \tag{62}$$

Now, we can write the true reverse posterior as

$$q(\mathbf{z}_s | \mathbf{z}_t, \mathbf{x}_0) = \begin{cases} \left\langle \left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}, \mathbf{x}_0 \right\rangle \mathbf{x}_0 + \left\langle \left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}, \mathbf{x}_0 \right\rangle \mathbf{m} & \mathbf{z}_t = \mathbf{m} \\ \mathbf{z}_t & \mathbf{z}_t \neq \mathbf{m} \end{cases} \tag{63}$$

Therefore, we get the following expression for the parameterized reverse posterior

$$p_\theta(\mathbf{z}_s | \mathbf{z}_t) = \begin{cases} \left\langle \left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}, \mathbf{x}_\theta(\mathbf{z}_t, t) \right\rangle \mathbf{z}_s + \left\langle \left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}, \mathbf{x}_\theta(\mathbf{z}_t, t) \right\rangle \mathbf{m} & \mathbf{z}_t = \mathbf{m} \\ \mathbf{z}_t & \mathbf{z}_t \neq \mathbf{m} \end{cases} \tag{64}$$

D.3 Derivation of Bond-Dependent NELBO Loss

The diffusion objective in its general form is given by

$$\begin{aligned} \mathcal{L}_{\text{NELBO}} &= \sum_{n=1}^{T-1} \mathbb{E}_{q(\mathbf{z}_{t(n)}|\mathbf{x}_0)} \left[\text{KL}\left(q(\mathbf{z}_{s(n)} | \mathbf{z}_{t(n)}, \mathbf{x}_0) || p_\theta(\mathbf{z}_{s(n)} | \mathbf{z}_{t(n)}) \right) \right] \\ &= \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)} \left[T \cdot \text{KL}\left(q(\mathbf{z}_s | \mathbf{z}_t, \mathbf{x}_0) \middle| \middle| p_\theta(\mathbf{z}_s | \mathbf{z}_t) \right) \right] \end{aligned} \tag{65}$$

First, we will derive an expression for the bond-dependent KL-divergence, which measures the difference between the learned reverse posterior $q\big(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_\theta(\mathbf{z}_t, t)\big)$ and the true reverse posterior $q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)$ conditioned on the training distribution $\mathbf{x}_0$.

$$
\begin{aligned}
&\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) \\
&= \sum_{\mathbf{z}_s=\mathbf{e}_k} q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) \log \frac{q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0)}{p_\theta(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m})} \\
&= \sum_{\mathbf{z}_s\in\{\mathbf{x}_0,\mathbf{m}\}} q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) \log \frac{q(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0)}{p_\theta(\mathbf{z}_s|\mathbf{z}_t = \mathbf{m})} \\
&= q(\mathbf{z}_s = \mathbf{x}_0|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) \log \frac{q(\mathbf{z}_s = \mathbf{x}_0|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0)}{p_\theta(\mathbf{z}_s = \mathbf{x}_0|\mathbf{z}_t = \mathbf{m})} \\
&+ q(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0) \log \frac{q(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t = \mathbf{m}, \mathbf{x}_0)}{p_\theta(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t = \mathbf{m})} \\
&= \left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \log \frac{\left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_0}{\left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)} \\
&+ \left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_0 \log \frac{\left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_0}{\left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)}
\end{aligned} \tag{66}
$$

In the case where the true token $\mathbf{x}_0 = \mathbf{b}$, we can simplify to

$$
\begin{aligned}
\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) &= \left(\frac{s}{t} - \frac{s^w}{t^w} + 1 - \frac{s}{t}\right) \log \frac{\left(\frac{s^w}{t^w} - \frac{s}{t} + \frac{s}{t}\right)\mathbf{x}_0^\top \mathbf{x}_0}{\left(\frac{s^w}{t^w} - \frac{s}{t} + \frac{s}{t}\right)\mathbf{x}_0^\top \mathbf{x}_\theta(\mathbf{z}_t, t)} \\
&= -\left(1 - \frac{s^w}{t^w}\right) \log\left(\mathbf{x}_0^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\right) \\
&= -\left(\frac{t^w - s^w}{t^w}\right) \log\langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle
\end{aligned} \tag{67}
$$

Substituting $s = t - \frac{1}{T}$, we can simplify s^w to

$$
\begin{aligned}
s^w &= \left(t - \frac{1}{T}\right)^w \\
&= \left[t\left(1 - \frac{1}{tT}\right)\right]^w \\
&= t^w \left(1 - \frac{1}{tT}\right)^w] \\
&= t^w \left(1 - \frac{w}{tT} + o\left(\frac{1}{T^2}\right)\right) && ((1+x)^w = 1 + wx + o(x^2)) \\
&= t^w - \frac{wt^{w-1}}{T} + t^w o\left(\frac{1}{T^2}\right)
\end{aligned} \tag{68}
$$

where $o\left(\frac{1}{T^2}\right)$ denotes higher order terms that grow slower than $\frac{1}{T^2}$.

Now, we can write

$$
\begin{aligned}
\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) &= -\left(\frac{t^w - \left(t^w - \frac{wt^{w-1}}{T} + t^w o\left(\frac{1}{T^2}\right)\right)}{t^w}\right) \log\langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&= -\left(\frac{\frac{wt^{w-1}}{T} - t^w o\left(\frac{1}{T^2}\right)}{t^w}\right) \log\langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&= -\left(\frac{w}{tT} - o\left(\frac{1}{T^2}\right)\right) \log\langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle
\end{aligned} \tag{69}
$$

In the case where the true token $\mathbf{x}_0 \neq \mathbf{b}$, we can simplify to

$$\begin{aligned}
\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) &= \left(1 - \frac{s}{t}\right) \log \frac{\left(1 - \frac{s}{t}\right) \mathbf{x}_0^\top \mathbf{x}_0}{\left(1 - \frac{s}{t}\right) \mathbf{x}_0^\top \mathbf{x}_\theta(\mathbf{z}_t, t)} \\
&= -\left(1 - \frac{s}{t}\right) \log \left(\mathbf{x}_0^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\right) \\
&= -\left(\frac{t-s}{t}\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle
\end{aligned} \tag{70}$$

Similarly, substituting $s = t - \frac{1}{T}$, we have

$$\begin{aligned}
\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) &= -\left(\frac{t - \left(t - \frac{1}{T}\right)}{t}\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&= -\frac{1}{tT} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle
\end{aligned} \tag{71}$$

Now, we can combine the two cases using the indicator functions $\mathbf{1}[\mathbf{x}_0 = \mathbf{b}]$ that evaluates to 1 when $\mathbf{x}_0 = \mathbf{b}$ and 0 otherwise and $\mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}]$ that evaluates to 1 when $\mathbf{x}_0 \neq \mathbf{b}$ and 0 otherwise. Since this definition of KL divergence is only applicable when $\mathbf{z}_t = \mathbf{m}$, we have

$$\begin{aligned}
&\mathrm{KL}(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0)||p_\theta(\mathbf{z}_s|\mathbf{z}_t)) \\
&= \left[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \left(\frac{w}{tT} - o\left(\frac{1}{T^2}\right)\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{1}{tT} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \right]
\end{aligned} \tag{72}$$

Substituting this back into the equation for the discrete-time diffusion loss, we get

$$\begin{aligned}
\mathcal{L}_{\text{NELBO}} &= \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)} \left[T \cdot \mathrm{KL}\Big(q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_0) \big| \big| p_\theta(\mathbf{z}_s|\mathbf{z}_t) \Big) \right] \\
&= \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)} T \cdot \Bigg[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \left(\frac{w}{tT} - o\left(\frac{1}{T^2}\right)\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&\qquad - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{1}{tT} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \Bigg] \\
&= \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)} \Bigg[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \left(\frac{wT}{tT} - To\left(\frac{1}{T^2}\right)\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&\qquad - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{T}{tT} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \Bigg] \\
&= \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{z}_t|\mathbf{x}_0)} \Bigg[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \left(\frac{w}{t} - To\left(\frac{1}{T^2}\right)\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&\qquad - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{1}{t} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \Bigg]
\end{aligned} \tag{73}$$

Finally, taking the limit as $T \to \infty$, the higher-order term $\lim_{T\to\infty} To\left(\frac{1}{T^2}\right) = 0$ and we get

$$\begin{aligned}
\mathcal{L}_{\text{NELBO}}^{\infty} &= \lim_{T\to\infty} \mathcal{L}_{\text{NELBO}} \\
&= \lim_{T\to\infty} \mathbb{E}_{t \in \{\frac{1}{T}, \frac{2}{T}, \ldots, 1\}} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \Bigg[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \left(\frac{w}{t} - To\left(\frac{1}{T^2}\right)\right) \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&\qquad - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{1}{t} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \Bigg] \\
&= \mathbb{E}_{t \sim \mathcal{U}(0,1]} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \Bigg[-\mathbf{1}[\mathbf{x}_0 = \mathbf{b}] \frac{w}{t} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \\
&\qquad - \mathbf{1}[\mathbf{x}_0 \neq \mathbf{b}] \frac{1}{t} \log \langle \mathbf{x}_0, \mathbf{x}_\theta(\mathbf{z}_t, t)\rangle \Bigg]
\end{aligned} \tag{74}$$

which is the continuous-time NELBO loss for a single token. Therefore, the loss across a sequence of L tokens denoted as $\mathbf{x}_0^{(\ell)}$, we have

$$\mathcal{L}_{\text{NELBO}}^{\infty} = \mathbb{E}_{t\sim\mathcal{U}(0,1]}\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\Bigg[-\sum_{\ell:\mathbf{x}_0^{(\ell)}=\mathbf{b}}\frac{w}{t}\log\langle\mathbf{x}_0,\mathbf{x}_\theta(\mathbf{z}_t,t)\rangle - \sum_{\ell:\mathbf{x}_0^{(\ell)}\neq\mathbf{b}}\frac{1}{t}\log\langle\mathbf{x}_0,\mathbf{x}_\theta(\mathbf{z}_t,t)\rangle\Bigg] \tag{75}$$

which proves the loss defined in (9).

D.4 Gradient Flow of Invalid Loss

In this section, we show that the penalty for invalid token samples through the `argmax` function on predicted logits can be effectively backpropagated through the model parameters via our `softmax` scaling strategy. Here, we will denote the predicted probability for the token $k = \arg\max_j \left(\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\right)$ with the highest probability as $x_{\theta,k}^{(\ell)}$ and all remaining token probabilities as $x_{\theta,j}^{(\ell)}$ for $j = [1\ldots|\mathcal{V}|]$.

First, we define the softmax function as

$$\text{SM}\big(x_{\theta,k}^{(\ell)}\big) = \frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})} \tag{76}$$

The partial derivative of the softmax probability x_θ^j for every token j is given by equation

$$\frac{\partial}{\partial x_{\theta,j}^{(\ell)}}\left(\frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right) = \frac{\left(\frac{\partial}{\partial x_{\theta,j}^{(\ell)}}\exp(x_{\theta,k}^{(\ell)})\right)\left(\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})\right) - \left(\frac{\partial}{\partial x_{\theta,j}^{(\ell)}}\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})\right)\left(\exp(x_{\theta,k}^{(\ell)})\right)}{\left(\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})\right)^2} \tag{77}$$

Therefore, we have two cases for the derivative: first, the derivative with respect to $x_{\theta,k}^{(\ell)}$ which denotes the predicted probability for the token that was sampled, and second, the derivative with respect to $x_{\theta,j}^{(\ell)}$ for $j \neq k$ which denotes the predicted probabilities for all remaining tokens.

For the first case when $j = k$, the partial derivative simplifies to

$$\begin{aligned}
\frac{\partial}{\partial x_{\theta,k}^{(\ell)}}\left(\frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right) &= \frac{\exp(x_{\theta,k}^{(\ell)})\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)}) - \exp(x_{\theta,k}^{(\ell)})\exp(x_{\theta,k}^{(\ell)})}{\left(\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})\right)^2} \\
&= \left(\frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right)\left(\frac{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)}) - \exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right) \\
&= \text{SM}(x_{\theta,k}^{(\ell)})\left(1 - \text{SM}(x_{\theta,k}^{(\ell)})\right)
\end{aligned} \tag{78}$$

For all $j \neq k$, the derivative simplifies to

$$\begin{aligned}
\frac{\partial}{\partial x_{\theta,j}^{(\ell)}}\left(\frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{K}\exp(x_{\theta,j}^{(\ell)})}\right) &= \frac{0 - \exp(x_{\theta,j}^{(\ell)})\exp(x_{\theta,k}^{(\ell)})}{\left(\sum_{j=1}^{K}\exp(x_{\theta,j}^{(\ell)})\right)^2} \\
&= -\left(\frac{\exp(x_{\theta,j}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right)\left(\frac{\exp(x_{\theta,k}^{(\ell)})}{\sum_{j=1}^{|\mathcal{V}|}\exp(x_{\theta,j}^{(\ell)})}\right) \\
&= -\text{SM}(x_{\theta,j}^{(\ell)})\text{SM}(x_{\theta,k}^{(\ell)})
\end{aligned} \tag{79}$$

The parameters θ are updated such that the predicted probability of sampling the token ℓ with `argmax` probability $x_{\theta,k}^{(\ell)}$ which resulted in an invalid peptide SMILES sample is reduced. The gradient update is minimized for predicted probabilities near 0 and 1, suggesting that the loss function

pushes the model towards higher confidence predictions from uncertain predictions to minimize invalid sampling.

$$x'^{(\ell)}_{\theta,k} \leftarrow x^{(\ell)}_{\theta,k} - \eta \cdot \text{SM}(x^{(\ell)}_{\theta,k}) \left(1 - \text{SM}(x^{(\ell)}_{\theta,k})\right) \tag{80}$$

where η is the learning rate.

In contrast, the parameters of the remaining tokens $x^{(\ell)}_{\theta,j}$ are updated so that the predicted probability of sampling the other tokens increases proportionally to their original softmax probabilities. This prevents extreme changes in the predicted probabilities of the remaining tokens and ensures that the token distribution remains relatively consistent with the previous iteration.

$$x'^{(\ell)}_{\theta,j} \leftarrow x^{(\ell)}_{\theta,j} + \eta \cdot \text{SM}(x^{(\ell)}_{\theta,j})\text{SM}(x^{(\ell)}_{\theta,k}) \tag{81}$$

Here, we show our invalid loss effectively updates parameters to reduce the position-specific token probabilities that result in invalid sequence samplings and push the model predictions toward other high-likelihood tokens.

E Model Architectures and Training Details

E.1 RoFormer Architecture Details

To predict the token probabilities at each reverse step $\mathbf{x}_\theta(\mathbf{z}_t, t)$, we trained a RoFormer model [74] that leverages rotary positional embeddings (RoPE) robust to varying input lengths and long-range dependencies between tokens. The specific hyperparameters of our model are given below.

Table 8: Roformer Architecture Hyperparameters

Hyperparameter	**PepTune**
Input Dimension	581 (vocab size)
Hidden Dimension	768
Intermediate Dimension	3072
Number of Layers	8
Attention Heads	8
Max Positional Embeddings	1035
Hidden and Attention Dropout Probability	0.1

E.2 PDB Docking Structures

Table 9: PDB structures used for docking.

Protein	**PDB**
GFAP	6A9P
TfR	3KAS
GLP-1R	3C5T
AMHR2	7L0J
GLAST	5LM4
NCAM1	2HAZ
RBX1	1LDJ

E.3 Target-Binding Prediction Model

We trained a multi-head cross-attention network with ESM-2 [49] protein sequence embeddings and PeptideCLM [20] peptide SMILES embeddings for the target binding affinity model. We trained on 1806 sequences from the PepLand [91] canonical and non-canonical binding datasets containing the

protein-target sequence, peptide SMILES sequence, and the experimentally-validated $K_d/K_i/IC50$ binding affinity score. After training for 50 epochs, the regression model achieved a strong Spearman correlation coefficient of 0.949 on the test dataset and 0.633 on the validation dataset.

Table 10: Target-Binding Affinity Predictor

Layers	**Protein Dimension**	**Peptide Dimension**
Embedding Module	1280	768
Linear Layer	512	512
Layer Norm	512	512
Cross-Attention $\times 3$		
Multi-Head Attention ($h = 8$)	512	512
Linear Layer	2048	2048
ReLU	2048	2048
Dropout	2048	2048
Linear Layer	512	512
Shared Prediction Head		
Linear Layer	1024	
ReLU	1024	
Dropout	1024	
Regression Head	1	
Classification Head	3	

E.4 Boosted Trees for Peptide SMILES Property Prediction

Here, we present the details on the training and hyperparameters of our trained XGBoost boosted tree regression model for membrane permeability prediction and the boosted tree binary classification model for solubility, hemolysis, and non-fouling.

Table 11: XGBoost Hyperparameters for Classification and Regression

Classification Hyperparameters		**Regression Hyperparameters**	
Hyperparameter	**Value/Range**	**Hyperparameter**	**Value/Range**
Objective	`binary:logistic`	Objective	`reg:squarederror`
Lambda	$[1e{-}8, 10.0]$	Lambda	$[0.1, 10.0]$ (log scale)
Alpha	$[1e{-}8, 10.0]$	Alpha	$[0.1, 10.0]$ (log scale)
Colsample by Tree	$[0.1, 1.0]$	Gamma	$[0, 5]$
Subsample	$[0.1, 1.0]$	Colsample by Tree	$[0.5, 1.0]$
Learning Rate	$[0.01, 0.3]$	Subsample	$[0.6, 0.9]$
Max Depth	$[2, 30]$	Learning Rate	$[1e{-}5, 0.1]$
Min Child Weight	$[1, 20]$	Max Depth	$[2, 30]$
Tree Method	`hist`	Min Child Weight	$[1, 20]$
		Tree Method	`hist`
		Scale Pos Weight	$[0.5, 10.0]$ (log scale)

E.5 Evaluation Metrics

Validity is defined as the fraction of peptide SMILES that pass our SMILES2PEPTIDE filter (Algorithm 8), indicating that it translates to a synthesizable peptide.

Uniqueness is defined as the fraction of mutually distinct peptide SMILES.

Diversity is defined as one minus the average Tanimoto similarity between the Morgan fingerprints of every pair of generated sequences, which measures the similarity in structure across generated

peptides.

$$\text{Diversity} = 1 - \frac{1}{\binom{N_{\text{generated}}}{2}} \sum_{i,j} \frac{\mathbf{f}(\mathbf{x}_i) \cdot \mathbf{f}(\mathbf{x}_j)}{|\mathbf{f}(\mathbf{x}_i)| + |\mathbf{f}(\mathbf{x}_j)| - \mathbf{f}(\mathbf{x}_i) \cdot \mathbf{f}(\mathbf{x}_j)} \tag{82}$$

where $\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}(\mathbf{x}_j)$ are the 2048-dimensional Morgan fingerprint with radius 3 for a pair of generated sequences $\mathbf{x}_i$ and $\mathbf{x}_j$.

Similarity to Nearest Neighbor (SNN) is defined as the maximum Tanimoto similarity between a generated sequence $\mathbf{x}_i$ with a sequence in the dataset $\tilde{\mathbf{x}}_j$.

$$\text{SNN} = \max_{j \in |\mathcal{D}|} \left(\frac{\mathbf{f}(\mathbf{x}_i) \cdot \mathbf{f}(\tilde{\mathbf{x}}_j)}{|\mathbf{f}(\mathbf{x}_i)| + |\mathbf{f}(\tilde{\mathbf{x}}_j)| - \mathbf{f}(\mathbf{x}_i) \cdot \mathbf{f}(\tilde{\mathbf{x}}_j)} \right) \tag{83}$$

Randomness is defined as the Shannon Entropy [48] on tokenized sequences given as:

$$E = -\sum_{i}^{L} p_i \log_2(p_i) \tag{84}$$

where p_i is the probability of i-th unique token divided by the total number of tokens L in the sequence.

KL-Divergence is defined as the divergence between the token distribution in the generated peptide SMILES p_i and the token distribution in the training data.

$$\text{KL}(P||Q) = \sum_{i \in \mathcal{V}} \begin{cases} p_i \log_2(\frac{p_i}{q_i}) & \text{if } q_i > 0 \\ p_i \log_2(\frac{p_i}{10^{-9}}) & \text{if } q_i = 0 \end{cases} \tag{85}$$

where p_i is the probability of token i in the training data, and q_i is the probability of token i in the generated data.

F Further Experiments

F.1 Case Study for Time-Dependent Multi-Objective Guidance

Some properties of peptides require more intense guidance towards specific structural or sub-structural features, while others may only require small changes in the side chain composition or non-natural modifications. To enable the prioritization of properties during guidance, we introduce a time-dependent multi-objective guidance strategy that guides the generation based on only a subset of properties depending on the current iteration number of the MCTS search. To achieve this, we define a K-dimensional vector $\mathbf{i} = [i_1, i_2, \ldots, i_K]$ where each i_k is the iteration number to begin guidance for the kth objective. Properties where $i_k = 0$ are used to guide all iterations, whereas properties where $i_k > 1$ are used to guide only the iterations from $i_k \to N_{\text{iter}}$.

Our time-dependent guidance operates as follows. During the expansion and rollout steps on iteration i, the rolled-out child sequences $\mathbf{x}_{s,i}$ that are non-dominated across the sub-vector of property scores $\mathbf{s}_i = [s_k \mid i_k \leq i \leq N_{\text{iter}}]$ dependent on the iteration i is added to the Pareto-optimal set $\mathcal{P}^*$. Therefore, $\mathbf{x}_s$ does not need to be non-dominated in the properties k where $i_k > i$. Similarly, only the sequences $\mathbf{x}^* \in \mathcal{P}^*$ that become dominated when adding $\mathbf{x}_s$ in the subset of properties represented in $\mathbf{s}_i(\mathbf{x}^*)$ are removed from $\mathcal{P}^*$.

$$\mathcal{P}'^* = \mathcal{P}^* \cup \left\{ (\mathbf{z}_s, \mathbf{s}(\mathbf{x}_s)) \mid \forall \mathbf{x}^* \in \mathcal{P}^* \;\; \mathbf{s}_i(\mathbf{x}_s) \succeq \mathbf{s}_i(\mathbf{x}^*) \right\} \tag{86}$$

$$\mathcal{P}'^* = \mathcal{P}^* \setminus \left\{ \mathbf{x}^* \mid \exists \mathbf{x}_s \in \text{children}(\mathbf{z}_t) \text{ s.t. } \mathbf{s}_i(\mathbf{x}_s) \succ \mathbf{s}_i(\mathbf{x}^*) \right\} \tag{87}$$

Then, during selection, we only consider the cumulative rewards $\mathbf{W}_i = [W_k \mid i_k \leq i \leq N_{\text{iter}}]$ for the properties where i_k s.t. $i_k \leq i \leq N_{\text{iter}}$ when computing the selection score vector $\mathbf{U}_i$ to form the Pareto-optimal selection set.

$$\mathcal{P}'^*_{\text{select}} = \mathcal{P}^*_{\text{select}} \cup \left\{ \mathbf{z}_s \mid \forall \mathbf{x}^* \in \mathcal{P}^*_{\text{select}} \;\; \mathbf{U}_i(\mathbf{z}_t, \mathbf{z}_s) \succeq \mathbf{U}_i(\mathbf{z}_t, \mathbf{z}^*) \right\} \tag{88}$$

$$\mathcal{P}'^*_{\text{select}} = \mathcal{P}^*_{\text{select}} \setminus \left\{ \mathbf{z}^* \mid \exists \mathbf{z}_s \in \text{children}(\mathbf{z}_t) \text{ s.t. } \mathbf{U}_i(\mathbf{z}_t, \mathbf{z}_s) \succ \mathbf{U}_i(\mathbf{z}_t, \mathbf{z}^*) \right\} \tag{89}$$

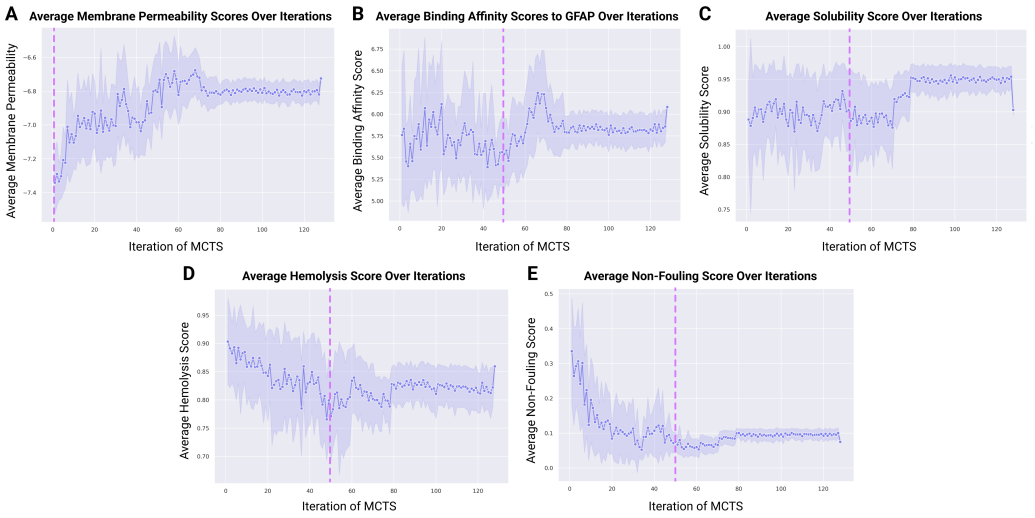


Figure 16: **Time-Dependent Multi-Objective Guidance.** (**A**) Plot of average membrane permeability score for 50 sampled sequences in the expansion and rollout step over iterations where the MCTS search is conditioned on permeability for all iterations. (**B**) Plot of average predicted binding affinity score to GFAP over iterations when conditioned starting from epoch 50. (**C**, **D**, **E**) Plot of average predicted solubility, hemolysis, and non-fouling scores over iterations when conditioned starting from epoch 50. Pink dotted lines denote the iteration where the MCTS search began conditioning on the property

Finally, we select the next node $\mathbf{z}_s \sim \mathcal{P}'^{*}_{\text{select}}$ uniformly at random from the Pareto-optimal selection set.

To test this strategy, we generated 100 peptides conditioned only on membrane permeability for the first 50 iterations since we found it as the most challenging property to optimize. Then we conditioned all properties including membrane permeability, binding affinity to GFAP, solubility, hemolysis, and non-fouling. We show that during the first 50 iterations, all properties except membrane permeability show relatively constant average scores, whereas the permeability score increased (Figure 16). Then, after the 50 iteration mark, GFAP binding affinity and solubility curves increased significantly while the hemolysis and non-fouling curves increased slightly for the remainder of the iterations (Figure 16). Although all the results in this paper leverage peptides without time-dependent guidance, this serves as a proof of concept for future experiments varying the start times across properties to refine certain properties at later time steps where the generated sequences are already constrained to specific predefined substructures.

F.2 Ablation Studies

In this section, we report the performance of our multi-objective MCTS-guided discrete diffusion model over different hyperparameter settings. Specifically, we discuss the effects of changing the number of expanded children nodes, the number of iterations, and the tokenization scheme.

Number of Children. The number of children M is the hyperparameter that determines the batch size during the expansion step of MCTS. A small number of expanded nodes would limit the degree of exploration and number of generated sequences for evaluation at each iteration. If the initial iterations resulted in sub-optimal unmasking steps for all children, this could prevent the algorithm from discovering a local or global optimum across objectives. Suppose the number of children is too large. In that case, this can result in a lack of diversity if several sequences from the same expansion step are added to the Pareto-optimal set given their sequence similarity, leading to similar property scores. A large M also slows down runtime significantly. To determine a value for M within the two extremes, we evaluated the performance of the MCTS search for $M = 10, 50, 70, 100$. Overall, we found that $M = 50$ yields consistently increasing scores across all properties, which we use for the remainder of the study.

Number of Iterations. The number of iterations N_{iter} determines the number of selection, expansion, rollout, and backpropagation loops executed in a single MCTS run. In addition, N_{iter} is the maximum value of t or the number of unmasking steps that can be executed before the rollout begins, which corresponds to the maximum tree depth. We found that equating the number of diffusion steps T to the number of MCTS iterations N_{iter} results in convergence on the property prediction scores as the selection process becomes biased towards a single unmasking scheme. As shown in Figure 4, all property scores converge for $N_{\text{iter}} = T = 128$, which we use for the remainder of the study.

Tokenization Scheme. To evaluate the effect of different tokenization methods on the generation quality, we experimented with three different tokenization schemes: SMILES Pair Encoding (SPE) tokenization with the trained vocabulary used by PeptideCLM and Atom Pair Encoding (APE) tokenization for SMILES and SELFIES [38] representations. Overall, we found that the SPE tokenization scheme decreased perplexity and maintained precision while capturing common peptide motifs like bonds and recurring side chains.

Table 12: Effect of Tokenization on Sequence Length, Training, and Validation Loss after Convergence

Tokenization Scheme	Vocab Size	Avg Sequence Length in Data	Train Loss (↓)	Val Loss (↓)	Val PPL (↑)
SMILES SPE Tokenizer	**581**	84	**0.65**	**0.75**	**2.12**
SMILES APE Tokenizer	605	19	1.33	2.32	10.18
SELFIES APE Tokenizer	605	21	1.56	2.50	12.12

G ALGORITHMS

Algorithm 1 outlines the forward pass diffusion process (training) for PepMDLM, our unconditional peptide SMILES generator. Algorithms 2, 3, 4, 5 and 6 describe PepTune, our MCTS-guided peptide SMILES generator. Algorithms 7 and 8 describe the bond mask function and peptide sequence decoder which can also act as a validity filter.

Algorithm 1 PepMDLM Training

Inputs: Batched training examples $\mathbf{x}_0$
Output: Trained unconditional MDLM for peptide SMILES generation $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$
procedure TRAIN
  Sample $t \sim \text{Uniform}(0,1)$ ▷ *sample continuous times*
  ▷ *bond-dependent masking schedule* ◁
  $\alpha_t(\mathbf{x}_0) \leftarrow \big(\mathbf{1} - \textsc{BondMask}(\mathbf{x}_0)\big)(1-t) + \textsc{BondMask}(\mathbf{x}_0)(1-t^w)$
  ▷ *mask each sequence in batch at varying degrees* ◁
  Sample $\mathbf{z}_t \sim \text{Cat}(\mathbf{z}_t; (\alpha_t(\mathbf{x}_0))\mathbf{x}_0 + (1-\alpha_t(\mathbf{x}_0))\mathbf{m})$
  $\mathbf{x}_\theta(\mathbf{z}_t, t) \leftarrow \text{RoFormer}_\theta(\mathbf{z}_t, t)$ ▷ *predict token logits with RoFormer backbone*
  if $\mathbf{z}_t^{(\ell)} \neq \mathbf{m}$ **then**
    $\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t) \leftarrow \mathbf{z}_t$ ▷ *carry-over unmasking*
  else if $\mathbf{z}_t^{(\ell)} = \mathbf{m}$ **then**
    $\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t) \leftarrow \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t) - \infty\mathbf{m}$ ▷ *zero-masking probability*
  end if
  $\mathcal{L}_{\text{NELBO}}^{\infty} \leftarrow \frac{1}{|B|}\sum_{\mathbf{x}_0 \in B}\Bigg(-\sum_{\ell:\mathbf{x}_0^{(\ell)}=\mathbf{b}} \frac{w}{t}\log\langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\rangle -$
  $\sum_{\ell:\mathbf{x}_0^{(\ell)}\neq\mathbf{b}} \frac{1}{t}\log\langle \mathbf{x}_0^{(\ell)}, \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\rangle\Bigg)$
  $\tilde{\mathbf{x}}_0^{(\ell)} \leftarrow \arg\max \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t, t)$
  $\mathcal{L}_{\text{invalid}} \leftarrow \frac{1}{|B|}\sum_{\mathbf{z}_t \in B}\Bigg(\sum_{\ell=1}^{L} \tilde{\mathbf{x}}_0^{(\ell)\top}\text{SM}\big(\mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t,t)\big)\cdot \mathbf{1}[\tilde{\mathbf{x}}_0 \text{ is Invalid}]\Bigg)$
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{NELBO}}^{\infty} + \mathcal{L}_{\text{invalid}}$ ▷ *total loss*
  $\theta' \leftarrow \theta - \eta\nabla_\theta\mathcal{L}$ ▷ *update backbone parameters to minimize loss*
end procedure

Algorithm 2 Reverse Diffusion Step

Inputs: Partially unmasked sequence at time t $\mathbf{z}_t$
Output: Token probability distribution $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$ for all positions in the sequence with the bond-dependent reverse posterior and SUBS parametrization
procedure REVERSEDIFFUSIONSTEP
  $\mathbf{x}_\theta(\mathbf{z}_t, t) \leftarrow \text{RoFormer}_\theta(\mathbf{z}_t, t)$
  $s \leftarrow t - \frac{1}{T}$
  if $\mathbf{z}_t = \mathbf{m}$ **then**
    $p_\theta(\mathbf{z}_s|\mathbf{z}_t) \leftarrow \left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\mathbf{z}_s + \left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\mathbf{m}$
    $p_\theta(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t) \leftarrow 0$ ▷ *zero-masking probability*
  else if $\mathbf{z}_t \neq \mathbf{m}$ **then**
    $\mathbf{z}_s \leftarrow \mathbf{z}_t$ ▷ *carry-over unmasking*
  end if
  return $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$
end procedure

Algorithm 3 PepTune: Multi-Objective Guided Discrete Diffusion with Monte-Carlo Tree Search (MCTS)

Inputs: Trained MDLM denoiser $p_\theta(\mathbf{z}_s|\mathbf{z}_t)$, score function $\mathbf{s}(\mathbf{x}) : \mathcal{V}^L \to \mathbb{R}^K$ containing classifiers for K objectives $s_1, s_2, \ldots, s_K$, number of time steps T, number of iterations N_{iter}
Output: Set of Pareto-optimal sequences for the objectives and their K-dimensional classifier score vectors $\mathcal{P}^* = \{(\mathbf{x}_i^*, \mathbf{s}_i^*)\}$
procedure SAMPLEPEPTUNE
  $\mathbf{z}_T \leftarrow [\text{MASK}]^L$ ▷ *initialize fully masked sequence*
  $\mathcal{P}^* \leftarrow \{\}$ ▷ *initialize Pareto-front*
  for $i = 1, \ldots, N_{\text{iter}}$ **do**
    $\mathbf{z}_{t(n)}, t(n) \leftarrow \text{SELECT}(\mathbf{z}_T)$ ▷ *select expandable leaf node unmasked to time $t(n)$*
    $\mathbf{r} \leftarrow 0$ ▷ *initialize vector that will store the sum of all rewards from expanded children*
    $\text{children}(\mathbf{z}_{t(n)}) \leftarrow \text{BATCHEDREVERSESTEP}(\mathbf{z}_t)$
    `num_rollout_steps` $\leftarrow T - t(n)$
    $\vec{t} \leftarrow [\frac{n}{T}, \frac{n-1}{T}, \ldots \frac{1}{T}]$
    $dt \leftarrow \frac{1}{T}$
    for $i = 1, \ldots, M$ **do**
      $\mathbf{z}_{t(n)} \leftarrow \mathbf{z}_{s,i}$
      for n in range(`num_rollout_steps`) **do** ▷ *rollout to fully unmasked sequence*
        $p_{\theta,i}(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)}) \leftarrow \text{REVERSEDIFFUSIONSTEP}(\mathbf{z}_{t(n)})$
        $\mathbf{z}_{t(n)} \leftarrow \arg\max p_{\theta,i}(\mathbf{z}_{s(n)}|\mathbf{z}_{t(n)})$
      end for
      $\mathbf{x}_{s,i} \leftarrow \arg\max p_{\theta,i}(\mathbf{x}_{s,i}|\mathbf{z}_{t(1)})$ ▷ *get clean sequence*
      $\mathbf{s}(\mathbf{x}_{s,i}) \leftarrow \mathbf{s}(\mathbf{x}_{s,i})$ ▷ *compute score vector*
      ▷ *add sequence if non-dominated* ◁
      $\mathbf{r}(\mathbf{z}_{s,i}), \mathcal{P}^* \leftarrow \text{UPDATEPARETOFRONT}\big(\mathcal{P}^*, (\mathbf{z}_{s,i}, \mathbf{s}(\mathbf{z}_{s,i}))\big)$
      $\text{children}(\mathbf{z}_t).\text{append}\big(\mathbf{z}_{s,i}, \mathbf{s}(\mathbf{z}_{s,i})\big)$ ▷ *add child node*
      $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{r}(\mathbf{z}_{s,i})$ ▷ *add child reward to total reward for node $\mathbf{z}_t$*
    end for
    $\mathbf{z} \leftarrow \text{parent}(\mathbf{z}_{s,i})$
    while $\mathbf{z}$ not None **do** ▷ *backpropagate scores*
      $\mathbf{W}(\mathbf{z}) \leftarrow \mathbf{W}(\mathbf{z}) + \mathbf{r}$
      $N_{\text{visits}}(\mathbf{z}) \leftarrow N_{\text{visits}}(\mathbf{z}) + 1$
      $\mathbf{z} \leftarrow \text{parent}(\mathbf{z})$ ▷ *repeat for parent node until root node*
    end while
  end for
  return $\mathcal{P}^*$ ▷ *return Pareto-optimal sequences*
end procedure

Algorithm 4 Batched Reverse Step

Inputs: Partially unmasked sequence $\mathbf{z}_t$ at time t (representing the selected node in MCTS search), value of k for top k sampling ($k = 0$ for batched Gumbel-max sampling), total time steps T
Output: Set of M slightly unmasked sequences children$(\mathbf{z}_t) = \{\mathbf{z}_{s,1}, \dots \mathbf{z}_{s,M}\}$ at time s that become the child nodes of $\mathbf{z}_t$

procedure BATCHEDREVERSESTEP
  children$(\mathbf{z}_t) \leftarrow \{\}$
  $\mathbf{x}_\theta(\mathbf{z}_t, t) \leftarrow \text{RoFormer}_\theta(\mathbf{z}_t, t)$
  $s \leftarrow t - \frac{1}{T}$
  if $\mathbf{z}_t = \mathbf{m}$ **then**
    $p_\theta(\mathbf{z}_s|\mathbf{z}_t) \leftarrow \left(\left(\frac{s}{t} - \frac{s^w}{t^w}\right)\mathbf{b} + \frac{t-s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\mathbf{z}_s + \left(\left(\frac{s^w}{t^w} - \frac{s}{t}\right)\mathbf{b} + \frac{s}{t}\mathbf{1}\right)^\top \mathbf{x}_\theta(\mathbf{z}_t, t)\mathbf{m}$
    $p_\theta(\mathbf{z}_s = \mathbf{m}|\mathbf{z}_t) \leftarrow 0$ ▷ *zero-masking probability*
  else if $\mathbf{z}_t \neq \mathbf{m}$ **then**
    $\mathbf{z}_s \leftarrow \mathbf{z}_t$ ▷ *carry-over unmasking*
  end if
  for $\ell = 1 \dots L; i = 1 \dots M; j = 1 \dots |\mathcal{V}|$ **do**
    $u_{i,j}^{(\ell)} \sim \text{Uniform}(0, 1)$
    $G_{i,j}^{(\ell)} \leftarrow -\log\left(-\log(u_{i,j}^{(\ell)} + \epsilon) + \epsilon\right)$
  end for
  for $i = 1 \dots M$ **do** ▷ *define slightly different distribution for each sample in batch*
    if $k = 0$ **then**
      $\tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}|\mathbf{z}_t\big) \leftarrow \log p_\theta\big(\mathbf{z}_{s,i}|\mathbf{z}_t\big) + \mathbf{G}_i$ ▷ *batched Gumbel-max distributions*
      $\mathbf{z}_{s,i} \sim \tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}|\mathbf{z}_t\big)$
    else if $k > 0$ **then**
      $\tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) \leftarrow \text{SM}\Big(\text{top}k\big\{\log p_\theta\big(\mathbf{z}_{s,i}^{(\ell)}|\mathbf{z}_t^{(\ell)}\big) + \mathbf{G}_i^{(\ell)}\big\}\Big)$ ▷ *batched top k sampling*
    end if
    $\mathbf{z}_{s,i} \sim \tilde{p}_{\theta,i}\big(\mathbf{z}_{s,i}|\mathbf{z}_t\big)$
    children$(\mathbf{z}_t)$.append$(\mathbf{z}_{s,i})$
  end for
  return children$(\mathbf{z}_t)$
end procedure

Algorithm 5 Selection

Inputs: Masked root node $\mathbf{z}_{t(T)}$
Output: Expandable leaf node $\mathbf{z}_t$
procedure SELECT
  while True **do**
    if $\mathbf{z}_t$ is non-leaf node and $t \neq 0$ **then**
      $\mathcal{P}^*_{\text{select}} \leftarrow \{\}$ ▷ *initialize Pareto front of select scores*
      for $\mathbf{z}_{s,i}$ in children$(\mathbf{z}_t)$ **do**
        if $\mathbf{z}_{s,i}$ is non-leaf or expandable leaf node **then**
          $\mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i}) \leftarrow \frac{\mathbf{W}(\mathbf{z}_{s,i})}{N_{\text{visit}}(\mathbf{z}_{s,i})} + c \cdot p_\theta(\mathbf{z}_{s,i}|\mathbf{z}_t) \frac{\sqrt{N_{\text{visit}}(\mathbf{z}_t)}}{1+N_{\text{visit}}(\mathbf{z}_{s,i})}$
          $\mathcal{P}^*_{\text{select}} \leftarrow$UPDATEPARETOFRONT$\left(\mathcal{P}^*_{\text{select}}, (\mathbf{z}_{s,i}, \mathbf{U}(\mathbf{z}_t, \mathbf{z}_{s,i}))\right)$
        end if
      end for
      ▷ *set parent node for next iteration as a child node selected uniformly at random from Pareto-optimal set* ◁
      $\mathbf{z}_t \sim \mathcal{P}^*_{\text{select}}$
      SELECT$(\mathbf{z}_t)$ ▷ *recursively call select until leaf node is reached*
    else if $t = 0$ **then** ▷ *node is already fully unmasked*
      SELECT$(\mathbf{z}_T)$ ▷ *restart selection process from root node*
    else ▷ *return leaf node for expansion*
      return $\mathbf{z}_t$
    end if
  end while
end procedure

Algorithm 6 Update Pareto Front

Inputs: Current Pareto-front sequences and score vectors $\mathcal{P}^* = \{(\mathbf{x}_i^*, \mathbf{s}_i^*)\}$, newly sampled sequence and score vector $(\mathbf{z}_s, \mathbf{s}(\mathbf{x}_s))$
Output: Reward vector $\mathbf{r}(\mathbf{z}_s)$ and updated Pareto-optimal set $\mathcal{P}^*$
procedure UPDATEPARETOFRONT
  if $\mathcal{P}^*$ is empty **then**
    $\mathcal{P}^*$.append($(\mathbf{z}_s, \mathbf{s}(\mathbf{x}_s))$)
    $\mathbf{r}(\mathbf{z}_s) \leftarrow \mathbf{1}^K$ ▷ *set reward vector to ones*
  else
    ▷ *vector of boolean flags indicating which sequences are nondominant to* $\mathbf{x}$ ◁
    nondominateFlag ← new bool[$|\mathcal{P}^*|$]
    toDelete ← {}
    $\mathbf{r}(\mathbf{z}_s) \leftarrow \mathbf{0}^K$ ▷ *set reward vector to zeroes*
    for $(\mathbf{x}_i^*, \mathbf{s}_i^*)$ in $\mathcal{P}^*$ **do**
      ▷ *define vector with 1 where* $\mathbf{x}_s$ *is non-dominated in the property* ◁
      $\mathbf{n} \leftarrow [n_k = 1 \text{ if } s_k(\mathbf{x}_s) \succeq s_{k,i}^*]$
      ▷ *define vector with 1 where* $\mathbf{x}_{s,i}$ *is dominant in the property* ◁
      $\mathbf{d} \leftarrow [d_k = 1 \text{ if } s_k(\mathbf{x}_s) \succeq s_k^*]$
      $\mathbf{r}(\mathbf{z}_{s,i}) \leftarrow \mathbf{r}(\mathbf{z}_s) + \mathbf{n}$ ▷ *update reward vector*
      if $(\forall n_k \in \mathbf{n} \text{ s.t. } n_k = 1) \wedge (\exists d_k \in \mathbf{d} \text{ s.t. } d_k = 1)$ **then** ▷ $\mathbf{x}$ *dominates* $\mathbf{x}^*$
        toDelete.append($\mathbf{x}^*$)
        nondominateFlag[i] ← True
      else if $\forall n_k \in \mathbf{n}$ s.t. $n_k = 1$ **then** ▷ $\mathbf{x}$ *is not dominated by* $\mathbf{x}^*$
        nondominateFlag[i] ← True
      else ▷ $\mathbf{x}^*$ *dominates* $\mathbf{x}$
        nondominateFlag[i] ← False
      end if
    end for
    ▷ *if* $\mathbf{x}_s$ *is either dominant or non-dominated by all* $\mathbf{x}^*$ *in Pareto-optimal set* $\mathcal{P}^*$*, then add to* $\mathcal{P}^*$ ◁
    if $\forall i$ nondominateFlag[i] = True **then**
      $\mathcal{P}^*$.append($\mathbf{z}_s, \mathbf{s}(\mathbf{x}_s)$)
    end if
    for $\mathbf{x}$ in toDelete **do**
      $\mathcal{P}^*$.delete($\mathbf{x}^*, \mathbf{s}^*$)
    end for
  end if ▷ *return reward vector and updated Pareto-optimal set*
  return $\mathbf{r}(\mathbf{z}_s), \mathcal{P}^*$
end procedure

Algorithm 7 Bond Mask

Inputs: List of peptide SMILES strings `smiles_list`
Output: Position-wise bond mask for each sequence with 1 in positions of peptide bonds and 0 otherwise `mask`

```
procedure BondMask
    bond_patterns ← [(r'OC(=O)', 'ester'),
                     (r'N(C)C(=O)', 'n_methyl'),
                     (r'C(=O)N(C)', 'n_methyl'),
                     (r'N[12]C(=O)', 'peptide'),
                     (r'C(=O)N[12]?', 'peptide')]

    for batch_idx, smiles in enumerate(smiles_list) do
        positions ← empty_list()                    ▷ list to store bond positions
        used ← empty_set()                          ▷ set to track used positions
        for pattern, bond_type in bond_patterns do  ▷ identify bonds using patterns
            for match in re.finditer(pattern, smiles) do

                if not any(p ∈ range(match.start(), match.end()) for p in used) then
                    positions.append({start: match.start(), end:
                    match.end(),
                                      type: bond_type, pattern:
                    match.group()})
                    used.update(range(match.start(), match.end()))
                end if

            end for
        end for
        for pos in positions do                     ▷ update the mask for the current SMILES
            mask[batch_idx, pos[start]:pos[end]] ← 1
        end for
    end for
    return mask
end procedure
```

Algorithm 8 SMILES2PEPTIDE

```
Inputs: SMILES String s
Output: Batch of M sequences at time s.
procedure ANALYZER
    if s is correct SMILES format then,
        if s contains peptide bond [NC(=O)] or N-methylated peptide bond [N(C)C(=O)] then,
            IS_PEPTIDE ← TRUE
            positions ← empty_list()
            for pattern,bond_type in bond_patterns do
                for match in re.finditer(pattern,smiles) do
                    positions ← BONDMASK
                    segments ← empty_list()
                    positions.sort()
                    if positions[0]['start'] > 0 then,                ▷ first segment
                        segments.append(content: smiles[0:positions[0]['start']],
                                        bond_after: positions[0]['pattern'])
                    end if
                    for i in len(positions)-1 do                      ▷ other segments
                        current = positions[i]
                        next_pos = positions[i+1]
                        segments.append(content: smiles[current['end']:next_pos['start']],
                                        bond_before: current['pattern'],
                                        bond_after: next_pos['pattern'])
                    end for
                    if positions[-1]['end'] < len(smiles) then,       ▷ last segment
                        segments.append(content: smiles[positions[-1]['end']:],
                                        bond_after: positions[-1]['pattern']
                        )
                    end if
                end for
            end for
            for residue in segments do
                residue ← Regex pattern          ▷ Empirical Amino Acid Regex Pattern
            end for
        end if
    end if
end procedure
```