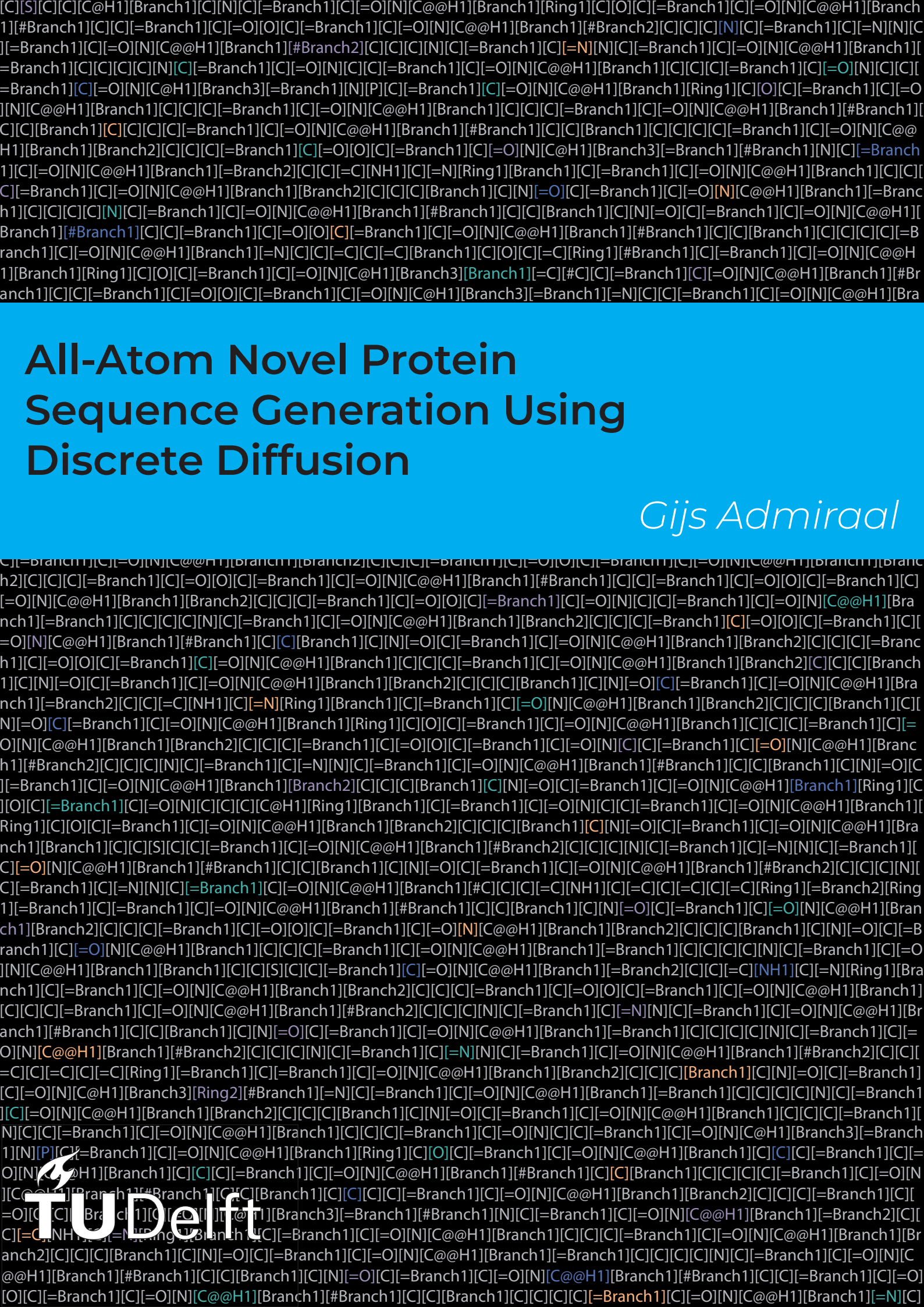
All-Atom Novel Protein Sequence Generation Using Discrete Diffusion

Gijs Admiraal



All-Atom Novel Protein Sequence Generation Using Discrete Diffusion

by

Gijs Jacobus Admiraal

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday December 2, 2024 at 09:30 AM.

Student number:	4871669	
Project duration:	November, 2023 – December, 2024	
Thesis committee:	Dr. Amelia Villegas-Morcillo,	TU Delft, Daily Supervisor
	Dr. Jana Weber,	TU Delft, Daily Supervisor
	Prof. Dr. Marcel Reinders,	TU Delft, Thesis Advisor
	Dr. Wendelin Böhmer,	TU Delft, External Committee Member

PREFACE

This report presents the results of my research on protein sequence generation using an all-atom approach, marking the completion of my Master's degree in Computer Science at Delft University of Technology. Exploring the fields of deep learning and bioinformatics has been both challenging and rewarding, making this journey truly inspiring.

I am deeply grateful to my supervisor, Dr. Jana Weber, for her guidance, patience, and constructive feedback, which shaped the direction of my research. My daily supervisor, Dr. Amelia Villegas-Morcillo, provided numerous invaluable insights and motivation that were essential throughout this process. I also want to thank Prof. Dr. Marcel Reinders for his advice and feedback, and Dr. Wendelin Böhmer for his time and contributions as an external committee member.

Lastly, I sincerely thank my family and friends for their unwavering support, for their encouragement and understanding during the more challenging moments of this project.

Gijs Admiraal
Delft, December 2024

All-Atom Novel Protein Sequence Generation Using Discrete Diffusion

Gijs Jacobus Admiraal
Technische Universiteit Delft

ABSTRACT

Advancing protein design is crucial for breakthroughs in medicine and biotechnology, yet traditional approaches often fall short by focusing solely on representing protein sequences using the 20 canonical amino acids. This thesis explores discrete diffusion models for generating novel protein sequences with an all-atom representation, specifically SELFIES a widely used molecular string representation. This all-atom approach considers the atomic composition of each amino acid in the protein. Enabling the inclusion of non-canonical amino acids and post-translational modifications. Using a modified ByteNet architecture and the D3PM framework, we compare the effects of this all-atom representation to the standard amino acid representation on the generated proteins' quality, diversity and novelty. Additionally, we see how a uniform or absorbing noise process affects the results. While models trained on the all-atom representation struggle to generate fully valid proteins consistently, those successfully designed showed improved novelty and diversity. Moreover, the all-atom representation can achieve comparable structural reliability results from OmegaFold to the amino acid models. Lastly, our results show that the use of an absorbing noise schedule is the most effective for both the all-atom and amino acid representation.

1 INTRODUCTION

The ability to successfully design proteins enables transformative solutions for medicine, industry, and environmental sciences [1]. By creating novel proteins or enhancing the design of existing ones, we can develop targeted therapeutics, efficient vaccines, and specialized enzymes for industrial and environmental applications [2]. Traditionally, protein design focuses on manipulating either the amino acid sequence [3], the three-dimensional structure [4], or both to achieve desired functions [5].

Proteins are macromolecules composed of long chains of amino acids linked by peptide bonds. The specific sequence of these amino acids determines how a protein folds into its unique molecular spatial structure, which dictates its function [6]. A protein can be represented by its amino acid sequence or its 3D structure. Conventionally, protein sequences are represented as sequences of the 20 canonical amino acids found in nature [3]. This sequence-based representation aligns with biological processes, where ribosomes in our cells translate mRNA sequences into these polypeptide chains [7].

However, this traditional sequence representation has limitations. It does not account for proteins incorporating non-canonical amino acids or ones that undergo post-translational modifications (PTMs) [8]. Non-canonical amino acids extend beyond the standard 20 canonical amino acids and can impart new functionalities to proteins [9]. PTMs involve chemical modifications after protein synthesis on individual amino acids or the protein level. These modifications can further diversify the function of a protein. One such protein that undergoes PTMs on a protein level is insulin, a critical hormone that regulates metabolism in animals [10]. This process is illustrated in Figure 1. Relying solely on amino acid sequences that use the representation of the 20 canonical amino acids can thus be insufficient for designing proteins which undergo these changes. Additionally, when using the amino acid representation it becomes inefficient to include the hundreds of extra non-canonical amino acids into its token list.

In addition to these limitations in sequence representation, there is a quantitative and qualitative lack of protein structural data. Proteins are dynamic and can adopt multiple conformations [12][13]. The techniques used to capture a protein structure in the lab provide only static snapshots, failing to represent the full spectrum of dynamic behaviours [14]. Furthermore, acquiring a single protein's structural data is time-consuming and resource-intensive, resulting in limited datasets that may not represent the full diversity of natural proteins. These limitations necessitate alternative approaches to understanding and predicting protein structures.

Recent advancements, such as AlphaFold [15], have made substantial progress in addressing the challenge of the time-consuming process of capturing protein structures by enabling accurate structure predictions from amino acid sequences [16]. This breakthrough allows us to mitigate some limitations of structural data scarcity by computationally predicting a protein static structure, making it more feasible to design proteins based solely on their sequences.

Building on recent advancements, computational methods and machine learning have significantly transformed protein design by enabling efficient exploration of the protein search space [17][18].

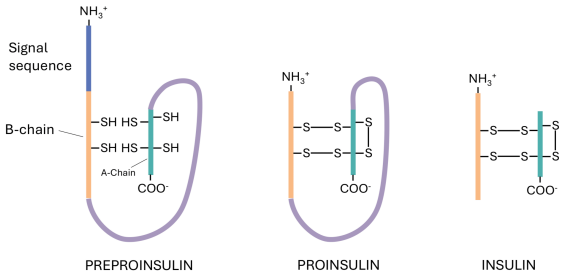


Figure 1: An illustration detailing the process of insulin synthesis through post-translational modifications (PTMs). Initially, the preproinsulin protein is modified by forming disulphide bonds, followed by two precise cleavages. This process removes a central segment, called the pro-peptide, resulting in the mature insulin structure composed of two distinct polypeptide chains, which are linked by disulphide bonds. (Adapted from [11])

Among these methods, generative models—particularly deep learning architectures like Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and diffusion models—have gained attention for their ability to create novel outputs. GANs [19] can generate high-quality outputs but often suffer from a lack of diversity. Conversely, VAEs [20] offer more diverse outputs through an encoder-decoder framework but may compromise on quality. Diffusion models [21] have emerged as promising methods that demonstrated great success in producing both diverse and high-quality outputs, albeit with increased computational demands.

Diffusion models [21][22][23] use a noising process to progressively transform input data into noise. By learning to reverse this process, the models can generate new samples from pure noise. Significant advancements have been made using diffusion models in fields such as computer vision [24] and protein design [4]. These models possess features highly relevant to novel protein generation, including the ability to produce diverse outputs that can be conditionally guided toward specific design objectives. Additionally, they support inpainting, allowing them to fill in missing portions of partially complete inputs.

However, most of these diffusion models are designed for continuous data spaces, which are well-suited for images and 3D structures but not for discrete data types. Proteins, at the sequence level, are inherently discrete, composed of sequences of amino acids represented by categorical variables. This discrete nature presents unique challenges that continuous diffusion models are not equipped to handle effectively.

Therefore, when it comes to protein sequence design, the discrete nature of amino acid sequences necessitates the use of discrete diffusion models. The earliest diffusion models operating over discrete state spaces considered diffusion processes over binary random variables [21]. Subsequent work extended these models to categorical random variables with transition matrices characterized by uniform transition probabilities [25], which govern the noising process within the model. Further research introduced a general framework that allows for various transition probabilities, including transitions toward a masked state [26]. This general model dubbed Discrete Denoising Diffusion Probabilistic Model (D3PM), demonstrated competitive results on image and text data compared to continuous diffusion models.

Later developments unified auto-regressive models with diffusion models into a new class called Auto-regressive Diffusion Models (ARDMs) [27]. ARDMs demonstrated similar performance to D3PM with absorbing transitions while significantly reducing the number of sampling steps. Auto-regressive models operate sequentially, sampling one token at a time, whereas diffusion models reconstruct sequences progressively over time steps. To accelerate sampling, ARDMs leverage a parallel generation trick [28], which uses dynamic programming to limit the number of sampling steps. This approach enables the unmasking of multiple tokens simultaneously at each step, speeding up sampling times.

While advancements in discrete diffusion models address the challenges of generating protein sequences in categorical spaces, their success hinges on the use of robust all-atom representations that ensure the generated outputs are chemically valid. One promising approach is SELF-referencing Embedded Strings (SELFIES) [29] which can represent simple to complex molecules sequentially enabling its use in computational methods. SELFIES has been designed specifically to be used in generative models and has shown promise in generating longer and more complex molecules [30]. Its syntax is designed such that each possible sequence of tokens encodes a valid molecule, which is essential for protein design, where preserving chemical validity ensures feasible and functional outputs.

Building on these advancements, our work seeks to overcome current challenges in protein sequence design by addressing limitations in existing methods. We propose using an all-atom representation, specifically SELFIES, of proteins combined with discrete diffusion models for protein sequence design. This all-atom approach considers every atom in the protein, capturing detailed molecular compositions, and providing greater flexibility and precision. This could allow for future integration of information related to non-canonical amino acids and PTMs. Since both the amino acid and the all-atom representations can be tokenized, a discrete diffusion process is preferred over a continuous one. Our work leverages the D3PM framework due to its competitive performance and the flexibility offered by interchangeable transition probabilities.

In this context, this Thesis addresses three key research questions: First, what is the quality of proteins generated by a discrete diffusion model using an all-atom representation? To measure quality we propose various metrics such as the presence of a continuous protein backbone and the number of correctly generated amino acids. Second, how does an all-atom representation impact the novelty, diversity, and structural correctness of generated protein sequences compared to traditional amino acid-level representations? Finally, what are the effects of applying different transition matrices or noise schedules—specifically uniform and absorbing—on the ability of discrete diffusion models to generate protein sequences?

The remainder of this thesis is organized as follows. In Section 2, we review the current related work on all-atom representations and the application of discrete diffusion in protein design. Section 3 provides the necessary background information supporting our methodology, which is detailed in Section 4, including the discrete diffusion setup and our proposed metrics. We present our results, showing the viability of the all-atom representation and the effectiveness of the absorbing noise schedule in Section 5. This is followed by an in-depth discussion in Section 6. Finally, we conclude with suggestions for future work in Section 7 and summarize our findings in Section 8. The code and implementation details for this work are available on GitHub[1].

2 RELATED WORK

In this section, we look at works that have used the all-atom representation for sequence design as well as structure design in proteins. Additionally, we compare our work with other works that revolve around protein design using discrete diffusion models.

[1] https://github.com/Intelligent-molecular-systems/All-Atom-Protein-Sequence-Generation

2.1 All-Atom Sequence Representation in Protein Design

One recent work has explored the use of an all-atom protein sequence representation for *de novo* generation [31]. This study employs two Generative Pre-trained Transformer (GPT) [32] models: one trained solely on canonical amino acids and another capable of generating sequences with non-canonical amino acids. The first model is trained on sequences constrained to only canonical amino acids, while the second is trained on a dataset incorporating molecular fragments attached randomly to protein side chains, which represent random non-canonical amino acids. While their method shows promise in expanding the diversity of generated proteins beyond the standard amino acids, it employs an auto-regressive GPT model instead of a diffusion model. In contrast to the auto-regressive nature of GPT models, diffusion models offer the advantage of generating entire sequences in parallel, which can lead to more efficient and diverse protein generation. Additionally, their research does not include a detailed analysis of faulty generated sequences at the atom level, as well as a more extensive elaboration on their method of classifying true proteins.

Although only one study focuses on all-atom sequence representation in protein design, this approach has been applied to other molecular design tasks. For instance, recent research has explored all-atom representations for the generative design of polymers [33] and smaller molecules [34], showcasing the versatility of this representation across different molecular domains.

2.2 All-Atom Structure Representation in Protein Design

All-atom representations at the structural level differ fundamentally from sequence-level representations, as they typically describe the full molecular structure in 3D space, including all atoms in the protein. However, such approaches are generally restricted to the 20 canonical amino acids, and the representation of sequences often remains at the level of amino acids rather than individual atoms.

For example, ESM3 [35] employs a multimodal generative language model to predict protein sequences, structures, and functions. While it uses a detailed molecular representation for structure prediction, its sequence representation is limited to amino acids, not individual atoms. Other works have leveraged all-atom diffusion models for protein co-design, simultaneously generating both sequence and structure [36][37]. These models incorporate an all-atom approach to side chain structures but still represent sequences using the 20 canonical amino acids. Similarly, studies focusing on all-atom structural predictions or incorporating ligands and modifications aim for more fine-grained all-atom structure design compared to backbone-only approaches, yet they do not extend this level of detail to sequence design with an all-atom perspective [38][39][40].

2.3 Discrete Diffusion in Protein Design

EvoDiff [41], utilizes a discrete diffusion framework to generate novel protein sequences. They employ a D3PM framework with both uniform and evolutionary-informed noising processes. When employing a masking noising process they utilize the ARDM. However, their work does not make use of its parallelization trick, losing its benefits for faster sampling times.

Other studies have explored discrete diffusion in protein design with different emphases. Functional-Group-Based Diffusion Model (D3FG) [42], combining discrete and continuous diffusion processes for pocket-specific molecule generation in drug design. Their model applies discrete diffusion to categorical data like functional groups while using continuous diffusion for atom positions and orientations. Discrete Flow Models (DFMs) [43] use discrete diffusion for protein co-design, where the sequence is modelled discretely, and the structure uses continuous diffusion, allowing for joint generation of protein sequences and structures. A different research presented diffusioN Optimized Sampling (NOS) [44], a guidance method that uses an absorbing state or masking in its noising process on the amino acid representation.

While these approaches advance the field, they primarily focus on the 20 canonical amino acids and do incorporate an all-atom sequence approach that could lead to the inclusion of non-canonical amino acids or post-translational modifications.

3 BACKGROUND

Here we examine the technical details of various design choices for our research. We discuss the general continuous and discrete diffusion models. Next, the ByteNet architecture, used for the generative diffusion process is discussed. Lastly, the all-atom representation is examined.

3.1 DDPM

Three sub-types of diffusion processes exist: Denoising Diffusion Probabilistic Models (DDPMs) [21], Score-based Generative Models (SGMs) [45], and Stochastic Differential Equations (SDEs) [46]. These diffusion processes share the common goal of learning a data distribution by iteratively adding noise to the input in the forward process. Subsequently, they systematically learn to remove the noise in the backward process. The sub-types vary in their approaches to executing both the forward and backward diffusion passes. A diffusion process can utilize various architectures for the backwards process, tailored to its specific requirements, as well as adapting the approach of noise addition and removal for optimal performance. By understanding the process of removing the corruption, the model can generate novel outputs from the learned data distribution.

DDPMs are a type of generative model capable of creating new data samples from a specified data distribution, using a dual Markov chain approach. In this approach, both the forward and backward processes are defined as Markov processes —a sequence of events where the state of the previous event dictates the probability of the next. A schematic overview of the whole DDPM process is given in Figure 2.

In the DDPM framework, the forward diffusion process iteratively transforms the original distribution over a specified number of steps, denoted as T. This transformation gradually introduces noise, ultimately converging toward a simpler prior distribution,

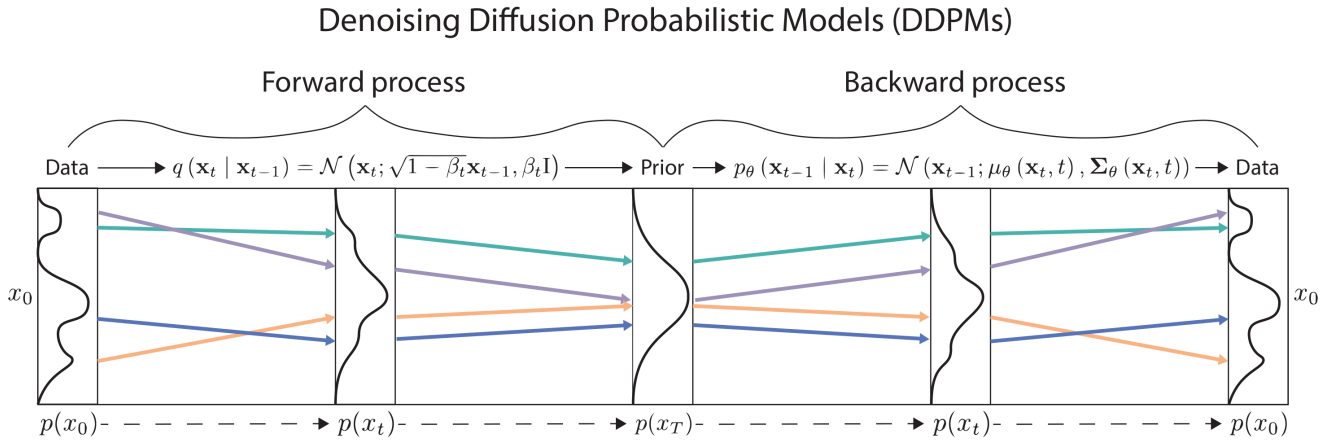


Figure 2: Schematic of a Denoising Diffusion Probabilistic Model (DDPM) using a continuous Gaussian noising process. This figure illustrates the forward process, which progressively transforms the original complex data into noise, and the backward process, which reverses the noising to generate new data samples. This enables us to generate novel data by learning the underlying distribution of the training data. (Adapted from [46])

often a standard Gaussian distribution. The amount of noise added at each step is controlled by a predefined noise schedule, denoted as β_t.

Formally, the forward process is defined by the probability $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, where $\boldsymbol{x}_t$ signifies the original input with noise corresponding to time step t. When a DDPM is used with a continuous Gaussian noising process its forward process is given as:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{1}$$

The backward diffusion process uses a neural network architecture θ that learns to predict the noise added in a forward step. This backward process reconstructs the original input based on the predicted noise at each time step. The backward process is formally given as $p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$, though in practice, the model often directly predicts $\boldsymbol{x}_0$ from $\boldsymbol{x}_t$ during training. The backwards process with a Gaussian noising process is given as:

$$p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{x}_t, t), \Sigma_\theta(\boldsymbol{x}_t, t)) \tag{2}$$

To optimize the generative model $p_\theta(\boldsymbol{x}_0)$ and fit it to the data distribution $q(\boldsymbol{x}_0)$, the following variational upper bound on the negative log-likelihood is minimized:

$$\begin{aligned} L_{\text{vb}} = \mathbb{E}_{q(\boldsymbol{x}_0)}[&\underbrace{D_{\text{KL}}\left[q\left(\boldsymbol{x}_T \mid \boldsymbol{x}_0\right) \| p\left(\boldsymbol{x}_T\right)\right]}_{L_T} \\ &+ \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}\left[D_{\text{KL}}\left[q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0\right) \| p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)\right]\right]}_{L_{t-1}} \\ &\underbrace{-\mathbb{E}_{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}\left[\log p_\theta\left(\boldsymbol{x}_0 \mid \boldsymbol{x}_1\right)\right]}_{L_0}] \end{aligned} \tag{3}$$

This equation represents the sum of Kullback–Leibler (KL) divergences between the forward and backward processes at each time step, which the model aims to minimize during training. The KL divergence measures the statistical distance between a reference and a second probability distribution. The term L_T represents the divergence at the final step, while L_0 is the reconstruction loss for the original data sample. The intermediate terms L_{t-1} account for the reconstruction terms between adjacent noisy steps.

Lastly, careful selection of the prior distribution is warranted. The prior distribution must allow for a tractable forward posterior process $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$ to calculate the KL-divergence loss. Additionally, it must allow efficient computation of $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$ using $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ for any time t. These criteria are met when working with a standard Gaussian noise process.

3.2 D3PM

The Discrete Denoising Diffusion Probabilistic Model (D3PM) [26] is a discretized generalized version of the Denoising Diffusion Probabilistic Model (DDPM). Since not all data such as text and amino acid tokens can be captured from a continuous setting, it is desirable to transform the DDPM in a discrete setting.

In D3PM, the forward process for a scalar random variable with K categories $x_t, x_{t-1} \in 1, \ldots, K$ is defined by a probabilistic transition matrix, represented as $[\boldsymbol{Q}_t]_{ij} = q(x_t = j \mid x_{t-1} = i)$. When we denote the row vector $\boldsymbol{x}$ as its one-hot version, we can write

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \text{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_{t-1}\boldsymbol{Q}_t) \tag{4}$$

, where $\text{Cat}(\boldsymbol{x}; \boldsymbol{p})$ is a categorical distribution over the one-hot row vector $\boldsymbol{x}$ with probabilities given by the row vector $\boldsymbol{p}$, and $\boldsymbol{x}_{t-1}\boldsymbol{Q}_t$ is a row vector-matrix product.

From this notation, we derive the two criteria necessary for a noise distribution in a diffusion process.

$$q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_0\right) = \text{Cat}(\boldsymbol{x}_t; p = \boldsymbol{x}_0\bar{\boldsymbol{Q}}_t) \text{ with } \bar{\boldsymbol{Q}}_t = \boldsymbol{Q}_1\boldsymbol{Q}_2 \ldots \boldsymbol{Q}_t \tag{5}$$

$$q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0\right) = \frac{q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0\right) q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0\right)}{q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_0\right)} \tag{6}$$

$$= \text{Cat}\left(\boldsymbol{x}_{t-1}; p = \frac{\boldsymbol{x}_t\boldsymbol{Q}_t^\top \odot \boldsymbol{x}_0\bar{\boldsymbol{Q}}_{t-1}}{\boldsymbol{x}_0\bar{\boldsymbol{Q}}_t\boldsymbol{x}_t^\top}\right) \tag{7}$$

Where Equation 5 shows how noise for any time step t can be efficiently calculated. Equation 7 describes how the tractable forward posterior can be calculated using Bayes' rule.

Using this approach, we are free to set the transition matrices to any noise schedule. Options include uniform noise, which applies a uniform noising process overall categories; a masking or absorbing noise process, where states gradually transition to an absorbing state; or a discretized Gaussian distribution. If there are inter-token relationships, such as evolutionary relationships between amino acids, a noise schedule informed by BLOSUM-62 [47] can be applied. An example of a uniform and an absorbing transition matrix can be seen in Figure 3. For both a uniform and absorbing noise schedule we can set the noising parameter to $\beta_t = (T - t + 1)^{-1}$ as given by the original D3PM work [26].

Lastly, an updated loss function is integrated into the diffusion model. The authors use an alternative hybrid loss function which leads to improved quality of samples:

$$L_{\text{hybrid}} = L_{\text{vb}} + \lambda L_{\text{simple}} \tag{8}$$

$$= L_{\text{vb}} + \lambda \mathbb{E}_{q(\boldsymbol{x}_0)} \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)} \left[-\log \widetilde{p}_\theta \left(\boldsymbol{x}_0 \mid \boldsymbol{x}_t \right) \right] \tag{9}$$

Where they introduce an extra denoising objective for the $\boldsymbol{x}_0$-parametrization of the reverse process, that encourages good predictions of the data $\boldsymbol{x}_0$ at each time step. This added objective corresponds to the cross-entropy term of L_0 in Equation 3 at $t = 1$ and is weighted by the λ parameter.

3.3 ByteNet

ByteNet [48] is a convolutional neural network (CNN) architecture designed for sequence-to-sequence tasks, such as machine translation. The architecture utilizes an encoder-decoder structure, where dilated convolutions are applied in the latent space, allowing the model to capture long-range dependencies within the sequence. Each sequence passes through multiple ByteNet blocks, where dilation functions act as a context window. A context window is the receptive field within which the model can "see" and process surrounding tokens in the sequence. It defines the number of tokens the model considers at a given position, helping it capture dependencies across various ranges without requiring recurrent processing. In each block, the dilation factor, denoted as k, increases exponentially for each subsequent layer, following the relation:

$$k = 2^{(n \mod p)}$$

. where n is the layer index, and $p = \lfloor \log_2 r \rfloor + 1$, with r being the maximum dilation factor at the last block. This exponential growth in dilation allows ByteNet to efficiently cover long contexts in the sequence without increasing the number of layers, which improves the model's efficiency and effectiveness for tasks involving long sequences.

Within each ByteNet block, several operations occur, as shown in Figure 4. The layers include normalization (LayerNorm) and activation functions (GeLU), followed by 1x1 convolutions and dilated convolutions with varying dilation factors. These operations ensure that the network captures both local and long-range features, while the residual connections enable efficient training by allowing information to bypass certain layers, facilitating gradient flow and avoiding vanishing gradients.

ByteNet stands out for its ability to leverage parallel computation across sequences due to its fully convolutional design, making it highly efficient, especially when handling long input sequences. Unlike transformers, which experience quadratic scaling with sequence length and can become computationally intensive. Notably, studies have shown that ByteNet achieves comparable performance to transformers in tasks such as masked protein sequence modelling [49].

3.4 SELFIES and All-Atom Representation

Proteins, being chains of bonded amino acids, are intuitively represented by their amino acid sequences. However, they can also be described by their detailed molecular structures. Representing a molecule as a linear string is challenging due to non-linear features like branches and rings. To address this, various techniques [50] have been developed, including SMILES [51], InChI [52], and

$$\begin{bmatrix} 1-\frac{2}{3}\beta_t & \frac{\beta_t}{3} & \frac{\beta_t}{3} \\ \frac{\beta_t}{3} & 1-\frac{2}{3}\beta_t & \frac{\beta_t}{3} \\ \frac{\beta_t}{3} & \frac{\beta_t}{3} & 1-\frac{2}{3}\beta_t \end{bmatrix} \quad \begin{bmatrix} 1-\beta_t & 0 & 0 & \beta_t \\ 0 & 1-\beta_t & 0 & \beta_t \\ 0 & 0 & 1-\beta_t & \beta_t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 3: Illustration of two transition matrices for a random variable with three categories at an arbitrary time step t: the uniform transition matrix (left) and the absorbing transition matrix (right). The uniform matrix consists of three core categories with equal transition probabilities to other categories. The absorbing matrix includes an additional masked fourth category which is the only category to which we can transition. As t increases, β_t increases and the matrix converges to the specified noise distribution.

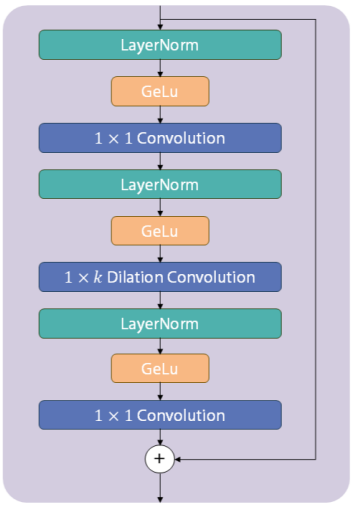


Figure 4: A schematic of a ByteNet block. Each block includes multiple layers of operations: layer normalization, GeLU activations, 1x1 convolutions, and a key 1xk dilated convolution layer. The dilation factor, k, increases exponentially across layers, allowing the network to capture long-range dependencies. Residual connections are incorporated to aid in model training and gradient flow. (Adapted from [49])

deep-learning approaches such as DeepSMILES [53] and SELFIES [29].

Simplified molecular-input line-entry system (SMILES) strings [51] have been a prominent method for representing molecular graphs in computational chemistry since 1988. In SMILES, molecules are defined as sequences of atoms represented by letters, with branches denoted by parentheses and ring closures indicated by matching numbers. While SMILES grammar allows for the description of complex structures and properties like stereochemistry and chirality, it is not inherently robust; generative models can produce invalid strings that do not correspond to valid molecular graphs.

To tackle this, SELF-referencing embedded strings (SELFIES) [29] offer a 100% robust molecular string representation, meaning that any combination of tokens corresponds to a chemically valid molecule. This robustness is achieved because SELFIES are designed to prevent the generation of syntactically and semantically invalid molecules by construction. This property is crucial in generative tasks where producing invalid sequences is undesirable.

In SELFIES, overloading is used to encode chemical structures in a way that eliminates common syntactic errors found in SMILES, such as unbalanced branch parentheses or incorrect ring identifiers. Overloading, in this context, means that certain tokens serve multiple purposes depending on their position and context in the sequence. For example, special tokens like [Branch1] or [Ring1] initiate branches or rings, and rather than requiring explicit end symbols, the subsequent tokens determine the length and connectivity of these features. This approach simplifies the representation and ensures structural validity throughout the sequence. Examples of both representations, SMILES and SELFIES can be seen in Figure 5.

Moreover, the SELFIES grammar dynamically tracks the number of available bonds to prevent the generation of semantically incorrect molecules. If a sequence exhausts the available bonds, the grammar omits further tokens, ensuring the molecule remains chemically valid.

4 METHODOLOGY

4.1 Dataset

Our diffusion model is trained and evaluated on the UniRef50 dataset [54], a subset of UniProt's Reference Clusters, which groups protein sequences with a 50% sequence identity threshold. UniRef50 was selected for its balance between comprehensive coverage and sequence diversity, as it provides a diverse collection of protein sequences while mitigating redundancy through the clustering of similar sequences. This ensures the model is exposed to a wide range of protein sequences while avoiding excessive similarity. Derived from UniRef90 seeds using MMseqs2 [55], the clusters rank proteins and combine closely related sequences into a single record to mitigate similarity issues. Additionally, this is an extensive dataset containing millions of available sequences giving us enough data to sufficiently train our models.

We obtained protein sequences from the UniRef50 dataset on May 7th 2024. To prepare the dataset, several steps were taken to filter the sequences. First, only sequences containing the 20 canonical amino acids were retained, while sequences with non-canonical amino acids were removed. This is done to simplify our model and because we cannot convert most sequences with non-standard amino acids to an all-atom representation, since these amino acids are often marked as unknown. Second, to manage computational resources and avoid excessive sequence length expansion with the SELFIES representation, we limited the maximum sequence length to 100 amino acids. Finally, sequences shorter than 30 amino acids were excluded, as these shorter sequences are often less representative of complex protein structures and may not adequately reflect the diversity needed for robust model training.

After filtering, the final dataset contains around 14 million protein sequences. Detailed data analysis figures, including sequence length distributions and amino acid frequency, are available in Appendix A, providing further insight into the dataset's composition. We split this dataset into training and validation sets with a rough 90/10 ratio. Care is taken to ensure that each split maintains a similar distribution of protein sequence lengths to ensure balance across all subsets.

4.2 Protein Sequence Representation

All possible tokens, for both representations, used in this research can be found in Appendix B.

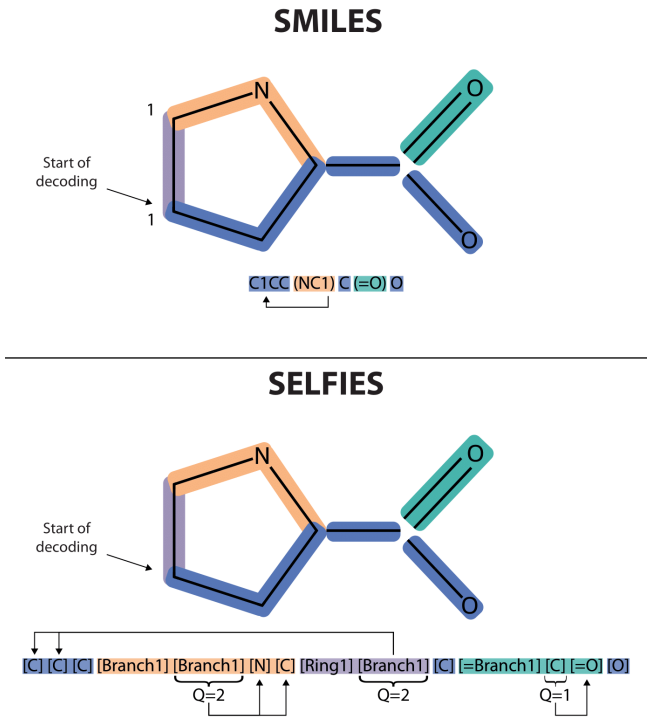


Figure 5: Comparison of SMILES and SELFIES representations for the amino acid proline. Both examples show how branch and ring formation are handled in their respective representations. For the branches and rings in the SELFIES representations, overloading symbols are shown with their length or connectivity (Q). (Adapted from [29])

4.2.1 Amino acid. The amino acid representation is directly derived from the UniRef50 dataset. In this format, each protein sequence is tokenized using the 20 canonical amino acids, represented by their respective single-letter codes. An example sequence in this representation is as follows:

RDGQKGGLEGLRQKGWSILNLL...

4.2.2 All-atom. The all-atom representation that we have chosen for our research is the SELFIES representation because of its ability to always generate valid molecules. This allows us to process outputs directly instead of first validating the output chemically. Each sequence is translated from its amino acid sequence into an RDKit molecule object from which we can find its corresponding SELFIES representation using the RDKit [56] and SELFIES [57] packages. In this format, a protein sequence is encoded with 21 SELFIES tokens. An example protein sequence in the all-atom representation appears as follows:

[C][C][C@H1][Branch1][C][C][C@H1][Branch2][#C]
[Ring1][N][C][=Branch1][C][=O][C@H1][Branch1]
[Ring1][C][O][N][C][=Branch1]...

4.3 Diffusion Framework and Architecture

We opted to use the D3PM framework due to its competitive performance and the flexibility offered by interchangeable transition probabilities. To model the conditional probability $p_\theta(\hat{x}_0|x_t)$, we chose the ByteNet architecture, which demonstrated promising results in protein sequence design for EvoDiff [41], particularly in handling long sequences efficiently and capturing long-range dependencies. Additionally, since the EvoDiff codebase was made publicly available, we could directly adapt their work to suit our needs.

During the training phase, the model input is a protein sequence and a diffusion time step. The protein sequences are tokenized according to the model's representation and mapped to embedding vectors of dimension d_{model}. The diffusion time steps t are encoded to vectors of dimension d_{model} using sinusoidal positional encoding. The sequence embeddings and diffusion step encodings are added element-wise and are then fed through several ByteNet blocks. Lastly, the output of the last ByteNet block is embedded back into the protein sequence representation space using a linear layer.

During the inference or sequence generation process, we start with a fully noised sequence, dependent on the noise schedule. We give this sequence together with the maximal time step (T) as input to our model. We then iteratively, predict $p_\theta(\hat{x}_0|x_t)$ and calculate its posterior $q\left(x_{t-1} \mid x_t, \hat{x}_0\right)$ (see Equation 7). Using this posterior we sample the next time step x_{t-1} using a multinomial distribution and feed it into our model together with time step $t-1$. After progressing through all time steps we end up with x_0. An example of how the generation progress looks like on both an all-atom and an amino acid sequence for both our noise schedules can be found in Appendix C

4.4 All-Atom-Level Evaluation

The model, trained using the SELFIES representation, focuses exclusively on sequences that represent proteins. However, our experiments revealed that the outputs are not always proteins and are not always composed solely of the 20 canonical amino acids. To assess how well the model generates canonical proteins, we developed a set of metrics that evaluate the presence of peptide bonds and a continuous backbone, constitutional and stereochemical correctness of the generated amino acids, and constructing the amino acid sequence of the generated molecules.

Each SELFIES sequence can be converted into a molecular structure due to the inherent properties of the SELFIES representation. The first step in our analysis involves converting the SELFIES sequence into a SMILES sequence, which is then transformed into an RDKit [56] molecule object for detailed examination. This conversion into this molecule object enables us to perform computational analyses on the molecular structures generated by the model.

Our evaluation method relies on performing substructure searches within the molecule and conducting graph traversals of the side chains. Since certain amino acids are substructures of others, directly searching for complete amino acids is ineffective. We begin by identifying peptide bonds[2] through substructure searches. Identifying peptide bonds allows us to locate a continuous backbone within the molecule, which is essential for subsequent analysis. In Figure 6 we can see a generic amino acid, a peptide bond in a continuous backbone, and two examples of how graph traversal is done on the side chain.

If a continuous backbone and its peptide bonds are identified, we proceed to analyse the side chains attached to each α-carbon atom. From the α-carbon, we find the beginning of the side chain. If no side chain is found, we classify that peptide bond as Glycine, the only amino acid without a side chain. If a side chain is present, we observe the first atom; if it is a carbon atom, we have identified an α-β carbon bond. We then perform a breadth-first search (BFS) graph traversal of the side chain starting from the α-β carbon bond, exploring new bonds not already part of the backbone or the side chain. If we find a side chain that does not start with a β-carbon we mark that residue as β-carbon lacking and we do not analyse the side chain further.

An important aspect of our analysis is checking the stereochemistry of the amino acids. All amino acids, except glycine, have a chiral centre, which means they can exist in two forms [58]. These forms, called stereoisomers, have the same molecular composition but differ in that their spatial 3D structures are mirror images of each other flipped around the chiral centre. Although the two stereoisomers exhibit identical physical and chemical properties, in living organisms only amino acids of one type are found. It is thus crucial that we check our generated proteins for the stereochemistry of their amino acids.

Using the identified amino acids, we apply two key metrics. The first is constitutional correctness, where we assess whether the atomic structure of the side chain matches that of a canonical amino acid. Secondly, we check for stereochemical correctness,

[2] A peptide bond in the SELFIES representation: [C][C][=Branch1][C][=O][N]

where we determine whether the stereochemistry of the amino acid corresponds to the *L*-form used by living organisms. If an amino acid meets both criteria, we classify it as canonical; otherwise, it is considered non-canonical.

After applying these metrics, we construct an amino acid sequence for the identified backbone. Canonical amino acids are recorded as their respective symbols, while non-canonical ones are denoted with an 'X'. From this found amino acid sequence, we reconstruct the molecule into SMILES and SELFIES representations. Comparing these reconstructed sequences to the original outputs from the model allows us to evaluate whether the model generated extra atoms connected to the backbone (SMILES check) or if the protein included redundant tokens in the SELFIES sequence (SELFIES check).

Based on these analyses, we categorize the generated molecules into four classes, as summarized in Table 1:

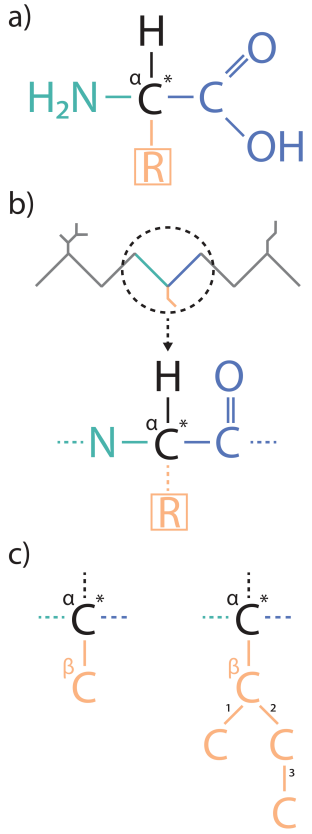


Figure 6: a) The molecular structure of an amino acid in its non-ionized form, showing the central α-carbon (black), the carboxyl group (blue), the amino group (green), and the variable side chain (orange). The asterisk marks the chiral centre. **b)** A section of a protein backbone highlighting a peptide bond, with the α-carbon indicated as a reference point for side chain analysis. **c)** Two examples of side chain structures with α-β carbon bonds: the left side chain corresponds to Alanine and the right to Isoleucine. The numbers illustrate the graph traversal order over the side chain bonds, which is used to analyse the complete structure of the side chain during evaluation. Notably, Alanine can be seen as a substructure of Isoleucine.

- **Canonical Protein**: A molecule composed exclusively of canonical amino acids with correct stereochemistry, and which passes the SMILES check. This indicates that the generated molecule exhibits all the traits of a canonical protein constructed solely from amino acids.
- **Non-Canonical Protein**: A molecule that contains any non-canonical amino acids or amino acids that are only constitutionally correct but not stereochemically correct. Additionally, it is considered non-canonical if the molecule does not pass the SMILES check — indicating the presence of additional atoms at the beginning or end of the backbone.
- **β-Carbon Lacking Protein**: A molecule where the side chain (not Glycine) does not begin with a carbon (i.e. lacks a β-carbon). This deviates from the standard amino acid structure and known non-canonical amino acids.
- **Not a Protein**: A molecule where we cannot construct a continuous backbone or cannot find any peptide bonds. Such molecules do not meet the basic structural criteria of a protein.

By applying these metrics, we can assess how effectively the model generates canonical proteins and identify where mistakes are made, thereby relating to our research objective.

4.5 Protein-Level Evaluation

The following metrics are observed on the amino acid sequences generated by the models. This includes sequences from the all-atom model that are labelled non-canonical proteins. This is because the following metrics allow for unknown amino acids, marked as 'X', in the input.

4.5.1 BLAST (Diversity and Novelty). To measure novelty and diversity, we compare each generated sequence using Basic Local Alignment Search Tool (BLAST) [59]. This method finds regions of similarity between biological sequences. The program can compare protein sequences to sequence databases and calculate the statistical significance.

To measure the novelty of the generated sequences we compare against the training dataset. To measure the diversity of the generated sequences, we perform pairwise comparisons among the generated sequences themselves, here we filter out matches where the query ID and match ID are the same. These metrics can provide insight into whether the model is producing novel and varied protein sequences.

For matches between queries and databases, we obtain several metrics. The first is the e-value, which tells us about the statistical significance of the match, with scores below 1e−5 marked as significant. Second, the score tells us how good the match is, looking at the match-up between amino acids and their evolutionary close neighbours. A higher score means more sequence similarity between the query and subject. Query cover describes how large the alignment is relative to the query, with 100% being a complete cover. Lastly, percentage identity refers to the percentage of identical matches between the query and subject over the aligned region, where 100% is an identical match.

Table 1: Classification criteria for generated molecules based on structural features and validation checks. Symbols used: ✓(criterion met), ✗(criterion not met), and – (criterion not applicable).

	Not a protein	β-C lacking protein[1]	Non-canonical protein	Canonical protein
Peptide bond present	–	✓	✓	✓
Continuous backbone present	✗	✓	✓	✓
Amino acid side chain without β-C present[2]	–	✓	✗	✗
Non-canonical amino acid present	–	–	✓	✗
Constitutional correct canonical amino acid present	–	–	✓	✗
Stereochemical correct canonical amino acid present	–	–	–	✓
SMILES check[3]	–	–	✗	✓
SELFIES check[4]	–	–	–	–

[1] A protein is a β-C lacking protein if it has at least one amino acid which does not have a side chain that starts with a β-C that is not Glycine.
[2] Any amino acid that has a side chain that does not start with a β-C which is not the amino acid Glycine.
[3] The SMILES check signifies if the generated molecule has an extra atom at the beginning or end of the backbone.
[4] The SELFIES check signifies that the generated sequence uses all of its SELFIES tokens to encode for the generated molecule. This check does not contribute to the classification.

For each search, we record the number of matches, and unique matches as well as the e-value, score, query cover and percentage identity for the matches. When we filter out all non-significant matches above an e-value threshold of $1e-5$ our experiments revealed there were no matches (see Appendix D). This is why we have opted to put our filter threshold at 0.05 since this strikes a balance between sensitivity and specificity in detecting significant sequence alignments.

4.5.2 OmegaFold Structure Prediction. We validate the structural foldability of the generated protein sequences using OmegaFold a sequence-to-structure prediction model. We utilize OmegaFold over other sequence-to-structure such as AlphaFold because of its faster runtime, and competitive accuracy [60]. OmegaFold provides predicted Local Distance Difference Test (pLDDT) scores for each residue in the protein, ranging from 0 to 100. These scores serve as confidence estimates indicating the reliability of the predicted amino acid positions in the protein structure. Higher pLDDT scores correspond to greater confidence in the prediction. In our evaluation, we use pLDDT scores to assess the structural viability of the generated proteins. Specifically, we average the pLDDT scores across the whole sequence and consider sequences with average pLDDT scores above 70 to be reliably predicted and below 50 to be unreliably predicted, following established conventions in protein structure prediction [15].

Proteins with higher pLDDT scores throughout their sequences are more likely to fold into stable, functional structures and are thus more promising candidates for potential applications and experimental validation. This structural evaluation complements our sequence-based analyses, providing a comprehensive assessment of the generated proteins' structural viability.

4.6 Experimental Setup

We used the 38M parameter ByteNet model defined in the EvoDiff work. However, its implementation was limited to amino acid sequence representation. Therefore, we made modifications to meet the specific needs of this study, including incorporating an all-atom representation using SELFIES and the use of an absorbing noising process. We also optimized the training loop to better handle the resumption of training by storing and loading the progression through the dataset, learning rate scheduler, and optimizer. Lastly, we integrated evaluation metrics for both the all-atom and protein levels, combined with pipeline code to streamline the evaluation process.

We trained four discrete diffusion models for protein sequence generation to compare the two different protein sequence representations and noising schedules. Each model was trained on the same dataset with the same architecture and hyperparameters (listed in Appendix E). Two of the models were trained using the all-atom representation and the other two on the amino acid representation. For each of the two representations, we trained the models using a uniform noising process and an absorbing noising process.

Each model was trained for several epochs on a single NVIDIA A40 GPU. Due to the increased number of tokens in the all-atom representation, these models completed 8 epochs, while the amino acid models trained for 30 epochs. The total training time was 312 hours for the all-atom models and 208 hours for the amino acid models. Training and validation curves, provided in Appendix F, show that the models have largely converged, with minimal variation in loss across later epochs.

For evaluation, we selected evenly spaced checkpoints throughout the training process to calculate the average validation loss. While the checkpoint with the lowest average validation loss was used for further analyses, it is important to note that the differences between average losses across checkpoints were negligible. This indicates that the specific choice of checkpoint does not significantly affect the model's performance, as all the selected checkpoints represent a similar level of convergence.

The evaluation process for the generated sequences is depicted in Figure 7. This schematic outlines the evaluation steps taken to assess both the atom-level and protein-level performance of the models.

In the original work of D3PM [26], the authors found that models trained on text performed better with different λ values in the loss function (see Equation 9) for different noising processes. With a uniform noising process, $\lambda = 0$ gave the best results, while the

absorbing state model achieved the best results with $\lambda = 0.1$. Therefore, for our experiments, we have chosen to use the same λ values in our D3PM loss functions.

4.6.1 All-Atom Model Performance on Atom-Level Metrics. In the first experiment, we evaluate the performance of the all-atom model on atom-level metrics. We generate 1000 sequences of random lengths between the minimal and maximum lengths found in the training set (225 - 1907). We compare the performance of the two noise schedules, uniform and absorbing, using our atom-level metrics. By analysing the atom-level performance, we can directly compare how well each noise schedule impacts the quality of generated sequences at a granular level. This set of experiments helps us understand the model's ability to handle the atomic details in protein design.

4.6.2 Protein-Level Evaluation on Both Representations. After evaluating the atom-level performance, we filter out sequences that are not proteins or that are β-C lacking protein. For each noise schedule, we obtain three sets of sequences for further analysis. A set of only canonical proteins generated by the all-atom model. A set of only non-canonical proteins generated by the all-atom model. Thirdly, a set of 1000 sequences is generated by the amino acid model between minimal and maximum lengths found in the training set (30 - 100).

For these three sets, we conduct our protein-level evaluation. These protein-level analyses allow us to compare how each model and noise schedule performs in generating structurally valid, novel and diverse sequences. This stage is crucial for assessing whether the all-atom representation can match or exceed the performance of the amino acid representation in generating canonical and non-canonical proteins.

4.6.3 Comparing Results Across Different Sequence Lengths. In the final experiment, we evaluate how each model performs across different sequence lengths. By selecting the generated sequences based on length, we compare the models using our established metrics. This analysis reveals how increasing sequence length affects the models' ability to generate valid and structurally sound proteins, particularly highlighting any challenges faced by the SELFIES models due to the increased complexity with longer sequences.

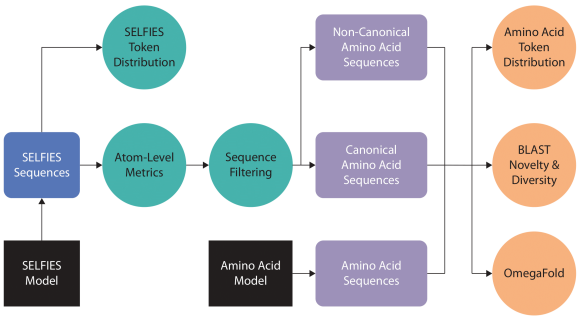


Figure 7: Evaluation workflow for generated sequences. SELFIES sequences generated by the models are analysed through various stages, including token distribution analysis, atom-level metrics, and protein-level evaluations; novelty and diversity using BLAST and OmegaFold Structure Foldability.

5 RESULTS

This section addresses our research questions by evaluating the effectiveness of different sequence representations and noise schedules in generating valid and diverse protein sequences. We analyse the performance of the models using both atom-level and sequence-level metrics and assess the novelty and diversity of the generated sequences through BLAST analysis and structural predictions through OmegaFold.

5.1 Atom-Level Metrics

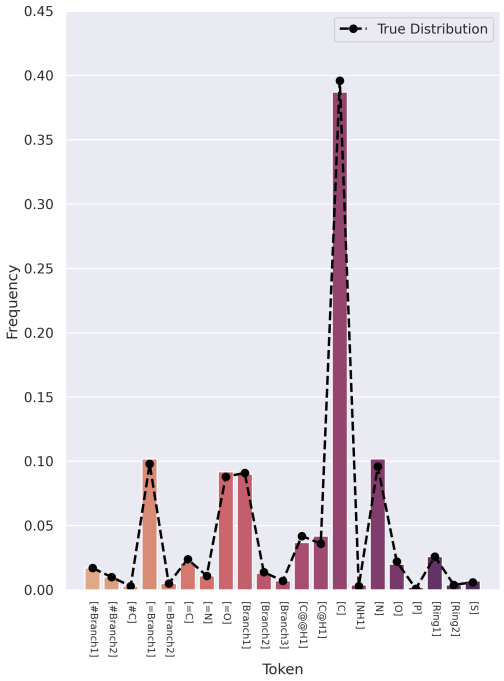
To understand the viability of the all-atom SELFIES representation, we generate 1,000 sequences for both the uniform and absorbing noise models. We compare the sequences based on how well they capture chemical validity and structural integrity at the atomic level.

5.1.1 SELFIES Token Distribution. We first examine the SELFIES token distribution for both the uniform and absorbing models, as shown in Figure 8. Both models generate distributions that closely align with the true distribution derived from the training set. Notably, the carbon token ([C]) is the most abundant across all distributions, consistent with the organic composition of proteins. The absence of significant deviations in token distributions suggests that both models successfully capture the statistical properties of the training data.

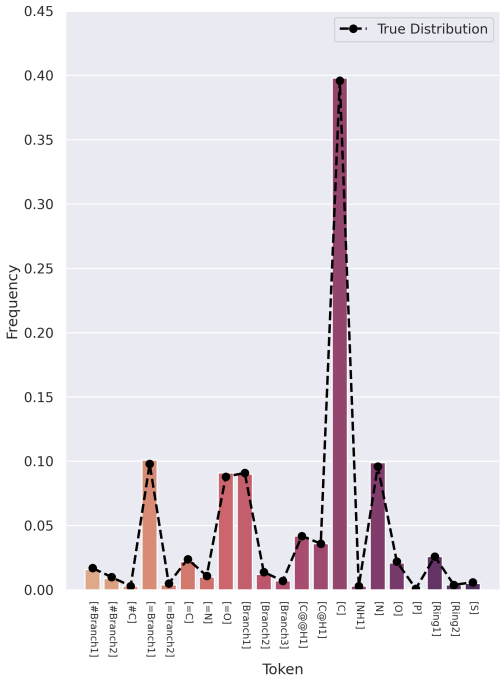
5.1.2 Unused SELFIES Tokens. Next, we analyse unused SELFIES tokens for sequences generated by each model. To detect unused tokens, we converted each generated SELFIES sequence to SMILES format and then back to SELFIES. This round-trip conversion is necessary because SMILES representations only include tokens that contribute to valid chemical structures. By comparing the original SELFIES sequence to the reconverted one, we can identify unused tokens at the end of the sequence that did not contribute to forming a valid molecule. This method effectively highlights portions of the generated sequences that are syntactically correct in SELFIES but chemically irrelevant or invalid.

The absorbing model exhibits a lower average number and an average relative of unused SELFIES tokens, averaging 239.093 unused tokens per sequence (18.9% relative), compared to 686.273 unused tokens (55.1%) for the uniform model. The high unused token ratio for the uniform model likely results from its noise process, which permits token alterations late in the generation process, disrupting sequence coherence. Whereas, the absorbing model can not change already unmasked tokens.

Unused tokens indicate that the models sometimes generate sequences with portions that are chemically irrelevant or invalid. Reducing the number of unused tokens is essential for improving the usability of the generated protein sequences, as it could increase the likelihood of producing valid and functional proteins.



(a) SELFIES - uniform noising



(b) SELFIES - absorbing noising

Figure 8: Distribution of SELFIES tokens generated by the uniform and absorbing noise models compared to true distribution taken from the training set. Both models closely match the true distribution.

5.1.3 Protein Classification. We further analyse the generated sequences to see which of our defined protein classes they fall into, as summarized in Table 2. Both models perform similarly in generating sequences with peptide bonds, a basic feature of proteins. However, the absorbing model outperforms the uniform model in generating sequences with continuous backbones, producing 239 compared to 52. This difference shows the absorbing model's ability to maintain better continuity in its generated sequence. This is likely due to the absorbing model using more of its SELFIES tokens and thus making fewer mistakes which leads to being able to produce a continuous backbone more often.

The models also differ in their ability to generate canonical and non-canonical proteins. The absorbing model generates 150 non-canonical proteins and 77 canonical proteins validated by SMILES conversion, compared to only 44 and 4 from the uniform model. This indicates that the absorbing model better captures the structural requirements for canonical protein synthesis since a greater portion of its backbones are converted to canonical proteins. Both models do show comparable performance in passing the SELFIES validation check for canonical proteins. This shows that when a model constructs a viable canonical protein sequence it has a high chance of utilizing all its available tokens.

The absorbing model generates 12 β-C lacking proteins out of 239 molecules with a continuous backbone compared to 4 out of 52 generated by the uniform model. This increase might be attributed to the absorbing model's noise schedule or its tendency to generate more sequences with continuous backbones, which could inadvertently include such anomalies. However, these occurrences are both low and suggest that both models predominantly generate sequences with correct amino acid side chains.

5.1.4 Amino Acid level Metrics. Building upon the protein classification, we next delve into the amino acid-level metrics to further assess the models' performance. We evaluate the amino acid composition of the generated sequences that at least contain a backbone structure, as summarized in Table 3.

Table 2: Summary of the 1,000 generated SELFIES sequences by the uniform and absorbing SELFIES models categorized by protein features. The absorbing model outperforms the uniform model in generating sequences with continuous backbones and canonical proteins.

	Uniform Noising	Absorbing Noising
Nr. Peptide bond[1]	990	995
Nr. Continuous backbone	52	239
Nr. Non-canonical proteins	44	150
Nr. SMILES check[2]	4	77
Nr. SELFIES check[3]	3	52
Nr. β-C lacking proteins	4	12

[1] A sequence contains at least one peptide bond

[2] The sequence is a canonical protein. If a sequence passes the SMILES check, no extra atoms are connected to the beginning or end of the backbone.

[3] The sequence is a canonical protein. If a sequence passes the SELFIES check that means all of its tokens in the SELFIES sequence are used to create the canonical protein.

The results show that the absorbing model generates 12,647 constitutionally correct canonical amino acids compared to 1,168 for the uniform model —a tenfold increase. This suggests that the absorbing model better captures amino acid-level features. Moreover, the absorbing model achieves higher average ratios of constitutional and stereochemical correctness for generated molecules with a continuous backbone (97.8%) compared to the uniform model (90.1%).

This indicates that the absorbing model is more adept at generating amino acids that are both structurally accurate and correctly stereochemically configured. For both models, when they generate a canonical amino acid, it is almost always stereochemically correct. This showcases that the models capture the dataset's stereochemical properties accurately, as all amino acids in the dataset are stereochemically correct. Interestingly, we have the same number of β-C lacking proteins as β-C lacking amino acids, meaning that each of these proteins only contains one β-C lacking amino acid. Lastly, the absorbing model generates longer proteins, especially non-canonical proteins, likely as a result of the uniform model making more critical mistakes.

5.2 Protein-Level Metrics

To evaluate the performance of our discrete diffusion models using both the amino acid and all-atom (SELFIES) representations, we generated 1,000 sequences from each amino acid model (uniform and absorbing noise schedules). For the SELFIES models, we filter the 1000 sequences to remove the ones that could not be processed due to invalid molecular structures. After this we obtain a smaller set of sequences: 44 non-canonical and 4 canonical sequences for the uniform noise schedule and 150 non-canonical and 77 canonical sequences for the absorbing noise schedule.

5.2.1 Amino Acid Token Distributions. First, we look at how each model captures the statistical properties of the training data by looking at the amino acid token distributions. Figure 9 presents the amino acid token distributions for the sequences generated by each model, compared to the true amino acid distribution from the training data. The amino acid models, both with uniform and absorbing noise schedules, produce amino acid token distributions that closely match the true distribution, indicating that they effectively capture the underlying amino acid composition of natural proteins. However, the match is not as close as how the SELFIES models align with their respective token distributions (see Figure 8).

In contrast, the amino acid sequences extracted from the SELFIES models exhibit more deviations from the true amino acid distribution. When sequences generated by the SELFIES models are converted to the amino acid representation, their amino acid token distributions are less accurate compared to those of the amino acid models. Notably, the SELFIES models show a higher proportion of the simplest amino acid Glycine (G). Additionally, all SELFIES models display a lower frequency of Threonine (T) and Valine (V) compared to the true distribution. Interestingly, these amino acids are not among the most complex amino acids.

Overall, the amino acid models more accurately replicate the true amino acid distribution compared to the SELFIES models, indicating better performance in capturing natural protein compositions, but the SELFIES models are not far off. It is important to acknowledge that the sample size for the SELFIES uniform model is significantly smaller than that of the other models, especially for canonical sequences generated by the SELFIES uniform model (only 4 sequences). Consequently, the observed amino acid distribution for this model may not be representative of its true performance, and conclusions drawn from this limited data should be interpreted with caution.

Table 3: Comparison of amino acid-level metrics for sequences with continuous backbones generated by the uniform and absorbing noise models. The absorbing model generates a larger number of amino acids overall, including both non-canonical and canonical amino acids, and achieves higher correctness ratios.

	Uniform Noising	Absorbing Noising
Nr. β-C lacking AAs	4	12
Nr. ncAAs[1]	93	276
Nr. const. correct[2]	1168	12647
Nr. stereo. correct[2]	1180	12634
Avg. ratio ncAAs per seq.[1]	8.9%	1.9%
Avg. ratio const. correct per seq.[2]	90.1%	97.8%
Avg. ratio stereo. correct per seq.[2]	97.8%	97.8%
Avg. non-canonical seq. len.[3]	24.091	60.90
Avg. canonical seq. len.[3]	30.50	37.27

[1] Non-canonical amino acids

[1] Canonical amino acids

[2] The average amino acid sequence length of a protein converted from a SELFIES sequence.

5.2.2 BLAST Analysis for Novelty and Diversity. To assess the novelty and diversity of the generated sequences, we perform BLAST searches, focusing on matches with an e-value lower than 0.5, which indicates some significant similarity while not being too strict.

As shown in Table 4, the amino acid models exhibit distinct patterns in novelty. The absorbing amino acid model generates more matches (336 matches) compared to the uniform model (46 matches). However, these matches correspond to considerably fewer unique sequences (54 unique matches for the absorbing model vs. 43 for the uniform model), resulting in a slightly worse Seq/U-Mat ratio (18.5 vs. 23.3). This indicates that the absorbing model produces sequences more similar to those in the training set, reflecting lower novelty. Despite these differences, the average e-values, scores, query coverage, and percentage identities are comparable between the two models.

Regarding diversity, both amino acid models perform similarly, with slight variations in the average e-values and scores. The absorbing model shows a marginally lower average score, implying that it generates sequences slightly less similar to each other.

Turning to the SELFIES models, the results show a marked difference in performance. The SELFIES absorbing model, both for canonical and non-canonical sequences, has minimal matches (one unique match in each case), indicating a high level of novelty. This implies that the generated sequences are substantially different

Table 4: Comparison of BLAST results for novelty and diversity among the uniform amino acid, absorbing amino acid, and SELFIES models. Results are filtered to include matches with an e-value lower than 0.5. Columns: Seq —number of sequences; Mat —matches; U-Mat —unique matches; Seq/U-Mat —ratio of sequences to unique matches; E-val —average e-value (± standard deviation); Score —average score (± standard deviation); Q-Cov —average query coverage percentage (± standard deviation); Idn —average percentage identity (± standard deviation).

	Seq	Mat ↓	U-Mat ↓	Seq/U-Mat ↑	E-val ↓	Score ↓	Q-Cov ↓	Idn ↓
BLAST Novelty								
Amino Acid - Uniform	1000	46	43	23.3	0.03 ± 0.01	80 ± 2.8	68 ± 16	17 ± 2.6
Amino Acid - Absorbing	1000	336	54	18.5	0.02 ± 0.01	81 ± 2.8	67 ± 15	16 ± 3.2
SELFIES - Uniform (non-canonical)	44	0	–	–	–	–	–	–
SELFIES - Uniform (canonical)	3	0	–	–	–	–	–	–
SELFIES - Absorbing (non-canonical)	150	1	1	150	0.02 ± 0	82 ± 0	70 ± 0	24 ± 0
SELFIES - Absorbing (canonical)	77	1	1	77	0.04 ± 0	77 ± 0	86 ± 15	16 ± 3.2
BLAST Diversity								
Amino Acid - Uniform	1000	54	54	18.5	0.03 ± 0.01	81 ± 3.2	53 ± 17	11 ± 2.4
Amino Acid - Absorbing	1000	49	49	20.4	0.03 ± 0.01	51 ± 2.8	52 ± 19	11 ± 3.1
SELFIES - Uniform (non-canonical)	44	1	1	1	0.03 ± 0	33 ± 0	61 ± 0	6 ± 0
SELFIES - Uniform (canonical)	3	0	–	–	–	–	–	–
SELFIES - Absorbing (non-canonical)	150	5	5	30.0	0.03 ± 0.01	42 ± 3.7	40 ± 8.6	8.4 ± 1.4
SELFIES - Absorbing (canonical)	77	2	2	35.0	0.03 ± 0.01	35 ± 0	54 ± 16	7.0 ± 0

from those in the training set. However, the small number of generated sequences, particularly for the canonical SELFIES uniform model, limits the robustness of this conclusion.

When assessing diversity, the SELFIES absorbing model outperforms the amino acid models, with higher Seq/U-Mat ratios (35 for canonical and 30 for non-canonical sequences) than those generated by amino acid models. Additionally, the SELFIES absorbing model outperforms in terms of score, query coverage, and identity metrics, suggesting it generates sequences that are both novel and diverse.

From the SELFIES uniform model, no conclusion can be drawn for both the novelty and diversity metrics due to a lack of usable matches frmr its small number of sequences.

5.2.3 Structural Viability Assessed by pLDDT Scores. We further evaluate the structural soundness of the generated sequences using OmegaFold to predict their structures and calculate per-residue predicted Local Distance Difference Test (pLDDT) scores. We average this score over the whole sequence. A higher pLDDT score indicates greater confidence in the predicted structure, with scores above 70 suggesting reliable folding.

Figure 10 shows the distribution of the average pLDDT scores for each model. The absorbing noise schedule produces sequences with higher pLDDT scores across amino acid and SELFIES models. This suggests that the absorbing noise schedule is more effective in generating structurally viable proteins.

Table 5 provides statistical details of the pLDDT score distributions. Although the SELFIES models generate fewer sequences than the amino acid models, they achieve higher or comparable average pLDDT scores. For instance, the SELFIES models all have higher average pLDDT scores, compared to 53.65 for the uniform amino acid model. The canonical proteins generated by the SELFIES absorbing have the highest mean with 63.03 compared with 58.44 for the absorbing amino acid model. Lastly, the generated distribution from the SELFIES models almost all have lower standard deviations than the amino acid models. This suggests that the all-atom representation may contribute to generating proteins with better predicted structural stability.

However, none of the models achieves a consistent average pLDDT score that approaches the threshold of 70, which is considered indicative of reliable protein folding. This highlights a limitation of the current models in producing sequences that can be confidently folded into stable structures, emphasizing the need for further refinement for all models.

5.3 Comparing Against Different Sequence Lengths

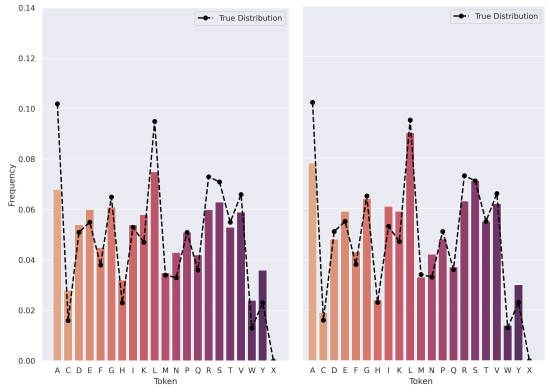
We now examine how grouping sequences by their lengths affects the results, analysing both atom-level and protein-level metrics. Sequence length can significantly impact the performance and generalizability of protein generation models. By grouping sequences by length, we aim to uncover how our models handle varying complexities inherent in proteins of different sizes.

5.3.1 Atom-Level Metrics. The results for the SELFIES models across various sequence lengths are presented in Table 6. We divided all possible sequence lengths into seven approximately evenly spaced categories. From these results, it becomes evident that as the generated sequence length increases, the models' performance

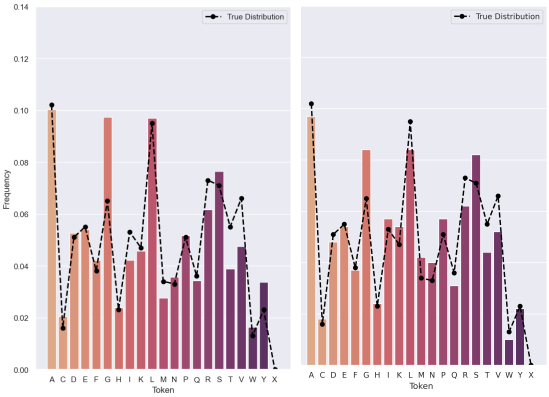
Table 5: Distribution statistics of the per-sequence average pLDDT scores assigned by OmegaFold. Higher scores suggest more structurally viable proteins.

	Seq	Avg. pLDDT ↑	Std. pLDDT ↓
Amino Acid - Uniform	1000	53.65	12.69
Amino Acid - Absorbing	1000	58.44	13.10
SELFIES - Uniform[1]	44	61.87	11.74
SELFIES - Uniform[2]	3	57.27	4.510
SELFIES - Absorbing[1]	150	56.67	13.80
SELFIES - Absorbing[2]	77	63.03	11.36

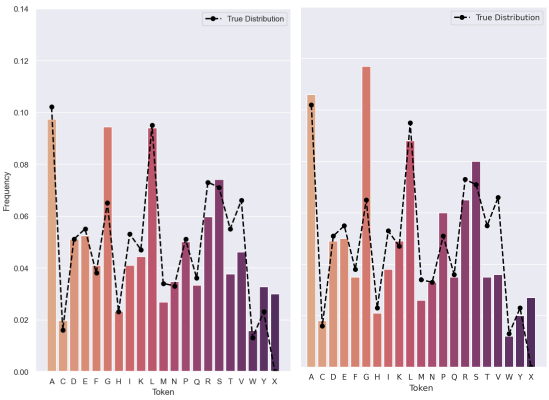
[1] Non-canonical proteins.
[2] Canonical proteins.



(a) Amino acid - uniform noising **(b)** Amino acid - absorbing noising

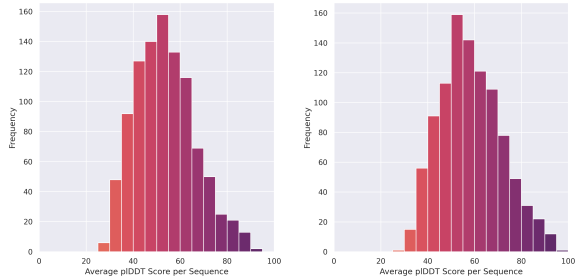


(c) SELFIES - uniform noising (canonical) **(d)** SELFIES - absorbing noising (canonical)

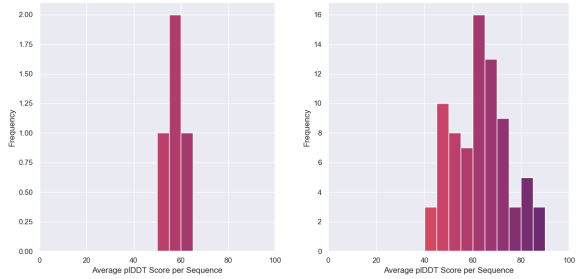


(e) SELFIES - uniform noising (non-canonical) **(f)** SELFIES - absorbing noising (non-canonical)

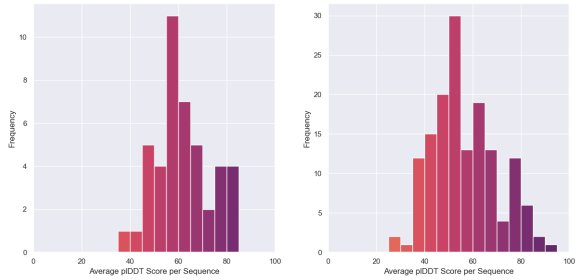
Figure 9: Amino acid token distributions from sequences generated using amino acid and SELFIES representations with uniform and absorbing noise schedules. Sequences generated in the SELFIES representation are filtered into canonical and non-canonical proteins and converted to their respective amino acid representations for comparison with the true amino acid token distribution.



(a) Amino acid - uniform noising **(b)** Amino acid - absorbing noising



(c) SELFIES - uniform noising (canonical) **(d)** SELFIES - absorbing noising (canonical)



(e) SELFIES - uniform noising (non-canonical) **(f)** SELFIES - absorbing noising (non-canonical)

Figure 10: Distribution of the OmegaFold average pLDDT scores for sequences generated using the amino acid and SELFIES representations with uniform and absorbing noise schedules. Sequences generated in the SELFIES representation are filtered into canonical and non-canonical proteins and converted to their respective amino acid representations. A pLDDT score of 70 and above is indicative of structurally reliable predictions.

deteriorates. Nonetheless, both models consistently produce at least one peptide bond per sequence across all length categories.

For the Uniform model, there is a steady decrease in the number of detected continuous backbones, non-canonical proteins, and canonical proteins as sequence length increases. Simultaneously, the average number of unused SELFIES tokens and the relative percentage of unused SELFIES tokens increase sharply. This suggests that errors accumulate with increasing sequence length, likely leading to incomplete or collapsed molecular structures. This observation is consistent with the model's difficulty in producing

Table 6: Summary of atom-level metrics for the SELFIES models for both uniform and absorbing noise processes. The results are shown for all lengths and grouped into seven evenly-spaced length categories.

	All Lengths	225–465	465–705	705–945	945–1185	1185–1425	1425–1665	1665–1907
Uniform Noising								
Nr. Peptide bond[1]	980	128	152	139	131	134	150	146
Nr. Continuous backbone	52	28	12	7	3	1	1	0
Nr. Non-canonical proteins	44	24	10	6	3	0	1	0
Nr. Canonical proteins	4	3	1	0	0	0	0	0
Avg. Unused SELFIES	686.3	84.45	212.4	368.5	668.4	902.2	1133	1376.
Avg. Relative unused SELFIES	55.09%	23.22%	35.52%	44.14%	62.46%	68.35%	73.31%	76.87%
Avg. Non-canonical seq. len.[2]	24.09	21.79	21.90	39.00	25.00	-	9.000	-
Avg. Non-canonical seq. len.[2]	30.50	27.00	41.00	-	-	-	-	-
Absorbing Noising								
Nr. Peptide bond[1]	995	142	139	135	157	132	143	147
Nr. Continuous backbone	239	72	60	47	28	18	7	7
Nr. Non-canonical proteins	150	30	31	35	26	17	7	4
Nr. Canonical proteins	77	41	27	7	2	0	0	0
Avg Unused SELFIES	239.1	28.57	66.27	113.8	216.1	283.5	389.8	560.5
Avg Relative unused SELFIES	18.87%	7.986%	10.93%	13.43%	19.96%	21.84%	25.35%	31.81%
Avg Non-canonical seq. len.	60.90	26.40	42.03	59.40	81.15	90.24	109.4	138.0
Avg Canonical seq. len.	37.27	25.49	45.44	61.86	82.50	-	-	-

[1] A sequence contains at least one peptide bond

[2] The average amino acid sequence length of a protein converted from a SELFIES sequence.

proteins longer than 40 amino acids, despite being trained on sequences up to 100 amino acids. This inability to generalize to longer sequences indicates potential limitations in the noising schedule.

In contrast, the Absorbing model exhibits a slower decline in performance metrics with increasing sequence length. The unused SELFIES tokens metrics increase more gradually, and the model can produce proteins even for the longest SELFIES sequence lengths tested. While it can generate non-canonical proteins at all lengths, it fails to produce canonical proteins for SELFIES sequences longer than 1185 tokens. Notably, the model produces proteins with amino acid sequence lengths ranging from 26 to 138, extending far beyond the maximum sequence length of 100 on which the model was trained. This suggests that the absorbing noise process may aid in generalizing to longer sequences.

Overall, the Absorbing model demonstrates a superior ability to handle longer sequences, suggesting that the absorbing noise process enhances the model's generalization capabilities.

5.3.2 Protein-level metrics. We now examine how different amino acid sequence lengths affect the protein-level metrics. We consider only the sequences from the SELFIES models that are either non-canonical or canonical proteins, grouping them by their sequence length in the amino acid representation. Due to the relatively low number of BLAST matches for both novelty and diversity metrics, we do not include these results in this section, as drawing conclusions from such small sample sizes would be unreliable. Therefore, we focus solely on the OmegaFold results presented in Table 7.

From these results, we observe a clear trend: the longer the amino acid sequence, the lower the average pLDDT score. This is not unexpected, as longer sequences are inherently more complex and pose greater challenges for both sequence generation and structure prediction algorithms like OmegaFold. Longer sequences have a larger conformational space and are more susceptible to cumulative errors during generation, leading to less accurate or less stable predicted structures. This trend is consistent across all models and noise processes evaluated.

Moreover, when we exclude protein sequences shorter than 30 amino acids, the absorbing amino acid model consistently achieves higher average pLDDT scores compared to the other models across all sequence lengths. This indicates that the OmegaFold results from the previous section are influenced by an over-representation of shorter sequences in the SELFIES models, which leads to skewed average pLDDT scores. Because these shorter sequences are less complex, they inherently have higher average pLDDT scores, elevating the overall average for the SELFIES models. In contrast, the amino acid models were not tasked with generating sequences below this length, resulting in lower overall averages. From this, we can conclude that for generating sequences most likely to be structurally sound, the absorbing amino acid model is the preferred choice.

However, the SELFIES absorbing model demonstrates the ability to generate protein sequences across a wide range of amino

Table 7: Summary of OmegaFold results for both the SELFIES and amino acid models using both uniform and absorbing noise processes. The results are shown for all lengths and grouped into nine length categories. The first (0–30) and last (100–200) are larger categories that handle outliers, while the rest are evenly spaced from 30 to 100, following the training set.

	All Lengths	0–30	30–40	40–50	50–60	60–70	70–80	80–90	90–100	100-200
Amino Acid - Uniform										
Nr. of proteins	1000	0	138	144	141	150	134	149	144	0
Avg. pLDDT ↑	53.65	–	64.00	60.31	56.42	53.62	50.05	46.66	45.00	–
Std. pLDDT ↓	12.69	–	12.54	10.94	11.40	11.09	9.494	11.14	9.388	–
Amino Acid - Absorbing										
Nr. of proteins	1000	0	133	144	152	142	138	138	153	0
Avg. pLDDT ↑	58.44	–	65.54	63.84	60.84	57.78	55.03	53.01	53.43	–
Std. pLDDT ↓	13.10	–	12.17	13.09	12.16	12.54	12.08	11.38	12.26	–
SELFIES - Uniform (non-canonical)										
Nr. of proteins	44	32	8	1	2	1	0	0	0	0
Avg. pLDDT ↑	61.87	64.98	55.85	50.38	45.95	53.70	–	–	–	–
Std. pLDDT ↓	11.74	11.17	8.66	–	9.109	–	–	–	–	–
SELFIES - Uniform (canonical)										
Nr. of proteins	4	2	1	1	0	0	0	0	0	0
Avg. pLDDT ↑	57.27	55.09	55.12	63.81	–	–	–	–	–	–
Std. pLDDT ↓	4.510	3.500	–	–	–	–	–	–	–	–
SELFIES - Absorbing (non-canonical)										
Nr. of proteins	150	23	22	20	19	11	12	13	13	17
Avg. pLDDT ↑	56.67	67.51	69.75	56.85	51.51	53.74	49.10	50.98	51.28	46.36
Std. pLDDT ↓	13.80	12.06	11.82	14.49	11.54	8.304	8.679	9.613	5.249	10.33
SELFIES - Absorbing (canonical)										
Nr. of proteins	77	31	15	14	10	5	1	0	1	0
Avg. pLDDT ↑	63.03	68.34	64.15	56.57	59.15	58.09	53.51	–	45.37	–
Std. pLDDT ↓	11.36	10.13	13.05	6.949	9.825	9.020	–	–	–	–

acid lengths, achieving slightly lower overall pLDDT scores compared to the amino acid models. Notably, it reaches the highest average pLDDT score (69.75) for a specific subset of sequences (30 – 40), approaching the 70 threshold. This highlights its potential for producing structurally sound sequences within certain categories. While its overall performance in structural confidence is slightly weaker, the model remains a viable option, particularly for applications requiring diversity in sequence lengths.

6 DISCUSSION

Our study demonstrates the feasibility of generating valid protein sequences using an all-atom SELFIES representation within discrete diffusion models, marking a step forward in protein design methodologies.

Notably, the absorbing SELFIES model exhibits superior performance over the uniform SELFIES model across multiple atom-level metrics. It excels in producing sequences with fewer unused tokens, continuous backbones, and a higher proportion of constitutionally and stereochemically correct amino acids. These attributes highlight its ability to capture the complex structural requirements essential for functional proteins. The improved performance can be attributed to the absorbing noise schedule, which preserves structural integrity by preventing late-stage disruptions during sequence generation. In contrast, the uniform noise model allows alterations throughout the generation process, leading to higher rates of unused tokens and less structurally sound sequences.

Despite these advancements, both SELFIES models generate a significant proportion of unusable sequences, as indicated by the presence of unused tokens and invalid structures. This limitation suggests room for improvement in the model's ability to generate fully valid protein sequences. The high number of unused SELFIES tokens in the uniform model likely indicates mistakes resulting in molecule collapse, given that the training data does not include sequences with unused tokens.

Comparing the outputs from the SELFIES-based models and the traditional amino acid models reveals a disparity in the number of valid proteins generated. While both models were configured to generate 1,000 sequences, the SELFIES models produced significantly fewer valid proteins, whereas the amino acid models consistently output valid proteins. This discrepancy arises from the inherent complexity of the all-atom SELFIES representation, which introduces a larger sequence space and increases the likelihood of generating invalid sequences. Maintaining an equal number of generation attempts for both models allows for a fair assessment of their efficiency in navigating their respective sequence spaces to produce valid proteins, which is crucial for practical applications where computational resources are limited.

After filtering out unusable sequences, we observe that the absorbing SELFIES model generates sequences that are more novel and diverse compared to the amino acid models. These sequences exhibit slightly lower structural scores than those from the absorbing amino acid model, as indicated by the pLDDT values. Although the overall pLDDT scores for all models are below the threshold

for reliable folding (70), the all-atom representation shows promise by achieving higher average scores within certain sequence length subsets. This suggests a slight trade-off between novelty, diversity and structural stability, where the all-atom representation enhances novelty and diversity at the potential cost of structural confidence.

The BLAST analysis further underscores the high novelty of the generated sequences. None of the sequences from any model has significant BLAST matches with an e-value less than $1e^{-5}$, indicating that all models consistently produce sequences that are highly novel and diverse. Both desirable properties when designing proteins, must be balanced with functional viability, as highly novel and diverse sequences may risk generating non-functional proteins.

Several limitations remain in our study. The all-atom models, while generating more novel and diverse sequences, do not fully capture the amino acid token distribution as accurately as the amino acid models. This discrepancy could be attributed to factors such as model architecture configuration, scale, or the inherent complexity of the all-atom representation. Specifically, the observed biases in amino acid distributions—such as the higher proportion of Glycine (G) in the SELFIES models—highlight the need to understand how the all-atom representation and noise schedules influence amino acid selection during generation. These biases may result from the all-atom representation's sensitivity to molecular complexity or the filtering process favouring simpler amino acids. Addressing these biases is essential, as they may affect the balance between diversity, novelty, and structural viability in the generated sequences.

Additionally, the relatively small sample size of valid sequences generated by the SELFIES models, particularly with the uniform noise schedule, affects the statistical power of our conclusions. This limitation necessitates caution in interpreting the results and indicates the need for further research to improve the models' efficiency in producing valid sequences.

Our analysis of sequence lengths reveals that longer sequences pose greater challenges for both generation and structural prediction. The absorbing SELFIES model demonstrates an ability to generate protein sequences across a wide range of lengths, achieving slightly lower overall pLDDT scores compared to the amino acid models. Notably, it attains the highest average pLDDT score for sequences within certain length categories, approaching the threshold indicative of reliable protein folding.

7 FUTURE WORK

To address the limitations identified and enhance the models' performance and reliability, future research should concentrate on several key areas. Since the current architecture was adopted from previous work and not specifically optimized for our application, refining and scaling the models is essential. Exploring different model architectures or scaling up the current ones could enable the capture of more complex patterns in protein structures, thereby improving the quality of the generated sequences. Moreover, exploring the effect of different values for the reweighting term (λ), which influences the model's learning dynamics, could enhance performance for protein sequence design, as the current values were directly taken from the original D3PM work without optimization for our specific use case.

Optimizing the noise schedules or experimenting with alternative ones may further enhance sequence generation by achieving a better balance between diversity and structural integrity. One such noise schedule could be a BERT-like [61] noise schedule which is a combination of a uniform and an absorbing noise schedule.

Expanding and diversifying the dataset is another crucial step. Currently, we limited our dataset to sequences shorter than 100 amino acids, whereas proteins can be much longer. Investigating whether the all-atom model can handle longer sequences could provide insights into its scalability and impact on results. Integrating data that includes non-canonical amino acids and post-translational modifications (PTMs) may enable the model to generate viable proteins not possible with the traditional amino acid representation. This enriched dataset could be used to fine-tune a pre-trained all-atom model or to train a new model from scratch, potentially enhancing the diversity and functionality of the generated proteins.

Mitigating biases in amino acid distributions is also important, as is addressing the over-representation of Glycine (G). Investigating the causes of these biases could involve examining the overall amino acid distribution before filtering out non-canonical and canonical proteins. Additionally, implementing a filter to exclude sequences with a high percentage of unknown amino acids (X) may improve dataset quality. Understanding whether the observed biases persist before filtering could inform strategies to mitigate them, potentially improving results in both BLAST and OmegaFold analyses.

Further analysis is needed to understand the difference in performance across various sequence lengths. Investigating why the SELFIES absorbing model can produce significantly longer sequences than those in the training set, and whether these sequences are composed mainly of simple or small amino acids, could provide valuable insights. Additionally, analysing the token distributions for both representations across different sequence lengths may reveal patterns affecting model performance.

Exploring all-atom sequences generated by auto-regressive models, such as GPT architectures, could improve atom-level metrics. Since these models generate tokens sequentially from left to right—the same direction in which SELFIES sequences are read—they may reduce errors by ensuring each token is conditioned on the preceding context. This sequential dependency could lead to more consistent and continuous molecules, potentially yielding different overall results.

Enhancing the molecular validity of the generated proteins is another important area for future work. Incorporating more stringent chemical validity constraints during training, such as validity checks for chemical structures, could reduce the generation of invalid sequences. Additionally, utilizing post-processing steps or error-correction mechanisms may correct minor errors, improving structural soundness. Integrating a regularization term into the loss function that penalizes unused SELFIES tokens could encourage the model to generate more complete and valid sequences.

Lastly, exploring alternative representations may offer new avenues for improvement. Grouping SELFIES tokens into motifs or biologically inspired functional groups could simplify the representation and reduce the likelihood of errors propagating during

sequence generation. This approach might mitigate the accumulation of unusable SELFIES tokens at the end of sequences, enhancing overall sequence validity.

By focusing on these areas, future research can build upon the findings of this study to develop more effective models for protein sequence generation, ultimately advancing computational protein design and its applications in biotechnology and medicine.

8 CONCLUSION

In summary, our work illustrates the significant potential of discrete diffusion models using an all-atom SELFIES representation for protein sequence generation. This approach offers a more detailed and flexible framework compared to traditional amino acid representations, enabling the incorporation of non-canonical amino acids and post-translational modifications. The absorbing SELFIES model, in particular, demonstrates capability in capturing complex structural features and generating novel and diverse sequences, indicating its promise for innovative protein design.

However, challenges remain in enhancing the validity and structural reliability of the generated proteins. Addressing these issues is crucial for translating computational designs into functional biological molecules. Future research should focus on refining the models to reduce the proportion of unusable sequences, addressing biases in amino acid distributions, and improving structural stability across varying sequence lengths. By tackling these limitations, we can advance the development of the design of proteins through the use of generative models.

REFERENCES

[1] Sotirios Koutsopoulos. *Peptide applications in biomedicine, biotechnology and bioengineering*. Woodhead Publishing, 2017.

[2] Xingjie Pan and Tanja Kortemme. Recent advances in de novo protein design: Principles, methods, and applications. *Journal of Biological Chemistry*, 296, 2021.

[3] Zachary Wu, Kadina E Johnston, Frances H Arnold, and Kevin K Yang. Protein sequence design with deep generative models. *Current opinion in chemical biology*, 65:18–27, 2021.

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

[5] Chentong Wang, Sarah Alamdari, Carles Domingo-Enrich, Ava Amini, and Kevin K Yang. Towards deep learning sequence-structure co-generation for protein design. *arXiv preprint arXiv:2410.01773*, 2024.

[6] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. Analyzing protein structure and function. In *Molecular Biology of the Cell. 4th edition*. Garland Science, 2002.

[7] Geoffrey M Cooper. Translation of mrna. *The Cell: A Molecular Approach*, 2, 2000.

[8] Travis S Young and Peter G Schultz. Beyond the canonical 20 amino acids: expanding the genetic lexicon. *Journal of Biological Chemistry*, 285(15):11039–11044, 2010.

[9] Bart Brouwer, Franco Della-Felice, Jan Hendrik Illies, Emilia Iglesias-Moncayo, Gerard Roelfes, and Ivana Drienovská. Noncanonical amino acids: Bringing new-to-nature functionalities to biocatalysis. *Chemical reviews*, 124(19):10877–10923, 2024.

[10] Gisela Wilcox. Insulin and insulin resistance. *Clinical biochemist reviews*, 26(2):19, 2005.

[11] Jack Westin. Post-translational modification of proteins [online image]. Jack Westin MCAT Content. Accessed: 2024-11-06.

[12] Kwangho Nam and Magnus Wolf-Watz. Protein dynamics: The future is bright and complicated! *Structural Dynamics*, 10(1), 2023.

[13] Nobuhiko Tokuriki and Dan S Tawfik. Protein dynamism and evolvability. *Science*, 324(5924):203–207, 2009.

[14] Katherine Henzler-Wildman and Dorothee Kern. Dynamic personalities of proteins. *Nature*, 450(7172):964–972, 2007.

[15] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[16] Letícia MF Bertoline, Angélica N Lima, Jose E Krieger, and Samantha K Teixeira. Before and after alphafold2: An overview of protein structure prediction. *Frontiers in bioinformatics*, 3:1120370, 2023.

[17] Jane R Allison. Computational methods for exploring protein conformations. *Biochemical Society Transactions*, 48(4):1707–1724, 2020.

[18] Subrata Pramanik, Francisca Contreras, Mehdi D Davari, and Ulrich Schwaneberg. Protein engineering by efficient sequence space exploration through combination of directed evolution and computational design methodologies. *Protein Engineering: Tools and Applications*, pages 153–176, 2021.

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[20] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[21] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[23] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[24] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation 2023. *arXiv preprint arXiv:2208.12242*, 2022.

[25] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021.

[26] 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*, 34:17981–17993, 2021.

[27] Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

[28] Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.

[29] 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.

[30] Daniel Flam-Shepherd, Kevin Zhu, and Alán Aspuru-Guzik. Language models can learn complex molecular distributions. *Nature Communications*, 13(1):3293, 2022.

[31] Daniel Flam-Shepherd, Kevin Zhu, and Alán Aspuru-Guzik. Atom-by-atom protein generation and beyond with language models. *arXiv preprint arXiv:2308.09482*, 2023.

[32] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI*, 2018.

[33] Seonghwan Kim, Charles M Schroeder, and Nicholas E Jackson. Open macromolecular genome: Generative design of synthetically accessible polymers. *ACS Polymers Au*, 3(4):318–330, 2023.

[34] Camille Bilodeau, Wengong Jin, Tommi Jaakkola, Regina Barzilay, and Klavs F Jensen. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 12(5):e1608, 2022.

[35] Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pages 2024–07, 2024.

[36] Alexander E Chu, Jinho Kim, Lucy Cheng, Gina El Nesr, Minkai Xu, Richard W Shuai, and Po-Ssu Huang. An all-atom protein generative model. *Proceedings of the National Academy of Sciences*, 121(27):e2311500121, 2024.

[37] Jiale Liu, Zheng Guo, Hantian You, Changsheng Zhang, and Luhua Lai. All-atom protein sequence design based on geometric deep learning. *Angewandte Chemie*, page e202411461, 2024.

[38] Namrata Anand, Raphael Eguchi, and Po-Ssu Huang. Fully differentiable full-atom protein backbone generation. 2019.

[39] Rohith Krishna, Jue Wang, Woody Ahern, Pascal Sturmfels, Preetham Venkatesh, Indrek Kalvet, Gyu Rie Lee, Felix S Morey-Burrows, Ivan Anishchenko, Ian R Humphreys, et al. Generalized biomolecular modeling and design with rosettafold all-atom. *Science*, 384(6693):eadl2528, 2024.

[40] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick,

et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024.

[41] Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Alex X Lu, Nicolo Fusi, Ava P Amini, and Kevin K Yang. Protein generation with evolutionary diffusion: sequence is all you need. *BioRxiv*, pages 2023–09, 2023.

[42] Haitao Lin, Yufei Huang, Odin Zhang, Yunfan Liu, Lirong Wu, Siyuan Li, Zhiyuan Chen, and Stan Z Li. Functional-group-based diffusion for pocket-specific molecule generation and elaboration. *Advances in Neural Information Processing Systems*, 36, 2024.

[43] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

[44] Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36, 2024.

[45] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in neural information processing systems*, 34:11287–11302, 2021.

[46] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[47] Sean R Eddy. Where did the blosum62 alignment score matrix come from? *Nature biotechnology*, 22(8):1035–1036, 2004.

[48] N Kalchbrenner. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.

[49] Kevin K Yang, Nicolo Fusi, and Alex X Lu. Convolutions are competitive with transformers for protein sequence pretraining. *Cell Systems*, 15(3):286–294, 2024.

[50] Mario Krenn, Qianxiang Ai, Senja Barthel, Nessa Carson, Angelo Frei, Nathan C Frey, Pascal Friederich, Théophile Gaudin, Alberto Alexander Gayle, Kevin Maik Jablonka, et al. Selfies and the future of molecular string representations. *Patterns*, 3(10), 2022.

[51] 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.

[52] Stephen R Heller, Alan McNaught, Igor Pletnev, Stephen Stein, and Dmitrii Tchekhovskoi. Inchi, the iupac international chemical identifier. *Journal of cheminformatics*, 7:1–34, 2015.

[53] Noel O'Boyle and Andrew Dalke. Deepsmiles: an adaptation of smiles for use in machine-learning of chemical structures. 2018.

[54] Uniprot: the universal protein knowledgebase in 2023. *Nucleic Acids Research*, 51(D1):D523–D531, 2023.

[55] Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

[56] RDKit: Open source cheminformatics. Rdkit: Open-source cheminformatics. https://www.rdkit.org, 2024. Accessed: 2024-10-19.

[57] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Selfies python package. https://pypi.org/project/selfies/, 2024. selfies, Version 2.1.1, Released: Jul 15, 2024.

[58] Yongchan Jeong, Hyo Won Kim, JiYeon Ku, and Jungpil Seo. Breakdown of chiral recognition of amino acids in reduced dimensions. *Scientific Reports*, 10(1):16166, 2020.

[59] Scott McGinnis and Thomas L Madden. Blast: at the core of a powerful and diverse set of sequence analysis tools. *Nucleic acids research*, 32(suppl_2):W20–W25, 2004.

[60] Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, et al. High-resolution de novo structure prediction from primary sequence. *BioRxiv*, pages 2022–07, 2022.

[61] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

A DATASET ANALYSIS

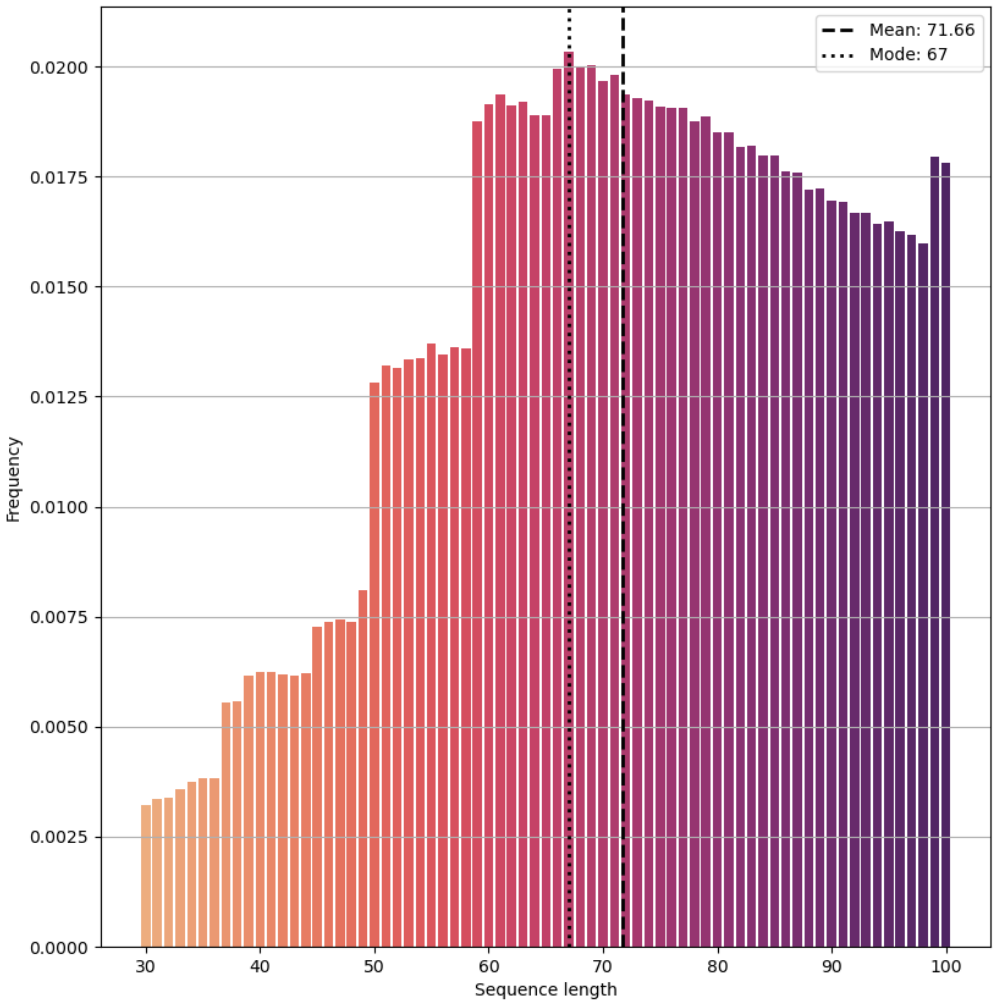


Figure 11: Distribution of protein sequence lengths in the amino acid representation, ranging from lengths 30 to 100. Both the mean and mode are approximately at length 70, indicating that most proteins in the dataset are around this length. There are relatively few proteins with lengths between 30 and 50. After peaking at length 70, the frequency of sequence lengths declines steadily from 70 to 98. Notably, there is a spike in frequency at lengths 99 and 100.

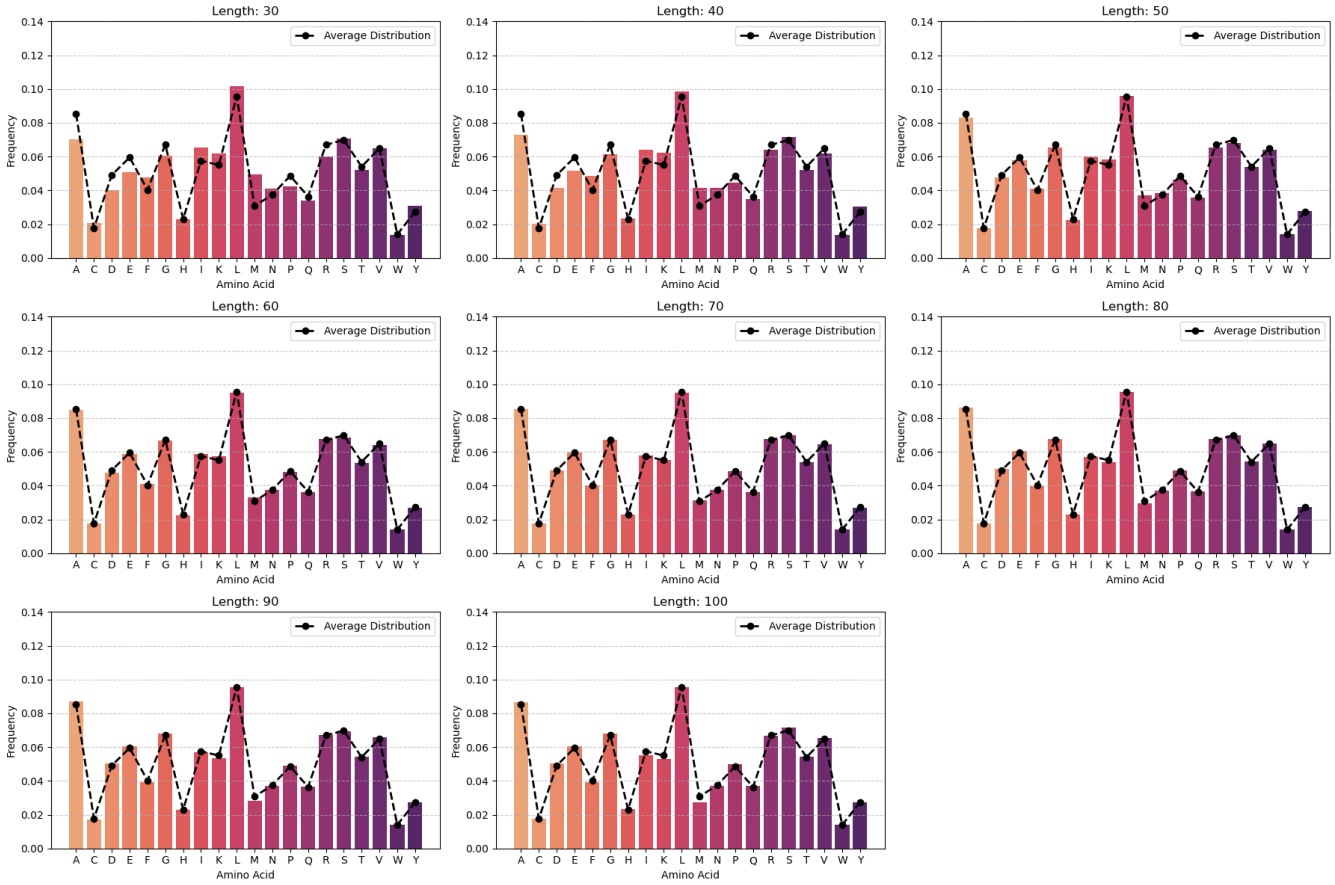


Figure 12: Amino acid token distributions for protein sequences grouped by length categories (30–40, 40–50, ..., 90–100), alongside the average amino acid distribution for the entire dataset. Sequences within the length categories of 50–100 exhibit amino acid distributions that closely align with the average distribution, as expected given that the majority of sequences fall into this range. This alignment indicates that these sequences are highly representative of the overall dataset. Sequences in the shorter length categories (30–50) display minor deviations from the average distribution but still provide a reasonable approximation of the dataset's amino acid composition.

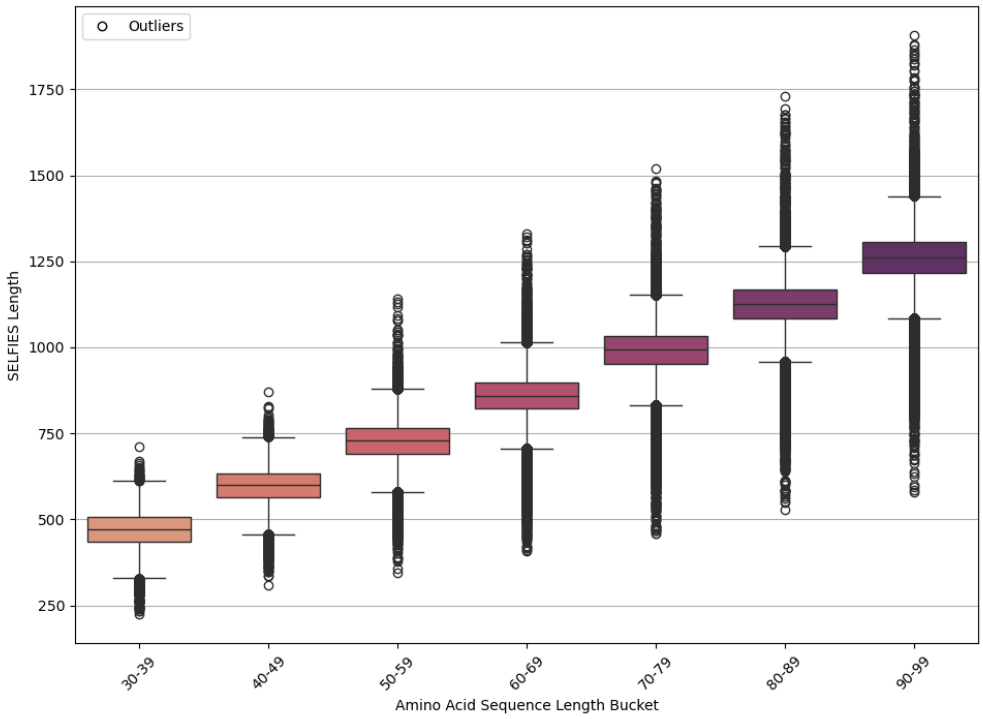


Figure 13: Distribution of SELFIES lengths by amino acid sequence length buckets. The box plots show that as amino acid sequence length increases, the average SELFIES length scales up approximately linearly. While the spread (interquartile range) increases gradually with longer sequences, indicating a relatively stable distribution, the number of outliers grows more noticeably. This suggests that while longer amino acid sequences generally correspond to longer SELFIES representations, there is an increasing degree of variability at these lengths.

B TOKEN DEFINITIONS USED IN THIS RESEARCH

Table 8: SELFIES and amino acid tokens used in this research

SELFIES Tokens	Amino Acid Tokens
[#Branch1]	A
[#Branch2]	C
[#C]	D
[=Branch1]	E
[=Branch2]	F
[=C]	G
[=N]	H
[=O]	I
[Branch1]	K
[Branch2]	L
[Branch3]	M
[C@@H1]	N
[C@H1]	P
[C]	Q
[NH1]	R
[N]	S
[O]	T
[P]	V
[Ring1]	W
[Ring2]	Y
[S]	

C PROGRESS OF DENOISING PROTEIN SEQUENCES

Table 9: Sequence generation progression for amino acid representation with both a uniform and an absorbing noise schedule

Time step	Sequence State - Uniform Noise Schedule	Sequence State - Absorbing Noise Schedule
500	PTSKPINIGQRDRPTVMHYEKPHTHTFQWM	##############################
450	GHIKHGSHRTVWFYNNNASMWRRHQQLTVR	M###############G#D###########
400	MPRVHQSHVTVWCVTNNQSRWSYHGDDTER	M###D#####W###I#G#D######A####
350	MPRVHGSDVTVICVTNNQSIYSYHGDRTER	M###D#####W###I#GSD######A####
300	MPRVHGSDVTVIMVTNRQSIYSQHGDRQER	M###D#Y###W###I#GSDA#####AR###
250	MPRVHGSDVTVIMVTNRQSIYSQHGDRQER	M#VFD#Y#CSW###I#GSDA##N#TAR##T
200	MPRVHGSDVTVIMVTNRQSIYSQHGDRQER	M#VFD#Y#CSW#A#I#GSDA##N#TAR##T
150	MPRVHGSDVTVIAVTNRQSIYSMHGDRQEA	M#VFD#YECSW#A#I#GSDAY#NRTART#T
100	MPRVHGSDVTVIAVTNRQSIYSMHGDRQLA	MPVFDCYECSWSAFIIGSDAYGNRTART#T
50	MPRVHGSDVTVIAVTNRQSIYSMHGDRQLA	MPVFDCYECSWSAFIIGSDAYGNRTART#T
0	MPRVHGSDVTVIAVTNRQSIYSMHGDRQLA	MPVFDCYECSWSAFIIGSDAYGNRTARTDT

Table 10: Sequence generation progression for SELFIES representation with a uniform noise schedule

Time step	Sequence State - Uniform Noise Schedule
500	[#Branch1][Ring2][C@H1][P][P][C@H1][=N][C][P][#C][O][#C][N][N][#C][#Branch1][P][=Branch2][N] …
450	[S][C][Ring1][=Branch2][#Branch1][C@H1][=Branch2][#Branch1][Ring2][NH1][Ring2][C][C@@H1][=O] …
400	[C][#Branch2][C@H1][=Branch2][#Branch1][C@H1][Branch3][P][Ring2][Branch1][Ring2][C][S][S][NH1] …
350	[C][#Branch2][C@H1][Branch1][#Branch1][C@H1][N][Ring1][Ring2][Ring1][Ring2][=O][=O][C@H1] …
300	[C][NH1][C@H1][Branch1][C][C@H1][N][Ring1][#Branch2][Ring1][Ring2][=O][=O][C@H1][Branch1][=O] …
250	[C][NH1][C@H1][Branch1][C][C][N][Ring1][Branch3][Ring1][Ring2][=C][=O][Branch2][Branch1][=C][C@H1] …
200	[C][NH1][C@H1][Branch1][C][C][N][Ring1][Branch3][Ring1][C][=Branch1][=O][Ring2][N][Branch1][C@H1] …
150	[C][C][C@H1][Branch1][C][C][N][Ring1][Branch3][Ring1][C][=Branch1][=O][=O][N][Branch1][C@H1] …
100	[C][C][C@H1][Branch1][C][C][N][Branch1][C][Ring1][C][=Branch1][C][=O][N][Branch1][C@H1][C@@H1] …
50	[C][C][C@H1][Branch1][C][C][C@H1][Branch1][C][Ring1][C][=Branch1][C][=O][N][Branch1][N][=C][#C] …
0	[C][C][C@H1][Branch1][C][C][C@H1][Branch1][C][N][C][=Branch1][C][=O][N][C@H1][Branch2][=C] …

Table 11: Sequence generation progression for SELFIES representation with an absorbing noise schedule

Time step	Sequence State - Absorbing Noise Schedule
500	## …
450	####[C]###[Branch1]##################[C@H1]##################[#Branch2]###################### …
400	##[C@H1]#[C][C]##[Branch1]####[C]############[C]#[C@H1]############[=Branch1]#[=O]##[#Branch2] …
350	##[C@H1]#[C][C]##[Branch1]#[N]##[C]#####[C]######[C]#[C@H1]############[=Branch1]#[=O]##[#Branch2] …
300	##[C@H1]#[C][C]##[Branch1]#[N]##[C][=O]####[C]######[C]#[C@H1][Branch1]#####[C]#[C]####[=Branch1] …
250	##[C@H1][Branch1][C][C]##[Branch1]#[N]##[C][=O]####[C]######[C]#[C@H1][Branch1][#Branch2]###[N][C] …
200	##[C@H1][Branch1][C][C]##[Branch1]#[N]#[=Branch1][C][=O]####[C]######[C]#[C@H1][Branch1][#Branch2] …
150	#[C][C@H1][Branch1][C][C]#[Branch2][Branch1]#[N]#[=Branch1][C][=O][C@H1]###[C][C]##[N]##[C]#[C@H1] …
100	#[C][C@H1][Branch1][C][C]#[Branch2][Branch1]#[N][C][=Branch1][C][=O][C@H1]###[C][C]#[N][N]##[C] …
50	[C][C][C@H1][Branch1][C][C]#[Branch2][Branch1][=N][N][C][=Branch1][C][=O][C@H1]#[=Branch1][C][C][C] …
0	[C][C][C@H1][Branch1][C][C][C@H1][Branch2][Branch1][=N][N][C][=Branch1][C][=O][C@H1][Branch1] …

D BLAST RESULTS FOR DIFFERENT E-VALUE THRESHOLDS

Table 12: BLAST search results for 1,000 amino acid sequences generated using both uniform and absorbing noising processes, evaluated across different e-value[1] thresholds. The table illustrates how varying the e-value threshold impacts the number of total matches and unique query IDs obtained for both novelty and diversity metrics. As the e-value threshold becomes less stringent (increasing from lower to higher values), both the total matches and unique query IDs increase for both noising processes. This analysis informs the selection of an e-value threshold of 0.05 for our final results, providing a balance between sensitivity and specificity in detecting significant sequence alignments. Comparing the results between the uniform and absorbing noising processes reveals differences in the number of matches found, indicating the impact of the noising process on the generated sequences.

Threshold e-value	Uniform Noising Process				Absorbing Noising Process			
	Novelty		Diversity		Novelty		Diversity	
	Match Count	Unique IDs	Match Count	Unique IDs	Match Count	Unique IDs	Match Count	Unique IDs
1×10^{-5}	0	0	0	0	0	0	0	0
1×10^{-4}	0	0	0	0	4	2	0	0
1×10^{-3}	0	0	0	0	31	4	0	0
1×10^{-2}	6	6	9	9	143	11	10	10
5×10^{-2}	46	43	54	54	336	54	49	49
1×10^{-1}	88	86	130	120	535	83	78	74

[1] Each BLAST query match has an e-value which signifies the significance of the match. A match which has an e-value lower than $1e-5$ is considered to be significant.

E MODEL HYPERPARAMETERS

Table 13: Summary of hyperparameter configurations used in our D3PM implementation with the ByteNet architecture, including details on the dataset processing, optimizer settings, and learning rate scheduler. Each hyperparameter is listed alongside its value and source, indicating whether it was adopted from the EvoDiff implementation, based on findings from D3PM research, or specifically chosen in this work.

Hyperparameter	Value	Source
ByteNet		
Embedding Dimension (d_{embed})	8	EvoDiff
Model Dimension (d_{model})	1024	EvoDiff
Activation Function	GELU	EvoDiff
Slim	True	EvoDiff
Number of Layers (n_{layers})	16	EvoDiff
Kernel Size	5	EvoDiff
Max Dilation Value (r)	128	EvoDiff
Diffusion Time steps	500	EvoDiff
Number of Tokens Amino Acid (n_{tokens})	20	-
Number of Tokens SELFIES (n_{tokens})	21	-
Loss Reweighting Uniform (λ)	0	D3PM
Loss Reweighting Absorbing (λ)	0.1	D3PM
Causal	False	EvoDiff
Dropout	0.1	EvoDiff
Tie Weights	False	EvoDiff
Final Norm	False	EvoDiff
Dataset and Batch sampling		
Dataset	UniRef50	-
Max Tokens	40000	EvoDiff
Max Batch Size	800	EvoDiff
Bucket Size	1000	EvoDiff
Max Epoch	500	EvoDiff
Optimizer and Scheduler		
Optimizer	Adam	EvoDiff
Learning Rate	1×10^{-4}	EvoDiff
Weight Decay	0.0	EvoDiff
Scheduler	LambdaLR	EvoDiff
Warm-up Steps	10000	EvoDiff

F TRAINING AND VALIDATION CURVES

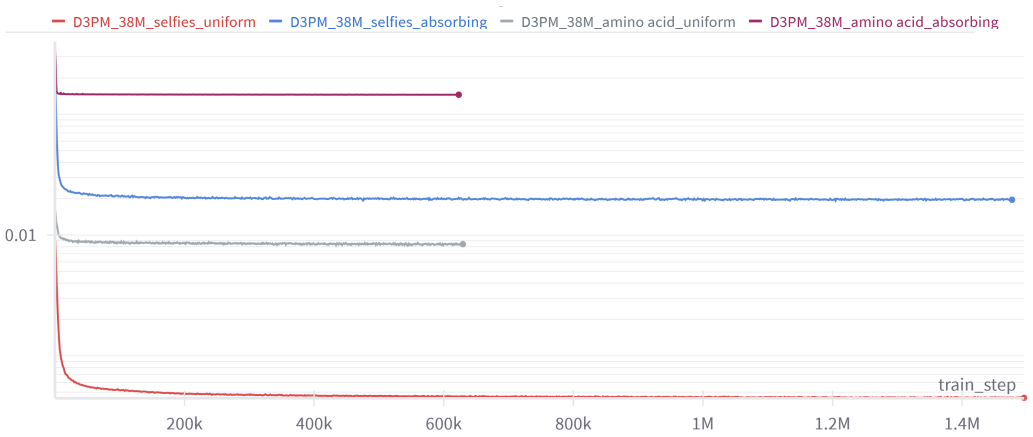
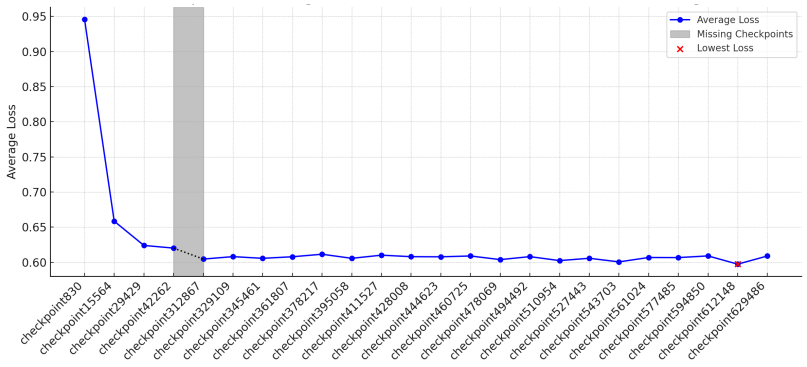
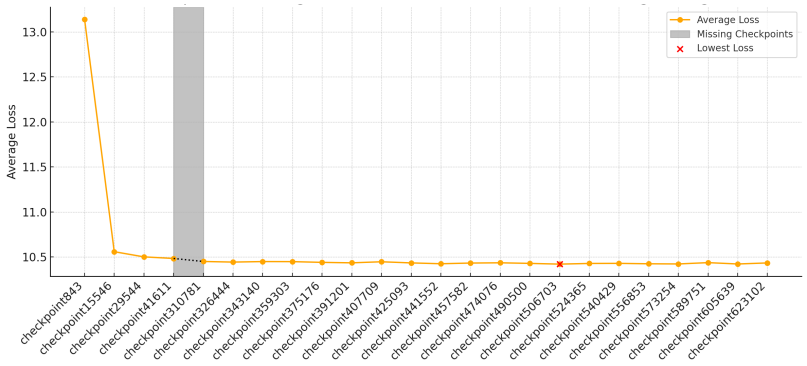


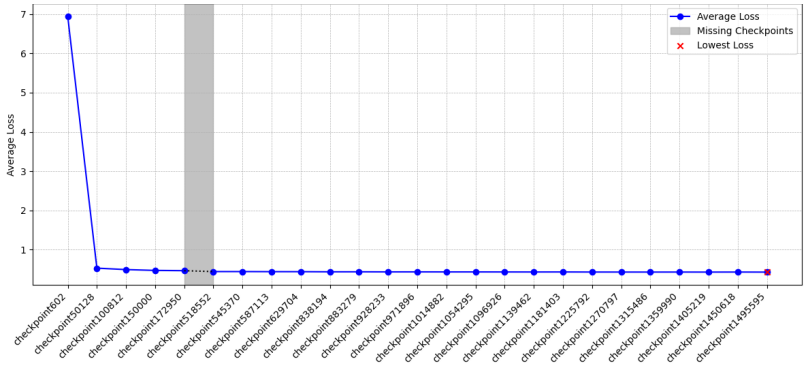
Figure 14: Training loss per step for the amino acid and SELFIES models under both uniform and absorbing noise schedules. The y-axis is in log scale, showing the loss values across the entire training run. All models appear to have converged, with the SELFIES models exhibiting a minimal decrease in loss over time compared to the amino acid models.



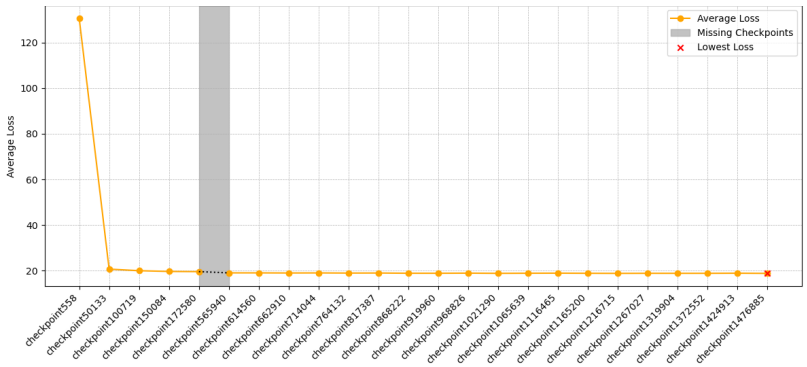
(a) Amino Acid Model - Uniform



(b) Amino Acid Model - Absorbing



(c) SELFIES Model - Uniform



(d) SELFIES Model - Absorbing

Figure 15: Average validation loss across different checkpoints for the amino acid and SELFIES models using both uniform and absorbing noise schedule. Initial checkpoints were missing due to memory constraints but some early checkpoints were later recovered from a backup, allowing for a more complete evaluation. The lowest average validation loss checkpoint is highlighted, indicating the best-performing model. The trend shows that the models have largely converged, with minimal variation in loss across the later checkpoints.