DDS-E-Sim: A Transformer-Based Generative Framework for Simulating Error-Prone Sequences in DNA Data Storage

Anonymous Author(s)

Abstract

The fast-growing amount of data needs reliable and long-lasting storage solutions. DNA has emerged as a promising medium due to its high information density and long-term stability. However, DNA storage is a complex process where each stage introduces noise and errors, including synthesis errors, storage decay, and sequencing errors, which require error-correcting codes (ECCs) for reliable data recovery. To design an optimal data recovery method, a comprehensive understanding of the noise structure in a DNA data storage channel is crucial. Since running DNA data storage experiments in vitro is still expensive and time-consuming, a simulation model is quite necessary that can mimic the error patterns in the real data and simulate the experiments. Existing simulation tools often rely on fixed error probabilities or are specific to certain technologies. In this study, we present a transformer-based generative framework for simulating errors in a DNA data storage channel. Our simulator takes oligos (DNA sequences to write) as input and generates erroneous output DNA reads that closely resemble the real-life output of common DNA data storage pipelines. It captures both random and biased error patterns, such as k-mer and transition errors, regardless of the process or technology. We demonstrate the effectiveness of our simulator by analyzing two datasets processed with distinct technologies. In the first case, processed with Illumina MiSeq, sequences simulated by DDS-E-SIM exhibit a total error rate deviation of only 0.1% from the original dataset. The second, processed with Oxford Nanopore Technologies, shows a 0.7% deviation. Both base-level and k-mer errors closely align with the original dataset. Additionally, our simulator generates 100,743 unique oligos from 35,329 sequences, with each sequence read five times, demonstrating its ability to simulate biased errors and stochastic properties simultaneously. Our simulator outperforms existing simulators with superior accuracy and the ability to handle diverse sequencing technologies.

CCS Concepts

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference acronym 'XX, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

Keywords

DNA data storage, Transformer, Generative model, Simulator, Noise structure

ACM Reference Format:
Anonymous Author(s). 2018. DDS-E-Sim: A Transformer-Based Generative Framework for Simulating Error-Prone Sequences in DNA Data Storage. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX).* ACM, New York, NY, USA, 13 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In the age of data expansion, the world produces 10^{18} bytes of data daily [48]. To manage the relentless growth of data, durable storage solutions (>50 years) are essential [78]. DNA has emerged as a promising medium for data storage due to its high information density and durability [45]. DNA can encode at most two bits per nucleotide or 455 exabytes per gram of ssDNA [18]. Despite being utterly dense, DNA Data Storage (DDS) requires very low maintenance power ($< 10^{-10}$ watts per gigabyte) [24]. Encoding binary data into nucleotide sequences involves several stages: synthesis, storage, sampling, and sequencing [13]. Synthesis errors, including insertion, deletion, and substitution, arise from imperfect nucleoside activation [75]. Coupling efficiencies, typically between 98% and 99.5%, directly influence the integrity of sequence lengths [75]. Storage decay, such as depurination and deamination, exacerbates the risks to data fidelity [75]. Lastly, sequencers are used to generate reads from stored DNA fragments, which are then decoded back into binary information [40]. DDS errors vary across processes and technologies. Numerous synthesis and sequencing technologies exist each exhibiting distinct error profiles. Artificial DNA synthesis employs chemical and enzymatic methods [74]. The first generation of methods uses column-based solid-phase phosphoramidite chemistry, the second integrates high-throughput array-based synthesis driven by microchip advancements, and the third heralds a transformative leap in synthetic biology through enzymatic synthesis [74]. Among high-throughput synthesis methods, photolithographic synthesis stands out for its scalability and cost-effectiveness compared to alternative techniques, but it comes with higher error rates [26] . In a DNA-of-things scenario, deletion, insertion, and substitution errors occur in this method at 0.075 nt^{-1}, 0.012 nt^{-1}, and 0.025 nt^{-1}, respectively, with sequence coverage following a lognormal distribution (mean: 1, SD: 0.44) [26]. There are numerous sequencing methods as well, each optimizing accuracy, speed, and cost differently, leading to varying error profiles. Illumina sequencing is known for its high accuracy but still produces substitution errors, particularly in homopolymer regions [64]. PacBio sequencing offers long reads that aid in resolving complex genomic regions, although it is more prone to errors due to insertions and deletions [57]. Oxford Nanopore Technology (ONT) provides ultra-long reads for specific applications, but also encounters relatively frequent

substitution and indel errors [58]. Due to the inherent errors in synthesis and sequencing, an effective error-correcting code (ECC) is crucial for reliable data recovery [21].

DDS errors are both synchronous and asynchronous with higher-order statistics [38]. As a result, accurately characterizing channel statistics is highly complex and designing ECC is equally challenging [38]. The high cost remains a major bottleneck in advancing DDS research, particularly in the development of efficient encoding, decoding, error detection, and ECCs. In the 1980s, synthesizing 10 nucleotides cost $6,000 [51]. Today, it's under $1 per base [23], but storing a minute of high-quality audio still costs $100,000. Sequencing a human genome now costs $1,000 [10]. Simulation can be a powerful tool for channel verification [32], synthesis technique analysis [41], bound analysis [42], and new ECC scheme evaluation [6] in DDS. DDS error simulation precisely models the intrinsic errors encountered throughout the DNA storage pipeline, including synthesis imperfections, molecular stability fluctuations, PCR-induced biases, and sequencing inaccuracies [3]. Users can calibrate settings based on empirical datasets or leverage biologically validated presets [3]. By integrating mutational biases, degradation kinetics, and amplification-induced distortions, the simulation offers a biologically faithful representation of DDS dynamics [20]. We can efficiently emulate the entire DNA storage process in silico at minimal cost [38]. Simulation also provides a rigorous testing ground for novel error-correction algorithms and enhance the resilience of DDS [75]. But error simulation in DDS is complicated due to various factors. **Firstly**, due to variations in procedures and technologies across experiments, the error variation and its impact on data recovery remain largely unexplored [70]. Both data-dependent and process-dependent errors exhibit biased statistics like uneven k-mer error patterns and transition errors [38, 70]. Yet most errors remain inherently stochastic. The Bias-Variance Tradeoff in generation learning involves balancing errors from two sources: bias, caused by overly simplistic models leading to underfitting, and variance, caused by sensitive models overfitting to small data fluctuations [50]. Reducing bias typically results in increased variance, and vice versa [50]. This dynamic poses a considerable challenge in developing models that both capture biased errors and preserve stochastic properties. Most simulators rely on predefined error probabilities [60], limiting their ability to address stochastic errors. Conversely, stochastic models fail to encompass biased errors, creating a gap in accurate error simulation [4, 62, 75]. **Secondly**, DDS exhibits distinct error behaviors depending on processes. For example, photolithographic synthesis and enzymatic synthesis each present unique error patterns. Simulators are typically designed for specific stages [33] or technologies [43], such as Illumina or Oxford Nanopore. However, when attempting to integrate these stages for end-to-end simulations, existing simulators may fail. This underscores the need for a universal simulator capable of combining different stages and technologies, enabling researchers to streamline diverse DDS methods into cohesive workflows. **Thirdly,** universal modeling remains a hurdle in DDS. For instance, Illumina uses fluorescence for short, precise reads [59] (error rate: 0.1-1%, length: 300 bp), while Nanopore detects electrical signals for long, error-prone reads (error rate: 5-15%, length: >1 kb) [46]. Biochemical fluctuations and signal inconsistencies across technologies and processes introduce complex error distortions that vary significantly between systems [79]. This undermines cross-model adaptation within stochastic frameworks or statistical approaches [75]. Deep learning models specialize in capturing temporal dependencies in sequential data and generates coherent, contextually rich sequences in text, music, and beyond [76]. Deep learning can also be used to capture error patterns in DDS with precision by dynamically adapting and extracting features. A data centric GAN-based simulator has been proposed for DNA channel modeling to enable end-to-end simulation [38]. But, it struggles to learn from Nanopore sequencing due to the intricate nature of long-read data. Our objective is to present a universal DDS error simulation framework that captures advanced statistics and intrinsic stochasticity with accuracy. Our key contributions to replicate natural, real-world lab errors in the DDS are described below:

- We introduce a universal transformer-based generative framework for stochastic error simulation in DDS. Our framework successfully simulates biased error statistics and distributions observed in the DDS pipeline irrespective of process or technology. We successfully demonstrate the robustness of our data-centric universal modelling by simulating error profiles from two distinct datasets across two different technologies.
- To uphold stochasticity aligned with the error statistics, we propose a Gaussian noise-based stochasticity technique which both generates randomness and matches the statistical distribution.
- We compare the proposed transformer-based generative framework with alternative architectures trained by us, as well as existing models from the literature. We analyze each structure and their suitability and weaknesses in the simulation task. To evaluate performance, we quantify base-level statistics, k-mer error patterns, and position-based skewed error distributions across all models.

2 Background and Related Work

Sequential data processing often utilizes Gated Recurrent Units (GRUs) [16] and Long Short-Term Memory (LSTM) networks [31], both of which are variants of Recurrent Neural Networks (RNNs) [56]. But when dealing with both asynchronous and synchronous errors in DDS sequences, especially for long sequences, these models may face significant obstacles. Again, Encoder-decoder combinations were explored [65], [17], [8], [25] for generative learning, but bottlenecks persisted in this domain. To the best of our knowledge, Transformer architectures have not been extensively explored in DDS error simulation. The Transformer [67], proposed by Vaswani et al., revolutionized sequential tasks through attention mechanisms, leading to the rise of pre-trained models like BERT [19] and GPT [53], [54], [22]. These models now lead the way in generating text, images, and so on. The Pre-Layer Normalization Transformer [7] enhanced training stability by integrating layer normalization inside the residual block, improving gradient flow [72]. Pretrained models lack vector embeddings for A, C, T, and G, requiring direct learning from training data. Thus, pretrained models offer no advantage here.

Previous studies extensively characterized each stage from synthesis to sequencing [39, 49, 52, 73] but these single-stage studies can not be directly applied when stages are combined. Hamoum [28] modeled one stage of Nanopore Sequencing accurately but missed end-to-end error evolution. Heckel and Chen [15, 29] studied sequence loss errors, while Schwarz [60] examined within-sequence errors in general context, unsuited for oligo pool DNA storage. Yuan et al. [75] demonstrated an end-to-end stochastic model in DeSP but struggled with complex non-linear noise. Current simulators like ART [34], Flux [27], and pRIS [33] address specific sequencing stages, but they only focus on 1st order statistics. Deep Simulator [43, 44] mimics Nanopore sequencing, improved by NanosigSim [14] with GRU, and Nanopore SimulatION [55] simulates raw current values. Schwarz et al. [60] simulate error sequences with user-defined probabilities. Various studies evaluate the performance of encoding mechanisms [12], report biochemical properties like GC content and melting temperature [61], or simulate digital encoding in bacterial systems [1, 66]. Yet, computational tools that model error propagation during synthesis and preservation in DNA storage are scarce. Most simulators use hard-coded error probabilities or are tailored to specific technologies, such as Illumina [60], Oxford Nanopore [9], or Twist Bioscience. Universal error modeling in DDS through deep learning frameworks is still largely unexplored.

3 Methods

We present a new approach by adapting the Transformer in a Beta-VAE framework. This facilitates Disentangled Error Representation in DNA sequences. We use attention mechanisms to capture error patterns. A Causal Masking strategy preserves sequential dependencies throughout the learning. We add Gaussian noise to induce stochasticity to enhance output randomness and simulate diverse error scenarios. Figure 1 shows an overview of our proposed model.

3.1 Problem Formulation

Data is encoded in oligonucleotides in DDS and later read from synthesized multiple oligos. The goal of the model is to replicate the sequential processes of the DDS, from the input oligo sequence to the output read sequence. Let $X = \{A, C, G, T\}$ represent the set of nucleotide bases, and $X_e = \{A, C, G, T, S, E, P\}$ represent the set of extended symbols, where 'S' is the start marker, 'E' is the end marker, and 'P' is padding. The input to the simulator is a nucleotide sequence, denoted as $x \in X^T$, where $x = x_1 x_2 \dots x_T$, with $x_i \in X$ for $1 \le i \le T$. The output of the simulator is a DNA sequence $\hat{x} \in X_e^M$, referred to as the read, where $\hat{x} = \hat{x}_1 \hat{x}_2 \dots \hat{x}_M$, with $\hat{x}_i \in X_e$ for $1 \le i \le M$. The task is to simulate the DDS process, introduce noise and preserve sequential dependencies. So the final read sequence $\hat{x}$ is an error-prone version of the input sequence x.

3.2 Proposed Stochastic Generative Learning

Inducing Stochasticity with Gaussian Noise in DNA Sequences. In DDS, DNA error-prone sequences must exhibit randomness at each coverage level. We implement a character-to-one-hot encoding scheme to represent sequences in a machine-readable format and assign each character a unique binary vector. We preprocess the sequences by truncating those exceeding a defined length. We introduce noise to the sequences to inject stochasticity into the model. Unbiased synthesis and sequencing processes naturally exhibit Gaussian-like behavior [15]. A case in point is the synthesis stage, where the count of oligos may vary, and the count distribution can be approximated by a normal distribution as dictated by the Central Limit Theorem across different techniques [70]. Modeling noise with a Gaussian distribution aligns well with the inherent stochasticity of oligo generation. The dataset is initialized with original and read sequences, along with a configurable noise factor that dictates the intensity of Gaussian noise. To introduce the noise, we use the equation $y = x + \mathcal{N}(\mu, \sigma^2)$, where y represents the noisy sequence, x is the original sequence, μ is the mean (typically 0), and σ^2 is the variance, which controls the intensity of the noise. Through a random search experiment, we determine an optimal noise factor. We efficiently manage the sequences by calculating their total count and retrieving them by index. Upon access, we convert the sequences into tensors and then apply Gaussian noise while clamping values within the valid one-hot encoding range. Hence, we effectively simulate real-world variability and focus on increased output randomness and stochastic errors, while preventing overfitting.

Proposed Generative model for DNA Error Simulation. Our model is based on Transformer [68] to learn error patterns from DNA sequences. We adapt the Transformer model for generative learning in a Beta-VAE setting [30] to improve the representation of sequence variability. We incorporate a masked attention mechanism into the encoder and decoder, drawing inspiration from its use in music representation learning as demonstrated in [37]. Additionally, we use a causal masking strategy to maintain the sequential dependencies within the sequences. Let $x_{1..T}$ represent the original DNA sequence with T nucleotides, where x_i denotes the i-th nucleotide in the sequence. After noise induction, each nucleotide first passes through a local encoder E_{local} with shared parameters, resulting in nucleotide-level representations $y^e_{1..T}$, where: $y^e_i = E_{\text{local}}(x_i)$. The Transformer encoder then takes $y^e_{1..T}$ as input and calculates the parameters of the latent code z as follows:

$$g^e_{1..T} = E_{\text{Transformer}}(y^e_{1..T})$$

$$[\mu_i, \log(\sigma_i^2)] = W_i g^e_i + b_i \tag{1}$$

$$z_i \sim \mathcal{N}(\mu_i, \sigma_i^2) \tag{2}$$

The latent variable z is then passed to the Transformer decoder $D_{\text{Transformer}}$, along with the representations of previously decoded nucleotides, to decode error-prone DNA sequences sequentially. This strategy is formally defined as:

$$g^d_i = D_{\text{Transformer}}(z_{1:i}, y^d_{0:i-1}) \tag{3}$$

$$\hat{x}_i = D_{\text{local}}(g^d_i) \tag{4}$$

Here, $\hat{x}_{1..T}$ represents the error-prone DNA sequence, and $y^d_i = E_{\text{local}}(\hat{x}_i)$ are the representations of previously decoded nucleotides. A special embedding y^e_0 is used to signify the start of the DNA sequence, functioning similarly to the Start of Sequence (SOS) tag in natural language processing. To avoid data leakage and overfitting, the model is designed not to see error-prone sequences during training?. Once the model generates its output, we independently calculate the loss and provide feedback. It ensures accurate and reliable learning throughout the training.

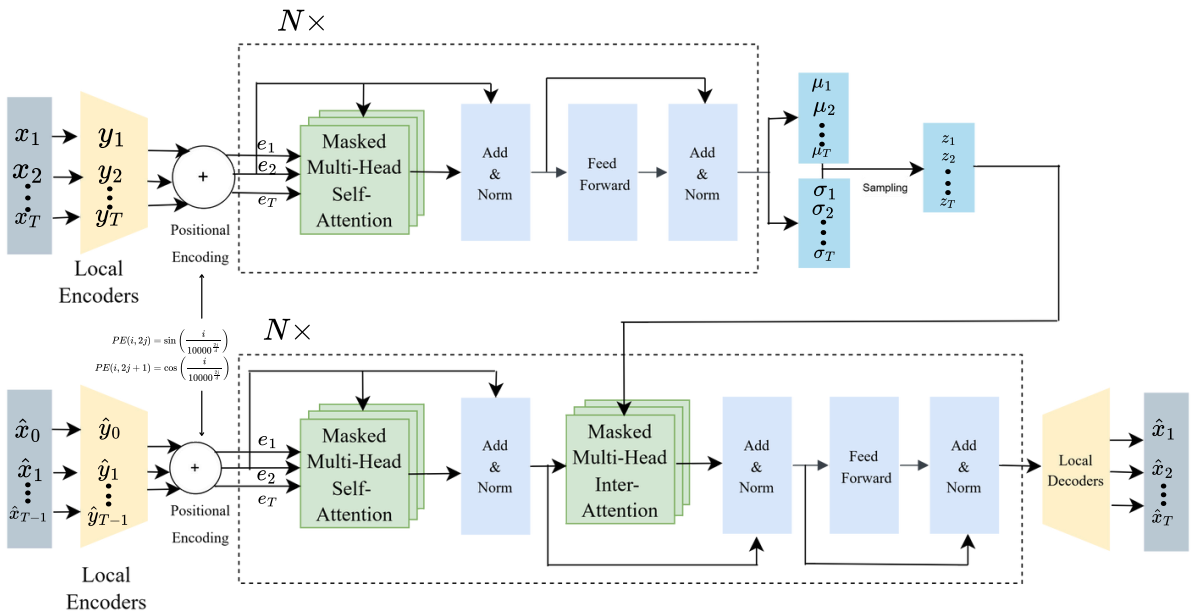

Figure 1: Proposed Transformer-based generative framework for DDS-E-SIM. It uses a Transformer with masked attention in all layers of encoder and decoder and causal masking to optimize sequence modeling.

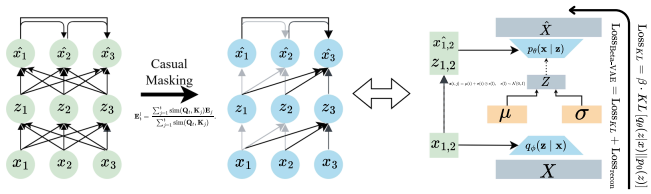

Figure 2: An overview of causal written Casual in the imageI am fixing it masking in the Transformer-VAR architecture is presented. On the left, the changes in the dependency graph after applying casual masking is shown. On the right, we present a theoretically equivalent Beta-VAE model for error simulation.

Causal Masking for Sequential Cohesion Control (CM). In DDS, the nucleotide sequence must follow a precise order to ensure accurate encoding and retrieval. During encoding and decoding, the model, akin to biological transcription, must not access future positions in the sequence to preserve data integrity. For instance, DNA sequences are synthesized in short segments, known as oligonucleotides, and the order of these segments is critical for their correct assembly during synthesis [71]. At the sequencing stage, DNA is read in a defined, sequential order to reconstruct the original data accurately [69]. Any disruption to this sequential order could lead to errors in data retrieval, analogous to mutations in genetic information [69]. To emulate this biological process, we apply causal masking across all layers of the encoder and decoder as shown in Figure 2. It ensures the model follows the inherent directionality and order of the encoded data. This masking mechanism ensures that the i-th base in the sequence is influenced only by bases j where $j \leq i$. Each base is allowed to attend to only its preceding bases, preventing any forward-looking context or data leakage. We implement this operation as a cohesion regulation: (1) the encoder is constrained to access only the context preceding the current position ($x_{1..i}$), and (2) the decoder is restricted to utilizing the context and previously decoded segments up to the current step ($\hat{x}_{1..i-1}$ and $z_{1..i}$). Formally, this causal masking changes the computation as follows:

$$E'_i = \frac{\sum_{j=1}^{i} \mathrm{sim}(Q_i, K_j) E_j}{\sum_{j=1}^{i} \mathrm{sim}(Q_i, K_j)}. \tag{5}$$

Where E'_i is the updated error-prone DNA base at position i. Q_i is the query vector derived from the DNA base at position i. K_j is the key vector derived from the DNA base at position j. E_j is the error probability or representation of the DNA base at position j. $\mathrm{sim}(Q_i, K_j)$ is the similarity score (e.g., dot product or learned metric) between Q_i and K_j, representing how much the DNA base at position i is influenced by the base at position j.

Table 1: Error rates of original and model-generated outputs across datasets D-I and D-N

Model	D-I				D-N			
	Substitution (%)	Insertion (%)	Deletion (%)	Total Error (%)	Substitution (%)	Insertion (%)	Deletion (%)	Total Error (%)
Original	49.4	25.3	25.3	1.5	28.2	35.9	35.9	8.1
DDS-E-SIM	**45.7**	**24.6**	**29.7**	**1.4**	**27.7**	**34.4**	**37.9**	**7.4**
AR-T	51.2	24.4	24.4	2.2	-	-	-	-
MAL-VAE	57.1	21.5	21.5	2.5	-	-	-	-
WGAN [38]	42.4	20.1	37.4	3.2	36.7	30.2	33.1	5.3

Disentangled Error Representation. **Firstly**, DNA sequences are inherently complex because they contain hierarchical and structured dependencies [35]. For example, certain genes are organized into clusters on the DNA strand, such as the Hox gene cluster, which is essential for the coordinated expression of genes during development [35]. Additionally, regulatory elements like enhancers, silencers, and promoters interact in a hierarchical manner to control gene expression [77]. So errors in DNA sequences inherently carry biases and exhibit dominance in certain regions or motifs [2]. Repetitive sequences or regions with high GC content causes sequencing errors. The model needs to capture the variability and error patterns inherent in DNA sequences while distinguishing between different factors of variation, such as k-mer, positional skewness, and dominance of specific transitions. k-mer is a factor? I thought factors are regulatory elements etc. k-mer statistics are results of the variation The Beta-VAE, which introduces the hyperparameter β, helps enforce a more disentangled latent space and allows the model to better understand and control the various types of errors present in DNA sequences.How? Not clear reading first time. By adjusting β, the model can separate these independent factors like k-mer, which is crucial for modeling the distinct error types. **Secondly**, error patterns often depend on the sequence order and structure, and it is crucial to preserve these sequential dependencies [69, 71]. The Beta-VAE facilitates the learning of latent variables that not only capture high-level structure but also model local dependencies within the sequence. This ability to disentangle the latent factors helps preserve the sequential relationships between regulatory elements and genes. So the generated error-prone sequences will retain both the structural and sequential integrity of the original DNA sequences. The objective function can be expressed as:

$$L(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta \cdot D_{KL}(q_\phi(z|x) \| p(z)),$$

where $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$ is the expected log-likelihood of the data under the approximate posterior distribution $q_\phi(z|x)$, $D_{KL}(q_\phi(z|x) \| p(z))$ is the Kullback-Leibler divergence between the approximate posterior $q_\phi(z|x)$ and the prior $p(z)$, β is the hyperparameter that controls the strength of the disentanglement regularization, encouraging the model to learn more independent latent factors.

4 EXPERIMENT

4.1 Dataset Description

We evaluate our model using two distinct datasets from two separate processes. The first dataset (D-I) is based on the oligo design from [36], where synthesis was performed using Twist Bioscience, and sequencing was done with the Illumina Miseq Reagent v3 kit. We handle 18,000 oligos (length 152, GC 45–55%, homopolymer ≤3) as per [38] and obtain 15,126,429 reads. The dataset is split into training (14,400 oligos, 12,108,573 reads) and testing (3,600 oligos, 2,999,656 reads). The second dataset (D-N) contains DNA sequences synthesized by Twist Bioscience and amplified using polymerase chain reaction (PCR) taken from [63]. The PCR products were ligated to Oxford Nanopore Technologies (ONT) sequencing adapters following the LQK-LSK 109 kit protocol, and sequenced using the ONT MinION. It includes 10,000 randomly generated DNA strings of length 110 and 269,709 noisy nanopore reads. We preprocess D-N following the same approach as D-I.

Evaluation Metrics. We determine uneven base-level errors using the concept of hamming distance [11]. Uneven transition errors among nucleotide bases are analyzed using the similar method. We compute the k-mer mismatch rate for 2-mers and 3-mers individually to uncover motif-specific bias in each model. Positional error in the test dataset is examined at each index.

Implementations and Model Details. We implement our model using PyTorch on an NVIDIA A100 GPU. we apply a padding of 5 to standardize input size. Using random search, we determine an optimal noise factor of 0.03 to induce stochasticity effectively. The model features an encoder with a 7-dimensional input, 256 model dimensions, 8 attention heads, 3 layers, a 128-dimensional latent space, a 1024 feedforward dimension, and a 0.1 dropout rate. The decoder mirrors this structure with a 7-dimensional output.

5 Results and Discussion

We first conduct a comparative evaluation of the following methods: (1) DDS-E-SIM: the proposed architecture integrating a transformer-Beta-VAE with causal masking; (2) AR-T: a variation of the proposed architecture using a transformer with causal masking but without Beta-VAE; (3) MAL-VAE: It employs a Multi-Attention-LSTM with three LSTMs and a VAE, detailed architecture in the Appendix A.1; and (4) WGAN: proposed by [38], is as a universal deep learning-based model used for comparison which is the only end-to-end deep learning model to the best of our knowledge that can simulate DDS errors Check the previous modified line.. A comprehensive analysis and ablation study follow, and a brief summary of the adopted datasets is presented in Table 1.

Table 2: The base transition error rates of D-I and different models

	D-I				DDS-E-SIM				MAL-VAE				WGAN			
	A	C	G	T	A	C	G	T	A	C	G	T	A	C	G	T
A	–	**0.47**	0.36	0.17	–	**0.42**	0.39	0.19	–	0.25	**0.49**	0.26	–	0.30	0.30	**0.39**
C	**0.35**	–	0.31	0.34	**0.68**	–	0.15	0.17	**0.86**	–	0.06	0.08	0.23	–	0.06	**0.71**
G	0.38	0.10	–	**0.53**	**0.54**	0.12	–	0.34	**0.67**	0.13	–	0.21	0.36	0.07	–	**0.57**
T	0.33	0.18	**0.49**	–	0.36	0.18	**0.45**	–	0.34	0.23	**0.43**	–	0.22	**0.59**	0.20	–

5.1 Performance on D-I Benchmark

Uneven Base-Level Error Statistics. We first examine the single-base error profile of the proposed model and compare it with the error statistics of actual data, as shown in Table 1. Additionally, we compare these results with other models we developed and baseline models. Substitution is the dominant error type for D-I shown in each models. DDS-E-SIM exhibits the smallest deviation from the original data, with error rate differences of 3.7% in substitution, 0.7% in insertion, and 4.4% in deletion. In contrast, WGAN shows greater deviations of 7%, 5.2%, and 12.1%, respectively. The total error probability in DDS-E-SIM is 1.4%, closely aligning with the original 1.5%, while AR-T, MAL-VAE, and WGAN have significantly higher error rates of 2.2%, 2.5%, and 3.2%, respectively. Furthermore, we analyze base transition errors, as this could help design more efficient error-correcting codes and facilitate decoding. The transition probabilities of each base are shown in Table 2, where the row labels denote the transmitting bases, and the column labels represent the transmitted bases. The bold values indicate the transitions with the highest probability for the four reference nucleotides. From the table, we observe that transitions such as A to C, C to A, G to A, and T to G exhibit the highest probabilities for each reference base. These results closely mirror the original data statistics, with the exception of base G, where the G to T transition is the most prominent in the original dataset. Each of these highest transitions is nearly 2-fold than other two transitions. However, WGAN fails to capture dominant error transitions, particularly A to T, C to T, and T to C, with a bias toward T. MAL-VAE misses the dominance of two base-level transitions, only capturing C to A and T to G.

Table 3: K-mer error rates for D-I, D-N and Different Models

Model	AA	CC	TT	GG	CCC	AAA	GGG	TTT
D-I	0.05	0.05	0.01	0.05	0.07	0.49	0.17	0.14
DDS-E-SIM	0.07	0.05	0.01	0.03	0.07	0.41	0.19	0.16
D-N	0.05	0.06	0.08	0.15	0.32	0.09	0.14	0.08
DDS-E-SIM	0.04	0.06	0.11	0.07	0.31	0.15	0.17	0.09

Uneven k-mer Error Patterns. We analyze total 2-mer and 3-mer error rates to identify motif-specific biases in both the original dataset and DDS-E-SIM in Table 3. In DDS-E-SIM, the 2-mer and 3-mer error ratios align with dataset D-I, where 3-mer errors exceed 2-mer. Among them, AAA shows the highest error tendency, while TT has the lowest in both dataset and DDS-E-SIM. We further examine the individual error trends. For deletions, the most error-prone 2-mer patterns for all bases are homopolymer repeats (AA, CC, TT, GG). In contrast, insertions are more frequent in 2-mer patterns with specific combinations: AG, CG, TG, and GA for A, C, T, and G, respectively. Substitution error rates show minimal variation across patterns, but the most erroneous 2-mer patterns for each base are also homopolymer repeats, with G-oriented patterns exhibiting the highest substitution rates, consistent with single-base error trends. In 3-mer analysis, deletions are again prominent in homopolymer repeats (AAA, CCC, TTT, GGG), aligning with 2-mer deletion trends. However, 3-mer repetitions show higher deletion tendencies, indicating longer homopolymers are more deletion-prone. Insertions frequently occur in patterns like AGG, CGG, TGG, and GAA, with prefixes matching the most error-prone 2-mers (AG, CG, TG, GA), implying strong influence from neighboring bases. Substitutions in 3-mer patterns with the first nucleotide altered mirror the errors seen in 2-mers.

Positional Skewness in DDS-E-SIM Error Distribution. The DDS-E-SIM model, driven by its inherent stochastic nature, generates distinct sequence variations. We show that the positional distribution of errors at the nucleotide level fluctuates with each coverage, yet we find that the overall error distribution remains statistically consistent in Figure 3. This consistency ensures robust predictive accuracy over multiple iterations. Insertion errors exhibit a relatively smooth distribution. However, we find that pronounced spikes appear at the terminal regions of the sequences in both the orginal and generated sequences. In contrast, substitution errors display a chaotic, erratic pattern with no discernible order across indices. Deletion errors present a more structured behavior. We find that while deletion rates remain low across most of the sequence, they surge dramatically at the terminal regions. This is similar to the insertion pattern observed earlier. Despite the variations in error rates across different error types, the generated sequences closely match the overall error distribution of orginal dataset.

Adaptive Stochasticity in DNA sequeneces. Though DDS-E-SIM incorporates universal error statistics of D-I, its output is expected to be stochastic. It generates diverse sequences for the same input, with the distribution aligning with experimental data at each coverage. To induce stochasticity, we add Gaussian noise. To verify randomness in oligos, we generate reads from 35,329 fixed sequences, resulting in a total of 100,743 unique oligos at coverage 5. To assess potential overfitting or data leakage, we analyze positional error distributions, as shown in Figure 4. Base variations between the original dataset and the generated output indicate that our model avoids overfitting and data leakage. Thus, we not only validate the uniqueness across the entire DNA strand but also assess the index level deviation for individual base level. When producing different output read sequences from a single input sequence, the

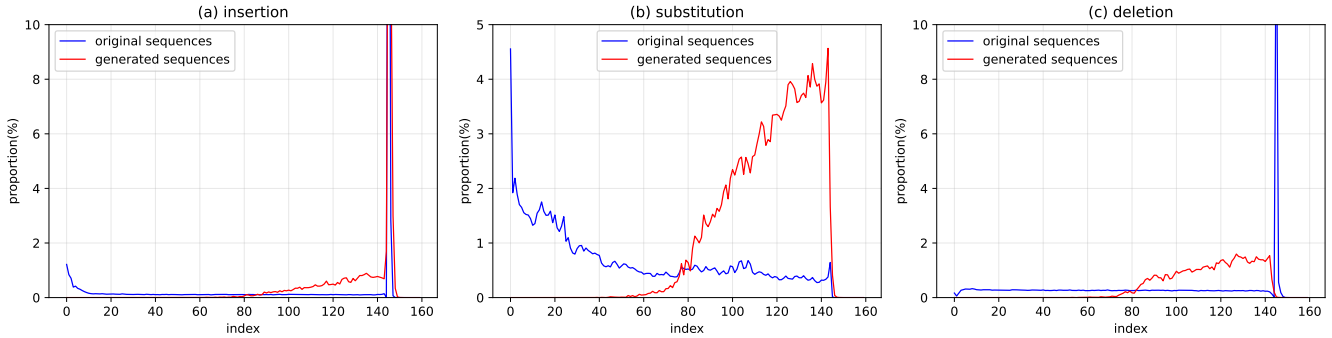

Figure 3: Insertion, substitution, and deletion error rates at each index

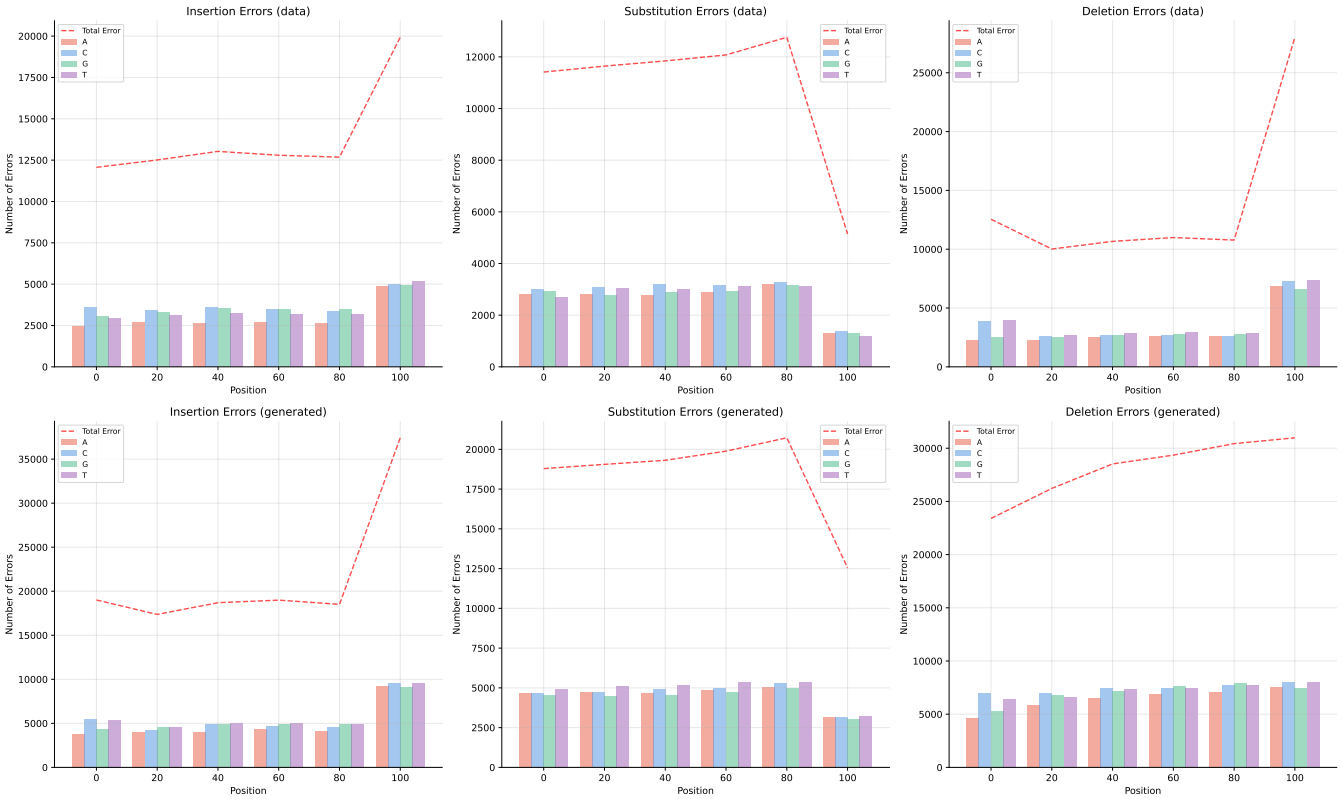

Figure 4: Stochastic insertion, substitution, and deletion errors in generated reads from a fixed single oligo in case of D-I. The x-axis represents the error index, with different colors denoting various erroneous bases. The dotted line indicates the total number of errors for all bases at each index.

exact error locations diverge unpredictably for each base, illustrating the intrinsic randomness of the model. However, AR-T produces 48,751 random error sequences, MAL-VAE generates 52,456, while WGAN generates 60,743 unique oligos, significantly fewer than the 100,743 generated by DDS-E-SIM at coverage 5.

5.2 Performance on D-N Benchmark

We further validate the universality of our model on a different dataset derived from a distinct process as shown in Table 1. DDS-E-SIM achieves stable convergence with minimal deviation—0.5% in substitution, 1.5% in insertion, and 2% in deletion. In contrast, WGAN shows significant deviation—8.5% in substitution, 5.7% in insertion, and 2.8% in deletion. The total error rate is 8.1% for the original model, 7.4% for DDS-E-SIM, and 5.3% for WGAN. Our model

also identifies the dominant base transition patterns, as shown in Table 4. We evaluate error rates for 2-mers and 3-mers in Table 3. DDS-E-SIM closely mirrors the error ratios of the D-N dataset in most cases. Both datasets show that CCC (0.32 in D-N, 0.31 in DDS-E-SIM) exhibits the highest error rate, while AA is the least error-prone. Additionally, 3-mers tend to have a higher error tendency compared to 2-mers. We assess the randomness of each base at every position similar to D-I in Figure 6 in Appendix.

Table 4: The transition error rates of D-N and DDS-E-SIM

	D-N				DDS-E-SIM			
	A	C	G	T	A	C	G	T
A	–	0.22	**0.58**	0.19	–	0.22	**0.57**	0.21
C	0.21	–	0.17	**0.62**	0.21	–	0.17	**0.62**
G	**0.61**	0.18	–	0.22	**0.59**	0.17	–	0.24
T	0.19	**0.61**	0.20	–	0.20	**0.59**	0.21	–

5.3 Performance Evaluation and Ablation Study

Universality Analysis. We evaluate our model on two distinct datasets derived from separate processes. As shown in Table 1, 2, 4, our model consistently outperforms other baselines and provides superior results across both datasets. WGAN, in contrast, faces significant challenges with the D-N dataset, due to its inability to account for higher error rates. This failure stems from reliance on three distinct generators designed to accommodate the inherent bias in a specific process, where substitution dominates. WGAN employs GRU units for insertion and deletion generation and Transformer models for substitution to address this bias [38]. However, the D-N dataset deviates from this pattern, where substitution errors do not dominate, exposing the limitations of WGAN. In contrast, DDS-E-SIM operates as a data-centric, process-independent model built within a unified deep learning framework. Our model uses a single transformer to handle all error types and ensures robustness regardless of error type dominance. Transformer handles long-range dependencies more effectively due to their self-attention mechanism which LSTMs in MAL-VAE or GRUs in WGAN can not replicate efficiently. Unlike WGAN, which is highly specialized for a particular dataset structure, our architecture is agnostic to any specific process. DDS-E-SIM is inherently more universal and adaptable to diverse error landscapes across different systems.

Non-Deterministic Error Modeling. We experiment with (1) DDS-E-SIM with Beta-VAE and (2) AR-T without Beta-VAE, maintaining the same Transformer. Both models successfully replicate error statistics from wet lab experiments (Table 1). AR-T generates 48,751 random error sequences, significantly fewer than the 100,743 produced by DDS-E-SIM. The deterministic mapping of AR-T lacks inherent randomness, whereas DDS-E-SIM leverages VAE-driven probabilistic mapping to achieve smooth latent space interpolations and preserve randomness. Causal masking in Encoder and Decoder of Transformer enforces a strict unidirectional attention flow and prevents information leakage from future tokens. They effectively reduce overfitting, as demonstrated in Figure 4.

Noise Injection for stochasticity. We experiment with adding noise in two ways to increase randomness in generated outputs at different coverages and prevent overfitting. In the first method (1), we add noise at the preprocessing stage of DNA sequences before they enter the local encoder. we observe an error rate of only 1.4%, with expected randomness shown in Figure 4. In the second method (2), we introduce noise directly in the latent space of a Beta-VAE. We sample the latent variable z from a Gaussian distribution and add 3-5% noise to regularize the model. However, this method leads to an unexpectedly high error rate of 15.1%. The noise disrupts the latent space and prevents accurate sequence reconstruction. We observe repetitive homopolymeric errors, particularly in long consecutive sequences of A, G, T, and C, especially at the terminal.

We also experiment with a trimming method on the DNA sequence dataset to remove low-quality regions using a sliding window approach (window size = 5, threshold = 35). The window slides across each sequence, flagging regions below the threshold for trimming. We determine the first and last positions where quality remains above 35 and ensure a minimum sequence length of 50 bases. Sequences below this length are discarded. Given the average sequence length of 152, which is manageable for the transformer, we exclude this method from our main proposed structure for now.

. While both DDS-E-SIM and the wet lab dataset exhibit a higher error tendency in homopolymers, MAL-VAE fails to capture uneven k-mer error patterns, as seen in Table **??**. The LSTM-based structure struggles to reflect long-range dependencies and attention to specific sequence positions. This limitation results in a failure to capture k-mer patterns effectively.

6 Conclusion

In this work, we present DDS-E-SIM, a generative framework for simulating error-prone sequences in DDS. By combining a Transformer-based generative model with causal masking and adaptive stochasticity techniques, DDS-E-SIM captures higher-order error statistics in DNA sequences. Our results demonstrate state-of-the-art performance, effectively address data leakage, and accurately model error statistics and stochasticity.

DDS-E-SIM has limitations. Firstly, the full spectrum of temporal and spatial error patterns remains unexplored, and additional wet lab datasets are needed to improve its robustness across different stages. Secondly, the current model lacks modules for distinct stages. Future research will focus on developing stage-based modular architectures to simulate error-prone sequences at each step of the DDS process. This will enable real-life implementation as part of pipelines. Thirdly, error correction algorithms have not yet been tested, and their potential improvements using our simulation model remain to be explored. DDS-E-SIM offers a promising method for simulating error-prone sequences in DDS. Future work will focus on enhancing robustness, implementing stage-based modules, and integrating error correction, paving the way for real-world pipeline applications.

References

[1] Amay Agrawal, Dixita Limbachiya, Ravikumar M., Taslimarif Saiyed, and Manish K. Gupta. 2019. Bacsoft: A tool to archive data on bacteria. *CoRR* abs/1903.01902 (2019).

[2] Manuel Allhoff, Alexander Schönhuth, Marcel Martin, Ivan G Costa, Sven Rahmann, and Tobias Marschall. 2013. Discovering motifs that induce sequencing

errors. *BMC Bioinformatics* 14, S1 (2013). doi:10.1186/1471-2105-14-S1-S1

[3] Jamie J. Alnasir, Thomas Heinis, and Louis Carteron. 2022. DNA Storage Error Simulator: A Tool for Simulating Errors in Synthesis, Storage, PCR and Sequencing. arXiv:2205.14437 [q-bio.QM] https://arxiv.org/abs/2205.14437

[4] Jamie J. Alnasir, Thomas Heinis, and Louis Carteron. 2022. DNA Storage Error Simulator: A Tool for Simulating Errors in Synthesis, Storage, PCR and Sequencing. arXiv:2205.14437 [q-bio.QM] https://arxiv.org/abs/2205.14437

[5] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein Generative Adversarial Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 214–223. https://proceedings.mlr.press/v70/arjovsky17a.html

[6] K.E. Baddour and N.C. Beaulieu. 2005. Autoregressive modeling for fading channel simulation. *IEEE Transactions on Wireless Communications* 4, 4 (2005), 1650–1662. doi:10.1109/TWC.2005.850327

[7] A. Baevski and M. Auli. 2019. Adaptive input representations for neural language modeling. In *Proc. 7th ICLR*. New Orleans, LA, USA, 1–13.

[8] D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. 3rd ICLR*. San Diego, CA, USA, 1–22.

[9] David R Bentley, Shankar Balasubramanian, Harold P Swerdlow, Gordon P Smith, Jane Milton, Cheryl G Brown, Kevin P Hall, Dominic J Evers, Chris L Barnes, Hannah R Bignell, et al. 2008. Accurate whole human genome sequencing using reversible terminator chemistry. *Nature* 456, 7218 (2008), 53–59.

[10] Twist Bioscience. n.d.. Genes. Accessed: [Date Accessed].

[11] Abraham Bookstein, Vladimir A Kulyukin, and Timo Raita. 2002. Generalized hamming distance. *Information Retrieval* 5 (2002), 353–375.

[12] James Bornholt, Randolph Lopez, Douglas M. Carmean, Luis Ceze, Georg Seelig, and Karin Strauss. 2016. A dna-based archival storage system. In *ASPLOS'16, Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems* (Atlanta, Georgia, USA). ACM, 637–649.

[13] Luis Ceze, Jeff Nivala, and Karin Strauss. 2019. Molecular digital data storage using DNA. *Nature Reviews Genetics* 20 (2019), 456 – 466. https://api.semanticscholar.org/CorpusID:148570002

[14] Wenhui Chen, Pengfei Zhang, Lei Song, Jian Yang, and Chunyu Han. 2020. Simulation of nanopore sequencing signals based on BiGRU. *Sensors* 20, 24 (Dec 2020), 7244.

[15] Y-J Chen, CN Takahashi, L Organick, C Bee, SD Ang, P Weiss, B Peck, G Seelig, L Ceze, and K Strauss. 2020 journal=Nature Publishing Group. Quantifying molecular bias in DNA data storage. *Nature Communications* 11, 1 (2020 journal=Nature Publishing Group), 1–9.

[16] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl.* 103–111.

[17] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*. 1724–1734.

[18] George M. Church, Yuan Gao, and Sriram Kosuri. 2012. Next-generation digital information storage in dna. *Science* 337, 6102 (2012), 1628.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. 2019 Conf. North Amer.* 4171–4186.

[20] Sanket Doshi, Mihir Gohel, and Manish K. Gupta. 2024. A Bird-Eye View on DNA Storage Simulators. *arXiv preprint* 2404.04877v1 (April 2024). arXiv:2404.04877 [cs.IT] https://arxiv.org/abs/2404.04877 License: arXiv.org perpetual non-exclusive license.

[21] Yaniv Erlich and Dina Zielinski. 2016. DNA Fountain enables a robust and efficient storage architecture. *Science* 355 (2016), 950 – 954. https://api.semanticscholar.org/CorpusID:13470340

[22] T. Brown et al. 2020. Language models are few-shot learners. In *Proc. Adv. Neural Inf. Process. Syst.*, Vol. 33. 1877–1901.

[23] eurofins Genomics. 2017. Price list.

[24] Andy Extance. 2016. How DNA could store all the world's data? *Nature* 537 (2016), 22–24.

[25] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proc. 34th ICML*. 1243–1252.

[26] Andreas L. Gimpel, Wendelin J. Stark, Reinhard Heckel, and Robert N. Grass. 2024. Challenges for error-correction coding in DNA data storage: photolithographic synthesis and DNA decay. *Digital Discovery* 3 (2024), 2497–2508. Issue 12. doi:10.1039/D4DD00220B

[27] Thasso Griebel, Benedikt Zacher, Paolo Ribeca, Emanuele Raineri, Vincent Lacroix, Roderic Guigó, and Michael Sammeth. 2012. Modelling and simulating generic RNA-seq experiments with the flux simulator. *Nucleic Acids Research* 40, 20 (Nov 2012), 10073–10083.

[28] B Hamoum, E Dupraz, E Conde-Canencia, and D Lavenier. 2021. Channel model with memory for DNA data storage with nanopore sequencing. In *2021 11th International Symposium on Topics in Coding (ISTC)*. IEEE, 1–5.

[29] R Heckel, G Mikutis, and RN Grass. 2019. A characterization of the DNA data storage channel. *Scientific Reports* 9, 1 (2019), 1–12.

[30] Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)* 3 (2017).

[31] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (1997), 1735–1780.

[32] Syed Mahamud Hossein, Tabatabaei Yazdi, Yongbo Yuan, Jian Ma, Huimin Zhao, and Olgica Milenkovic. 2015. A Rewritable, Random-Access DNA-Based Storage System. *Scientific Reports* 5 (2015). https://api.semanticscholar.org/CorpusID:14734380

[33] Xuan Hu, Jian Yuan, Yuanzhong Shi, Jing Lu, Baozhen Liu, Zhijian Li, Yingrui Chen, Dezhi Mu, Hongyan Zhang, Na Li, et al. 2012. PIRS: Profile-based Illumina pair-end reads simulator. *Bioinformatics* 28, 11 (Jun 2012), 1533–1535. doi:10.1093/bioinformatics/bts187

[34] Wenhan Huang, Li Li, James R Myers, and Gabor T Marth. 2012. ART: A next-generation sequencing read simulator. *Bioinformatics* 28, 4 (2012), 593–594.

[35] Balaji VS Iyer, Martin Kenward, and Gaurav Arya. 2011. Hierarchies in eukaryotic genome organization: Insights from polymer theory and simulations. *BMC Biophysics* 4 (2011). doi:10.1186/2041-1480-4-8

[36] J. Jeong, S. J. Park, J. W. Kim, J. S. No, H. H. Jeon, J. W. Lee, A. No, S. Kim, and H. Park. 2021. Cooperative sequence clustering and decoding for DNA storage system with fountain codes. *Bioinformatics* 37, 19 (Oct 11 2021), 3136–3143. doi:10.1093/bioinformatics/btab246

[37] Junyan Jiang, Gus G. Xia, Dave B. Carlton, Chris N. Anderson, and Ryan H. Miyakawa. 2020. Transformer VAE: A Hierarchical Model for Structure-Aware and Interpretable Music Representation Learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 516–520. doi:10.1109/ICASSP40776.2020.9054554

[38] Sanghoon Kang, Yunfei Gao, Jaeho Jeong, Seong-Joon Park, Jae-Won Kim, Jong-Seon No, Hahyeon Jeon, Jeong Wook Lee, Sunghwan Kim, Hosung Park, and Albert No. 2023. Generative Adversarial Networks for DNA Storage Channel Simulator. *IEEE Access* 11 (Jan 2023), 123456–123465. doi:10.1109/ACCESS.2023.3235201 Received 28 November 2022, accepted 31 December 2022, date of publication 9 January 2023, date of current version 12 January 2023.

[39] Sriram Kosuri and George M Church. 2014. Large-scale de novo DNA synthesis: technologies and applications. *Nature Methods* 11, 5 (2014), 499–507.

[40] Dave Landsman, Karin Strauss, and Tom Coughlin. 2023. The DNA Data Storage Model. *Computer* 56, 7 (July 2023), 78–85. doi:10.1109/MC.2023.3272188

[41] Henry H. Lee, Reza Kalhor, Naveen Goela, Jean-Chrysostome Bolot, and George M. Church. 2019. Terminator-free template-independent enzymatic DNA synthesis for digital information storage. *Nature Communications* 10 (2019). https://api.semanticscholar.org/CorpusID:174807463

[42] Andreas Lenz, Paul H. Siegel, Antonia Wachter-Zeh, and Eitan Yaakobi. 2018. Coding over Sets for DNA Storage. *CoRR* abs/1812.02936 (2018). arXiv:1812.02936 http://arxiv.org/abs/1812.02936

[43] Yu Li, Ruoyu Han, Chengdong Bi, Min Li, Shuaishuai Wang, and Xin Gao. 2018. DeepSimulator: A deep simulator for Nanopore sequencing. *Bioinformatics* 34, 17 (2018), 2899–2908.

[44] Yu Li, Shuaishuai Wang, Chengdong Bi, Zhiwei Qiu, Min Li, and Xin Gao. 2020. DeepSimulator1.5: A more powerful, quicker and lighter simulator for nanopore sequencing. *Bioinformatics* 36, 8 (Apr 2020), 2578–2580.

[45] Shuguang Liu, Zhenxing Ye, and Maolong Chen. 2020. Next Generation Information Storage Technology: DNA Data Storage. In *Proceedings of the 2020 12th International Conference on Machine Learning and Computing* (Shenzhen, China) *(ICMLC '20)*. Association for Computing Machinery, New York, NY, USA, 213–217. doi:10.1145/3383972.3383999

[46] Wang Liu-Wei, Wiep van der Toorn, Patrick Bohn, Martin Hölzer, Redmond P Smyth, and Max von Kleist. 2024. Sequencing accuracy and systematic errors of nanopore direct RNA sequencing. *BMC genomics* 25, 1 (2024), 528.

[47] Chaoyue Ma, Ying Wang, Feng Li, Haiyan Zhang, et al. 2024. Constructing Attention-LSTM-VAE Power Load Model Based on Multiple Features. *Advances in Mathematical Physics* 2024 (2024), 1041791. doi:10.1155/2024/1041791

[48] Bernard Marr. 2018. How much data do we create every day? The mind-blowing stats everyone should read.

[49] Michael L Metzker. 2009. Sequencing technologies—the next generation. *Nature Reviews Genetics* 11, 1 (2009), 31–46.

[50] Saeed Mirshekari. 2024. Bias vs. Variance: How to Balance Model Performance in Machine Learning. https://saeedmirshekari.com/blog/bias-vs-variance-how-to-balance-model-performance-in-machine-learning/ Accessed: 2025-01-30.

[51] Catherine Offord. 2017. Making DNA data storage a reality.

[52] W Pan, M Byrne-Steele, C Wang, S Lu, S Clemmons, RJ Zahorchak, and J Han. 2014. DNA polymerase preference determines PCR priming efficiency. *BMC Biotechnology* 14, 1 (2014), 1–17.

[53] A. Radford, L. Metz, and S. Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* 1511.06434 (2015).

[54] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. *Language models are unsupervised multitask learners.* Technical Report. OpenAI Blog. 9 pages.

[55] Christoph Rohrandt, Nikolai Kraft, Philipp Gieselmann, Benedikt Brandl, Benedikt M Schuldt, Ulrich Jetzek, and Franz-Josef Müller. 2018. Nanopore SimulatION—A raw data simulator for nanopore sequencing. In *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM).* 1–8.

[56] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323 (Oct. 1986), 533–536.

[57] E Sacristán-Horcajada, S González de la Fuente, R Peiró-Pastor, F Carrasco-Ramiro, R Amils, J M Requena, J Berenguer, and B Aguado. 2021. ARAMIS: From systematic errors of NGS long reads to accurate assemblies. *Journal Name* (2021). Article notes, copyright and license information.

[58] K. Sahlin and P. Medvedev. 2021. Error correction enables use of Oxford Nanopore technology for reference-free transcriptome analysis. *Nature Communications* 12, 2 (2021). doi:10.1038/s41467-020-20340-8

[59] Melanie Schirmer, Rosalinda D'Amore, Umer Z. Ijaz, Neil Hall, and Christopher Quince. 2016. Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data. *BMC Bioinformatics* 17 (March 11 2016), 125. doi:10.1186/s12859-016-0976-y

[60] Marcus Schwarz, Marc Welzel, Tomiris Kabdullayeva, Annika Becker, Bernd Freisleben, and Dominik Heider. 2020. MESA: Automated assessment of synthetic DNA fragments and simulation of DNA synthesis, storage, sequencing and PCR errors. *Bioinformatics* 36, 11 (Jun 2020), 3322–3326.

[61] Shalin Shah, Dixita Limbachiya, and Manish K. Gupta. 2013. DNACloud: A tool for storing big data on DNA. *CoRR* abs/1310.6992 (2013).

[62] SNIA. 2022. DNASSIM Full System Simulator for DNA Storage. https://www.snia.org/educational-library/dnassim-full-system-simulator-dna-storage-2022 Accessed: 2025-01-30.

[63] Sundara Rajan Srinivasavaradhan, Sivakanth Gopi, Henry D. Pfister, and Sergey Yekhanin. 2024. Trellis BMA: Coded Trace Reconstruction on IDS Channels for DNA Storage. arXiv:2107.06440 [cs.IT] https://arxiv.org/abs/2107.06440

[64] Nicholas Stoler and Anton Nekrutenko. 2021. Sequencing error profiles of Illumina sequencing instruments. *NAR Genomics and Bioinformatics* 3, 1 (Mar 2021), lqab019. doi:10.1093/nargab/lqab019

[65] I. Sutskever, O. Vinyals, and Q. V. Le. 2014. Sequence to sequence learning with neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, Vol. 27. 1–15.

[66] Toshinori Suzuki, Shinzo Ohsumi, and Keisuke Makino. 1994. Mechanistic studies on depurination and apurinic site chain breakage in oligodeoxyribonucleotides. *Nucleic Acids Research* 22, 23 (1994), 4997–5003.

[67] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *Proc. Adv. Neural Inf. Process. Syst.* 5998–6008.

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems*, Vol. 30. 5998–6008.

[69] C. Wang, G. Ma, D. Wei, et al. 2022. Mainstream encoding–decoding methods of DNA data storage. *CCF Transactions on High Performance Computing* 4 (2022), 23–33. Issue March 2022. doi:10.1007/s42514-022-00094-z

[70] Yixin Wang, Md. Noor-A-Rahim, Erry Gunawan, Yong L. Guan, and Chueh L. Poh. 2023. Modelling, Characterization of Data-Dependent and Process-Dependent Errors in DNA Data Storage. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 20, 3 (2023), 2147–2158. doi:10.1109/TCBB.2022.3233914

[71] Y. Wang, M. Noor-A-Rahim, J. Zhang, et al. 2019. High capacity DNA data storage with variable-length Oligonucleotides using repeat accumulate code and hybrid mapping. *Journal of Biological Engineering* 13, 1 (2019), 89. doi:10.1186/s13036-019-0211-2

[72] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu. 2020. On layer normalization in the transformer architecture. In *Proc. 37th PMLR.* 10524–10533.

[73] SMHT Yazdi et al. 2015. DNA-based storage: trends and methods. *arXiv preprint arXiv:1507.01611* (2015).

[74] Meng Yu, Xiaohui Tang, Zhenhua Li, Weidong Wang, Shaopeng Wang, Min Li, Qiuliyang Yu, Sijia Xie, Xiaolei Zuo, and Chang Chen. 2024. High-throughput DNA synthesis for data storage. *Chem. Soc. Rev.* 53 (2024), 4463–4489. Issue 9. doi:10.1039/D3CS00469D

[75] Lekang Yuan, Zhen Xie, Ye Wang, and Xiaowo Wang. 2022. DeSP: a systematic DNA storage error simulation pipeline. *BMC Bioinformatics* 23, 1 (2022), 185. doi:10.1186/s12859-022-04723-w

[76] S. Zgür and M. Orman. 2023. Application of deep learning technique in next generation sequence experiments. *J Big Data* 10 (2023), 160. doi:10.1186/s40537-023-00838-w Received: 18 September 2022; Accepted: 03 October 2023; Published: 17 October 2023.

[77] Zecheng Zhang, Chunxiuzi Liu, Yingjun Zhu, Lu Peng, Weiyi Qiu, Qianyuan Tang, He Liu, Ke Zhang, and Zengru Di. 2024. Evolutionary tinkering enriches the hierarchical and nested structures in amino acid sequences. *Phys. Rev. Research* 6 (2024), 023215. doi:10.1103/PhysRevResearch.6.023215

[78] Victor Zhirnov, Reza M. Zadegan, Gurpreet S. Sandhu, George M. Church, and William L. Hughes. 2016. Nucleic acid memory. *Nature Materials* 15, 4 (2016), 366–370. doi:10.1038/nmat4524

[79] Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. 2021. Long-short transformer: Efficient transformers for language and vision. *Advances in neural information processing systems* 34 (2021), 17723–17736.

A Appendix

A.1 Additional Models Implementation

Multi-Attention-LSTM-Beta VAE (MAL-VAE) . We introduce the Beta-VAE with Attention-LSTM, designed to efficiently handle high-dimensional data as shown in Figure 5. In the encoder, the DNA sequence is processed using Attention-LSTM, which effectively captures the complex error statistics in the DNA sequence. In the decoder, Attention-LSTM layers are used to model dependencies between positions in the DNA sequence. It preserves the sequence dynamics in the reconstructed error-prone output.

The input data is represented as $x = [x_1, x_2, \ldots, x_n]$, where x_n denotes the nucleotide at the n-th position in the DNA sequence. The corresponding output data is represented as $y = [y_1, y_2, \ldots, y_n]$, and the reconstructed error-prone output after the model processes the data is denoted as $\hat{y} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n]$. During the encoding process, we obtain the mean $\mu = [\mu_1, \mu_2, \ldots, \mu_n]$ and the variance $\sigma = [\sigma_1, \sigma_2, \ldots, \sigma_n]$ that characterize the latent variables. The latent variables $z = [z_1, z_2, \ldots, z_n]$ are then sampled, with $\epsilon = [\epsilon_1, \epsilon_2, \ldots, \epsilon_n]$ representing the noise used in the reparameterization trick, where $\epsilon \sim N(0, 1)$.

The key difference in Beta-VAE is the introduction of a beta parameter, β, that controls the trade-off between the reconstruction loss and the KL divergence. This parameter is introduced in the loss function to enforce a more structured and disentangled representation of the latent space. Thus, the KL divergence in the Beta-VAE is modified to:

$$\text{Loss}_{KL} = \beta \cdot KL[q_\theta(z|x) \| p_0(z)] \tag{6}$$

where β is a hyperparameter that determines the relative importance of the KL term compared to the reconstruction loss. When $\beta = 1$, the Beta-VAE becomes equivalent to the standard VAE. Increasing β enforces more disentanglement at the cost of reconstruction accuracy, while decreasing β gives more importance to reconstruction.

To obtain the posterior distribution of the latent variables $p(z|x)$, we use variational inference. The true posterior distribution is approximated by a variational distribution $q_\theta(z|x)$, which is parameterized by θ. The goal is to minimize the Kullback-Leibler (KL) divergence between the true posterior $p(z|x)$ and the variational distribution $q_\theta(z|x)$, which can be expressed as:

$$p(z|x) = \frac{p_0(z)P(x|z)}{P(x)} \tag{7}$$

The KL divergence is computed as:

$$KL[q_\theta(z|x) \| p(z|x)] = -\int q_\theta(z|x) \log \frac{p(z,x)}{q_\theta(z|x)} dz + \log P(x) \tag{8}$$

The Evidence Lower Bound (ELBO) is formulated as:

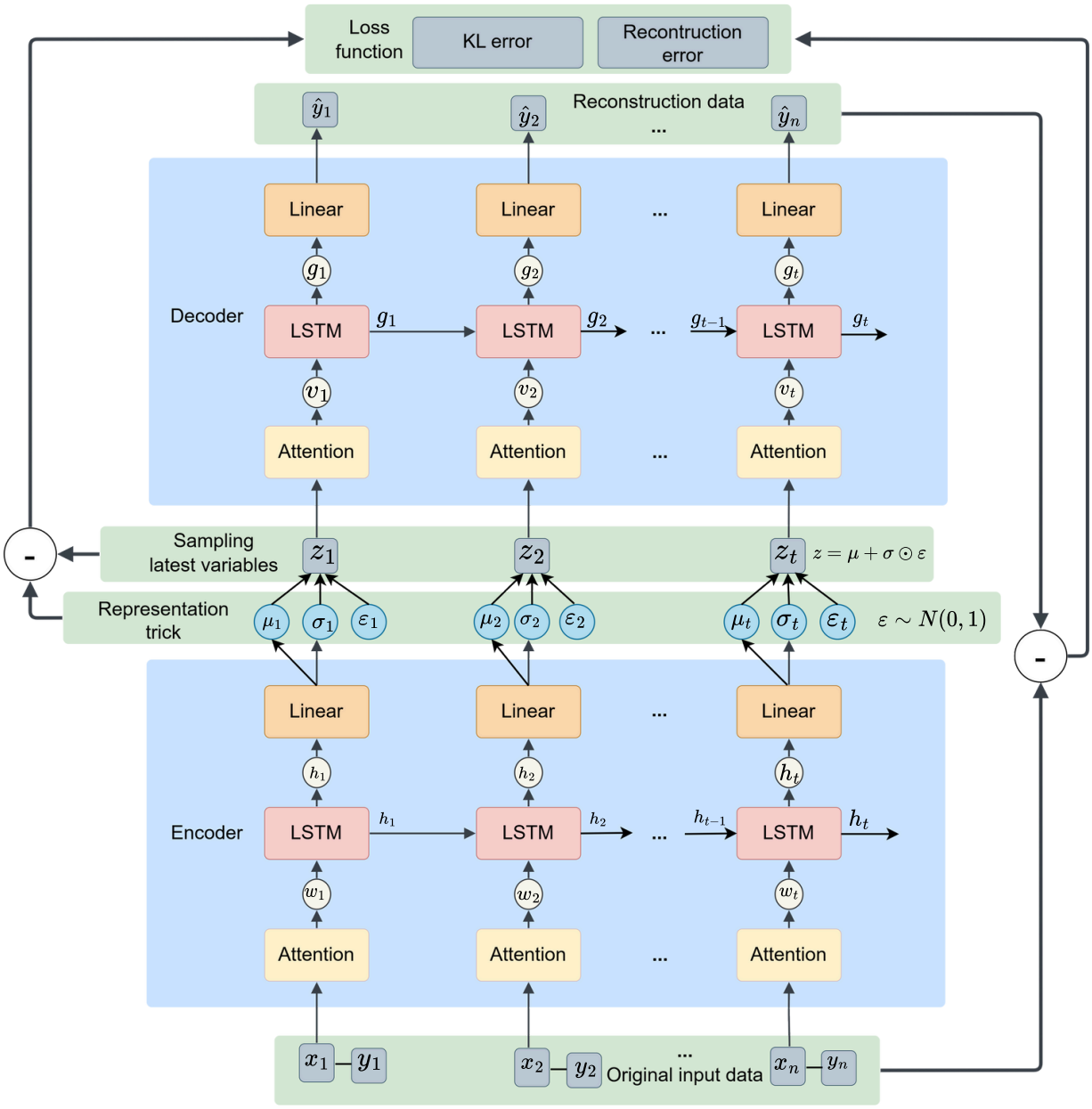

Figure 5: The architecture of the Attention-LSTM-VAE model [47]

$$ELBO(x) = -\int q_\theta(z|x) \log \frac{p_0(z)}{q_\theta(z|x)} dz + \mathbb{E}_{q_\theta(z|x)}[\log P(x|z)] \tag{9}$$

Minimizing the KL divergence is equivalent to maximizing the ELBO. We apply the reparameterization trick to allow gradient-based optimization. For the variational distribution $q_\theta(z|x)$, we assume a Gaussian distribution with diagonal covariance:

$$\log q_\theta(z|x^{(i)}) = \log N(z; \mu^{(i)}, \sigma^{(i)}) \tag{10}$$

where $\mu^{(i)}$ and $\sigma^{(i)}$ represent the mean and standard deviation for each input $x^{(i)}$.

Next, we compute the KL divergence between $q_\theta(z|x)$ and the prior $N(0, I)$:

$$KL[q_\theta(z|x)\|N(0,I)] = \frac{1}{2}\sum_{j=1}^{D}\left[1 + \log(\sigma_j^{(i)})^2 + (\mu_j^{(i)})^2 + (\sigma_j^{(i)})^2\right] \tag{11}$$

The objective function to optimize combines the modified KL divergence and the reconstruction error:

$$ELBO(x) = -\beta \cdot KL[q_\theta(z|x)\|p_0(z)] + \mathbb{E}_{q_\theta(z|x)}[\log P(x|z)] \tag{12}$$

where the latent variables are sampled as:

$$z^{(i)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(i)}, \quad \epsilon^{(i)} \sim N(0, I) \tag{13}$$

Here, $\odot$ denotes element-wise multiplication between $\sigma^{(i)}$ and $\epsilon^{(i)}$, and $\epsilon^{(i)}$ is sampled from a standard normal distribution.

The loss function for the Beta-VAE with Attention-LSTM is the sum of the KL divergence term and the reconstruction error, expressed as:

$$\text{Loss}_{\text{Beta-VAE}} = \text{Loss}_{KL} + \text{Loss}_{\text{recon}} \tag{14}$$

where the reparameterization is given by:

$$z(i, j) = \mu(i) + \sigma(i) \odot \epsilon(l), \quad \epsilon(l) \sim N(0, I) \tag{15}$$

Beta-VAE with Attention-LSTM simplifies calculations despite its complexity. It boosts efficiency and robustness for handling noisy DNA storage data and error-prone sequence patterns.

Wasserstein Generative Adversarial Network (WGAN). We propose a WGAN framework to address mode collapse [5] as shown in Figure **??**. The generator G takes an error-free DNA sequence $X_1 = \{x_1, x_2, \ldots, x_T\}$ and generates an error-prone sequence $X_2 = \{x_1, x_2, \ldots, x_T\}$, reflecting stochastic and base-level errors where T represents the length of the sequence. The generator G is trained with a profile P, which represents the statistical error distribution derived from actual data. The input X_1 is processed by G to produce a simulated sequence R_{gen} that mimics errors. The discriminator D is designed to differentiate between R_{gen} and true error-prone sequences $O_1 = \{o_1, o_2, \ldots\}$, guiding G to generate biologically plausible errors through adversarial training. This process results in paired outputs: an aligned oligo O_{aligned} and the aligned read R_{aligned}, both of which are stored in the FASTQ format.

Autoregressive Transformer (AR-T). We utilize AR-T in the section 3.2, excluding the VAE settings.

A.2 ADDITIONAL IMPLEMENTATION OF MODELS

MAL-VAE. We set the LSTM hidden size to 256 to capture complex dependencies. The output size is also 7, ensuring the model reconstructs the input. The sequence length is 154, representing the length of the input sequence. We handle the latent space with a latent dimension of 128. We divide the sequence into 5 parts using a division constant. We implement three LSTM layers in both the encoder and decoder. We train the model for 15 epochs to optimize performance.

A.3 ADDITIONAL ANALYSIS

We analyze stochasticity in the context of the D-N dataset as shown in 6. The figure illustrates stochastic insertion, substitution, and deletion errors in generated reads from a fixed single oligo. Despite variations in the number of bases at each coverage across different indices, the error distribution remains consistent. However, in the D-N dataset, the substitution error rate is significantly low, leading to a noticeable mismatch in the overall error distribution.

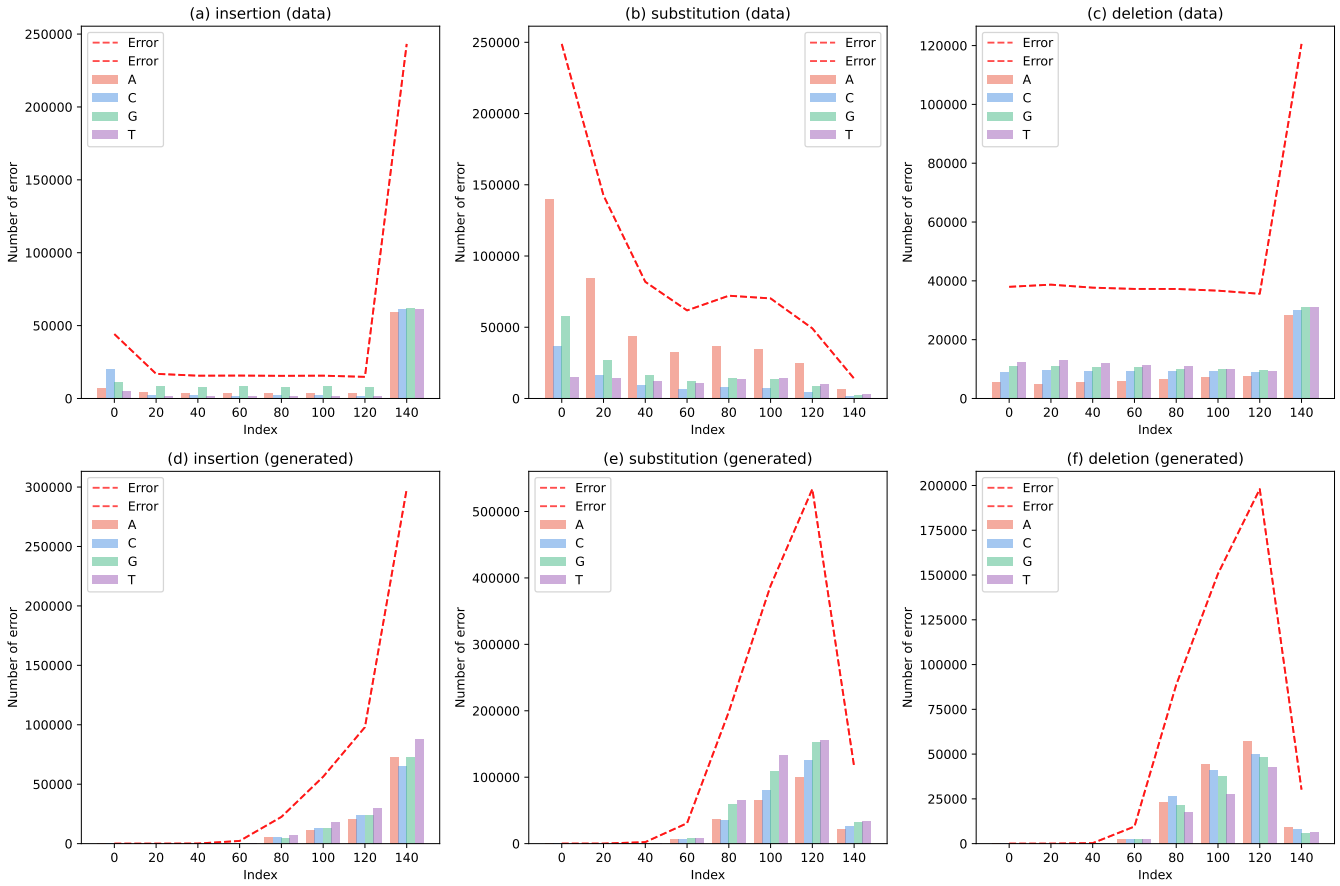

Figure 6: Stochastic insertion, substitution, and deletion errors in generated reads from a fixed single oligo in case of D-N. The x-axis represents the error index, with different colors denoting various erroneous bases. The dotted line indicates the total number of errors for all bases at each index.