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

Mst. Fahmida Sultana Naznin[1], Swarup Sidhartho Mondol[1], Adnan Ibney Faruq[1], Debashmita Saha[1], Ahmed Mahir Sultan Rumi [1], A. B. M. Alim Al Islam[1]

[1]Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh
Correspondence: nazninfahmidasultana@gmail.com

Abstract

DNA has emerged as a promising medium for long-lasting data stoage due to its high information density and long-term stability. However, DNA storage is a complex process where each stage introduces noise and errors. 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 tools often rely on fixed error rates or are specific to certain technologies. We propose DDS-E-Sim, a transformer-based probabilistic generative framework that simulates errors in a DNA data storage channel, regardless of the process or technology. DDS-E-Sim successfully captures the error distribution of DNA storage pipelines and learns to stochastically generate erroneous DNA reads. Given oligos (DNA sequences to write), it outputs erroneous reads resembling real pipelines capturing both random and biased errors, such as k-mer and transition errors. Evaluations on two distinct technology-specific datasets show high fidelity and universality: DDS-E-SIM exhibit a total error rate deviation of only 0.1% and 0.7% respectively on the datasets processed with Illumina MiSeq and Oxford Nanopore. Additionally, our simulator generates 100,743 unique oligos from 35,329 sequences, with coverage 5 (each sequence read five times) in the test datasets, demonstrating its ability to simulate biased errors and stochastic properties simultaneously.

1 Introduction

In the era of data expansion, the world produces 10^{18} bytes daily [26], demanding durable storage ($>$50 years) [41]. DNA offers high information density and stability with minimal maintenance power [11]. DNA Data Storage (DDS) comprises multiple stages - synthesis, storage, sampling, and sequencing, each introducing potential insertion, deletion, and substitution errors[6, 39]. DDS errors are both synchronous and asynchronous with complex statistics [20]. Despite reduced synthesis and sequencing costs [28, 10, 5], large-scale DDS remains costly. Simulation provides a low-cost, probabilistic approach for channel verification, synthesis analysis, and ECC evaluation [14, 22, 23, 4] by modelling synthesis imperfections, molecular decay, PCR biases, and sequencing noise [2, 9].

Numerous prior DDS studies characterize single-stage error simulation separately [38, 21, 29, 27], but fail to capture end-to-end error evolution. Existing simulators such as ART [16], Flux [12], and pIRS [15] focus on specific sequencing stages. Nanopore-focused tools (DeepSimulator [24, 25], NanosigSim [7], NanoSim [30]) remain technology-specific. DeSP [39] is a flexible model which

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: accepted at third SPIGM workshop @ NeurIPS 2025.

can adapt to diverse experiment conditions but it uses fixed user-defined parameters. WGAN [20] is a recently developed universal simulator that is not process dependent but it struggles with longer Nanopore sequences because of its GAN architecture. Moreover, extensive hyperparameter tuning is required for stable GAN training.

Conventional DNA data storage error modelling involves numerous challenges. Firstly, Errors in DNA data storage display both systematic biases and stochastic behaviour [37, 20]. Most of the existing simulators rely on predefined, hard-coded probabilities [31], which overlook stochastic variability. Conversely, some of the models may achieve that but fail to account for systematic biases [32, 3]. We address the problem with a Transformer-based generative model within a Beta-VAE framework that additionally incorporates Gaussian noise for stochasticity and retains dominant biased patterns. Secondly, error behaviours differ by technology[15, 24]. For instance, Illumina uses fluorescence for short, precise reads, while Nanopore detects electrical signals for long, relatively more error-prone reads. Most DDS simulators remain technology-specific. Researchers need to switch simulators at different stages or datasets to simulate end-to-end errors, as error profiles vary with technology and even with system configurations, which creates a major bottleneck. This highlights the need for a universal simulator that integrates multiple stages and technologies. A unified framework would streamline diverse DDS methods and facilitate system design. Our framework is data-centric rather than process-centric, and its validation on diverse technological datasets ensures robust cross-process generalization. In summary, our main contributions are as follows:

- We introduce a novel transformer-based probabilistic generative model within a Beta-VAE framework for error simulation in DDS. We additionally incorporate a Gaussian noise–based perturbation to further enhance stochasticity, which successfully introduces randomness and aligns the generated samples with the statistical distribution of the datasets. Our simulator is universal and simulates erroneous DNA reads regardless of technology.
- We successfully demonstrate the robustness of our data-centric universal modelling by simulating error profiles from two distinct datasets across different technologies. We conduct extensive experiments to quantify insertion, deletion, and transition error rates, as well as k-mer error patterns. Our simulator generates DNA reads that closely match real error profiles observed in the test datasets and outperforms existing tools.

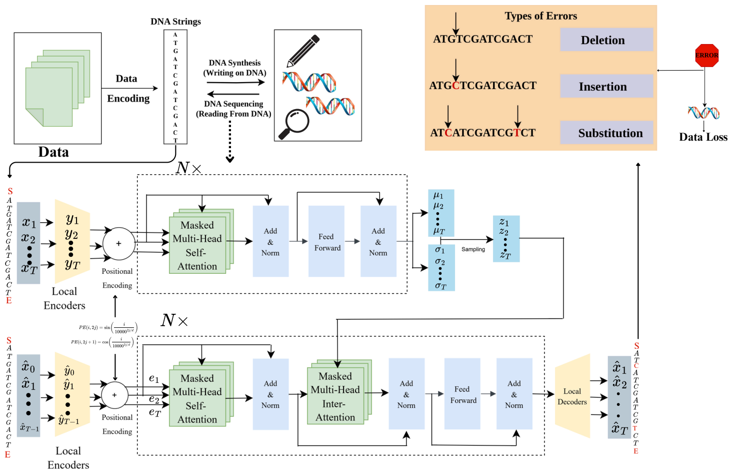
DNA Strings
Data
Data Encoding
DNA Synthesis (Writing on DNA)
DNA Sequencing (Reading From DNA)
Types of Errors
ATGTCGATCGACT
Deletion
ATGCTCGATCGACT
Insertion
ATCATCGATCGTCT
Substitution
ERROR
Data Loss
$N\times$
x_1 x_2 x_T
y_1 y_2 y_T
Local Encoders
Positional Encoding
Masked Multi-Head Self-Attention
Add & Norm
Feed Forward
μ_1 μ_2 μ_T
σ_1 σ_2 σ_T
Sampling
z_1 z_2 z_T
$PE(i,2j) = \sin\left(\frac{i}{10000^{2j/d}}\right)$
$PE(i,2j+1) = \cos\left(\frac{i}{10000^{2j/d}}\right)$
$\hat{x}_0$ $\hat{x}_1$ $\hat{x}_{T-1}$
$\hat{y}_0$ $\hat{y}_1$ $\hat{y}_{T-1}$
e_1 e_2 e_T
Masked Multi-Head Inter-Attention
Local Decoders
$\hat{x}_1$ $\hat{x}_2$ $\hat{x}_T$

Figure 1: An overview of DDS-E-Sim architecture and DNA data storage pipeline.

2 Methods

We use a Transformer within a Beta-VAE framework to learn disentangled error representation in DNA sequences. An overview of the architecture is shown in Figure 1.

2.1 Problem Formulation

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 $x = x_1 x_2 \dots x_N$, with $x_i \in X_e$ for $1 \le i \le N$. The output of the simulator is an error-prone DNA sequence $\hat{x}$, 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 $1 \le i \le M$. Initially, sequence length variation was addressed using padding but we later adopted a sliding-window strategy, which proved to be a slightly more effective choice.

2.2 Proposed Stochastic Generative Learning

Inducing Stochasticity with Gaussian Noise in DNA Sequences In DDS, error-prone sequences must reflect stochasticity across coverage levels. We represent characters through one-hot encoding and manage long sequences by first applying padding, then switching to a sliding-window strategy sliding over the sequence with window and step size. We add Gaussian noise to model synthesis and sequencing variability as unbiased synthesis and sequencing processes naturally exhibit Gaussian-like behaviour [8, 37]. Noisy sequences are generated as $y = x + \mathcal{N}(\mu, \sigma^2)$, where x is the original sequence, y the perturbed 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.

Transformer Architecture for Capturing Error Patterns Our model integrates a Transformer [34] in a Beta-VAE framework [13] to learn error patterns from DNA sequences. We incorporate masked and causal attention to preserve sequential dependencies [19]. Let $x_{1..T}$ be a DNA sequence of length T, where x_i denotes the i-th nucleotide. After noise addition, each nucleotide passes through a local encoder E_{local}: $y_i^e = E_{\text{local}}(x_i)$ yielding nucleotide-level representations $y_{1..T}^e$. The Transformer encoder computes the latent parameters:

$$g_{1..T}^e = E_{\text{Transformer}}(y_{1..T}^e)$$
$$[\mu_i, \log(\sigma_i^2)] = W_i g_i^e + b_i, \quad z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$$

The latent codes are decoded sequentially using the Transformer decoder $D_{\text{Transformer}}$ and local decoder D_{local}:

$$g_i^d = D_{\text{Transformer}}(z_{1:i}, y_{0:i-1}^d), \quad \hat{x}_i = D_{\text{local}}(g_i^d)$$

Here $\hat{x}_{1..T}$ is the generated error-prone sequence, and $y_i^d = E_{\text{local}}(\hat{x}_i)$. A special embedding y_0^e denotes the start of sequence, analogous to the SOS token in Natural Language Processing.

Causal Masking for Sequential Cohesion Control (CM) In DDS, nucleotide sequences must follow a strict order to ensure correct encoding and retrieval. Similar to biological transcription and sequencing, the model must not access future positions, as order disruption leads to retrieval errors [36, 35]. To emulate this, we apply causal masking across encoder and decoder layers, enforcing directionality so that the i-th base attends only to positions $j \le i$. The masked attention is defined as

$$E_i' = \frac{\sum_{j=1}^{i} \text{sim}(Q_i, K_j) E_j}{\sum_{j=1}^{i} \text{sim}(Q_i, K_j)},$$

where E_i' is the updated representation of base i, Q_i and K_j are query and key vectors, E_j is the base representation, and $\text{sim}(Q_i, K_j)$ denotes their similarity score.

Balancing Error Bias Modelling and Stochasticity with Beta-VAE DNA sequences are inherently complex because they contain hierarchical and structured dependencies [17]. 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 [17]. Additionally, regulatory elements like enhancers, silencers, and promoters interact in a hierarchical manner to control gene expression [40]. Thus, errors in DNA sequences inherently carry biases and exhibit dominance in certain regions or motifs [1]. Repetitive sequences or regions with high GC content causes sequencing errors. The model needs to capture error patterns inherent in DNA sequences, such as k-mer biases, positional skewness, and dominance of specific transitions. At the same time, it should introduce stochasticity to ensure diverse and realistic error representations. We use Beta-VAE with transformer architecture to achieve that. The objective function can be expressed as:

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

Here, $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$ 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 $p(z)$. Here, the value of β is of utmost importance. We observe that high value β over-regularizes the latent space , leading to under-representation of motif-dependent DDS error patterns. A moderate setting of β = 0.5 preserved the essential error dependencies while maintaining sufficient regularization to prevent overfitting.

3 Experiments and Results

We conduct extensive experiments on two datasets from different sequencing platforms and compare our model against established baselines. We analyze insertion, substitution, and deletion error rates, as well as k-mer error patterns and positional error distributions, to comprehensively assess how well the models capture the inherent error characteristics of DNA data storage pipelines.

3.1 Dataset Description

We evaluate two technology-specific datasets to assess universality. The first dataset [18] has 18,000 oligos (length 152, GC 45–55%, homopolymer $\leq$3) sequenced via Illumina (15,126,429 reads; train: 12,108,573, test: 2,999,656). The second dataset [33] has 10,000 oligos (length 110) sequenced via ONT MinION (269,709 reads; train: 242,738, test: 26,971). We represent datasets D-I (Illumina) and D-N (Nanopore) respectively.

3.2 Performance Evaluation

We conduct an ablation study with three variants: (1) **Transformer+Beta-VAE:** our proposed model, (2) **Transformer**: excluding VAE from the our architecture, and (3) **VAE:** a Multi-Attention-LSTM with three LSTMs and a VAE (using LSTM instead of transformer in our architecture). In addition, we compare against **WGAN** [20], the only universal deep learning simulator we are aware of. By contrast, simulators such as DeepSimulator [25] and Illumina [16] are technology-specific and cannot be applied across datasets from different sequencing platforms.

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

Model	D-I				D-N			
	Substitution (%)	Insertion (%)	Deletion (%)	Total Error (%)	Substitution (%)	Insertion (%)	Deletion (%)	Total Error (%)
Original read	49.4	25.3	25.3	1.5	28.2	35.9	35.9	8.1
Transformer+Beta-VAE	**45.7**	**24.6**	**29.7**	**1.4**	**27.7**	**34.4**	**37.9**	**7.4**
Transformer	51.2	24.4	24.4	2.2	27.7	32.4	39.9	14.0
VAE	57.1	21.5	21.5	2.5	38.8	30.6	30.6	15.2
WGAN [20]	42.4	20.1	37.4	3.2	36.7	30.2	33.1	5.3

Uneven Base-Level Error Statistics We evaluate single-base errors against original data and baselines in Table 1. For D-I, substitution is the dominant form of error. DDS-E-Sim shows the smallest deviations from original reads and outperforms others by a good margin. Dominant transitions are well captured in our proposed model as shown in Table 2a. DDS-E-Sim also achieves minimal deviations For D-N and accurately models base transitions as shown in Table 2b. DDS-E-Sim achieves an impressive total error deviation of only 0.1% and 0.7% in the two datasets.

Uneven k-mer Error Patterns We analyze 2-mer and 3-mer error rates to assess motif-specific biases (Table 3). In dataset D-I, our model reproduces the expected trends: 3-mer errors exceed 2-mer errors, with AAA being the most error-prone motif and TT the least. In dataset D-N, k-mer analysis shows CCC as the most error-prone motif and AA as the least, with 3-mer errors again higher than 2-mers, thereby preserving higher-order error patterns. Overall, the model effectively captures both motif- and length-dependent error characteristics.

Positional Skewness and Adaptive Stochasticity To evaluate the possibility of overfitting or data leakage, we examine the positional error distributions as shown in Figures 2 and 3. These figures present both the total error counts and the base-specific error counts across sequence positions for insertions, deletions, and substitutions. The alignment of trends between the original datasets and the

Table 2: Transition error rates across datasets D-I and D-N. Each row shows relative error rates of other bases for the base being substituted from. For each transition, the error rate of the best performing model (least deviation from the original read) is shown in bold.

(a) Dataset D-I

	Original read				DDS-E-Sim				WGAN			
	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**	–	0.30	0.30	0.39
C	0.35	–	0.31	0.34	**0.68**	–	**0.15**	**0.17**	0.23	–	0.06	0.71
G	0.38	0.10	–	0.53	0.54	**0.12**	–	0.34	**0.36**	0.07	–	**0.57**
T	0.33	0.18	0.49	–	**0.36**	**0.18**	**0.45**	–	0.22	0.59	0.20	–

(b) Dataset D-N

	Original read				DDS-E-Sim				WGAN			
	A	C	G	T	A	C	G	T	A	C	G	T
A	–	0.22	0.58	0.19	–	**0.22**	**0.57**	**0.21**	–	0.30	0.29	0.41
C	0.21	–	0.17	0.62	**0.21**	–	**0.17**	**0.62**	0.16	–	0.16	0.68
G	0.61	0.18	–	0.22	**0.59**	**0.17**	–	**0.24**	0.23	0.22	–	0.55
T	0.19	0.61	0.20	–	**0.20**	**0.59**	**0.21**	–	0.22	0.53	0.25	–

Table 3: K-mer error rates of different models across datasets D-I and D-N. For each k-mer, the error rate of the best performing model (least deviation from the original read) is shown in bold.

Model	AA	CC	TT	GG	CCC	AAA	GGG	TTT
Original read (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
WGAN	0.08	0.04	0.02	**0.03**	0.06	0.39	0.21	**0.15**
Original read (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
WGAN	**0.04**	0.07	**0.09**	**0.09**	**0.32**	**0.14**	**0.16**	**0.08**

generated outputs demonstrates that our model avoids overfitting and data leakage. For dataset D-I, the generated sequences exhibit stochastic variation while maintaining overall consistency with the empirical error profiles (Figure 2). Specifically, insertion errors display terminal spikes, substitution errors are generally high, and deletion errors remain generally low except at terminal regions. Overall, the generated sequences successfully capture the error patterns observed in the original dataset.

Similarly, we analyze positional distribution in the context of the D-N dataset as shown in Figure 3. Again, our model fairly matches the positional error distribution of the dataset except the distribution of substituition errors. In the D-N dataset, the substitution error rate is comparatively low, which is a probable reason for the noticeable mismatch in the overall error distribution.

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, 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%. In the second method, 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 regularise 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. Thus, we proceed with the first method.

Stochasticity in DNA sequence reads We demonstrated that DDS-E-Sim effectively captures the error patterns across multiple sequencing technologies. However, to faithfully mimic the DNA data storage pipeline, its output must also be stochastic; that is, it should generate diverse erroneous sequences for the same input while still following the underlying error distribution. To achieve this, we integrate a Beta-VAE into our architecture for stochastic read generation and further introduce Gaussian noise to the input sequences, which proves to be highly effective. To validate randomness, we generated reads from 35,329 fixed sequences at a coverage of 5 (simulating five reads per

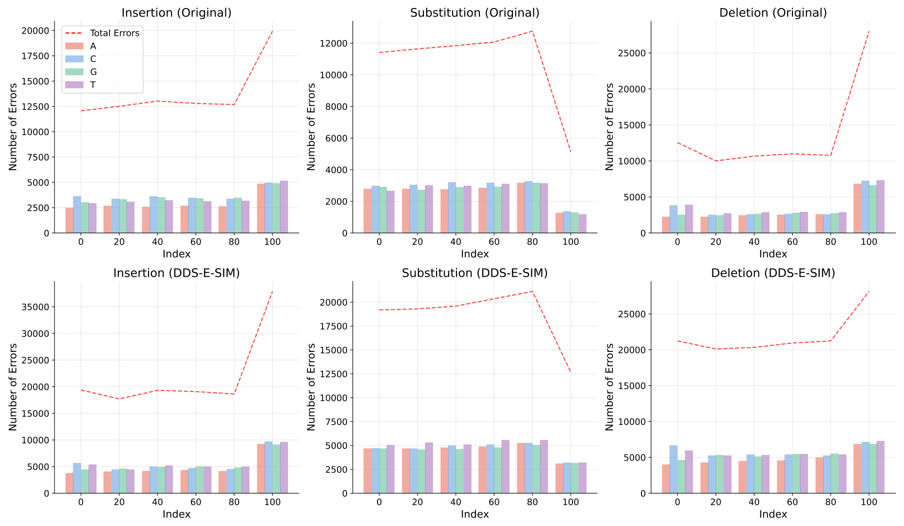

Figure 2: Stochastic insertion, substitution, and deletion error count in original and generated reads in dataset D-I. The x-axis denotes sequence position indices and the y-axis denotes error count. The bar plots represent base-specific error counts and the dotted line indicates the aggregate error count across all bases.

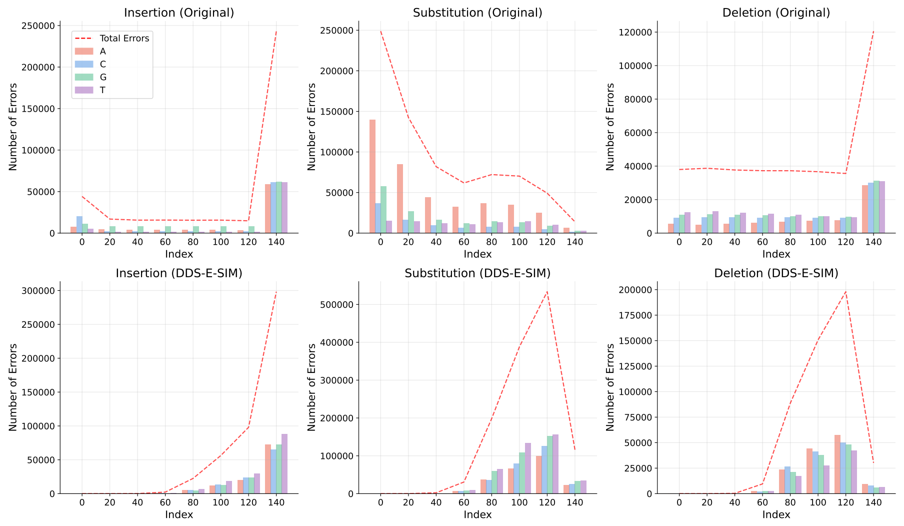

Figure 3: Stochastic insertion, substitution, and deletion error count in original and generated reads in dataset D-N. The x-axis denotes sequence position indices and the y-axis denotes error count. The bar plots represent base-specific error counts and the dotted line indicates the aggregate error count across all bases.

sequence), resulting in 100,743 unique oligos. This demonstrates that our model preserves not only the error distributions but also the essential stochasticity of the DNA data storage process.

4 Conclusion and Future Work

We present a universal transformer-based generative framework for simulating error-prone DNA sequences regardless of process or technology. The model captures higher-order error patterns while preserving stochasticity, achieving state-of-the-art performance. We are actively working to extend the framework with designing and combining stage-specific modules to simulate errors at each DDS stage and integrate error-correction algorithms for testing and optimization in the near future.

References

[1] Manuel Allhoff, Alexander Schönhuth, Marcel Martin, Ivan G Costa, Sven Rahmann, and Tobias Marschall. Discovering motifs that induce sequencing errors. *BMC Bioinformatics*, 14(S1), 2013.

[2] Jamie J. Alnasir, Thomas Heinis, and Louis Carteron. Dna storage error simulator: A tool for simulating errors in synthesis, storage, pcr and sequencing, 2022.

[3] Jamie J. Alnasir, Thomas Heinis, and Louis Carteron. Dna storage error simulator: A tool for simulating errors in synthesis, storage, pcr and sequencing, 2022.

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

[5] Twist Bioscience. Genes, n.d. Accessed: [Date Accessed].

[6] Luis Ceze, Jeff Nivala, and Karin Strauss. Molecular digital data storage using dna. *Nature Reviews Genetics*, 20:456 – 466, 2019.

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

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

[9] Sanket Doshi, Mihir Gohel, and Manish K. Gupta. A bird-eye view on dna storage simulators. *arXiv preprint*, 2404.04877v1, April 2024. License: arXiv.org perpetual non-exclusive license.

[10] eurofins Genomics. Price list, 2017.

[11] Andy Extance. How dna could store all the world's data? *Nature*, 537:22–24, 2016.

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

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

[14] Syed Mahamud Hossein, Tabatabaei Yazdi, Yongbo Yuan, Jian Ma, Huimin Zhao, and Olgica Milenkovic. A rewritable, random-access dna-based storage system. *Scientific Reports*, 5, 2015.

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

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

[17] Balaji VS Iyer, Martin Kenward, and Gaurav Arya. Hierarchies in eukaryotic genome organization: Insights from polymer theory and simulations. *BMC Biophysics*, 4, 2011.

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

[19] Junyan Jiang, Gus G. Xia, Dave B. Carlton, Chris N. Anderson, and Ryan H. Miyakawa. 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)*, pages 516–520, 2020.

[20] 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. Generative adversarial networks for dna storage channel simulator. *IEEE Access*, 11:123456–123465, Jan 2023. Received 28 November 2022, accepted 31 December 2022, date of publication 9 January 2023, date of current version 12 January 2023.

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

[22] Henry H. Lee, Reza Kalhor, Naveen Goela, Jean-Chrysostome Bolot, and George M. Church. Terminator-free template-independent enzymatic dna synthesis for digital information storage. *Nature Communications*, 10, 2019.

[23] Andreas Lenz, Paul H. Siegel, Antonia Wachter-Zeh, and Eitan Yaakobi. Coding over sets for DNA storage. *CoRR*, abs/1812.02936, 2018.

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

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

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

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

[28] Catherine Offord. Making dna data storage a reality, 2017.

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

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

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

[32] SNIA. Dnassim full system simulator for dna storage, 2022. Accessed: 2025-01-30.

[33] Sundara Rajan Srinivasavaradhan, Sivakanth Gopi, Henry D. Pfister, and Sergey Yekhanin. Trellis bma: Coded trace reconstruction on ids channels for dna storage, 2024.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[35] C. Wang, G. Ma, D. Wei, et al. Mainstream encoding–decoding methods of dna data storage. *CCF Transactions on High Performance Computing*, 4:23–33, 2022.

[36] Y. Wang, M. Noor-A-Rahim, J. Zhang, et al. High capacity dna data storage with variable-length oligonucleotides using repeat accumulate code and hybrid mapping. *Journal of Biological Engineering*, 13(1):89, 2019.

[37] Yixin Wang, Md. Noor-A-Rahim, Erry Gunawan, Yong L. Guan, and Chueh L. Poh. Modelling, characterization of data-dependent and process-dependent errors in dna data storage. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 20(3):2147–2158, 2023.

[38] SMHT Yazdi et al. Dna-based storage: trends and methods. *arXiv preprint arXiv:1507.01611*, 2015.

[39] Lekang Yuan, Zhen Xie, Ye Wang, and Xiaowo Wang. Desp: a systematic dna storage error simulation pipeline. *BMC Bioinformatics*, 23(1):185, 2022.

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

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

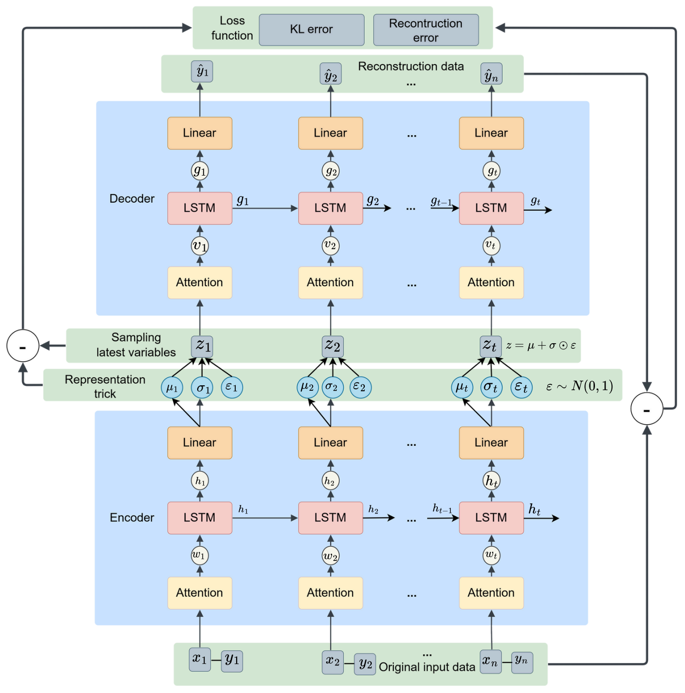

Figure 4: The architecture of the Attention-LSTM-VAE model

A Appendix

A.1 Additional Models Implementation

Multi-Attention-LSTM-Beta VAE (VAE) We design a Beta-VAE architecture with Attention-LSTM for our ablation study to highlight the benefit of the transformer architectue in our propsed model. This architecture designed to efficiently handle high-dimensional data as shown in Figure 4. 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:

$$\text{Loss}_{KL} = \beta \cdot KL[q_\theta(z|x) \| p_0(z)] \tag{1}$$

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:

$$p(z|x) = \frac{p_0(z)P(x|z)}{P(x)} \tag{2}$$

The KL divergence is computed as:

$$KL[q_\theta(z|x) \| p(z|x)] = -\int q_\theta(z|x) \log \frac{p(z,x)}{q_\theta(z|x)} dz + \log P(x) \tag{3}$$

The Evidence Lower Bound (ELBO) is formulated as:

$$ELBO(x) = -\int q_\theta(z|x) \log \frac{p_0(z)}{q_\theta(z|x)} dz + \mathbb{E}_{q_\theta(z|x)}[\log P(x|z)] \tag{4}$$

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:

$$\log q_\theta(z|x^{(i)}) = \log N(z; \mu^{(i)}, \sigma^{(i)}) \tag{5}$$

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 $N(0, I)$:

$$KL[q_\theta(z|x) \| N(0, I)] = \frac{1}{2} \sum_{j=1}^{D} \left[1 + \log(\sigma_j^{(i)})^2 + (\mu_j^{(i)})^2 + (\sigma_j^{(i)})^2 \right] \tag{6}$$

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

$$ELBO(x) = -\beta \cdot KL[q_\theta(z|x) \| p_0(z)] + \mathbb{E}_{q_\theta(z|x)}[\log P(x|z)] \tag{7}$$

where the latent variables are sampled as:

$$z^{(i)} = \mu^{(i)} + \sigma^{(i)} \odot \epsilon^{(i)}, \quad \epsilon^{(i)} \sim N(0, I) \tag{8}$$

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:

$$\text{Loss}_{\text{Beta-VAE}} = \text{Loss}_{KL} + \text{Loss}_{\text{recon}} \tag{9}$$

where the reparameterization is given by:

$$z(i, j) = \mu(i) + \sigma(i) \odot \epsilon(l), \quad \epsilon(l) \sim N(0, I) \tag{10}$$

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.

Autoregressive Transformer (Transformer) We utilize the Transformer architecture as shown in our proposed architecture in Figure 1 excluding the Beta-VAE framework for our ablation study.

A.2 Additonal Implementation Details

DDS-E-Sim 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. We train our model with a batch size of 256 for 20 epochs using the Adam optimizer (learning rate = 1×10^{-4}, $\beta = 0.5$).

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 sequences into a few parts (usually 5) and feed that to the LSTMs as LSTMs cannot handle long sequences. We implement three LSTM layers in both the encoder and decoder. We train the model for 15 epochs.