Generative DNA: Representation Learning for DNA-based Approximate Image Storage

Giulio Franzese*,Yiqing Yan, Giuseppe Serra, Ivan D'Onofrio, Raja Appuswamy, Pietro Michiardi
Department of Data Science, EURECOM
Biot, France
Email: *giulio.franzese@eurecom.fr

***Abstract*—Synthetic DNA has received much attention recently as a long-term archival medium alternative due to its high density and durability characteristics. However, most current work has primarily focused on using DNA as a precise storage medium. In this work, we take an alternate view of DNA. Using neural-network-based compression techniques, we transform images into a latent-space representation, which we then store on DNA. By doing so, we transform DNA into an approximate image storage medium, as images generated back from DNA are only approximate representations of the original images. Using several datasets, we investigate the storage benefits of approximation, and study the impact of DNA storage errors (substitutions, indels, bias) on the quality of approximation. In doing so, we demonstrate the feasibility and potential of viewing DNA as an approximate storage medium.**

***Index Terms*—DNA storage, representation learning, approximate storage, compression**

I. Introduction

Recently, analysts have predicted that the Global Datasphere, the sum total of all data generated, will reach 160 Zettabyte by 2025 [19]. Unfortunately, all current storage media, be it NAND flash, HDD, or tape, suffer from fundamental density and durability limitations that complicates cost-effective storage of cold data [2]. Thus, researchers have started exploring the use of radically new storage media that can offer orders of magnitude improvement in both density and durability. One such media that has received a lot of attention recently is Deoxyribo Nucleic Acid (DNA).

DNA is a macro-molecule that is composed four sub-molecules called nucleotides: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). DNA used for data storage is a short, single-stranded sequence of nucleotides, also referred to as an oligo. Using DNA as a digital storage medium requires mapping digital data into a sequence of strings that represent oligos using an encoding algorithm. Once encoded, the strings are used to *synthesize* oligos using a chemical process. Data stored in DNA is read back by *sequencing* the oligos which produces several noisy copies of the original strings, also called reads. A consensus procedure is then used to infer the original oligonucleotide sequences based on sequencing reads, and the inferred strings are then passed to a decoder to recover the original data.

Till date, current work on DNA storage has focused on dealing with errors introduced by the imprecise nature of synthesis and sequencing [5], [9], [8], [17], [16]. The typical approach adopted is to use error-correction techniques from coding theory, and high sequencing coverage (the average number of reads that correspond to an oligo) to be able to recover data despite errors. Such an approach is necessary for applications like long-term archival or digital preservation, where DNA should function as a precise storage medium similar to current storage technologies like Hard Disk Drives or tape. However, a large amount of generated data is increasingly used only as input to data analytics and machine learning algorithms/models that can produce functionally identical outcomes using approximate versions of the original data. Researchers have used this insight to show that DATA-DRIVEN compression schemes can provide substantial reduction in storage size with respect to generic compression schemes(JPEG,PNG) for applications that can tolerate approximations [23].

In this work, we investigate the utility of DATA-DRIVEN compression schemes in reducing the cost of DNA storage by viewing it as an approximate storage medium. In particular, we build generative models (Section II) that compress and represent images using a latent space. We propose to store this learned representation in DNA. In contrast to the current approach of masking the errors introduced by the DNA storage channel, we take the alternate approach (Section III) of exposing it to the decoder of our generative model that uses the error-induced latent space to generate different versions of the original images. In order to prove the feasibility and benefit of our approach, we build an full-system prototype and using a thorough simulation study (Section IV), we analyze the impact of DNA storage errors on (i) the quality of reconstructed image and (ii) the classification accuracy when such images are used for training a machine learning model, for both approximate storage and precise storage cases. We show that by treating DNA as an approximate storage medium, we can dramatically reduce synthesis and sequencing costs, as generative models can (i) provide at least $10\times$ better compression than generic schemes, and thereby reduce synthesis costs, and (ii) recover data back at much lower coverage and tolerate many errors, and thereby reduce sequencing costs.

II. Background: Representation Learning

Given a dataset of observations $\mathrm{D} = \{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^M$ also referred to as TRAIN set, it is possible to build parametric (θ) probabilistic models $p_\theta(\cdot)$ to describe the density of the

observed data. The optimal set of parameters θ^*, is obtained by maximizing the probability of the observed dataset

$$\theta^* = \arg\max_{\theta} p_\theta(\mathrm{D}). \tag{1}$$

Assuming independent datapoints $\mathbf{x}_i$ the (log) of the objective function has factorized form

$$\log\left(p_\theta(\mathrm{D})\right) = \sum_{i=1}^{N} \log\left(p_\theta(\mathbf{x}_i)\right). \tag{2}$$

In this work we focus on latent variable models, in particular Variational Autoencoders (VAE)[14]. The set of latent variables $\mathbf{z} \in \mathbb{R}^K$, $K \ll M$, are linked to the observations $\mathbf{x}$ through the conditional parametric density $p_\theta(\mathbf{x}|\mathbf{z})$. Usually the parametric densities are constructed using neural networks, as we do in this work. The parametric density is then expressed as $p_\theta(\mathbf{x}|\mathbf{z}) = p(\mathbf{x};\psi)$, where $\psi = \mathrm{Dec}_\theta(\mathbf{z})$ and $\mathrm{Dec}(\cdot)$ is a *decoder* Neural Network. The directed graphical model corresponds to the joint distribution of observations and latent variables, i.e. $p_\theta(\mathbf{x},\mathbf{z})$. The marginal distribution over the observed variables $p_\theta(\mathbf{x})$, is given by

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x},\mathbf{z})\, d\mathbf{z} = \int p_\theta(\mathbf{x}|\mathbf{z}) p(\mathbf{z})\, d\mathbf{z}. \tag{3}$$

Notice that in general Eq. (3) does not have a closed form solution. Consequently, neither Eq. (2), the target of our optimization, does. This intractability is solved by introducing a new distribution $q_\phi(\mathbf{z}|\mathbf{x})$ constructed by a second neural network $q_\phi(\mathbf{z}|\mathbf{x}) = p(\mathbf{z};\gamma), \gamma = \mathrm{Enc}_\phi(\mathbf{x})$, where $\mathrm{Enc}(\cdot)$ is an *encoder* network. By means of Jensen's inequality [6] we manipulate Eq. (3) as follows

$$\begin{aligned}
&\log\left(p_\theta(\mathbf{x}_i)\right) = \log\left(\int p_\theta(\mathbf{x}_i|\mathbf{z}) \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)} q_\phi(\mathbf{z}|\mathbf{x}_i)\, d\mathbf{z}\right) \geq \\
&\int \log\left(p_\theta(\mathbf{x}_i|\mathbf{z}) \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}_i)}\right) q_\phi(\mathbf{z}|\mathbf{x}_i)\, d\mathbf{z} = \\
&\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[p_\theta(\mathbf{x}_i|\mathbf{z})\right] - \mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{x}_i)||p(\mathbf{z})\right],
\end{aligned} \tag{4}$$

where $\mathbb{KL}$ is the Kullback-Leibler divergence [6].

The first term of Eq. (4) can be trivially estimated with

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}\left[p_\theta(\mathbf{x}|\mathbf{z})\right] \simeq p_\theta(\mathbf{x}_i|\mathbf{z}), \quad \mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}_i) \tag{5}$$

In this work, we select $q_\phi(\mathbf{z}|\mathbf{x}_i) = \mathcal{N}(\mathbf{z};\mu,\mathrm{diag}(\sigma)^{\circ 2}))$, that corresponds to $\gamma = \left[\mu, \log(\sigma)\right] = \mathrm{Enc}_\phi(\mathbf{x})$ [14]. Notice that the invertible log transformation is included in the parametrization. By selecting as target prior distribution $p(\mathbf{z}) = \mathcal{N}(\mathbf{z};\mathbf{0},\mathbf{I})$, the Kullback-Leibler term of Eq. (4) has closed form solution

$$\mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{x}_i)||p(\mathbf{z})\right] = \frac{1}{2}\sum_i \sigma_i^2 + \mu_i^2 - \log\sigma_i^2 - K. \tag{6}$$

Combining Eq. (5) and Eq. (6) it is possible to evaluate Eq. (4). Since the networks Enc, Dec are differentiable, it is possible to compute the gradient with respect to ϕ, θ of Eq. (4), and consequently of the bound of Eq. (2). The optimization procedure, now including also the parameters ψ, can then be performed iteratively with any gradient based optimizer (in this work, we chose ADAM [13]).

III. Design: Representations as encodings

In this section, we present our methodology for storing quantized latent representations using DNA. We use the VAE model to *compress* the data and store only the latent variables representing it on the storage medium, in a similar vein to the work in [23]. Once the VAE has been trained, it is possible to encode data belonging to a previously unseen set, the TEST set $\mathrm{D}^* = \{\mathbf{x}_i^*\}_{i=1}^T$ by means of

$$\left[\gamma^1, \ldots, \gamma^T\right] = \mathrm{Enc}(\mathrm{D}^*) \tag{7}$$

Since the latent variables γ dimension is much smaller than that of the original datapoints, as $2K \ll M$, we can expect to achieve a compression factor of $\frac{M}{2K}$.

However, latent parameters are continuous: it is thus not possible to directly store them as they are produced. Although discrete latent variable models exist [20], [12], [18], [21], [7], i.e. $\mathbf{z} \in \mathcal{Z}$ where $\mathcal{Z}$ is some finite cardinality space, we propose instead to perform quantization of the parameters of the latent variables after the training. Indeed, training discrete latent variable models is extremely challenging and inefficient, whereas empirical results indicate neural networks to be robust to quantization [10], [11].

As a baseline, we quantize the parameters of the latent variables by means of a uniform quantizer $\gamma_q = \mathrm{quant}(\gamma)$. We encode the TEST set with the trained Enc network as $\left[\gamma^1, \ldots, \gamma^T\right] = \mathrm{Enc}(\mathrm{D}^*)$ and feed the result element-wise to a uniform quantizer with N_{bit} of precision and range $[-3, 3]$. In Section IV we refer to this implementation variant as VAEU.

Additionally, we propose an ad-hoc quantization scheme based on the latent parameters statistics. We estimate, on the TRAIN set, the element-wise empirical means and standard deviations of the encodings $\left[\gamma^1, \ldots, \gamma^N\right] = \mathrm{Enc}(\mathrm{D}) \rightarrow \left[\mathbf{m}, \mathbf{s}\right]$. For all datapoints belonging to the TEST set D^*, we first apply to the generic latent variable $\gamma_i^{(j)}$ the (invertible) nonlinear transformation $f : \frac{1}{2}(1 + \frac{\mathrm{erf}(\cdot - m_i)}{\sqrt{2s_i^2}})$. Then we apply a uniform quantizer considering N_{bit} of precision. Given the encoded parameters representing a TEST datapoint, it is possible to recover the original data by applying $f^{(-1)}$ to the quantized values and subsequently feed the values to the decoder network Dec. In Section IV we refer to this variant as VAEN.

The latent space representation is materialized as a binary file. To convert this file into strings that represent DNA to be synthesized, we use *pg_oligo_archive*, a tool developed in prior work [2] which builds on work done by Goldman et al. [9]. As each DNA strand is limited to length to a few hundred nucleotides due to synthesis limitations, *pg_oligo_archive* divides the binary files into chunks, such that each chunk corresponds to a single DNA strand. It then uses a uses a Huffman dictionary to convert each chunk from binary into ternary format and adds an index to each chunk to be able to infer the order of oligos back again during recovery as DNA, in itself, is not an explicitly addressed storage medium. Finally, a rotational code is used to convert the ternary form into a quaternary DNA string to ensure that the generated oligos have a balanced G-C content (ratio of Gs

TABLE I: Enc$(\cdot)$

CV$(1,32)$,BN,RELU
CV$(32,64)$,BN,RELU
CV$(64,128)$
FC(32)

TABLE II: Dec$(\cdot)$

FC(32768)
DC$(128,64)$,BN,RELU
DC$(64,32)$,BN,RELU
DC$(32,1)$SIGM

	MS-SSIM	PSNR	ACC
VAEN	0.863	15.000	0.752
VAEU	0.775	12.881	0.422

TABLE III: Metrics comparison between VAEN and VAE

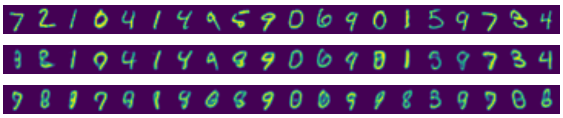


Fig. 1: Comparison of reconstructed images with a coverage of 3. From top to bottom: original images, VAEN reconstructions, VAEU.

and Cs) and no homopolymer repeats (repeated sequences of a particular nucleotide), as oligos with extreme GC content or repeats are known to create problems during synthesis and sequencing. The decoding pipeline of *pg_oligo_archive* starts with sequenced reads, which are noisy copies of the original oligos. From these reads, a consensus procedure is used to infer the original oligos. The inferred oligos are then reordered using the index stored in each oligo, and the reverse decoding from quaternary to ternary to binary is performed to restore back the original data.

As we show in the evaluation, the fraction of data that can be successfully recovery depends on the sequencing coverage used, where coverage roughly corresponds to the number of reads that correspond to a reference oligo. It is well known that bias in synthesis and sequencing steps lead to variation in coverage. Thus, while some oligos are sequenced and read multiple times, others can be completely omitted (or dropped out). In order to recover data completely when DNA is used as a precise storage medium, most current approaches use a reasonably high coverage ($10\times$-$100\times$). One of the goals of our work is to show that when we use DNA as an approximate storage medium with generative compression, we can tolerate much lower coverage levels.

IV. EXPERIMENTS

In this section we experimentally validate our proposal by comparing it against JPEG and PNG on the standard MNIST dataset [15].

Architecture and training details. The Enc and Dec networks are represented in Table I and Table II respectively. In our notation, a single line of a table represents a layer. The convention is the following: at the beginning of the row, we have the type of layer, the indication of presence of BatchNorm if the BN acronym is present, as well as the nonlinearity associated with the layer (if present). The first type of layers we consider is the convolutional one CV(N_{in},N_{out}), where N_{in},N_{out} are the number of input and output channels respectively. For all CV layers we consider kernel size 3 and stride 2. A similar convention holds for the deconvolutional layers (DC(N_{in},N_{out})). For DC layers we consider kernel size 4 and stride 2. Finally we consider fully connected layers (FC(N_{out})), where N_{out} is the cardinality of the output. We use as nonlinearities the rectified linear unit (RELU) and the standard sigmoid (SIGM).

We use the ADAM optimizer with default settings, learning rate 0.001, 100 training epochs and batch size 64.

Storage of Enc **and** Dec **network weights.** We assume, as done in [23], that the neural architectures are available to the data reader. This can be achieved either by storing the network weights on the DNA, or transmitted through a separate channel [1]. We stress that the overhead introduced by such an operation is independent on the dataset size, and thus negligible in the large data regime.

Comparison Metrics. The quantitative metrics we consider are the standard Peak Signal to Noise Ratio (PSNR) and MultiScale Structural Similarity Index Measure (MS-SSIM). We include such metrics to have experimental uniformity with the literature but we stress that such metrics are known to be not perfect in terms of actual representation of the perceptual quality of an image [3]. For this reason, we also include the classification accuracy ACC of a pretrained classifier on the reconstructed data. While this last metric can be considered more semantically meaningful than PSNR and MS-SSIM we resort also to a qualitative inspection of the results by comparing a random subset of the reconstructed images for all the considered algorithms. In all the experiments we report the Bit Per Pixel (BPP) for all the considered algorithms as a quantitative indicator of the compression ratio.

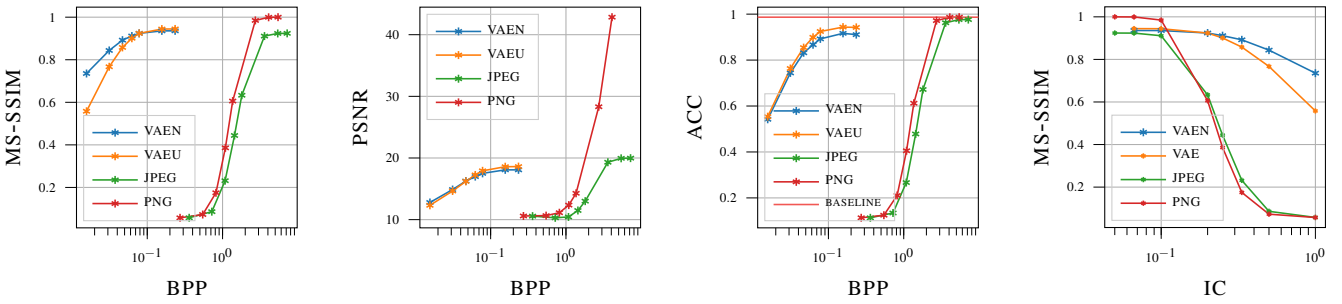
A. Results

Next, we present our results where we simulate an approximate DNA storage medium, as if data would be sent through a communication channel.

Simple Binary Channel. We begin by simulating DNA storage with a simple binary channel with a bit flip probability of 0.05. The qualitative and quantitative results, presented in Fig. 1 and Table III respectively, show that the VAEN variant (non uniform quantization) outperform the baseline VAEU. In this simple experiment, PNG and JPEG encoding schemes failed to reconstruct any of the images that were compressed and sent through the channel.

Full DNA simulated channel. In this more realistic experiment, we simulate DNA storage using *pg_oligo_archive* to encode the latent space representation of images into oligo strings. To simulate the DNA channel, we rely on a widely used sequencing simulator *randomreads*[1] that takes the oligos generated by *pg_oligo_archive* as a reference and simulates the sequenced reads that contain various substitution, insertion, and deletion errors using the default Illumina sequencing error

[1]https://sourceforge.net/projects/bbmap/



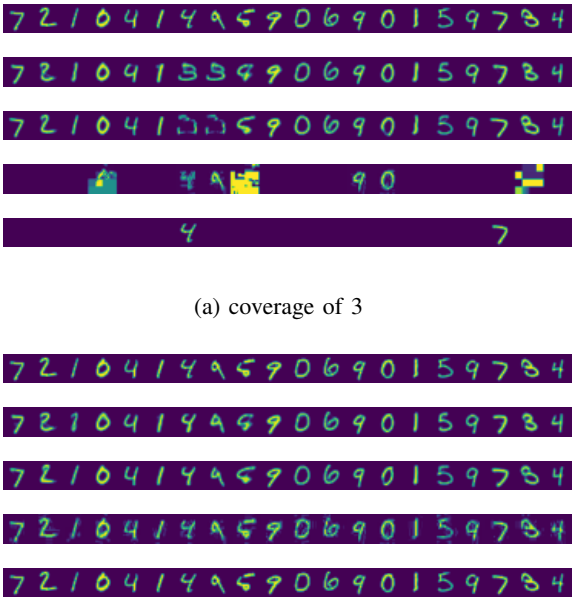
(a) (b) (c) (d)

Fig. 2: Comparison of different compression algorithms

model. In order to simulate the effect of coverage on accuracy, we configure the simulator to generate different sets of reads with different sequencing coverage ranging from 1× to N×, where N is configured to achieve complete recovery with our decoding pipeline. We use the decoding pipeline we developed in prior work to infer original oligos from the reads and convert them back into their binary form.

Quantitative results are presented in Fig. 2a,Fig. 2b,Fig. 2c in terms of MS-SSIM,PSNR and retained ACC. By comparing the two proposed approaches, VAEU and VAEN, with the standard compression schemes PNG and JPEG a clear trend repeats for all metrics: our methods achieve orders of magnitude better compression rates (notice that the BPP is represented in log-scale). In Fig. 2d we report the MS-SSIM as a function of the inverse of the coverage (IC $= 1/coverage$), clearly showing that although at high recovery rates JPEG and PNG outperform the VAE schemes, the latter are much more resilient to channel errors. A qualitative comparison of the various algorithms, for a coverage of 3 and 10, is presented in Fig. 3a and Fig. 3b respectively. At a low coverage the VAE variants are more resilient to channel errors (the VAEN seems qualitatively better) while the decoding of JPEG and PNG fails in the majority of the cases. This is not surprising as JPEG and PNG are designed to operate in error-free settings. At higher coverages qualitatively we observe no difference among different algorithms.

In summary, our approach outperforms classical methods by more than 10× and is more resilient to channel errors. As the cost of DNA synthesis is directly proportional to the size of data stored and the sequencing coverage used, it is clear that using generative models for compressing data and viewing DNA as an approximate storage medium can provide a substantial cost reduction for applications that do not need precise storage.



(a) coverage of 3

(b) coverage of 10

Fig. 3: Comparison of reconstructed images with different coverages. From top to bottom (in both figures): original images, VAEN reconstructions, VAEU, JPEG and PNG.

V. Conclusions

Recent research has clearly demonstrated the benefit of using DNA as a precise digital storage medium. However, cost remains one of the main road blocks in wide spread adoption of DNA-based data storage. In this work, we open up DNA for wider adoption by showing that it is possible to achieve orders of magnitude reduction in cost by targeting workloads that do not need precise storage, and by treating DNA as an approximate storage medium. Central to our approach was the use of data-driven, generative machine learning models that can provide much a higher compression rate and error tolerance over generic compression schemes. There are several avenues of future that we are pursuing. First, we are planning on carrying out real synthesis and sequencing experiments to validate our idea with real wet-lab experiments. Second, we are expanding this work to investigate the effect of long-read sequencers and new enzymatic synthesis approaches that introduce more errors. Third, we are exploring data types other than images to identify the utility of approximate DNA storage in other settings. Finally, we are exploring the design of joint source–channel encoding schemes based on neural networks that have gained popularity recently[22], [4], [24] in the context of DNA storage.

REFERENCES

[1] R. Appuswamy and V. Joguin. Universal layout emulation for long-term database archival. In *CIDR*, 2021.

[2] R. Appuswamy, K. Lebrigand, P. Barbry, M. Antonini, O. Madderson, P. Freemont, J. MacDonald, and T. Heinis. OligoArchive: Using DNA in the DBMS storage hierarchy. In *CIDR*, 2019.

[3] Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6228–6237, 2018.

[4] K. Choi, K. Tatwawadi, A. Grover, T. Weissman, and S. Ermon. Neural joint source-channel coding. In *International Conference on Machine Learning*, pages 1182–1192. PMLR, 2019.

[5] G. M. Church, Y. Gao, and S. Kosuri. Next-Generation Digital Information Storage in DNA. *Science*, 337(6102), 2012.

[6] T. M. Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[7] Y. Dubois, B. Bloem-Reddy, K. Ullrich, and C. J. Maddison. Lossy compression for lossless prediction. In *Neural Compression: From Information Theory to Applications – Workshop @ ICLR 2021*, 2021.

[8] Y. Erlich and D. Zielinski. DNA Fountain enables a robust and efficient storage architecture. *Science*, 355(6328), 2017.

[9] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney. Toward Practical High-capacity Low-maintenance Storage of Digital Information in Synthesised DNA. *Nature*, 494, 2013.

[10] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan. Deep learning with limited numerical precision. In *International conference on machine learning*, pages 1737–1746. PMLR, 2015.

[11] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898, 2017.

[12] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[14] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[15] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[16] OligoArchive. Oligoarchive: Eu fet initiative on intelligent dna data storage. https://oligoarchive.eu/.

[17] L. Organick, S. D. Ang, Y.-J. Chen, R. Lopez, S. Yekhanin, K. Makarychev, M. Z. Racz, G. Kamath, P. Gopalan, B. Nguyen, C. N. Takahashi, S. Newman, H.-Y. Parker, C. Rashtchian, K. Stewart, G. Gupta, R. Carlson, J. Mulligan, D. Carmean, G. Seelig, L. Ceze, and K. Strauss. Random access in large-scale DNA data storage. *Nature Methods*, 11(5), 2014.

[18] A. Razavi, A. van den Oord, and O. Vinyals. Generating diverse high-fidelity images with vq-vae-2. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[19] D. Reinsel, J. Gantz, and J. Rydning. Data age 2025: The evolution of data to life-critical. 2017.

[20] J. T. Rolfe. Discrete variational autoencoders. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[21] Y. Ruan, K. Ullrich, D. Severo, J. Townsend, A. Khisti, A. Doucet, A. Makhzani, and C. J. Maddison. Improving lossless compression rates via monte carlo bits-back coding. *ICLR 2021 neural compression workshop*, 2021.

[22] Y. M. Saidutta, A. Abdi, and F. Fekri. Joint source-channel coding over additive noise analog channels using mixture of variational autoencoders. *IEEE Journal on Selected Areas in Communications*, 2021.

[23] S. Santurkar, D. Budden, and N. Shavit. Generative compression. In *2018 Picture Coding Symposium (PCS)*, pages 258–262. IEEE, 2018.

[24] Y. Song, M. Xu, L. Yu, H. Zhou, S. Shao, and Y. Yu. Infomax neural joint source-channel coding via adversarial bit flip. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5834–5841, 2020.