Creating Synthetic Genomes using Diffusion Models

Philip Kenneweg
AG Machine Learning
University of Bielefeld
pkenneweg@*

Raghuram Dandinasivara
AG Genome Data Science
University of Bielefeld
rdandinasivara@*

Alexander Schönhut
AG Genome Data Science
University of Bielefeld
aschoen@*

Barbara Hammer
AG Machine Learning
University of Bielefeld
bhammer@*
*@techfak.uni-bielefeld.de

Abstract

In this paper, we introduce the first diffusion model designed to generate complete synthetic human genomes. These genomes closely mimic real human genomes and serve as a nearly perfect substitute for genuine ones in disease classification tasks. This innovation addresses a significant challenge in current research: real human genomes are highly sensitive private data and cannot be widely distributed due to privacy concerns. By using synthetic genomes, researchers can train AI models and identify correlations without stringent data privacy measures. Additionally, training diffusion models allows the inclusion of unlabeled data, reducing the reliance on large labeled datasets.

1 Introduction

The field of deep learning has enabled various breakthrough technologies in the biotech sector [Jumper et al., 2021, Wong et al., 2024, Cheng et al., 2023]. While these technologies constitute major advancements, they investigated inputs in the form of small sections of available data due to computational constraints. In this work we take the first steps towards processing the whole human genome at once using modern deep learning techniques, while keeping the required compute to a minimum.

A major problem when analyzing human genomes is the size of the data as well as the high amount of complexity inherent in the data and unclear interaction pathways between different genomes.

In Yelmen et al. [2021] they use GAN networks to generate small sections of human genomes and use a variety of metrics like linkage disequilibirum and nearest neighbour adversarial accuracy to measure the quality of their creates genomes.

We postulate that by compressing the human genome using prior biological knowledge, as well as using the latest deep learning techniques, we can solve the common problem of sharing highly sensitive data by replacing the real data with high quality synthetic data.

Preprint. Under review.

2 Related Work

2.1 Unsupervised Learning on Genetic Data

A popular paradigm in recent successful deep learning schemes, is pre-training large models on unlabeled data to make use of large amounts of training data, without needing tedious or sometimes impossible human provided labels.

The transformer model [Vaswani et al., 2017] popularized this approach on the natural language domain by combining it with architectures capable of learning arbitrary relations in data.

The same approach is possible for the genetics domain and will enable many use cases previously thought impossible, similar to the language domain.

2.2 Working with Long Sequences

A common problem when working with genetic data is the size of the data and the long range interactions which have to be considered.

HyenaDNA [Nguyen et al., 2023] is a powerful state of the art transformer based architecture which is able to process up to a million tokens at once using an adapted form of attention. But even with this latest technology it fails to be able to process whole human genomes which are about 3 billion base pairs long.

Research in deep learning has investigated projecting data into a shared embedding space in which long sequences can be greatly compressed. The most widely known use case for this is the Vision Transformer [Dosovitskiy et al., 2021] which works directly with images, even though transformers typically can not process sequences of these sizes due to computational limitations. They achieve this feat by compressing regions, they call patches of an image into a single embedding. This shared projection is typically done by a small MLP or CNN which is the same for each image region.

Other works, especially in the diffusion domain, for example Stable Diffusion by Rombach et al. [2021] have also adopted the paradigm of no longer working directly on the data but rather on a compressed version of it. This is typically called working in the embedding space and we take great inspiration from this principle.

2.3 Diffusion Models

In this section we will give a rough outline of the diffusion process used. For a more detailed look at diffusion processes we recommend the works by Ho et al. [2020], Song et al. [2022]

2.3.1 Training

Diffusion Models are a relatively new sub-field in generative AI which add Gaussian noise $\epsilon = N(\mu, \sigma)$ with variance $\sigma = 1$ and mean $\mu = 0$ to data $x \in D$ to generate noised data x_n according to some noise schedule in $t \in (0, T)$ steps.

First we define some helpful variables:

$$\alpha(t) = 1 - \beta(t); \overline{\alpha}(t) = \prod_{s=1}^{t} \alpha_s; \tilde{\beta}(t) = \frac{1 - \overline{\alpha}(t-1)}{1 - \overline{\alpha}(t)} \beta(t) \tag{1}$$

where β_t represents the variances of the forward process, in practice it is a pre-determined scalar schedule increasing linearly from $\beta(1) = 10^{-4}$ to $\beta(T) = 0.02$, see Ho et al. [2020].

$$x_n(t) = \sqrt{\overline{\alpha}(t)} \cdot \epsilon + \sqrt{1 - \overline{\alpha}(t)} \cdot x \tag{2}$$

The process is designed for a data distribution D with mean $\sigma = 1$ and variance $\mu = 0$, which are easy pre-processing steps.

The recovery task is given the noised image and the noise schedule t, x_n to predict the previously added noise ϵ we call the predicted noise ϵ_p. This is typically realized by an artificial neural network E.

$$\epsilon_p = E(t, x_n(t)) \tag{3}$$

To recover the original data x in a single step the predicted x_p can be computed according to:

$$x_p(t, x) = \frac{x_n(t) - (1 - \sqrt{\overline{\alpha}(t)}) \cdot \epsilon_p}{\sqrt{\overline{\alpha}(t)}} \tag{4}$$

The loss function $L(x)$ of our neural network is simply:

$$L(x) = ||\epsilon, \epsilon_p(x)|| \tag{5}$$

where $||\cdot, \cdot||$ is some distance norm, in our case we use the L2 norm also called mean squared error. The loss is not based on the recovered data x_p as the predicted noise ϵ_p contains the same information but with less intermediate steps. During training t is drawn from a uniform random distribution for each input sample.

2.3.2 Data Generation

Generating new data is done analogously to recovering the original data. Starting from complete noise $x_{n,T} = N(0, I)$, the newly generated data-point is recovered by iteratively refining the prediction of new noised x_p according to Eq 4 but each step t new noise is added to remain in the same distribution as the original x_n.

$$x_{n,t-1} = \frac{1}{\sqrt{\alpha(t)}} \cdot (x_t - \frac{1 - \alpha(t)}{\sqrt{1 - \overline{\alpha}(t)}} \epsilon_p(x_{n,t}, t)) + \sqrt{\tilde{\beta}(t)} N(0, I) \tag{6}$$

This process is called DDPM [Ho et al., 2020] and has been highly successful in generating realistic artificial images and other data.

Furthermore, more information can be incorporated by presenting so called conditioning information y to the neural network, changing Equation 3 to:

$$\epsilon_p = E(t, x_n(t), y) \tag{7}$$

typically y is an input vector containing additional information about the data sample x. For example, in the image case this could be the caption of the image, or whether or not a certain class is present in the image.

3 Methods

To generate synthetic data and thereby enable sharing of genetic data without privacy concerns we train a diffusion model on real data to produce an arbitrary amount of synthetic data.

3.1 Data

Although in this work we establish a model that is generic with respect to the kind of SNPs of a human genome it can predict, we focus on the occurrence/prevalence of ALS as a phenotype in the following. The reasons for this are the availability of the excellent, sufficiently large amounts of data that were raised in the frame of Project MinE pro [2018]. Most importantly, we emphasize ALS, based on the aforementioned large missing heritability. For analogous reasons, also the earlier related studies [Auer et al., 2012, Dolzhenko et al., 2017] rely on data raised in the frame of Project MinE.

The data on which we are working on has 18.279 genes sequenced for 10.405 humans. Each of these genes is a sequence of between 5-100 different SNPs.

3.2 Data Pre-processing

First we take a closer look at the available data and how it can be compressed to facilitate deep learning on it.

A human genome consists of 3 billion base pairs. Analyzing the whole genome for each use case is generally not possible due to its length. A common alternative in biotech is analyzing the parts of the human genome which change between individual humans. This is done using single nucleotide polymorphisms (SNP). Data in this form is available from pro [2018].

To further compress this data and make it usable we follow the protocol from Luo et al. [2023], where these SNPs are individually compressed using Principle Component Analysis (PCA) of varying lengths, up to a maximum of 8 dimensions. This further reduces the dimensionality while incurring minimal compression loss ($< 1\%$). A rough outline of this process can be seen in Figure 1

For further pre-processing we zero-pad all PCAs to be of length 8 to produce a vector of $x \in \mathbb{R}^{18279 \times 8}$ as a representation of our data. We further pad this with zeroes to be $x \in \mathbb{R}^{18432 \times 8}$ for efficiency gains on modern architectures (18432 is divisible by 2^{11}). We found that all of this zero-padding, while necessary, results in samples leaving the typically input distribution during the diffusion generation process. To remedy this we set the loss $L(x)$ at all zero padded position to zero, as well as modify the diffusion process by clamping all zero padded positions of x_p to zero. Note that the zero padding is the same for all inputs, so this operation is easy and efficient to apply.

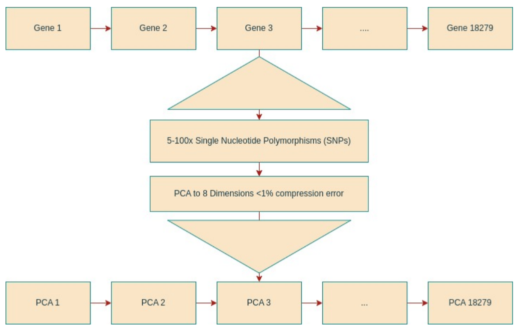

Gene 1
Gene 2
Gene 3
....
Gene 18279
5-100x Single Nucleotide Polymorphisms (SNPs)
PCA to 8 Dimensions <1% compression error
PCA 1
PCA 2
PCA 3
...
PCA 18279

Figure 1: Pre-processing pipeline

3.3 Model Architecture

We base our Diffusion Model on the popular Unet Architecture by Ronneberger et al. [2015] but make some changes due to the sequential nature of genetic data. We evaluate multiple possible Unet architectures where we replace the 2D variants of convolutions with their 1D equivalents or with a Multi-Layer-Perceptron. Another variant which was tried, was a standard transformer encoder structure similar to the one in Devlin et al. [2019], but with learnable positional embeddings, as well as two additional tokens to inject the time t and class y information and an embedding layer similar to the one used for images in VIT[Dosovitskiy et al., 2020] to reduce the number of input tokens. In Figure 2 we visualize the proposed UnetMLP architecture. The 1D convolutional architecture includes multi-headed attention in the intermediate layers to share global information, this is very similar to what is done for images and needed to reduce the sequential bias introduced by convolutions. The dense layer architecture does not incorporate this feature.

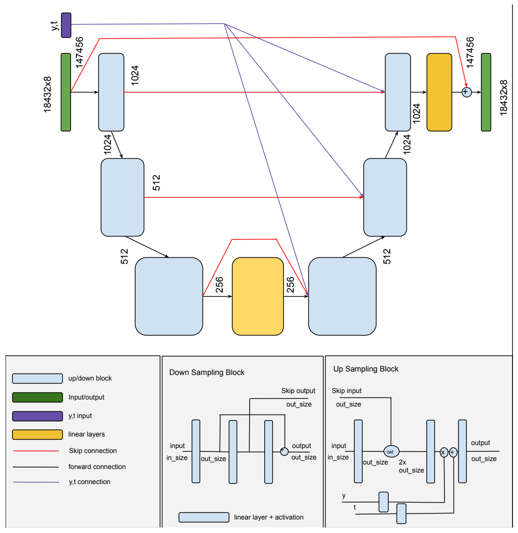


Figure 2: A structural overview of the architecture of the MLP diffusion model.

In general we want to point out that the approach using convolutions introduces a sequential bias, which does not make sense in this context, but it greatly limits the amount of parameters which have to be learned.

In contrast the dense model does not incorporate any bias, but it has been shown in many domains that even though in principle fully connected layers can approximate arbitrary functions, they fail to find suitable solutions due to their over-parameterization.

A fully attention based model like the transformer encoder also does not incorporate any spatial bias and should be suitable for this kind of data.

3.4 Combining Models

Since CNN and MLP based models focus on different aspects of the structure of the genome we suggest combining them into a single network thus eliminating both weaknesses. We call this combination CNN + MLP. We combine them by simply adding the predicted noise of each single model $MLP(x)$ and $CNN(x)$ according to:

$$\textit{MLP + CNN}(x) = (1 - \lambda(t)) \cdot MLP(x) + \lambda(t) \cdot CNN(x) \tag{8}$$

where $\lambda(t)$ is a learnable function in the range $[0, 1]$, realized by a simple multi layer perceptron with 2 layers which has the noise schedule t as input.

In summary we propose 5 different Model architectures for the Diffusion Task: Baseline Model, Unet MLP, Unet CNN, Unet MLP + CNN and Transformer.

We perform extensive hyper-parameter tuning on all types of models and present the best performing results.

3.5 Evaluation

The automatic evaluation of genetic diffusion models is a challenging task. Diffusion models were first pioneered in the realm of image generation, where it is comparatively easy to check if the generated data is of high quality. A human is able to inspect the generated images and can manually determine if they are realistic. In the genetic case this is no longer possible, as a human observer is not able to easily tell if a genetic sequence is of high fidelity.

Furthermore, common automatic measurements from the image domain can not be applied to the genetics domain, complicating evaluation. Fréchet inception distance (FID) [Heusel et al., 2017] and Inception score (IS) [Salimans et al., 2016] are both automatic metrics from the image domain which do not rely on a human observer and are widely regarded as good baselines for synthetic image evaluation. The problem with applying these to the genetic domain is that they rely on models which are pre-trained on large scale data to project images into a meaningful embedding space. While this kind of data is easily available in the image domain, in the genetic domain neither pre-trained models, nor the data to train them, nor data standardization are agreed upon or easy to obtain.

3.5.1 Nearest Neighbour adversarial accuracy

To find alternatives to these common scores we look at other scores which compare distributions. The Nearest Neighbour adversarial accuracy is defined as Yale et al. [2019]:

$$\begin{aligned} AA_{truth} &= \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(d_{TS}(i) > d_{TT}(i)) \\ AA_{syn} &= \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(d_{ST}(i) > d_{SS}(i)) \end{aligned} \tag{9}$$

Where $d(i)$ is denoting the distance of datapoint with index i to the nearest neighbour, S is denoting the synthetic and T the real data origin. For example $d_{ST}(i)$ is the distance between the synthetic datapoint with index i and the nearest neighbour from the real distribution. A score of $AA_{truth} = 0.5$ and $AA_{syn} = 0.5$ is considered optimal. To calculate the distance to the nearest neighbour $d(i)$ we use the cosine distance metric d_{cos}. e.g:

$$d_{cos}(x, y) = \arccos(\frac{x \cdot y}{|x| \cdot |y|})\frac{1}{\pi} \tag{10}$$

3.5.2 Nearest Neighbour test

The nearest neighbour (NN) test with score S compares which distribution the k nearest neighbours belong to.

$$S(D_1, D_2) = \frac{1}{\#D_1} \sum_{D_1}^{x} \frac{1}{k} \sum_{0}^{k} \delta(x_k) \tag{11}$$

where $\#D$ is the number of samples in D and x_k the k-th nearest neighbour to x in distribution $D_a = D_1 \cup D_2$, according to some distance metric, we choose the cosine distance metric. We define $\delta(x_k) = 1$ if $x_k \in D_1$ and 0 otherwise. An optimal score would be $S = 0.5$.

3.5.3 UMAP

Another common evaluation possibility is the visualization of neighbourhood structure using algorithms like UMAP [McInnes et al., 2018] or T-SNE [van der Maaten and Hinton, 2008]. Both of these methods are highly popular in the realm of Machine Learning. We choose to use UMAP due to its better performance on high dimensional data.

3.5.4 Amyotrophic Lateral Sclerosis (ALS) classification

As the final and most important evaluation metric we use a real life binary classification task. The classification task we choose is to determine whether or not the genetic disease Amyotrophic Lateral Sclerosis is present. This task is especially suitable to evaluate if a realistic genome is produced due to its relatively high complexity, as well as the comparatively large amount of training samples which are present for this task. The task is based on 10.405 examples which were diagnosed by medical professionals into 3192 ALS positive and 7213 negative samples. The task does not exhibit simple reliance on one or even multiple genomes and at least 1000 genomes are needed to obtain good performance [Luo et al., 2023].

We train a simple classifiers to perform binary classification. If the classifier can also be trained on synthetic data and still performs well on a holdout test set of real data we can conclude that the features used for ALS classification were well reproduced. To ablate against the type of classifier having a large impact we evaluate using 3 different classifiers: MLP, Transformer and CNN.

All evaluations will be performed on a balanced hold out test set of 520 positive and 520 negative examples for a total of 1040 samples or 10% of the total available data. We follow the experimental protocol of Luo et al. [2023] to achieve easy to compare results.

4 Experiments

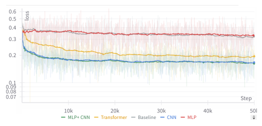
In this section we will look at the results of the models proposed in Section 3.

4.1 Training Metrics

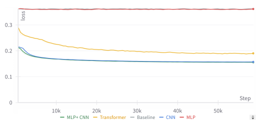
First we show loss curves during training on train and validation data of the different diffusion model types, see Figure 3.

We observe that no models have any problems with over-fitting. The CNN based models overall performs best with regards towards minimizing the train loss.

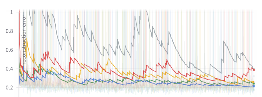
However, if we take a closer look at the reconstruction error of the model, visualized in Figure

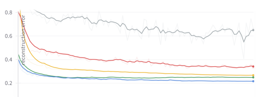


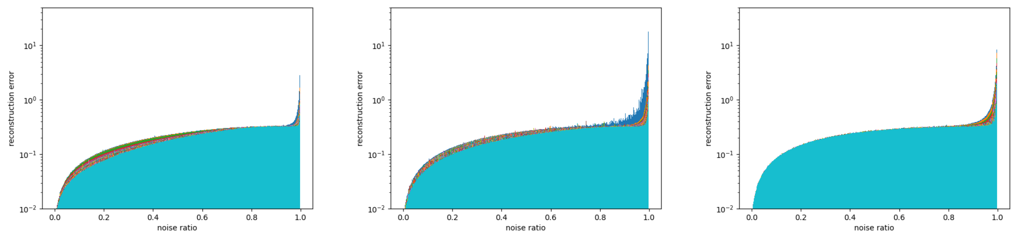
(a) train loss



(b) validation loss







(a) CNN (b) Transformer (c) MLP

Figure 4: Reconstruction error vs noise curves during training for different diffusion model backbones. Note the logarithmic scaling of the y-axis.

3, we note that the MLP model actually performs surprisingly similar to the CNN model and that even though only very small drops in loss during training can be observed for the MLP, the reconstruction error keeps improving. The Transformer based architecture seems to be performing decently in loss and reconstruction error. We note that reconstruction error is highly related to the loss, but scaled by the noise schedule t, e.g. this error focuses more on large values of t.

To further analyze this, in Figure 4 we visualize the reconstruction error curves vs amount of noise added of the different diffusion models. We can see that while the CNN has an overall better performance it focuses more on the fine detail of the structure e.g. recovering from small noise values, while the MLP architecture is more focused on recovering rough structure (e.g. high noise).

4.2 Evaluation

Next we take a closer look at the performance of a classifier trained on synthetic vs real data, see Table 1. We observe the MLP generated data performing superior compared to all other synthetic data types with around 84.7% accuracy, whereas the CNN based data fails to capture the relations important for ALS classification with the best performance of 62.64% and the Transformer generated data having a maximum accuracy of 64.62% on the real test data. Training on real data is performing best with 87.6% accuracy, as expected.

In Figure 5, we visualize the distribution of different data types using UMAP [McInnes et al., 2018] with cosine distance metrics. Euclidean distance metrics are notoriously bad for high dimensional data and return uniform distributions. We observe that the UnetMLP generated data is closest to the real data in the UMAP visualization. The Baseline generated data is close

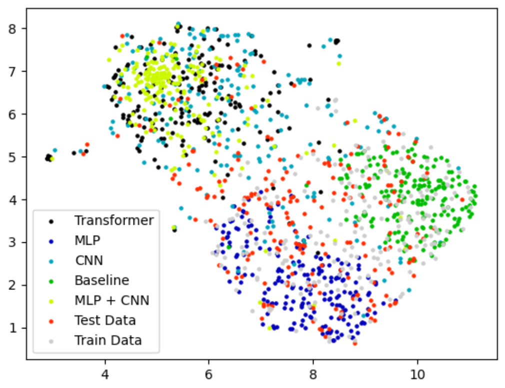


Figure 5: UMAP visualizations of data of different origins. Red and grey points are test and train data (real). Blue is MLP, green is Baseline. Yellow, black and bright blue are MLP + CNN, Transformer and CNN respectively.

Table 1: Accuracy on a hold out test set after training different ALS classifier (MLP, Transformer or CNN) on different synthetically generated data types (generated by: MLP, Transformer, CNN, MLP + CNN or Baseline). The best synthetic data for each classifier type is marked in **bold**.

Classifier	Real Data	CNN	MLP	MLP + CNN	Transformer	Baseline
MLP	87.60	62.64	**84.71**	82.21	64.62	70.19
Transformer	82.11	54.23	**76.73**	71.06	56.92	45.10
CNN	73.17	51.15	**67.11**	50.00	50.29	50.00
average	80.79	56.01	**76.18**	66.01	57.28	55.10

Table 2: Result of nearest neighbour test (k=10) for all datasets. Closer to 0.5 is better. Best performance is marked in **bold**.

comparison data	CNN	MLP	MLP + CNN	Transformer	Baseline
train	0.7835	0.0165	**0.4039**	0.7935	0.0265
test	0.7755	0.2825	**0.3740**	0.765	0.3515

to some of the training data (grey) but comparatively little of the test data (red), indicating over-fitting on the training set, this is especially interesting as Figure 3 does not indicate any over-fitting during training. The Transformer, CNN and MLP + CNN generated points are all clustering together with only small overlap with the training or test data. Even though this UMAP visualization can give us an indication of neighbourhood structure, we want to emphasize that the visualizations can be highly misleading. In this example, MLP + CNN and CNN generated points are similarly visualized while the classification task performance, see Table 1, is very different in between both groups.

Next we look at the nearest neighbour test and the nearest neighbour adversarial accuracy, see Tables 3 and 2. We observe MLP+CNN performing best for the nearest neighbour test and CNN performing best for nearest neighbour adversarial accuracy. While these test shed some light on the quality of the generated data, we do not think that distance metrics on the high dimensional input data are especially meaningful. This strongly reduces the value of these tests.

5 Conclusion

In this work we presented the first full human genome diffusion models. By creating synthetic human genomes we are able to address the problem of sharing highly sensitive data by replacing it with non sensitive synthetic data. We show that the synthetic data quality is sufficient to train on and achieve good results for ALS classification, as well as use a variety of other metrics to evaluate the quality of the generated data.

Further work is needed to extend this proof of concept to real world applications. The first step could be to improve the generated genomes, either by increasing the available training data for training the diffusion model or improving the diffusion model architecture.

We look forward to the input of the scientific community and potential applications.

We publish all code at [anonymized for review].

Table 3: Result of Nearest Neighbour adversarial accuracy for generated datasets. Closer to 0.5 is better. Best performance is marked in **bold**.

	CNN	MLP	MLP + CNN	Transformer	Baseline
AA_{truth}	0.755	0.225	**0.415**	0.915	0.35
AA_{syn}	**0.62**	1.0	0.925	0.65	0.995

References

Project mine: study design and pilot analyses of a large-scale whole-genome sequencing study in amyotrophic lateral sclerosis. *European Journal of Human Genetics*, 26(10):1537–1546, 2018.

Paul L Auer, Jill M Johnsen, Andrew D Johnson, Benjamin A Logsdon, Leslie A Lange, Michael A Nalls, Guosheng Zhang, Nora Franceschini, Keolu Fox, Ethan M Lange, et al. Imputation of exome sequence variants into population-based samples and blood-cell-trait-associated loci in african americans: Nhlbi go exome sequencing project. *The American Journal of Human Genetics*, 91(5):794–808, 2012.

Jun Cheng, Guido Novati, Joshua Pan, Clare Bycroft, Akvilė Žemgulytė, Taylor Applebaum, Alexander Pritzel, Lai Hong Wong, Michal Zielinski, Tobias Sargeant, et al. Accurate proteome-wide missense variant effect prediction with alphamissense. *Science*, 381(6664):eadg7492, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

Egor Dolzhenko, Joke Jfa Van Vugt, Richard J Shaw, Mitchell A Bekritsky, Marka Van Blitterswijk, Giuseppe Narzisi, Subramanian S Ajay, Vani Rajan, Bryan R Lajoie, Nathan H Johnson, et al. Detection of long repeat expansions from pcr-free whole-genome sequence data. *Genome research*, 27(11):1895–1903, 2017.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. 2020.

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

Xiao Luo, Xiongbin Kang, and Alexander Schönhuth. Predicting the prevalence of complex genetic diseases from individual genotype profiles using capsule networks. In *Nature Machine Intelligence*. Nature Publishing, 2023. ISBN 978-3-319-24574-4.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. doi: 10.21105/joss.00861. URL `https://doi.org/10.21105/joss.00861`.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton Rabideau, Stefano Massaroli, Yoshua Bengio, Stefano Ermon, Stephen A. Baccus, and Chris Ré. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL `http://jmlr.org/papers/v9/vandermaaten08a.html`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Felix Wong, Erica J Zheng, Jacqueline A Valeri, Nina M Donghia, Melis N Anahtar, Satotaka Omori, Alicia Li, Andres Cubillos-Ruiz, Aarti Krishnan, Wengong Jin, et al. Discovery of a structural class of antibiotics with explainable deep learning. *Nature*, 626(7997):177–185, 2024.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P Bennett. Privacy preserving synthetic health data. In *ESANN 2019-European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2019.

Burak Yelmen, Aurélien Decelle, Linda Ongaro, Davide Marnetto, Corentin Tallec, Francesco Montinaro, Cyril Furtlehner, Luca Pagani, and Flora Jay. Creating artificial human genomes using generative neural networks. *PLoS genetics*, 17(2):e1009303, 2021.