Generative Moment Matching Networks for Genotype Simulation

Maria Perera[1], Daniel Mas Montserrat[2], Míriam Barrabés[1], Margarita Geleta[3], Xavier Giró-i-Nieto[1], and Alexander G. Ioannidis[2,*]

[1]Universitat Politècnica de Catalunya, [2]Stanford University, [3]UC Irvine

***Abstract*— The generation of synthetic genomic sequences using neural networks has potential to ameliorate privacy and data sharing concerns and to mitigate potential bias within datasets due to under-representation of some population groups. However, there is not a consensus on which architectures, training procedures, and evaluation metrics should be used when simulating single nucleotide polymorphism (SNP) sequences with neural networks. In this paper, we explore the use of Generative Moment Matching Networks (GMMNs) for SNP simulation, we present some architectural and procedural changes to properly train the networks, and we introduce an evaluation scheme to qualitatively and quantitatively assess the quality of the simulated sequences.**

I. INTRODUCTION

Genomic studies increasingly make use of large datasets and biobanks composed of thousands to hundreds of thousands of individual sequences. However, there are several challenges when dealing with such personal genomes: first, privacy restrictions can limit the scope of data sharing and access. Second, some of these datasets are very large introducing difficulty in storage, transfer, and processing. Finally, many of these datasets are highly imbalanced with extreme overrepresentation of some population groups (commonly European-descent individuals), and underrepresentation of others, leading to models that perform poorly when faced with individuals from underrepresented population groups [1]. Effective simulation tools, such as generative neural networks, can help to mitigate some of these challenges. By sharing trained generative networks, new synthetic sequences can be created to train machine learning models without the need to share the original sequences, (which can be prohibitive due to their size or privacy). Furthermore, through conditional synthesis (generation of sequences given a desired phenotype or ancestry), bias within datasets or biobanks can be partially removed by augmenting datasets with simulated samples from underrepresented groups.

Modern biobanks used in GWAS and similar studies typically include sequences of single nucleotide polymorphisms (SNPs). SNPs are the positions (less than 1%) within the human genome that are known to change between individuals. These positions (when biallelic) can be codified in a binary system, with 0's denoting the common variant, and 1's the minority variant. The nature of the sequences needs to be taken into account when designing generative networks: SNP sequences are sparse (they include many more 0s than 1s), high-dimensional, and commonly lack repetitive motifs. This completely differs in nature from other common signals, such as images or audio. The frequency distribution and correlations between SNPs vary across population groups, which can cause the techniques and treatments developed using data from one group not to generalize well to other population groups.

In this paper, we make use of a Generative Moment-Matching Network [2] to generate realistic genotype sequences given a desired ancestry. GMMNs, described in more detail in the following sections, are able to generate synthetic data that match the statistical moments of real data. This method has the same intent as other generative approaches, such as Generative Adversarial Network (GAN) [3] or Variational Autoencoders (VAEs) [4]. However, GMMNs can be easier to train than GANs or VAEs. Training a GAN requires solving a commonly difficult min-max optimization problem between two networks: a generator, and a discriminator. Similarly, a VAE tries to minimize reconstruction error and a KL-Divergence with two networks: an encoder and a decoder (generator). While such approaches require jointly training two networks, GMMNs require only training one unique network (the generator), which can lead to lower memory and computational requirements. In addition, GMMNs do not need to observe the data directly (as GANs or VAEs do), but instead can observe "sketches" (e.g. random feature descriptors [5]) that capture the statistical properties of the database as a whole. This allows for a privacy-friendly setting and permits a more accessible data sharing framework.

The contributions of this work are as follows: we introduce two modifications to the GMMN framework, a binarization step, and an iterative re-start of random projections, both described in section 3, that allow for a correct training and sequence simulation. Additionally, we propose two evaluation schemes: a qualitative evaluation through dimensionality reduction techniques, and a quantitative evaluation through supervised machine learning based classifiers.

II. RELATED WORK

Generating realistic synthetic genotype sequences remains a challenge, as there are several evolutionary (selection and mutation) and demographic forces (drift) influencing genomes. Accurate simulation models should provide realistic allele frequency patterns and linkage disequilibrium

*Corresponding author: ioannid@stanford.edu

(LD) profiles. There is an extensive literature of simulation methods based on evolutionary models such as the Wright-Fisher model [6]. Such techniques simulate recombination and create synthetic descendants by using available genetic sequences as ancestors. However, these models ignore evolutionary forces including selection and mutation. Coalescent simulation [7], [8] is a standard procedure to generate artificial genotypes under various models, usually seen as an extension of the classic Wright-Fisher model. Examples of such include Hudson's ms program [9] that allows for mutation and migration within subpopulations, and ms reimplementations with improvements for modern datasets such as msprime [10], mbs [11], msms [12]. Because these models simplify reality, each of them has their own limitations, as the goal of any simulation tool is to find a compromise between the accuracy and efficiency [13].

Recently, new data-driven deep learning based simulation procedures have been introduced into the field – without defining an explicit evolutionary and demographic model. Connectionist approaches rely on learning models capable of generating realistic genotypes. The first technique to simulate SNP sequences from a population-based perspective was a class-conditional VAE-GAN by Montserrat et al. [14], which showed success on training local ancestry inference techniques. Yelmen et al. generate realistic surrogates of real genotype snippets with GANs and Restricted Boltzman Machines (RBM) [15]. Battey et al. tested VAE for genotype simulation and showed scaling issues of the method [16]. More recently, Geleta et al. introduced an ancestry-conditional VAE model for realistic genotype simulation [17], [18] with high-fidelity with respect to real genotype entropy variation and LD patterns. Generative models, additionally, provide a means for estimating meaningful low-dimensional representations which can be used for genotype imputation, classification, and visualization [17]. While being promising, such generative models do not provide extensive evaluation procedures to properly determine the quality of the simulated sequences. Some methods adapt a utility-preserving evaluation [14] where the quality of the sequences is assessed by their capacity to train neural networks in downstream task. Other methods use first order statistics or visualizations with a limited amount of dimensionality reduction techniques [15], [17]. Therefore, a proper procedure for simulation quality evaluation is still required.

III. PROPOSED METHOD

A. Generative Moment Matching Networks

GMMNs [2] are deep generative models able to generate new samples that statistically resemble the training samples. Such networks learn a mapping $\tilde{x} = g(z)$ from a known distribution (typically Gaussian) into the data space; new samples are generated by sampling random vectors from $z \sim \mathcal{N}(0, I)$ and processing them through the network. GMMNs are trained by minimizing Maximum Mean Discrepancy (MMD) criterion, which is a frequentist estimator that tries to determine if two sets of samples, $X = \{x_i\}_1^N$ and $Y = \{y_i\}_1^N$, (in this case real and fake sequences) come from the same or different distributions. Specifically, MMD can be represented as the mean squared difference between two statistics:

$$\ell_{\text{MMD}^2} = \left\| \frac{1}{N} \sum_{i=1}^{N} \phi(x_i) - \frac{1}{M} \sum_{j=1}^{M} \phi(y_j) \right\|^2 \tag{1}$$

where $\mu_x = \frac{1}{N} \sum_{i=1}^{N} \phi(x_i)$ is a statistic, or "sketch", computed on the samples X. If the statistics μ_x and μ_y are similar, then the two datasets are likely to come from the same distribution.

Taking ϕ as the identity function, the statistic μ_x is equivalent to the sample mean, which for genotype sequences represents the frequency of each SNP. Other choices of ϕ can be used to capture higher order moments of the data distribution. Namely, as specified in [2], the kernel trick can be applied to obtain an infinite dimensional feature space that can capture all moments of the data. However, this requires computing a pairwise kernel distance with every training sample for each batch during the training process, which can be computationally expensive. Instead, we can use random features [5] to approximate any kernel with a finite number of dimensions. The random features are computed as $\phi(x) = f(Wx)$, where $f(\cdot)$ is a non-linearity, and W is a random projection matrix following some pre-specified distribution. When the number of features (dimension of $f(Wx)$) tends to infinity, the feature space tends to the reproducing kernel Hilbert space of the actual kernel. The type of kernel that is approximated depends on the distribution of W and the activation function applied. In this work we make use of ReLU random features which approximate the arc-cosine kernel [19].

Note that in order to train the network, the actual real genomic sequences do not need to be accessed, but just the statistic, or sketch, μ_x is required. Typically, the sketch has a dimension D with $D \ll NM$, where N is the number of samples, and M is the number of SNPs at each sequence. This provides a more compact representation of the data which is easier to share and to incorporate into privacy-preserving pipelines. However, this advantage is lost if the kernel trick is applied as in [2], where the actual samples are required to compute the pairwise kernel distances.

B. Network Architecture

In this work, we perform ancestry conditional simulation by training a different GMMN for each ancestry group. Each GMMN is trained with a different sketch μ computed with the data of the specific ancestry group. This differs from other approaches were the ancestry label is provided as an extra input to the network [14]. While training a different network for each ancestry group can require a larger number of parameters (as many networks need to be trained), it provides a more flexible setting in scenarios where only the data of one ancestry group needs to be shared by sharing the specific trained network. For each ancestry-dependent network, we use the same architecture: a linear layer of

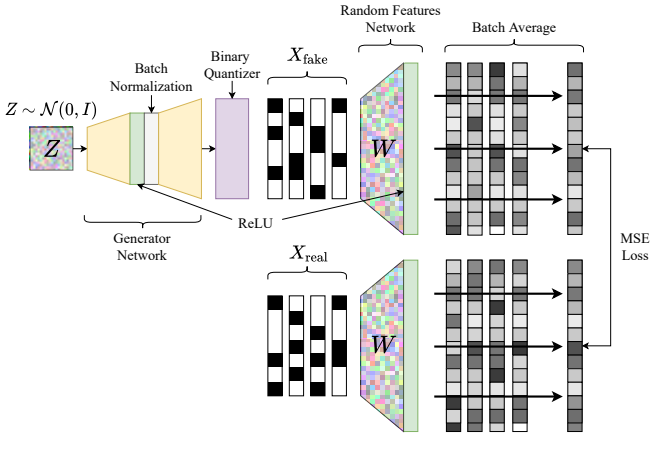

Fig. 1: GMMN with Random Features.

dimension 5000×4096, followed by a ReLU activation and a batch norm, followed by another linear layer of dimension 4096×5000, finishing with a binary quantizer. Each network is trained with the MMD loss with their respective sketch vectors, where the sketch of the real data is pre-computed to save computing time.

The random features are implemented with a random linear layer of dimension 5000×50000, sampled from a Gaussian distribution, followed by a ReLU activation function. Note that the random linear layer that computed the random features is not trained.

C. Output Binarization

Typically, generative networks output the simulated values through a linear or sigmoid layer. However, when simulating SNP sequences, only binary values are desired. Furthermore, binarizing the data through direct thresholding is a non-differentiable operator and does not allow the MSE loss signal to backpropagate through the network. Therefore, a differentiable alternative is required. In this work we make use of the same approach as in [20] where a hard threshold is applied in the forward pass (leading to binary simulated sequences), but the backward pass is approximated as if a scaled and shifted tanh operator was used during the forward pass.

D. Random Features Re-start

In order to improve the quality of the simulation, we propose the use of an iterative re-start strategy of the random projections and sketch vectors. After a fixed number of weight update steps or number of training epochs, the random weights of the random feature projections are re-sampled and the sketch vector of the training dataset is recomputed. This allows us to improve the accuracy of the simulation and avoids having the network overfit on the specific features and statistical moments captured by the sketch vector. Note that a sketch with much larger dimensionality would allow for the same level of simulation accuracy, however, a larger dimensionality of the sketch requires a larger number of features to be computed at every batch leading to a slower training procedure. By performing re-sampling we obtain networks that capture rich features without the need of computing very high-dimensional features at every batch.

E. Privacy And Data Sharing

Because access to the data is not required to train the network, and only access to the sketches μ are needed, our approach allows for a privacy-friendly setting with easy data sharing. By simply sharing the sketch vectors and the random projection matrices, different users can simulate new data by training GMMNs. Furthermore, as shown in [21] some computations can be already performed directly with the sketches, without the need of simulation of actual sequences. However, the sharing of sketches and/or trained networks, does not provide any strong guarantees on the privacy of the sequences used for training. Paradigms such as differential privacy can be incorporated easily into this framework in order to provide strong theoretical privacy guarantees, as it has been explored in [22], but we leave the exploration of its applications in genetic sequences as future work.

The use of different sketches for each ancestry group, and the re-start of random projection and sketches, can be seen as reflecting a real scenario where incoming available data comes from different sources (hospitals, academic labs, companies) each capturing populations from different countries and regions.

F. Dimensionality Reduction for Qualitative Evaluation

When simulating natural signals such as images or audio, visual or auditory inspection can be used to evaluate the quality of the generated data. However, the high-dimensional nature of genomic data does not allow for an effective evaluation of simulated sequences through direct subjective inspection. Therefore, dimensionality reduction of the sequences is required in order to perform visual evaluation. Principal Components Analysis (PCA) has already been used in previous works such as [17] to visualize the synthesized sequences. We extend this approach into a more generic framework: we suggest that any dimensionality reduction technique that allows for projecting new unseen samples can be used to perform qualitative evaluation by first learning a projection into a lower dimensional space (usually 2 or 3 dimensions) with the real data, and later projecting the synthetic data using the learnt projection. If the projection of the synthetic data differs from the projection of the real data, that prima facie indicates that the simulation is not accurate. If the projections are not distinguishable between real and fake data, it is a sign that the simulation might be accurate. Visual inspection of dimensionality reduction projections is of course not enough to confirm the quality of the simulated data, and quantitative techniques are also required. Some dimensionality reduction techniques that can be used for such purpose include PCA, Linear Discriminat Analysis (LDA), Independent Component Analysis (ICA), Autoencoders (AE), ISOMAP [23], Parametric Uniform Manifold Approximation and Projection (UMAP) [24], and

many others. In this work we showcase the use of PCA, ISOMAP, and Parametric UMAP. When applied to genetic data, most of the previously mentioned techniques will generate clusters according to the population structure of the data which is commonly correlated with the geography.

G. Supervised Discriminators for Quantitative Evaluation

In this work, we explore the use of the accuracy of classifiers trained to distinguish between real and fake sequences as a quantitative measure of the quality of the simulation. Intuitively, if the simulation method is accurate, the classifiers should not be able to discriminate between real and fake samples and should obtain a classification close to 50% (random chance). In practice, many classifiers can memorize the training data obtaining a 100% classification accuracy within the training set. Therefore, we make use of the classification accuracy within a test set (sequences unseen during the training of the classifier) as an evaluation metric for the simulation method. Note that these classifiers have a similar role as the discriminator in GANs [25], but instead of providing a training gradient (as in GANs), they act as evaluation measures of the simulation quality. In fact, such classifiers can be used to evaluate the quality of any type of simulator, not necessarily based on Neural Networks.

We use Logistic Regression [26], K-Nearest Neighbors Classifier (KNN) [27] and Multi-Layer Perceptrons [28] for the evaluation task. We simulate the same numbers of synthetic sequences per ancestry as real sequences available. We randomly select 67% of both real and synthetic samples to comprise the training set and use the remainder as the testing set. We use five different random splits of the dataset and perform multiple times training and validation of the machine learning classifiers. Finally, we compute the average performance of each model and use the standard deviation to measure the variation in the accuracy results.

IV. EXPERIMENTAL RESULTS

We use a subset of the dataset presented in [29]. This dataset includes human labeled genomic sequences from several publicly available databases. Our dataset contains 5000 SNPs from chromosome 22 codified in a binary format for 1683 individuals. Specifically, it includes a total of 358 Europeans (EUR), 403 Africans (AFR), 486 East Asians (EAS), and 436 South Asians (SAS) individuals.

We train multiple GMMNs to explore (1) the effect of replacing the sigmoid by a differential quantizer (binarization layer), (2) of using ReLU features instead of the mean of the dataset (ϕ = Identity), and (3) of re-starting random features during training. In table I, we present the classification accuracy (real vs fake) of the 3 discriminators: logistic regression, k-NN, and MLP. Note that simple classifiers (logistic regression and k-NN) can be fooled easily, leading to a discriminative accuracy close to random chance (50%), while the MLP obtains a better discriminative accuracy. Because some classifiers discriminate better than the others, we report the worst case scenario as the classification accuracy furthest from random chance (50% accuracy) within the $\Delta = |0.5 - Acc|$ column. A setting where at least one of the classifiers can classify real vs fake with 100% accuracy, will lead to $\Delta = 0.5$, while a simulator that can fool all classifiers, will lead to $\Delta = 0.0$. Therefore, a good simulation technique will have a Δ close to 0, while a bad one will have a Δ close to 0.5.

As shown in the first two rows of table I, including a differentiable binarization step instead of a sigmoid layer, leads to a significant improvement in the simulation quality (Δ decreases from 0.49 to 0.25). This can be observed in figure 2, where the PCA of simulated data is shown. Subfigure b) and c) shows the simulated data after and before thresholding of the sigmoid layer and d) shows the simulated data using the quanitzer layer. The real and fake data can be visually discriminated if a sigmoid layer is used. The use of ReLU features (instead of an identity mapping) might not lead to improvements on the simulation accuracy as shown in the second and third row of table I. Due to the stochastic nature of random features, some instances of the random mapping might not capture a correct set of features. However, by including random re-starts (last row of table I) the simulation accuracy with random features is highly increased obtaining sequences that practically fool all the trained discriminators. Finally, in figure 3 we show visualizations of UMAP and ISOMAP using our best configuration of GMMNs (Quantizer + Random Features + RF re-start), and we show that no apparent differences between real and simulated sequences are present.

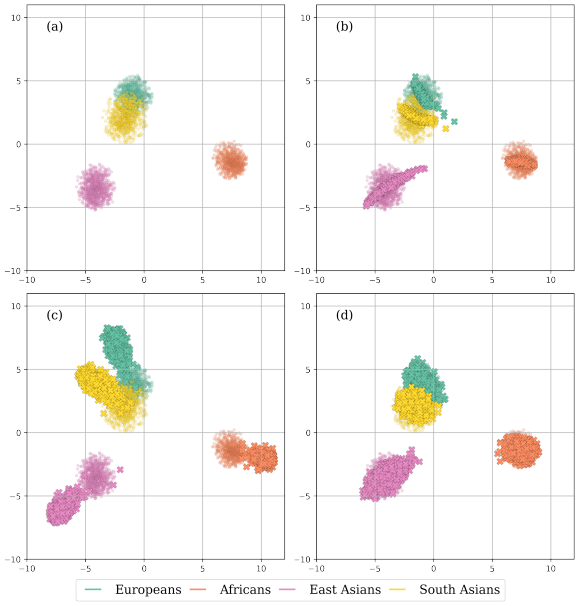

Fig. 2: PCA of: (a) real data and generated data (b) with sigmoid output, (c) with sigmoid + threshold output, and (d) with binary quantizer output. Real data PCA is plotted in the background in b, c, and d.

V. CONCLUSIONS

In this work, we showed that GMMNs are able to successfully generate new, realistic samples given a desired ancestry

Network Output	ϕ	RF Re-start	Logistic	KNN	MLP	$\Delta = \lvert 0.5 - Acc \rvert$
Sigmoid	Identity	-	0.99 ± 0.01	0.51 ± 0.01	0.99 ± 0.001	0.49
Quantizer	Identity	-	0.44 ± 0.009	0.57 ± 0.005	0.75 ± 0.001	0.25
Quantizer	ReLU features	✗	0.46 ± 0.01	0.52 ± 0.004	0.81 ± 0.03	0.31
Quantizer	ReLU features	✓	0.44 ± 0.007	0.54 ± 0.008	0.49 ± 0.05	**0.06**

TABLE I: Mean and standard deviation of accuracies of the supervised discriminators with different GMMN configurations.

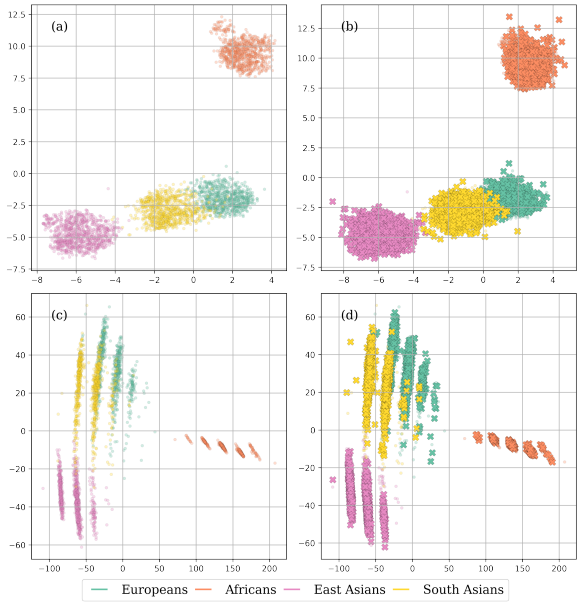

Fig. 3: UMAP of (a) real data and (b) generated data. ISOMAP of (c) real data, and (d) generated data.

and that simulation accuracy can be improved by using differentiable binarization and restarting the random features during training. Furthermore, we showed how evaluation of the simulated data can be performed with dimensionality reduction techniques and supervised classifiers. Finally, we discuss the potential of the presented framework to become a useful tool for privacy-preserving data sharing mechanism for modern day biobanks, and most importantly, a way to partially mitigate population bias within datasets.

REFERENCES

[1] Alice B Popejoy and Stephanie M Fullerton. Genomics is failing on diversity. *Nature News*, 538(7624):161, 2016.

[2] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727. PMLR, 2015.

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

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[5] Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5. Citeseer, 2007.

[6] Simon Gravel. Population Genetics Models of Local Ancestry. *Genetics*, 191(2):607–619, 06 2012.

[7] J.F.C. Kingman. The coalescent. *Stochastic Processes and their Applications*, 13(3):235–248, 1982.

[8] Richard R Hudson et al. Gene genealogies and the coalescent process. *Oxford surveys in evolutionary biology*, 7(1):44, 1990.

[9] Richard R. Hudson. Generating samples under a Wright–Fisher neutral model of genetic variation . *Bioinformatics*, 18(2):337–338, 02 2002.

[10] Jerome Kelleher, Alison M Etheridge, and Gilean McVean. Efficient coalescent simulation and genealogical analysis for large sample sizes. *PLOS Computational Biology*, 12(5):1–22, 05 2016.

[11] Kosuke Teshima and Hideki Innan. Mbs: Modifying hudson's ms software to generate samples of dna sequences with a biallelic site under selection. *BMC bioinformatics*, 10:166, 02 2009.

[12] Gregory Ewing and Joachim Hermisson. MSMS: a coalescent simulation program including recombination, demographic structure and selection at a single locus. *Bioinformatics*, 26(16):2064–2065, August 2010.

[13] Gary K Chen, Paul Marjoram, and Jeffrey D Wall. Fast and flexible simulation of dna sequence data. *Genome research*, 19(1):136–142, 2009.

[14] Daniel Mas Montserrat, Carlos Bustamante, and Alexander Ioannidis. Class-conditional vae-gan for local-ancestry simulation. *arXiv preprint arXiv:1911.13220*, 2019.

[15] 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):1–22, 02 2021.

[16] CJ Battey, Gabrielle C Coffing, and Andrew D Kern. Visualizing population structure with variational autoencoders. *G3*, 11(1):1–11, 2021.

[17] Margarita Geleta, Daniel Mas Montserrat, Carlos Bustamante, Xavier Giro-i Nieto, and Alexander Ioannidis. Deep variational autoencoders for population genetics. *biorxiv*, 2022.

[18] Margarita Geleta. Unsupervised learning with applications in genomics. B.S. thesis, Universitat Politècnica de Catalunya, 2021.

[19] Youngmin Cho. *Kernel methods for deep learning*. University of California, San Diego, 2012.

[20] Haotong Qin et al. Forward and backward information retention for accurate binary neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2250–2259, 2020.

[21] Antoine Chatalic, Vincent Schellekens, Florimond Houssiau, Yves-Alexandre de Montjoye, Laurent Jacques, and Rémi Gribonval. Compressive learning with privacy guarantees. *Information and Inference*, 2021.

[22] Frederik Harder, Kamil Adamczewski, and Mijung Park. Dp-merf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In *International Conference on Artificial Intelligence and Statistics*, pages 1819–1827. PMLR, 2021.

[23] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[24] Tim Sainburg, Leland McInnes, and Timothy Q Gentner. Parametric umap embeddings for representation and semisupervised learning. *Neural Computation*, 33(11):2881–2907, 2021.

[25] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[26] Christopher M. Bishop. Probabilistic Discriminative Models. In *Pattern Recognition and Machine Learning*, chapter 4.3, pages 205–210. Springer-Verlag, New York, 2005.

[27] T. Cover and P. Hart. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[28] David E. Rumelhart and James L. McClelland. *Learning Internal Representations by Error Propagation*, volume 2, pages 318–362. 1987.

[29] Albert Dominguez Mantes et al. Neural admixture: rapid population clustering with autoencoders. *bioRxiv*, 2021.