Comparing the Diversity and Similarity of Molecules Generated by GAN, VAE, Flow, and Diffusion Models Using SELFIES

Colten Phillips,[1,*] Isaac Cassulis,[2,†] Timothy Lund,[1,‡] and Wei Hu[1,§]

[1] *Department of Computer Science, Houghton College, Houghton, NY 14744, USA*
[2] *Department of Data Science, Houghton College, Houghton, NY 14744, USA*

(Dated: February 10, 2023)

Medical research and development is always an important issue, particularly as COVID-19 is prevalent throughout the world. One way to speed up development of medicines and vaccines is through molecular research, which is the aim of our study. We intended to compare and analyze the efficacy of several reinforcement learning approaches, using the Python programming language, to create new, valid molecules using SELFIES representations. SELFIES strings ensure valid molecular formats and are machine readable, which means SELFIES representations are ideal for machine learning. Using variational auto-encoders (VAE), flow-based generative models, generative adversarial models (GAN), and diffusion models, we compared their success in generating diverse molecules as well as the similarity of generated molecules. We found the diffusion model outperformed at generating dissimilar molecules, while the GAN model performed the best across all metrics.

Keywords: machine learning, neural networks, SELFIES, diffusion models, flows, variational autoencoders, generative adversarial networks, molecular diversity

I. INTRODUCTION

We are exploring the performances of four machine learning approaches to molecule generation using SELFIES. SELFIES is an improved version of SMILES, which is a way to encode molecular representations into strings in machine-readable format. While SMILES strings are complex and often lead to invalid results in machine learning, SELFIES strings minimize memory usage and always produce valid molecular structures [1]. Resources for understanding SELFIES can be found in [1] and [2]. To generate diverse molecules with SELFIES, we will be using the following four approaches: variational autoencoders (VAE), generative adversarial networks (GAN), flow-based models, and diffusion models. These four models are trained on the QM9 dataset, and their performance is compared to determine the best approach to molecular generation using SELFIES.

II. RESEARCH DESIGN AND METHODS

A. Variational Autoencoders

The autoencoder is an unsupervised neural network (NN) that compresses (reduces the dimensionality of) input data through an encoder, learns from the compressed data in the latent representation, and attempts to reproduce the input through decoding the latent space (decompression). Compression, then, allows a NN to learn the true size of the input, as if it is able to recreate data with an input size of 400 dimensions from a compressed size of 125 hidden layers, the true size of the data is much less than 400 [3]. This approach is useful as it allows models to remove excess information from input data, as well as generate new information from the input data, e.g., add color to a black and white image, increase the resolution of images, and remove unwanted blemishes [4].

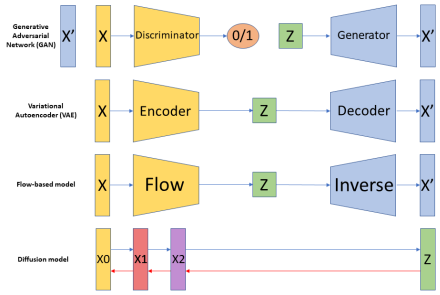

FIG. 1. Overview of the four models, inspired by [5]

The Variational Autoencoder (VAE) is one NN that allows us to generate data from the latent space, and instead of learning a particular estimate of a parameter, the model learns the probability distribution of the parameter [6]. (FIG. 1).

B. Flow Models

Normalizing flows use sequences of invertible mappings to transform a simple probability density into a complex one, and the term 'flow' comes from this sequence of invertible mappings (FIG. 1). The term 'normalizing' stems from the initial distribution being modified until we have a valid probability distribution [7]. If we take

* colten.phillips22@houghton.edu
† isaac.cassulis22@houghton.edu
‡ timothy.lund23@houghton.edu
§ wei.hu@houghton.edu

data z, we can calculate the density $p_Z(z)$ by inverting the transformation f with $\epsilon = f_\theta^{-1}(z)$ using the change-of-variables formula:

$$p_Z(z) = p_\varepsilon\left(f_\theta^{-1}(z)\right)\left|\det\frac{\partial f_\theta^{-1}(z)}{\partial z}\right| \text{[8]}.$$

Each new distribution is substituted for the old distribution, transforming and changing until we reach our target complexity. The transformation function f_θ should be invertible (reversable) and its Jacobian determinant should be easy to compute [9]. A helpful resource for understanding the Jacobian matrix and its determinant can be found here: [10]. The training criterion for normalizing flows is the negative log-likelihood $\log p(x)$ over our training data D:

$$L(D) = -\frac{1}{|D|}\sum_{x\epsilon D}\log p(x) \text{ [9]}.$$

C. Diffusion Models

Diffusion models train using a forward and backward process that preserves the dimensionality of the data (FIG. 1). In the forward process, a Markov Chain of diffusion steps is defined, which gradually adds random noise to the training data until it approximates an isotropic Gaussian distribution. In the backward process, the trained network aims to generate the desired original samples from Gaussian noise [5].

Given data x_0 sampled from the real data distribution $q(x)$, the forward process adds a small amount of Gaussian noise in T steps, producing noisy samples $x_1, \ldots, x_T$. These steps are controlled by the variance schedule $\{\beta_t \in (0,1)\}_{t=1}^T$ [5].

We can sample step x_t given the previous step x_{t-1} by the conditional probability

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}).$$

Reparameterizing so that x_t is conditioned on x_0 rather than x_{t-1} allows us to sample x_t at any arbitrary time step:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{a}_t}x_0, (1-\bar{a}_t\mathbf{I})),$$

where

$$\bar{a}_t = \prod_{i=1}^{T} a_i,\ a_t = 1-\beta_t.$$

The goal of the reverse process is to sample from $q(x_{t-1}|x_t)$, where we take $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. We define a model p defined by parameters θ to estimate these conditional probabilities, such that

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t,t), \Sigma_\theta(x_t,t)).$$

where μ_θ and Σ_θ are parameterized as deep neural networks that predict the mean and covariance, respectively [5].

We use variational lower bound to calculate the loss between our model estimate $p_\theta(x_{t-1}|x_t)$ and the true reverse conditional probability $q(x_{t-1}|x_t)$ [5].

D. Generative Adversarial Networks

Generative adversarial networks (GANs) utilize an actor-critic architecture with two models - the generator and discriminator (FIG. 1). The generator G functions as the actor, learning the real data distribution to generate synthetic samples given some noise variable input z. The discriminator D acts as the critic, and estimates the probability of a given sample coming from the real dataset. The two models play a zero-sum game, where the generator is optimized to trick the discriminator and the discriminator is optimized to distinguish fake samples from the real ones [11].

We define the loss function

$$L(G,D) = \int_x p_r(x)\log\left(D(x)\right) + p_g(x)\log\left(1-D(x)\right)dx$$

where p_r is the distribution over real data x, and p_g is the generator's learned distribution over x. G is optimized when p_g approaches p_r [11].

III. RESULTS

The models used by Krenn et al. (2020) displayed molecular diversity as the number of unique molecules divided by the number of samples from the dataset [2]. We looked to improve their measurements of molecular diversity by adding fingerprinting to our models. A molecular fingerprint is a collection of structural information about a molecule that is encoded in bit strings. Using RDKit, which uses a Daylight-like fingerprint based on hashing molecular subgraphs, we can compare the Tanimoto similarity of generated molecules' fingerprints. The Tanimoto similarity metric was chosen as it performs favorably compared with several other similarity metrics [12], [13], and was declared to be the best similarity coefficient when the size of molecules is unknown [14]. The Tanimoto coefficient ranges from 0 to 1 and measures the number of common bits; the closer to 0 the number is, the fewer bits the molecules have in common, and the closer to 1 the number is, the more bits the molecules have in common. Validity is the measurement of whether or not generated molecules are syntactically valid. Our measurement of reconstruction measures the ability of the models, given a particular molecule, to reconstruct the original molecule from the latent space. All our models sampled from the $0SelectedSMILES_QM9$ dataset [2]. Each model ran for 1000 episodes with a sample size of 1000, and used SELFIES encoding. Where applicable,

our learning rate was always 0.0001. The architectures of the VAE and GAN models are identical to the models used in [2].

We worked with four different flows, with varying degrees of success, modifying the flows found in [15]. We adjusted the prior distribution to be the length of the maximum size one of the molecules could be in a one hot encoded state which is 378. We also adjusted the flow to have only 1 flow model of size 378×378. Of the four flows, only two produced reliable results: the Affine Half Flow and the Invertible Flow. The other two, the Inverse Autoregressive Flow and the Affine Constant Flow produced unreliable results as discussed later. For our diffusion model, we used the architecture found in [16]. We adjusted the three layers in the network to be 378×1492, 1492×1492, and 1492×756 respectively.

IV. CONCLUSION

Every model except the VAE performed at 100% in our validity metric, which is expected since none of the other models feature any form of compression (FIG. 4). The VAE still performed respectably, with an average near 81%. The GAN performed best in our diversity metric with a score near 58%, and the Invertible flow was close behind, around 50% (FIG. 2). The diffusion model performed very poorly, with an average diversity of under 3%. After 1000 episodes, all models except diffusion had a Tanimoto similarity between 0.051 and 0.055, while the diffusion scored much closer to 0 (FIG. 3). Remember that the lower the Tanimoto similarity, the fewer bits molecules have in common. Lastly, the VAE model performed reconstruction nearly 82% of the time, while the Invertible flow was much lower, around 7.5% (FIG. 5).

From our results, the diffusion model is by far the best at generating dissimilar molecules; however, it generates significantly fewer unique molecules than the other models. The GAN model had the highest diversity score, generating the most unique molecules and had the second lowest Tanimoto similarity; thus, the GAN is the best model for producing unique, diverse molecules. See Table I for a breakdown of the final values for all models. However, there were flows that performed better, but had issues discussed below.

V. FUTURE WORK

Both the Affine Constant Flow and the Inverse Autoregressive Flow returned unreliable results due to the models returning invalid outputs; we were unable to determine exactly why, given time restraints. However, as the below graphs show, both these flow models performed very well prior to the issues occurring (FIG. 7, FIG. 8). For future work, our first and obvious step would be to investigate the causes for these issues, and attempt solutions. With no prior indication, the models seem to fail at arbitrary episodes during training. Some initial research indicates that this may be an issue with the gradient, but we were unable to come to any conclusions. These two flows scored very high in diversity and reconstruction, and seem very promising if the issues can be sorted out (FIG. 6, FIG. 9).

[1] A. Aspuru-Guzik, "Molecular graph representations andselfies: A 100 percent robust molecular string representation," (2021).
[2] M. Krenn, F. H. A. Nigam, P. Friederich, and A. Aspuru-Guzik, "Selfies," (2020).
[3] P. Foy, "Generative modeling: What is a variational autoencoder (vae)?" (2021).
[4] R. Chandradevan, "Autoencoders are essential in deep neural nets," (2017).
[5] L. Weng, lilianweng.github.io/lil-log (2021).
[6] R. Neo, "Beginner guide to variational autoencoders (vae) with pytorch lightning," (2021).
[7] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," (2016), arXiv:1505.05770 [stat.ML].
[8] C. Shi, M. Xu, Z. Zhu, W. Zhang, M. Zhang, and J. Tang, CoRR **abs/2001.09382** (2020), 2001.09382.
[9] L. Weng, lilianweng.github.io/lil-log (2018).
[10] S. Cristina, "A gentle introduction to the jacobian," (2021).
[11] L. Weng, lilianweng.github.io/lil-log (2017).
[12] A. Bender, J. L. Jenkins, J. Scheiber, S. C. K. Sukuru, M. Glick, and J. W. Davies, Journal of Chemical Information and Modeling **49**, 108 (2009).
[13] D. Bajusz, A. Rácz, and K. Héberger, Journal of Cheminformatics **7** (2015), 10.1186/s13321-015-0069-3.
[14] P. Willett, Drug Discovery Today **11**, 1046 (2006).
[15] A. Karpathy, "Normalizing flows," (2019).
[16] ACIDS, "Denoising diffusion probabilistic models," (2021).

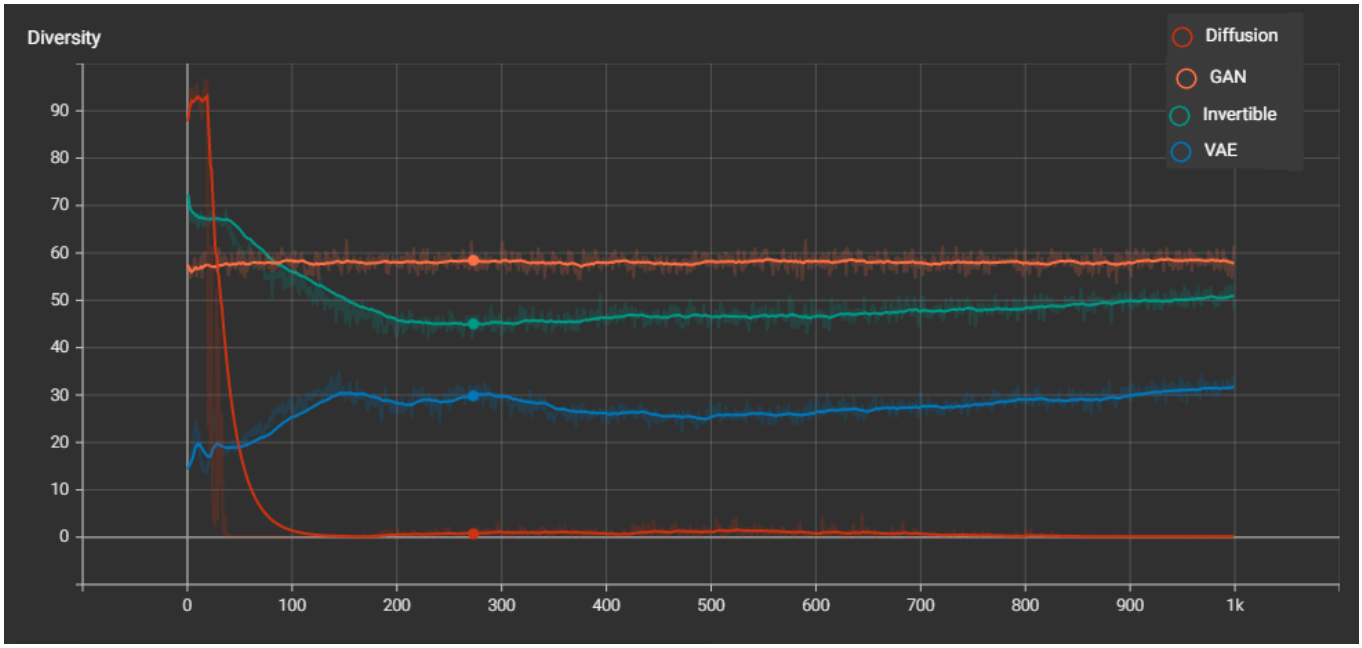

FIG. 2. Diversity Measurements For All Models (Smoothed)

FIG. 3. Tanimoto Similarity For All Models (Smoothed)

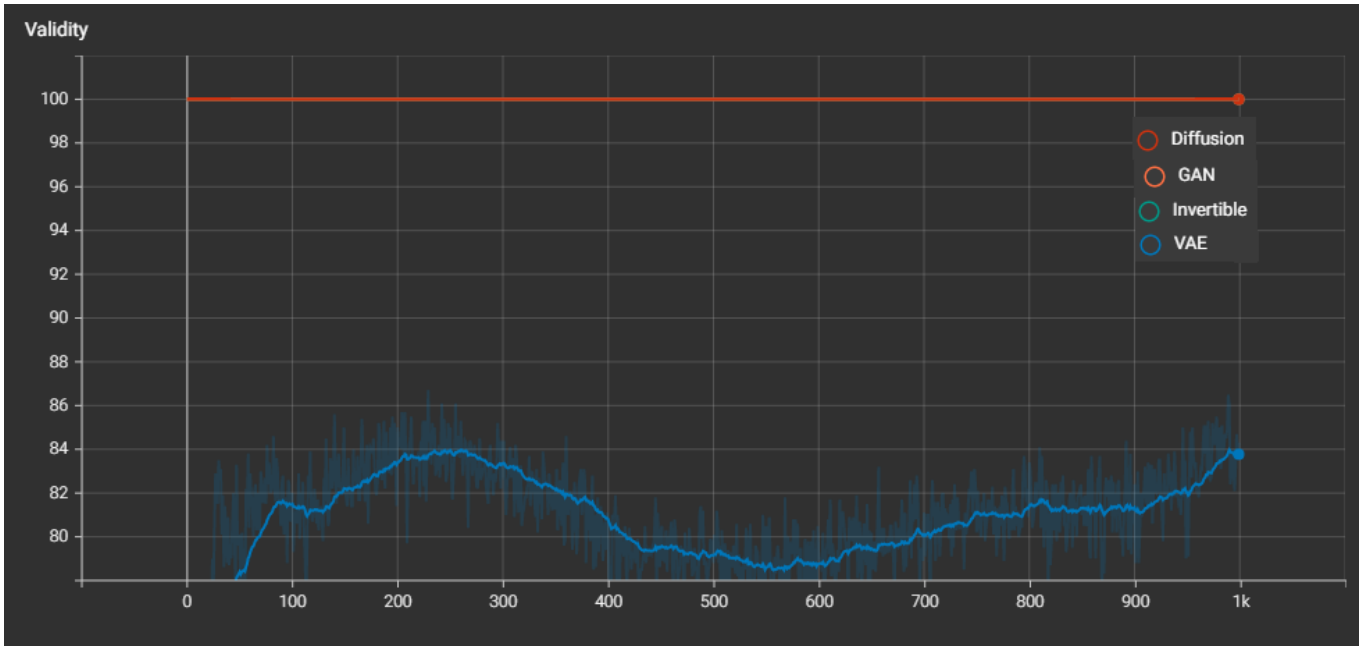

FIG. 4. Validity Measurements For All Models (Smoothed)

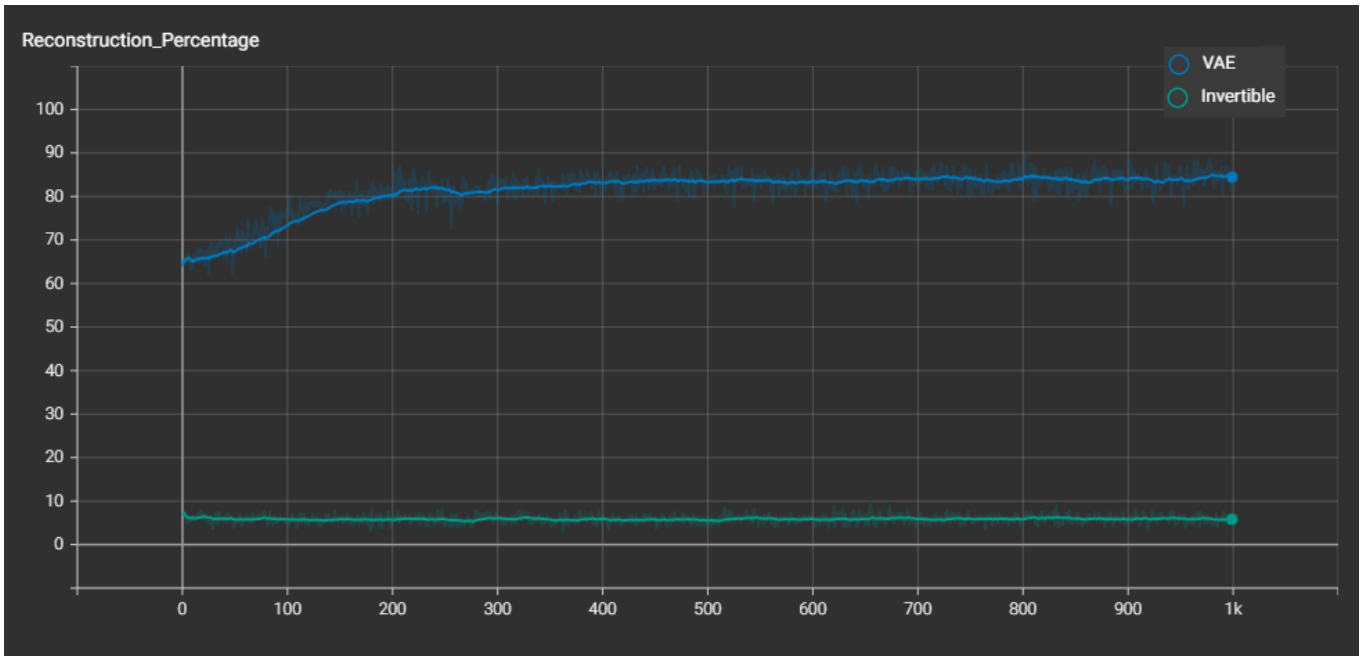

FIG. 5. Reconstruction Rates For the VAE and Flow (Smoothed)

Model	Final Validity	Final Diversity	Final Tanimoto	Final Reconstruction
VAE	84.3000%	31.2000%	0.0545	81.1400%
GAN	100.0%	61.5000%	0.0513	N/A
Flow	100.0%	47.9000%	0.0551	7.4290%
Diffusion	100.0%	0.1000%	0.0000	N/A

TABLE I. Final Values of Each Model After 1000 Episodes

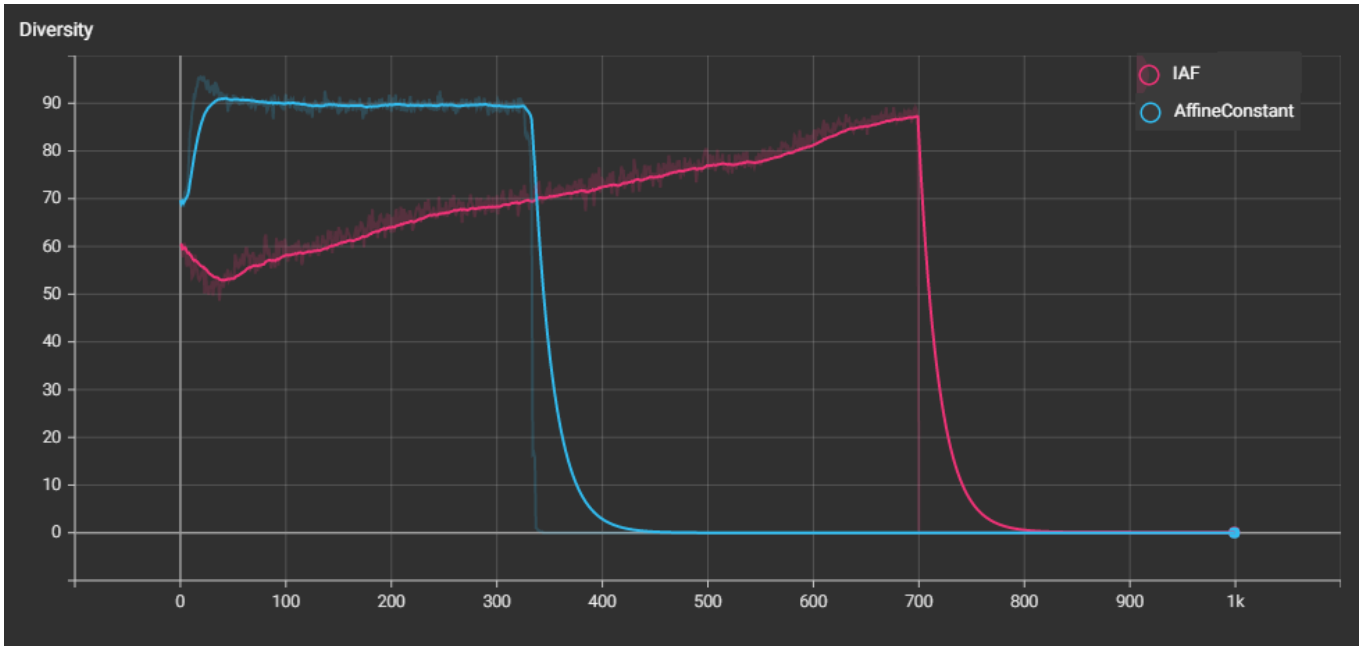

FIG. 6. Diversity Measurements For the IAF and Affine Constant Flow (Smoothed)

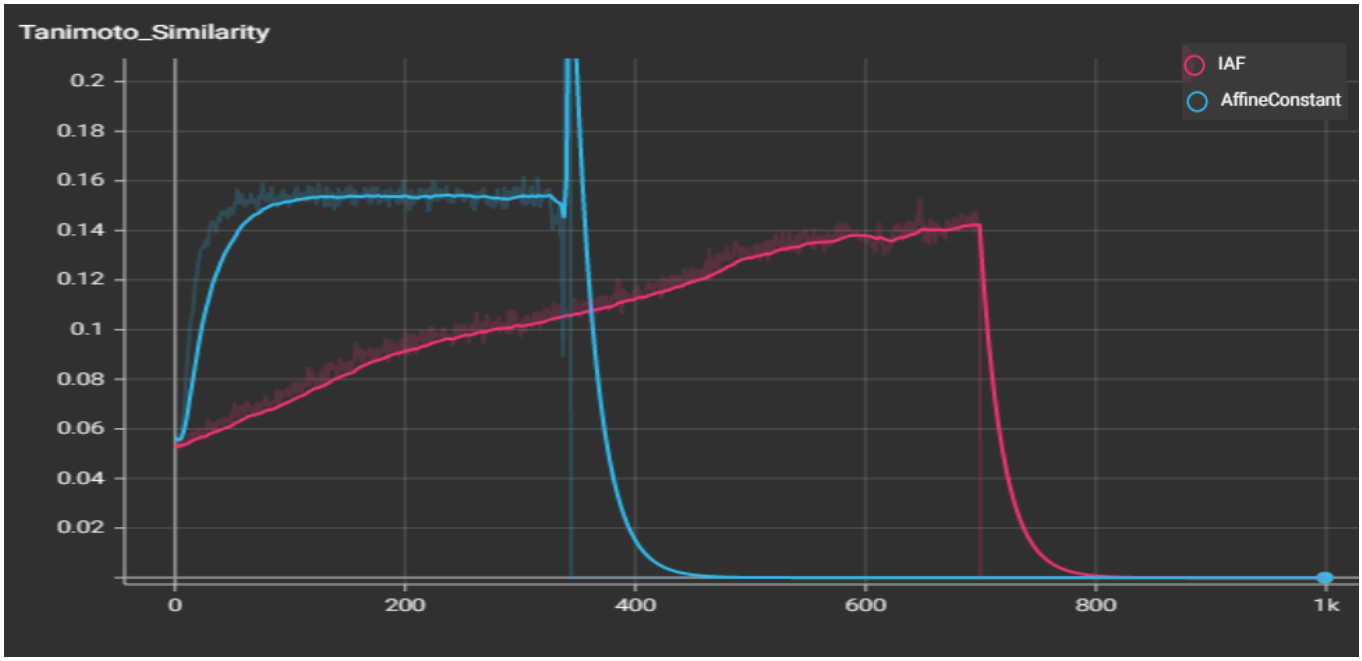

FIG. 7. Tanimoto Similarity For the IAF and Affine Constant Flow (Smoothed)

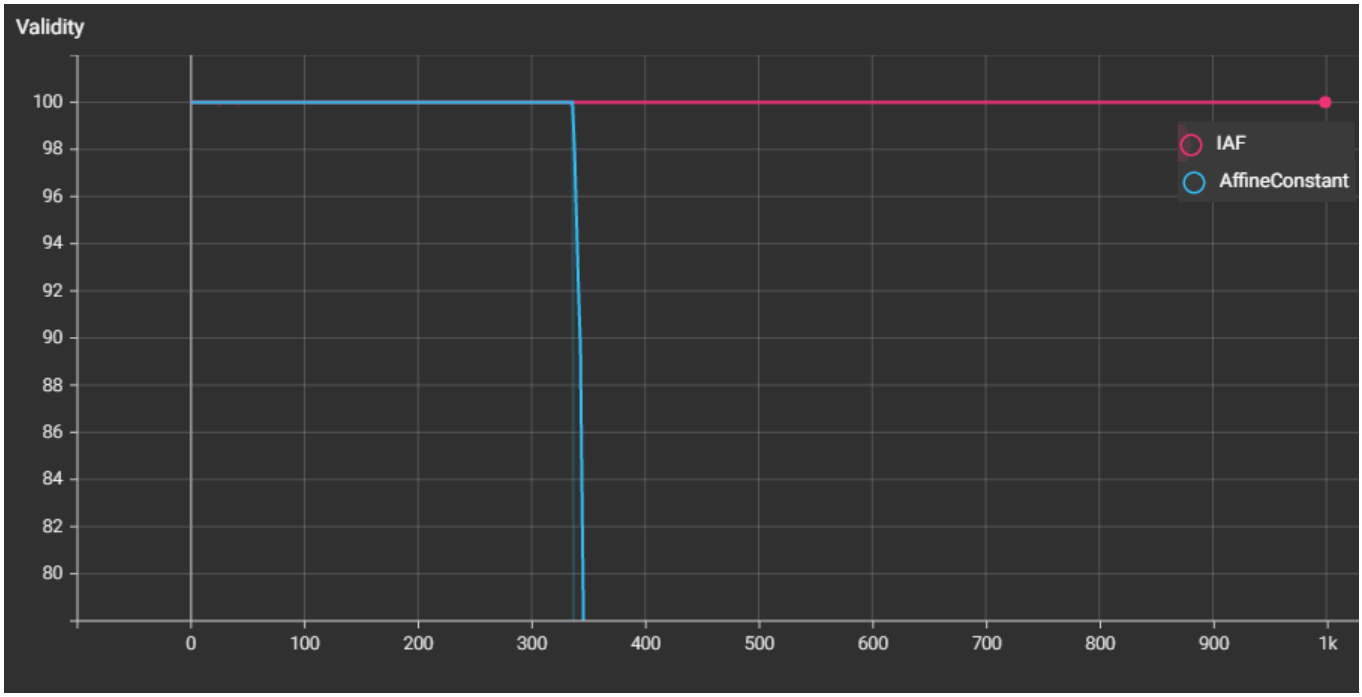

FIG. 8. Validity Measurements For the IAF and Affine Constant Flow (Smoothed)

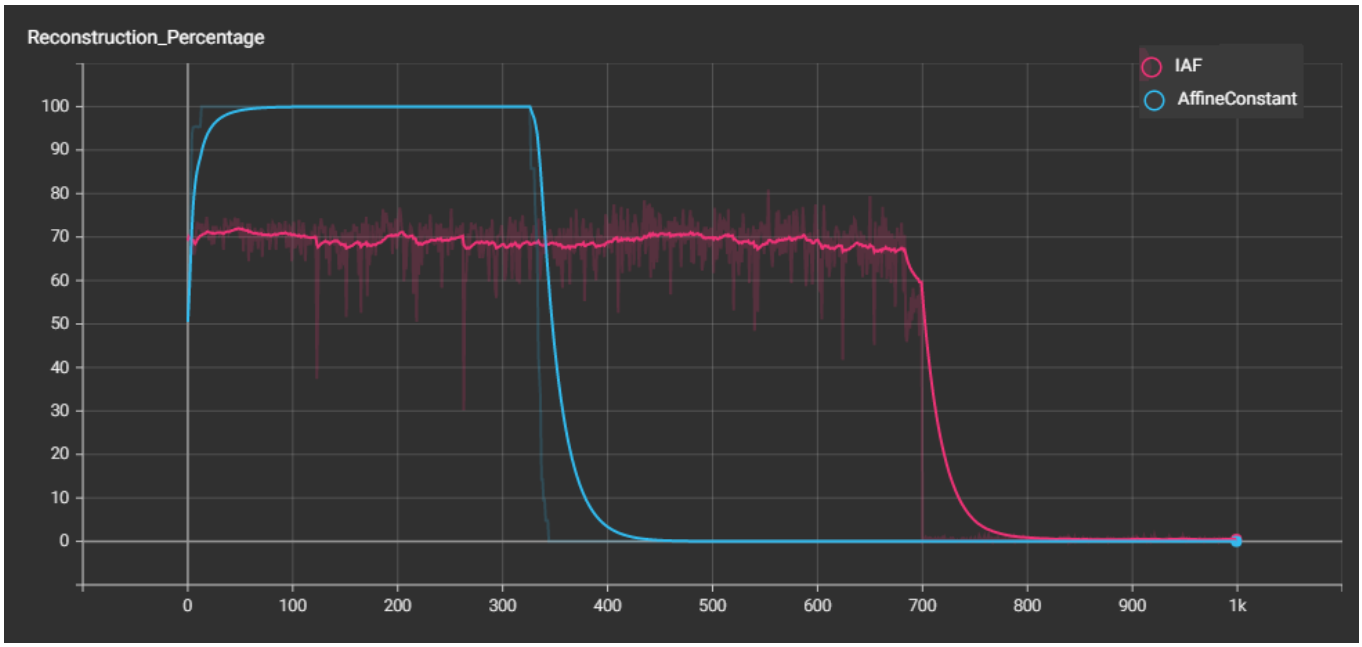

FIG. 9. Reconstruction Rates of the IAF and Affine Constant Flow (Smoothed)