Investigating the performance of deep learning methods for Hi-C resolution improvement

Ghulam Murtaza[1],
Atishay Jain[1], Madeline Hughes[1], Thulasi Varatharajan[2], and Ritambhara Singh[1,3,*]

[1]*Department of Computer Science, Brown University*
[3]*Center for Computational Molecular Biology, Brown University*
[2]*Department of Biology, Dietrich School of Arts and Sciences at University of Pittsburgh*
[*]*Corresponding Author*

Abstract

Motivation: Hi-C is a widely used technique to study the 3D organization of the genome. Due to its high sequencing cost, most of the generated datasets are of coarse quality, consequently limiting the quality of the downstream analyses. Recently, multiple deep learning-based methods have been proposed to improve the quality of these data sets by increasing their resolution through upscaling. However, the existing works do not thoroughly evaluate these methods using Hi-C reproducibility metrics across different Hi-C experiments to establish their generalizability to real-world scenarios. This study extensively compares deep learning-based Hi-C upscaling methods on real-world, Hi-C datasets with lower read counts. We evaluate these models using Hi-C reproducibility metrics and downstream evaluation on data from three cell lines – GM12878 (lymphoblastoid), K562 (human erythroleukemic), and IMR90 (lung fibroblast) – obtained from different Hi-C experiments.

Results: We show that the existing deep-learning-based Hi-C upscaling techniques are inadequate for effectively upscaling Hi-C data with low read counts. We observe that retraining these models with real-world datasets substantially improves the performance of these methods, both Hi-C similarity metrics, and downstream tasks. However, we find that Hi-C upscaling methods struggle to capture the underlying biological signals on sparse datasets even when retrained with real-world datasets. Therefore, our study highlights the need for rigorous evaluation and identifies specific improvement areas concerning current deep learning-based Hi-C upscaling methods. We facilitate our thorough analysis by developing a computational pipeline that provides a unified platform for pre-processing, upscaling, and evaluating the generated Hi-C data quality for for users interested in benchmarking or using these methods.

Availability: `https://github.com/rsinghlab/Investigation-of-HiC-Resolution-Improvement-Methods`

1 Author Summary

This paper comprehensively evaluates Hi-C deep-learning-based upscaling methods using seven datasets across four similarity metrics and two downstream evaluation methods. To facilitate this study and upcoming methods, we developed a pipeline that provides a unified interface that abstracts underlying Hi-C data representations to accelerate the development and evaluation processes. Our work finds that the existing methods trained with synthetically generated datasets tend to perform significantly worse on the datasets we have curated from real-world experiments. Moreover, in our work, we find that retraining the existing methods improves the performance of these methods on real-world datasets. Finally, even though we see significant improvements in specific metrics, we also observe comparable performance in downstream methods implying that using a single class of metrics is inadequate for estimating the biological usefulness of enhanced Hi-C datasets.

2 Introduction

3D organization of the genome plays a vital role in cell fate and disease onset. A high-throughput chromosome conformation capture experiment, or Hi-C, is a genome-wide sequencing technique that allows researchers to understand and study the 3D organization of the genome [9]. Each read pair, sequenced using Hi-C, corresponds to an observed 3D contact between two genomic loci. This contact information captures both local and global interactions of the genomic regions. In the past decade, analysis of Hi-C data has facilitated many studies that have led to the discovery of important genomic structural features, including but not limited to A/B compartments [9] that denote active and inactive genomic regions, topologically associated domains (TAD) [2] that represent highly interactive genomic regions, and enhancer-promoter interactions [14] that are involved in the regulation of genes. Therefore, Hi-C experiments have proven to be crucial in advancing our understanding of the spatial structure of the genome and its relationship with gene regulation machinery.

When studying the spatial structure of the DNA, the quality of the downstream analysis of a Hi-C experiment is highly dependent on the quality of its contact map. Data from a Hi-C experiment coalesces into a matrix (or contact map) in which rows and columns correspond to fixed-width windows ("bins") tiled along the genomic axis, and values in the matrix are counts of read pairs that fall into the corresponding bins. The bin size typically ranges from 1 kilo basepair (kbp) to 1 mega basepair (mbp), where the choice of the bin size depends on the number of paired end-reads or the quality of the experiment. Low-quality experiments result in sparser read counts that require large bin sizes to account for the sparsity, resulting in a "low-resolution" contact map. Consequently, the downstream analysis of the generated matrix does not yield fine genomic features like enhancer-promoter interactions that typically occur in the 5 kbp to 10 kbp range [6]. Similarly, the output of a high-quality Hi-C experiment with a high number of read counts results in a "high-resolution" contact map with small bin sizes. This map enables the identification of the fine-grained genomic features. However, due to the quadratic scaling of the sequencing cost, most of the Hi-C datasets have relatively low read counts and, consequently, low-resolution ($\geq$ 40 kb) contact maps [17]. Constructing Hi-C matrices with high resolution, possibly requiring billions of reads [6], is often infeasible. The absence of such matrices makes the comprehensive analysis of the spatial structure of the DNA difficult.

Recently, researchers have proposed several computational methods to upscale [1] the Hi-C matrices by imputing the missing contacts. Notably, deep-learning-based methods have had remarkable success in upscaling Hi-C matrices. HiCPlus [17], and HiCNN [11] employed convolutional layers with mean squared error loss to optimize the model weights to generate an upscaled Hi-C matrix conditional on downsampled low-quality input matrices. HiCGAN [10], DeepHiC [5], and VeHiCle [4] used GANs (Generative Adversarial Networks) to optimize the discriminator loss, guiding the model to produce Hi-C matrices with sharper and more realistic features in comparison to HiCPlus and HiCNN. These methods (except VeHiCle) report improved performance on synthetically downsampled Hi-C datasets using image-based evaluation metrics like correlation. However, neither synthetic datasets nor correlation metrics capture the underlying properties of the real-world Hi-C datasets because they make broad simplifying assumptions about the real-world data. Therefore, these results cannot provide insights into the generalizability and performance benefits of using these deep learning methods for the real-world Hi-C upscaling task. Furthermore, while VeHiCle does utilize real-world datasets for training and evaluation, it restricts its analysis to four samples (across four cell lines) originating from the same experiment. As a result, it does not assess the performance benefits on the datasets with varying levels of sparsity originating from different experiments. Consequently, it is unclear if these deep-learning-based methods trained on a subset of Hi-C datasets can generalize to datasets potentially belonging to different cell lines, sparsity levels, and experiments. Therefore, it is critical to thoroughly evaluate these techniques using Hi-C-specific similarity metrics across datasets collected from various sources to unveil the underlying trade-offs of using them in real-world scenarios.

[1] Synonymous with improving Hi-C quality, which subsequently increases the Hi-C resolution

In this study, we design an extensive set of experiments that compare the performance of deep learning-based Hi-C upscaling methods on real-world low-quality Hi-C datasets. We pipeline that abstracts away the process of handling inputs to various Hi-C pre-processing, upscaling and evaluation platforms to accelerate both analysis and benchmarking of Hi-C related toolkits. This pipeline contains seven datasets spanning three cell lines (GM12878 (lymphoblastoid), K562 (human erythroleukemic), and IMR90 (lung fibroblast)) procured from various Hi-C experiments and sources, six Hi-C upscaling methods (Gaussian Smoothing, HiCPlus, HiCNN, HiCNN2, DeepHiC and VeHiCLe) and twelve evaluation metrics divided into three categories (correlation-based metrics, Hi-C similarity metrics, and downstream evaluation metrics). We find that the performance of all deep learning models deteriorates when upscaling real-world low-read-count matrices compared to equally sparse downsampled Hi-C map inputs. For example, our results show that HiCNN (the best generalizing model) on a real-world low quality (datasets with low read counts) dataset with $\frac{1}{50}$ sparsity shows a 41% reduction in one of the Hi-C similarity metric in comparison to on a synthetically downsampled data with $\frac{1}{50}$ sparsity (relative to high quality dataset [6]). To alleviate this disparity, we retrain the best performing deep learning model with various input augmentation strategies, such as adding noise to downsampled Hi-C data, ensembling downsampled inputs, and retraining with real-world low-quality input datasets. We find that training with a real-world low-quality dataset substantially improves the performance of the deep learning model. Specifically, our HiCNN retrained with a real-world dataset shows an improvement of 40% on a $\frac{1}{50}$ low-read count dataset over the baseline HiCNN.

Additionally, we perform two downstream analyses - (1) recreating the 3D chromatin structure from upscaled Hi-C matrices and (2) comparing Topologically associating domain (TAD) recovery profiles using Insulation vectors in the upscaled Hi-C matrices. We observe a similar trend for 3D chromatin structure reconstruction, where we recover more biological information through methods trained with real-world low-quality datasets. Although methods trained with downsampled datasets perform better in our TAD recovery analysis, our retrained methods still show comparable performance in all datasets. Moreover, none of the existing techniques improve the TAD recovery over the baseline for our sparsest dataset, suggesting potential overfitting towards datasets with higher read counts. Lastly, we conclude that there exists scope for further improvements in training strategies, model architecture, and evaluation protocols to ensure the deep learning models' generalizability to real-world Hi-C upscaling tasks.

3 Methods

3.1 Data pre-processing

We collected the Hi-C datasets for our study from multiple experimental sources. Following the existing deep-learning studies [4, 5, 10, 11, 17], we used Hi-C matrices for the GM12878, IMR90, and K562 cell lines from Rao *et al.* [6] as our primary High Read Count (HRC) datasets. We define HRC datasets as Hi-C experiments that are high-quality and formulated as high-resolution matrices. We selected seven additional low-quality, low-resolution, or Low Read Count (LRC) Hi-C experiments to test the performance of deep learning models on the real-world Hi-C upscaling task. Five of these Hi-C experiments are for GM12878 cell line (two from ENCODE experiment ID ENCSR382RFU, one file from ENCODE experiment ID ENCSR968KAY and remaining two from GEO accession code GSM1551582 and GSM1551575), one for K562 cell line (GEO accession code GSM1551622), one for IMR90 cell line (GEO accession code GSM1551606). Note that the GEO accession datasets are experimental replicates of the original Rao *et al.* dataset and represent a subset of the HRC Hi-C matrix. Table 1 summarizes the attributes of each dataset. We chose the bin size of 10kb resolution, and MAPQ value of $>= 30$ with KR normalization [8] to balance the Hi-C matrices and filtered out reads with low statistical confidence to remove spurious reads and experimental artifacts. We further filtered out inter-chromosomal, and X chromosome reads as done in previous works. Therefore, we have twenty-two

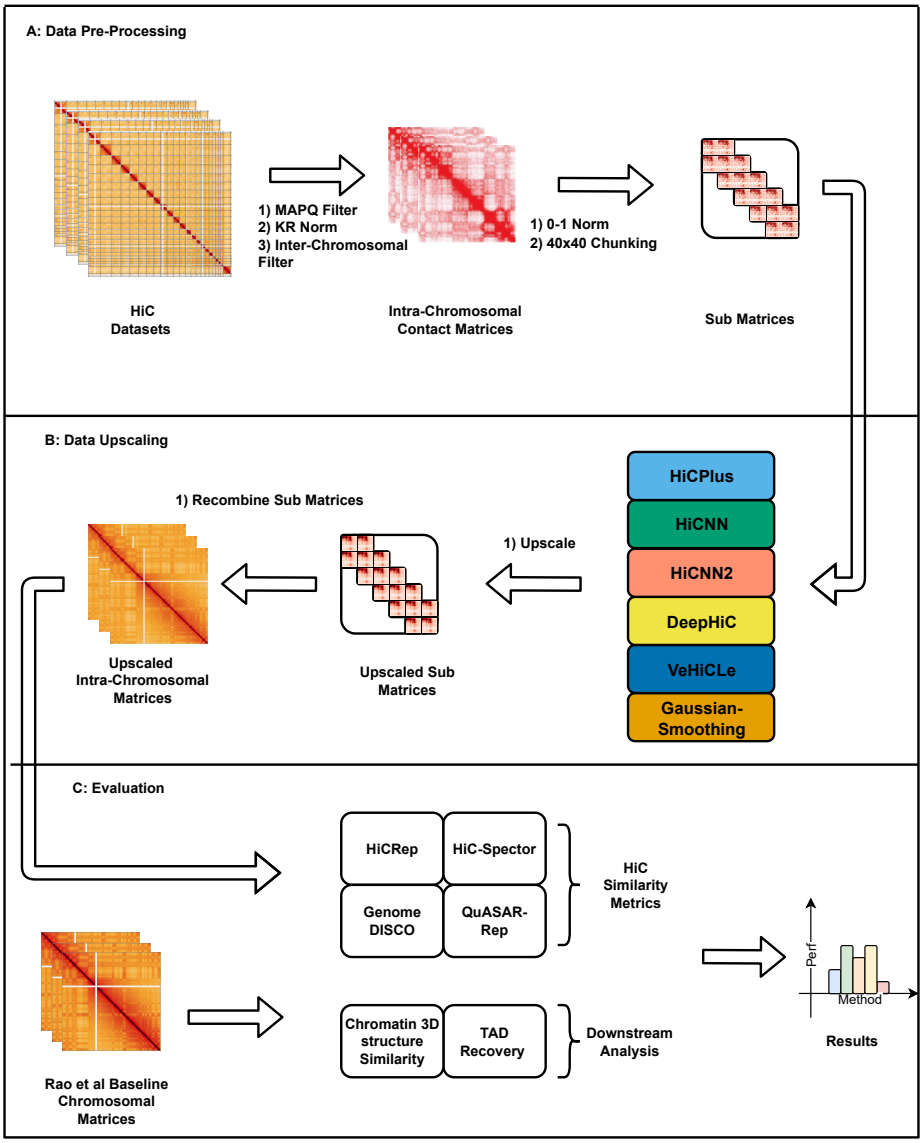

Figure 1: **Overview of our benchmarking study: A** Data pre-processing pipeline - We filtered the Hi-C matrices with the same MAPQ value ($\geq$30) and normalized them with the same KR normalization algorithm to ensure a fair comparison. We removed inter-chromosomal contacts because of their extremely sparse nature and performed a 0-1 normalization on intra-chromosomal matrices to reduce the impact of extreme values. Finally, we cropped appropriately sized sub-matrices to ensure that input is in the correct format for each upscaling algorithm. **B** We upscaled the sub-matrices using a wide variety of deep learning-based upscaling models and then recombined them to form upscaled intra-chromosomal Hi-C matrices. **C** We used multiple Hi-C-based similarity metrics for evaluation of the upscaling methods as well as downstream analyses to provide a comprehensive set of results that we can use to analyze the performance of each upscaling model.

files containing intra-chromosomal contact values in a dense matrix format after pre-processing each dataset.

We performed additional pre-processing specifically for the existing deep-learning-based models. These models need the input matrices to be cropped and be of a particular size (either 200x200 or 40x40). Moreover, if the input, in addition to KR normalization, is also normalized to have values between 0 and 1, performance increases ([5]). Therefore, we pre-processed all the Hi-C matrices, chromosome-wise, by clamping all the values between 0 and the 99.95^{th} percentile value of their read counts. Then we divided all the read count values by 99.95^{th} percentile values to standardize the matrices. Next, we cropped these matrices into sub-matrices that

Dataset	Absolute Read Counts	Relative Read Counts	Source
GM12878-HRC-1	1,844,107,778	1	GEO (GSE63525)
GM12878-LRC-1	42,453,795	1/44	ENCODE (ENCSR382RFU)
GM12878-LRC-2	37,079,587	1/50	ENCODE (ENCSR382RFU)
GM12878-LRC-3	70,138,184	1/25	ENCODE (ENCSR968KAY)
GM12878-LRC-4	202,380,884	1/9	GEO (GSM1551575)
GM12878-LRC-5	18,696,952	1/100	GEO (GSM1551582)
IMR90-HRC-1	735,043,093	1	GEO (GSE63525)
IMR90-LRC-1	75,193,876	1/10	GEO (GSM1551606)
K562-HRC-1	641,402,880	1	GEO (GSE63525)
K562-LRC-1	44,882,605	1/14	GEO (GSM1551622)

Table 1: **Summary of the datasets and their sources.** All experiments used Mbol enzyme and filtered fragments to be in the size range of 300-500 using SPRI beads.

deep-learning-based models can finally use to upscale. Like previous works, we obtained these sub-matrices from the 200 bin range around the diagonal of the Hi-C matrix, which contains most of the biological features of the Hi-C matrices [17]. We used the standard dataset splits used by the existing works. The test set contains chromosomes 19-22, the validation set contains chromosomes 9-12, and the remaining chromosomes are in the training set. We exclusively trained (or retrained) all models with GM12878 cell line datasets and prediction on K562 and IMR90 represents the cross-celltype predictions.

3.2 Deep learning-based Hi-C upscaling models

We ran several state-of-the-art deep learning-based Hi-C upscaling methods to provide comprehensive performance results. We use the PyTorch implementations of these methods provided by the papers to ensure a fair comparison of the off-the-shelf version of each technique. Our selected deep-learning models can be divided into two broad categories, those that employed adversarial loss to optimize the weights and those that did not. HiCPlus [17] uses a three-layer convolutional network to optimize with MSE loss. HiCNN [11] extended the HiCPlus to a 54-layer architecture that had residual network (ResNet) layers [3] which resulted in a stable model with faster training. We also added HiCNN2 [12] that builds up on HiCNN [11] by adding global residual learning by concatenating hierarchical features from each residual block. HiCNN2 also added a modified version of VSDR [7] and HiCPlus to finally generate an output based on all of the three models as mentioned above. HiCGAN [10], DeepHiC [5] and VeHiCLe [4] employed Generative Adversarial Networks (GANs). GANs jointly optimize two models, a generator that produces inputs and a discriminator that tells fake inputs from real ones, to learn to create increasingly more realistic outputs. HiCGAN used the MSE loss and the discriminator loss to optimize the weights. DeepHiC made extensions over HiCGAN to introduce perceptual loss and total variation loss to force the models to generate outputs with sharper and more realistic features. Lastly, VeHiCle made two further modifications to the DeepHiC. It replaced the perceptual loss with a domain-specific Hi-C loss by using an unsupervised model trained to generate Hi-C inputs. 2) It added an insulation loss that forces the model to learn the underlying biological structure (specifically topologically associated domains) to generate biologically informative Hi-C matrices.

For our evaluations, we retrained HiCPlus, HiCNN, and HiCNN2 because all of these models predicted raw contact counts rather than a contact map with values between [0, 1], which made these models' outputs incomparable and also possibly at a disadvantage against the other models [5]. Moreover, DeepHiC used model ensembling (a model for 1/16, 1/25, 1/50, and 1/100 downsampling ratio inputs) to get performance across the entire range of Hi-C input sparsities. To make the comparison fair, we trained an additional three versions of

HiCNN and HiCPlus suited to upscale inputs with 1/25, 1/50, and 1/100 to ensure a fair comparison at inputs from multiple downsampling ratios. Note that we exclude HiCGAN [10], a Generative Adversarial Network (GAN) model, from our comparisons as we were unable to set up the provided code base because it depended on deprecated python packages. For DeepHiC and VeHiCle, we used the provided weights. As a baseline upscaling algorithm, we added Gaussian smoothing with kernel size of $n = 17$ and 2D Gaussian distribution with $\sigma_x = \sigma_y = 7$. Lastly, we did not perform hyper-parameter tuning for model retraining to compare the available versions of these methods. For additional information on these methods, please refer to Supplementary Section S1.

3.3 Evaluation metrics

Recent work has shown that correlation metrics (such as Pearsons and Spearmans) fail to assign an accurate reproducibility score for Hi-C experiments due to their inability to account for the underlying data distributions (e.g., distance effect in Hi-C matrices) [16]. Most current Hi-C upscaling studies have used correlation metrics such as Pearsons, Spearman's, and Structural similarity index metric (SSIM) to evaluate their performance. Due to the limitations of the correlation-based metrics, we conduct our analysis primarily based on Hi-C similarity metrics. To evaluate the state-of-the-art deep learning models, we use all four Hi-C specific similarity metrics (GenomeDISCO, HiCRep, Hi-C-Spector, and QuASAR-Rep) designed to calculate the Hi-C contact map reproducibility scores. Each metric compares the "ground truth" original high-resolution Hi-C map and the upscaled Hi-C map generated from the deep learning methods to provide a similarity score. To conserve space and keep the evaluation concise, we only show results on SSIM and GenomeDISCO in the main text, and results for the rest of the metrics are in the supplementary section. SSIM uses two images' luminance, contrast, and structure to calculate the similarity. GenomeDISCO uses Graph random walks to smooth out the contact matrices at varying scales to compute concordance scores between the two input maps. For detailed information on these metrics (and other metrics), refer to Supplementary Section S2.

3.4 Downstream analysis strategies

We perform two downstream analyses to assess the quality of biological information that we can recover from the upscaled Hi-C matrices generated by the deep learning methods from LRC matrices. Although similarity scores produced by QuASAR-Rep, GenomeDISCO, HiC-Rep, and HiC-Spector, are at their core, driven by underlying biology, they never directly compare the biological content obtained from the Hi-C analysis. Therefore, we set up two downstream evaluation techniques to ensure a comprehensive evaluation. First, we use 3DMax [13] to generate a 3D model of chromatin from the upscaled matrices. We use out-of-the-box parameters to construct the 3D models of the chromatin and compare them using the Template modeling score (TM-Score) to estimate the reconstruction accuracy. A higher accuracy indicates more similar 3D strucutre which consequently corresponds to potentially having similar biological properties. Secondly, we implemented Insulation Score [1] to compare TAD structures in upscaled matrices against the HRC matrices. We used the parameters used by the VeHiCle [4] to estimate similarity of TAD boundaries by comparing the similarity by computing the L2 distance between the insulation vectors.

4 Results

In this section, based on the pitfalls identified by Sean et al [15] we design an evaluation study to establish how well the existing methods generalize to their real-world use cases. 1) We setup the existing methods [4, 5, 10, 11, 17] to establish that correlation metrics' potential to mislead to a wrong conclusion about performance of these methods. 2) We report the performance of the existing methods on real-world Low Read Count (LRC) datasets.

3) We conduct 3D chromatin reconstruction and TAD boundary comparison analysis to assess the quality of biological information retrievable from upscaled matrices to establish their relevance for downstream applications.

4.1 HiCPlus, HiCNN and DeepHiC give best performance for downsampled input Hi-C datasets across different downsampling fractions

We have set up a pipeline similar to DeepHiC, which dictates the downsampling ratios ($\frac{1}{16}$ to $\frac{1}{100}$), evaluation metrics, and cell lines (GM12878, IMR90, and K562) we have used to evaluate the performance of Hi-C upscaling models. This experiment explores our hypothesis that correlation-based metrics (such as SSIM) can potentially lead to false conclusions. We show the Figure 2 **A** visual output of the methods to confirm the validity of our implementations. DeepHiC's output shows the highest similarity with the HRC matrix, while our non-deep-learning-based baseline struggles to generate a reasonable output. Eventhough, the output of the HiCPlus looks different from the HRC matrix, but it still shows a high score on the qualitative metrics. We condense our quantitative results in the Figure 2 **B** where we show measurements on GenomeDISCO and SSIM. Results for the remaining metrics are in the Supplementary Tables S1 - S12.

In the Figure 2 **B** we show the upscaling performance (y-axis either SSIM or GenomeDISCO) at different sparsity ratios (x-axis) for the downsampled datasets. Based on the results, while all of these methods provide good Hi-C upscaling performance on lower downsampling ratios, HiCPlus, HiCNN and DeepHiC show similar or comparable performance on higher downsampling ratios which provides empirical proof that they generalize better on the downsampled datasets. Moreover, we observe that VeHiCLe shows consistent performance but worse than other methods, which is expected given VeHiCle was trained to upscale real-world LRC Hi-C matrices. While SSIM suggests that VeHiCle and HiCNN2 perform better than Gaussian smoothing, GenomeDISCO implies an opposite conclusion. Thus confirming our hypothesis that SSIM (a correlation based metric) can lead to false conclusions. Considering these insights, we make two key choices for the rest of our evaluations. 1) We only consider HiCPlus, HiCNN and DeepHiC as our prime candidates for retraining since they provide the best generalizability. 2) We only consider the results from the Hi-C specific similarity metrics when comparing the performance of Hi-C upscaling methods.

4.2 Investigating the performance of deep-learning methods on real-world LRC datasets

We conduct a series of tests to establish whether the existing Hi-C upscaling methods can achieve similar or comparable performance on real-world Hi-C matrices. We hypothesize that downsampled Hi-C matrices do not have the same underlying distribution as the real-world LRC matrices because the uniform downsampling of the read counts fails to incorporate the impact of experimental noise and consequently generates unrealistic Hi-C matrices. If our hypothesis is proven true, based on the pifalls identified in [15] we expect the models trained with downsampled datasets will not perform well on real-world LRC matrices because the model.

In this evaluation phase, we investigate the following. (1) Is there a difference in the underlying distribution of real-world LRC datasets and the downsampled HRC datasets? (2) What is the impact of these differences on the performance of the models trained with downsampled datasets? (3) What retraining strategies can potentially improve the generalizability of the existing methods to real-world LRC datasets?

4.2.1 Distribution of downsampled HRC Hi-C input is different from the real-world LRC datasets

As a preliminary analysis of the data, we compare the distributions of both downsampled and real-world LRC Hi-C matrices. To compare the distributions, we compare the similarity of our Hi-C matrices against HRC matrices at a range of Genomic distances from the diagonal to visualize how the correlation varies. We show results for each LRC dataset 3 and compare its distribution with a downsampled dataset with a comparable number of reading

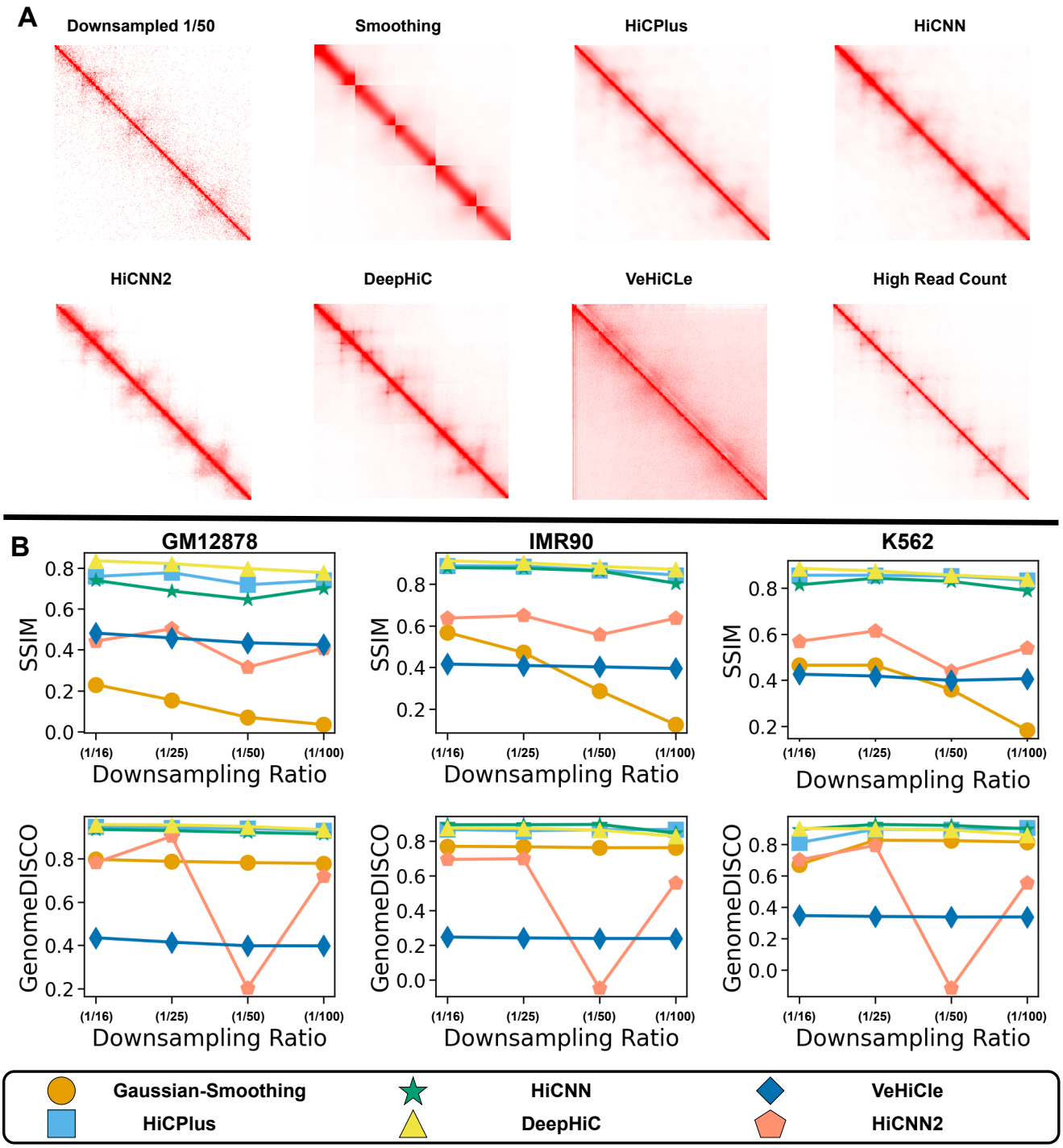

Figure 2: **Performance evaluation of deep learning-based methods for upscaling downsampled input Hi-C datasets**. **A** Visual Comparison of deep learning-based methods for Hi-C matrix enhancement on first 2MB of chromosome 20 with 1/50 downsampled input. **B** On the x-axis, we have the downsampling ratio, and on the y-axis, we have either SSIM or GenomeDISCO. We observe as the downsampling ratio increases, HiCPlus, DeepHiC, and HiCNN show similar or comparable performance. The figure also presents a discrepancy between the trends observed in SSIM and GenomeDISCO. SSIM suggests that VeHiCle performs better than the Gaussian Distribution baseline, while GenomeDISCO suggests otherwise. This discrepancy highlights the limitations of the correlation/image-based metrics for evaluating Hi-C sample similarity and their potential to lead to false conclusions.

counts. In these graphs, higher the correlation at any given genomic distance means that the dataset more closely resembles the HRC dataset. If the distribution of the LRC dataset is similar to the Downsampled dataset, we would observe that they have an equivalent (or similar) correlation with the HRC dataset at all genomic distances.

In the Figure 3 observe the considerable difference between the correlation curve of the LRC dataset and the Downsampled dataset curve. Across all cell lines, sparsities, and genomic distances, real-world LRC datasets always show a smaller PCC correlation with the target HRC matrix than generated downsampled datasets. This observation provides empirical evidence that downsampled datasets are a poor analog of real-world LRC datasets. Moreover, we also observe that as the sparsity increases, the difference in the PCC across genomic distances increases. For example, in dataset GM12878-LRC-4 with 1/13 sparsity, we observe a difference in PCC against downsampled to be 0.2 (0.63 vs. 0.81). Whereas GM12878-LRC-5 has a sparsity of 1/100, we observe a 0.38 (0.18 vs. 0.56) difference. This difference in correlation suggests that experimental noise is amplified non-linearly with respect to the read count ratio.

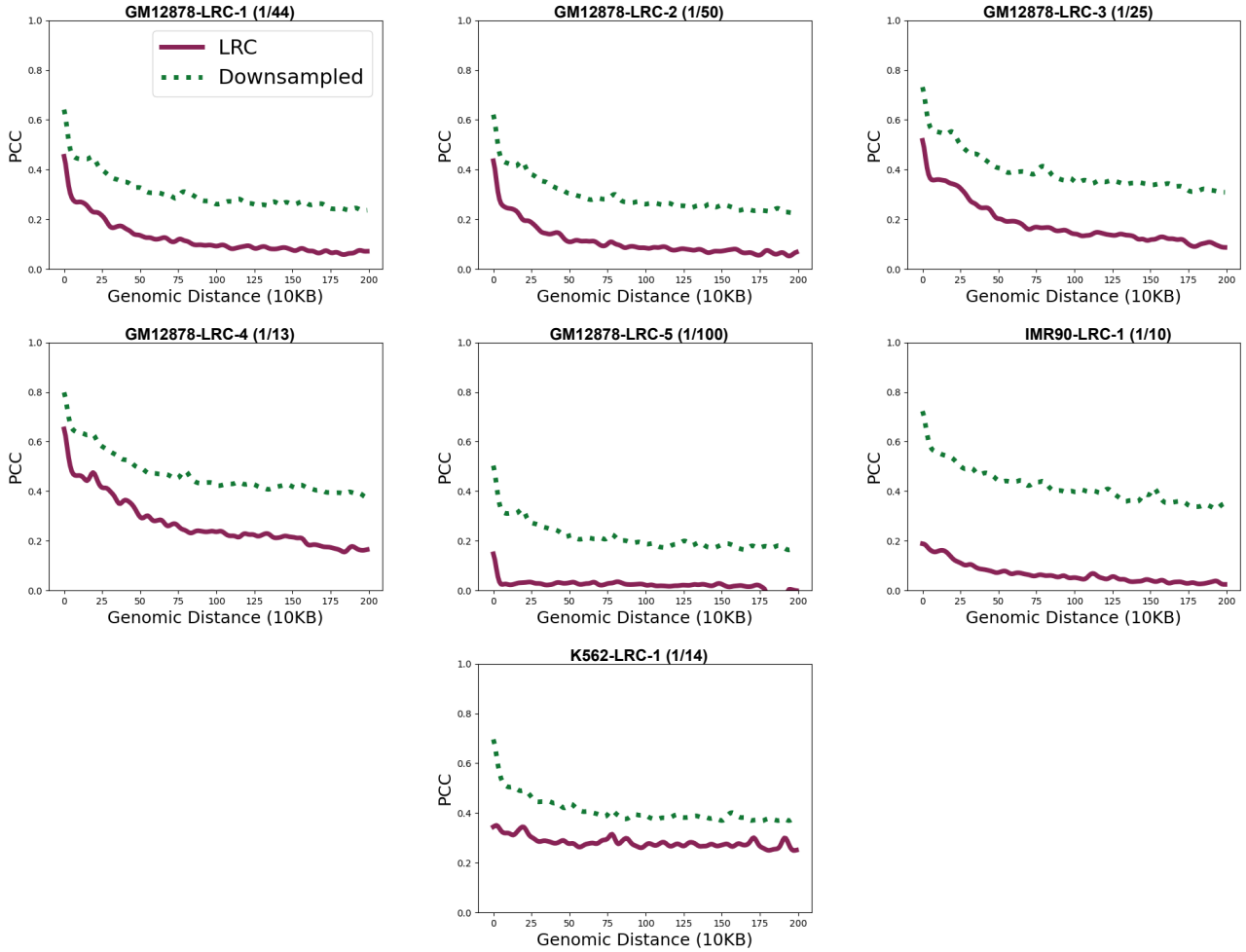

Figure 3: **Comparing the distributions of downsampled Hi-C datasets with the real-world LRC datasets.** The figure compares the read correlation of count distribution (y-axis) across a range of genomic distances (x-axis). Our preliminary analysis shows that LRC datasets show a smaller correlation with the HRC dataset across the entire range of genomic distances than a similarly downsampled dataset.

This analysis confirms our initial hypothesis that uniformly downsampling read counts fail to produce Hi-C matrices with identical information content as the real-world LRC matrices. Consequently, we anticipate that the models trained with these downsampled inputs will perform worse on real-world LRC datasets. Lastly, we

expect the performance difference to be more considerable on matrices with higher sparsity because they have higher experimental noise content.

4.2.2 Deep learning models trained on downsampled Hi-C input perform worse on real-world LRC datasets

We showed a steep difference in the distribution of the real-world LRC and the downsampled datasets. To investigate the impact of that difference, we evaluate the performance of the models trained with the downsampled Hi-C matrices (Baseline version) on real-world LRC matrices. We run these models on seven real-world datasets categorized based on their relative read count ratios. Figure 4(A) shows the visual output of these methods, and directly comparing them with Figure 2 (A), we can observe that all methods have a smaller number of finer features. Figure 4 (B-C) presents the performance results for the GenomeDISCO metric. Results for the rest of the metrics are in the supplementary section Tables S13 - S19.

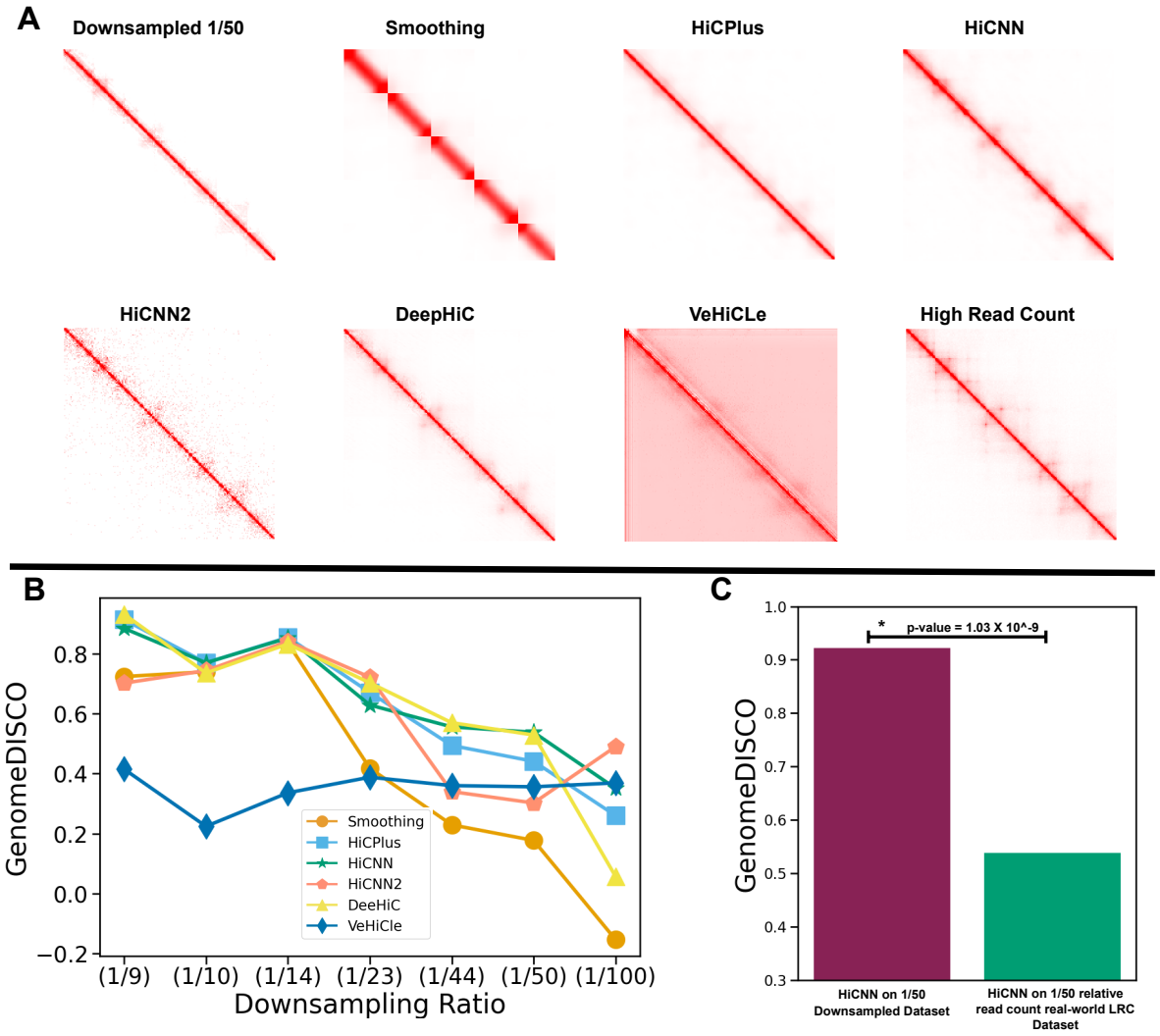

Figure 4: **Performance comparison for upscaling real-world LRC Hi-C datasets. A** We visualize the outputs of our methods on GM12878-LRC-3's first 2MB of chromosome 20. **B** We see that the performance of the deep learning models degrades as the read counts of the input dataset decrease. **C**): The performance of HiCNN (best performing model on average) shows a substantial decrease in performance on LRC datasets compared to the downsampled datasets of the same ($\frac{1}{50}$) relative read count ratio

We observe in Figure 4 **B** that all models show a degradation in performance as the sparsity of the input dataset increases. This result is in tandem with our observation that suggests that the difference in the distribution gets worse as we increase the sparsity. Of all the evaluated methods, VeHiCLe shows the best generalization across the datasets, even though it performs worse than the other methods. Conversely, while HiCPlus and DeepHiC also show good performance, HiCNN shows better generalizability across all datasets, concurrently having similar or better performance than HiCPlus or DeepHiC. Moreover, we additionally compare the performance of HiCNN on downsampled datasets against its performance on real-world LRC datasets in Figure 2(C) with similar relative read counts. On a real-world LRC dataset with the same relative read counts, we see a substantial drop in performance (t-test *p-value* $= 1.03 \times 10^{-9}$), emphasizing the overfitting in HiCNN towards downsampled datasets. In conclusion, these results show that none of the existing methods are tuned to reliably upscale real-world Hi-C data, particularly high sparsity datasets. Since HiCNN provides the best performance across 16 out of 28 (combinations of seven datasets and four Hi-C similarity metrics) metrics. Consequently, we, in subsequent sections, evaluate potential strategies to rectify this performance difference we observe in real-world datasets against downsampled datasets.

4.2.3 Retraining deep learning models with real-world LRC datasets improves performance

To determine the effects of different retraining strategies on the generalizability of the HiCNN model on real-world datasets, we retrain the HiCNN with the following inputs:

1. Downsampled datasets augmented with Gaussian, Random, or Uniform Noise
2. Real-world LRC datasets
3. Ensemble of downsampled datasets with a range of downsampling ratios
4. Ensemble of real-world LRC datasets with varying read counts

The critical insight behind all of these retraining strategies is to expose the model to a more realistic distribution in the training phase to improve the generalizability and performance in the evaluation phase. Most of the retraining strategies provided modest improvements. Most notably, retraining with Gaussian Noise and Real-world LRC ensembles provided an average gain of 13 % and 10%, respectively, over the baseline. In comparison, retraining with Uniform noise decreased the performance by 23%. We show detailed results in the supplementary Tables S20 - S26. Similar to our previous performance analysis, we show the visual outputs of the first 2MB of chromosome 20 of the GM12878-LRC-3 dataset in Figure 5 **A** of our methods to establish the validity of the outputs. Our visualizations show that the output of HiCNN-LRC-1 shows a blocky output which hints toward potential overfitting in the model. In Figure 5 **B-C**, we conduct a quantitative analysis of models retrained with a single LRC dataset. Our analysis shows that retrained versions of HiCNN show better performance than the baseline across all datasets except IMR90-LRC-1 $\frac{1}{10}$. Moreover, of all the retrained versions of HiCNN, we show that retraining HiCNN with a dataset with a modest downsampling ratio of 1/23 (GM12879-LRC-3) shows the best generalizability.

To further emphasize the performance improvement, we compare the performance of the HiCNN-Baseline model against the performance of the retrained HiCNN-LRC-3 (the best performing model) in Figure 5 (C). The bar graph shows a significant difference in GenomeDISCO scores of both of these models (t-test *p-value* $= 5.86 \times 10^{-8}$). We observe similar trends and observations on the other Hi-C-specific similarity metrics, although the improvements are not as significant.

In the results, IMR90-LRC-1 is the only case where HiCNN retrained on the real-world LRC dataset does not perform well. This observation confirms our initial hypothesis that datasets with low correlation with HRC datasets are significantly more challenging to solve. In Figure 3, we observe that IMR90, even though it has

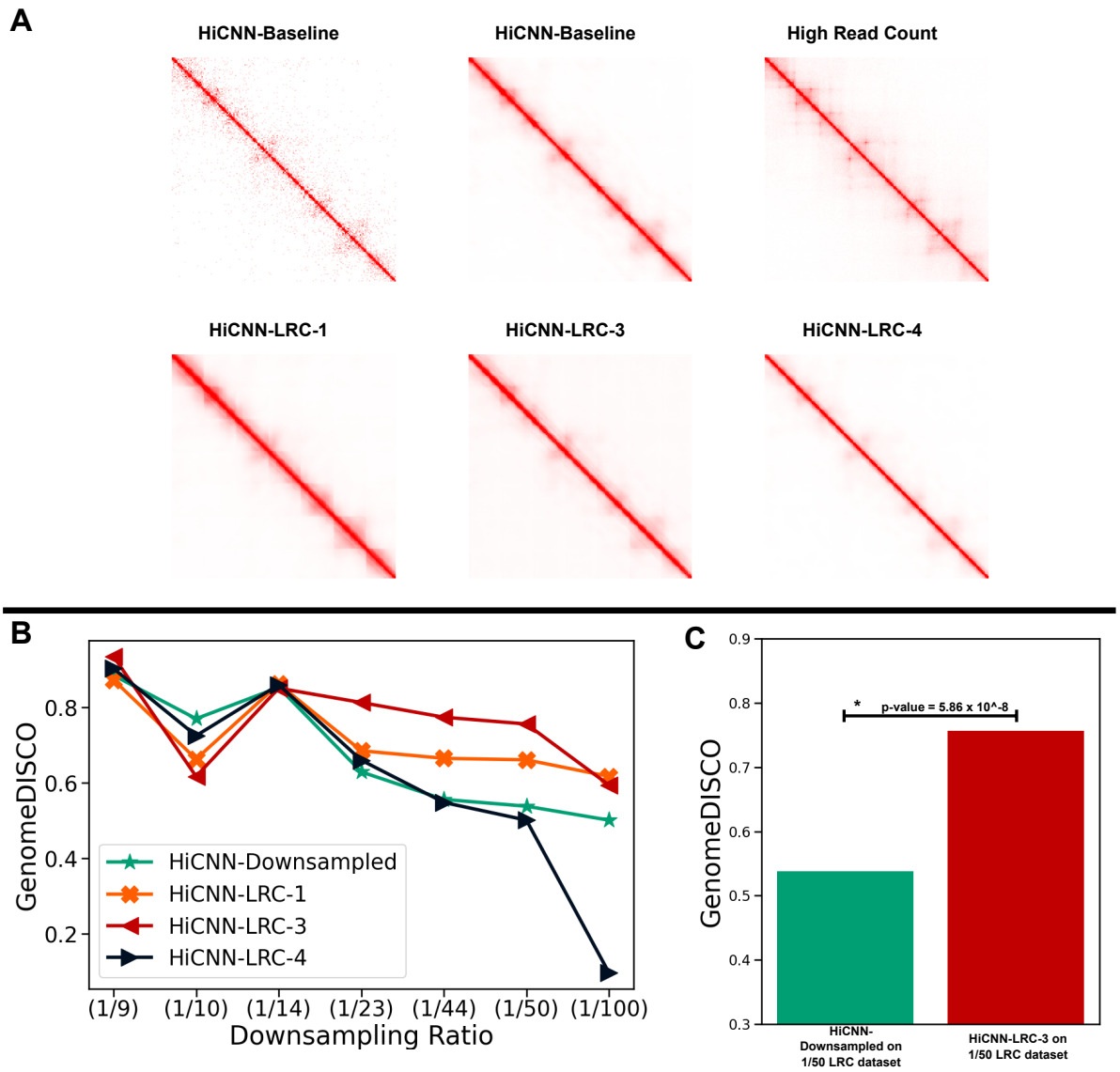

Figure 5: **Retraining HICNN on real-world LRC datasets. A** We show **B** We compare the performance of the baseline HiCNN (HiCNN-Downsampled) model with that of the HiCNN model retrained with real-world LRC datasets (HiCNN-LRC). The results show that HiCNN-LRC performs better and generalizes better to inputs with low relative read count ratios. **B**: To elaborate on the performance improvement, we show that HiCNN-LRC shows substantial improvement (with t-test *p-value* $=1.02\times10^{-6}$ in performance over HiCNN-Downsampled on 1/50 relative read count real-world dataset.

low sparsity, still shows a small correlation across the genomic distances. Lastly, we follow a similar scenario for the GM12878-LRC-5 dataset, where we have a small correlation with the HRC dataset and accordingly turns out to be a more complex task to solve for all the deep-learning-based methods. Consequently, these two cases suggest room for further improvement in the entire training pipeline to mitigate the potential overfitting towards datasets with a high correlation with the HRC dataset.

4.3 Downstream Analysis

To determine whether the upscaled Hi-C matrices retain important biological signals, we analyze the 3D structure of chromatin and significant interactions reconstructed from the upscaled matrices. Unlike previous

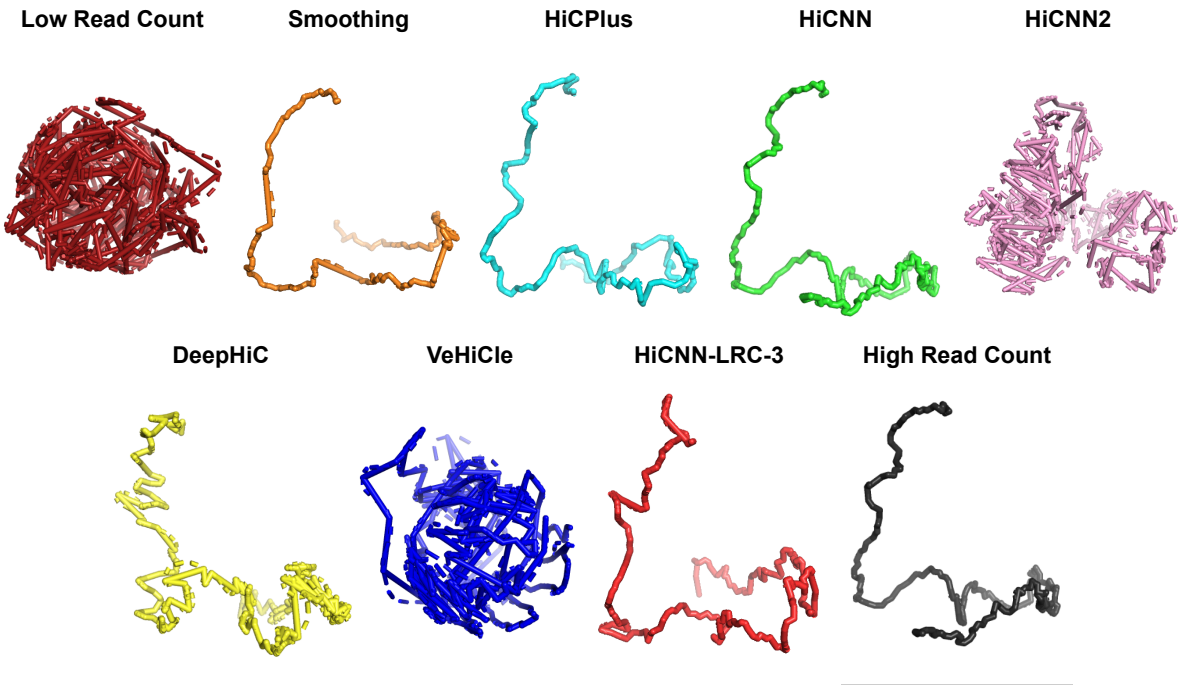

Figure 6: **3D reconstruction of Chromosome 20 (14–16) MB region** Most models including HiCNN-LRC-3 produce highly similar 3D structures of the chromatin.

works [5, 10, 11, 17], we perform downstream analysis on upscaled real-world LRC datasets to provide insights into the generalizability of these methods in real-world scenarios.

4.3.1 Retrained Models generate highly similar 3D structure of chromatin

We use the upscaled or enhanced Hi-C maps produced by all methods for real-world LRC input matrices for all the cell lines, and we use the 3DMax tool [13] to construct the 3D structure for the 200×200 Hi-C sub-matrices for all of the sub-matrices in the test chromosomes. We use TM-score to compare the spatial conformation of the chromatin, where a higher TM-score is proportional to a higher structural similarity of the chromatin. The authors of VeHiCle perform a similar analysis [4], where they compare the recovered 3D structure of the chromatin from the upscaled Hi-C matrices to establish that their upscaled matrices are biologically informative. We also visulize the generated 3D conformations using Pymol toolkit in Figure 6. We observe that VeHiCLe and HiCNN2 generate 3D structures that are noisy and potentially also not biologically informative.

We summarize the analysis results in Table 2. We observe a similar trend, where we show that the retrained HiCNN (HiCNN-LRC-3) outperforms other methods in 4 out of 7 cell lines. Moreover, we observe that HiCNN-LRC-3 performs better in cell lines with either the same sparsity or similar sparsity to the cell line used for training.

4.3.2 Retrained models closely follow behind Baseline models in recovering TAD boundaries

We estimated the TAD profile by comparing the Insulation vector of enhanced Hi-C matrices against the HRC matrices. We generated the Insulation vector using the code provided in the VeHiCLe codebase with its default parameters on 200×200 Hi-C submatrices of all the test chromosomes. We then, similar to VeHiCle, used the L2 norm to compute the similarity between these vectors to estimate the similarity of the TAD profiles. This method finds the TAD boundaries by sliding a window across the diagonal of the matrix to construct a

	Baseline	Smoothing	HiCPlus	HiCNN	HiCNN2	DeepHiC	VeHiCLe	HiCNN-LRC-3
GM12878-LRC-1	0.306	0.443	0.609	0.620	0.467	0.582	0.452	**0.621**
GM12878-LRC-2	0.303	0.411	0.556	**0.616**	0.467	0.543	0.446	0.599
GM12878-LRC-3	0.358	0.527	0.688	0.671	0.535	0.761	0.482	**0.740**
GM12878-LRC-4	0.480	0.804	0.850	0.836	0.631	0.832	0.502	**0.858**
GM12878-LRC-5	0.315	0.280	0.385	**0.518**	0.437	0.310	0.409	0.469
IMR90-LRC-1	0.284	0.601	0.649	**0.663**	0.525	0.598	0.414	0.614
K562-LRC-1	0.334	0.658	0.694	0.687	0.679	0.658	0.507	**0.701**

Table 2: **TM scores of HRC Chromatin structures vs upscaled and LRC Chromatin structures**. The best score for each dataset is bolded. HiCNN-LRC-3 outperforms other methods in four out of seven datasets and has comparable performance in the two lower sparsity datasets.

1D insulation vector. Zeros on this insulation vector represent the discovered TAD boundaries.

We summarize our findings in the Table 3. This analysis finds that HiCNN and HiCPlus show the best performance in three out of seven datasets, with HiCNN-LRC-3 closely following in all six of these datasets. However, none of the upscaling methods improve over the baseline in the GM12879-LRC-5 dataset. This observation provides a critical insight that centering analysis on a single class of metrics can potentially lead to false conclusions regarding the performance and generalizability of these methods. Lastly, the same observation hints toward potential bias in methods that, even when retrained with appropriate datasets, cause the models to struggle in capturing the underlying biological signals. Regardless, we arrive at the same conclusion in the context of all the evaluation methodologies that retraining with appropriate datasets improves generalizability of Hi-C upscaling methods.

	Baseline	Smoothing	HiCPlus	HiCNN	HiCNN2	DeepHiC	VeHiCLe	HiCNN-LRC-3
GM12878-LRC-1	0.01015	0.00733	0.00456	**0.00352**	0.00466	0.00772	0.02641	0.00363
GM12878-LRC-2	0.00901	0.00711	0.00520	**0.00393**	0.00556	0.00700	0.02800	0.00417
GM12878-LRC-3	0.00647	0.00495	0.00271	**0.00226**	0.00318	0.00707	0.01856	0.00255
GM12878-LRC-4	0.00551	0.00380	**0.00131**	0.00147	0.00265	0.00550	0.16561	0.00197
GM12878-LRC-5	**0.00672**	0.00689	0.01627	0.01678	0.01569	0.02156	0.04074	0.02070
IMR90-LRC-1	0.03551	0.02941	**0.01882**	0.02482	0.03059	0.03500	0.03358	0.02540
K562-LRC-1	0.01554	0.01336	**0.00637**	0.01127	0.01058	0.02590	0.03928	0.01509

Table 3: **Insulation Score of Enhanced matrices against HRC matrices**. The lower the score, the better the TAD recovery; the best score for each dataset is bolded. HiCNN and HiCPlus show the best performance in six out of seven datasets, where HiCNN-LRC-3 closely follows both in all six datasets.

5 Conclusion

This work has conducted a series of experiments to explore how the existing Hi-C upscaling methods perform in their intended real-world scenarios. Our results strongly suggest that none of the proposed methods achieved similar or comparable performance. Among the existing techniques, HiCNN, a non-adversarial model, showed the best generalizability, hinting at potential overfitting in the nuanced methods towards Hi-C matrices of a particular modality. We suggest that the upcoming methods employ stricter evaluation protocols that include a wide range of metrics and real-world datasets to ensure that the advertised performance would be comparable in real-world scenarios. We would still want to highlight that we neither evaluated these methods on single-cell datasets nor Hi-C experiments treated with different experimental protocols. We hypothesize that the performance of these methods on those datasets would be worse.

To provide potential solutions to improve the generalizability, we explore retraining with different modal-

ities of input augmentation. Our analysis found that retraining HiCNN with 1/25 sparsity provided the best generalizability across the seven Hi-C datasets procured from various sources and cell lines with varying sparsities. Although we didn't observe significant improvements in TAD recall analysis, retrained HiCNN still showed better generalizability than the best-performing methods. Regardless, the conflicting takeaways from experiments imply that using a single class of metrics is inadequate for estimating the biological usefulness of enhanced Hi-C matrices. Lastly, we do not retrain any of the more nuanced methods such as VeHiCLe or DeepHiC. We still believe that existing methods and potentially all the upcoming methods can use our procured datasets and processing pipeline to streamline the process of training and evaluation.

6 Acknowledgments

We thank Justin Zook and Justin Wagner from NIST (National Institute of Standards and Technology) for providing incredibly valuable feedback on the draft. We also would like to thank members of Singh Lab at Brown, specifically Hyeyeon Hwang for sharing their valuable experience about HiC downstream analysis methods and having many helpful discussions with us on the draft.

References

[1] E. Crane, Q. Bian, R. P. McCord, B. R. Lajoie, B. S. Wheeler, E. J. Ralston, S. Uzawa, J. Dekker, and B. J. Meyer. Condensin-driven remodelling of x chromosome topology during dosage compensation. *Nature*, 523(7559):240–244, 2015.

[2] J. R. Dixon, S. Selvaraj, F. Yue, A. Kim, Y. Li, Y. Shen, M. Hu, J. S. Liu, and B. Ren. Topological domains in mammalian genomes identified by analysis of chromatin interactions. *Nature*, 485(7398):376–380, 2012.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[4] M. Highsmith and J. Cheng. Vehicle: A variationally encoded hic loss enhancement algorithm for improving and generating hi-c data. *Scientific Reports*, 11(1), 2021.

[5] H. Hong, S. Jiang, H. Li, G. Du, Y. Sun, H. Tao, C. Quan, C. Zhao, R. Li, W. Li, and et al. Deephic: A generative adversarial network for enhancing hic data resolution. *PLOS Computational Biology*, 16(2), 2020.

[6] F. Jin, Y. Li, J. R. Dixon, S. Selvaraj, Z. Ye, A. Y. Lee, C.-A. Yen, A. D. Schmitt, C. A. Espinoza, B. Ren, and et al. A high-resolution map of the three-dimensional chromatin interactome in human cells. *Nature*, 503(7475):290–294, 2013.

[7] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks, 2015.

[8] P. A. Knight and D. Ruiz. A fast algorithm for matrix balancing. *IMA Journal of Numerical Analysis*, 33(3):1029–1047, 2012.

[9] E. Lieberman-Aiden, N. L. van Berkum, L. Williams, M. Imakaev, T. Ragoczy, A. Telling, I. Amit, B. R. Lajoie, P. J. Sabo, M. O. Dorschner, R. Sandstrom, B. Bernstein, M. A. Bender, M. Groudine, A. Gnirke, J. Stamatoyannopoulos, L. A. Mirny, E. S. Lander, and J. Dekker. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, 2009.

[10] Q. Liu, H. Lv, and R. Jiang. hicGAN infers super resolution Hi-C data with generative adversarial networks. *Bioinformatics*, 35(14):i99–i107, 07 2019.

[11] T. Liu and Z. Wang. HiCNN: a very deep convolutional neural network to better enhance the resolution of Hi-C data. *Bioinformatics*, 35(21):4222–4228, 04 2019.

[12] T. Liu and Z. Wang. Hicnn2: Enhancing the resolution of hi-c data using an ensemble of convolutional neural networks. *Genes*, 10(11), 2019.

[13] O. Oluwadare, Y. Zhang, and J. Cheng. A maximum likelihood algorithm for reconstructing 3d structures of human chromosomes from chromosomal contact data. *BMC Genomics*, 19(1), 2018.

[14] Rao and et al. A 3d map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell*, 159(7):1665–1680, 2014.

[15] S. Whalen, J. Schreiber, W. S. Noble, and K. S. Pollard. Navigating the pitfalls of applying machine learning in genomics. *Nature Reviews Genetics*, 23(3):169–181, 2021.

[16] G. G. Yardımcı, H. Ozadam, M. E. Sauria, O. Ursu, K.-K. Yan, T. Yang, A. Chakraborty, A. Kaul, B. R. Lajoie, F. Song, and et al. Measuring the reproducibility and quality of hi-c data. *Genome Biology*, 20(1), 2019.

[17] Y. Zhang, L. An, J. Xu, B. Zhang, W. J. Zheng, M. Hu, J. Tang, and F. Yue. Enhancing hi-c data resolution with deep convolutional neural network hicplus. *Nature Communications*, 9(1), 2018.

Investigating the performance of deep learning methods for Hi-C resolution improvement

Supplementary Material

S1 Details of the evaluated methods

- **Gaussian Smoothing** We applied the Gaussian Smoothing filter, commonly used as a baseline [17], to establish emphasize the performance benefits of the deep-learning-based methods. This method uses a 2D kernel of shape $n \times n$ where n is a hyper-parameter. Each kernel value follows a 2D Gaussian distribution with hyperparameters σ_x and σ_y that represent the relative importance of neighboring features in prediction. The smoothing operation convolves this kernel on each pixel of a 2D image, or in our case, a HiC matrix read count, to update its value. This updated value contains the average of the neighboring values weighted by 2D Gaussian distribution in the kernel. This smoothing operation removes noise in the input matrix and improves the peak signal to noise ratio (PSNR) at the cost of blurring the features. For our experiments, we performed a grid search and found the kernel size of $n = 17$ and $\sigma_x = \sigma_y = 7$ to give the best reproducibility score on the validation set of LRC HiC matrices.

- **HiCPlus [17]** HiCPlus is the first application of deep learning to improve HiC resolution. It utilizes a standard three-layer convolutional neural network (CNN) architecture to upscale a low-resolution HiC matrix by mapping it to the target high-resolution matrix. To optimize its parameters, HiCPlus uses a mean squared error (MSE) loss. HiCPlus inputs HiC matrices as sub-matrices of size 40×40 binned at 10Kbp resolution and taken from 2 Mbp distance from the diagonal. All the following deep learning-based models follow the same input formulation. For this study, we make certain modifications to the original code to make HiCPlus comparable to the more recent implementations. For example, the original HiCPlus was trained to generate raw read counts rather than normalized HiC matrices. Therefore, we retrain the model to work with normalized high-resolution and low-resolution pairs of HiC sub-matrices. Moreover, the original implementation was trained to upscale only the matrices that had been downsampled to $\frac{1}{16}$ of the read counts of a high-resolution HiC map. We retrain the HiCPlus with three additional input HiC datasets with $\frac{1}{25}$, $\frac{1}{50}$, and $\frac{1}{100}$ downsampling ratios to explore the performance on matrices beyond the original downsampled version.

- **HiCNN [11]** HiCNN model also uses a CNN architecture like HiCPlus, except that it consists of a much deeper 54-layer neural network with ResNet layers [3]. The choice of ResNet layers provides two key benefits - (1) it provides additional architectural complexity to learn relevant non-linear relationships between the inputs and the outputs, and (2) they train significantly faster than regular CNN layers saving substantial time during training. In addition, the skip connections in ResNet layers further avoid model overfitting on the data. HiCNN, similar to HiCPlus, uses an MSE loss to optimize its parameters and learn the mapping between the input low-resolution matrix and the high-resolution target matrix. It also produces raw read counts and is trained with downsampling ratios of $\frac{1}{8}$ and $\frac{1}{16}$ and $\frac{1}{25}$. Therefore, for consistency, we retrain the model with additional datasets with downsampling rations of $\frac{1}{50}$ and $\frac{1}{100}$ and standardized values.

- **HiCNN2 [12]** HiCNN2 extends the architecture of HiCNN by ensembling multiple methods to achieve better resolution. There are three different ConvNets in HiCNN2; the first ConvNet is similar to HiCNN in terms of how it uses global and local residual learning; it also concatenates features across residual blocks to improve performance. The second ConvNet is a modified version of VDSR [7] which only uses

the global residual learning. The third ConvNet is the HiCPlus model. Each model produces a 28x28 output matrix that is then combined through weights tuned during the training process. This method, similar to HiCPlus and HiCNN, also predicts raw read counts and is only trained with $\frac{1}{8}$ and $\frac{1}{16}$ and $\frac{1}{25}$ downsampling ratios. Similar to both HiCNN and HiCPlus, we retrain HiCNN2 to predict normalized read counts across all four downsampling ratios.

- **HiCGAN [10]** HiCGAN paper argues that the Mean Squared Loss function used in both HiCNN and HiCPlus causes these models to generate over-smooth matrices. Therefore, it proposes using a Generative Adversarial Network (GAN) model for the HiC resolution improvement task. A GAN architecture consists of (1) a generator and (2) a discriminator. The generator's objective is to produce data that increasingly resembles the original distribution, and the goal of the discriminator is to identify fake (generated) data from the original data. This coupled training causes both models to get iteratively better at their tasks. HiCGAN uses a specialized form of GAN, which is called the conditional GAN (cGAN). In cGANs the generator produces an output conditional on the provided input instead of a random noise input. To optimize the parameters in the model, HiCGAN uses the discriminator loss and the MSE loss to generate matrices that are highly similar to the target HiC matrices. The HiCGAN paper shows that this method produces better-quality HiC matrices with sharper and more prominent features than the previous method.

- **DeepHiC [5]** DeepHiC paper, like HiCGAN [10], argues that the Mean Squared Loss function used in both HiCNN and HiCPlus causes these models to generate over-smooth matrices. Therefore, it uses Generative Adversarial Network (GAN) model for the HiC resolution improvement task. A GAN architecture consists of (1) a generator and (2) a discriminator. The generator's objective is to produce data that increasingly resembles the original distribution, and the goal of the discriminator is to identify fake (generated) data from the original data. This coupled training causes both models to get iteratively better at their tasks. DeepHiC substantially revises the previously proposed loss functions to contain additional functions that include Total Variation Loss and Perceptual Loss. These loss function along side Mean Squared Error Loss and Discriminator Loss causes the model to generate matrices with sharper features that are more biologically informative [5]. The paper also shows that training the deep learning models on standardized HiC matrices improves their performance further. Moreover, the paper trains the DeepHiC model with a downsampling ratio of up to $\frac{1}{100}$ to have model weights available for even the sparsest real-world HiC matrices.

- **VeHiCLE [4]** VeHiCLE is another GAN-based model like HiCGAN and DeepHiC. However, it makes additions to both the model architecture and loss functions used while training. Apart from using a GAN architecture, it also trains a variational auto-encoder (VAE). The output obtained by passing the HiC matrices through the trained VAE is used in a loss function to train the GAN. This loss obtained from the VAE is called the variation loss. VeHiCLE also uses the adversarial and MSE losses seen in previous methods. However, it introduces yet another loss called insulation loss. The insulation loss is a biologically inspired loss that utilizes insulation scores used to identify TADs in a HiC contact matrix. VeHiCLE is trained on input and target matrices of sizes 269×269, unlike the previous methods that used 40×40 sub-matrices as inputs. The paper shows that this increased matrix size improves performance, thus hypothesizing that the 40×40 matrices are too small to adequately capture information about large-scale HiC features (such as TADs). We created new datasets for VeHiCle that had HiC sub-matrices of size 269x269 to ensure a fair comparison of VeHiCle with other deep learning based methods.

S2 Details of the evaluation metrics

- **Mean Squared Error (MSE)** Mean Squared Error or MSE is the squared differences of the errors between the predicted value and the true/actual value. Smaller Mean Squared error is representative of better reproduction of the signal and consequently better quality of the predicted signal.

- **Mean Average Error (MAE)** Similar to Mean Square Error, Mean Average Error is the average of the differences of errors (rather than squared) between the predicted value and the true/actual value. MAE value of zero corresponds to an excellent signal reproduction.

- **Peak Signal to Noise Ratio (PSNR)** Peak Signal to Noise Ratio or PSNR is the ratio of maximum possible value or power of the signal to the power of the noise in the signal. PSNR is generally calculated in Decibels, a logarithmic scale, to compensate for the wide range of values the ratio can take. The higher the ratio value in decibels, the better the quality of the signal with respect to the baseline signal.

- **Structural Similarity Index Measure (SSIM)** Structural Similarity Index Measure (SSIM) is a metric that measures the perceived perceptual quality of an image against an original undistorted and higher quality image. SSIM measures this perceptual quality by comparing the luminance, contrast, and structural properties in small local regions of the images. A weighted sum of these properties allows SSIM to assign a similarity score that closely mimics the way humans perceive differences in images. However, we postulate that the HiC contact maps should be compared based on their underlying biological properties instead of their visual similarities. Therefore, assigning a similarity score based on SSIM score may hold little biological relevance and might lead to misleading conclusions about the quality of the generated datasets.

- **Pearson Correlation Coefficient (PCC)** Pearsons Correlation Coefficient (PCC) is a linear measure of the correlation between two sets of data distribution. PCC is a ratio of covariance between two variables and its product with their standard distribution. This metric essentially measures covariance between the two datasets, normalized to have a value between -1 and 1. Here, -1 or 1 values imply highly negatively or positively correlated, respectively, and a 0 value implies no correlation between the two datasets.

- **Spearman's rank Correlation Coefficient (SCC)** Spearman's rank Correlation Coefficient (SCC) measures the statistical dependence between the rank of two variables. This measure essentially captures how well two variables can be described using a monotonic function. SCC between two variables is equal to the PCC of the rank of variables. Thus, SCC has a value of +1 or -1 when either of the variables is a perfect monotone of the other. It has a value of 0 when they do not correlate monotonically.

S3 Results

S3.1 Deep-learning methods perform well on downsampled input HiC datasets

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.11008	0.20963	5.52605	0.06288	0.54485	0.73332	0.19384	0.94997	0.41990	0.88825	0.55806
Smoothing	0.05320	0.15188	11.04733	0.23010	0.88949	0.90885	0.33079	0.81729	0.38020	0.79692	0.81914
HiCPlus	0.00509	0.03447	22.66080	0.75783	0.97225	0.91466	0.34649	0.95498	0.60261	0.94751	0.87723
HiCNN	0.00686	0.04023	19.87945	0.73869	0.97361	0.91818	**0.35235**	0.95683	0.62891	0.93605	0.88724
HiCNN2	0.01298	0.06117	17.05465	0.44195	0.92467	0.67383	0.19575	0.85917	0.44753	0.78190	0.72373
DeepHiC	**0.00070**	**0.01620**	**31.82955**	**0.83453**	**0.98884**	**0.92307**	0.34917	**0.96802**	**0.68794**	**0.95887**	**0.90429**
VeHiCLe	0.01066	0.05420	19.63624	0.48169	0.86545	0.77206	0.26001	0.78715	0.21611	0.43482	0.63759

Table S1: Results on GM12878 16 downsampled datasets. DeepHiC performs better than other methods across both Correlation based metrics and HiC similarity metrics

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.14377	0.25005	1.49535	0.03989	0.49769	0.66919	0.15932	0.92578	0.38485	0.89943	0.44045
Smoothing	0.08012	0.19404	8.73757	0.15470	0.85884	0.90685	0.32165	0.81356	0.37874	0.78812	0.81436
HiCPlus	0.00336	0.02831	24.53244	0.77686	0.97559	0.90979	0.33828	0.94572	0.55925	0.94260	0.86946
HiCNN	0.00981	0.04836	18.11373	0.68715	0.97487	0.91296	**0.34588**	0.95328	0.59943	0.93014	0.87757
HiCNN2	0.01251	0.05768	17.51163	0.50216	0.91970	0.66639	0.18676	0.86711	0.42760	0.90421	0.72275
DeepHiC	**0.00081**	**0.01705**	**31.27965**	**0.82070**	**0.98720**	**0.91708**	0.34127	**0.96361**	**0.64535**	**0.95708**	**0.88990**
VeHiCLe	0.01407	0.05942	18.09908	0.45769	0.83495	0.76435	0.24685	0.77469	0.20689	0.41448	0.61900

Table S2: Results on GM12878 25 downsampled datasets. DeepHiC performs better than other methods across both Correlation based metrics and HiC similarity metrics

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.22831	0.34438	-2.27733	0.01839	0.40562	0.56025	0.11189	0.87268	0.34186	0.87636	0.27463
Smoothing	0.15512	0.29079	5.36004	0.07054	0.76404	0.89967	0.29960	0.80697	0.37629	0.78215	0.80310
HiCPlus	0.00712	0.04177	20.12533	0.71753	0.96711	0.90049	0.32710	0.93468	0.52822	0.93967	0.84947
HiCNN	0.01186	0.05329	16.96175	0.64668	0.97198	0.90252	**0.33399**	0.93879	0.53168	0.92207	0.85616
HiCNN2	0.00830	0.05842	19.69436	0.31494	0.96636	0.78714	0.28900	0.92074	0.44416	0.20139	0.66291
DeepHiC	**0.00106**	**0.01886**	**30.21573**	**0.79694**	**0.98340**	**0.90609**	0.32571	**0.95122**	**0.59611**	**0.94936**	**0.86555**
VeHiCLe	0.01766	0.06433	16.55151	0.43423	0.80144	0.75356	0.23243	0.76061	0.20069	0.39781	0.59373

Table S3: Results on GM12878 50 downsampled datasets. DeepHiC performs better than other methods across both Correlation based metrics and HiC similarity metrics

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.39112	0.50501	-3.88875	0.00814	0.29224	0.42670	0.07376	0.79629	0.29694	0.82143	0.15745
Smoothing	0.31078	0.45397	2.05268	0.03460	0.59325	0.87810	0.26368	0.80245	0.36545	0.77866	0.77964
HiCPlus	0.00403	0.03214	23.81592	0.73873	0.96994	0.88044	0.30308	0.92197	0.49171	0.93053	0.81302
HiCNN	0.00746	0.04162	19.04613	0.70089	0.96051	0.89346	**0.31776**	0.91847	0.49907	0.91466	0.82914
HiCNN2	0.01394	0.06352	16.79451	0.40715	0.92456	0.72274	0.20909	0.84106	0.42721	0.71825	0.71545
DeepHiC	**0.00134**	**0.02056**	**29.29867**	**0.77731**	**0.97921**	**0.89628**	0.31379	**0.93651**	**0.51943**	**0.93469**	**0.83514**
VeHiCLe	0.01788	0.06461	16.94529	0.42453	0.79083	0.74686	0.22448	0.75045	0.18824	0.39741	0.58180

Table S4: Results on GM12878 100 downsampled datasets. DeepHiC performs better than other methods across both Correlation based metrics and HiC similarity metrics

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.03737	0.08877	6.64005	0.28986	0.68672	0.63914	0.20019	0.90999	0.39523	0.87071	0.42887
Smoothing	0.01195	0.04828	16.95480	0.56837	0.87453	0.84507	0.32326	0.77594	0.31362	0.77083	0.75939
HiCPlus	0.00148	0.01485	27.28074	0.88915	0.94130	0.84992	0.34487	0.93037	0.45515	0.86772	0.80780
HiCNN	0.00236	0.01672	23.97865	0.88023	0.91371	**0.85522**	0.34572	0.94226	0.44505	**0.89537**	**0.82312**
HiCNN2	0.00577	0.02919	20.39477	0.63682	0.84881	0.52231	0.17185	0.62998	0.29779	0.69503	0.52334
DeepHiC	**0.00044**	**0.01006**	**33.65334**	**0.91307**	**0.98456**	0.85517	**0.35463**	**0.95579**	**0.53280**	0.87724	0.82238
VeHiCLe	0.00825	0.05057	17.89664	0.41548	0.73459	0.70770	0.24177	0.70174	0.17317	0.24675	0.56352

Table S5: Results on IMR90 16 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.04860	0.10668	5.45128	0.24266	0.63000	0.56973	0.16081	0.87701	0.35757	0.85631	0.33602
Smoothing	0.01763	0.06413	15.11927	0.47155	0.88580	0.84136	0.31745	0.77120	0.30927	0.76824	0.75520
HiCPlus	0.00129	0.01387	27.94291	0.88697	0.95182	0.84117	0.33723	0.91784	0.42476	0.86009	0.79463
HiCNN	0.00230	0.01570	24.11307	0.87699	0.92612	**0.84564**	0.33641	0.93730	0.43705	**0.89594**	**0.80836**
HiCNN2	0.00405	0.02760	22.79690	0.64886	0.85392	0.41410	0.15812	0.61922	0.29669	0.69940	0.54848
DeepHiC	**0.00050**	**0.01064**	**33.10889**	**0.90347**	**0.98142**	0.84544	**0.34281**	**0.94650**	**0.48503**	0.87485	0.80114
VeHiCLe	0.00915	0.05115	15.79572	0.40891	0.69354	0.69756	0.23419	0.68460	0.16766	0.24162	0.54375

Table S6: Results on IMR90 25 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.08166	0.15353	3.78173	0.15967	0.51603	0.46564	0.10890	0.79918	0.31554	0.78231	0.22441
Smoothing	0.04080	0.11215	11.12661	0.28780	0.87981	**0.83163**	0.30094	0.76428	0.30604	0.76276	0.73414
HiCPlus	0.00179	0.01582	25.71932	0.86778	0.93440	0.82803	0.31992	0.89189	0.38935	0.86692	0.77112
HiCNN	0.00198	0.01562	25.45118	0.86441	0.93226	0.82985	0.31959	0.91281	0.41454	**0.89732**	**0.78386**
HiCNN2	0.00238	0.02373	25.33843	0.55594	0.94030	0.58122	0.24342	0.80274	0.29253	-0.04901	0.55408
DeepHiC	**0.00068**	**0.01192**	**31.83227**	**0.88632**	**0.97714**	0.83103	**0.32452**	**0.92295**	**0.42838**	0.86516	0.77780
VeHiCLe	0.00990	0.05186	14.68944	0.40227	0.65132	0.68521	0.22479	0.67300	0.15702	0.23817	0.52535

Table S7: Results on IMR90 50 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.16025	0.24749	2.27176	0.07434	0.37324	0.36589	0.06970	0.69388	0.24675	0.62852	0.15882
Smoothing	0.10696	0.20803	6.71718	0.12422	0.79504	0.81559	0.27235	0.75616	0.30164	0.76285	0.71214
HiCPlus	0.00154	0.01541	27.02108	0.84453	0.94577	0.80157	0.29851	0.87408	**0.38097**	**0.86850**	0.73292
HiCNN	0.00266	0.02126	24.37863	0.80503	0.89862	0.81561	0.31078	0.87707	0.36919	0.84766	**0.75884**
HiCNN2	0.00466	0.02618	21.80412	0.63649	0.86523	0.63345	0.21072	0.71249	0.32085	0.55964	0.58718
DeepHiC	**0.00091**	**0.01326**	**30.64917**	**0.87179**	**0.97092**	**0.81928**	**0.31152**	**0.88591**	0.37715	0.82886	0.74365
VeHiCLe	0.00997	0.05284	16.10471	0.39461	0.63850	0.67703	0.21464	0.66993	0.15128	0.23827	0.51515

Table S8: Results on IMR90 100 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.02538	0.07454	9.67603	0.27663	0.80698	0.74438	0.24426	0.93167	0.47647	0.86159	0.48389
Smoothing	0.01493	0.05922	14.66759	0.46544	0.88072	0.84929	0.29794	0.79601	0.24941	0.67093	0.08894
HiCPlus	0.00170	0.01631	25.69619	0.85559	0.93009	0.85191	0.31514	0.93004	0.31919	0.81081	0.66157
HiCNN	0.00195	0.01738	25.02932	0.81416	0.93415	0.79958	0.27148	0.86991	0.37988	0.89527	0.67128
HiCNN2	0.00576	0.03214	20.97862	0.56878	0.83442	0.51721	0.15785	0.61557	0.33908	0.70040	0.47186
DeepHiC	**0.00050**	**0.01186**	**33.12735**	**0.88565**	**0.98065**	**0.86251**	**0.32202**	**0.94623**	**0.52998**	**0.90146**	**0.80142**
VeHiCLe	0.00841	0.04959	16.19766	0.42623	0.70242	0.72151	0.22834	0.71047	0.18979	0.34539	0.52783

Table S9: Results on K562 16 downsampled datasets. DeepHiC performs better than other methods across both Correlation based metrics and HiC similarity metrics

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.04149	0.10290	4.50442	0.17428	0.72379	0.63700	0.17994	0.88770	0.40723	0.88690	0.31200
Smoothing	0.01493	0.05922	14.66759	0.46544	0.88072	0.84929	0.29794	0.79601	0.34057	0.82650	0.72263
HiCPlus	0.00170	0.01631	25.69619	0.85559	0.93009	0.85191	**0.31514**	0.93004	0.48648	0.89595	0.76858
HiCNN	0.00249	0.01792	23.12219	0.84254	0.90950	0.84942	0.30372	0.93294	0.45453	**0.92703**	0.76492
HiCNN2	0.00408	0.03010	22.84774	0.61379	0.83815	0.44536	0.14771	0.67995	0.32472	0.79335	0.49396
DeepHiC	**0.00059**	**0.01247**	**32.46218**	**0.87454**	**0.97669**	**0.85333**	0.31238	**0.93543**	**0.48984**	0.89828	**0.77525**
VeHiCLe	0.00920	0.05050	14.70262	0.41815	0.66788	0.71134	0.22063	0.70218	0.17716	0.34025	0.51091

Table S10: Results on K562 25 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.05470	0.12437	2.78744	0.13111	0.66226	0.56852	0.14429	0.85001	0.37733	0.87367	0.23005
Smoothing	0.02245	0.07957	12.70751	0.35983	0.89182	**0.84598**	0.29185	0.79367	0.33469	0.82364	0.71568
HiCPlus	0.00168	0.01570	25.47291	0.85192	0.94043	0.84376	**0.30792**	**0.91951**	**0.45171**	0.89354	**0.75199**
HiCNN	0.00252	0.01809	23.15744	0.82921	0.91605	0.83457	0.29130	0.90839	0.42766	**0.92052**	0.73511
HiCNN2	0.00330	0.02919	22.86454	0.44024	0.92192	0.59882	0.22965	0.76871	0.32985	-0.11782	0.50568
DeepHiC	**0.00077**	**0.01376**	**31.32310**	**0.85642**	**0.96902**	0.83938	0.29588	0.90587	0.42946	0.89208	0.74214
VeHiCLe	0.01018	0.05470	14.32275	0.39940	0.62821	0.69813	0.20856	0.69303	0.16082	0.33655	0.48385

Table S11: Results on K562 50 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

	MSE	MAE	PSNR	SSIM	PCC	SCC	MI	HiCRep	HiC-Spector	GenomeDISCO	QuASAR-Rep
Baseline	0.09401	0.18075	1.68807	0.06823	0.53550	0.46277	0.09678	0.76518	0.31659	0.81228	0.13776
Smoothing	0.05184	0.13839	8.79676	0.18269	0.88171	**0.83714**	0.27589	0.78442	0.33440	0.81512	0.69259
HiCPlus	0.00213	0.01727	23.95191	0.83432	0.92462	0.83129	**0.29220**	**0.89649**	**0.42489**	**0.90453**	**0.72303**
HiCNN	0.00281	0.02265	23.35855	0.78902	0.88725	0.82000	0.27931	0.87650	0.38153	0.89902	0.70354
HiCNN2	0.00598	0.03109	19.75335	0.54068	0.81558	0.62753	0.18724	0.72062	0.33601	0.55550	0.53318
DeepHiC	**0.00094**	**0.01483**	**30.51630**	**0.84185**	**0.96355**	0.82666	0.28438	0.87208	0.39388	0.85955	0.70302
VeHiCLe	0.00997	0.05101	14.92427	0.40675	0.61493	0.68867	0.19857	0.68984	0.16288	0.33667	0.46767

Table S12: Results on K562 100 downsampled datasets. DeepHiC performs better than other methods on Correlation based metrics white it share its lead in HiC similarity metrics with other methods

S3.2 Deep learning models trained on downsampled HiC input perform worse on real-world LRC datasets

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.728	0.301	0.435	0.203
Smoothing	0.701	0.327	0.229	0.699
HiCPlus	**0.906**	0.452	0.494	0.754
HiCNN	0.901	**0.453**	0.556	**0.772**
HiCNN2	0.703	0.354	0.340	0.564
DeepHiC	0.901	0.425	**0.570**	0.755
VeHiCLe	0.692	0.165	0.360	0.523

Table S13: Comparison of models trained on downsampled datasets on GM12878-LRC-1 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.708	0.287	0.404	0.172
Smoothing	0.677	0.315	0.177	0.687
HiCPlus	**0.893**	**0.430**	0.441	0.726
HiCNN	0.889	**0.430**	**0.538**	**0.748**
HiCNN2	0.640	0.342	0.302	0.550
DeepHiC	0.883	0.397	0.529	0.731
VeHiCLe	0.679	0.160	0.356	0.502

Table S14: Comparison of models trained on downsampled datasets on GM12878-LRC-2 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.854	0.407	0.536	0.329
Smoothing	0.770	0.357	0.418	0.735
HiCPlus	**0.942**	0.517	0.671	0.812
HiCNN	**0.942**	**0.528**	0.629	0.811
HiCNN2	0.893	0.423	**0.722**	0.638
DeepHiC	0.934	0.525	0.703	**0.813**
VeHiCLe	0.753	0.213	0.388	0.593

Table S15: Comparison of models trained on downsampled datasets on GM12878-LRC-3 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.923	0.398	0.781	0.524
Smoothing	0.821	0.376	0.724	0.808
HiCPlus	0.949	0.580	0.914	0.861
HiCNN	0.950	0.605	0.885	**0.868**
HiCNN2	0.927	0.466	0.702	0.763
DeepHiC	**0.957**	**0.625**	**0.932**	0.859
VeHiCLe	0.781	0.217	0.415	0.619

Table S16: Comparison of models trained on downsampled datasets on GM12878-LRC-4 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.288	0.066	-0.732	0.073
Smoothing	0.487	0.241	-0.154	**0.550**
HiCPlus	0.615	**0.342**	0.260	0.514
HiCNN	**0.629**	0.337	0.351	0.501
HiCNN2	0.531	0.317	**0.490**	0.524
DeepHiC	0.539	0.249	0.057	0.496
VeHiCLe	0.624	0.131	0.369	0.403

Table S17: Comparison of models trained on downsampled datasets on GM12878-LRC-5 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.529	0.145	0.598	0.169
Smoothing	0.699	0.289	0.740	0.696
HiCPlus	0.733	0.314	**0.770**	0.716
HiCNN	**0.742**	**0.316**	**0.770**	**0.730**
HiCNN2	0.703	0.266	0.744	0.628
DeepHiC	0.684	0.273	0.736	0.591
VeHiCLe	0.517	0.132	0.224	0.480

Table S18: Comparison of models trained on downsampled datasets on IMR90-LRC-1 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
Baseline	0.767	0.354	0.795	0.190
Smoothing	0.816	0.345	0.836	0.687
HiCPlus	0.893	0.397	**0.855**	0.718
HiCNN	**0.895**	**0.405**	0.853	**0.728**
HiCNN2	0.863	0.352	0.841	0.646
DeepHiC	0.871	0.362	0.833	0.691
VeHiCLe	0.691	0.156	0.336	0.485

Table S19: Comparison of models trained on downsampled datasets on K562-LRC-1 dataset. Best score is bolded for each metric.

S3.3 Retraining deep learning models with real-world LRC datasets improves performance

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.901	0.453	0.556	0.772
HiCNN-LRC-1	0.903	0.408	0.665	**0.773**
HiCNN-LRC-3	0.907	0.400	**0.774**	0.758
HiCNN-LRC-4	**0.918**	**0.449**	0.548	0.753
HiCNN-Synthetic-Ensemble	0.883	**0.449**	0.563	0.754
HiCNN-Real-Ensemble	0.867	0.414	0.631	0.769
HiCNN-Random-Noise	0.894	0.442	0.394	0.734
HiCNN-Gaussian-Noise	0.886	0.371	0.660	0.688
HiCNN-Uniform-Noise	0.889	0.437	0.377	0.730

Table S20: Comparison of retrained models on GM12878-LRC-1 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.889	0.43	0.538	0.748
HiCNN-LRC-1	0.899	0.405	0.661	**0.757**
HiCNN-LRC-3	0.897	0.374	**0.756**	0.736
HiCNN-LRC-4	**0.907**	0.426	0.501	0.726
HiCNN-Synthetic-Ensemble	0.868	**0.430**	0.528	0.731
HiCNN-Real-Ensemble	0.853	0.402	0.606	0.748
HiCNN-Random-Noise	0.881	0.425	0.339	0.714
HiCNN-Gaussian-Noise	0.868	0.367	0.654	0.675
HiCNN-Uniform-Noise	0.877	0.420	0.325	0.710

Table S21: Comparison of retrained models on GM12878-LRC-2 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.942	0.528	0.629	0.811
HiCNN-LRC-1	0.911	0.415	0.685	0.807
HiCNN-LRC-3	0.934	0.484	**0.813**	0.817
HiCNN-LRC-4	**0.948**	**0.527**	0.660	0.811
HiCNN-Synthetic-Ensemble	0.933	0.510	0.664	0.803
HiCNN-Real-Ensemble	0.935	0.483	0.731	**0.818**
HiCNN-Random-Noise	0.930	0.494	0.567	0.785
HiCNN-Gaussian-Noise	0.931	0.430	0.707	0.739
HiCNN-Uniform-Noise	0.920	0.494	0.545	0.779

Table S22: Comparison of retrained models on GM12878-LRC-3 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.950	0.605	0.885	**0.868**
HiCNN-LRC-1	0.915	0.440	0.873	0.846
HiCNN-LRC-3	0.936	0.552	**0.935**	0.864
HiCNN-LRC-4	**0.953**	**0.609**	0.903	**0.866**
HiCNN-Synthetic-Ensemble	0.942	0.602	0.875	0.862
HiCNN-Real-Ensemble	0.948	0.575	0.923	0.865
HiCNN-Random-Noise	0.933	0.562	0.844	0.857
HiCNN-Gaussian-Noise	0.938	0.544	0.857	0.841
HiCNN-Uniform-Noise	0.922	0.554	0.825	0.855

Table S23: Comparison of retrained models on GM12878-LRC-4 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.629	0.337	0.351	0.501
HiCNN-LRC-1	0.676	**0.334**	**0.617**	0.518
HiCNN-LRC-3	0.631	0.203	0.594	0.476
HiCNN-LRC-4	0.600	0.303	0.096	**0.523**
HiCNN-Synthetic-Ensemble	0.628	0.305	0.467	0.468
HiCNN-Real-Ensemble	0.663	0.332	0.523	0.463
HiCNN-Random-Noise	0.615	**0.334**	0.175	0.512
HiCNN-Gaussian-Noise	**0.686**	0.268	0.598	0.413
HiCNN-Uniform-Noise	0.615	0.324	0.109	0.522

Table S24: Comparison of retrained models on GM12878-LRC-5 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	**0.742**	**0.316**	**0.770**	**0.730**
HiCNN-LRC-1	0.720	0.287	0.663	0.720
HiCNN-LRC-3	0.739	0.313	0.617	0.728
HiCNN-LRC-4	0.741	0.313	0.725	0.723
HiCNN-Synthetic-Ensemble	0.735	0.307	0.689	0.716
HiCNN-Real-Ensemble	0.737	0.301	0.612	0.707
HiCNN-Random-Noise	0.732	0.306	0.722	0.724
HiCNN-Gaussian-Noise	0.731	0.284	0.562	0.675
HiCNN-Uniform-Noise	0.725	0.302	0.723	0.722

Table S25: Comparison of retrained models on IMR90-LRC-1 dataset. Best score is bolded for each metric.

	HiCRep	HiCSpector	GenomeDISCO	QuASAR-Rep
HiCNN-Baseline	0.895	**0.405**	0.853	0.728
HiCNN-LRC-1	0.878	0.356	0.862	0.721
HiCNN-LRC-3	0.890	0.403	0.851	0.729
HiCNN-LRC-4	0.897	**0.405**	0.860	0.719
HiCNN-Synthetic-Ensemble	0.897	0.399	**0.864**	**0.733**
HiCNN-Real-Ensemble	**0.902**	0.388	**0.864**	0.722
HiCNN-Random-Noise	0.892	0.399	0.853	0.726
HiCNN-Gaussian-Noise	0.893	0.391	0.857	0.724
HiCNN-Uniform-Noise	0.885	0.400	0.853	0.724

Table S26: Comparison of retrained models on K562-LRC-1 dataset. Best score is bolded for each metric.

S3.4 Retrained models acheive better 3D recontructions than baseline models

	GM12878-LRC-1	GM12878-LRC-2	GM12878-LRC-3	GM12878-LRC-4	GM12878-LRC-5	IMR90-LRC-1	K562-LRC-1
Baseline	0.306	0.303	0.358	0.480	0.315	0.284	0.334
Smoothing	0.443	0.411	0.527	0.804	0.280	0.601	0.658
HiCPlus	0.609	0.556	0.688	0.850	0.385	0.649	0.694
HiCNN	0.620	0.616	0.671	0.836	0.518	**0.663**	0.687
HiCNN2	0.467	0.467	0.535	0.631	0.437	0.525	0.679
DeepHiC	0.582	0.543	**0.761**	0.832	0.310	0.598	0.658
VeHiCLe	0.452	0.446	0.482	0.502	0.409	0.414	0.507
HiCNN-LRC-1	**0.692**	**0.654**	0.743	0.853	**0.518**	0.583	0.678
HiCNN-LRC-3	0.621	0.599	0.740	**0.858**	0.469	0.614	0.701
HiCNN-LRC-4	0.598	0.566	0.656	0.846	0.380	0.630	0.679
HiCNN-Synthetic-Ensemble	0.654	0.609	0.660	0.808	0.466	0.593	0.681
HiCNN-Real-Ensemble	0.674	0.641	0.737	0.835	0.482	0.563	0.684
HiCNN-Random-Noise	0.487	0.449	0.583	0.815	0.474	0.633	0.673
HiCNN-Gaussian-Noise	0.540	0.547	0.580	0.787	0.459	0.598	**0.728**
HiCNN-Uniform-Noise	0.455	0.437	0.574	0.793	0.432	0.621	0.664

Table S27: This table contains the 3D reconstruction scores for all the retrained models along with all models trained with downsampled datasets. The best (highest) score for each dataset (on column) is bolded. Retrained models attain competitive scores in all the datasets.

S3.5 Retrained models attain competitive TAD reconstruction profiles

	GM12878-LRC-1	GM12878-LRC-2	GM12878-LRC-3	GM12878-LRC-4	GM12878-LRC-5	IMR90-LRC-1	K562-LRC-1
Baseline	0.01015	0.00901	0.00647	0.00551	**0.00672**	0.03551	0.01554
Smoothing	0.00733	0.00711	0.00495	0.00380	0.00689	0.02941	0.01336
HiCPlus	0.00456	0.00520	0.00271	**0.00131**	0.01627	**0.01882**	**0.00637**
HiCNN	**0.00352**	0.00393	**0.00226**	0.00147	0.01678	0.02482	0.01127
HiCNN2	0.00466	0.00556	0.00318	0.00265	0.01569	0.03059	0.01058
DeepHiC	0.00772	0.00700	0.00707	0.00550	0.02156	0.03500	0.02590
VeHiCLe	0.02641	0.02800	0.01856	0.16561	0.04074	0.03358	0.03928
HiCNN-LRC-1	0.00357	**0.00385**	0.00268	0.00211	0.01867	0.02467	0.01325
HiCNN-LRC-3	0.00363	0.00417	0.00255	0.00197	0.02070	0.02540	0.01509
HiCNN-LRC-4	0.00374	0.00430	0.00232	0.00169	0.01717	0.02664	0.01428
HiCNN-Synthetic-Ensemble	0.00556	0.00623	0.00295	0.00157	0.01758	0.02381	0.00917
HiCNN-Real-Ensemble	0.00738	0.00751	0.00392	0.00178	0.01736	0.02304	0.00841
HiCNN-Random-Noise	0.00409	0.00469	0.00239	0.00153	0.01749	0.02533	0.01143
HiCNN-Gaussian-Noise	0.00520	0.00615	0.00287	0.00145	0.01657	0.02484	0.00974
HiCNN-Uniform-Noise	0.00404	0.00457	0.00249	0.00162	0.01738	0.02414	0.01099

Table S28: This table contains the insulation scores for all the retrained models along with all models trained with downsampled datasets. The best (lowest) score for each dataset (on column) is bolded. Retrained models attain competitive scores in all the datasets.