UNIVERSITY OF CALIFORNIA
RIVERSIDE

Studying the Spatial Organizations of Chromosomes With Machine Learning

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Yangyang Hu

December 2021

Dissertation Committee:

Dr. Wenxiu Ma, Chairperson
Dr. Tao Jiang
Dr. Stefano Lonardi
Dr. Evangelos Papalexakis

The Dissertation of Yangyang Hu is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

Firstly I would like to express my most appreciation to my advisor Dr. Wenxiu Ma, for her invaluable guidance and unparalleled support in both my research and career. Her patience and insightful feedback pushed me to sharpen my thinking and brought the research work to a higher level. I would like to extend my sincere thanks to my committee: Dr. Tao Jiang, Dr. Stefano Lonardi and Dr. Evangelos E. Papalexakis, for their insightful feedback and practical suggestions, as well as their guidance throughout my studies.

I am grateful to Dr. Xin Gao and Cheng Wang for collaborating with the measurement of methods in Chapter 2. I want to thank Dr. Wei Wu for valuable discussions and the reviewers for constructive suggestions about the rank in Chapter 4.

In addition, I would like to thank all the members, former and present, in Dr. Ma's lab: Tiantian Ye, Luke Klein, Huiling Liu, Jinli Zhang, Li Ma, Biswanath Chowdhury, Rui Ma, Jingong Huang, Guoyao Hao, Sydney Pun and Eleonora Khabirova for their helpful feedback, discussion, and suggestions. Last but not least, I would like to thank my parents for their love and support.

This work is supported by grants from the National Science Foundation (DBI-1751317) and the National Institute of Health (R35GM133678). The text of this dissertation, in part, is a reprint of the material as it appears in *Bioinformatics*, published in July 2021. The co-author Wenxiu Ma listed in that publication directed and supervised the research which forms the basis for this dissertation.

To my parents for all the support.

ABSTRACT OF THE DISSERTATION

Studying the Spatial Organizations of Chromosomes With Machine Learning

by

Yangyang Hu

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, December 2021
Dr. Wenxiu Ma, Chairperson

The Hi-C technique has enabled genome-wide mapping of chromatin interactions and investigated the organizational principles of the spatial structure of the genome. However, computational methods for studying Hi-C are still in the early stage. Pioneering models have been developed to reconstruct the 3D genome structures. But there is no consistent measuring result between the modeling methods, thereby imposing challenges for the users to interpret the organization of the 3D structures and to understand the genome functions.

Moreover, 3D genome modeling becomes more complicated at higher resolution because of the sparsity and diversity in bulk Hi-C and the restrictions of computational resources. Furthermore, high-resolution Hi-C requires costly, deep sequencing; therefore, it has only been achieved for a limited number of cell types. Neural networks have been developed as a remedy to these problems at high resolution.

In this work, we first reviewed the 3D structure modeling methods comprehensively. We developed a simulation method based on the 3D conformations from single-cell modeling and several evaluation metrics for measuring the similarity between structures. We profiled the performance

of existing bulk Hi-C based 3D genome structure modeling methods using both simulated and real bulk Hi-C.

Next, we proposed a novel method, GIST, for predicting 3D structures at a high resolution based on Auto-encoder with GAT. We convert the Hi-C into a heterogeneous graph, and GIST encodes the graph as a population of 3D conformations optimized by edge classification. We demonstrated that GIST produced chromosome structures consistent with FISH and outperformed existing 3D modeling methods. We illustrated the diversity of 3D structure predictions by evaluating the active and inactive X chromosome structures.

Lastly, we proposed a novel method, EnHiC, for predicting high-resolution Hi-C from low-resolution input based on a generative adversarial network. Inspired by non-negative matrix factorization, EnHiC fully exploits the unique properties of Hi-C and extracts rank-1 features from multi-scale low-resolution matrices to enhance the resolution. We demonstrated that EnHiC accurately and reliably enhanced the resolution of Hi-C and outperformed other GAN-based models. EnHiC-predicted high-resolution matrices facilitated the accurate detection of topologically associated domains and fine-scale chromatin interactions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Hi-C overview

The high-throughput chromosome conformation capture technologies, such as Hi-C [44] and its variants [19, 29, 59], have enabled us to detect genome-wide chromatin interactions at up to kilobase resolution, thereby providing an unprecedented opportunity to investigate the organizational mechanisms and principles of the three-dimensional (3D) genome architecture.

The Hi-C experiments assay chromatin contacts from a population of thousands to millions of cells denoted as bulk Hi-C. The bulk Hi-C contact frequency matrix presents the total contact frequencies from a cell population, where higher contact frequencies are assumed to represent closer spatial distance in the 3D space [44]. The bulk Hi-C data assembling DNA-DNA contacts forms as a symmetric non-negative square matrix and the values indicate the number of contacts between all pairs of genomic loci. The resolution of Hi-C is the length of one genomic bin of the contact matrix. The higher frequency indicates the more interaction between a pair of genomic loci in spatial.

1.2 3D genome architecture

The 3D genome architecture plays an important role on cell differentiation and development which modulates biological processes such as regulation, transcription, DNA replication, and epigenetic changes[15]. Recent years, the developments of the high-throughput chromosome conformation capture (Hi-C) techniques have enabled us to investigate the organizational principles of spatial structure (3D) of genome. Inferring the 3D structure from Hi-C data provides a comprehensive insight of chromosomes folding and compressing into the nucleus. The folding conformation of genome sequence is crucial to nuclear processes and functional properties[72], such as DNA replication and transcription.

The distinct organization levels are revealed by the experimental access to the 3D organization of the genome in the nuclear space. By analyzing the Hi-C data, the hierarchical organizations can be identified at different resolutions, from active/inactive chromosomal compartments (on a multi-Mb scale) [44], to topologically associated domains (TADs, from 10 kb to 1 Mb on average) [17] and fine-scale chromatin loops (less than 10 kb) [59, 48].

A large number of pioneering computational models have been developed to reconstruct the 3D genome structures from Hi-C. Most of them use bulk Hi-C data as input since bulk Hi-C contact frequency indicates the relative contact intensity for the distance inference in 3D space. On the contrary, the single-cell Hi-C based modeling methods are built on strong biological distance assumptions to deal with the low contact frequency and sparsity issue.

It is complex to portray the organization of chromatin since it is tightly folding and highly organized in the nucleus. With the development of bulk Hi-C experiments, recently several computational and statistical modeling tools have been published to infer the 3D genome structure

from Hi-C contacts. However, it is challenging to predict the 3D genome structure only given a Hi-C contact frequency matrix because of the noise, sparsity and the restriction of computational resource. Therefore, these tools often model the structure not only based on Hi-C contact matrix itself, but also based on different assumptions such as the polymer physic energy constraints and the observations from other orthogonal experiments. As a result, even using the same Hi-C data as input, different modeling tools can yield different 3D structures, thereby imposing challenges for the users to interpret the 3D structures organization and to understand the genome functions.

To tackle this problem, we developed effective evaluation metrics to measure the performance of these 3D chromatin structure modeling tools. To fairly compare these methods on the same input and template-based simulated data, we profiled the performance of existing bulk Hi-C based 3D genome structure modeling methods by testing on simulated and real bulk Hi-C data.

1.3 Modeling 3D structure from high-resolution Hi-C data

Models generate a consensus structure as a unique mean representation, alternatively, generate a set of structures as ensembles of Hi-C interactions. The major categories of these models are:

1. Multidimensional Scaling (MDS) based model transforms the Hi-C frequencies to the Euclidean space and then represents the information of distance matrix into 3D space. MDS based methods focus on the transformation and imputing the missing data for the distance matrix. For instance, ShRec3D[42] searches the shortest path between loci in the distance matrix to correct the bias and fill missing data.

2. Probability based model has a distribution to depict the Hi-C frequency between two loci. The coordinates of loci are parameters, estimated by Maximum Likelihood Estimation. For instance, the pastis[79] assumes the counts between loci follow a Poisson distribution parameterized by their spatial distance.

3. Restraint based model employs a objective function to quantify the restraints between counts and loci. The coordinates of loci are parameters, estimated by optimizing the objective function. For instance, The ChromSDE[97] estimates the coordinates by minimizing the L1/L2 losses and regularizations. The LorDG[76] formulates a bell-shape Lorentzian function as constraints.

4. Hybrid model composes of mixed algorithms together. For instance, GEM[99] infers the structure(s) based on energy and manifold learning. The MiniMDS[60] leverages a divide and merge strategy and MDS to infer the structure.

At relative low-resolution(hundred kilo-bases resolution), the diversity at local area is not significant hence this issue is compatible to the consensus assumption models. They perform better because of the computational efficiency. At high-resolution, the size of input Hi-C matrix increases quadratically. For instance, there are ~ 1300 loci in Chromosome X at 100kb, ~ 13000 loci at 10kb corresponding. The model costs more computational resources to search for the best solution. It consumes more memory/time if considering the Hi-C matrix as a dense matrix.

Moreover, the issue of diversity in the Hi-C is inevitable if we want to generate finer structures at high-resolution. Optimization gets hard to search the global solution as the restraints increasing, not to mention the conflicts in the restraints because of the diversity.

To alleviate these problems at high resolution, we proposed a self-supervised **G**raph-based network for **I**nference of Hi-C **S**patial s**T**ructure, named GIST, to estimate the a population of spatial conformations with proportions from Hi-C data using graph neural network in the framework of Auto-Encoder. Self-supervised learning is an unsupervised learning training model without extra information. The pseudolabels of samples are from prior knowledge. We treat the Hi-C matrix as a heterogeneous graph and there is no preliminary assumption or converting function between Hi-C frequency and Euclidean distance. In contract to the previous models, GIST is neural network which learns the parameters through a set of sub-graphs and estimates the entire structure in the prediction step. Therefore, the model gets optimized based on one sample/batch of sub-graph(s) instead of the entire Hi-C. To the best of our knowledge, this is the first study to apply graph neural network to 3D genome modeling.

1.4 Enhancement of Hi-C resolution

Studies of Hi-C data have revealed the multi-scale organization of the 3D genome, including active/inactive chromosomal compartments [44], topologically associated domains (TADs) [17], and fine-scale chromatin loops [59, 48]. Large-scale chromatin structures, such as compartments and TADs, can be identified from relatively low-resolution (50kb to 1Mb) Hi-C contact matrices. However, detecting fine-scale chromatin loops often requires high-resolution (i.e., 10 kb or finer) contact matrices. Moreover, fine-resolution Hi-C data are more compatible with other genomic and epigenomic data, and could therefore facilitate the interrogation of genome regulation and function.

However, high-resolution chromatin contact maps require costly, deep sequencing, and have been achieved in only a limited number of cell lines. For instance, a kilobase-resolution

Hi-C map of human lymphoblastoid GM12878 cells required five billion chromatin contacts [59]. Without sufficient sequencing depth, the observed Hi-C contact maps are often sparse and noisy, which imposes great computational challenges on the identification of chromatin loops between distal regulatory elements and their target genes. Therefore, computational approaches to enhance the resolution of Hi-C contact maps would greatly facilitate the investigation of the 3D genome at a finer scale, and are therefore in great demand.

Several pioneering works on predicting higher-resolution contact frequency matrices from low-resolution Hi-C data have emerged since 2018. The HiCPlus method [94] was the first attempt to enhance Hi-C data resolution with a convolutional neural network (CNN) by minimizing the L2 mean square error (MSE) loss function. Similar to the image super-resolution approach [96], HiCPlus extracts hidden features from high-resolution Hi-C matrices in the training process and then predicts high-resolution Hi-C matrices from low-resolution input data. Later, HiCNN [47] proposed the HiCNN model, which employs a more complex (with more than 14 layers) and efficient CNN model with residual learning by utilizing skip connections. However, both HiCPlus and HiCNN use the MSE loss; therefore, they are sensitive to outliers and would result in blurred output when the input Hi-C matrix is sparse.

More recently, several generative adversarial network (GAN) models, such as hicGAN [46], Deephic [22], and HiCSR [16], have been proposed to enhance Hi-C matrix resolution. The general GAN framework consists of two neural networks: a generator and a discriminator that contest with each other. In the training step, the generator learns to create a candidate to deceive the discriminator, while the discriminator learns to distinguish the generated candidate from the true data. First, hicGAN [46] adopts the SRGAN model [39] in image super-resolution to enhance resolution of Hi-C

matrices. The hicGAN model uses a skip-connection network as the generator and replaces the traditional pixel-wise MSE loss with a purely adversarial loss. As a result of minimizing the adversarial loss, hicGAN often misses fine-scale image details. Later, Deephic [22] proposed Deephic, a model similar to hicGAN. To recover fine-scale image details, Deephic uses a mixture loss function that consists of the MSE loss, total variation loss, perceptual loss, and adversarial loss. The perceptual loss component was derived from the VGG-type model [70]. However, this perceptual loss causes unwanted natural image textures in the output. Lastly, the HiCSR model [16] uses a skip-connection network as the generator and a CNN as the discriminator. Their loss function consists of the L1 mean absolute error (MAE) loss, feature loss, and adversarial loss. The feature loss was derived from a pre-trained model, which is a denoising autoencoder modified from an image restoration architecture [49].

The previously proposed models, hicGAN, Deephic, and HiCSR, have demonstrated the power of the GAN framework in predicting high-resolution Hi-C matrices. However, these models treat the Hi-C matrix as a one-channel image and their GAN networks are primarily built on image super-resolution models. As a result, their predictions often contain image artifacts and, therefore, do not accurately represent the underlying chromatin interaction features of the Hi-C data.

To tackle this problem, we developed a new GAN-based model, EnHiC, to enhance the resolution of Hi-C contact frequency matrices. Specifically, we propose a novel convolutional layer (the *Decomposition & Reconstruction Block*, see Methods) that accounts for the non-negative and symmetric properties of Hi-C matrices. In our GAN framework, the generator extracts rank-1 features from multiple scales of low-resolution matrices and predicts the high-resolution matrix via a series of sub-pixel CNN layers [68]. Accordingly, the discriminator decomposes a high-resolution

Hi-C matrix into multiple lower-resolution matrices and extracts the corresponding rank-1 features to determine whether the high-resolution matrix is derived from the generator or the true data.

We evaluated the performance of our EnHiC model using published Hi-C datasets in three human cell lines: GM12878 (lymphoblastoid cells), IMR90 (lung fibroblast cells), and K562 (leukemia cells) [59]. We demonstrated that EnHiC accurately enhanced the resolution of Hi-C data and achieved high similarity scores with respect to the true high-resolution data, outperforming previously proposed GAN-based models. Using the model trained in one cell type, EnHiC effectively enhanced the resolution of insufficient sequenced Hi-C data in other cell types. In addition, using the EnHiC-enhanced data, we successfully recovered Hi-C-specific features, such as TADs and significant chromatin interactions.

1.5 Roadmap

In this chapter, we introduced the bulk Hi-C data, and the motivations of 1) profiling 3D modeling methods, 2) developing the 3D modeling method at high-resolution Hi-C data, and 3) enhancing the resolution of Hi-C matrix. In the following chapters, We introduced these 3 parts about studying the spatial organizations of chromosomes.

In the Chapter 2, we made an evaluation of chromatin structure modeling models. We first made a comprehensive survey of publicly available Hi-C based chromatin 3D structure modeling methods in the Section 2.1. Then we reported the evaluations of methods in the simulation (Section 2.2.1 and 2.2.2) and real (Section 2.2.3 and 2.2.4) datasets as well as the runtime performance (the Section 2.3). The details about measurement methods we developed are introduced in the Appendix A: simulation method, alignment methods, etc.

In the Chapter 3, we introduced a model modeling 3D structures by using graph attention network based on the framework of Auto-Encoder in the Section 3.1. In the Section 3.2, we validated our model based on independent fluorescence in situ hybridization (FISH) data and compared with the top 5 methods from Chapter 2 at high-resolution (10kb) . Moreover, we evaluated the diversity of structures in the chromosome X using Chromosome Xa and Xi FISH data.

In the Chapter 4, we introduced a novel method, EnHiC, for predicting high-resolution Hi-C matrices from low-resolution in the Section 4.1. We validated our EnHiC model and compared with previously proposed GAN-based models in the Section 4.2 using published Hi-C datasets in three human cell lines: GM12878 (lymphoblastoid cells), IMR90 (lung fibroblast cells), and K56252 (leukemia cells). Moreover, we demonstrated that EnHiC-predicted matrices facilitated more accurate detection of TADs and fine-scale chromatin interactions.

Finally, the discussion and conclusions are in the Chapter 5.

Chapter 2

Comparison of 3D chromatin structure modeling methods

2.1 3D chromatin structure modeling methods

We summarized existing Hi-C-based chromatin 3D structure modeling tools. To the best of our knowledge, we found 22 publicly available bulk Hi-C-based modeling tools AutoChrom3D[57], BACH[23], Chromosome3D[2], ChromSDE[97], Chrom3D[56], Duan MDS/ MetricMDS[19], NMDS[6], FisHiCal[67], GEM[99], Gen3D[53], HSA0/ HSA1[100], InfMod3DGen[84], LorDG[76], MOGEN[75], MCMC5C[62], miniMDS[60], Pastis-PM1/Pastis-PM2[79], PGS[24], ShRec3D[42], TADbit[64], tREX/tPAM[54], 3D-GNOME[73, 74], and 4 single-cell Hi-C-based modeling tools ISDHiC[10], Nuc-dynamics[71], MBO[55], SIMBA3D[61]. In Table 2.1, we summarized and compared the modeling pipeline of these 26 bulk Hi-C based methods. We do not discuss the four single-cell Hi-C based methods because their computational pipelines and assumptions are different

from bulk Hi-C based methods. In this work, we focused on 3D modeling methods for bulk Hi-C data and thoroughly evaluated 13 methods(15 results, including two optimizations) that provide publicly available software tools:

BACH [23], which stands for Bayesian 3D constructor for Hi-C data, is a consensus method that assumes the local genomic region of interest exhibits a consensus 3D chromosomal structure in a cell population, and contact counts follow a Poisson distribution. BACH refines spatial structure and parameters by Gibbs sampling, adaptive rejection sampling, and hybrid Monte Carlo approaches.

Chromosome3D [2] is a population method that is based on a distance geometry simulated annealing method on the Crystallography and NMR System (CNS) platform. It makes a strong assumption to simulate the Hi-C contact frequencies as the protein carbon-alpha–carbon-alpha distances in order to use them as restraints for the overall optimization process on the CNS platform.

MCMC5C is a probabilistic model (normal distribution) based on a contact-to-distance transfer function and generates spatial structures for both 5C and Hi-C data via the Markov Chain Monte Carlo (MCMC) method [62]. Specifically, the Metropolis-Hastings algorithm is used to sample the structures. Since MCMC5C is a probabilistic model, the output is an ensemble of structures. The structure with the highest likelihood is defined as the best one in the ensemble.

ChromSDE [97] is a consensus method that adds an additional regularization to the original MDS object function to maximize the distance between beads that do not have any observed contacts. Then it respectively relaxes the two types of original problems to two kinds of semidefinite programming (SDP) problems and solves the SDPs by a partial proximal point method. Finally, ChromSDE uses gradient descent searching and Newton methods to locally search the solution to the original non-convex problem with the SDP result as an initialized structure.

GEM [99] is a population method that combines a t-SNE based manifold learning algorithm and polymer energy constraints. Instead of translating contact frequencies to Euclidean distances as commonly used in other methods, GEM measures the Kullback–Leibler (KL) divergence between high dimensional distribution from Hi-C data and the 3D coordinates.

HSA, including two variants HSA0 and HSA1, predict chromatin structures by integrating one or more Hi-C contact maps with different restriction enzymes [100]. HSA utilizes the generalized linear model with an iterative algorithm, which combines Hamiltonian dynamics with simulated annealing. The difference between HSA0 and HSA1 is that HSA0 uses the general linear model (GLM) under a Poisson function, while HSA1 adds an additional constraint of a Markov Chain to restrict the distances of adjacent beads. HSA1 provides an option for Markov modeling when the contact maps are sparse and of low sequencing coverage.

LorDG is a population method that uses a Lorentzian function to quantify the distance restraints [76]. LorDG takes advantage of the derivatives characteristic of the Lorentzian function, which can reduce the influence of the inconsistent contacts on the score function and, at the same time, filter noise in the contact matrix.

Multi-dimensional scaling, also referred to as metric MDS, is a classic approach to estimating coordinates of points from pairwise Euclidean distance matrix [35]. [19] adapted the metric MDS approach to chromatin structure modeling and proposed a consensus method by converting the contact frequencies in a Hi-C map to a pairwise Euclidean distance matrix and then inferring the spatial structure via MDS.

MiniMDS [60] is a consensus method based on MDS. It constructs a hierarchical model to reduce the complexity of MDS when the size of the Hi-C contact matrix is large. The idea of

miniMDS is straightforward: it divides the genome into non-overlapping TAD regions, estimates the structure of each TAD separately, and then combines these sub-structures together.

Non-metric MDS[34, 6](NMDS) proposed to use the NMDS approach to inter chromatin structures from Hi-C. Specifically, NMDS assumes that if the loci with more observed Hi-C contacts are physically closer in 3D space than the loci with less observed contacts. Thus, NMDS uses a general decreasing function of the contact counts to replace the Euclidean wish distance in MDS.

PASTIS is a consensus method that proposes to cast the problem of structure inference as a maximum likelihood problem [79]. Briefly, it defines a probabilistic model where contact counts are parameterized by the 3D structure which it intends to infer. Similar to the BACH method [23], PASTIS assumes that the contact counts are independent Poisson random variables. PASTIS finds the optimal 3D structures by maximizing the log-likelihood of the Poisson model via the Limited-memory BFGS (L-BFGS) algorithm [45]. The results are called Pastis-PM1 and Pastis-PM2 in our experiments. The Pastis-PM1 uses a default transfer function, whereas the second method (Pastis-PM2) optimizes the parametric family in transfer function automatically.

ShRec3D [42] is a consensus method using a classical multi-dimensional scaling algorithm to make a dimensional deduction from a high dimensional Gram distance matrix to the 3D coordinate space. In addition, ShRec3D leverages the shortest path searching to update the distance matrix to reduce noise and fill missing data.

TADbit [64] uses a consensus method based on the Integrative Modeling Platform (IMP) to model chromatin 3D structures. The IMP method first models a pool of structures trying to satisfy as many distance constraints as possible from Hi-C data and polymer physics restraints and then uses the Markov Cluster method to identify a consensus structure.

Method	**Normalization**[1]	**Modeling** [2] **Unit**	f_{HiC}[3] **F Usage**	$F \sim D^{\alpha}$	f_{Bio}[4]	f_{Phy}[5]	**Assumption**[6]	**Sampling**[7] **Approach**	**Output**[8]	**whole**[9] **genome**	**Test**[10]
ShRec3D	-	FB	F-D	$\alpha = -1$	-	-	Consensus	SDP	1	✓	✓
Duan MDS	✓	FB	F-D	$\alpha = -1/3$	✓	✓	Consensus	IPOPT	1	✓	-
Metric MDS	✓	FB	F-D	$\alpha = -1/3$	-	-	Consensus	L-BFGS	1	-	✓
NMDS	✓	FB	O	-	-	-	Consensus	L-BFGS/IsotonicReg	1	-	✓
PM1	✓	FB	P	$\alpha = -1/3$	-	-	Consensus	L-BFGS	1	✓	✓
PM2	✓	FB	P	Searching	-	-	Consensus	L-BFGS	1	✓	✓
ChromSDE	-	FB	F-D	Searching	-	-	Consensus	GDS/QN SDP	1	-	✓
miniMDS	-	FB	F-D	$\alpha = -0.25$	-	-	Consensus	SMACOF	1	✓	✓
BACH	✓	FB	P	Searching	-	-	Consensus	MCMC	1	-	✓
BACH-Mix	✓	FB	P	Searching	-	-	Population	MCMC	≥ 1	✓	-
HSA0	✓	FB	P	Searching	-	-	Consensus	SA	1	-	✓
HSA1	✓	FB	P	Searching	-	✓	Consensus	SA	1	-	✓
LorDG	-	FB	F-D	Searching	-	-	Population	GAS	≥ 1	✓	✓
MOGEN	-	FB	O	-	✓	-	Population	GAS	≥ 1	✓	-
Chromosome3D	-	FB	F-D	$\alpha = -0.5$ Searching	-	✓	Population	CNS,DGSA	≥ 1	-	✓

Table 2.1 – continued from previous page

Method	**Normalization**	**Modeling Unit**	f_{HiC} **F Usage**	f_{HiC} $F \sim D^{\alpha}$	f_{Bio}	f_{Phy}	**Assumption**	**Sampling Approach**	**Output**	**whole genome**	**Test**
MCMC5C	-	FB	P	$\alpha = -0.5$ Searching	-	-	Population	MCMC	1	-	✓
TADbit	✓	FB	F-D	N/A	✓	✓	Population	IMP,MCMC	≥ 1	-	✓
InfMod3DGen	-	FB	P	Searching	-	✓	Population	GAS,EM	≥ 1	-	-
GEM	-	FB	P	-	-	✓	Population	GD	≥ 1	-	✓
3D-GNOME	✓	FB	F-D	$\alpha = -0.6$	✓	✓	Consensus	MCMC	1	-	-
FisHiCal	✓	FB	F-D	Searching	✓	-	Consensus	SMACOF	1	-	-
AutoChrom3D	✓	FB	F-D	-	✓	-	Consensus	N/A	1	-	-
Gen3D	-	FB	O	-	✓	-	Consensus	Adaption,SA	1	-	-
tREX	✓	FB	F-D	Searching	-	-	Population	MCMC	≥ 1	-	-
Chrom3D	✓	TAD	P	-	✓	-	Consensus	SA	1	✓	-
PGS	✓	TAD	P	-	✓	✓	Population	IMP	≥ 1	✓	-

Table 2.1: **Summary of 3D chromatin modeling methods**

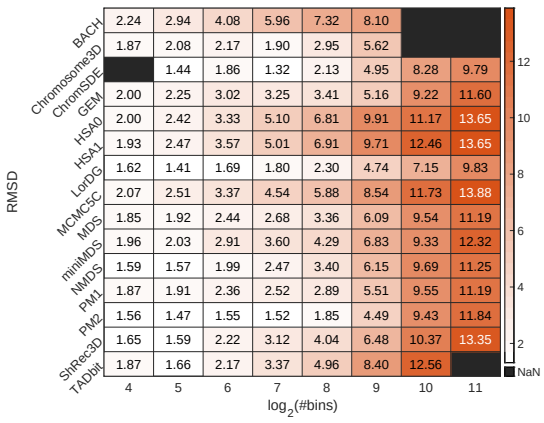
[1] Whether contains built-in normalization; [2] The modeling unit of output structure; [3] How the method used the contact frequency data, and how to find the parameter for frequency-expected distance conversion if exists; [4,5] Whether the method uses independent constraints biological observations (f_{Bio}) and polymer physic energy restriction (f_{Phy}); [6] The assumption in the method; [7] The sampling methods used to find the satisfied solution; [8] The number of the output modeling structures; [9] Whether the method can be used for whole genome structure modeling according to their papers; [10] Whether tested in this comparative study.

2.2 Evaluation

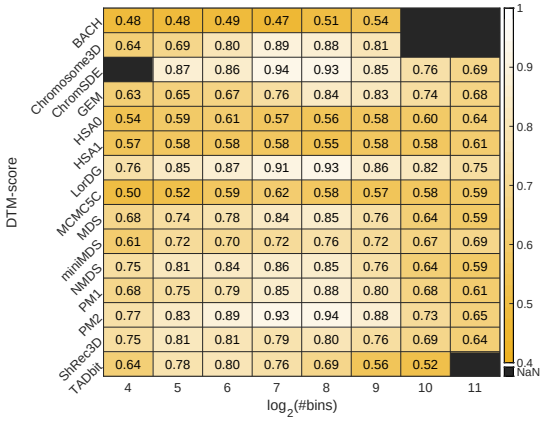
We conducted a comprehensive evaluation and comparison of these modeling methods on both simulated and real data. The benchmark datasets are introduced in Appendix A.1. We proposed a novel simulation approach (Appendix A.2) that follows the same physical model of the real data, in which the parameters are estimated from the real Hi-C data.

The simulated data is designed by a combination of the haploid template structures compiled by GSE80280[71], and single-cell and bulk Hi-C data from the same series. The real Hi-C data is from GM12878 reported by GSE63525[59]. For the simulated data, the ground-truth structures are known, whereas, for the real data, such information is unavailable. In total, 15 modeling tools were comprehensively evaluated on both simulated and real data. In addition, we identified local genomic features from the structures and evaluated their consistency with independent experimental data, including Lamina-associated domain (LAD) data and fluorescence in situ hybridization (FISH) data.

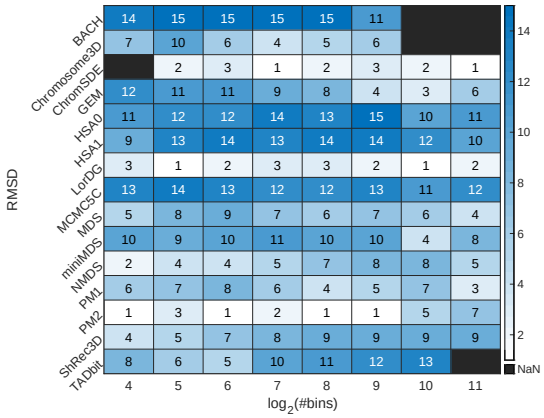
We developed two metrics for structure alignment: weighted-RMSD and Dynamic Template Modeling score (DTM-score), in the Appendix A.3, which evaluated structures quantitatively and robustly in the Section 2.2.1, and 2.2.2. These novel computational tools enabled us to evaluate the performance of the modeling methods from various perspectives, including the length of the chromatin, the sparsity of the Hi-C data, the resolution of the experiments, the consistency with independent experimental data, as well as runtime and memory complexity.


RMSD
BACH 2.24 2.94 4.08 5.96 7.32 8.10
Chromosome3D 1.87 2.08 2.17 1.90 2.95 5.62
ChromSDE 1.44 1.86 1.32 2.13 4.95 8.28 9.79
GEM 2.00 2.25 3.02 3.25 3.41 5.16 9.22 11.60
HSA0 2.00 2.42 3.33 5.10 6.81 9.91 11.17 13.65
HSA1 1.93 2.47 3.57 5.01 6.91 9.71 12.46 13.65
LorDG 1.62 1.41 1.69 1.80 2.30 4.74 7.15 9.83
MCMC5C 2.07 2.51 3.37 4.54 5.88 8.54 11.73 13.88
MDS 1.85 1.92 2.44 2.68 3.36 6.09 9.54 11.19
miniMDS 1.96 2.03 2.91 3.60 4.29 6.83 9.33 12.32
NMDS 1.59 1.57 1.99 2.47 3.40 6.15 9.69 11.25
PM1 1.87 1.91 2.36 2.52 2.89 5.51 9.55 11.19
PM2 1.56 1.47 1.55 1.52 1.85 4.49 9.43 11.84
ShRec3D 1.65 1.59 2.22 3.12 4.04 6.48 10.37 13.35
TADbit 1.87 1.66 2.17 3.37 4.96 8.40 12.56
4 5 6 7 8 9 10 11
log2(#bins)
12 10 8 6 4 2
NaN

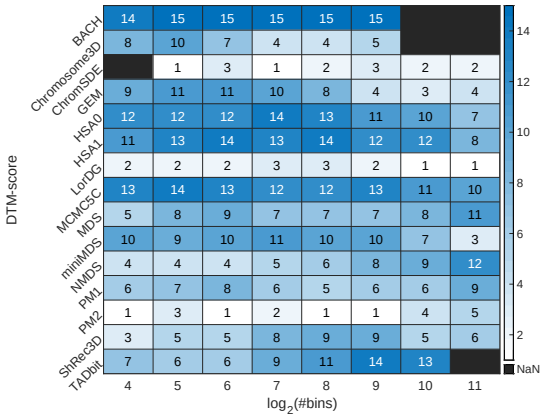
(a)


DTM-score
BACH 0.48 0.48 0.49 0.47 0.51 0.54
Chromosome3D 0.64 0.69 0.80 0.89 0.88 0.81
ChromSDE 0.87 0.86 0.94 0.93 0.85 0.76 0.69
GEM 0.63 0.65 0.67 0.76 0.84 0.83 0.74 0.68
HSA0 0.54 0.59 0.61 0.57 0.56 0.58 0.60 0.64
HSA1 0.57 0.58 0.58 0.58 0.55 0.58 0.58 0.61
LorDG 0.76 0.85 0.87 0.91 0.93 0.86 0.82 0.75
MCMC5C 0.50 0.52 0.59 0.62 0.58 0.57 0.58 0.59
MDS 0.68 0.74 0.78 0.84 0.85 0.76 0.64 0.59
miniMDS 0.61 0.72 0.70 0.72 0.76 0.72 0.67 0.69
NMDS 0.75 0.81 0.84 0.86 0.85 0.76 0.64 0.59
PM1 0.68 0.75 0.79 0.85 0.88 0.80 0.68 0.61
PM2 0.77 0.83 0.89 0.93 0.94 0.88 0.73 0.65
ShRec3D 0.75 0.81 0.81 0.79 0.80 0.76 0.69 0.64
TADbit 0.64 0.78 0.80 0.76 0.69 0.56 0.52
4 5 6 7 8 9 10 11
log2(#bins)
1 0.9 0.8 0.7 0.6 0.5 0.4
NaN

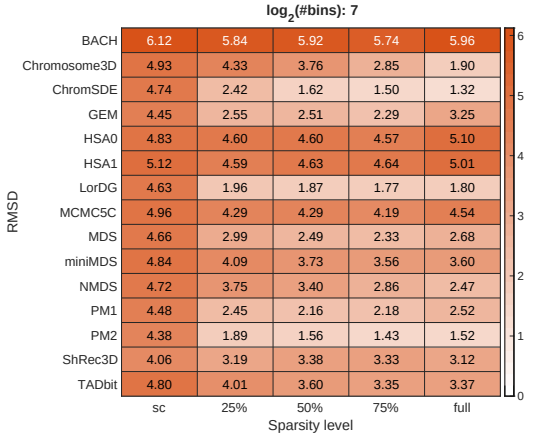
(b)


RMSD
BACH 14 15 15 15 15 11
Chromosome3D 7 10 6 4 5 6
ChromSDE 2 3 1 2 3 2 1
GEM 12 11 11 9 8 4 3 6
HSA0 11 12 12 14 13 15 10 11
HSA1 9 13 14 13 14 14 12 10
LorDG 3 1 2 3 3 2 1 2
MCMC5C 13 14 13 12 12 13 11 12
MDS 5 8 9 7 6 7 6 4
miniMDS 10 9 10 11 10 10 4 8
NMDS 2 4 4 5 7 8 8 5
PM1 6 7 8 6 4 5 7 3
PM2 1 3 1 2 1 1 5 7
ShRec3D 4 5 7 8 9 9 9 9
TADbit 8 6 5 10 11 12 13
4 5 6 7 8 9 10 11
log2(#bins)
14 12 10 8 6 4 2
NaN

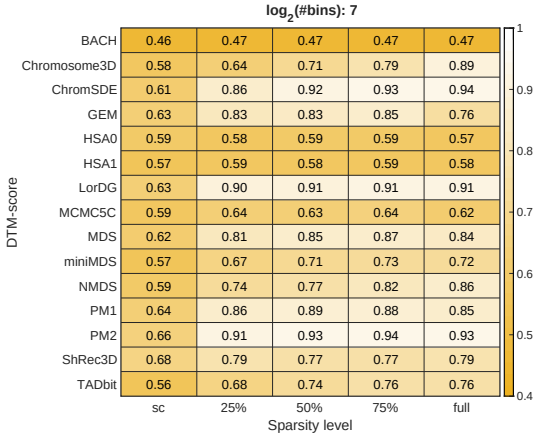
(c)


DTM-score
BACH 14 15 15 15 15 15
Chromosome3D 8 10 7 4 4 5
ChromSDE 1 3 1 2 3 2 2
GEM 9 11 11 10 8 4 3 4
HSA0 12 12 12 14 13 11 10 7
HSA1 11 13 14 13 14 12 12 8
LorDG 2 2 2 3 3 2 1 1
MCMC5C 13 14 13 12 12 13 11 10
MDS 5 8 9 7 7 7 8 11
miniMDS 10 9 10 11 10 10 7 3
NMDS 4 4 4 5 6 8 9 12
PM1 6 7 8 6 5 6 6 9
PM2 1 3 1 2 1 1 4 5
ShRec3D 3 5 5 8 9 9 5 6
TADbit 7 6 6 9 11 14 13
4 5 6 7 8 9 10 11
log2(#bins)
14 12 10 8 6 4 2
NaN

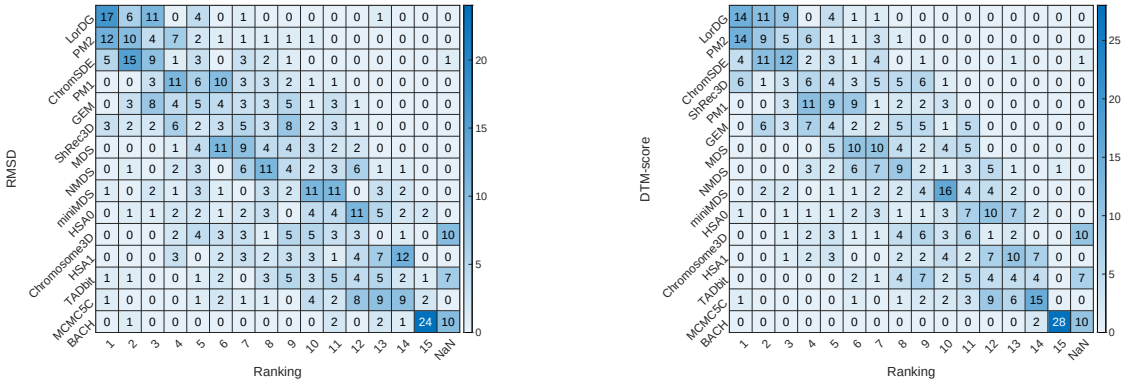
(d)


log2(#bins): 7
RMSD
BACH 6.12 5.84 5.92 5.74 5.96
Chromosome3D 4.93 4.33 3.76 2.85 1.90
ChromSDE 4.74 2.42 1.62 1.50 1.32
GEM 4.45 2.55 2.51 2.29 3.25
HSA0 4.83 4.60 4.60 4.57 5.10
HSA1 5.12 4.59 4.63 4.64 5.01
LorDG 4.63 1.96 1.87 1.77 1.80
MCMC5C 4.96 4.29 4.29 4.19 4.54
MDS 4.66 2.99 2.49 2.33 2.68
miniMDS 4.84 4.09 3.73 3.56 3.60
NMDS 4.72 3.75 3.40 2.86 2.47
PM1 4.48 2.45 2.16 2.18 2.52
PM2 4.38 1.89 1.56 1.43 1.52
ShRec3D 4.06 3.19 3.38 3.33 3.12
TADbit 4.80 4.01 3.60 3.35 3.37
sc 25% 50% 75% full
Sparsity level
6 5 4 3 2 1 0

(e)


log2(#bins): 7
DTM-score
BACH 0.46 0.47 0.47 0.47 0.47
Chromosome3D 0.58 0.64 0.71 0.79 0.89
ChromSDE 0.61 0.86 0.92 0.93 0.94
GEM 0.63 0.83 0.83 0.85 0.76
HSA0 0.59 0.58 0.59 0.59 0.57
HSA1 0.57 0.59 0.58 0.59 0.58
LorDG 0.63 0.90 0.91 0.91 0.91
MCMC5C 0.59 0.64 0.63 0.64 0.62
MDS 0.62 0.81 0.85 0.87 0.84
miniMDS 0.57 0.67 0.71 0.73 0.72
NMDS 0.59 0.74 0.77 0.82 0.86
PM1 0.64 0.86 0.89 0.88 0.85
PM2 0.66 0.91 0.93 0.94 0.93
ShRec3D 0.68 0.79 0.77 0.77 0.79
TADbit 0.56 0.68 0.74 0.76 0.76
sc 25% 50% 75% full
Sparsity level
1 0.9 0.8 0.7 0.6 0.5 0.4

(f)



(g) (h)

Figure 2.1: **Evaluation of 15 methods on simulation datasets. (a-b)** Heatmap of weighted RMSD values (a) and DTM-scores (b) of predicted 3D structures using simulated bulk Hi-C contact matrices. All methods are tested on 20 chromosomes at eight different resolutions (from 8 Mb to 100 kb), separately. The rows represent different methods; and the columns represent the structure sizes in $\log_2$ scale. The color shows the average weighted-RMSD values (a) or DTM-scores (b) between the predicted structures and ground truth structure. **(c-d)** The number in each cell indicates the column-wise rank of each method among the 15 compared ones. **(e-f)** Hierarchical clustering of weighted RMSD (e) and DTM-scores (f) of predicted 3D structures across various sparsity levels. The length of chromosomes around 2^7 is included. Simulated Hi-C contact matrices were generated at five different sparsity levels: bulk Hi-C experiment, single-cell Hi-C experiment, and three intermediate sparsity levels (25%, 50%, and 75%) in between. **(g-h)** Ranking in total. The row is the methods, and the column is the rank (15 methods in total). The value means the number of ranks one method got. The first value 17 represents LorDG got 17 times rank 1 for different lengths and sparsity. The last column is the number of no results.

2.2.1 Scalability with regard to structure length

The length (the number of beads) of the chromatin structure is one major factor that affects the performance of 3D modeling methods. Both the chromosome size and the resolution of the Hi-C contact frequency matrix determine the length of the resulting structure. Longer structures are computationally challenging to solve. In addition, fine-resolution Hi-C contact matrices are sparse, which are more difficult to infer accurate 3D structures. In our simulation studies, we generated Hi-C contact matrices for all 20 mouse chromosomes at eight different resolutions (8 Mb, 4 Mb, 2 Mb, 1 Mb, 800 kb, 400 kb, 200 kb, and 100 kb). The length of tested structures (and the resulting contact matrices) varied from ~ 20 to ~ 2000.

To evaluate the 3D modeling methods, we proposed two novel similarity measurements, namely the weighted-RMSD measure and the Dynamic Template Modeling score (DTM-score), for comparing the predicted chromosomal structure against the ground truth structure (Appendix A.3). First, the weighted-RMSD approach produces a weighted alignment of two structures and returns the weighted average distance between them. A smaller weighted-RMSD value indicates more similarities between the two structures. In practice, we normalized the weighted-RMSD value by the scale of the template structure so that the values could be compared across different template structures. Second, the DTM-score method is inspired by TM-align [95], a widely used structural similarity score for protein structure comparison. Briefly, DTM-score measures the similarity between two chromatin structures using the optimal local alignments. Unlike weighted-RMSD, the DTM-score is designed scale-free and ranges from 0 to 1, where the larger DTM-score indicates the better alignment.

We evaluated the modeling results from the 15 modeling methods using the weighted-RMSD values between the modeling structures and the ground truth structures on the simulated data (20 chromosomes $\times$ 8 resolutions $\times$ 5 sparsity levels). We summarized the average weighted-RMSD values as well as the corresponding rank for each method at various structure lengths (Fig. 2.1a, c). Consistent with our expectations, the longer the structures, the larger the weighted-RMSD scores. Overall, no method consistently outperformed the others. We observed that Pastis-PM2 ranks among the top for shorter structures(Fig. 2.1e), whereas LorDG performs the best for longer ones.

Then we assessed the performance of these methods using the DTM-scores (Fig. 2.1e, f). Consistent with the results obtained by weighted-RMSD, Pastis-PM2 was the best performer for shorter structures, followed by ChromSDE, whereas LorDG outperformed other methods for longer structures. For the majority of the methods, the performance tends to be better for mid-range structure length.

2.2.2 Robustness with regard to sparsity level

Due to the limit of sequencing depth, a typical Hi-C experiment might only capture a small subset of all possible chromatin contacts. Consequently, the observed Hi-C contact frequency matrix is often sparse with a high level of missing information. To test the capability of the 3D modeling methods on handling missing information, we simulated a series of Hi-C contact matrices with different degrees of sparsity (section A.2.1). Briefly, given a ground truth structure, the simulated Hi-C matrix is the combination of a signal matrix and a noise matrix. We adjusted the sparsity level of the signal matrix to simulate various sparsity levels from the one of a perfect bulk Hi-C experiment (referred to as bulk) to the one of a single-cell Hi-C experiment (referred to as single-cell), with three intermediate sparsity levels (25%, 50%, and 75%) in between.

As the sparsity level increased from bulk Hi-C level to single-cell level, the performance of all methods declined in both weighted-RMSD measures and DTM-score measures. At the single-cell degree of sparsity, the DTM-scores of all predicted structures are small, suggesting that these 3D modeling methods are specifically designed for bulk Hi-C data and therefore could not produce satisfactory structural predictions at single-cell settings.

To further illustrate the impact of sparsity level on the performance of the modeling methods, we grouped the chromosomal structures by their length in log scale and computed the average weighted-RMSD and DTM-score values for each method across various sparsity levels for each method within each structure length group (Fig. 2.1e, f).

Overall we observed that across all length groups, the majority of these methods showed improvements of weighted-RMSDs and DTM-scores as the sparsity level decreases, except for BACH, MCMC5C, HSA0, and HSA1, no benefit from the additional information in less sparse contact matrices (in Fig. 2.1f).

Summarizing the results obtained overall simulated datasets, we ranked the 15 methods based on their weighted-RMSD and DTM-score performance on both structure length and sparsity simultaneously (Fig. 2.1g, h). Overall, both weighted-RMSD and DTM-score measures suggest a similar ranking. LorDG is the most frequently top-ranked method, followed by a close runner-up, Pastis-PM2. Following them, ChromSDE, ShRec3D, Pastis-PM1, and GEM are the next groups that performed well in certain situations. In addition to evaluating the overall structural similarity between the modeled structures and the ground-truth structures, we further evaluated the local properties of the modeled structures by checking their consistency with independent, experimental data, including LAD data (GSE17051) and human FISH data (GM12878) [5].

2.2.3 Fine-scale local genomic feature detection (Consistency with LAD data)

The nuclear lamina provides anchoring sites for large genomic segments named lamina-associated domains (LADs). LAD regions are confined to either the nuclear membrane or the nuclear periphery. Since LADs data in GSE17051 is in mm9 coordinates, we converted it to mm10 by liftover and binned it at the same resolution as Hi-C data (100kb). LAD data is labeled as separate fragments in the sequence, and we denote each fragment as a sub-region of domains. Since LAD sub-regions are confined to underlie the inner nuclear membrane, we expect the Euclidean distance between LAD sub-regions in space is irrelative with the distance genomics between corresponding sub-regions. To identify this pattern, we used the rank-correlation, Kendall, to evaluate the average distance matrices of LAD sub-regions in the genomic distance and in space for each structure. We estimated the distribution of the Kendall correlation coefficients from all ground truth structures. The alternative hypothesis is the Kendall rank correlation coefficients from the 15 modeling methods are larger than ground truth (details in the Appendix A.4).

Table 2.2 shows the results on 4 chromosomes(1,9,X,19 with decreasing length of chromosomes) on the bulk Hi-C from haploid mESCs[71] at 100kb resolution (no outputs of BACH and TADbit). The Kendall rank correlation illustrates the linear correlation coefficient between sequence and spatial structure and *p-value* from t-test(right tail, $\alpha = 0.05$) indicates whether the Kendall correlation coefficients are significantly larger than the ground truth. In the table, most structures accepted the null hypothesis. However, the Kendall correlation coefficients of structures from MCMC5C, HSA0, HSA1, and mininMDS are significantly larger than the ground truth. In other words, the structure is more like a floating chain rather than compressing intense conformation. Therefore, it is unlikely to observe the folds of LAD regions in these spatial structures.

Method	Kendall Correlation	p-value	h
MCMC5C	0.9541	6.82E-09	1
HSA1	0.86869	3.66E-07	1
HSA0	0.82132	2.76E-06	1
miniMDS	0.64652	0.004546	1
LorDG	0.47672	0.42473	0
Chromosome3D	0.48216	0.43326	0
ChromSDE	0.3807	0.85002	0
GEM	0.28036	0.98928	0
PM2	0.23954	0.99793	0
PM1	0.22035	0.99911	0
MDS	0.19683	0.99972	0
NMDS	0.17057	0.99988	0
ShRec3D	0.17424	0.99989	0
BACH	NaN	NaN	NaN
TADbit	NaN	NaN	NaN

Table 2.2: **Kendall rank correlation and *p*-value** for LAD Evaluation

2.2.4 Fine-scale local genomic feature detection (Validation of 3D-FISH data)

Fluorescence in situ hybridization (FISH) experiment reveals the spatial distance between some DNA fragments in the live cell (Table 2.3). We tested the consistency between the relative order of the distance in the modeled structures by different methods and the one revealed in human FISH data from [59]. For each of chromosomes 11, 13, 14, and 17, it is labeled as 'T':True if the orders are consistent, otherwise 'F':False (Appendix A.3).

Chromosome	**L1 (position, Mb)**	**L2 (position, Mb)**	**L3 (position, Mb)**
17	66.76-66.79	67.22-67.25	67.68-67.71
14	71.60-71.63	72.20-72.23	72.80-72.83
11	130.72-130.75	130.29-130.32	129.86-129.89
13	86.37-86.40	85.46-85.49	84.55-84.58

Table 2.3: **3D-FISH loci.** The 3D-FISH coordinates[59] have confirmed that the 3D-distance between L1 and L2 (the two peak loci) was consistently shorter than the 3D-distance between L2 and L3 (one peak locus and one control locus).

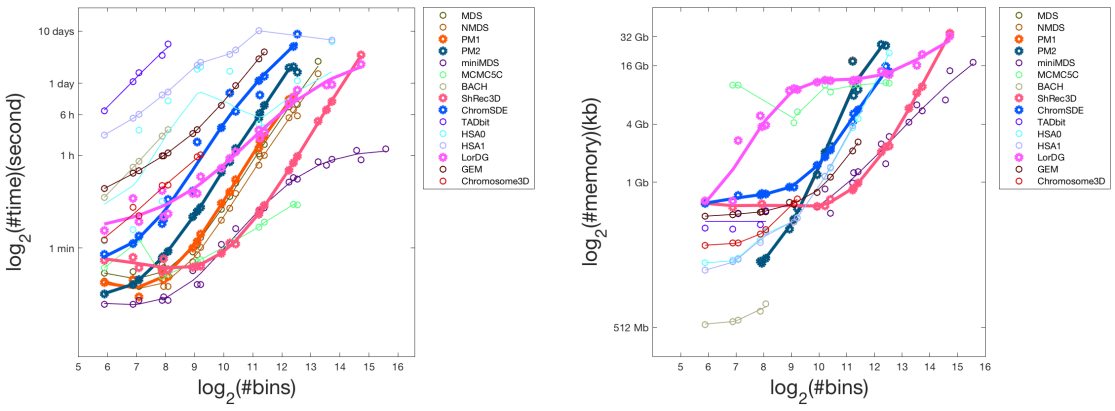
As Shown in Table 2.4, the structures by MDS, NMDS, Pastis-PM1, Pastis-PM2, ShRec3D, ChromSDE, HSA1, and LorDG are perfectly consistent with the FISH data. The structures modeled by miniMDS break the upstream and downstream relation by scalable modeling and MCMC5C outputs initialization (helix). Thus their structures are not consistent with the FISH data. GEM works perfectly under the 50kb resolution but does not work under the 25kb resolution. BACH, TADbit, and Chromosome3D work in neither 25kb nor 50kb resolutions.

Tool Name	Chr11	Chr13	Chr14	Chr17
MDS	T/T	T/T	T/T	T/T
NMDS	T/T	T/T	T/T	T/T
Pastis-PM1	T/T	T/T	T/T	T/T
Pastis-PM2	T/T	T/T	T/T	T/T
miniMDS	F/F	F/T	F/F	F/T
MCMC5C	F/F	F/F	F/F	F/F
BACH	-/-	-/-	-/-	-/-
ShRec3D	T/T	T/T	T/T	T/T
ChromSDE	T/T	T/T	T/T	T/T
TADbit	-/-	-/-	-/-	-/-
HSA0	T/T	F/T	T/T	T/T
HSA1	T/T	T/T	T/T	T/T
LorDG	T/T	T/T	T/T	T/T
GEM	T/-	T/-	T/-	T/-
Chromosome3D	-/-	F/-	-/-	-/-

Table 2.4: **Consistency with FISH data (50kb/25kb)**

2.3 Runtime and memory

We further tested the practical runtime and memory usage of the 15 methods. Chromosomes 2, 10, and 19 are used in this experiment because their length ranges over 2^6 to 2^{16} beads under the resolution range from 5kb to 1Mb.



(a) (b)

Figure 2.2: **Computational complexity of 15 tested methods.** (a) Runtime and (b) consumption of memory are recorded for each of the 15 tested methods. The colored curves represent the fitted regression lines. The bold curves are performances of the methods which are the top 5 in similarity test of simulated datasets.

The runtime and memory consumption of the 15 methods are shown in Fig. 2.2, in which the top 5 ranked methods on the simulated data, LorDG, Pastis-PM2, ChromSDE, ShRec3D, and Pastis-PM1, are in bold (Fig. 2.1g,h). Most methods finish running in 1 hour when the number of beads is less than 2000. In general, MCMC-based approaches cost much more time than others. Most methods show a close-to-linear trend in terms of runtime and memory usage in terms of the

log scale of the number of beads, which implies that the actual runtime and memory increases exponentially with respect to the length. Among all the 15 methods, miniMDS is the fastest one, BACH costs the most memory efficient one, and it can only run for a small range of length.

Most methods finished work within 1 day when structure 1000 beads. Most of them cannot finish if structure larger than $\sim$ 20000 beads within 2 weeks. The only method that works at the 5kb resolution is miniMDS, whereas ShRec3D and LorDG are the two that can reach the $\sim$ 16000 beads.

Chapter 3

Modeling 3D chromatin structure using a graph attention network

There is no explicit mechanism to transform the Hi-C matrix to Euclidean distance matrix or reverse. However, a graph is a bridge to connect both. In both matrices, the values at i-th row and j-th column represent the weight of an edge between genomic locus i and genomic locus j. The only difference is that the value in the Hi-C matrix is a similarity score, but the one in the Euclidean matrix is distance. The two graphs from Hi-C or Euclidean distance present the same genomics structures in the real world; hence, the graphs are matched (vertex i in the graph from Hi-C is paired with vertex i in the one from Euclidean distance matrix, and so does the edge).

The idea of our model is to find a distance matrix that is similar to the Hi-C matrix at the level of the graph. The distance matrix is calculated by a combination of 3D structures from the encoder. Then the decoder transforms the Euclidean distance matrix into a graph. The similarity between two paired graphs can be determined by comparing the corresponding edges.

The Hi-C matrix represents the interaction frequencies between pairs of genomic loci, and the distance matrix represents the Euclidean distance. The weight of the edge in both matrices is the continuous value. To simplify this problem, we discrete both weights and denote the object of self-supervised learning as an edge classification.

- Edge clusters in Hi-C: the label ($\mathbf{L}_{hic}$) of edge starts from intense interaction (label $\#0$) to non/weak interaction.
- Edge clusters in a Distance matrix: the label ($\mathbf{L}_{dist}$) of edge starts from nearby (label $\#0$) to far away.

The graph of Hi-C/Euclidean distance contains one type of vertex (genomic loci) and multiple types of edges, called a heterogeneous graph.

3.1 Overview of the model

The overview of model is shown in the Fig. 3.1 with three parts:

1. Embedding Network: Prepare node embeddings from Hi-C for the encoder.
2. Encoder Network: Encode node embeddings and heterogeneous graph (multiple adjacency matrices) using graph attention networks. The representations are denoted as coordinates of structures ($\mathcal{X} \in \mathbb{R}^{D_s \times N \times 3}$), where N is the number of loci and D_s is the number of 3D structures. We assume there are multiple inconsistent structures in one bulk Hi-C matrix, so we have diverse structures as an ensemble.
3. Decoder Network: Calculate the Euclidean distance matrix of representation and classify the edges.

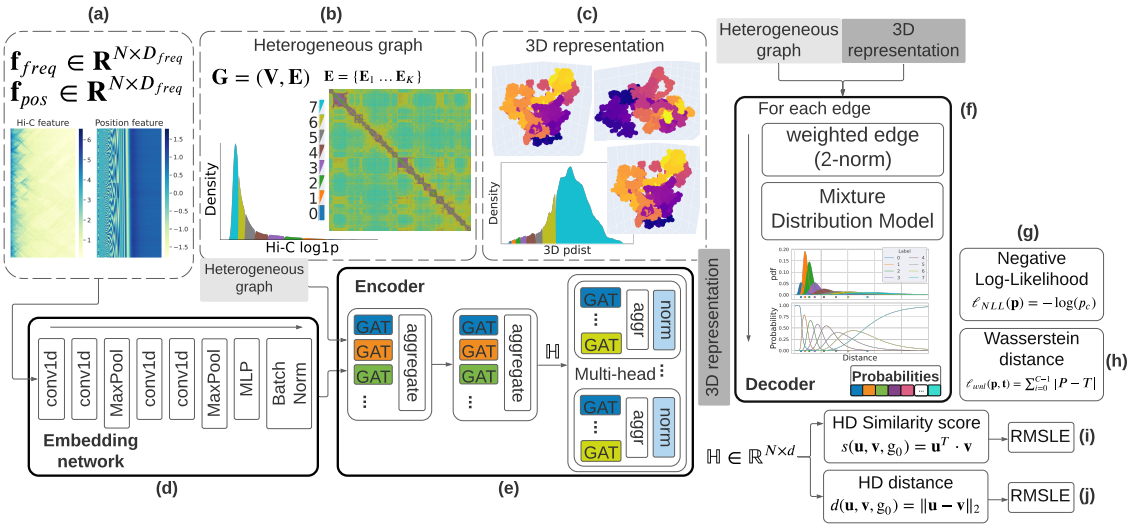


Figure 3.1: **The overview of model, GIST**. (**a**) The input features of Embedding network(d): Hi-C feature and position feature. (**b**) The input heterogeneous graph of Encoder. The histogram is the distributions of multi-types of edges from Hi-C and the matrix is the adjacency matrix from Hi-C. (**c**) The 3D representation is the output of Encoder. The histogram is the distributions of multi-types of edges from all spatial structures. (**d**) Embedding network generates the node feature for the encoder. (**e**) Encoder is a graph neural network, encoding the node feature into 3 dimensions based on the heterogeneous graph. (**f**) Decoder predicts the class of edges. (**g**, **h**) Entropy losses for classification. (**i**, **j**) Similarity and distance losses for the hidden high dimension $\mathbb{H}$. The colors in the (b,c,f) represent the types of edges. The GAT in (e) handles the graph corresponding to the type of edge in different colors.

3.1.1 Constructing Hi-C data as input

The Hi-C matrix (**HiC**) represents an undirected weighted complete graph in the form of an adjacency matrix, any pair of vertices (locus i and j) having an edge with weight $\mathbf{HiC}_{(i,j)}$.

A graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ is a structure to represent relations between objects, which comprises vertex features and edge features (e.g. weights) and relations (e.g. adjacency matrix). There is no necessary and inefficiency to package all information into an adjacency matrix ($\mathbf{HiC}$). Therefore, we separate Hi-C matrix into two parts:

- Feature: contains frequency and position features for node embeddings.
- Graph: contains multi-class edge information and labels of edge.

Feature

In the Hi-C matrix, the main diagonal is invalid, and the offset k above the main diagonal indicates the frequencies between two loci away from k genomic distance. To structure these information, the frequency feature straightens diagonals from genomic distance 1 to D_{freq}. It is defined as a vector $\mathbf{f}_{\text{freq}} \in \mathbb{R}^{D_{\text{freq}}}$ for i-th locus:

$$\mathbf{f}_{\text{freq}}(k-1) = \begin{cases} max(\mathbf{HiC}_{(i,i-k)}, \mathbf{HiC}_{(i,i+k)}) & \text{if } i-k >= 0 \text{ and } i+k < N \\ \mathbf{HiC}_{(i,i-k)} & \text{if } i+k >= N \\ \mathbf{HiC}_{(i,i+k)} & \text{if } i-k < 0 \end{cases}$$

where the index is 0-based and the range of offset $k = [1, \cdots, D_{\text{freq}}]$. $\mathbf{F}_{\text{freq}} \in \mathbb{R}^{N \times D_{\text{freq}}}$ is in the form of matrix, where N is the number of loci.

The frequency feature summaries the local information of each locus, then we employed the position feature[81] to record the order of locus in the one chromosome. The position features identify the loci, so it enables the model to split the graph into small pieces in the learning.

The position feature has the same dimension as frequency feature, D_{freq}. It encodes the index into a vector ($\mathbb{N} \mapsto \mathbb{R}^{D_{\text{freq}}}$):

$$\mathbf{f}_{\text{pos}}(x, i) = \begin{cases} \sin(\omega_k \cdot x) & \text{if } i = 2k \\ \cos(\omega_k \cdot x) & \text{if } i = 2k + 1 \end{cases}$$

where

$$\omega_k = \frac{1}{10000^{2k/D_{\text{freq}}}}$$

$$x = scale \times \text{index}$$

It is also 0-based, $k = \lfloor \frac{i}{2} \rfloor$, and $i = [0, \cdots, D_{\text{freq}} - 1]$. We scale the index because of the resolution of Hi-C. In the model, we set the unit as 1kb, so that $scale$ is 10 at 10kb resolution Hi-C data and assign D_{freq} 300 (3Mb). The frequency and position features are prepared for node embeddings before encoding. The long-range interactions are in the heterogeneous graph.

Heterogeneous Graph

As mentioned above, the graph is a heterogeneous graph with one type of vertex and multiple types of edges. As a self-supervised model for multi-class edge classification, the edge labels are generated from Hi-C data itself. Since the frequency is a non-negative scalar, we employed the 1D Gaussian mixture model to cluster the value of the natural logarithm of one plus the frequencies. The labels are ordered from intense (label $\#0$) to non/weak interaction by sorting the means of components decreasing. In our model, there are 8 clusters ($\#0, \cdots, \#7$) in the 1D Gaussian mixture model, and $\#8$ is for the invalid value (e.g., self-loop, the main diagonal of Hi-C matrix). The histogram in Fig. 3.1b is the classification of edges in Hi-C, and the corresponding edge classification in the spatial structures is in Fig. 3.1c.

As shown in the histogram and adjacency matrix in Fig. 3.1b, the edges are unproportionate. The last cluster(e.g. $\#7$) covers all weak/non-interaction up to 90% in Hi-C matrix, but the crucial edges(e.g. $\#0, \#1$) are less than 1%. To overcome this imbalance problem, the model only uses sub-graphs of the first 7 clusters as input and takes a sampling strategy for all types of edges in the update step (details in the Section 3.1.3).

3.1.2 Embedding model and Auto-Encoder

Embedding network model

Embedding network aims to reduce the dimension of features before passing into the Auto-Encoder. As shown in Fig. 3.1d, it consists of 1D Convolutional neural network layers and fully connected layers. In the forward, the network only considers the features in every single node. But in the backpropagation, it updates parameters together with the Auto-Encoder network, which contains the information of all types of edges in the graph.

Encoder network model

The encoder is a 3-layers graph neural network (in the Fig. 3.1e): two heterogeneous graph convolutional modules and one Multi-head heterogeneous graph convolutional module. All the convolutional modules to different types of edges are independent graph attention networks.

The first two layers embed the node features into hidden representations, and the multi-head layer projects the hidden representations into 3D space. The hidden representations are optimized by minimizing the high-dimensional loss functions (in the Section 3.1.3), and the 3D representations are fitted by negative log-likelihood and Wasserstein metric for the edge classification (in the

Section 3.1.3, and 3.1.3). Heterogeneous graph convolutional module applies graph convolutional layer to each type of edge and then aggregates results as output:

$$\mathbf{h}_{e_i}^{l+1} = \text{GAT}(subgraph_{e_i}, \mathbf{h}^l)$$

$$\mathbf{h}^{l+1} = \text{aggregate}(\mathbf{h}_{e_i}^{l+1} | i \in \mathcal{L}_{hic})$$

Where $\mathcal{L}_{hic}$ is the labels of edges passed into the encoder, $\mathcal{L}_{hic} \subset \mathbf{L}_{hic}$. The $subgraph_{e_i}$ is the corresponding subgraph of edge type i. $\mathbf{h}^l$ is the node representation as input of layer l. $\mathbf{h}_{e_i}^{l+1}$ is the node representation as output of layer l according to $subgraph_{e_i}$.

The GAT is Graph attention network[82] which learns the projection parameters $\mathbf{W} \in \mathbb{R}^{d_{out} \times d_{in}}$ and linear transformation $\mathbf{a} \in \mathbb{R}^{2d_{out}}$ for attention scores. Where d_{in} and d_{out} are the dimensions of input/output representations.

$$\mathbf{h}_{e_i}^{l+1}(u) = \sum_{v \in \mathcal{N}(u)} \alpha_{u,v} \mathbf{W} \mathbf{h}_{e_i}^{l}(v)$$

Where $\alpha_{u,v}$ is the attention between vertices u and $v \in \mathcal{N}(u)$, v is the neighbor of u.

$$\alpha_{u,v} = \text{softmax}_v(s_{u,v})$$

$$s_{u,v} = \text{LeakyReLU}(\mathbf{a}^T \cdot \mathbf{z})$$

$$\mathbf{z} = \text{concat}(\mathbf{W}\mathbf{h}_{e_i}(u), \mathbf{W}\mathbf{h}_{e_i}(v))$$

Where $\mathbf{h}_{e_i}^{l} \in \mathbb{R}^{N_{e_i} \times d_{in}}$ is node representation and $\mathbf{h}_{e_i}(u) \in \mathbb{R}^{1 \times d_{in}}$ and $\mathbf{W}\mathbf{h}_{e_i}(u) \in \mathbb{R}^{1 \times d_{out}}$ both are the representations of vertex u. N_{e_i} is the number of vertices in $subgraph_{e_i}$. Operation concat is concatenation of vectors/matrices.

For the first two layers, our model employs a learnable parameter $\mathbf{W}_{\text{agg}} \in \mathbb{R}^{(|\mathcal{L}_{hic}| d_{out}) \times d_{out}}$ to aggregate results:

$$\mathbf{h}^{l+1} = \underset{\forall i \in \mathcal{L}_{hic}}{\text{concat}}(\mathbf{h}_{e_i}^{l+1}) \cdot \mathbf{W}_{\text{agg}}$$

Multi-head Heterogeneous graph convolution module is similar to the previous two. Each attention head has own parameters and their outputs are not merged in our model. The only difference is the aggregation step:

$$\mathbf{h}^{l+1} = \text{mean}(\underset{\forall i \in \mathcal{L}_{hic}}{\text{stack}} (\mathbf{h}_{e_i}^{l+1}) \cdot \mathbf{U}^m)$$

Where $\mathbf{U}^m \in \mathbb{R}^{|\mathcal{L}_{hic}| \times |\mathcal{L}_{hic}|}$ is the learnable parameter in m-th head. In the aggregation step, it stacks representations of edge types and combines them by linear combination, then reduces them by mean. The implementation is based on Deep Graph Library[83].

Representation Normalization The encoder normalizes the 3D representations before output. We rescale the length between sequential nodes(loci) around 1. The normalization adjusts the distance between two continuous nodes where the edge between two usually is at label $\#0$ except gap of invalid data. So that this normalization also anchors the length of edges at Label $\#0$ (around 1 in our model).

Let $\mathbf{X} \in \mathbb{R}^{N \times 3}$ is one 3D representation from one head and $\mathbf{x}_i$ is the i-th node in the $\mathbf{X}$. The normalized representation $\mathbf{S}$ is

$$\begin{aligned}
\mathbf{s}_0 &= \mathbf{x}_0 \\
\mathbf{d} &= \{\mathbf{d}_i | \mathbf{d}_i = \mathbf{x}_i - \mathbf{x}_{i-1}, 0 < i < N\} \\
\mathbf{s}_i &= \mathbf{s}_{i-1} + \frac{\mathbf{d}_i}{\underset{\forall k, 0<k<N}{\text{median}} (\mathbf{d})}
\end{aligned}$$

In terms of the intra-chromosomal Hi-C, most nodes (loci) are sequential; the median of $\mathbf{d}$ is the distance between two continuous nodes removing the gap distances as outliers.

The normalization ensures the starting node is unmodified and all 3D structures at the same scale where the distance between continuous nodes is around 1 as the unit. Therefore, in the training step, it is eligible to split a large graph into multiple small subgraphs as long as the nodes

are sequential. For instance, we can extract nodes by sliding windows with overlap. Then the model predicts the entire structures once.

Decoder network model

The encoder embeds node features to 3D space from the heterogeneous graph of Hi-C, as shown in Fig. 3.1f. The representation of 3D space, $\mathcal{X} \in \mathbb{R}^{D_s \times N \times 3}$, is denoted as the structures of chromosome in 3D as the final outputs.

The decoder aims to classify the edge in terms of node 3D representation for edge classification. It first calculates the weighted Euclidean distance of edges and then classifies them by a mixture distribution where all components are the same types as normal distributions. The backpropagation updates the weights and parameters in the mixture distribution.

For any edge in the graph, the weighted Euclidean distance is:

$$\underset{\mathbf{u},\mathbf{v} \in graph}{w_{2\text{-}norm}(\mathbf{u}, \mathbf{v})} = \mathbf{a}_{euc}^{T} \|\mathbf{u} - \mathbf{v}\|_2$$

Where $\mathbf{u}$ and $\mathbf{v}$ are vertices. $\mathbf{a}_{euc} \in \mathbb{R}^{D_s}$ is the weights, the Euclidean norm is on the 3rd dimension of $\mathbf{u} \in \mathbf{R}^{D_s \times 1 \times 3}$. We transform the distance by natural logarithm before feeding it into the Gaussian mixture model.

$$l_w = \log(w_{2\text{-}norm})$$

The number of components is $|\mathbf{L}_{hic}| - 1$(without the invalid label), then the probability of i-th component given log-distance is:

$$Pr(C = i \mid L_w = l_w) = \frac{Pr(C = i) Pr(L_w \mid C = i)}{\sum_k Pr(L_w, C = k)} = \frac{\pi_i \mathcal{N}(\mu_i, \sigma_i^2)}{\sum_{k=0}^{|\mathbf{L}_{hic}|-2} \pi_k \mathcal{N}(\mu_k, \sigma_k^2)}$$

Where π_i is the probability of components i, denoted as $\pi_i \in \boldsymbol{\pi}, \sum_i \pi_i = 1$. The C is the random variable of components. Since the the distributions are applied to logarithm of distance, the distances in the same label follow Lognormal distribution as:

$$L_{w,C_i} \sim \mathcal{N}(\mu_i, \sigma_i^2)$$

$$W_{2\text{-}norm,C_i} \sim \text{LogNormal}(\mu_i, \sigma_i^2)$$

The L_w and $W_{distace}$ are random variables and C_i stands for $C = i$. Corresponding to the L_{hic}, the classes of L_{dist} start from short to long distances. To achieve so, the decoder enforces an ascending order of modes in components. The mode of LogNormal distribution is:

$$mode_i = \exp(\mu_i - \sigma_i^2)$$

Therefore, the components are ordered by modes. The decoder outputs the probabilities of edges, denoted as $\mathbf{p}_e \in \mathbb{R}^{|\mathbf{L}_{hic}|-1}$. The element $\mathbf{p}_{e,i} \in \mathbf{p}$ is the probability of edge e for the label i: $Pr(C = i|l_w)$. In Fig. 3.1f, the solid upper plot shows the probability density functions of types of edges, and the bottom plot is the probabilities of edge clusters after normalization. Both plots share the same Euclidean distance as the x-axis.

3.1.3 Losses

The objective of the model is to estimate representations in 3D space where the heterogeneous graph from the decoder matches with the heterogeneous graph of Hi-C. There are two parts of losses:

- Loss functions for edge classification in the Fig. 3.1g, 3.1h. Negative log-likelihood maximizes the probability of target, and Wasserstein distance minimizes the difference between labels distributions.

- Loss functions for hidden high-dimensional space, in the Fig. 3.1i, 3.1j, are supporting entries to combine the properties of similarity (Hi-C) and distance (Euclidean) in one space.

The loss value is based on each edge in subgraph(s). The loss function is

$$\begin{aligned}\mathcal{L}oss &= 5\frac{1}{|E_s|}\sum_{e\in E_s}\ell_{NLL}(\mathbf{p}_e, t_e) + 5\frac{1}{|E_s|}\sum_{e\in E_s}\ell_{wnl}(\mathbf{p}_e, t_e) + \\ &\quad \ell^{sim}_{rmsle}(\mathbf{u}, \mathbf{v}, g) + \ell^{dist}_{rmsle}(\mathbf{u}, \mathbf{v}, g)\end{aligned}$$

Where $|E_s|$ is a set of edges sampling from all edges to balance multi-classes. 5 is an artificial parameter, introduced in section3.1.3. $\mathbf{p}_e$ is the predicted probabilities of edge e from decoder. t_e is the target of edge e. The $\mathbf{u}$ and $\mathbf{v}$ are embeddings in hidden high-dimensional space where edge $e_{\mathbf{u},\mathbf{v}} \in E_g$, E_g is the set of edge in the graph g.

Categories balancing

As mentioned above, the edge classes are imbalancing: the proportion of weak/non-interaction is up to 90%. As a trade-off, the decoder focuses on the graphs of intense interactions of Hi-C and ignores the graph of weak/non-interaction to generate the representation of 3D structures. It makes the decoder efficient but loses the information indicating pairs of loci far away.

$$\mathbf{E}^i_{\text{sample}}\begin{cases}\sim \text{Multinomial}(n, \frac{1}{|\mathbf{E}_i|}) \text{ no replacement} & \text{, if } |\mathbf{E}_i| >= n \\ \text{repeat } \mathbf{E}_i, \lfloor\frac{n}{|\mathbf{E}_i|}\rfloor \text{ times} & \text{, otherwise}\end{cases}$$

As analogous to the negative links in the link prediction, weak/non interactions are the negative samples in our case. The model samples the edges before calculating losses. Let $\mathbf{E}$ is the set of all edges and $\mathbf{E}_i$ is the set of edges at label $\#i$, $\mathbf{E} = \{\mathbf{E}_i | i = [0, \cdots, |\mathbf{L}_{hic}| - 2]\}$. n is the number of samples, $n = 0.8 * |\mathbf{E}| * \frac{1}{|\mathbf{L}_{hic}|-1}$. Then we merge sets together with duplicate elements, $\mathbf{E}_s = \{\mathbf{E}^i_{sample} | i = [0, \cdots, |\mathbf{L}_{hic}| - 2]\}$.

Negative log-likelihood

Negative log-likelihood (NLL) loss function is powerful when dealing with multi-class classification by maximizing the log-likelihood of target class. In our model the predicted probability generated from the decoder directly and the target is the label of edge class. The negative log loss is:

$$\ell_{NLL}(\mathbf{p}) = -\log(p_c)$$

Where $\mathbf{p}$ is the probabilities of multi-classes(C classes), $\mathbf{p} \in \mathbb{R}^C$ and $\sum_{i=0}^{C-1} p_i = 1$. p_c is the predict probability of target (label c). The loss of one edge is shown as Equation.3.1.3 and the reduction of losses is mean.

Wasserstein metric

The Wasserstein metric measures the distance between two distributions over a region, also known as Earth mover's distance. In our case, the labels of edge classes are ordered, increasing when the length of the edge increases. The NLL only focuses on the target but ignores other missed situations. However, the Wasserstein metric measures the difference between two distributions of classes.

For instance, if the target is label 0 and one prediction $\mathbf{a}$ gets the highest probability at label 2 and another prediction $\mathbf{b}$ gets the highest probability at label 4, NLL measures no difference likely between $\mathbf{a}$ and $\mathbf{b}$, because both have low probabilities of label 0 but prediction $\mathbf{a}$ gains lower distance in Wasserstein metric because it is closer to the target.

In this model, we utilize the first Wasserstein distance between two 1D distributions:

$$\ell_{wnl}(\mathbf{p}, \mathbf{t}) = \sum_{i=0}^{C-1} |P - T|$$

Where P and T are the respective cumulative distribution functions (CDF) of two distributions $\mathbf{p}$ (prediction) and $\mathbf{t}$ (true). C is the number of classes, the region of both distributions. CDF is easy to generate by the cumulative sum of predicted probabilities or the one-hot.

Hidden High-dimensional loss functions

The representation of hidden high-dimensional space, $\mathcal{H} \in \mathbb{R}^{N \times d_{\text{hidden}}}$, is the embeddings before passing into multi-head Heterogeneous graph convolution module. These loss functions encourage the hidden high-dimensional embeddings to have both properties of Hi-C and manifold. In our case, the subgraph of edge label $\#0$ (subgraph_0) stores these neighbor information. Hi-C matrix describes the similarity between loci. Hence the loci connected by the edges $\#0$ are nearby. We employed dot product to measure the similarity at this high-dimensional space:

$$s(\mathbf{u}, \mathbf{v}, \text{subgraph}_0) = \mathbf{u}^T \cdot \mathbf{v}$$

On the other hand, the vertices in subgraph_0 are nearby in space. Euclidean distance metric on subgraph_0 constrains the embeddings locally resembling Euclidean space near each point as a topological space (manifold):

$$d(\mathbf{u}, \mathbf{v}, \text{subgraph}_0) = \|\mathbf{u} - \mathbf{v}\|_2$$

The high-dimensional losses only apply to the subgraph_0. The Root Means Squared Log Error is involved in optimizing both similarity and distance separately.

$$\ell_{rmsle}(\hat{y}, y) = \sqrt{\frac{1}{|E_0|} \sum_{E_0} (\log 1\text{p}(y) - \log 1\text{p}(\hat{y}))^2}$$

Where E_0 is the set of edge in subgraph_0. The similarity score is compared with normalized Hi-C frequency. The Euclidean distance is compared with the mode of component_0 from the decoder.

3.2 Results

We used published Hi-C datasets in human cell line: IMR90 (lung fibroblast cells)[59] to predict 3D structures and the fluorescent in situ hybridization (FISH) data[85] in the same cell line as the independent data for the validation and evaluation. The FISH experiment reveals the spatial distance between some DNA fragments in the live cell. The genome coordinates and genome annotation of topologically associating domains (TADs) are from FISH data, and all are lifted from hg18 to hg19 before the validation and the evaluation. The conformations of Chromosomes (20, 21, and 22) are applied in the validation, and the conformations of Chromosome Xa and Xi are in the evaluation. There are two parts in the independent validation:

- The accuracy of the A/B compartments partition at the levels of loci and TADs.
- The relative distance error at the TADs level.

Meanwhile, we compared our model with other modeling methods (ShRec3D, pastis, LorDG, GEM, and ChromSDE). In the evaluation, we predicted and identified the active and inactive conformations of chromosome X and compared the classification with the conformations from FISH data.

All models (GIST and others) predicted structures of Chromosomes 20, 21, and 22 for the validations, and only GIST predicted structures of Chromosome X for the evaluation. All the 3D structures are generated from Hi-C data at 10kb resolution. The normalization is an essential step in Hi-C data analysis because of various types of technical and biological biases in the raw Hi-C matrix. Here the modeling methods (pastis, LorDG, GEM, and ChromSDE) are all applied ICE to normalize Hi-C except ShRec3D, as it requires SCN explicitly. For GIST, we conducted a log1p transformation (i.e. $\log(1+x)$) after ICE normalization to generate the features (Hi-C and position

features). The heterogeneous graph has 9 clusters of edges: labels $\#0 \sim \#7$ are from 1D Gaussian mixture model and label $\#8$ indicates invalid data, e.g., the main diagonal. In the training step, one sample is one subgraph, and corresponding features are generated by a sliding window with overlap. The structures are predicted by the entire heterogeneous graph and features of the chromosome.

3.2.1 A/B compartments identification

We validated the structures by assessing the identification of A/B compartments. The partition algorithm[85] identifies the compartments of loci from the Euclidean distance matrix, which is developed for Hi-C analysis. The algorithm of A/B compartments partition is:

- Normalize the matrix by diagonal: $\text{norm}_{i,j} = \frac{M_{i,j}}{\text{diag}_k}$, where $k = |i - j|$, the diag_k is the mean of values that offset k above the main diagonal.
- Calculate the Pearson correlation for each pair of rows.
- Decompose the correlation by principal component analysis (PCA).

In the Hi-C data, we assigned the loci with positive first principle component (PC) coefficients to compartment A and negative coefficients to compartment B. However, the first principle component in the Euclidean distance matrix doesn't always work well because of the gap of missing data, as mentioned previously. Therefore, we selected the best one from the first 3 PCs.

The Table. 3.1 shows the accuracy of A/B compartments partition. The Table. 3.1a is the accuracy in assigning loci to A/B compartments and the Table. 3.1b is the accuracy at TADs level. The PC values are from the center of indices of TADs. As shown in both tables, the accuracies of GIST reached around $70\% \sim 80\%$. Moreover, it outperformed others in chromosomes 21 and 22.

	GIST	ShRec3D	LorDG	pastis	GEM	ChromSDE
20	74.65%	**79.54**%	63.08%	51.54%	51.49%	53.72%
21	**83.24**%	78.65%	72.28%	60.49%	55.56%	64.65%
22	**74.60**%	73.01%	73.74%	52.99%	61.66%	57.39%

(a) Accuracy in assigning loci to A/B compartments

	GIST	ShRec3D	LorDG	pastis	GEM	ChromSDE
20	72.41%	**82.76**%	62.07%	62.07%	62.07%	72.41%
21	**81.82**%	75.76%	63.64%	60.61%	72.73%	69.70%
22	**76.92**%	73.08%	**76.92**%	65.38%	65.38%	65.38%

(b) Accuracy in assigning TADs to A/B compartments

Table 3.1: **Accuracy of A/B compartments partition**

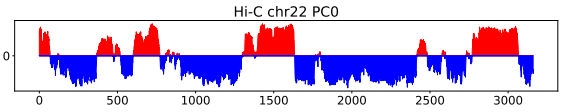
In chromosome 20, only the result of ShRec3D is better than ours. Fig. 3.2 illustrates the partition of the first PC from Hi-C and GIST at both levels in chromosome 22.

3.2.2 Relative distance error between TADs

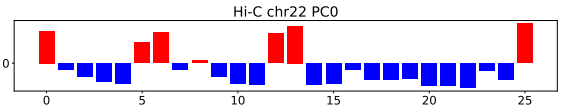
We assessed the accuracy of our prediction by measuring the relative error for all pairwise of TADs, defined as:

$$\text{error}_{i,j} = \frac{|f(d_{i,j}) - F_{i,j}|}{F_{i,j}}$$

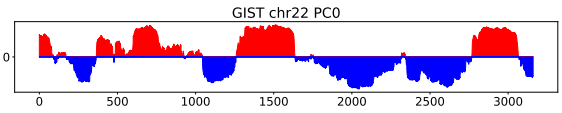
Where $d_{i,j}$ is the Euclidean distance between center of TADs i and TADs j in predicted structures and $F_{i,j}$ is corresponding relative distance from FISH data. f is the first degree polynomial equation



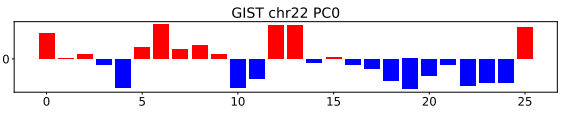
(a) The first PC of Hi-C, loci level



(b) The first PC of Hi-C, TADs level



(c) The first PC of distance matrix from GIST, loci level



(d) The first PC of distance matrix from GIST, TADs level

Figure 3.2: **The partition of A/B compartments in chromosome 22**.

to fit with least square:

$$f(d_{i,j}) = ad_{i,j} + b$$

Where a is the scale, and b is the intercept. The scale is to scale the coordinates, and the intercept is involved because of the gap of missing data in the Hi-C data. The gap stretches the structure in one direction resulting in the intercept in the Euclidean distance matrix.

The Fig. 3.3 illustrates the relative distance errors of the predictions from all methods in the chromosome 20, 21 and 22. The Fig. 3.3(a) is the scatter plot of pairwise distances between TADs from FISH and prediction. The y axis is the distance from FISH data and the x axis is from prediction of GIST. The correlation coefficient between the two datasets is 0.77. The Fig. 3.3(b) is the heatmap of relative distance error from GIST. Fig. 3.3(d) is the boxplot of relative errors between TADs from the six models, and Fig. 3.3(c) is the corresponding heatmap of the t-test between GIST and other methods in the chromosomes 20, 21, and 22. The p-values indicate that the relative errors from GIST are significantly less than others. The two p-values larger than 0.01 indicate that the errors from GIST and ShRec3d have identical average values in chromosomes 21 and 22. The rest

p-values of pairwise models are in the supplementary. According to the low relative errors, the predicted structures from GIST are more close to the coordinates from the FISH experiment at the TAD level.

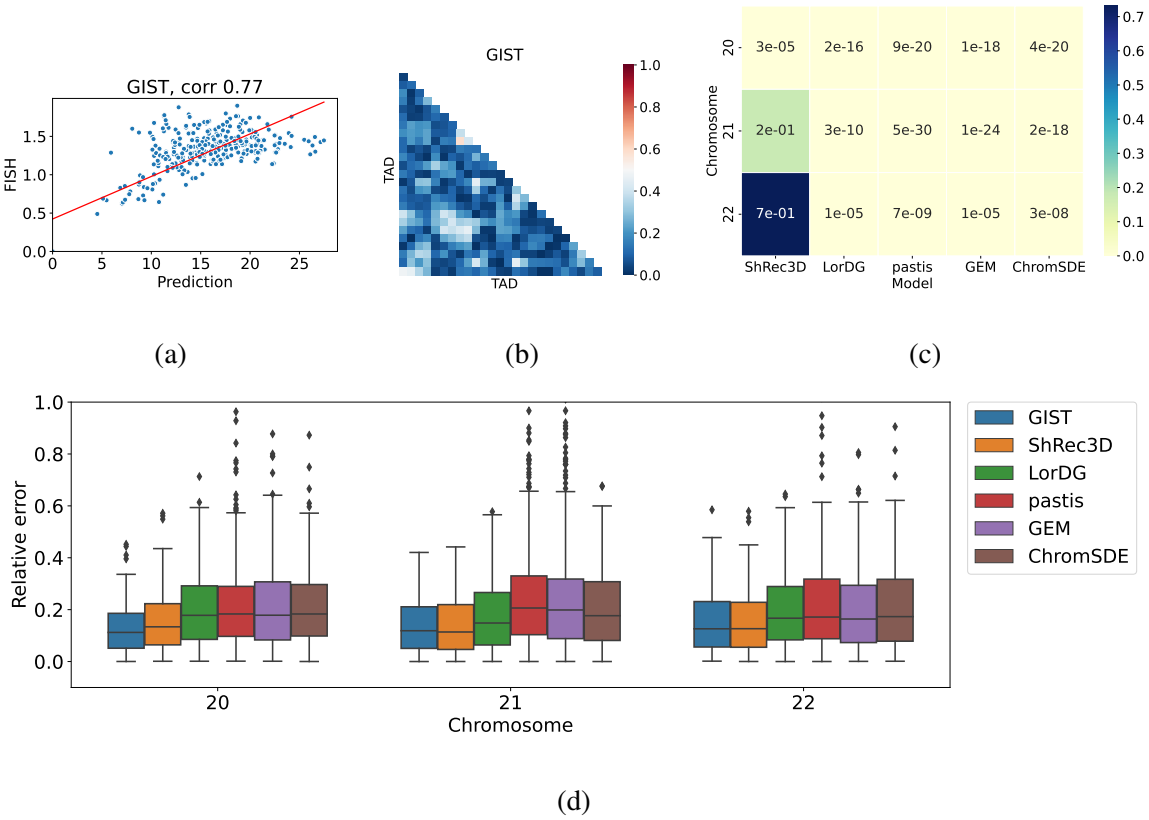


Figure 3.3: **The results of relative distance errors**. (**a**) Scatter plot of distance between TADs chromosome 22. The red line is the fitting curve: $a : 0.055$, $b : 0.423$. (**b**) The relative distance error between FISH data and prediction of GIST in chromosome 22. (**c**) Heatmap of p-values from t-tests between GIST and others in the chromosome 20, 21 and 22. The alternative hypothesis: less. (**d**) Boxplot of distances in the six models in the chromosome 20, 21 and 22.

3.2.3 Diversity of structures in the Chromosome X

The partition of A/B compartments and relative distance between pairwise TADs have validated the predictions from GIST. In contrast to the consistent assumption methods, GIST produces a set of predictions with proportions to trade-off limited consistent conformations against the inconsistency in the Hi-C. As known, the Hi-C matrix of Chromosome X ensembles the active (ChrXa) and inactive (ChrXi) states. To investigate the diversity of structures from GIST, we clustered the predicted structures of Chromosome X in the IMR90 cell line.

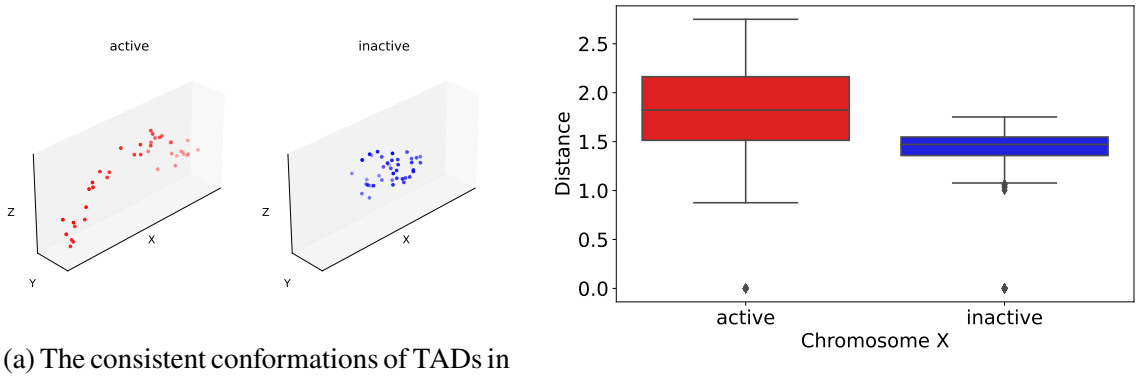
The goal of clustering is to distinguish the two statuses from predicted structures. We leveraged two clustering methods based on distance matrix decomposition and 3D coordinates alignment to evaluate the predictions:

1. **Distance matrix clustering method** extracts the features from the distance matrix using tensor decomposition and then performs hierarchical linkage clustering.

2. **3D coordinates clustering method** performs hierarchical linkage clustering on the alignment root mean squared distance (RMSD) values.

In the FISH data, there are 95 samples of ChrXa conformations, and 41 samples are complete (no NaN values for all TADs), and 48 out of 95 samples of ChrXi are complete. The consistent conformation of ChrXa(or ChrXi) is defined as the average of completed samples after alignment. The coordinates of TADs in FISH data are denoted as ground truth. Since the samples of ground truth are at the same scale, we rotated all samples to align the first one and then took the average of coordinates as a consistent conformation as shown in Fig. 3.4a. Fig. 3.4b is a boxplot of the distance between TADs in the chromosome Xa and Xi. The distances in Chromosome Xa are

significantly greater than the distances in Chromosome Xi. The corresponding TADs in chromosome Xa unfold in 3D visualization, and the TADs in chromosome Xi are compact.



(a) The consistent conformations of TADs in the ChrXa(red) and ChrXi (blue) from FISH data.

(b) The boxplot of distances between pairwise TADs in ChrXa (red) and ChrXi (blue).

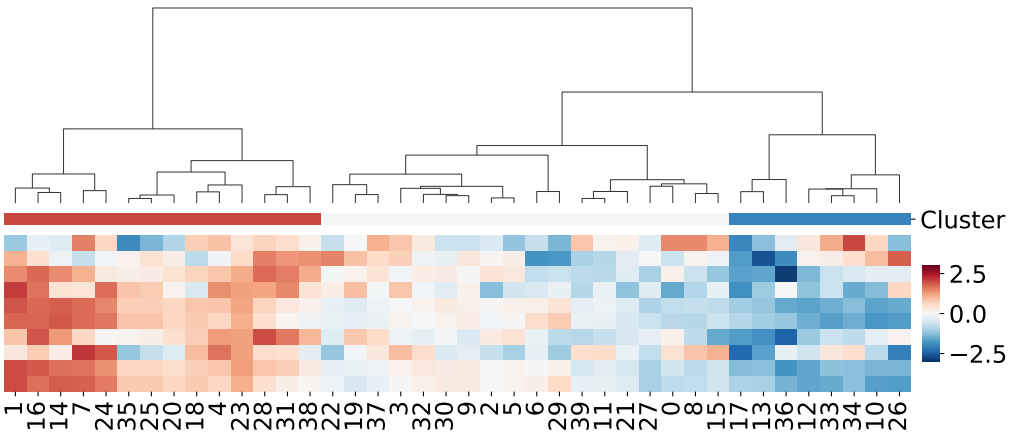
Figure 3.4: **The Chromosome Xa and Xi in FISH data**.

The distance matrix clustering method is introduced below:

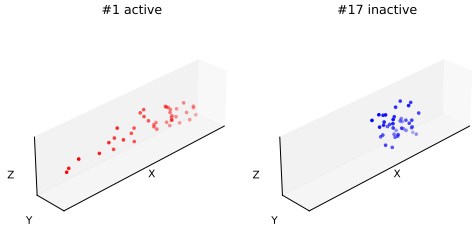
1. Calculate the Euclidean distance between TADs for each prediction, as a matrix $\mathbf{m} \in \mathbb{R}^{n \times n}$, and stack them as a third order tensor $\mathbf{T} \in \mathbb{R}^{D_s \times n \times n}$, where $D_s = 40$ in the evaluation.

2. Decompose the tensor by Parafac2[33, 31] with r rank: $\hat{\mathbf{T}}_i = \mathbf{B}_i \text{diag}(\mathbf{a}_i)\mathbf{C}$, where the matrix $\mathbf{T}_i \in \mathbb{R}^{n \times n}$ is the i-th slice in the the tensor $\mathbf{T}$. The $\mathbf{a}_i$ is a nonzero-vector and $\text{diag}(\mathbf{a}_i)$ is a diagonal matrix where $\mathbf{a}_i$ is the main diagonal. We denote the $\mathbf{a}_i$ is the feature of i-th structure and all features are denoted as matrix $\mathbf{A}$. Where the $\mathbf{B}_i \in \mathbb{R}^{n \times r}$ is a factor matrix for i-th slice and $\mathbf{C} \in \mathbb{R}^{r \times n}$ is factor matrix for all slices, and $r = 10$ in the evaluation.

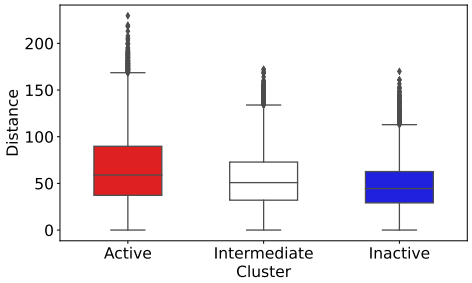
3. Cluster the structures based on the matrix $\mathbf{A}$ by linkage (metric: Euclidean, method: ward)



(a) Heatmap and dendrogram of feature matrix from parafac2 in chromosome X. The x axis is the index of predictions. The left branch of dendrogram in red represents active conformations. The middle branch in white contains the intermediate conformations. The right branch in blue represents inactive conformations.



(b) Two visualization demos for clustering: #1 prediction (active, red) and #17 prediction (inactive, blue)



(c) The boxplot of distances between pairwise TADs in predictions.

Figure 3.5: **The classification of predictions by Parafac2**.

In Fig. 3.5a, it shows the heatmap of feature matrix $\mathbf{A}$ and the dendrogram of clusters. One column in the heatmap is a feature vector of one prediction. As shown in the dendrogram, the left branch in red is classified into a cluster of active structures. The right branch in blue represents the inactive cluster. The $\#1$ and $\#17$ are two instances in active and inactive clusters, visualized in Fig. 3.5b. The white part is the intermediate cluster. As shown in Fig. 3.5c, the three boxplots are the distances of 3 clusters. The distances inactive cluster (red) are significantly greater than the distances in the inactive cluster (blue) (the p-value is close to 0). Moreover, we calculated the alignment root mean squared distance (RMSD) between coordinates of TADs in FISH data and coordinates of TADs in predictions. The evaluation reported the RMSD between predictions and all samples in the ChrXa/ChrXi. The 3D coordinates alignment clustering method is:

1. Resize the predictions at the same scale with the ground truth. The scale comes from the curve fit between distance matrices, the same method described in section3.2.2 (only use the scale a here).

2. Calculate the pairwise RMSD between scaled predictions and ground truths, using the Kabsch algorithm for alignment.

3. Cluster both predictions and ground truths based on the RMSD matrix by linkage (metric: Euclidean, method: ward(prediction)/complete(ground truth))

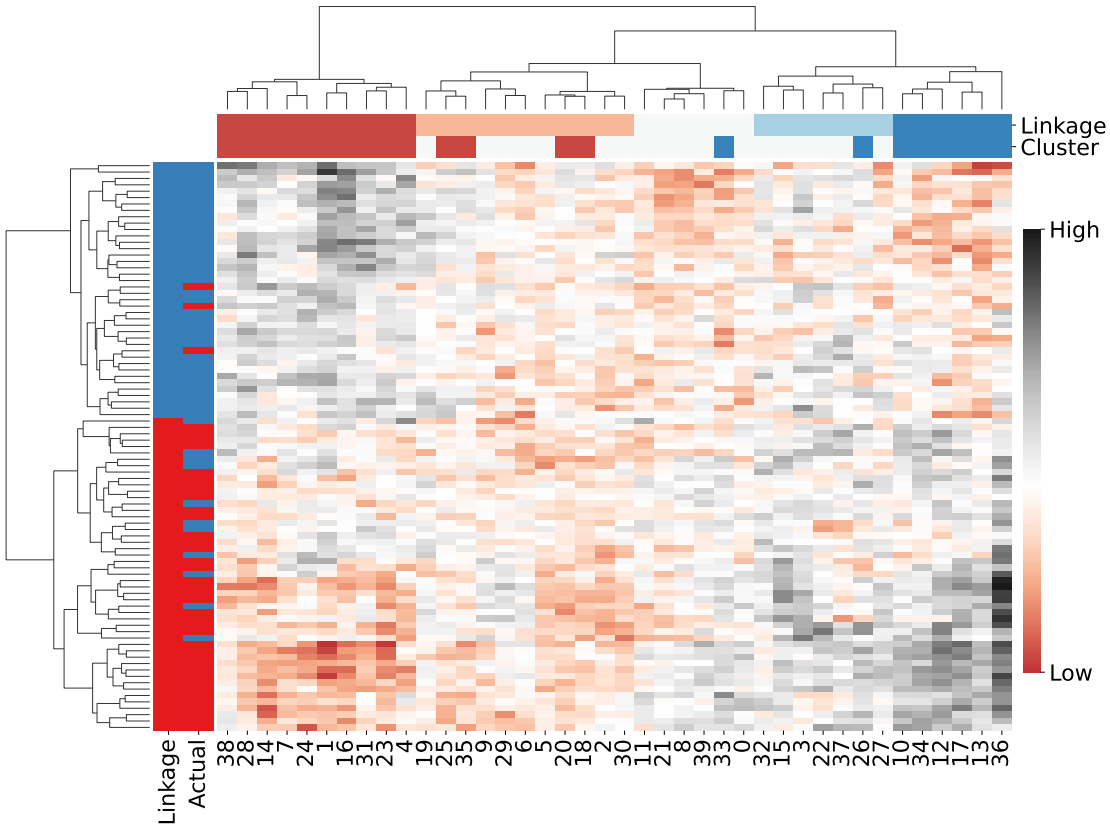


Figure 3.6: **Classification of Chromosome Xa and Xi**. Cluster the chromosome X active and chromosome X inactive based on alignment RMSD. The heatmap illustrates the matrix of alignment RMSD between all conformations and predictions.

Fig. 3.6 shows the heatmap of alignment RMSD between all conformations of ground truth and the predictions. The heatmap of RMSD value from red to grey indicates the distance from close(low) to far away(high), and the linkage clustering method is applied here too. The dendrogram on the top is the classification of predictions, and the left is the classification of ground truth. The red/blue column, named 'Actual', represents the true labels of 89 samples (41 Xa and 48 Xi conformations) and the red/white/blue row, named 'Cluster', is the classification from distance

matrix decomposition. Both column and row, named 'Linkage,' indicate the classification from left/top dendrograms, active conformations in red and inactive ones in blue.

The samples from FISH data are clustered into two groups based on the dendrogram of RMSD, sightly different from the true active/inactive labels. In terms of predictions, the predictions in the most left and right branches are classified as active(red) and inactive(blue). As shown in the heatmap, the top left dark(grey) area and the bottom left area in red indicate that the predictions in the left branch are far away from the ground truth of the inactive group but close to the active group and vice versa in the most right branch. The 3 clusters in between (light red, white, and light blue) are intermediate containing the intermediate group and a couple of structures of active/inactive clusters labeled by the distance matrix clustering method. These mismatches of both dendrograms are acceptable. After all, the individual structures are diverse, and the RMSD values are between the structures from GIST and FISH data.

In summary, the two clustering methods evaluated the diversity of the population of predicted structures from GIST in terms of distance matrices and spatial alignment separately. Both clustering methods generated the same identification of the active and inactive conformations in chromosome X within acceptable differences. In chromosome X, the set of predicted conformations from GIST consists of apparent active and inactive clusters and some intermediate conformations.

Chapter 4

Learning fine-resolution Hi-C using a generative adversarial network

Firstly, We introduce a few notations regarding the Hi-C contact frequency matrix. A bulk Hi-C experiment characterizes an ensemble of chromatin contacts from thousands or millions of cell nuclei. The raw data generated from the Hi-C experiment can be presented as a non-negative symmetric matrix $C_{N\times N}$, namely the contact frequency matrix, where N is the number of fixed-size non-overlapping bins in the genome. Each matrix element C_{ij} is the observed contact frequency between the genomic loci pair i and j. A higher contact frequency indicates a smaller spatial distance between a pair of genomic loci in cell nuclei. In short, we refer to the bulk Hi-C contact frequency matrix as the Hi-C matrix.

In our method, we aim to predict high-resolution Hi-C matrices from low-resolution input data. Here, high resolution indicates more chromatin interaction details (i.e., more valid pairs of sequencing reads), rather than a higher dimension of the Hi-C matrix. Given a Hi-C input dataset,

it can processed into a matrix of any arbitrary bin size. Therefore, a high-dimensional Hi-C matrix is not always of high resolution. In this work, we refer to the dimension of the Hi-C matrix as its scale. A lower-scale Hi-C matrix has a smaller number of rows and columns.

4.1 Overview of the model

In this section, we describe the framework of the EnHiC model. More details of the model are provided in Appendix Information. EnHiC is based on a GAN framework that contains a generator and a discriminator. Through competition between them, the generator learns to predict high-resolution Hi-C matrices from low-resolution input matrices, while the discriminator distinguishes the generator-predicted high-resolution matrices from real data.

The main difference between our model and other GAN-based approaches is that EnHiC exploits the unique properties of the Hi-C matrix and treats it as a multi-scale interaction contact map instead of a pure image. Specifically, EnHiC extracts rank-1 matrix features from low-resolution input data at multiple scales and learns to enhance the matrix resolution using these estimated rank-1 features. The overview of the EnHiC framework is illustrated in Figure 4.1.

4.1.1 Decomposition & Reconstruction Block

A key component in our model is the *Decomposition & Reconstruction Block*, as illustrated in Figure 4.1 and Appendix Figure B.1. In our model, we represent a Hi-C matrix as a multi-channel image (i.e., a tensor). Let c_{in} and c_{out} be the number of input and output channels, respectively. The input and output tensors are denoted by $\mathbf{X} \in \mathbb{R}^{N \times N \times c_{\text{in}}}$ and $\widehat{\mathbf{X}} \in \mathbb{R}^{N \times N \times c_{\text{out}}}$, where N is the dimension of the Hi-C matrix.

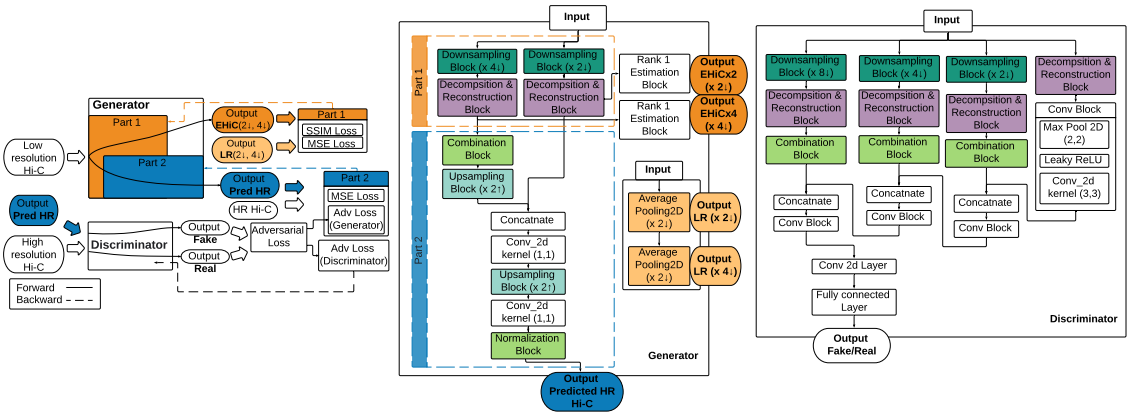


Figure 4.1: **The framework of the EnHiC model**. The details of the *Downsampling Block*, *Upsampling Block*, *Combination Block*, *Normalization Block*, *Rank-1 Estimation Block*, and *Decomposition & Reconstruction Block* are illustrated in the Appendix B.1.

The *Decomposition & Reconstruction Block* contains three layers:

- The decomposition layer, which passes $\mathbf{X}$ into a convolutional layer with kernel $(1, N)$. In contrast to the traditional convolutional layer, the kernel is a vector rather than a square matrix. The length of the kernel vector is the same as the height/width of the input tensor. Hence, the kernel only moves in one direction, and the number of shared parameters for this convolutional layer is $N \times c_{\text{in}} \times c_{\text{out}}$. The resulting tensor is denoted by $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_{c_{\text{out}}}] \in \mathbb{R}^{N \times 1 \times c_{\text{out}}}$, which represents the rank-1 features of the input data.

- The weighting layer, which scales the feature tensor as $\mathbf{V} = \mathbf{U}\mathbf{w}\mathbf{w}^\top$, where the weight vector $\mathbf{w} = [w_1, \ldots, w_{c_{\text{out}}}]^T$ is a learnable parameter.

- The reconstruction layer, which constructs the output tensor $\widehat{\mathbf{X}}$ using the weighted rank-1 features. For each channel k, we have a rank-1 matrix $\widehat{\mathbf{X}}_{::k} = \mathbf{v}_k \mathbf{v}_k^\top$, where $k = 1, \cdots, c_{\text{out}}$.

4.1.2 Generator

The generator consists of two parts: 1) extracting rank-1 matrix features from low-resolution input matrices at multiple scales and 2) enhancing Hi-C matrix resolution using the multi-scale features learned in the first part. The overview of the generator framework (G_1 in the orange dashed box and G_2 in the blue dashed box) is shown in Figure 4.1.

Because the low-resolution input matrix is often sparse, we first downscale the matrix to enhance its signal. The downscaling operation is achieved by shrinking the size of the matrix by an average-pooling layer. In our experiments, we aim to enhance the resolution of the Hi-C matrix by a factor of 16, which is equivalent to scaling up the matrix by a factor of 4 (i.e., multiplying both the height and width of the matrix by 4). Therefore, in our model, we generate two downscaled matrices by factors of 2 and 4 (denoted as LR($\times 2 \downarrow$) and LR($\times 4 \downarrow$), respectively). We use LR($\times 2 \downarrow$) and LR($\times 4 \downarrow$) as the ground truth to assist in the estimation of the rank-1 matrix features at the corresponding scales. Note that in our EnHiC framework, the number of downscaling operations can be adjusted for different applications. For instance, if we aim to enhance the Hi-C resolution by a factor of 100, it is recommended to include additional levels of downscaled matrices (and accordingly, more *Decomposition & Reconstruction Blocks*) to facilitate a better estimation of matrix features.

The first part of the generator (G_1) extracts multi-scale rank-1 features from the low-resolution input matrix. First, it transforms the input matrix ($N \times N$) into a tensor ($\frac{N}{r} \times \frac{N}{r} \times r^2$) using a space-to-depth layer (TensorFlow built-in function). The space-to-depth layer permutes the spatial blocks of the input matrix into the depth dimension without any loss of information. Then, a multi-channel image (tensor) is subsequently processed through the *Decomposition & Reconstruction Block* and its rank-1 features are extracted. Note that the input Hi-C matrix is

symmetric and non-negative, and our rank-1 approximations retain the symmetric and non-negative properties of the data. In our framework, we extract the rank-1 features for two different scales (r =2 and 4, respectively), and the two estimation matrices, denoted as EHiC($\times 2 \downarrow$) and EHiC($\times 4 \downarrow$), are compared against the true data, as shown in Figure 4.1.

The second part of the generator (G_2) recombines the rank-1 features from multiple scales and enhances the matrix resolution through a series of *Upsampling Blocks*. The *Upsampling Block* contains a sub-pixel convolutional layer [68] that upscales the previously learned features in low-resolution space to a high-resolution output. The upscaled tensor is subsequently averaged with its transpose to reinforce the symmetric property of the output matrix. In concert with the two *Decomposition & Reconstruction Blocks* in the first part, we have two *Upsampling Blocks*, each of which upscales the matrix dimension by a factor of 2 (i.e., enhancing the data resolution by a factor of 4). Therefore, the final output matrix has an enhanced resolution by a factor of 16 compared to the low-resolution input matrix. Details of the *Upsampling Block* are illustrated in Appendix Figure B.1.

4.1.3 Loss functions of the generator

The objective of the generator is to estimate the rank-1 features at multiple scales and to enhance resolution of the input matrix. Therefore, we design two loss functions for these two tasks separately. Although the extraction of rank-1 features can be obtained using a pre-trained model, we combine it in the generator network so that we can reuse the intermediate rank-1 feature data in the training process. Therefore, the generator has two loss functions and two back-propagation steps to update their associated parameters separately.

Loss function for low-resolution approximation (rank-1 matrix features) Inspired by NMF, the approximate low-resolution Hi-C matrix is calculated as a combination of rank-1 matrices. To

estimate these rank-1 matrices, we include both pixel-wise MSE loss and structural dissimilarity (DSSIM) measures in the loss function. The DSSIM metric is derived from the structural similarity (SSIM) metric [86] to quantify the perceptual differences between two images. Specifically, $\text{DSSIM} = \frac{1-\text{SSIM}}{2} \in [0, 1]$. As described above, the generator may involve more than one downscaled representation of the low-resolution input, so we denote the factor set as $\mathbf{f} = [f_1, \cdots, f_K]$ and the corresponding weights for the downscaled matrices as $\mathbf{w} = [w_1, \cdots, w_K]$, where $w_k = \frac{f_k^2}{\sum_k f_k^2}$. The loss function of rank-1 feature extraction is:

$$\ell_{G_1}(\hat{I}, I) = \sum_{k=1}^{K} w_k \left[\ell_{\text{MSE}}(\hat{I}, I) + \text{DSSIM}(\hat{I}, I))\right].$$

In our application, we downscale the low-resolution input matrix by two different factors. Hence, $K = 2$, $f_1 = 2$, and $f_2 = 4$.

Loss function for high-resolution enhancement In the second part of the generator, we feed the rank-1 matrix features extracted from multiple downscaled low-resolution data into several sub-pixel layers to enhance matrix resolution. The loss function for the prediction of a high-resolution matrix consists of the pixel-wise MSE loss and the adversarial loss:

$$\ell_{G_2}(I_{\text{SR}}, I_{\text{HR}}) = \alpha_0 \ell_{\text{MSE}}(I_{\text{SR}}, I_{\text{HR}}) + \alpha_1 \ell_{\text{adv}},$$

where the α_0 and α_1 are hyperparameters.

The adversarial loss ℓ_{adv} is a crucial part of the GAN framework that connects the generator and discriminator networks. For the generator, minimizing the loss is equivalent to minimizing the binary cross-entropy loss between the true label ($\mathbf{y}$) and the prediction ($\mathbf{x}$) of generated Hi-C matrices by the discriminator. That is, $\ell_{\text{bce}}(\mathbf{y}, \mathbf{x}) = -\frac{1}{N}\sum_{n=1}^{N}(y_n \cdot \log(x_n) + (1 - y_n) \cdot \log(1 - x_n))$.

To disorient the discriminator, all labels of the predicted matrices are set to true. More details on the adversarial loss are discussed in Section 4.1.5.

$$\ell_{\text{adv}} = \ell_{\text{bce}}(\mathbf{1}, D(G(I_{\text{LR}})))$$

$$= -\log(D(I_{\text{SR}}))$$

4.1.4 Discriminator

The discriminator aims to differentiate between high-resolution predictions from the generator and real high-resolution data. In our EnHiC model, the discriminator shares the same strategy of the multi-scale rank-1 approximation as the generator, as illustrated in Figure 4.1. First, the input matrix is converted to multiple downscaled tensors by space-to-depth layers (in the *Downsampling Block*) and the rank-1 matrix features are subsequently extracted from each of the downscaled tensors (in the *Decomposition & Reconstruction Block*). In our design, we extract rank-1 features from the original matrix as well as three downscaled matrices (by a factor of 2, 4, and 8, respectively). Second, these rank-1 matrix features are passed into a cascade of *Convolutional Blocks* to detect latent features at multiple resolutions. As shown in Figure 4.1, each *Convolutional Block* includes a Leaky ReLU layer, a max-pooling layer, and a 2D convolution layer. After pooling and convolution, the dimensions of rank-1 matrix features are reduced by a factor of 2. These higher-resolution features are then concatenated with lower-resolution features and passed into the subsequent *Convolutional Block*. Finally, after a fully connected layer, the discriminator outputs the probability that the input is real, that is, the true high-resolution data rather than a prediction from the generator.

4.1.5 Loss function of the discriminator

In the training process, the generator and discriminator compete with each other and are connected by a MinMax loss. The generator tries to minimize the following function while the discriminator attempts to maximize it:

$$\min_G \max_D \mathbb{E}_{I_{\text{HR}}} \left[\log\left(D(I_{\text{HR}})\right)\right] + \mathbb{E}_{I_{\text{LR}}} \left[\log\left(1 - D(G(I_{\text{LR}}))\right)\right],$$

where $D(\cdot)$ is the estimated probability by the discriminator. $\mathbb{E}_{I_{\text{HR}}}$ is the expected value over all true instances. $G(I_{\text{LR}})$ is the generator's output when fed with the low-resolution Hi-C matrix I_{LR}, which is also called the super-resolution Hi-C matrix I_{SR}. $\mathbb{E}_{I_{\text{LR}}}$ is the expected value over all generated instances.

The GAN framework has two adversarial loss functions: one for generator training (as discussed in Section 4.1.3) and one for discriminator training. The discriminator aims to maximize $\mathbb{E}_{I_{\text{HR}}} \left[\log\left(D(I_{\text{HR}})\right)\right] + \mathbb{E}_{I_{\text{LR}}} \left[\log\left(1 - D(G(I_{\text{LR}}))\right)\right]$. Thus, the adversarial loss of the discriminator can be expressed as a combination of two binary cross-entropy losses:

$$\begin{aligned}\ell_{\text{D}} &= \ell_{\text{bce}}(\mathbf{1}, D(I_{\text{HR}})) + \ell_{\text{bce}}(\mathbf{0}, D(G(I_{\text{LR}}))) \\ &= -\log(D(I_{\text{HR}})) - \log(1 - D(I_{\text{SR}}))\end{aligned}$$

4.2 Results

4.2.1 EnHiC accurately predicts high-resolution Hi-C matrices

First, we sought to evaluate the enhancement capability of our EnHiC model against two other GAN-based models, Deephic and HiCSR. It has been shown that Deephic and HiCSR outperformed previously proposed models, including HiCPlus, HiCNN, and hicGAN. Therefore,

these models were not included in our evaluation. All three models, EnHiC, Deephic, and HiCSR, were trained to predict a high-resolution (10kb) Hi-C matrix from a low-resolution (40kb) Hi-C matrix. In other words, the desired resolution enhancement factor was 16.

Data preprocessing

In our validation experiments, we used three published Hi-C datasets in different human cell lines: GM12878 (lymphoblastoid cells), IMR90 (lung fibroblast cells), and K562 (leukemia cells) [59]. Among them, the GM12878 dataset has the highest number of chromatin contacts (2.88 billion), followed by IMR90 (0.76 billion) and K562 (0.62 billion) (Appendix Table B.1). High-resolution (10kb) Hi-C matrices were obtained from the cooler database [1]. Low-resolution Hi-C matrices were generated using a random downsampling procedure. Here we used the default downsampling ratio of 16. In other words, the sequencing depth in the resulting low-resolution matrices was $1/16$ of the high-resolution data.

First, we trained the three models (EnHiC, Deephic, and HiCSR) on the most deeply sequenced Hi-C data generated from GM12878 cells. We used chromosomes 1-16 for training, chromosomes 17 and 18 for hyperparameter tuning, and chromosomes 19-22 and X for evaluation. After model training in the GM12878 data, we applied the three methods to the IMR90 and K562 data to investigate the enhancement performance across different cell types.

The raw Hi-C matrix contains various types of technical and biological biases. Therefore, normalization is an essential step in Hi-C data analysis. Many normalization methods based on matrix-balancing approaches have been proposed [25, 32, 44, 36, 65]. In the EnHiC model, we employ the Sequential Component Normalization (SCN) method [65] to normalize the input Hi-C matrix. The Deephic and HiCSR models do not require Hi-C-specific normalization of the input

matrix. Instead, Deephic uses the min-max normalization to scale the input data. HiCSR first conducts a log1p transformation (i.e., $\log(1 + x)$) and then a min-max normalization of the input data.

After normalization, the intra-chromosomal Hi-C matrices were divided into small pieces (submatrices of size $n \times n$) for both training and testing. Here, we set $n = 400$. Specifically, EnHiC first divides the Hi-C matrix into non-overlapping submatrices of size $\frac{n}{2} \times \frac{n}{2}$ and then combines two diagonal submatrices with their off-diagonal interacting submatrix to form an $n \times n$ matrix. This operation ensures that the resulting submatrices are symmetric. Deephic divides the Hi-C matrix into non-overlapping submatrices of size 40×40. HiCSR divides the Hi-C matrix into partially overlapping submatrices of size 40×40 with a step size of 28×28. Therefore, the input submatrices are of size 40×40 and the output submatrices are of size 28×28. Because the average TAD size is less than 1 Mb and most of the significant interactions are located inside TADs, we omitted submatrices with the genomic distances greater than 2 Mb.

Training and prediction

The EnHiC model was implemented in Python 3 with TensorFlow2; and the source code is available at `https://github.com/wmalab/EnHiC`. Both the training and prediction processes of the three assessed models were conducted on Intel Haswell CPU and NVIDIA Tesla K80 GPU with 128 GB of memory. For EnHiC, the number of epochs for training was set to 300 with parameters $\alpha_0 = 10$ and $\alpha_1 = 0.1$. The runtime of the training process was approximately 85 hours (17 min per epoch). More training details, including the configuration and visualization generated by TensorBoard, are available in Appendix. The runtimes for HiCSR (500 epochs) and Deephic (800 epochs) were approximately 2 to 4 days.

Model validation and evaluations in GM12878 data

After the training step, we first applied the three models (EnHiC, Deephic, and HiCSR) to the evaluation set (chromosomes 19-22 and X) in human GM12878 data to enhance the resolution of low-resolution Hi-C matrices (downsampled from high-resolution Hi-C matrices by a factor of 16). We denote the 10kb high-resolution Hi-C matrices obtained from the cooler database as the ground truth.

For each chromosome, we assembled the predicted submatrices into one intra-chromosomal matrix. Because different models use different normalization procedures, it is necessary to reverse the normalizations to facilitate a fair comparison with the same ground truth. Denote the model output as $\mathbf{X}$, and de-normalized result as $\widetilde{\mathbf{X}}$.

- Deephic uses the min-max normalization. Hence, the reversion is $\widetilde{\mathbf{X}} = \max \mathbf{X} + \min$, where $\max$ and $\min$ are maximal and minimal values in the ground truth, respectively.

- HiCSR uses both the log1p transformation and the min-max normalization. Therefore, the reversion is $\widetilde{\mathbf{X}} = e^{(\max \mathbf{X} + \min)} - 1$, where $\max$ and $\min$ are the maximal and minimal log1p values in the ground truth.

- EnHiC uses the SCN normalization, therefore the reversion is $\widetilde{\mathbf{X}} = \mathbf{X} \oslash \mathbf{b}\mathbf{b}^\top$, where $\mathbf{b}$ is the bias vector estimated from the ground truth using the SCN method and $\oslash$ is the element-wise division. In the form of each element, we have $\widetilde{X}_{ij} = \frac{X_{ij}}{b_i b_j}$.

After reverse normalization, we evaluated the prediction results of the three models with the ground truth using four metrics: two classic pixel-wise numeric errors (MAE and MSE) and two Hi-C-specific similarity metrics: HiCRep [90] and GenomeDISCO [78]. Appendix Table B.2

summarizes the MAE and MSE measurements of the EnHiC, Deephic, and HiCSR predictions. Overall, EnHiC achieved the best performance with the lowest MAE and MSE errors. We noticed that MAE and MSE errors were inflated in Deephic and HiCSR predictions. This is likely due to the reverse normalization procedure, where the MAE and MSE errors were amplified by the max value and exponential operation. Therefore, the MAE and MSE metrics were not effective in assessing the performance of the Hi-C enhancement. We present the results for reference because MAE is a component of the loss function in the HiCSR model, and MSE is included in the loss functions in both EnHiC and Deephic.

In addition to the MAE and MSE metrics, we also considered two popular similarity measurements specifically designed for assessing reproducibility of Hi-C matrices, HiCRep [90] and GenomeDISCO [78]. HiCRep calculates a stratum-adjusted correlation coefficient (SCC) between two Hi-C matrices. The resulting SCC values range from -1 to 1, where a larger SCC value indicates a higher similarity between the two matrices. GenomeDISCO treats the Hi-C matrix as a network; it applies random walks on the network to smooth the data and then calculates a reproducibility score at multiple scales. Similar to HiCRep, GenomeDISCO scores also range from -1 to 1, where higher scores representing the higher reproducibility. Besides HiCRep and GenomeDISCO, HiC-Spector [89] is another Hi-C reproducibility metric. HiC-Spector applies the adjacency matrix to impute missing values and then calculates a similarity score between two full matrices. In our experiments, since we only predicted a strip of data in the full matrix (i.e., submatrices with genomic distances shorter than 2 Mb), HiC-Spector is not applicable in our evaluation.

Table 4.1 summarizes the HiCRep and GenomeDISCO evaluation results of EnHiC, Deephic, and HiCSR. As shown in Table 4.1, The HiCRep SCC scores were greater than 0.94 for all

	HiCRep			GenomeDISCO		
Chromosome	EnHiC	Deephic	HiCSR	EnHiC	Deephic	HiCSR
19	**0.972**	0.942	0.970	**0.83**	0.768	0.677
20	**0.972**	0.941	0.967	**0.837**	0.777	0.65
21	**0.973**	0.966	0.968	**0.816**	0.771	0.636
22	**0.978**	0.974	0.973	**0.844**	0.786	0.716
X	**0.949**	0.930	0.945	**0.781**	0.743	0.639

Table 4.1: **Evaluation of high-resolution Hi-C matrices predicted by EnHiC, Deephic, and HiCSR.** Three models are evaluated on chrosomomes 19-22 and X in human GM12878 Hi-C data. Each model prediction result is compared against the ground truth, and the HiCRep and GenomeDISCO scores are calculated. The highest HiCRep and GenomeDISCO scores are highlighted in bold.

three methods, indicating that their high-resolution predictions are very similar to the ground truth. Among them, our EnHiC model achieved the highest HiCRep SCC values and GenomeDISCO scores for all five test chromosomes. These results demonstrated that EnHiC can accurately and robustly enhance the resolution of Hi-C matrices and outperformed existing GAN-based models.

Performance on IMR90 and K562 data

In the previous section, we have demonstrated the capability of EnHiC in recovering high-resolution Hi-C matrices from low-resolution input data. We then asked whether EnHiC can enhance Hi-C matrix resolution across different cell types. Towards this goal, we applied three models (EnHiC, Deephic, and HiCSR) that were previously trained on the deeply sequenced

GM12878 (lymphoblastoid cells) dataset to two other less-sequenced Hi-C datasets: IMR90 (lung fibroblast cells), and K562 (leukemia cells). The same data preprocessing was performed in each cell type; and HiCRep and GenomeDISCO similarity scores were calculated to evaluate the model predictions.

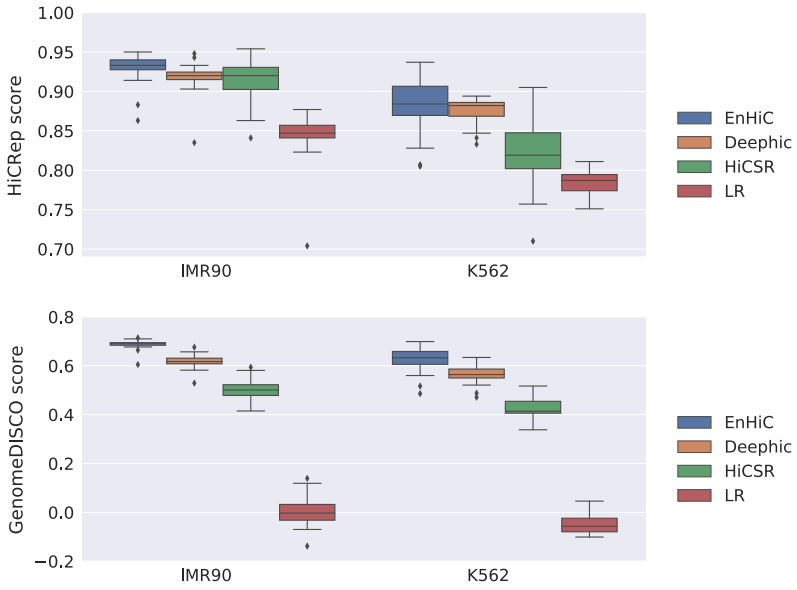


Figure 4.2: **The performance of predictions in different cell lines**. Evaluation of high-resolution Hi-C matrix predictions by EnHiC, Deephic, and HiCSR on human IMR90 and K562 Hi-C data (23 chromosomes). The models are first trained on GM12878 data and then applied to the other cell types. Each prediction result is compared against the ground truth, and the HiCRep and GenomeDISCO similarity scores are reported. Each box represents similarity scores of 23 chromosomes (1-22 and X). Low-resolution (LR) input data are included as the baseline.

Figure 4.2 illustrates the cross-cell-type performance of EnHiC, Deephic, and HiCSR. Overall, EnHiC outperformed both Deephic and HiCSR with the highest HiCRep and GenomeDISCO scores in both IMR90 and K562 datasets. We observed that the HiCRep and GenomeDISCO simi-

larity scores were relatively lower than the ones previously obtained from GM12878 data, but they were significantly higher that the baseline (low-resolution input data). In addition, the performance of all three models were slightly better in IMR90 than K562. This is likely due to the relatively higher sequencing depth in the IMR90 data (Appendix Table B.1). Taken together, these results indicated that EnHiC can effectively recover high-resolution matrices from insufficiently sequenced Hi-C data across cell types.

Performance on different downsampling ratios

In the training process, we generated low-resolution Hi-C matrices that were $16\times$ downsampled from high-resolution ground truth, i.e., the sequencing depth of the low-resolution input data was $1/16$ of the high-resolution data. We set the downsampling ratio at 16 to facilitate a fair comparison with previously published methods (Deephic and HiCSR). Although being trained by $16\times$ downsampled data, our EnHiC model is flexible and can be applied to low-resolution data with much less sequencing depth. Next, we sought to investigate the performance of our model using low-resolution input data generated with different downsampling ratios.

In this experiment, we generated low-resolution input data at six different downsampling ratios (4, 8, 16, 32, 48, and 64). We trained three models (EnHiC, Deephic, HiCSR) on the human GM12878 data using the same training set (chromosomes 1-16) and validation set (chromosomes 17-18) at $16\times$ downsampled ratio as previously described. We then evaluated the model performance using all 23 chromosomes at six different downsampled ratios, except for the $16\times$ downsampled data where the 18 training and validation chromosomes were excluded.

As shown in Figure 4.3, the HiCRep and GenomeDISCO similarity scores of low-resolution input baseline decreased sharply as the downsampling ratio increased. Notably, our

EnHiC model robustly and stably recovered high-resolution Hi-C matrices from low-resolution input data with large downsampled ratios. Moreover, EnHiC achieved higher HiCRep and GenomeDISCO scores than DeepHiC and HiCSR at almost all downsampled ratios. Although HiCSR performed slightly better than EnHiC by the HiCRep metric when the downsampling ratio was 4, its performance dropped sharply when the downsampling ratio increased. This is probably due to the pre-trained denoise model used in the loss function of HiCSR. Collectively, these results demonstrated that EnHiC can successfully predict high-resolution Hi-C matrices from insufficiently sequenced low-resolution data.

4.2.2 EnHiC facilitates accurate detection of TADs

TADs are functional units of chromatin, where chromatin interactions are observed more frequently within TADs than outside TADs. TAD boundaries are largely conserved across cell types and are enriched with CTCF and other chromatin-binding proteins [17]. To investigate whether high-resolution enhancing methods promote TAD detection, we compared the TADs identified from high-resolution predictions by EnHiC, Deephic, and HiCSR, with the TADs identified from the true high-resolution data.

Several computational methods exist for detecting TADs in Hi-C contact maps. Here, we used the hicFindTADs method in the HiCExplorer package [87]. We calculated Jaccard scores to assess the consistency between TADs detected from model predictions and TADs detected from true high-resolution (HR) data. The Jaccard score measures the similarity between two sets and is defined as the ratio of the intersection size over the union size. Jaccard score has been commonly used to quantify similarities of TAD and chromatin loop detections [22, 20]. Here we calculated

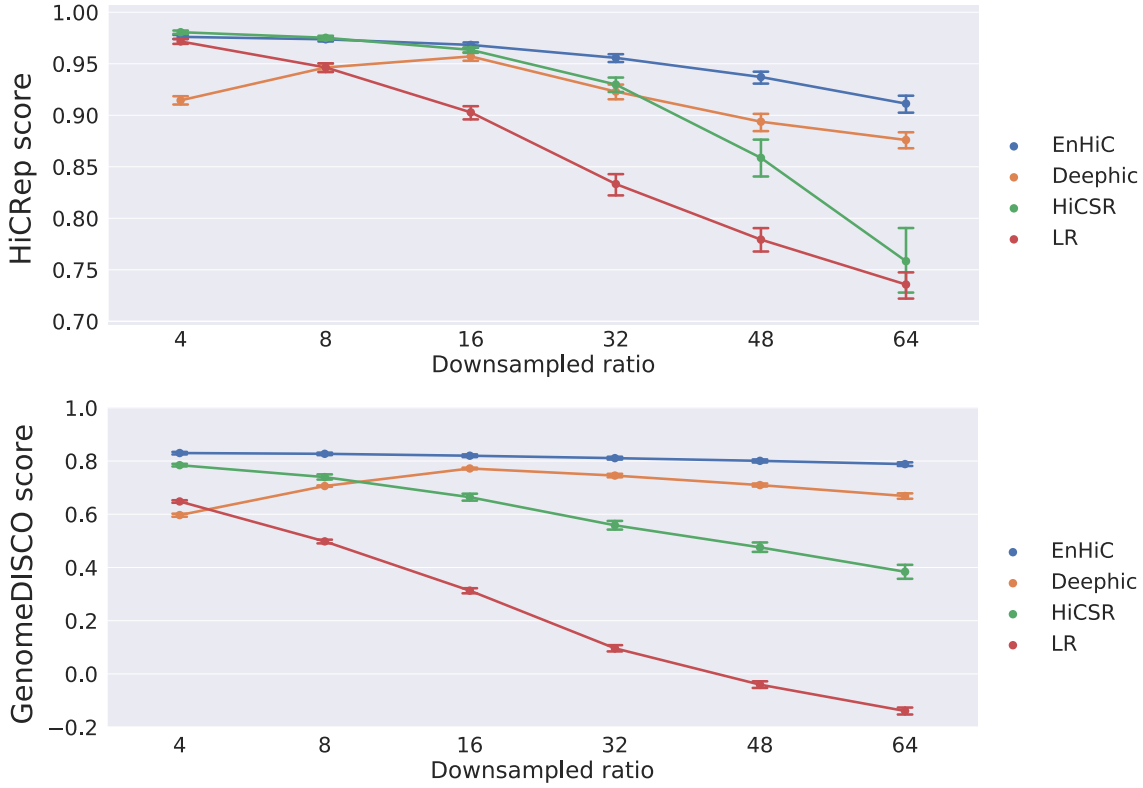


Figure 4.3: **The performance of predictions in different sequencing depths**. Performance of high-resolution Hi-C matrix predictions by EnHiC, Deephic, and HiCSR on GM12878 data at various downsampling ratios (4, 8, 16, 32, 48, and 64). Each prediction result is compared against the ground truth; and the HiCRep and GenomeDISCO reproducibility scores are reported. The mean values and error bars are calculated using scores from 23 chromosomes (1-22 and X). Low-resolution (LR) input data are included as the baseline.

Jaccard scores of TAD boundaries and allowed the boundaries to be shifted within 5 bins between the two sets.

$$\text{Jaccard score} = \frac{\text{TAD}_{\text{HR}} \cap \text{TAD}_{\text{prediction}}}{\text{TAD}_{\text{HR}} \cup \text{TAD}_{\text{prediction}}}$$

Figure 4.4 illustrates the Jaccard score evaluation of various methods in the validation dataset (chromosomes 17 and 18) and the test dataset (chromosomes 19-22, and X). The TADs detected from low-resolution input matrices were also included as baselines. Overall, EnHiC promoted accurate TAD detection; and the identified TADs were highly consistent with the ones identified from the true high-resolution data. In most cases, except for chromosome 21, high-resolution predictions from GAN-based models resulted in more accurate TAD detection than low-resolution input matrices (Figure 4.5). Overall, EnHiC yielded the highest Jaccard scores for five out of seven chromosomes, and outperformed both Deephic and HiCSR.

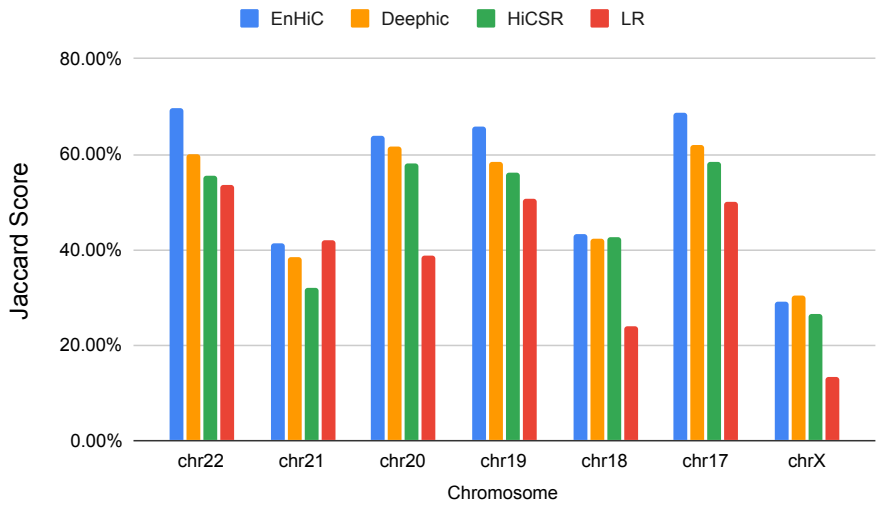


Figure 4.4: **The Jaccard scores of TADs**. TADs detected from high-resolution predictions by EnHiC, Deephic, and HiCSR were compared with TADs detected from real high-resolution (10kb) Hi-C data, for chromosomes 17-22, and X. TAD detection results from low-resolution (LR) input data were also included.

We also characterized the ChIP-seq profiles of several chromatin structural proteins and histone marks at the detected TAD boundaries in EnHiC-predicted matrices (Appendix B.6). Con-

sistent with the previous findings [17], we observed that CTCF, members of the cohesin complex (SMC3 and RAD21), RNA polymerase PolII binding, and H3K4me3 and H3K27me3 histone modifications were enriched at TAD boundaries, whereas H3K9me3 was depleted at such boundaries.

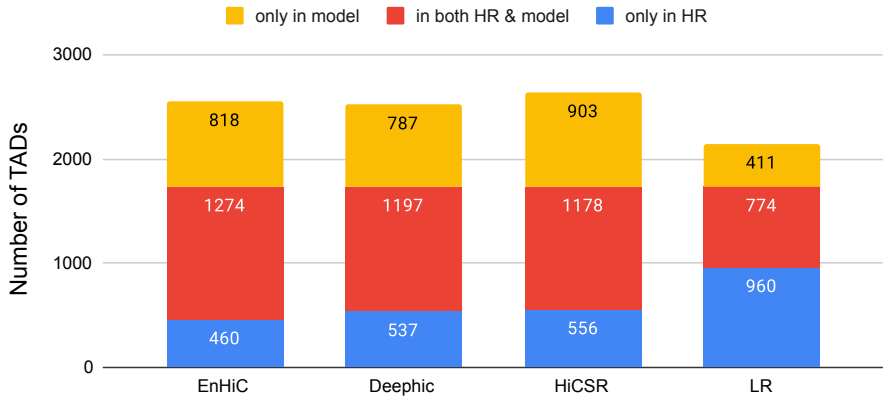


Figure 4.5: **Detection of TADs**. Numbers of TADs detected by each model. The results of seven chromosomes (17-22, and X) are summed. The red bars represent common TADs in both the true high-resolution (HR) matrices and model predictions. The blue (yellow) bars represent unique TADs detected only in the HR (predicted) matrices.

We further examined TAD detection results in two local regions (chr17:72-74 Mbp and chr19:14-16 Mbp), as illustrated in Appendix Figure B.8. The low-resolution input matrices are sparse and noisy; therefore, the detected TADs are often merged or split. Our EnHiC model accurately predicted high-resolution matrices from low-resolution input data. As a result, the TADs detected from EnHiC predictions were in agreement with the TADs from the true high-resolution data in both examples. We observed that both Deephic and HiCSR predictions overinflated the contact frequencies and Deephic predictions contained unwanted image textures, thereby resulting in inaccurate TAD detection.

4.2.3 EnHiC-predicted high-resolution matrices promote precise identifications of significant chromatin interactions

Next, we investigated whether the EnHiC-predicted high-resolution Hi-C data could facilitate the identification of fine-scale chromatin loops. We applied Fit-Hi-C [4] to identify significant interactions within 1 Mb genomic distances and compared the overlaps between the real and predicted Hi-C matrices. The Jaccard score was used to assess consistency between the significant interactions in the two matrices.

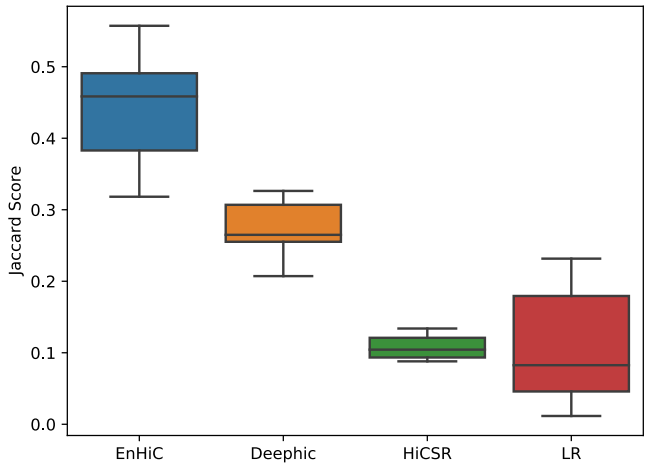


Figure 4.6: **The Jaccard scores** of significant interactions between the true high-resolution Hi-C and model predictions. The results from low-resolution (LR) input data were included as baseline. Each box depicts the Jaccard scores of seven chromosomes (17-22, and X).

As shown in Figure 4.6, EnHiC evidently outperformed the other two GAN-based prediction models with significantly higher Jaccard scores (t-tests, p-values: 2.57×10^{-4} (EnHiC vs. Deephic), 1.23×10^{-7} (vs. HiCSR), and 4.98×10^{-6} (vs. LR)). The low-resolution Hi-C input matrices lack sufficient sequencing depth; therefore, they are not suitable for the identification

of fine-scale chromatin interactions, especially when the genomic distance increases (Appendix Figure B.7).

We further looked at two example regions, chromosome 17:32-34Mbp (Figure 4.7) and chromosome 19:14-16 Mbp (Figure 4.8). As demonstrated in both regions, EnHiC successfully recovered the high-resolution matrices and produced highly similar chromatin loop identifications as those identified from real high-resolution data. As previously observed, Deephic and HiCSR tended to overinflate the contact matrix, thereby leading to a large number of false discoveries of significant interactions. The high false discovery rate is likely due to the preprocessing procedures or loss functions in these models. For example, HiCSR uses a log1p transformation in its preprocessing step, which may inflate low contact frequencies. In addition, Deephic uses a perceptual loss; as a result, its predictions contained unwanted image textual artifacts. Our EnHiC model is specifically designed to account for the unique data properties in the Hi-C matrix; therefore, the EnHiC-predicted matrices faithfully present high-resolution details in the Hi-C matrix.

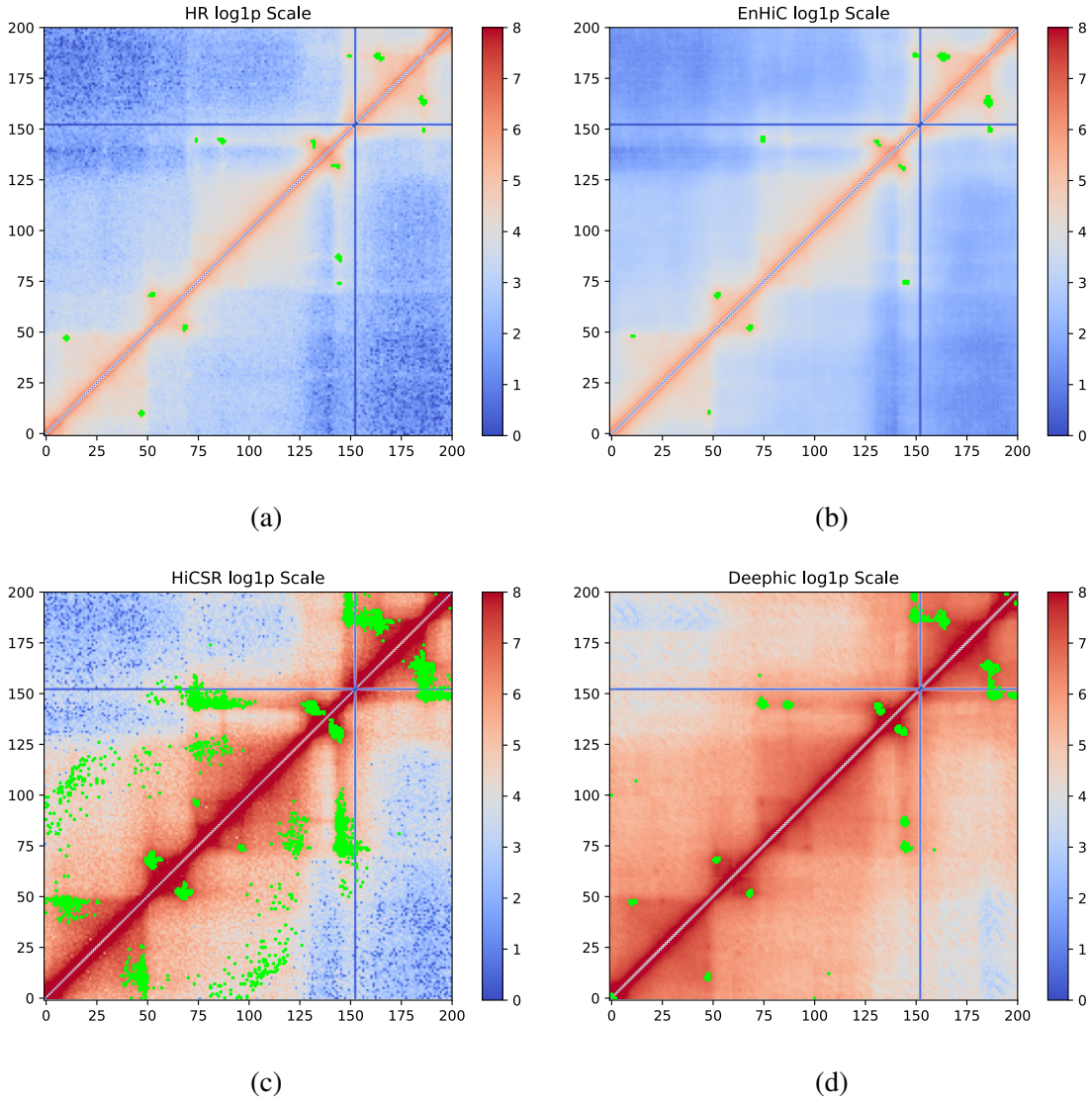


Figure 4.7: **Identification of significant interactions**. Significant chromatin interactions identified in chromosome 17 from 32Mbp to 34Mbp. **(a)** high resolution (HR) Hi-C at 10kb, **(b)** EnHiC prediction, **(c)** HiCSR prediction, and **(d)** Deephic prediction. Significant interactions were identified using FitHiC and are highlighted in green. Hi-C matrices are plotted on a log1p scale.

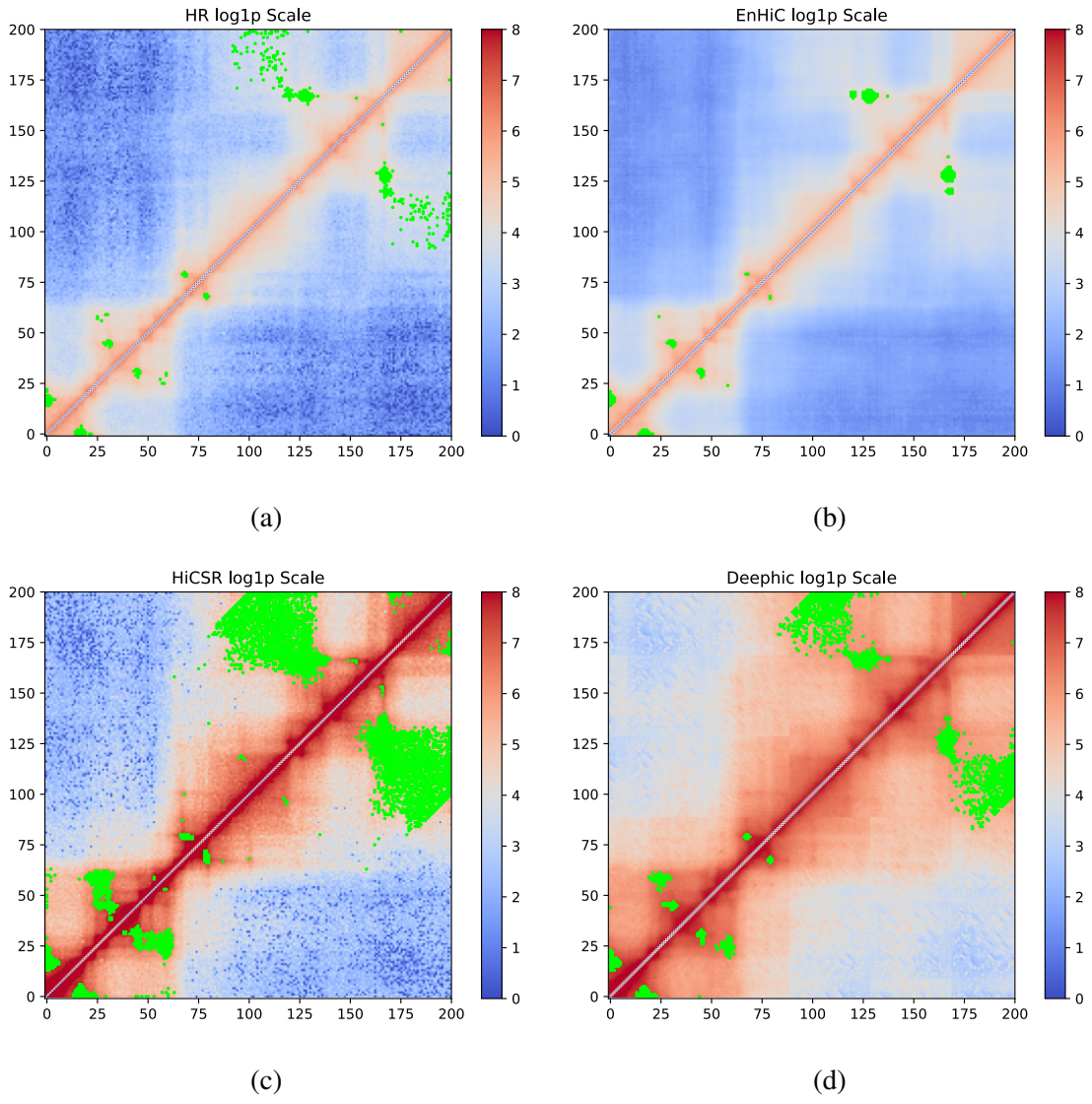


Figure 4.8: **Identification of significant interactions**. Significant chromatin interactions identified in chromosome 19 from 14Mbp to 16Mbp. **(a)** high resolution (HR) Hi-C at 10kb, **(b)** EnHiC prediction, **(c)** HiCSR prediction, and **(d)** Deephic prediction. Significant interactions were identified using FitHiC and are highlighted in green. Hi-C matrices are plotted on a log1p scale.

Chapter 5

Conclusions

In this study, we evaluated 3D modeling methods in the simulated data and real data. Then we proposed a 3D modeling method using a graph attention network which outperformed the top tanked methods in previous evaluations using independent real data at high resolution (10kb). We also proposed a generative adversarial framework to enhance the sequencing depth of the Hi-C matrix at high resolution (10kb).

5.1 Comparison of 3D modeling methods

In Chapter 2, we first made a comprehensive survey of publicly available Hi-C based chromatin 3D structure modeling methods. We focused on the 26 bulk Hi-C based modeling methods and explicitly compared their modeling pipelines. Based on the analysis of these methods, we selected 13 methods(15 results including two optimizations) from them to test their modeling performance. However, the established testing data and evaluating metrics don't offer a solid comparison result between the modeling structures from different tools. We developed a template-

based data simulation from haploid mouse single-cell Hi-C modeling structures by controlling the length and sparsity level to analyze the modeling performance in different cases. Then we proposed two coordinate-based evaluating metrics. The weighted RMSD estimates the weighted distance based on a global alignment. The DTM-score estimates the similarity between structures using the optimal local alignment. With the metrics, we comprehensively measure the performance for each method. Finally, we evaluated the local features of modeling results on independent data: LAD and FISH data.

In summary, most results from modeling methods are satisfied with the validation of independent data. The structures from LorDG, Pastis-PM2/PM1, ChromSDE, ShRec3D, and GEM got top ranks in the simulation data using two coordinate-based evaluating metrics.Most methods finished work within 1 day when structure ~ 1000 beads. Most of them cannot finish if structure larger than ~ 20000 beads within 2 weeks.

5.2 3D modeling for Hi-C at high-resolution

In Chapter 3, we proposed an Auto-Encoder framework, GIST, for predicting a population of 3D representations with proportions from bulk Hi-C matrix. Specifically, the Hi-C matrix is transformed into a heterogeneous graph, then GIST encodes the node features into 3D space by graph neural network and decodes the Euclidean distance between nodes into categories of edge. The representations are achieved by self-supervised learning through edge classification.

Existing modeling methods, such as ShRec3D, pastis, ChromSDE, GEM, and LorDG generate the "wish distance" from the Hi-C matrix and search the solution based on various objective functions. Most "wish distance" functions follow the power law, and pastis estimates the expected

contacts from Euclidean distance to fit the Hi-C contacts using the power-law too. Unlike other models, our GIST model utilizes the graph to connect Hi-C contact and Euclidean distance instead of specific functions.

Moreover, the consistent assumption-based methods are fast but abnegate the diversity in the Hi-C matrix. The inconsistent assumption-based methods. For example, ShRec3D generates the structure of chromosome 22 at 10kb resolution($\sim$3000 points) within 30 minutes, but GEM took two more days and five days for chromosome 20 at 10kb resolution($\sim$5000 points). It will consume much more time to model chromosome X($\sim$ 13000 points) at 10kb. That's why we don't evaluate the diversity of structures from GEM in Chromosome X. Our GIST model learns parameters from a set of sub-graphs instead of one graph of the entire chromosome and predicts the entire one. For the same chromosome 22, it took $\sim$ 30 seconds for one iteration and completed learning within 1 hour and predicted 40 structures within 5 minutes. It took around $\sim$ 6 hours for chromosome X.

We demonstrated the performance of our GIST model using the Hi-C dataset on human cell line: IMR90. We first validated our GIST model using independent FISH and compared it with the other five methods in terms of the partition of A/B compartments and relative TADs distance. Overall, our GIST model evidently outperformed others. Additionally, we demonstrated that GIST facilitated a diversity of conformations in the Hi-C matrix.

We envision a few possible extensions and future directions based on this work. First, the graph passed into the GIST model is from bulk Hi-C. It is possible to integrate types of spatial data in the graph, e.g., genome architecture mapping (GAM), SPRITE, and ChIA-Drop. Another integration is to leverage single-cell Hi-C matrices to lead the diversity of conformations. Second, hyper-parameter tuning is also one direction in the future. Currently, we applied AIC/BIC to

measure the performance of GMM in the preprocessing of Hi-C edge classification. However, other parameters in the model are selected arbitrarily, e.g., the number of conformations. It's essential to select the hyper-parameters automatically, e.g., Keras tuner or Ray tune.

5.3 Enhancement of Hi-C resolution

In Chapter 4, we proposed a generative adversarial framework, EnHiC, for predicting high-resolution Hi-C matrices from low-resolution input data. Specifically, high-resolution enhancement is achieved through the extraction of rank-1 matrix features from multi-scale low-resolution input samples and subsequent upsampling processes via sub-pixel CNN layers.

Existing resolution-enhancement models, such as Deephic and HiCSR, treat Hi-C matrices as single-channel images and leverage on the established neural networks of image super-resolution models. Although such models can produce super-resolution Hi-C matrices, their predictions often overinflate the Hi-C matrix features and sometimes contain unwanted natural image artifacts. Unlike other models, our EnHiC model utilizes the unique properties of Hi-C data.

Inspired by NMF, our EnHiC model uses similar notions of rank-1 features and matrix factorization. However, our model is different from NMF in the following aspects. First, our model attempts to decompose a set of submatrices instead of a full matrix. In the decomposition step, it searches for a rank-1 solution that fits all submatrices. Here we limit the rank to 1 to bypass the problem of picking the appropriate number of ranks in a low-rank solution. Second, our model optimizes the rank-1 matrix decomposition via the *Decomposition & Reconstruction Block* in the GAN framework. The difference between the input Hi-C matrix and its rank-1 approximation is characterized by a loss function consisting of the L2 MSE loss and structural dissimilarity.

We demonstrated the performance of our EnHiC model using Hi-C datasets on three human cell lines. Overall, our EnHiC model evidently outperformed two other GAN-based methods, Deephic and HiCSR, achieving low prediction errors and high reproducibility scores when compared with the true high-resolution data. Moreover, the EnHiC model is capable of recovering high-resolution Hi-C matrices across different cell types and from insufficiently sequenced input data. Additionally, we demonstrated that EnHiC-predicted matrices facilitated more accurate and precise detection of TADs and fine-scale chromatin interactions.

We envision a few possible extensions and future directions based on this work. First, EnHiC uses SCN normalization in the preprocessing step. The SCN normalization helps to reduce systematic biases in Hi-C data and rescales the intensity values to real numbers between [0,1]. It is possible to add alternative options of other Hi-C normalization methods in the implementation. And we do not expect the choice of normalization methods to have a major impact on the model performance. Second, EnHiC requires the input matrices to be symmetric. In our experiments, when dividing the entire Hi-C matrix into small submatrices, we merged two on-diagonal submatrices with one off-diagonal matrix to generate a symmetric matrix. This divide-and-merge strategy may cause artifacts at the edges of the submatrices. One possible future extension is to build a paired layer that simultaneously estimates the row and column vectors to relax the symmetry requirement. Third, to effectively extract multi-scale rank-1 features, large input matrices are recommended. In the current setting, we used 400×400 submatrices to achieve the desired enhancement factor of 16. Increasing the dimension of the input matrices would require more memory allocation and result in a heavier computation load. One possible future extension is to build a distributed implementation to mitigate the burden on each node.

Bibliography

[1] Nezar Abdennur and Leonid A Mirny. Cooler: scalable storage for hi-c data and other genomically labeled arrays. *Bioinformatics*, 36(1):311–316, 2020.

[2] Badri Adhikari, Tuan Trieu, and Jianlin Cheng. Chromosome3d: reconstructing three-dimensional chromosomal structures from hi-c interaction frequency data using distance geometry simulated annealing. *BMC genomics*, 17(1):886, 2016.

[3] Haitham Ashoor, Xiaowen Chen, Wojciech Rosikiewicz, Jiahui Wang, Albert Cheng, Ping Wang, Yijun Ruan, and Sheng Li. Graph embedding and unsupervised learning predict genomic sub-compartments from hic chromatin interaction data. *Nature communications*, 11(1):1–11, 2020.

[4] Ferhat Ay, Timothy L Bailey, and William Stafford Noble. Statistical confidence estimation for hi-c data reveals regulatory chromatin contacts. *Genome research*, 24(6):999–1011, 2014.

[5] Brian J Beliveau, Eric F Joyce, Nicholas Apostolopoulos, Feyza Yilmaz, Chamith Y Fonseka, Ruth B McCole, Yiming Chang, Jin Billy Li, Tharanga Niroshini Senaratne, Benjamin R Williams, et al. Versatile design and synthesis platform for visualizing genomes with oligopaint fish probes. *Proceedings of the National Academy of Sciences*, 109(52):21301–21306, 2012.

[6] Shay Ben-Elazar, Zohar Yakhini, and Itai Yanai. Spatial localization of co-regulated genes exceeds genomic gene clustering in the saccharomyces cerevisiae genome. *Nucleic acids research*, 41(4):2191–2201, 2013.

[7] Adrian W Bowman and Adelchi Azzalini. *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations*, volume 18. OUP Oxford, 1997.

[8] Christa Buecker, Rajini Srinivasan, Zhixiang Wu, Eliezer Calo, Dario Acampora, Tiago Faial, Antonio Simeone, Minjia Tan, Tomasz Swigut, and Joanna Wysocka. Reorganization of enhancer patterns in transition from naive to primed pluripotency. *Cell stem cell*, 14(6):838–853, 2014.

[9] Marcus Buschbeck, Iris Uribesalgo, Indra Wibowo, Pau Rué, David Martin, Arantxa Gutierrez, Lluís Morey, Roderic Guigó, Hernán López-Schier, and Luciano Di Croce. The histone

variant macroh2a is an epigenetic regulator of key developmental genes. *Nature structural & molecular biology*, 16(10):1074, 2009.

[10] Simeon Carstens, Michael Nilges, and Michael Habeck. Inferential structure determination of chromosomes from single-cell hi-c data. *PLoS computational biology*, 12(12):e1005292, 2016.

[11] Axel Cournac, Hervé Marie-Nelly, Martial Marbouty, Romain Koszul, and Julien Mozziconacci. Normalization of a chromosomal contact map. *BMC genomics*, 13(1):436, 2012.

[12] Thomas Cremer and Marion Cremer. Chromosome territories. *Cold Spring Harbor perspectives in biology*, 2(3):a003889, 2010.

[13] Elzo de Wit and Wouter De Laat. A decade of 3c technologies: insights into nuclear organization. *Genes & development*, 26(1):11–24, 2012.

[14] Job Dekker. The three'c's of chromosome conformation capture: controls, controls, controls. *Nature methods*, 3(1):17, 2006.

[15] Job Dekker. Gene regulation in the third dimension. *Science*, 319(5871):1793–1794, 2008.

[16] Michael C Dimmick, Leo J Lee, and Brendan J Frey. Hicsr: a hi-c super-resolution framework for producing highly realistic contact maps. *bioRxiv*, 2020.

[17] 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.

[18] Juliane C Dohm, Claudio Lottaz, Tatiana Borodina, and Heinz Himmelbauer. Substantial biases in ultra-short read data sets from high-throughput dna sequencing. *Nucleic acids research*, 36(16):e105, 2008.

[19] Z. Duan, M. Andronescu, K. Schutz, S. McIlwain, Y. J. Kim, C. Lee, J. Shendure, S. Fields, C. A. Blau, and W. S. Noble. A three-dimensional model of the yeast genome. *Nature*, 465:363–367, 2010.

[20] Mattia Forcato, Chiara Nicoletti, Koustav Pal, Carmen Maria Livi, Francesco Ferrari, and Silvio Bicciato. Comparison of computational methods for hi-c data analysis. *Nature methods*, 14(7):679–685, 2017.

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[22] Hao Hong, Shuai Jiang, Hao Li, Guifang Du, Yu Sun, Huan Tao, Cheng Quan, Chenghui Zhao, Ruijiang Li, Wanying Li, et al. Deephic: A generative adversarial network for enhancing hi-c data resolution. *PLoS computational biology*, 16(2):e1007287, 2020.

[23] Ming Hu, Ke Deng, Zhaohui Qin, Jesse Dixon, Siddarth Selvaraj, Jennifer Fang, Bing Ren, and Jun S Liu. Bayesian inference of spatial organizations of chromosomes. *PLoS computational biology*, 9(1):e1002893, 2013.

[24] Nan Hua, Harianto Tjong, Hanjun Shin, Ke Gong, Xianghong Jasmine Zhou, and Frank Alber. Pgs: a dynamic and automated population-based genome structure software. *bioRxiv*, page 103358, 2017.

[25] Maxim Imakaev, Geoffrey Fudenberg, Rachel Patton McCord, Natalia Naumova, Anton Goloborodko, Bryan R Lajoie, Job Dekker, and Leonid A Mirny. Iterative correction of hi-c data reveals hallmarks of chromosome organization. *Nature methods*, 9(10):999–1003, 2012.

[26] Maxim Imakaev, Geoffrey Fudenberg, Rachel Patton McCord, Natalia Naumova, Anton Goloborodko, Bryan R Lajoie, Job Dekker, and Leonid A Mirny. Iterative correction of hi-c data reveals hallmarks of chromosome organization. *Nature methods*, 9(10):999, 2012.

[27] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[28] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[29] R. Kalhor, H. Tjong, N. Jayathilaka, F. Alber, and L. Chen. Genome architectures revealed by tethered chromosome conformation capture and population-based modeling. *Nature biotechnology*, 30(1):90–98, 2012.

[30] Nicola A Kearns, Hannah Pham, Barbara Tabak, Ryan M Genga, Noah J Silverstein, Manuel Garber, and René Maehr. Functional annotation of native enhancers with a cas9–histone demethylase fusion. *Nature methods*, 12(5):401, 2015.

[31] Henk AL Kiers, Jos MF Ten Berge, and Rasmus Bro. Parafac2—part i. a direct fitting algorithm for the parafac2 model. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 13(3-4):275–294, 1999.

[32] Philip A Knight and Daniel Ruiz. A fast algorithm for matrix balancing. *IMA Journal of Numerical Analysis*, 33(3):1029–1047, 2013.

[33] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *arXiv preprint arXiv:1610.09555*, 2016.

[34] Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[35] Joseph B Kruskal. *Multidimensional scaling*. Number 11. Sage, 1978.

[36] Rajendra Kumar, Haitham Sobhy, Per Stenberg, and Ludvig Lizana. Genome contact map explorer: a platform for the comparison, interactive visualization and analysis of genome contact maps. *Nucleic acids research*, 45(17):e152–e152, 2017.

[37] Anshul Kundaje, Wouter Meuleman, Jason Ernst, Misha Bilenky, Angela Yen, Alireza Heravi-Moussavi, Pouya Kheradpour, Zhizhuo Zhang, Jianrong Wang, Michael J Ziller, et al. Integrative analysis of 111 reference human epigenomes. *Nature*, 518(7539):317, 2015.

[38] Jeffrey C Lagarias, James A Reeds, Margaret H Wright, and Paul E Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, 9(1):112–147, 1998.

[39] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[40] Da-Inn Lee and Sushmita Roy. Simultaneous smoothing and detection of topological units of genome organization from sparse chromatin contact count matrices with matrix factorization. *bioRxiv*, 2020.

[41] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[42] Annick Lesne, Julien Riposo, Paul Roger, Axel Cournac, and Julien Mozziconacci. 3d genome reconstruction from chromosomal contacts. *Nature methods*, 11(11):1141, 2014.

[43] Angsheng Li, Xianchen Yin, Bingxiang Xu, Danyang Wang, Jimin Han, Yi Wei, Yun Deng, Ying Xiong, and Zhihua Zhang. Decoding topologically associating domains with ultra-low resolution hi-c data by graph structural entropy. *Nature communications*, 9(1):1–12, 2018.

[44] 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.

[45] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[46] Qiao Liu, Hairong Lv, and Rui Jiang. hicgan infers super resolution hi-c data with generative adversarial networks. *Bioinformatics*, 35(14):i99–i107, 2019.

[47] Tong Liu and Zheng Wang. Hicnn: a very deep convolutional neural network to better enhance the resolution of hi-c data. *Bioinformatics*, 35(21):4222–4228, 2019.

[48] W. Ma, F. Ay, C. Lee, G. Gulsoy, X. Deng, S. Cook, J. Hesson, C. Cavanaugh, C. B. Ware, A. Krumm, J. Shendure, C. A. Blau, C. M. Disteche, W. S. Noble, and Z. Duan. Fine-scale chromatin interaction maps reveal the cis-regulatory landscape of lincRNA genes in human cells. *nmeth*, 12(1):71–78, 2015.

[49] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using convolutional auto-encoders with symmetric skip connections. *arXiv preprint arXiv:1606.08921*, 2016.

[50] Anzy Miller, Meryem Ralser, Susan L Kloet, Remco Loos, Ryuichi Nishinakamura, Paul Bertone, Michiel Vermeulen, and Brian Hendrich. Sall4 controls differentiation of pluripotent cells independently of the nucleosome remodelling and deacetylation (nurd) complex. *Development*, 143(17):3074–3084, 2016.

[51] Lluis Morey, Alexandra Santanach, Enrique Blanco, Luigi Aloia, Elphège P Nora, Benoit G Bruneau, and Luciano Di Croce. Polycomb regulates mesoderm cell fate-specification in embryonic stem cells through activation and repression mechanisms. *Cell stem cell*, 17(3):300–315, 2015.

[52] Kazuhiro Murakami, Ufuk Günesdogan, Jan J Zylicz, Walfred WC Tang, Roopsha Sengupta, Toshihiro Kobayashi, Shinseog Kim, Richard Butler, Sabine Dietmann, and M Azim Surani. Nanog alone induces germ cells in primed epiblast in vitro by activation of enhancers. *Nature*, 529(7586):403, 2016.

[53] Jackson Nowotny, Sharif Ahmed, Lingfei Xu, Oluwatosin Oluwadare, Hannah Chen, Noelan Hensley, Tuan Trieu, Renzhi Cao, and Jianlin Cheng. Iterative reconstruction of three-dimensional models of human chromosomes from chromosomal contact data. *BMC bioinformatics*, 16(1):338, 2015.

[54] Jincheol Park and Shili Lin. Statistical inference on three-dimensional structure of genome by truncated poisson architecture model. In *Ordered Data Analysis, Modeling and Health Research Methods*, pages 245–261. Springer, 2015.

[55] Jonas Paulsen, Odin Gramstad, and Philippe Collas. Manifold based optimization for single-cell 3d genome reconstruction. *PLoS computational biology*, 11(8):e1004396, 2015.

[56] Jonas Paulsen, Monika Sekelja, Anja R Oldenburg, Alice Barateau, Nolwenn Briand, Erwan Delbarre, Akshay Shah, Anita L Sørensen, Corinne Vigouroux, Brigitte Buendia, et al. Chrom3d: three-dimensional genome modeling from hi-c and nuclear lamin-genome contacts. *Genome biology*, 18(1):21, 2017.

[57] Cheng Peng, Liang-Yu Fu, Peng-Fei Dong, Zhi-Luo Deng, Jian-Xin Li, Xiao-Tao Wang, and Hong-Yu Zhang. The sequencing bias relaxed characteristics of hi-c derived data and implications for chromatin 3d modeling. *Nucleic acids research*, 41(19):e183–e183, 2013.

[58] Daan Peric-Hupkes, Wouter Meuleman, Ludo Pagie, Sophia WM Bruggeman, Irina Solovei, Wim Brugman, Stefan Gräf, Paul Flicek, Ron M Kerkhoven, Maarten van Lohuizen, et al. Molecular maps of the reorganization of genome-nuclear lamina interactions during differentiation. *Molecular cell*, 38(4):603–613, 2010.

[59] Suhas SP Rao, Miriam H Huntley, Neva C Durand, Elena K Stamenova, Ivan D Bochkov, James T Robinson, Adrian L Sanborn, Ido Machol, Arina D Omer, Eric S Lander, et al. A 3d map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell*, 159(7):1665–1680, 2014.

[60] Lila Rieber and Shaun Mahony. minimds: 3d structural inference from high-resolution hi-c data. *Bioinformatics*, 33(14):i261–i266, 2017.

[61] Michael Rosenthal, Darshan Bryner, Fred Huffer, Shane Evans, Anuj Srivastava, and Nicola Neretti. Bayesian estimation of three-dimensional chromosomal structure from single-cell hi-c data. *Journal of Computational Biology*, 2019.

[62] Mathieu Rousseau, James Fraser, Maria A Ferraiuolo, Josée Dostie, and Mathieu Blanchette. Three-dimensional modeling of chromatin structure from interaction frequency data using markov chain monte carlo sampling. *BMC bioinformatics*, 12(1):414, 2011.

[63] François Serra, Marco Di Stefano, Yannick G Spill, Yasmina Cuartero, Michael Goodstadt, Davide Baù, and Marc A Marti-Renom. Restraint-based three-dimensional modeling of genomes and genomic domains. *FEBS Letters*, 2015.

[64] François Serra, Davide Baù, Mike Goodstadt, David Castillo, Guillaume J Filion, and Marc A Marti-Renom. Automatic analysis and 3d-modelling of hi-c data using tadbit reveals structural features of the fly chromatin colors. *PLoS computational biology*, 13(7):e1005665, 2017.

[65] Nicolas Servant, Nelle Varoquaux, Bryan R Lajoie, Eric Viara, Chong-Jian Chen, Jean-Philippe Vert, Edith Heard, Job Dekker, and Emmanuel Barillot. Hic-pro: an optimized and flexible pipeline for hi-c data processing. *Genome biology*, 16(1):259, 2015.

[66] Sheel Shah, Yodai Takei, Wen Zhou, Eric Lubeck, Jina Yun, Chee-Huat Linus Eng, Noushin Koulena, Christopher Cronin, Christoph Karp, Eric J Liaw, et al. Dynamics and spatial genomics of the nascent transcriptome by intron seqfish. *Cell*, 174(2):363–376, 2018.

[67] Yoli Shavit, Fiona Kathryn Hamey, and Pietro Lio. Fishical: an r package for iterative fish-based calibration of hi-c data. *Bioinformatics*, 30(21):3120–3122, 2014.

[68] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[69] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[71] Tim J Stevens, David Lando, Srinjan Basu, Liam P Atkinson, Yang Cao, Steven F Lee, Martin Leeb, Kai J Wohlfahrt, Wayne Boucher, Aoife O'Shaughnessy-Kirwan, et al. 3d structures of individual mammalian genomes studied by single-cell hi-c. *Nature*, 544(7648):59, 2017.

[72] Quentin Szabo, Frédéric Bantignies, and Giacomo Cavalli. Principles of genome folding into topologically associating domains. *Science advances*, 5(4):eaaw1668, 2019.

[73] Przemyslaw Szalaj, Paul J Michalski, Przemysław Wróblewski, Zhonghui Tang, Michal Kadlof, Giovanni Mazzocco, Yijun Ruan, and Dariusz Plewczynski. 3d-gnome: an integrated web service for structural modeling of the 3d genome. *Nucleic acids research*, 44(W1):W288–W293, 2016.

[74] Zhonghui Tang, Oscar Junhong Luo, Xingwang Li, Meizhen Zheng, Jacqueline Jufen Zhu, Przemyslaw Szalaj, Pawel Trzaskoma, Adriana Magalska, Jakub Wlodarczyk, Blazej Ruszczycki, et al. Ctcf-mediated human 3d genome architecture reveals chromatin topology for transcription. *Cell*, 163(7):1611–1627, 2015.

[75] Tuan Trieu and Jianlin Cheng. Mogen: a tool for reconstructing 3d models of genomes from chromosomal conformation capturing data. *Bioinformatics*, 32(9):1286–1292, 2015.

[76] Tuan Trieu and Jianlin Cheng. 3d genome structure modeling by lorentzian objective function. *Nucleic acids research*, 45(3):1049–1058, 2016.

[77] O Ursu, N Boley, M Taranova, YXR Wang, GG Yardimci, WS Noble, et al. Genomedisco: a concordance score for chromosome conformation capture experiments using random walks on contact map graphs. biorxiv. 2017: 181842 available from: https://www. biorxiv. org/content/early/2017/08/29/181842.[cited 2018 jan 30].

[78] Oana Ursu, Nathan Boley, Maryna Taranova, YX Rachel Wang, Galip Gurkan Yardimci, William Stafford Noble, and Anshul Kundaje. Genomedisco: A concordance score for chromosome conformation capture experiments using random walks on contact map graphs. *Bioinformatics*, 34(16):2701–2707, 2018.

[79] N. Varoquaux, F. Ay, W. S. Noble, and J.-P. Vert. A statistical approach for inferring the 3D structure of the genome. *Bioinformatics*, 30(12):i26–i33, 2014.

[80] Nelle Varoquaux, Ferhat Ay, William Stafford Noble, and Jean-Philippe Vert. A statistical approach for inferring the 3d structure of the genome. *Bioinformatics*, 30(12):i26–i33, 2014.

[81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[82] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[83] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

[84] Siyu Wang, Jinbo Xu, and Jianyang Zeng. Inferential modeling of 3d chromatin structure. *Nucleic acids research*, 43(8):e54–e54, 2015.

[85] Siyuan Wang, Jun-Han Su, Brian J Beliveau, Bogdan Bintu, Jeffrey R Moffitt, Chao-ting Wu, and Xiaowei Zhuang. Spatial organization of chromatin domains and compartments in single chromosomes. *Science*, 353(6299):598–602, 2016.

[86] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[87] Joachim Wolff, Vivek Bhardwaj, Stephan Nothjunge, Gautier Richard, Gina Renschler, Ralf Gilsbach, Thomas Manke, Rolf Backofen, Fidel Ramírez, and Björn A Grüning. Galaxy hicexplorer: a web server for reproducible hi-c data analysis, quality control and visualization. *Nucleic acids research*, 46(W1):W11–W16, 2018.

[88] Eitan Yaffe and Amos Tanay. Probabilistic modeling of hi-c contact maps eliminates systematic biases to characterize global chromosomal architecture. *Nature genetics*, 43(11):1059, 2011.

[89] Koon-Kiu Yan, Galip Gürkan Yardımcı, Chengfei Yan, William S Noble, and Mark Gerstein. Hic-spector: a matrix library for spectral and reproducibility analysis of hi-c contact maps. *Bioinformatics*, 33(14):2199–2201, 2017.

[90] Tao Yang, Feipeng Zhang, Galip Gürkan Yardımcı, Fan Song, Ross C Hardison, William Stafford Noble, Feng Yue, and Qunhua Li. Hicrep: assessing the reproducibility of hi-c data using a stratum-adjusted correlation coefficient. *Genome research*, 27(11):1939–1949, 2017.

[91] Galip Gürkan Yardımcı, Hakan Ozadam, Michael EG Sauria, Oana Ursu, Koon-Kiu Yan, Tao Yang, Abhijit Chakraborty, Arya Kaul, Bryan R Lajoie, Fan Song, et al. Measuring the reproducibility and quality of hi-c data. *Genome biology*, 20(1):57, 2019.

[92] Ruochi Zhang and Jian Ma. Matcha: Probing multi-way chromatin interaction with hypergraph representation learning. *Cell systems*, 10(5):397–407, 2020.

[93] Ruochi Zhang, Yuesong Zou, and Jian Ma. Hyper-sagnn: a self-attention based graph neural network for hypergraphs. *arXiv preprint arXiv:1911.02613*, 2019.

[94] Yan Zhang, Lin An, Jie Xu, Bo Zhang, W Jim Zheng, Ming Hu, Jijun Tang, and Feng Yue. Enhancing hi-c data resolution with deep convolutional neural network hicplus. *Nature communications*, 9(1):1–9, 2018.

[95] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, 2004.

[96] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.

[97] ZhiZhuo Zhang, Guoliang Li, Kim-Chuan Toh, and Wing-Kin Sung. 3d chromosome modeling with semi-definite programming and hi-c data. *Journal of computational biology*, 20(11):831–846, 2013.

[98] Jingtian Zhou, Jianzhu Ma, Yusi Chen, Chuankai Cheng, Bokan Bao, Jian Peng, Terrence J Sejnowski, Jesse R Dixon, and Joseph R Ecker. Robust single-cell hi-c clustering by convolution- and random-walk–based imputation. *Proceedings of the National Academy of Sciences*, 116(28):14011–14018, 2019.

[99] Guangxiang Zhu, Wenxuan Deng, Hailin Hu, Rui Ma, Sai Zhang, Jinglin Yang, Jian Peng, Tommy Kaplan, and Jianyang Zeng. Reconstructing spatial organizations of chromosomes through manifold learning. *Nucleic acids research*, 46(8):e50–e50, 2018.

[100] Chenchen Zou, Yuping Zhang, and Zhengqing Ouyang. Hsa: integrating multi-track hi-c data for genome-scale reconstruction of 3d chromatin structure. *Genome biology*, 17(1):40, 2016.

Appendix A

Comparison of 3D modeling methods

A.1 Benchmark datasets

A.1.1 Single-cell chromatin structures

We evaluated chromatin structures compiled by different modeling methods against the structure templates reconstructed by [71]. Specifically, whole-genome structures at various resolutions from 100 kb to 8 Mb from each of the eight haploid mouse single-cells (GSE80280, as shown in Table A.1) were simulated via the method (NucDynamic [71]), a step-wise modeling method that simulates and refines structures from lower resolution to higher resolution. NucDynamic simulated the structures at 100k resolution and the refined structures at low resolutions, as the intermediate data, were also reconstructed in the duration. However, these resolutions cannot be hopped in one execution. The structures are consistent by fixing the random seed for initialization and all rounds started at 8M resolution. So that we have these structures as ground truth for 8 cells at 8 resolutions(8M, 4M, 2M, 1M, 800kb, 400kb, 200kb, 100kb).

The advantages of using these NucDynamic simulated single-cell haploid structures are several folds. First, the chromosomal structures generated by NucDynamic are stable and consistent across different resolutions and highly consistent with independent experimental datasets as demonstrated in [71]. Second, the structures were simulated from haploid cells, thereby avoiding the ambiguity of homologous structures in diploid cells. Third, single-cell Hi-C data presents a unique chromatin structure in contrast to bulk Hi-C data which presents an ensemble of chromatin structures. Lastly, both single-cell and bulk Hi-C data were available in the same cell type [71], which helps us estimate proper simulation parameters (Section A.2.1).

A.1.2 Hi-C datasets

In addition to simulation studies, we also evaluated the performance of the 3D chromatin structure modeling methods using several published Hi-C datasets (Table A.1). Briefly, we used the 3D structure predictions from mouse ESC Hi-C data [71] to evaluate the localization of LADs. Moreover, we used the modeling results from human GM12878 Hi-C data [59] for assessments of method stability across resolutions, software performance in terms of running time and memory, and for validation by 3D-FISH data. If no specified in document (we didn't go through the detail of implementations for all methods), the inputs of Hi-C are balanced matrices. MDS, NMDS, pastis-PM1, pastis-PM2 provide build-in functions for balancing the Hi-C matrix by iterative correction (ICE) [26] normalization. The TADbit has a interface for the raw Hi-C data. For the simulation Hi-C data and mouse ESC Hi-C data, the five methods above are fed by raw Hi-C data and the rest methods are fed by the data normalized by Sequential Component Normalization (SCN)[11]. The human GM12878 Hi-C data are normalized by KR before feeding into the modeling methods.

A.1.3 Orthogonal experimental datasets

We used two independent experimental datasets to further evaluate the inferred 3D chromatin structures: lamina-associated domains (LADs) and fluorescence *in situ* hybridization on three-dimensionally preserved nuclei (3D-FISH) data in Table 2.3. Briefly, we used the published LADs in mouse ESCs [37] to assess the spatial organizations of predicted chromosomal structures from bulk Hi-C data [71]. Additionally, the chromosomal structure predictions in human GM12878 cells were validated using published 3D-FISH data [59].

A.2 Simulation settings

In our simulation studies, we simulated Hi-C contact frequency matrices using the published whole genome structures by [71]. We then tested all 15 chromatin structure modeling methods on the simulated contact matrices and evaluated their predicted structures against the ground truth structures. We denote 8 genome-wide structures (20 chromosomes in each structure) at each resolution (100 kb, 200 kb, 400 kb, 800 kb, 1 Mb, 2 Mb, 4 Mb, and 8 Mb) as the ground truth. These structures were modeled by NucDynamic[71] using the 8 single-cell Hi-C datasets in haploid mouse cells, described in Section A.1.1. Moreover, we borrowed information from the published single-cell Hi-C and bulk Hi-C data from the same cell type [71] to estimate the simulation parameters.

At each resolution, we simulated Hi-C contact matrices using one single-cell structure template (single cell #3), as described in Section A.2.1. The evaluation metrics and strategies of the predicted 3D chromatin structures are discussed in Section A.3.

A.2.1 Simulating Hi-C from a single-cell structure template

Because all the 15 tested methods were specifically designed for bulk Hi-C data, it is improper to apply them on a single-cell contact matrix which is binary and highly sparse. Therefore, in our simulations we intend to simulate bulk-like Hi-C contact matrices from single-cell structure template at various sparsity levels.

Assuming the observed Hi-C contact frequency matrix contains a mixture of signal and noises, we simulated two matrices separately: one is the signal matrix $\mathbf{C^S}$ which presents the true underlying chromatin interactions, the other one is the noise matrix $\mathbf{C^N}$ which contains experimental biases and noises.

Simulating the signal matrix $\mathbf{C^S}$: To generate the signal matrix, we use the single-cell structure template $\mathbf{X}$ as the ground truth, and extract the information from the observed single-cell Hi-C contact frequency matrix $\mathbf{S}$ and the corresponding bulk Hi-C contact frequency matrix $\mathbf{P}$ in the same cell type. The simulated single-cell signal matrix $\mathbf{C^S}$ is constructed via the following steps:

1. Calculate the Euclidean distance matrix $\mathbf{D}$ of the single-cell structure template using its 3D coordinates $\mathbf{X}$. That is, $d_{ij} = \|x_i - x_j\|$, where x_i and x_j are the 3D coordinates of i-th and j-th bins in structure $\mathbf{X}$, respectively, $1 \leq i, j \leq n$.

2. Assume the relationship between genomic distance g_{ij} and Euclidean distance d_{ij} is $g_{ij} = \beta_1 d_{ij}^{\alpha_1}$ and the relationship between genomic distance g_{ij} and contact frequency c_{ij} is $c_{ij} = \beta_2 g_{ij}^{\alpha_2}$. Then the contact frequency can be calculated as $c_{ij} = \beta_1(\beta_2 d_{ij}^{\alpha_2})^{\alpha_1} = \beta d_{ij}^{\alpha}$, where $\alpha = \alpha_1\alpha_2$ and $\beta = \beta_1\beta_2$.

(a) Estimate α_1 and β_1 from the bulk Hi-C contact frequency matrix $\mathbf{P}$:

$$\alpha_1, \beta_1 = \text{argmin}_{\alpha_1,\beta_1} \sum_{i<j} (p_{ij} - \beta_1(j-i)^{\alpha_1})^2$$

(b) Estimate α_2 and β_2 from the Euclidean distance matrix $\mathbf{D}$:

$$\alpha_2, \beta_2 = \text{argmin}_{\alpha_2,\beta_2} \sum_{i<j} (d_{ij} - \beta_2(j-i)^{\alpha_2})^2$$

3. Assuming the contact counts as independent Poisson random variables, simulate the Hi-C contact frequency matrix $\mathbf{C}$ by $c_{ij} \sim \text{Poisson}(\lambda_{ij})$, where $\lambda_{ij} = \beta d_{ij}^{\alpha}$. Here, the values at main diagonal are sheared; the top 1% highest values are considered to be outliers and therefore trimmed.

4. Adjust the sparsity of the Hi-C contact frequency matrix $\mathbf{C}$ using a binary mask matrix $\mathbf{M}$ to obtain the filtered signal matrix $\mathbf{C^S}$. That is, $c_{ij}^S = c_{ij}$ if $m_{ij} = 1$; $c_{ij}^S = 0$, otherwise. In our experimental design, the lowest desired sparsity is the one of a single-cell Hi-C contact matrix and the highest desired sparsity is the full contact frequency matrix $\mathbf{C}$ generated from the previous step. Therefore, the binary mask matrix $\mathbf{M}$ is generated by adjusting β to achieve the desired 25%, 50%, 75% sparsity in between.

Simulating the noise matrix $\mathbf{C^N}$: Similar to the procedure developed by [91], we generate the noise matrix ($\mathbf{C^N}$) from the simulated full contact frequency matrix $\mathbf{C}$ by combining two types of Hi-C noise matrix: the genomic-distance noise matrix and the random-ligation noise matrix.

The first genomic-distance noise matrix reflects genomic distance effect, that is, the observed Hi-C contact frequency decreases as the genomic distance increases. Briefly, the genomic-distance noise matrix is generated by stratified shuffling each diagonal of the Hi-C matrix based on

the product of marginals. First, each diagonal of the Hi-C contact matrix is divided into multiple strata determined by the product of the corresponding row and column marginals. For a given genomic distance, the noise matrix were sampled within each stratum. Instead of uniform sampling from the entire diagonal, this stratified sampling strategy is specifically designed to account for the systematic biases such as GC content and mappability.

The second matrix, the namely random-ligation noise matrix, is simulated by generating Hi-C contacts between random bin pairs. The probability of selecting a bin pair is proportional to the product of the corresponding row or column marginals.

After combining these two matrices to obtain noise matrix $\mathbf{C^N}$, we add the signal matrix $\mathbf{C^S}$ and noise matrix ($\mathbf{C^N}$ to generate a series of bulk-like Hi-C contract matrices at various sparsity levels. We set the signal-to-noise ratio to be 10. That is, the total contact counts in the noise matrix is 10% of the one in the signal matrix.

A.3 Structure similarity metrics

In this section, we summarize three alignment-based structural similarity metrics (RMSD, weighted RMSD, and DTM-score) for the evaluation the prediction of 3D chromatin structures.

A.3.1 RMSD

The root mean square deviation (RMSD) metric [28] calculates the average distance between two structures. Given two structures denoted in their 3D coordinates $\mathbf{A}, \mathbf{B} \in R^{3\times n}$, their RMSD is defined as $\text{RMSD}(\mathbf{A}, \mathbf{B}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{a_i} - \mathbf{b_i}\|^2}$, where $\mathbf{a_i}, \mathbf{b_i} \in R^3$ are the i-th bead in structures $\mathbf{A}$ and $\mathbf{B}$, respectively, and $\|\cdot\|$ is the vector 2-norm for Euclidean distance calculation.

The RMSD metric has been widely used in comparing protein structures and chromatin structures. In practice, we translate both structures to the origin, and then rotate and scale one of the structures (referred as the input structure $\mathbf{M}$) with respect to the other one (referred as the template structure $\mathbf{T}$) so that the resulting RMSD of the two aligned structures is minimized. Assuming both $\mathbf{M}$ and $\mathbf{T}$ are already centered at the origin, we need to find the optimal rotation matrix $\mathbf{U} \in R^{3\times 3}$ and scaling factor s that minimizes RMSD$(s\mathbf{UM}, \mathbf{T})$.

The optimal rotation matrix $\mathbf{U}$ can be calculated analytically by the Kabsch algorithm [28], as described below. Note that, unlike protein structures, for chromatin structures we do not distinguish a native structure and its mirrored image. The optimal rotation matrix may involve possible reflection on the coordinate system.

1. Compute the covariance matrix $\mathbf{C} = \mathbf{MT}^\top$;
2. Perform singular value decomposition (SVD) of the covariance matrix $\mathbf{C} = \mathbf{VSW}^\top$;
3. Compute the rotation matrix $\mathbf{U} = \mathbf{WV}^\top$.

Once the rotation matrix $\mathbf{U}$ is obtained, we can search for the best scaling factor $s = \arg\min_s \text{RMSD}(s\mathbf{UM}, \mathbf{T})$, and then report the optimized RMSD score as RMSD$(s\mathbf{UM}, \mathbf{T})$.

Algorithm 1: RMSD fit, also known as the Kabsch algorithm

Input: input structure $\mathbf{M} \in R^{3\times n}$, template structure $\mathbf{T} \in R^{3\times n}$ (both centered at the origin)

Output: RMSD-score, scaling factor s, rotation matrix $\mathbf{U}$

1 Compute the covariance matrix $\mathbf{C} = \mathbf{M}\mathbf{T}^\top$;

2 Perform singular value decomposition (SVD) of the covariance matrix $\mathbf{C} = \mathbf{V}\mathbf{S}\mathbf{W}^\top$;

3 Compute the rotation matrix $\mathbf{U} = \mathbf{W}\mathbf{V}^\top$;

4 Search for the optimal scaling factor $s = \arg\min_s \text{RMSD}(s\mathbf{U}\mathbf{M}, \mathbf{T})$;

5 Compute RMSD-score $= \text{RMSD}(s\mathbf{U}\mathbf{M}, \mathbf{T})$.

Note that unlike protein structures, there is no standard unit for chromatin structures. In most cases, the scale of chromatin structures are arbitrarily determined by the modeling methods. As a result, when comparing two chromatin structures, the scale of the template structure impacts the the magnitude of the resulting RMSD. In other words, the RMSD metrics is not symmetric: the RMSD value of aligning structures $\mathbf{A}$ to $\mathbf{B}$ (where $\mathbf{B}$ is the template) does not always equal to the RMSD value of aligning $\mathbf{B}$ to $\mathbf{A}$ (where $\mathbf{A}$ is the template).

A.3.2 Weighted RMSD

The standard RMSD measure produces a global alignment between the two structures, which is very sensitive to local structural variations. To tackle this problem, we propose a novel strategy, named weighted RMSD, to align two structures by their distance correlation weighted RMSD fit.

The main idea of the weighted RMSD method is to assign higher weights to the beads that are very similar between two structures and lower weights to the beads that have larger displacement.

By such weighting strategy, we reduce the impact of variations between two structures on the scaling factor and rotation matrix, thereby improving the structure alignment. Specifically, for the i-th bead $\mathbf{m_i}$ in structure $\mathbf{M}$, we define its distance vector $\mathbf{d_{m_i}} = \{d_{m_{i1}}, \cdots, d_{m_{in}}\}$, where $d_{m_{ij}} = \|\mathbf{m_i}, \mathbf{m_j}\|$. In other words, $\mathbf{d_{m_i}}$ represents the relative spatial position of the i-th bead in structure $\mathbf{M}$ with respects to all other beads. Similarly, the distance vector $\mathbf{d_{t_i}}$ can be calculated for the corresponding i-th bead in structure $\mathbf{T}$. We then calculate the Pearson correlation coefficient between the pair of distance vectors $\rho_i = \text{corr}(\mathbf{d_{t_i}}, \mathbf{d_{m_i}})$. A positive ρ_i indicates that the two beads share similar relative spatial positions within their own structures. On the other hand, a negative ρ_i suggests that the corresponding position is likely an "outlier" in the structural alignment.

Based on the distance correlation coefficient, we define the weight of i-th position as $w_i = \rho_i + \frac{1}{100}$, if $\rho_i \geq 0$; and $w_i = \frac{\rho_i + 1}{100}$, otherwise. In other words, we translate the correlation coefficients to non-negative weights and reduce the weights for the outliers with negative correlation coefficients. We then normalize the weight vector $\mathbf{w} = \{w_i / \sum_{j=1}^{n} w_j\}$ such that the sum of all weights equals to 1. Now, instead of minimizing the standard RMSD score, we are interested in minimizing the weighted RMSD, which is defined as $\text{wRMSD}(\mathbf{A}, \mathbf{B}) = \sqrt{\sum_{i=1}^{n} w_i \|\mathbf{a_i} - \mathbf{b_i}\|^2}$.

The Kabsch algorithm [28] noted a means to incorporate weighting into the RMSD fit, as described below. Here we assume the weighted centers of the structures $\mathbf{T}$ and $\mathbf{M}$ are already at the origin.

1. Calculate the weighted covariance matrix $\mathbf{C_w} = \mathbf{M}\,\text{diag}(\mathbf{w})\mathbf{T}^\top$;
2. Perform singular value decomposition (SVD) of the weighted covariance matrix $\mathbf{C_w} = \mathbf{V_w S_w W_w}^\top$;
3. Compute the weighted rotation matrix $\mathbf{U_w} = \mathbf{W_w V_w}^\top$.

The optimal weighted scaling factor can then be obtained as $s_w = \arg\min_s \text{wRMSD}(s\mathbf{U_w M}, \mathbf{T})$. Lastly, the weighted RMSD score is calculated as $\text{wRMSD}(s_w\mathbf{U_w M}, \mathbf{T})$.

Algorithm 2: Weighted RMSD fit

Input: input structure $\mathbf{M} \in R^{3\times n}$, template structure $\mathbf{T} \in R^{3\times n}$

Output: weighted-RMSD-score, optimal scaling factor s, optimal rotation matrix $\mathbf{U}$

1 For the pair of point i in $\mathbf{M}$ and $\mathbf{T}$, calculate the distance vectors d_{m_i} and d_{t_i} and evaluate the weights $\mathbf{w}$ by traversing all points;

2 Calculate the weighted covariance matrix $\mathbf{C_w} = \mathbf{M}\,\text{diag}(\mathbf{w})\mathbf{T}^\top$;

3 Perform singular value decomposition (SVD) of the weighted covariance matrix $\mathbf{C_w} = \mathbf{V_w S_w W_w}^\top$;

4 Compute the weighted rotation matrix $\mathbf{U_w} = \mathbf{W_w V_w}^\top$.

5 Search for the optimal scaling factor $s_w = \arg\min_s \text{wRMSD}(s\mathbf{U_w M}, \mathbf{T})$;

6 Compute wRMSD-score $= \text{wRMSD}(s_w\mathbf{U_w M}, \mathbf{T})$.

The weighted RMSD leverages the rotation matrix $\mathbf{U_w}$ and scaling factor s_w to make structure $\mathbf{M}$ fit structure $\mathbf{T}$. If the structure $\mathbf{T}$ is from the set of ground truth, we normalize the weighted-RMSD-score by the average radius of the bead. The average radius of the bead is calculated by the mean distance between the consecutive points in the ground truth structure. Otherwise, we re-scale the average distance between points and the origin to 1 for both structures before feeding into weight RMSD. For example, in Section 2.2.1 and 2.2.2, the structures from modeling methods are compared with the ground truth, so we normalized the weighted-RMSD-score by the average radii. The value in Table 2.1(a,c,d,f) represents the times of the average radii.

A.3.3 DTM-score

Here we propose Dynamic Template Modeling alignment (DTM-align), a chromatin structure alignment algorithm. The DTM-align is inspired by the Template Modeling alignment (TM-align) method [95], a widely used approach for measuring similarity between two protein structures. The DTM-align algorithm searches for the optimal alignment between two chromatin structures of the same length, and reports a quantitative similarity measure DTM-score.

The definition of DTM-score is the same as the original TM-score [95], the gold standard measure of protein structure similarity. Specifically, DTM-score$(\mathbf{A}, \mathbf{B}, d_0) = \frac{1}{n}\sum_{i=1}^{n} \frac{d_0^2}{d_0^2+\|\mathbf{a_i},\mathbf{b_i}\|^2}$, where $\mathbf{a_i}, \mathbf{b_i}$ are the i-th beads in structures $\mathbf{A}$ and $\mathbf{B}$, respectively, and d_0 is the distance threshold estimated from randomly selected chromatin structure pairs, as described below. Unlike the traditional RMSD measure, DTM-score is less sensitive to local structural variations and lies in the range of $(0, 1]$. Higher DTM-score indicates similar chromatin folding between two structures, with the special case of DTM-score equals to 1 when two structures are identical; whereas lower DTM-score suggests different structures.

The distance threshold d_0 is an important parameter in the calculation of DTM-score. Specifically, d_0 represents the average Euclidean distance of corresponding bead pairs between two randomly selected chromatin structures. Because there is no standard unit for the scale of chromatin structures. The d_0 threshold needs to be estimated separately for different set of template structures at each resolution. Here we use the eight published single-cell structures predicted by NucDynamic [71] as the templates and thoroughly estimated d_0 for eight different resolutions from 8 Mb to 100 kb. Briefly, at each resolution, we employ a random trimming strategy to extract thousands of chromatin substructures from the eight whole-genome template structures. Then we perform the weighted

RMSD fit between random pairs of substructure with the same length (measured by the number of beads) and calculate the average bead-wise distance between the aligned pair. Lastly, we fit an exponential polynomial curve to estimate the relationship between d_0 and the substructure length n. We perform this estimation procedure to obtain an exponential polynomial function of $d_0(n)$ for each resolution.

Algorithm 3: DTM score

Input: input $\mathbf{M} \in R^{3\times n}$, template $\mathbf{T} \in R^{3\times n}$, distance threshold d_0

Output: DTM-score

for *fragment length* $l = n, \frac{n}{2}, \frac{n}{4}, \frac{n}{8}, \cdots, 4$ **do**

  for *fragment starting position* $i = 1, \cdots, n - l + 1$ **do**

    Extract the l-length substructures starting from the i-th bead:

    $\mathbf{M_{i:i+l-1}} = (\mathbf{m_i}, \cdots, \mathbf{m_{i+l-1}})$ and $\mathbf{T_{i:i+l-1}} = (\mathbf{t_i}, \cdots, \mathbf{t_{i+l-1}})$.

    Calculate the distance correlation vector $\rho \in R^n$, where ρ_j is the Pearson correlation coefficient of the relative spatial positions for the j-th pair of beads in the substructures $\mathbf{M_{i:i+l-1}}$ and $\mathbf{T_{i:i+l-1}}$, for $i \le j < i + l$ (as described in Section A.3.2); Set $\rho_j = -1$, if $j < i$ or $j \ge i + l$.

    Initialize the weight vector $\mathbf{w} \in R^n$: $w_j = \rho_j + 1$.

    Normalize the weight vector $\mathbf{w}$ such that the sum of all weights equals to 1. Denote the normalized weight vector as $\widehat{\mathbf{w}}$, where $\widehat{w}_j = w_j / \sum_{k=1}^{n} w_k$;

    Initialize $\Delta_{\widehat{\mathbf{w}}} = \widehat{\mathbf{w}}$;

    while *changes in the weight vector* $\Delta_{\widehat{\mathbf{w}}} \neq \mathbf{0}$ **do**

      Centralize $\mathbf{M}$ and $\mathbf{T}$ such that their weighted centers are at the origin. Denote the translated structures as $\widetilde{\mathbf{M}} = \mathbf{M} - \mathbf{M}\widehat{\mathbf{w}}^\top \mathbf{1}^{1\times n}$ and $\widetilde{\mathbf{T}} = \mathbf{T} - \mathbf{T}\widehat{\mathbf{w}}^\top \mathbf{1}^{1\times n}$;

      Perform the weighted RMSD fit between $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{T}}$ using weights $\widehat{\mathbf{w}}$, and obtain the optimal rotation matrix $\mathbf{U_w}$ and the optimal scaling factor s_w;

Algorithm 3: Continued DTM score

8

9

10

11 Compute the distance vector $\mathbf{d} \in R^n$, where $d_j = \|s_w \mathbf{U_w} \widetilde{\mathbf{m_j}}, \widetilde{\mathbf{t_j}}\|$ is the Euclidean distance between the j-th pair of beads in the aligned structures $s_w \mathbf{U_w} \widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{T}}$;

12 Perform the weighted RMSD fit between $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{T}}$ using weights $\widehat{\mathbf{w}}$, and obtain the optimal rotation matrix $\mathbf{U_w}$ and the optimal scaling factor s_w ;

13 Compute the distance vector $\mathbf{d} \in R^n$, where $d_j = \|s_w \mathbf{U_w} \widetilde{\mathbf{m_j}}, \widetilde{\mathbf{t_j}}\|$ is the Euclidean distance between the j-th pair of beads in the aligned structures $s_w \mathbf{U_w} \widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{T}}$;

14 Update the weight vector $\mathbf{w}^* \in R^n$: $w_j^* = 1$ if $d_j^{(k)} < d_0$; $w_j^* = 0$, otherwise. In other words, we keep a bead where the Euclidean distance between the aligned pair is less than our distance threshold d_0, and reconstruct a new pair of substructures accordingly.

15 Normalize the weight vector $\mathbf{w}^*$ such that the sum of all weights equals to 1. Denote the normalized weight vector as $\widehat{\mathbf{w}}^*$, where $\widehat{w}_j^* = w_j^* / \sum_{k=1}^n w_k^*$;

16 Update $\Delta_{\widehat{\mathbf{w}}} = \widehat{\mathbf{w}}^* - \widehat{\mathbf{w}}$ and $\widehat{\mathbf{w}} = \widehat{\mathbf{w}}^*$;

17 Compute the DTM-score: DTM-score$_{i,l} = \frac{1}{n} \sum_{j=1}^n \frac{d_0^2}{d_0^2 + d_j^2}$

18 Find the highest DTM-score among all i: DTM-score$_l = \max_i$ DTM-score$_{i,l}$

Reference	Type	GEO Accession	Notes
Stevens et al. [71]	Single-cell Hi-C	GSE80280	Eight single-cell Hi-C contact frequency matrices generated from haploid mouse ESCs at 100 kb resolution
Stevens et al. [71]	Single-cell structures		Eight single-cell structures simulated by viaNucDynamic at 100 kb resolution
Stevens et al. [71]	Bulk Hi-C	GSE80280	Bulk Hi-C contact frequency matrices generated from haploid mouse ESCs at 100 kb resolution (GEO sample ID: GSM2123564)
Rao et al. [59]	Bulk Hi-C	GSE63525	Bulk Hi-C contact frequency matrices generated from diploid human lymphoblastoid GM12878 cell line at various resolutions (1 Mb, 500 kb, 250 kb, 100 kb, 50 kb, 25 kb, 10 kb, and 5 kb)
Meuleman et al. [58]	LADs	GSE17051	Constitutive Lamina-Associated Domains (LADs) modeled from diploid mouse ESCs
Rao et al. [59]	3D-FISH		3D Fluorescence in situ hybridization (3D-FISH) data in human lymphoblastoid GM12878 cells

Table A.1: **Summary of datasets used in comparison study**

The pseudocode of the DTM-align algorithm is outlined in Algorithm 3. Given the input structure $\mathbf{M}$, the template structure $\mathbf{T}$, and the distance threshold d_0 for length-n structures, the DTM-align algorithm searches for the best alignment (in terms of the scaling factor s, the rotation matrix $\mathbf{U}$, and the tranlation matrix) such that DTM-score of the aligned structures is maximized.

To search for the optimal alignment, the DTM-align algorithm traverses through all possible chromatin fragments of various length from 4 to n and extracts the corresponding substructures from $\mathbf{M}$ and $\mathbf{T}$. Using each pair of substructures as the seed, the initial alignment is obtained using the distance correlation weighted RMSD fit (as previously described in Section A.3.2). Note that, although the initial alignment parameters (the scaling factor s and the rotation matrix $\mathbf{U}$) is calculated from the substructures, the same alignment parameters can be applied to the full structures to obtain a global alignment between $\mathbf{M}$ and $\mathbf{T}$. This initial global alignment is further optimized through the following heuristic iterations. At each iteration step, the Euclidean distance of each pair of equivalent beads in the current alignment is calculated. If the distance at the i-th position is larger than or equals to the distance threshold d_0, we consider such position as an outlier and exclude it in the next round of alignment. In other words, we use positions with close matches between two structures to iteratively improve the global alignment and optimize the match of the overall structural topology. Once the alignment parameters converge, the DTM-score of the aligned structures can be computed accordingly. Because the global structure alignment is sensitive to local structural variations, the algorithm traverses through all structures as the possible starting seeds, and reports the maximal DTM-score among all iterations and the corresponding alignment parameters as the final optimal alignment.

A.4 Evaluation using LAD data

The lamina-associated domains (LADs) data in mouse ESCs were obtained from [58]. Specifically, we downloaded the constitutive LADs that were largely invariant across different cell types and used UCSC LiftOver to convert the LAD coordinates from mm9 to mm10. In total, the constitutive LADs occupied 31.91% of the genome, with average length of 223.46kb. Next we binned the LADs into the same 100-kb resolution as the Hi-C data. We labeled a bin as a LAD bin if the majority of the bin is occupied by a LAD domain; otherwise, we label it as a non-LAD bin. After binning, the occupancy of LADs is 32.38%. We then evaluated the LAD lolization pattern on the predicted 3D structures of four chromosomes (chr1, chr9, chr19 and chrX), as described below.

Since the LADs are colocalized at the nuclear periphery, we expect the the Euclidean distances among LADs to be different from their genomic distances as shown in the 1D nucleotide sequence. Thus, we used the Kendall rank correlation coefficient to measure the discordance between the 3D Euclidean distance and 1D genomic distance among LADs, via the following three steps.

Suppose there are total K LADs and $L_k = \{s_k, e_k\}$ are the start and end positions of the k-th LAD.

1. Compute the genomic distance matrix $\mathbf{G} \in R^{K\times K}$, where g_{ij} is the average genomic distance between a pair of LADs L_i and L_j. That is, $g_{ij} = \frac{1}{(e_i-s_i)(e_j-s_j)} \sum_{l_i=s_i}^{e_i} \sum_{l_j=s_j}^{e_j} |l_i - l_j|$. The diagonal elements in $\mathbf{G}$ are set to be 0;

2. Compute the Euclidean distance matrix $\mathbf{D} \in R^{K\times K}$ based on the 3D structure $\mathbf{X}$, where d_{ij} is the average spatial distance between LAD pair L_i and L_j.

   That is, $d_{ij} = \frac{1}{(e_i-s_i)(e_j-s_j)} \sum_{l_i=s_i}^{e_i} \sum_{l_j=s_j}^{e_j} \|\mathbf{x_{l_i}} - \mathbf{x_{l_j}}\|$. The diagonal elements in $\mathbf{D}$ are set to be 0;

3. Calculate the Kendall rank correlation coefficient between corresponding rows (or column, as both matrices are symmetric) in $\mathbf{G}$ and $\mathbf{D}$ and report the average values of k correlation coefficients.

Hypothetically, if a chromosomal structure has a stick-like(helix) shape, the correlation coefficient between the genomic distance matrix and Euclidean distance matrix would be close to 1. In contrast, if the LAD regions on the structure are clustered and colocalize at the nuclear periphery, we would expect a small correlation coefficient value.

To estimate the null distribution of the expected Kendall correlation coefficients, we calculated the correlation coefficient values using the eight single-cell template structures from the same cell type [71] as ground truth. This procedure is performed on four chromosomes together. Then we calculated the t-test (right-tail, $\alpha = 0.05$) between the ground truth and the observed Kendall correlation coefficients. The right-tail p-value less than 0.05 indicates the Kendall correlation coefficients of modeling structures are significant larger than ground truth. In other words, the 3D structure displays minimal chromatin folding of LAD regions, therefore yielding to a high Kendall correlation coefficient.

A.5 Validation by 3D-FISH data

We used the published 3D-FISH data to evaluate the 3D structures predictions in human GM12878 cells[59]. They confirmed that in all four cases, the 3D-distance between L1 and L2 (the two contacting peak loci of a chromatin loop) was consistently shorter than the 3D distance between L2 and L3 (one peak locus and one control locus).

We validated the four chromatin loops in our predicted 3D chromatin structures at both 50 kb and 25 kb resolutions. At each resolution, we mapped the 3D-FISH loci L1 and L2 to the beads that contain the majority of the corresponding 30 kb oligo fragments. We then chose L3 to be the bead that has equal genomic distance away from L2 but on the opposite side.

For each modeling structure, we calculated the Euclidean distance d_1 between L1 and L2 and Euclidean distance d_2 between L2 and L3. We also traversed the entire chromosome and estimated the average Euclidean distance $\bar{d}$ between any two beads with the same genomic distance as L1 and L2. We determined the chromosome structure is consistent with 3D-FISH results, if $d_1 < d_2$ and $d_1 < \bar{d}$.

Appendix B

Enhancement of Hi-C resolution

B.1 Network

B.1.1 Details of the convolutional blocks used in the EnHiC model

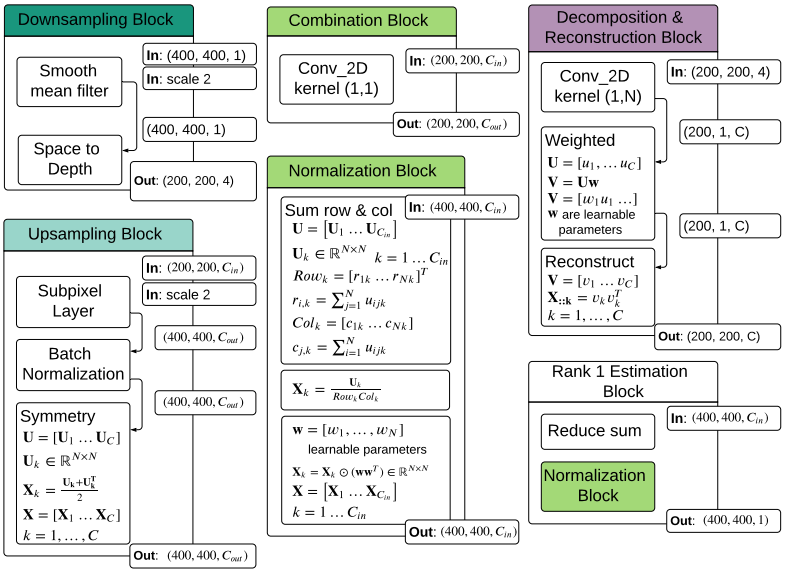


Figure B.1: **Layers of EnHiC model**. Details of the convolutional blocks used in EnHiC

Downsampling Block

The downsampling block shrinks the height/width of the matrix and rearranges the blocks of spatial data into depth (channel).

Algorithm 4: Downsampling Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$, ratio: r

Output: $\mathbf{X} \in \mathbb{R}^{\frac{N}{r} \times \frac{N}{r} \times c_{out}}$

1 Smooth the matrix by mean filtering: $\mathbf{X} = conv(\mathbf{U}), \mathbf{X} \in \mathbb{R}^{N \times N \times c_{in}}$, where the window size is $r \times r$

2 Rearrange blocks by *tf.nn.space_to_depth*: $\mathbf{X} = space\_to\_depth(\mathbf{X}, r)$

Combination Block

The combination block is a linear combination of the slices along the channels.

Algorithm 5: Combination Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$

Output: $\mathbf{X} \in \mathbb{R}^{N \times N \times c_{out}}$

1 2D convolution layer with kernel(1,1): $\mathbf{X} = conv2d(\mathbf{U})$

Decomposition & Reconstruction Block

This block aims to extract rank-1 matrices features.

Algorithm 6: Decomposition & Reconstruction Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$

Output: $\mathbf{X} \in \mathbb{R}^{N \times N \times c_{out}}$

1 Decompose the matrices using a 2D convolution layer with kernel(1,N):

$$\mathbf{U} = conv2d(\mathbf{U}), \mathbf{U} \in \mathbb{R}^{N \times 1 \times c_{out}}$$

2 Adjust weights of slices along the channels:

$$\mathbf{U} = [u_1, \dots u_{c_{out}}]$$

$$\mathbf{V} = \mathbf{U}\mathbf{w}, \mathbf{V} = [w_1 u_1 \dots]$$

where $\mathbf{w}$ are learnable parameters

3 Reconstruct the vectors into rank-1 matrix features:

$$\mathbf{V} = [v_1 \dots v_{c_{out}}]$$

$$\mathbf{X} = [\mathbf{X}_1 \dots \mathbf{X}_{c_{out}}], \mathbf{X_k} = v_k v_k^T, k = 1, \dots, c_{out}$$

Upsampling Block

This block employs a subpixel convolutional neural network layer to enhance the resolution from rank-1 matrix features.

Algorithm 7: Upsampling Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$, ratio: r

Output: $\mathbf{X} \in \mathbb{R}^{(Nr) \times (Nr) \times c_{out}}$

1 Subpixel layer: $\mathbf{U} = subpixel(\mathbf{U}), \mathbf{U} \in \mathbb{R}^{(Nr) \times (Nr) \times c_{out}}$

2 Batch Normalization: $\mathbf{U} = BN(\mathbf{U})$

3 Symmetry: For each channel, average the slice with its transpose

$$\mathbf{U} = [\mathbf{U}_1 \dots \mathbf{U}_{c_{out}}], \mathbf{U}_k \in \mathbb{R}^{(Nr) \times (Nr)}, k = 1, \dots, c_{out}$$

$$\mathbf{X} = [\mathbf{X}_1 \dots \mathbf{X}_{c_{out}}], \mathbf{X}_k = \frac{\mathbf{U_k} + \mathbf{U_k^T}}{2}, k = 1, \dots, c_{out}$$

Normalization Block

Similar to ICE (Imakaev et al., 2012), the normalization block learns the biases for rows and columns.

Algorithm 8: Normalization Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$

Output: $\mathbf{X} \in \mathbb{R}^{N \times N \times c_{in}}$

1 Marginalize the slice by row and column:

$$\mathbf{U} = [\mathbf{U}_1 \dots \mathbf{U}_{C_{in}}], \mathbf{U}_k \in \mathbb{R}^{N \times N}, k = 1, \dots, c_{in}$$

$$Row_k = [r_{1k} \dots r_{Nk}]^T, r_{i,k} = \sum_{j=1}^{N} u_{ijk}$$

$$Col_k = [c_{1k} \dots c_{Nk}], c_{j,k} = \sum_{i=1}^{N} u_{ijk}$$

2 Normalize slice:

$$\mathbf{X}_k = \frac{\mathbf{U}_k}{Row_k Col_k}$$

3 Calculate the biases:

$$\mathbf{w} = [w_1, \dots, w_N]$$

$$\mathbf{X}_k = \mathbf{X}_k \odot (\mathbf{w}\mathbf{w}^T) \in \mathbb{R}^{N \times N}$$

$$\mathbf{X} = [\mathbf{X}_1 \dots \mathbf{X}_{C_{in}}], k = 1 \dots c_{in}$$

Where $\odot$ is element-wise multiplication and $\mathbf{w}$ are learnable parameters.

Rank-1 Estimation Block

This block sums all channels together and normalizes the output.

Algorithm 9: Rank-1 Estimation Block

Input: $\mathbf{U} \in \mathbb{R}^{N \times N \times c_{in}}$

Output: $\mathbf{X} \in \mathbb{R}^{N \times N \times 1}$

1 Sum all channels together to reduce the number of channels:

$$\mathbf{X} = sum(\mathbf{U}), \mathbf{X} \in \mathbb{R}^{N \times N \times 1}$$

2 Pass the matrix to the Normalization Block

B.1.2 Generator

The architecture of the generator is illustrated below in Appendix Figure B.2:

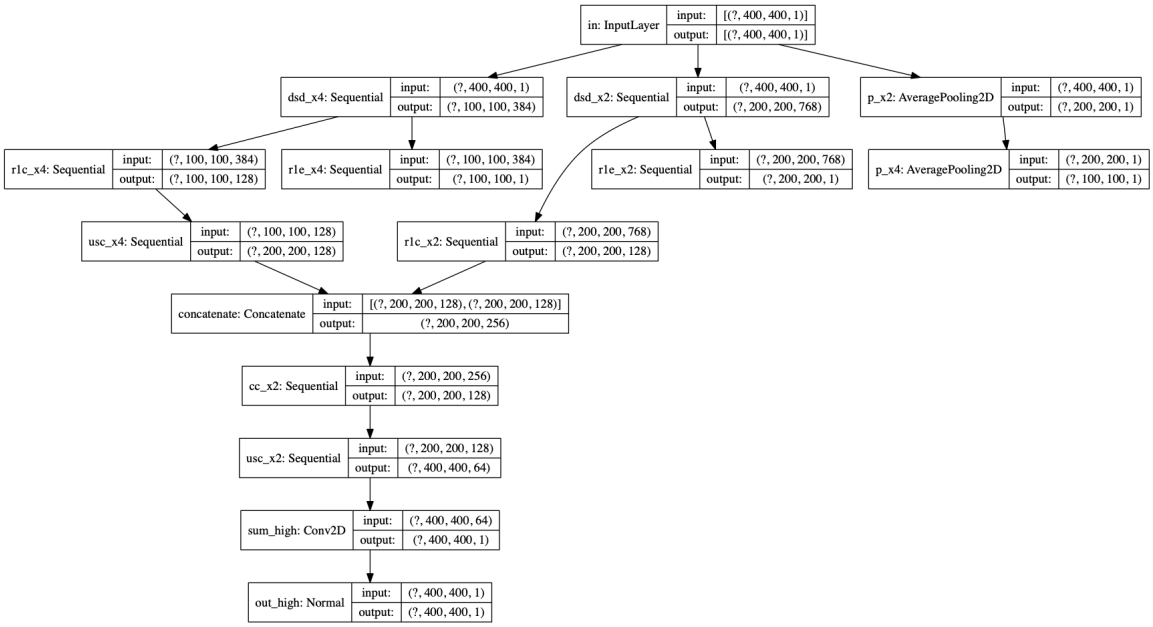


Figure B.2: **The architecture of the generator model**

B.1.3 Discriminator

The architecture of the discriminator is illustrated below in Appendix Figure B.3.:

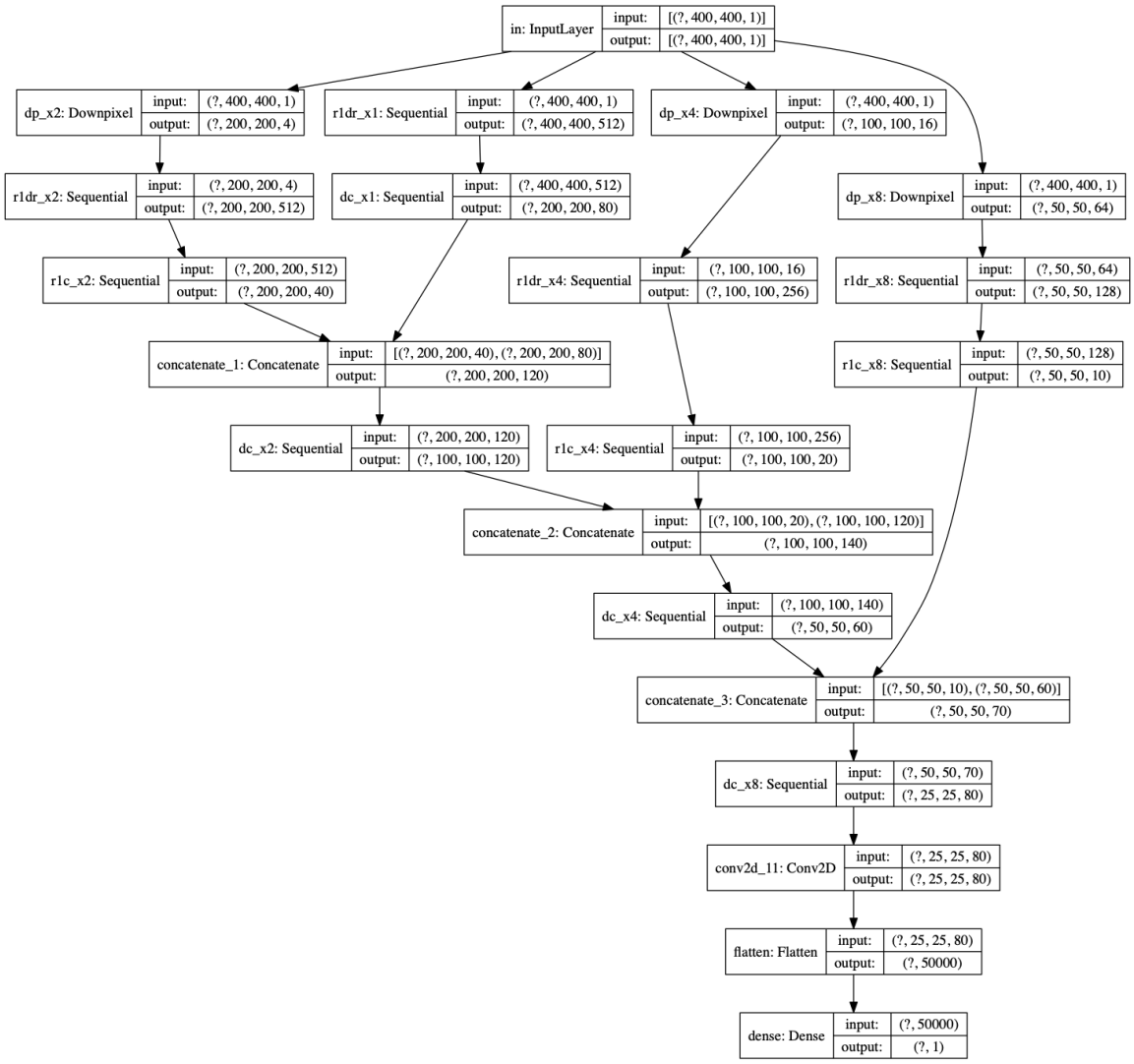


Figure B.3: **The architecture of the discriminator model**

B.1.4 Training and prediction

Appendix Table B.1 summarized the real Hi-C datasets used in the study. The filtered datasets were downloaded from the cooler database.

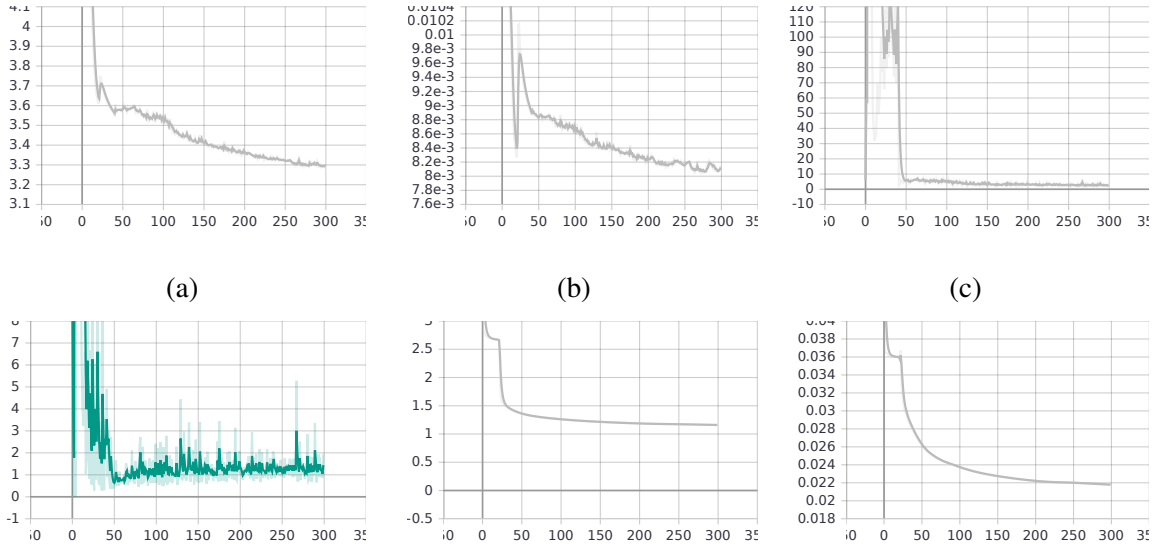
Cell line	RE	Assembly	Cis counts	Total counts
Rao2014 GM12878	MboI	hg19	2085711027	2884995088
Rao2014 IMR90	MboI	hg19	622282054	757393583
Rao2014 K562	MboI	hg19	473702480	617223144

Table B.1: **Summary of the Hi-C datasets used in Hi-C resolution enhancement**

Both training and prediction processes for the three models (EnHiC, Deephic, and HiCSR) were conducted using Intel Haswell CPU and NVIDIA Tesla K80 GPU wtih 128 GB of memory. The EnHiC model is implemented in Python with Tensorflow2.

We used chromosomes 1-16 for training, chromosomes 17 and 18 for hyper-parameters tuning, and chromosomes 19-22, and X for evaluation. The number of epochs for training is 300 and the parameters $\alpha_0 = 10, \alpha_1 = 0.1$. The runtime is approximately 85 hours (17 mins/epoch).

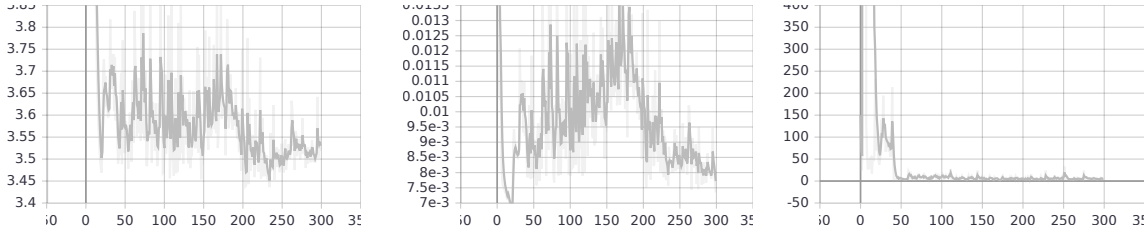
Below we plotted the MSE, DISSIM, and adversarial losses for the generator and discriminator in the training process in training dataset and validation dataset.



(a) (b) (c)

(d) (e) (f)

Figure B.4: **Training log**. **(a)** The MSE values of predictions at 10kb resolution in the training dataset, **(b)** The DISSIM values of predictions at 10kb resolution in the training dataset, **(c)** The adversarial loss values of generator in the training dataset, **(d)** The adversarial loss values of discriminator in the training dataset, **(e)** The weighted sum of MSE values of predictions at 20kb and 40kb resolutions in the training dataset. $\text{MSE} = \frac{\text{MSE}_{40kb}*4.0+\text{MSE}_{20kb}*16.0}{20.0}$, **(f)** The weighted of sum DISSIM values of predictions at 20kb and 40kb resolutions in the training dataset. $\text{DISSIM} = \frac{\text{DISSIM}_{40kb}*4.0+\text{DISSIM}_{20kb}*16.0}{20.0}$



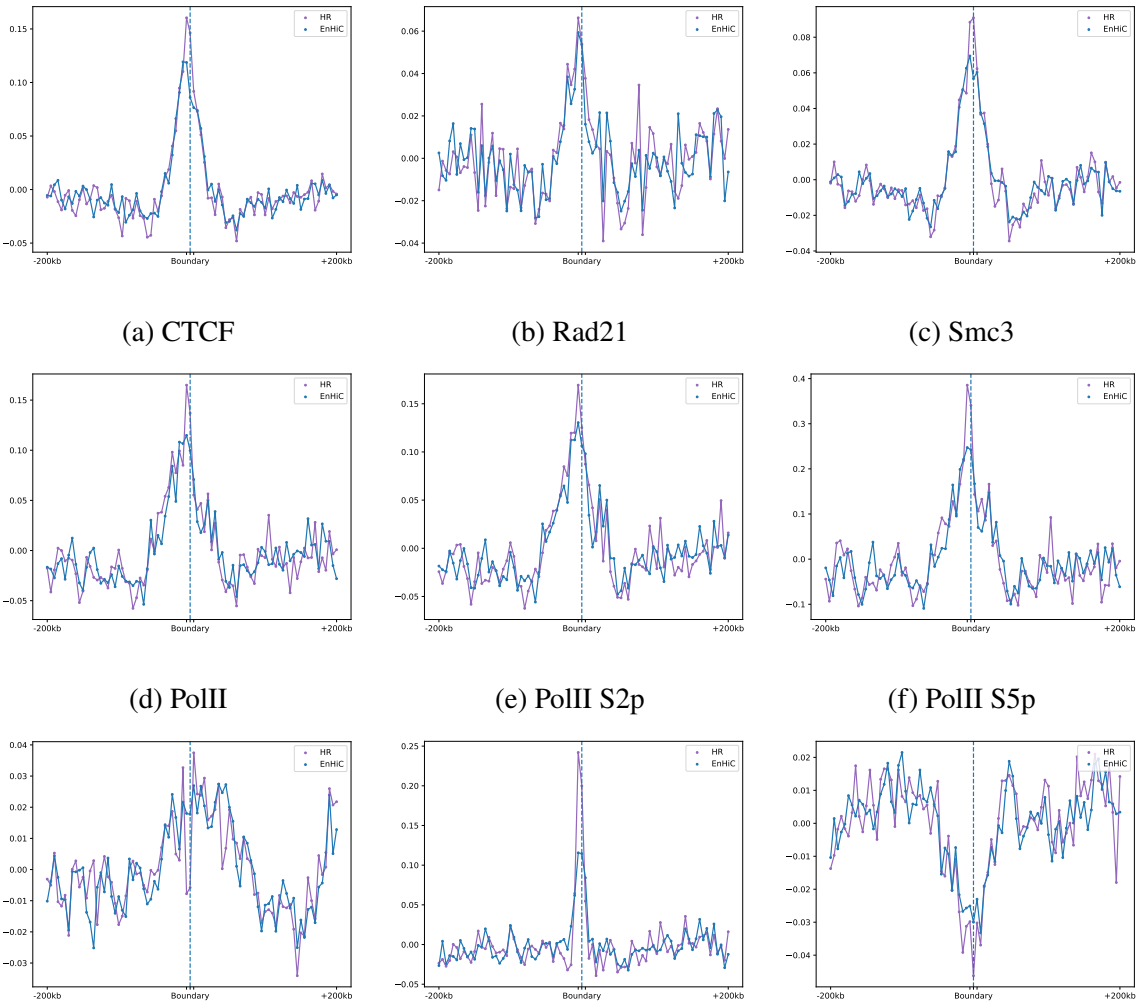
(a) (b) (c)

Figure B.5: **Training log**. **(a)** The MSE values of predictions at 10kb resolution in the validation dataset, **(b)** The DISSIM values of predictions at 10kb resolution in the validation dataset, **(c)** The adversarial loss values of predictions at 10kb resolution in the validation dataset.

	MAE			MSE		
Chr	Deephic	HiCSR	EnHiC	Deephic	HiCSR	EnHiC
19	3.46	4.46	0.29	665.6	2580.4	4.1
20	3.35	4.24	0.24	650.7	2453.1	2.5
21	5.71	8.57	0.52	966.6	5265.2	5.9
22	6.46	4.84	0.43	1417.1	1717.3	5.2
X	1.05	1.10	0.12	143.6	346.0	1.3

Table B.2: **The MAE and MSE errors**. Evaluation of the high-resolution Hi-C matrices predicted by Deephic, HiCSR, and EnHiC. Each prediction result is compared against the ground truth; the MAE and MSE errors are reported.

B.2 ChIP-seq enrichment/depletion



(a) CTCF (b) Rad21 (c) Smc3

(d) PolII (e) PolII S2p (f) PolII S5p

(g) H3K27me3 (h) H3K4me3 (i) H3K9me3

Figure B.6: **ChIP-seq enrichment/depletion at TAD boundaries**. ChIP-seq data were obtained from the ENCODE website, as documented in Appendix Table B.3.

Target	ChIP-seq file
CTCF	ENCFF271YKQ
Smc3	ENCFF235BXX
Rad21	ENCFF000WCT
H3K4me3	ENCFF818GNV
H3K9me3	ENCFF776OVW
H3K27me3	ENCFF594HSG
POlII	ENCFF368HBX
PolII S2p	ENCFF031RUV
PolII S5p	ENCFF002UPS

Table B.3: **ChIP-seq datasets**. ChIP-seq datasets obtained from the ENCODE website.

B.3 Significant chromatin interactions

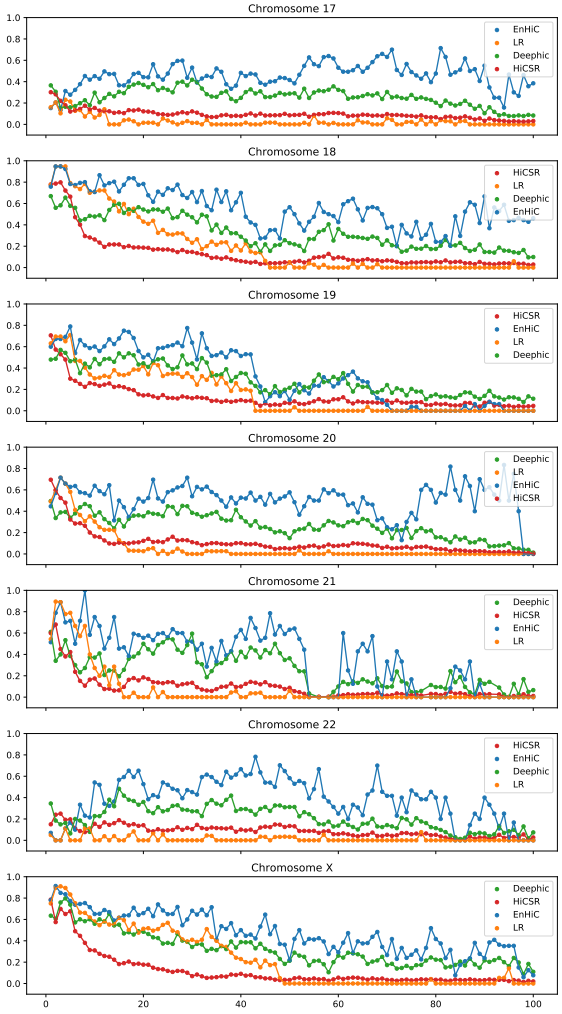


Figure B.7: **The Jaccard Scores of significant interactions**. The Jaccard Scores of significant interactions between high-resolution Hi-C and predictions/low-resolution input in seven chromosomes (17-22 and X). The LR represents the low-resolution Hi-C (40kb) downsampled from high-resolution Hi-C data. The x-axis (from 0 to 100) represents the genomic distance from 0 to 1000 kb.

B.4 TADs detection

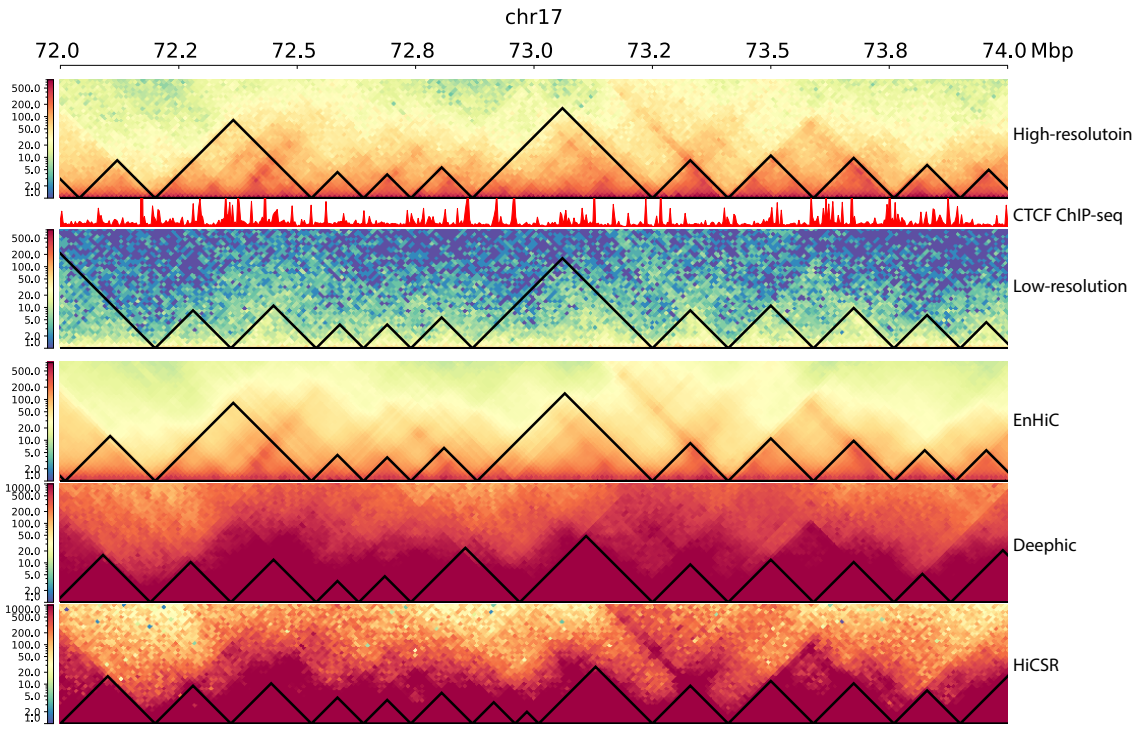


Figure B.8: **Examples of TAD detection results**. Chromosome 17 from 72Mbp to 74Mbp. TADs were identified using HiCExplorer. From top to bottom: true high-resolution (10kb) Hi-C data, CTCF ChIP-seq signal, low-resolution (40kb) input Hi-C data, and high-resolution predictions from EnHiC, Deephic, and HiCSR. For each Hi-C matrix, the heatmap of close-to-diagonal region is displayed with the color key from low (blue) to high (red) interaction frequencies. TADs are identified using HiCExplorer, and marked as black triangles.

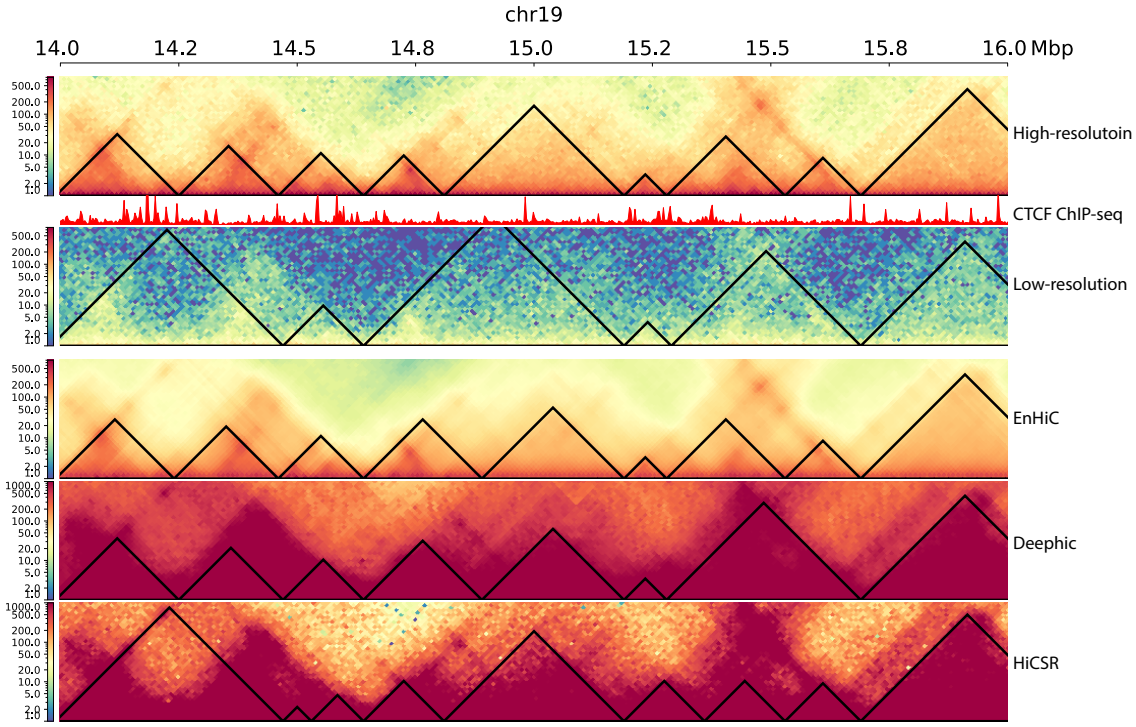


Figure B.9: **Examples of TAD detection results**. Chromosome 19 from 14Mbp to 16Mbp.