Predicting Gene Regulation and Expression with Graph Neural Networks

Evan Bell[1]
[1]*Dept. of Mathematics, Michigan State University, East Lansing, MI*

I. Problem and Goals

A. Initial Idea

Predicting gene expression from genomic data is an important task in computational biology. At the beginning of this study, we set out to predict gene regulation and expression by combining three different kinds of data: the raw (1D) DNA sequence from an organism, additional epigenomic data (also 1D), and a 3D graph of chromatin interactions.

Our goal was to develop a framework for simultaneously utilizing these three types of data to predict the expression level of various genes. Data for the true level of gene expression is also available from CAGE-seq studies, for example [1]. With large amounts of paired data in all of these modalities, it is reasonable to attempt to use a supervised approach to train a machine learning model for this prediction task. This was our aim at the start of the study.

B. Conceptual and Technical Motivations

One of the major challenges in predicting gene expression is the fact that chromatin can fold and coil up, creating a complex 3D structure. This means that two parts of the genome that are far apart in terms of base pair distance may actually be quite close together spatially. This makes it possible for regulatory elements to be quite far from the gene that they regulate, in terms of distance along the chromatin strand. Given this challenge, it is reasonable to suspect that incorporating information about the 3D structure of chromatin may make predicting gene expression easier.

Chromatin interaction data naturally has a graph structure: segments of the DNA strand can be represented by nodes, and two nodes are connected by an edge if they are in close contact with each other. There are a few potential ways that this graph data could be incorporated into a gene expression prediction pipeline. In this study, we employ graph neural networks for this task, leveraging their powerful expressive capabilities.

C. Evolution of Project Goals

At the beginning of the semester, we set out to reproduce the results of the existing study [2], which introduces two new approaches for predicting gene expression, called Epi-GraphReg and Seq-GraphReg. The Epi-GraphReg model only uses the 1D epigenomic data as input, and only attempts to predict the gene expression data, while incorporating the chromatin interaction graph. On the other hand Seq-GraphReg uses the DNA sequence itself as input, and attempts to predict the epigenomic data and gene expression data simultaneously.

As the semester went on, we realized that the goal of implementing and training both of these models was neither realistic nor essential. The focus of this project is to understand how graph data can be utilized to improve the prediction of gene expression, and to accomplish this we only need to implement one of these models. For simplicity,

we decided to only implement the Epi-GraphReg model.

II. Datasets

A. Initial Plan

Initially, we had planned to use real data in multiple formats. All the relevant data from DNase-seq (epigenome), CAGE-seq (gene expression), and Hi-C or HiChIP (3D interaction) studies is freely available in the ENCODE database [3], and can be accessed with the accession numbers listed in [2]. We were planning to use one organism/cell line's data only, such as the human K562 cell line.

B. Data Acquisition Challenges

Downloading the raw data from the ENCODE portal was relatively easy, but we quickly faced multiple challenges. First, the raw data was quite large, with some files being multiple gigabytes. Secondly, all three types of data require significant preprocessing to be used. The raw `BAM` files from the DNase-seq and CAGE-seq studies needed to be processed into `bigwig` files, which could then be loaded into TensorFlow data objects using custom scripts. For the 3D chromatin interaction data, the raw `fastq` data needs to be processed to find significant interactions, with a certain false discovery rate, to produce an `hic` file. The actual interaction graph can then be extracted from this file.

Clearly, these many preprocessing steps provided a significant challenge. Considering that the focus of this study is more on model architectures and understanding how 3D interaction data can be used to help predict gene expression, we decided to change our initial data plan.

C. Change of Data Plan

Due to the burdensome preprocessing required to use real data, as well as its prohibitively large size, we decided to begin our study with synthetic data. This had several advantages. Using synthetic data allowed us to use a smaller dataset, saving computational resources. It also allowed us to know the exact underlying model for the data, including the exact locations of the genes and regulatory elements. We also avoided practical challenges of attempting to align different types of data to one another. Moreover, we knew that any failure to predict the gene expression would not be due a lack of true relationships between the different types of data or errors in the preprocessing of the data.

III. Computational Approach

A. Originally Proposed Approach

A flowchart of the originally proposed approach is shown in Figure 1. We had initally planned to implement both the Epi-GraphReg model and the Seq-GraphReg model. Both of these models involve incorporating 3D interaction data and 1D epigenomic data into the pipeline. The difference is that Epi-GraphReg uses the epigenomic data to predict gene expression directly, whereas Seq-GraphReg uses the DNA sequence as input, and attempts to predict both the epigenetic data and the gene expression data simultaneously.

B. Original Analysis Plan

Originally, we planned to implement both of these methods, and compare them against each other, as well as some baselines.

The primary advantage of Epi-GraphReg and Seq-GraphReg is that they incorporate 3D interaction data into the prediction of gene expression. As an ablation study, we planned to also evaluate the performance of a model with the GAT layers removed, instead using only convolutions for feature extraction. We expected that this method may still give reasonable performance, since large components of genetic regulation are indeed local. We dubbed this simple baseline approach Epi-CNN. Additionally, we believed that removing the GAT layers may not be detrimental to performance when

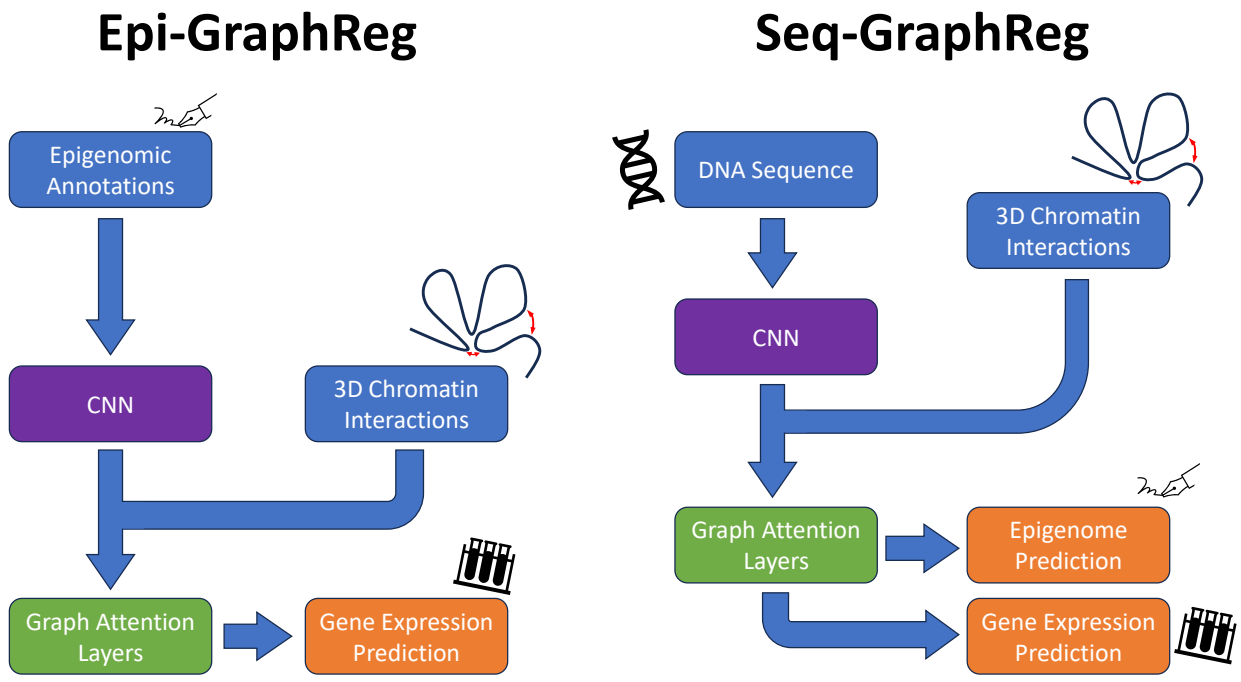

Fig. 1: Flowchart for the originally proposed Epi-GraphReg and Seq-GraphReg pipelines. The raw DNA sequence or epigenomic data is first processed using a CNN, as in previous methods. The 3D chromatin interaction data is then incorporated using graph attention layers before being processed to give a final output of the level of gene expression. In the case of Seq-GraphReg, only the DNA sequence is used as input, and both the epigenetic data and gene expression level are predicted simultaneously.

epigenetic data is given, but would be much more likely to severely harm performance when only the DNA sequence data is used, since the model needs to be able to learn interactions across a large number of base pairs to have a chance of successfully predicting the gene expression.

We also planned to compare against a linear regression based model, which had been used as a baseline in previous work [4]. While this method is extremely computationally inexpensive, it is necessarily significantly less expressive than deep neural networks. Because the model is so simple, we suspected that it may also be more robust to noise or other corruptions in the data. However, we also believed that this model would simply prove too restrictive to model the complex interactions present in the data.

We also proposed various potential metrics to measure the performance of these different models. Overall, the goal of our study is to predict the level of expression of different genes, using CAGE-seq data as the ground truth, so we needed a way to measure the difference between a model's predictions and this ground truth label. As the simplest possible metric, we proposed to use the mean squared error (MSE) as an evaluation criterion. However, MSE is only one important metric when evaluating performance. In particular, using MSE as a metric may be inappropriate in the case of underlying data distributions that are essentially multi-modal. Hence we also proposed to use the log-likelihood directly for evaluation. Using the log-likelihood may

also be more statistically justified in the case that underlying data are not normally distributed (they may be Poisson distributed, for example).

C. Steps from the Midterm to Now

Step 1. The first step that we took was generating synthetic data to use for training and evaluating our models. We generated data suitable for use with the Epi-CNN and Epi-GraphReg models. We needed to generate three types of data: synthetic epigenomic (DNase-seq) data, synthetic gene expression (CAGE-seq) data, and a graph of 3D chromatin interactions.

To do this, we began by generating DNase-seq data for 30 chromosomes. Each chromosome is composed of 1,000 bins. For each chromosome, we randomly chose 20 locations to act as genes and 20 locations to act as enhancers. In a 5 bin region around each gene or enhancer location, DNAse-seq values were drawn from a χ^2 distribution and multiplied by a draw from a uniform random variable. A 5 bin long 1D averaging filter was then applied to slightly smooth the data. Finally, uniformly distributed noise was added to random locations.

The 3D chromatin interaction graph was generated by iterating over each gene location, and connecting it to three random enhancer locations. Finally, using this graph, we generated CAGE-seq values by taking a linear combination of the DNase-seq values of the enhancers and the gene itself.

Examples of the generated DNase-seq and CAGE-seq data are shown in Figure 2. Figure 3 shows the DNase-seq for the chromosome, along with arcs showing the chromatin interaction graph.

Step 2. Step two was implementing the baseline Epi-CNN model. The authors of [2] implemented their models using TensorFlow, however we decided to opt for a more modern and more user friendly implementation using PyTorch Lightning. The entire Epi-CNN model, including training, validation, and testing, is implemented in just 70 lines of PyTorch code. This model consists of five blocks, with each block composed of a normalization layer, a 1D convolution, and a ReLU activation function. For the convolutions, we use a kernel size of 3 and a channel dimension of 64, except for the input and output. For the normalization, we use InstanceNorm [5], since it is very simple and generally effective. Other normalization schemes, such as BatchNorm, GroupNorm, or LayerNorm could likely be substituted with little effect on the model's performance. The authors of [2] chose to use BatchNorm [6].

Step 3. Our third step was implementing the Epi-GraphReg model. This model is the same as the Epi-CNN model, except it also includes graph attention layers [7] to incorporate the 3D interaction graph. Both model architectures are shown completely as block flow diagrams in Figure 4. We added three graph attention layers, again with normalization and ReLU. These layers were easily implemented using PyTorch Geometric. One key feature of our implementation compared to the implementation of [2] is the use of PyTorch Geometric graph objects. In [2], the chromatin interaction graph is stored as an adjacency matrix. For a small interaction graph, this may not be a computational problem, but it does create a quadratic memory cost in the number of nodes in the graph, which may impede scaling to larger datasets or higher resolution assays. The adjacency lists in our implementation require memory linear in the number of nodes and edges, improving computational efficiency.

Step 4. The next step was training both models. In order to do this, we used the PyTorch Lightning command line interface. This uses the training step methods built into each model, as well as a data module for loading the data, which also had to be written. Both models were trained using the Adam optimizer [8] with a learning rate of 10^{-3} for $1,000$ epochs (passes over the training set). The training and validation losses empirically converged for both models. Training was performed using a single

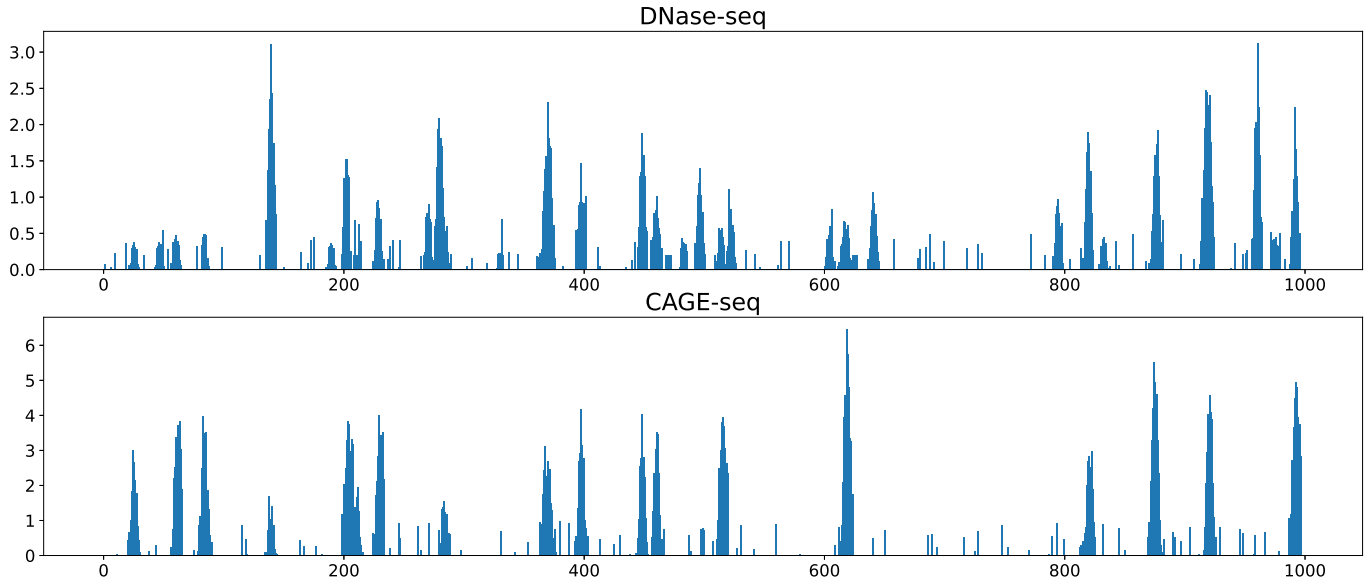

Fig. 2: Example of the synthetic DNase-seq and CAGE-seq data generated for one chromosome, with enhancers as the only regulatory elements. We can see that the DNase-seq has more peaks, since it shows regions of high chromatin availability for both genes and enhancers, while the CAGE-seq data shows levels of gene expression only.

NVIDIA RTX A5000 GPU and took approximately 1 minute for the Epi-CNN model and 5 minutes for the Epi-GraphReg model.

Step 5. Next, we had to evaluate both trained models and compare the results. This process included both quantitative and qualitative evaluations. To quantitatively compare the models, we computed the MSE of each model's predictions on the testing set. This was again done directly using the PyTorch Lightning command line interface. We also computed the MSE for each model only over regions of interest, i.e. regions that have genes. This is because it is easy to identify regions where gene expression is zero, so this metric may better reflect the performance of each model. We also produced plots of the predictions of each model, to better see where each model succeeded and struggled.

Step 6. After successfully completing steps 1–5, we wanted to further study the ability of the Epi-GraphReg model to leverage the topology of the interaction graph. To do this, we generated another synthetic dataset, this time including both enhancers and silencers as regulatory elements. The data is very similar to the dataset generated in step 1, except instead of having 20 enhancers, we chose 10 enhancers and 10 silencers. In this case, each gene is regulated by 3 enhancers, which increase its expression level, and 2 silencers, which decrease its expression level. This study is interesting because the model will only be able to tell the difference between these types of regulatory elements using the graph topology.

Step 7. We then trained both Epi-CNN and Epi-GraphReg models on this new dataset, using the same procedure described in step 4, and tested them using the same metrics described in step 5.

IV. Key Findings

A. 3D Interactions are Important

A primary finding of our study is that incorporating 3D chromatin interaction into the prediction of gene expression can significantly improve model

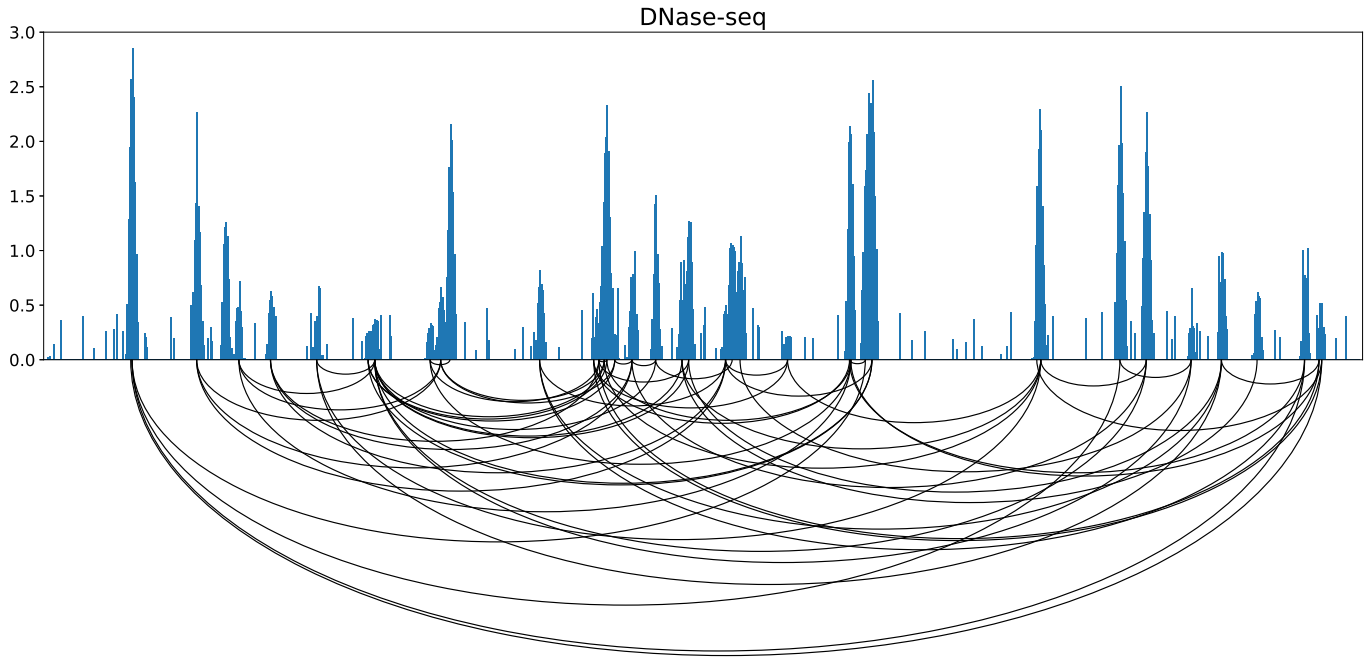

Fig. 3: Example of DNase-seq and the associated chromatin interaction graph for one of the test chromosomes. Arcs below the sequence represent locations of the chromatin strand that are in close contact. For this chromosome all regulatory elements are enhancers.

Model	Test MSE	Test MSE over ROI
Epi-CNN	1.105	0.373
Epi-GraphReg	**0.199**	**0.045**

TABLE I: Quantitative comparison of model performance on the testing set, for the dataset with only enhancers. Metrics are reported over both the entire chromosomes and just the regions of interest (ROI). Best performance is indicated in **bold**.

performance. Table I gives a quantitative comparison of the Epi-CNN model and the Epi-GraphReg model on the dataset with only enhancers. Indeed the Epi-GraphReg model achieves an MSE on the test set that is approximately 5 times lower than the Epi-CNN model. This gap widens even further when only the region of interest is considered, with the Epi-GraphReg model outperforming the Epi-CNN model by almost an order of magnitude in terms of MSE.

A qualitative comparison of model performance–again on the data with enhancers only–is shown in Figure 5. The Epi-GraphReg model is very accurate, with all peaks in the predicted CAGE-seq aligning closely with the ground truth. On the other hand, the Epi-CNN model is much more prone to either hallucinating peaks where genes are not actually expressed, or completely missing peaks where genes are expressed. However, as expected, the Epi-CNN model is generally capable of predicting where the level of gene expression is zero. This validates the claim that computing the MSE over regions of interest better reflects each model's true performance.

B. Epi-GraphReg Can Exploit Graph Topology

The second key finding of our study is that Epi-GraphReg is able to learn about different kinds of regulatory interactions only from graph topology. The second dataset we generated contains two types of regulatory elements: enhancers, which increase the level of expression of the genes that they are

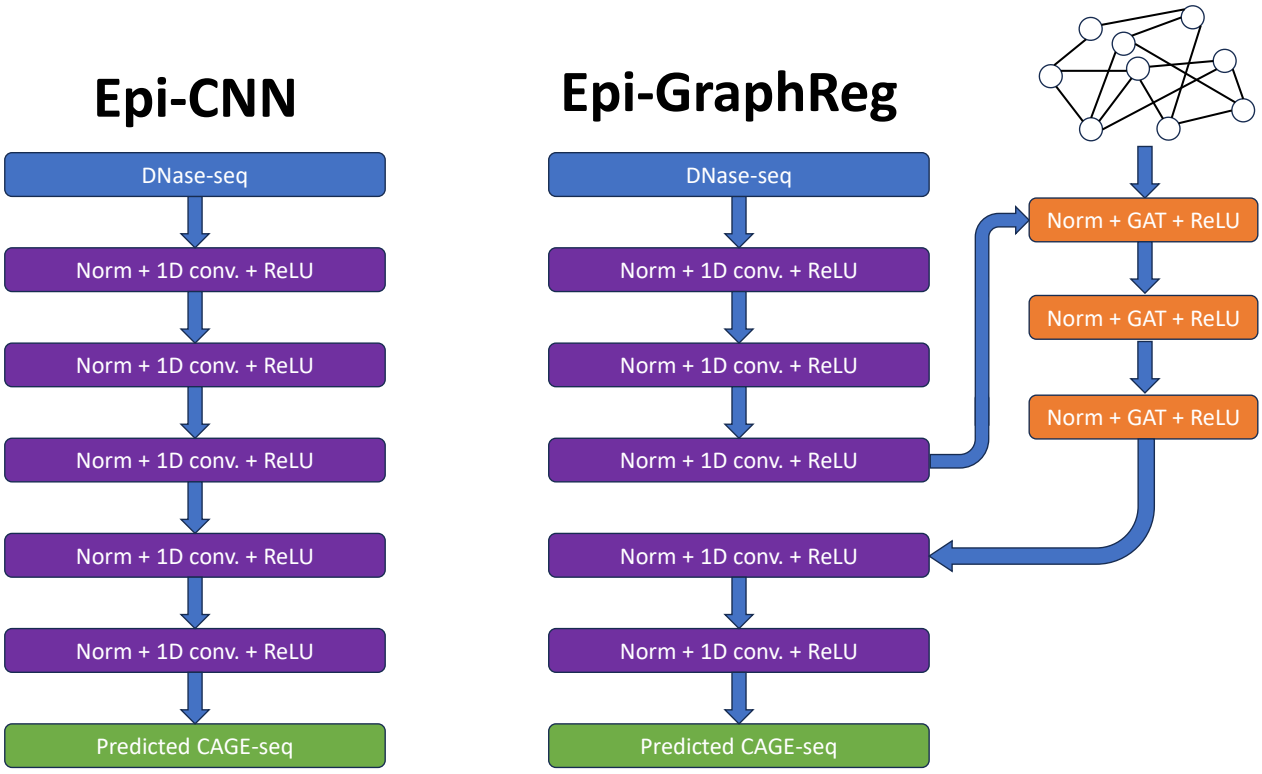

Fig. 4: Model architectures for the baseline Epi-CNN model and the Epi-GraphReg model. The only difference is that the Epi-GraphReg model has an additional module that incorporates the 3D chromatin interaction graph using graph attention layers. Otherwise, the network architectures are identical.

connected to, and silencers, which reduce it. In the generated DNase-seq data, these elements are completely indistinguishable. However, there is a difference in the interaction graph: each gene is connected to 3 enhancers, but to only 2 silencers. Hence the model must leverage the graph topology in order to make accurate predictions.

A quantitative comparison of the performance of the Epi-CNN and Epi-GraphReg models on this dataset is provided in Table II. A qualitative comparison on one test chromosome is shown in Figure 6. As expected, the Epi-GraphReg model again outperforms the Epi-CNN model by a wide margin. Moreover, the fact that Epi-GraphReg has successfully learned to predict the gene expression data shows that it is able to infer the regulatory function of genetic elements only from the graph topology. This is an interesting finding that helps to demonstrate the model's capability to learn complex (and hopefully realistic) relationships from the graph data.

Model	Test MSE	Test MSE over ROI
Epi-CNN	0.524	0.189
Epi-GraphReg	**0.140**	**0.062**

TABLE II: Quantitative comparison of model performance on the testing set, for the dataset with both enhancers and silencers. Metrics are reported over both the entire chromosomes and just the regions of interest (ROI). Best performance is indicated in **bold**.

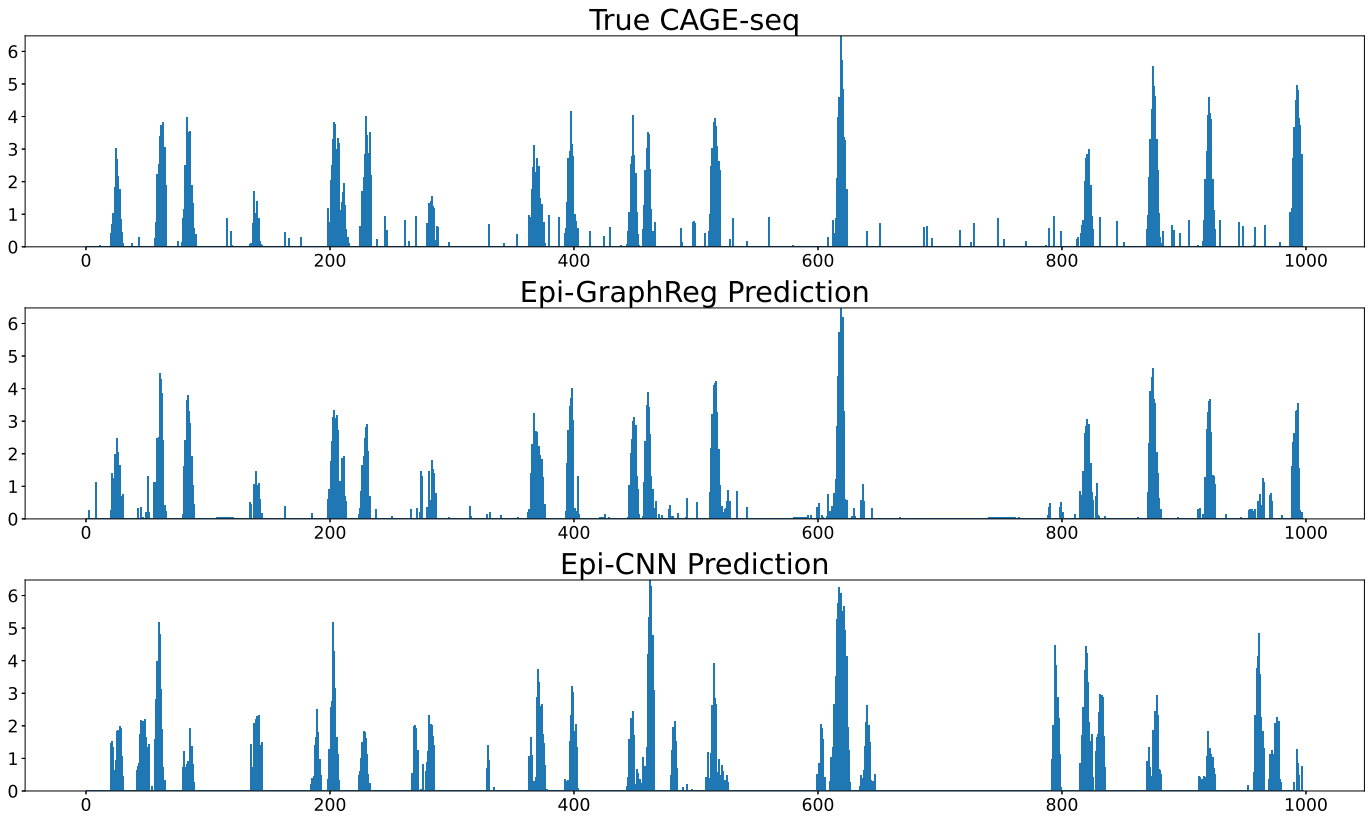

Fig. 5: Comparison of model predictions on one of the test chromosomes from the dataset with only enhancers. We see that the Epi-CNN model either misses gene expression peaks or hallucinates them, while the Epi-GraphReg model is generally accurate.

C. Code and Data Availability

All code and data for this project are available at https://github.com/evanbell02/EpiGraphReg. Pre-trained Epi-CNN and Epi-GraphReg models are also available for both datasets. Documenting this repository and making it presentable was not too challenging. Every result from this work should be easily reproducible. All Jupyter notebooks used to produce figures are also included, as is a file for building the needed conda environment. All file paths are relative, so the code should be able to be run immediately. Additionally, all training and testing functionality is available directly from the command line, so it should be straightforward to use this repository on any system.

V. Challenges

A. Key Technical Challenges

The largest technical challenge that we faced during this project was attempting to download and process real epigenomic and gene expression data. Ultimately, this challenge proved large enough that we decided to use synthetic data instead, so that we would be able to complete the more pertinent aspects of the study.

Another key technical challenge was designing the synthetic datasets. Creating data that resemble real DNase-seq and CAGE-seq data was somewhat difficult, since an underlying model for these data is not known. Additionally, we had to ensure that the task would be appropriately challenging for both the Epi-CNN and Epi-GraphReg models. We wanted

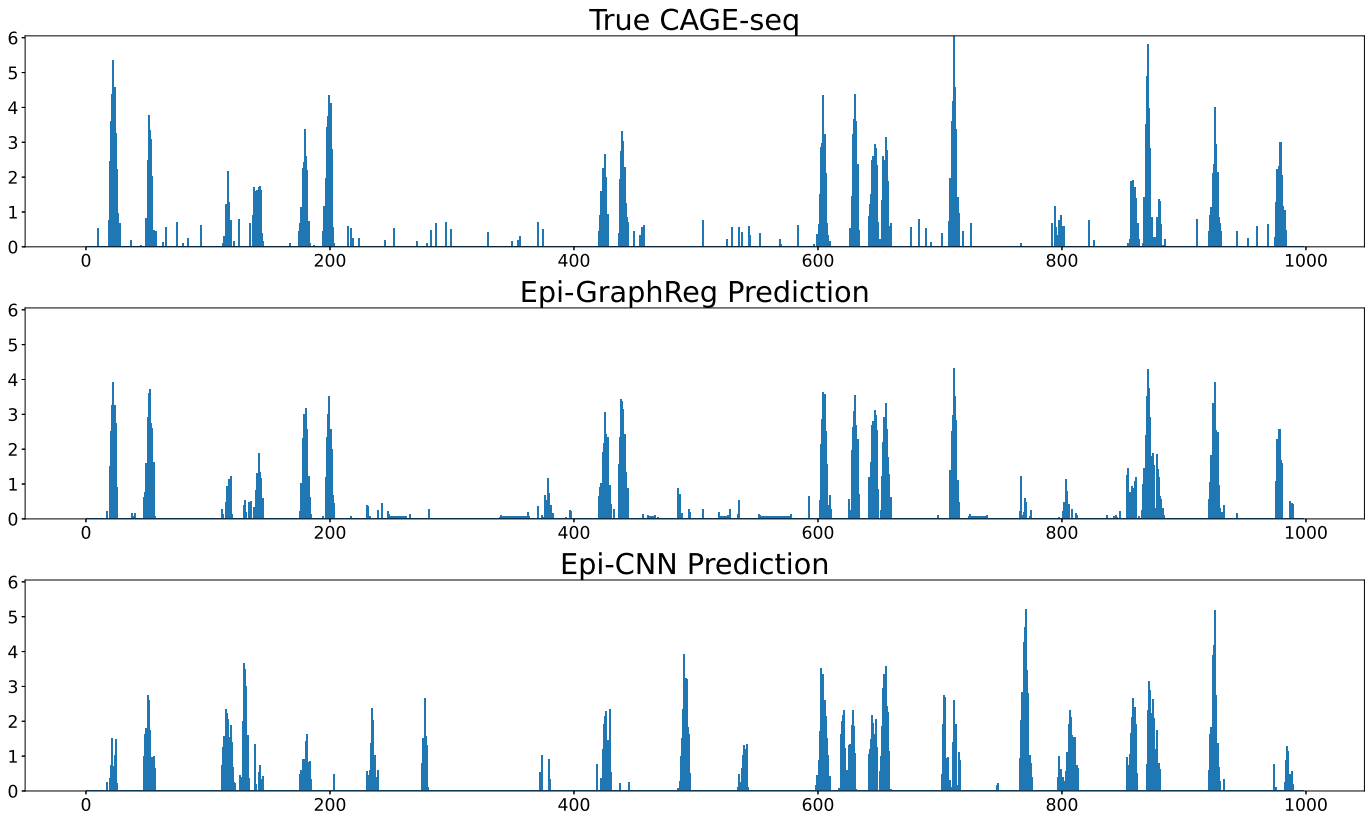

Fig. 6: Comparison of model predictions on one of the test chromosomes from the dataset with both enhancers and silencers. We again see that the Epi-GraphReg model is able to predict the locations and heights of gene expression peaks much more effectively than the Epi-CNN model. Its accurate predictions also serve as strong evidence that the Epi-GraphReg model can learn to distinguish different regulatory mechanisms purely from graph topology.

the relationship to be simple enough that the Epi-CNN would have some predictive capabilities, while not being so simple that the prediction task would be trivial for Epi-GraphReg. We ultimately believe that we achieved this goal.

B. Key Practical Challenges

One practical challenge in conducting this study was the need for computational resources to train the machine learning models. This is an additional reason that we decided to use a small synthetic dataset, as opposed to using real data. Training large models on large data may have yielded more realistic results, however it also would have used computing resources–especially GPU time–needed to meet other deadlines. Our synthetic "chromosomes" are smaller than the data used in [2] by a factor of 60. This also enabled us to use much smaller models, while still achieving good performance. Our implementations of the Epi-CNN and Epi-GraphReg models both have fewer than $100,000$ parameters.

Another practical challenge was ensuring that all results of the study would be easily reproducible. This involved a few extra steps that would not have been necessary otherwise. One such challenge was ensuring that all file paths are relative, rather than hard-coded. Another was keeping track of the individual model checkpoints and ensuring that they were properly organized and formatted. Our goal was to make sure that they can be used by anyone

with only a few lines of code. Additionally, it was important to maintain a clear and standard file structure, which would make the repository more readily understandable and accessible to end users.

VI. Reflection and Future Directions

If we were to carry out this study again, there are a few aspects that we might approach differently. In this section, we reflect on these areas and offer a few ideas for future extensions of this work.

A. Problem Statement

Overall, while the problem that we set out to address was interesting and biologically relevant, we did not identify its most salient aspects at the beginning of the project. Particularly, we focused more on studying the underlying biological problem than studying the computational methods. Instead of asking "how can we predict gene expression?" we should have asked "under what conditions can methods like Epi-GraphReg successfully predict gene expression?" This would have allowed us to narrow in on the key messages of our study sooner.

B. Data

While we are satisfied with the data that we eventually ended up using in this project, it is still true that the main shortcoming of this study is that only synthetic data was used. Using real data posed many significant challenges, which we would attempt to overcome if we were to carry out this study again. Realistically, this would likely involve finding collaborators with more expertise in working with these types of data.

C. Approach

Generally, we are satisfied with the computational approaches implemented. The Epi-CNN is a very reasonable and effective baseline model, and we successfully implemented the powerful Epi-GraphReg model. In terms of changes to our approaches, there are two main areas for improvement. First, since this study is mostly focused on comparing different architectures, it would benefit from more baselines. In particular, we never implemented the linear regression model, although it may have been another valuable baseline. We also could have tested graph neural network architectures other than GATs, such graph convolutional networks [9]. Secondly, implementing other metrics, such as the proposed log-likelihood metric may have offered more insights into each model's performance.

D. Code

At the end of this project, we produced a repository that can be used to easily reproduce all of our results. Ultimately, this was our main goal in terms of the code. We are satisfied with our implementation of the models themselves. PyTorch Lightning provides a very convenient framework for training and testing the models, and most architectural details (e.g. depth, number of channels) can be customized easily using the command line. One area where our practices could have been improved is in the script for making the synthetic datasets. Currently, this script does not parse command line arguments for configuration, and certain aspects of the data, such as how many enhancers and silencers regulate each gene, are hard coded. Additionally, there is no argument for changing the sizes of the generated train, test, and validation datasets. We would definitely improve this functionality if we were to write this code again.

E. Future Directions

In addition to the suggestions offered above, we now propose a few potential extensions of this work. One of the fundamental limitations of the Epi-GraphReg method is that the chromatin interaction graph reflects binary relationships: two parts of the

genome are either in close contact with one another or they are not. Extending this method to use edge weighted graphs, with edge weights reflecting the amount of contact may be beneficial to predicting the level of gene expression.

Another limitation of this method is that the chromatin interaction graph is assumed to be static and fixed for any particular chromosome. This is not a realistic assumption, since chromatin conformation can exhibit large dynamic changes over time [10]. Incorporating temporal conformation data presents new computational challenges, but it could also provide greater insights into the complex mechanisms of gene regulation.

A final potential extension of this work would be to single-cell epigenetic and gene expression data. Current studies use bulk assays to obtain data for particular cell types, rather than individual cells. Extending the computational methods to this challenging regime may provide new insights into gene regulation and cell differentiation, and may also have important applications in domains such as personalized medicine.

VII. Acknowledgements

The author would like to thank Dr. Jianrong Wang for offering helpful suggestions regarding the choice of evaluation metrics and potential experiments to explore the importance of graph topology. The author also thanks Hien Le for offering ideas for potential ablation studies, as well as suggestions for improving the exposition of this work.

References

[1] T. Shiraki et al. "Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage". In: *Proc Natl Acad Sci U S A* 100.26 (Dec. 2003), pp. 15776–15781.

[2] A. Karbalayghareh, M. Sahin, and C. S. Leslie. "Chromatin interaction-aware gene regulatory modeling with graph attention networks". In: *Genome Res* 32.5 (May 2022), pp. 930–944.

[3] Y. Luo et al. "New developments on the Encyclopedia of DNA Elements (ENCODE) data portal". In: *Nucleic Acids Res* 48.D1 (Jan. 2020), pp. D882–D889.

[4] H. U. Osmanbeyoglu et al. "Chromatin-informed inference of transcriptional programs in gynecologic and basal breast cancers". In: *Nat Commun* 10.1 (Sept. 2019), p. 4369.

[5] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. 2017. arXiv: 1607.08022 [cs.CV].

[6] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG].

[7] Petar Veličković et al. *Graph Attention Networks*. 2018. arXiv: 1710 . 10903 [stat.ML].

[8] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[9] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2017. arXiv: 1609.02907 [cs.LG].

[10] K. G. Alavattam et al. "Dynamic chromatin organization and regulatory interactions in human endothelial cell differentiation". In: *Stem Cell Reports* 18.1 (Jan. 2023), pp. 159–174.