1 Introduction

Distinct combinations of histone modifications are associated with different classes of functional genomic elements such as promoters, enhancers, and genes (Consortium et al., 2015). Chromatin immunoprecipitation sequencing (ChIP-seq) experiments are commonly used to obtain genome-wide profiles of histone modifications associated with different types of functional genomic elements. However, the quality of histone ChIP-seq data is affected by a myriad of experimental parameters such as the amount of input DNA, antibody specificity, ChIP enrichment, and sequencing depth. Making accurate inferences from chromatin profiling experiments that involve diverse experimental parameters is challenging.

Results: We introduce a convolutional denoising algorithm, Coda, that uses convolutional neural networks to learn a mapping from suboptimal to high-quality histone ChIP-seq data. This overcomes various sources of noise and variability, substantially enhancing and recovering signal when applied to low-quality chromatin profiling datasets across individuals, cell types, and species. Our method has the potential to improve data quality at reduced costs. More broadly, this approach—using a high-dimensional discriminative model to encode a generative noise process—is generally applicable to other biological domains where it is easy to generate noisy data but difficult to analytically characterize the noise or underlying data distribution.

Abstract

Motivation: Chromatin immunoprecipitation sequencing (ChIP-seq) experiments are commonly used to obtain genome-wide profiles of histone modifications associated with different types of functional genomic elements. However, the quality of histone ChIP-seq data is affected by a myriad of experimental parameters such as the amount of input DNA, antibody specificity, ChIP enrichment, and sequencing depth. Making accurate inferences from chromatin profiling experiments that involve diverse experimental parameters is challenging.

Results: We introduce a convolutional denoising algorithm, Coda, that uses convolutional neural networks to learn a mapping from suboptimal to high-quality histone ChIP-seq data. This overcomes various sources of noise and variability, substantially enhancing and recovering signal when applied to low-quality chromatin profiling datasets across individuals, cell types, and species. Our method has the potential to improve data quality at reduced costs. More broadly, this approach—using a high-dimensional discriminative model to encode a generative noise process—is generally applicable to other biological domains where it is easy to generate noisy data but difficult to analytically characterize the noise or underlying data distribution.

Availability: https://github.com/kundajelab/coda

Contact: akundaje@stanford.edu
certain widths and certain shapes. This means that a noisy signal can be denoised by a model that encodes prior expectations of what a clean signal should look like, just as humans use the regular structure in speech to decode noisy speech signals. Second, histone marks are correlated; thus, one noisy mark can be denoised using information from other noisy marks. Third, neural networks excel at flexibly learning complex non-linear relationships when given large amounts of data, making them ideal for genome-wide applications. Indeed, neural networks have recently been successfully applied to many biological domains (Angermueller et al., 2016b): for example, they have been used to predict regulatory sequence for genome-wide applications. Indeed, neural networks have recently been successfully applied to many biological domains (Angermueller et al., 2016b): for example, they have been used to predict regulatory sequence

to accomplish two tasks: a regression task to predict histone ChIP signal (i.e., the fold enrichment of ChIP reads over input DNA control) and a binary classification task to predict the presence or absence of a significant histone mark peak (Fig. 2). In total, if a given experiment has \( M \) marks, then we train \( 2M \) models separately (one regression and one classification model for each mark). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.

For computational efficiency, we first bin the genome into \( 25 \text{bp} \) bins, averaging the signal in each bin. Let \( L \) be the number of bins in the genome (i.e., the length of the genome divided by 25). Each individual model takes in an \( M \times L \) input matrix \( X \) and returns a \( 1 \times L \) output vector \( Y \) representing the predicted high-quality signal (in the regression setting) or peak calls (in the classification setting). It does this by feeding the noisy data through a first convolutional layer, a rectified linear unit (ReLU) layer, a second convolutional layer, and then a final ReLU or sigmoid layer (for regression or classification, respectively). For the first convolutional layer, we use 6 convolutional filters, each \( 51 \text{bins} \) in length; for the second convolutional layer, we use a single filter of length \( 1001 \). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.

For computational efficiency, we first bin the genome into \( 25 \text{bp} \) bins, averaging the signal in each bin. Let \( L \) be the number of bins in the genome (i.e., the length of the genome divided by 25). Each individual model takes in an \( M \times L \) input matrix \( X \) and returns a \( 1 \times L \) output vector \( Y \) representing the predicted high-quality signal (in the regression setting) or peak calls (in the classification setting). It does this by feeding the noisy data through a first convolutional layer, a rectified linear unit (ReLU) layer, a second convolutional layer, and then a final ReLU or sigmoid layer (for regression or classification, respectively). For the first convolutional layer, we use 6 convolutional filters, each \( 51 \text{bins} \) in length; for the second convolutional layer, we use a single filter of length \( 1001 \). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.

For computational efficiency, we first bin the genome into \( 25 \text{bp} \) bins, averaging the signal in each bin. Let \( L \) be the number of bins in the genome (i.e., the length of the genome divided by 25). Each individual model takes in an \( M \times L \) input matrix \( X \) and returns a \( 1 \times L \) output vector \( Y \) representing the predicted high-quality signal (in the regression setting) or peak calls (in the classification setting). It does this by feeding the noisy data through a first convolutional layer, a rectified linear unit (ReLU) layer, a second convolutional layer, and then a final ReLU or sigmoid layer (for regression or classification, respectively). For the first convolutional layer, we use 6 convolutional filters, each \( 51 \text{bins} \) in length; for the second convolutional layer, we use a single filter of length \( 1001 \). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.

For computational efficiency, we first bin the genome into \( 25 \text{bp} \) bins, averaging the signal in each bin. Let \( L \) be the number of bins in the genome (i.e., the length of the genome divided by 25). Each individual model takes in an \( M \times L \) input matrix \( X \) and returns a \( 1 \times L \) output vector \( Y \) representing the predicted high-quality signal (in the regression setting) or peak calls (in the classification setting). It does this by feeding the noisy data through a first convolutional layer, a rectified linear unit (ReLU) layer, a second convolutional layer, and then a final ReLU or sigmoid layer (for regression or classification, respectively). For the first convolutional layer, we use 6 convolutional filters, each \( 51 \text{bins} \) in length; for the second convolutional layer, we use a single filter of length \( 1001 \). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.

For computational efficiency, we first bin the genome into \( 25 \text{bp} \) bins, averaging the signal in each bin. Let \( L \) be the number of bins in the genome (i.e., the length of the genome divided by 25). Each individual model takes in an \( M \times L \) input matrix \( X \) and returns a \( 1 \times L \) output vector \( Y \) representing the predicted high-quality signal (in the regression setting) or peak calls (in the classification setting). It does this by feeding the noisy data through a first convolutional layer, a rectified linear unit (ReLU) layer, a second convolutional layer, and then a final ReLU or sigmoid layer (for regression or classification, respectively). For the first convolutional layer, we use 6 convolutional filters, each \( 51 \text{bins} \) in length; for the second convolutional layer, we use a single filter of length \( 1001 \). Each individual model optionally makes use of the noisy ChIP-seq data from all available marks but outputs only one target histone mark. This allows us to learn separate features for each mark and task while still leveraging information from multiple input histone marks; we find empirically that this improves performance.
completely separate set of computations for each bin and are therefore significantly more computationally expensive when it comes to making predictions across the entire genome.

Training and evaluation
We applied Coda to three distinct sources of noise: low sequencing depth, low cell input, and low ChIP enrichment. In all cases, the inputs to our model were noisy signal measurements of multiple histone marks (see Data Availability and Processing for more details), and we trained separate models to predict the high-quality signal and peak calls for each target mark.

For the regression tasks (predicting signal), we evaluated performance by computing the Pearson correlation and mean squared error (MSE) between the predicted and measured high-quality fold-enrichment signal profiles after an inverse hyperbolic sine transformation, which reduced the dominance of outliers. We compared this to the baseline performance obtained by directly comparing the noisy and high-quality signal profiles of the target mark (after the same inverse hyperbolic sine transformation).

For the classification tasks (predicting presence or absence of a peak), we compared our model's output to peaks called by the MACS2 peak caller (Feng et al., 2012) on the high-quality signal for the target mark. As our dataset is unbalanced – peaks only make up a small proportion of the genome – we evaluated performance by computing the area under the precision-recall curve (AUPRC), a standard measure of classification performance for unbalanced datasets (Davis and Goadrich, 2006). We compared the AUPRC of our model to a baseline obtained by comparing MACS2 peaks on the noisy data for the target mark to those obtained from the high-quality data for the target mark (see Data Availability and Processing for further details on dataset preparation).

We trained our models on 50,000 positions randomly sampled from peak regions of the genome and 50,000 positions sampled from non-peak regions, sampling from each autosome with equal likelihood. We defined peak regions using the output mark of interest and with the high-quality data. Further increasing dataset size did not increase performance; as each sample covered 25,025bp, 100,000 samples had good coverage of the entire genome. We selected the training dataset to be balanced because a uniformly drawn dataset would have had very few peaks; making it difficult for the model to learn to predict at peak regions; however, the test results reported in this paper are on the entire (unbalanced) genome. We used the Keras package (Francois Chollet, 2015) for training and AdaGrad (Duchi et al., 2011) as the optimizer, stopping training if validation loss did not improve for three consecutive epochs. We did not observe overfitting with our models (train and test error were comparable), and therefore opted not to use common regularization techniques such as dropout (Srivastava et al., 2014).

We chose model hyperparameters and architecture through hold-out validation on the low-sequencing-depth denoising task with GM12978 as the training cell line (Kasowski et al., 2013), holding out a random 20% subset of the training data for validation; this task will be discussed in more detail in the next section. The model architecture described above (6 convolutional filters each 51 bins in length in the first layer, and 1 convolutional filter of length 1001 in the second layer) yielded optimal validation performance out of the configurations we tried (varying the number of convolutional filters and the lengths of the filters by up to an order of magnitude). Adding an additional layer to the neural network brought a modest increase in performance at the cost of more computation time and complexity. To be sure that our model architecture generalized, we used the same architecture and hyperparameters for all denoising tasks without any further tuning.

3 Results
Removing noise from low sequencing depth data
A minimum of 40-50M reads is recommended for optimal sensitivity for histone ChIP-seq experiments in human samples targeting most canonical histone marks (Jung et al., 2014). As adhering to this standard can often be infeasible due to cost and other limitations, a substantial proportion of publicly available datasets do not meet these standards. Motivated by these constraints, we tested whether our model could recover high-read depth signal from low-read depth experiments.

Training and testing on the same cell type across different individuals
We evaluated Coda on lymphoblastoid cell lines (LCLs) derived from six individuals of diverse ancestry (European (CEU), Yoruba (YRI), Japanese, Han Chinese, San) (Kasowski et al., 2013). We used the CEU-derived cell line (GM12878) to train our model to reconstruct the high-depth signal (100M+ reads per mark: exact numbers in Data Availability and Processing) from a simulated noisy signal derived by subsampling 1M reads per mark. On the other five cell lines, Coda significantly improved Pearson correlation between the full and noisy signal (Fig. 3A, left) and the accuracy of peak calling (Fig. 3A, right). Using just 1M reads per mark, the predicted output of our model was equivalent in quality to signal derived from 15M+ reads (H3K27ac) and 25M+ reads (H3K36me3) (Fig. 3B). Fig. 4 shows how Coda can accurately reconstruct histone modification levels at the promoter of the PA53 gene, a master transcription factor required for differentiation into the B-lymphoid lineage (Nutt et al., 1999).

We confirmed Coda was not simply memorizing the profile of the training cell line (GM12878) and copying it to the test cell lines by examining differential regions, called by DESeq (Anders and Huber, 2010), between GM12878 and the other cell lines (Kasowski et al., 2013). Coda improved correlation and peak-calling even in those regions (Table 1). Similarly, it also improved correlation on the regions of the genome with enriched signal, i.e., called as statistically significant peaks (Table 2).

Table 1. Denoising differential regions (diff. reg.) between test cell line GM18526 and training cell line GM12878. Performance reported is improvement of the denoised model over baseline (original, subsampled reads) on the test cell line. In parentheses we report the baseline results followed by the denoised results. Peak-calling results on H3K27me3 are omitted due to low number of differential regions; all results on H3K36me3 are omitted due to low number of differential regions.

<table>
<thead>
<tr>
<th>histone mark</th>
<th>MSE (diff. reg.)</th>
<th>Pearson R (diff. reg.)</th>
<th>AUPRC (diff. reg.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H3K4me3</td>
<td>-75%</td>
<td>+39%</td>
<td>+14%</td>
</tr>
<tr>
<td>H3K27me3</td>
<td>-78%</td>
<td>+55%</td>
<td>+106%</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>-78%</td>
<td>+37%</td>
<td>+59%</td>
</tr>
<tr>
<td>H3K27ac</td>
<td>+15%</td>
<td>+14%</td>
<td>-</td>
</tr>
<tr>
<td>H3K27ac</td>
<td>-106%</td>
<td>-11%</td>
<td>+14%</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>+13%</td>
<td>+39%</td>
<td>+6%</td>
</tr>
</tbody>
</table>

Training and testing on different cell types across different individuals
We next assessed if Coda could be trained on one cell type in one individual and used to denoise low-sequencing-depth data from a different cell type in a different individual. As above, the model was trained to output high-depth data (30M reads) from low-depth data (1M reads). We used histone ChIP-seq data spanning T-cells (E037), monocytes (E029), mesenchymal stem cells (MSCs, E026), and fibroblasts (E056) from the Roadmap Epigenomics Consortium (Consortium et al., 2015). Coda substantially improved the quality of the low-depth signal on the test cell type for all pairs of cell types (Table 3), illustrating its ability to denoise low-depth data on a cell type even if high-depth training data for that cell type is not available.

Table 2. Denoising differential regions (diff. reg.) between test cell line GM18526 and training cell line GM12878. Performance reported is improvement of the denoised model over baseline (original, subsampled reads) on the test cell line. In parentheses we report the baseline results followed by the denoised results. Peak-calling results on H3K27me3 are omitted due to low number of differential regions; all results on H3K36me3 are omitted due to low number of differential regions.

<table>
<thead>
<tr>
<th>histone mark</th>
<th>MSE (diff. reg.)</th>
<th>Pearson R (diff. reg.)</th>
<th>AUPRC (diff. reg.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H3K4me3</td>
<td>-85%</td>
<td>+95%</td>
<td>+83%</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>-75%</td>
<td>+14%</td>
<td>+63%</td>
</tr>
<tr>
<td>H3K27me3</td>
<td>-78%</td>
<td>+14%</td>
<td>+63%</td>
</tr>
<tr>
<td>H3K27ac</td>
<td>-106%</td>
<td>+11%</td>
<td>+106%</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>-78%</td>
<td>+106%</td>
<td>-</td>
</tr>
</tbody>
</table>

The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under a CC-BY-NC 4.0 International license.
Coda outperforms linear baselines

We compared Coda to a linear and logistic regression baseline for signal denoising and peak calling, respectively. In both cases, we used an input region of the same size as Coda (i.e., 25,025bp centered on the location to be predicted, binned into 250bp bins). As noted above, the desire for computational efficiency in making genome-wide predictions across multiple marks limits the complexity of models that would be practically useful in genome-wide prediction.

When evaluated in the same cell type, different individual setting, Coda achieved 3x lower MSE on peak regions and 2x lower MSE on differential regions, with similar (very slightly better) MSE and correlation across the whole genome. This implies that Coda is better able to learn to match the exact values of the signal tracks on “difficult” regions (i.e., where there is the greatest deviation from the training signal), even though the linear model matches the rough shape. These regions are important to predict well because they can give insight into the differences between individuals and cell types.

We note that many forms of smoothing can be represented via linear regression. For example, a standard Gaussian filter can be interpreted as taking a linear combination of surrounding points with fixed coefficients. The comparison against a linear regression baseline therefore sets an upper

---

**Table 2. Denoising peak regions between test cell line GM18526 and training cell line GM12878.** Performance reported is improvement of the denoised model over baseline (original, subsampled reads) on the test cell line. In parentheses we report the baseline results followed by the denoised results.

<table>
<thead>
<tr>
<th>Histone mark</th>
<th>MSE (peaks)</th>
<th>Pearson R (peaks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H3K4me1</td>
<td>-86% (3.69, 0.49)</td>
<td>+56% (0.44, 0.70)</td>
</tr>
<tr>
<td>H3K4me3</td>
<td>-83% (2.93, 0.50)</td>
<td>+11% (0.78, 0.87)</td>
</tr>
<tr>
<td>H3K27ac</td>
<td>-87% (3.36, 0.43)</td>
<td>+28% (0.65, 0.83)</td>
</tr>
<tr>
<td>H3K27me3</td>
<td>-96% (2.20, 0.23)</td>
<td>+103% (0.18, 0.36)</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>-93% (3.78, 0.25)</td>
<td>+120% (0.32, 0.70)</td>
</tr>
</tbody>
</table>

---

**Table 3. Cross cell-type experiments.** Rows are train cell type, while columns are test cell type. In parentheses we report the baseline results followed by the denoised results, averaged across all histone marks used.

<table>
<thead>
<tr>
<th>Histone mark</th>
<th>Monocytes</th>
<th>MSCs</th>
<th>Fibroblasts</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-cells</td>
<td>+33% (0.51, 0.67)</td>
<td>+58% (0.44, 0.70)</td>
<td>+78% (0.36, 0.65)</td>
</tr>
<tr>
<td>Monocytes</td>
<td>-</td>
<td>+59% (0.44, 0.70)</td>
<td>+79% (0.36, 0.65)</td>
</tr>
<tr>
<td>MSCs</td>
<td>-</td>
<td>-</td>
<td>+81% (0.36, 0.66)</td>
</tr>
<tr>
<td>AUPRC</td>
<td>+116% (0.31, 0.66)</td>
<td>+186% (0.31, 0.72)</td>
<td>+94% (0.35, 0.69)</td>
</tr>
<tr>
<td>Monocytes</td>
<td>+139% (0.31, 0.73)</td>
<td>+94% (0.35, 0.69)</td>
<td>+100% (0.35, 0.71)</td>
</tr>
<tr>
<td>MSCs</td>
<td>-</td>
<td>-</td>
<td>+100% (0.35, 0.71)</td>
</tr>
</tbody>
</table>
Table 4. Low-cell-input experiments. We report improvement of the denoised model output over baseline (original low-input experiments), as compared to high-input experiments. In parentheses we report the baseline results followed by the denoised output over baseline (original low-input experiments), as compared to high-input experiments.

<table>
<thead>
<tr>
<th>Mark</th>
<th>ULI-NChIP</th>
<th>MOWChIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>H3K4me3</td>
<td>-61%</td>
<td>-22%</td>
</tr>
<tr>
<td>H3K9me3</td>
<td>-46%</td>
<td>-12%</td>
</tr>
<tr>
<td>H3K27me3</td>
<td>-41%</td>
<td>+21%</td>
</tr>
</tbody>
</table>

Comparisons to denoising and imputation

Next, we studied Coda’s performance in two additional settings: pure denoising (using the noisy target mark as the only input mark) and imputation from noise (using all noisy histone marks but the target mark as the input marks). This is in contrast to the standard setting described above, where we use all noisy histone marks, including the noisy version of the target mark, to recover a high-quality version of the target mark.

In the denoising case, Pearson correlation dropped by 0.03 points and AUPRC dropped by 0.05, on average, compared to when all marks were used as input. Thus, additional marks provided some information, but the denoised signal was still substantially better than the original subsampled signal.

In the imputation case, performance dropped somewhat on the narrow marks (H3K4me1, H3K4me3, H3K27ac) –0.12 correlation, –0.18 AUPRC) and dropped more on the broad marks (H3K27me3, H3K36me3; –0.29 correlation, –0.30 AUPRC). The gap in correlation was even larger within peak regions. Thus, having a noisy version of the target mark substantially boosts recovery of the high-quality signal.

Removing noise from low cell input

Conventional ChIP-seq protocols require a large number of cells to reach the necessary sequencing depth and library complexity (Brand’Amour et al., 2015; Cao et al., 2015), precluding profiling when input material is limited. Several ChIP-seq protocols were recently developed to address this problem. We studied ULI-NChIP-seq (Brand’Amour et al., 2015) and MOWChIP-seq (Cao et al., 2015), which use low cell input (10^2-10^3 cells) to generate signal that is highly correlated, when averaged over bins of size 2-4kb, with experiments with high cell input. However, at a finer scale of 25bp, the low-input signals from both protocols are poorly correlated with the high-input signals (Table 4).

We thus used Coda to recover high-resolution, high-cell-input signal from low-cell-input signal specific to each protocol. For ULI-NChIP-seq, we used a single mouse embryonic stem cell dataset (Brand’Amour et al., 2015). For MOWChIP-seq, we trained on data from the human LCL, GM12878 and tested on hematopoietic stem and progenitor cells (HSPCs) from mouse fetal liver (Cao et al., 2015). Coda successfully denoised the low-input signal from both protocols (Table 4). Fig. 5 illustrates our model denoising MOWChIP-seq signal across the Runx1 gene, a key regulator of HSPCs (North et al., 2002): the results of peak calling were too noisy, even on the original 10,000-cell data, to allow for any qualitative judgment of improvement.
enrichment experiments. In parentheses we report the baseline results followed by the model output over baseline (low-enrichment experiments), as compared to high-enrichment experiments. We report improvement of the denoised signal over the corrupted signal, more closely resembling the uncorrupted signal. In contrast, the denoised signal was significantly biased by the reads from the control experiment and had significantly lower levels of H3K27me3 at DHSs, whereas the denoised signal had low levels of H3K4me1 and H3K4me3 peaks that were missed in the noisy data while removing a spurious H3K27me3 peak call. Note that we show the noisy peak calls to allow for comparisons; Coda uses only the noisy signal, not the peak calls, as input. The signal tracks are in arcsinh units, with the following y-axis scales: H3K27ac: 0-60, H3K27me3, H3K36me3, and H3K4me1: 0-40, H3K4me2: 100. The shading of the peak tracks that the model outputs represent the strength of the peak call on a scale of 0-1.

Table 5. Low-enrichment experiments. We report improvement of the denoised model output over baseline (low-enrichment experiments), as compared to high-enrichment experiments. In parentheses we report the baseline results followed by the denoised results.

<table>
<thead>
<tr>
<th>Histone Mark</th>
<th>MSE (0.35, 0.09)</th>
<th>Pearson R (0.64, 0.91)</th>
<th>AUPRC (0.29, 0.92)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H3K4me1</td>
<td>-75%</td>
<td>+42%</td>
<td>+215%</td>
</tr>
<tr>
<td>H3K4me3</td>
<td>-86%</td>
<td>+54%</td>
<td>+94%</td>
</tr>
<tr>
<td>H3K27ac</td>
<td>-70%</td>
<td>+57%</td>
<td>+121%</td>
</tr>
<tr>
<td>H3K27me3</td>
<td>-61%</td>
<td>+88%</td>
<td>+242%</td>
</tr>
<tr>
<td>H3K36me3</td>
<td>-82%</td>
<td>+47%</td>
<td>+168%</td>
</tr>
</tbody>
</table>

4 Conclusion

We describe a convolutional denoising algorithm, Coda, that uses paired noisy and high-quality samples to substantially improve the quality of new, noisy ChIP-seq data. Our approach transfers information from generative noise processes (e.g., mixing in control reads to simulate low-enrichment, or performing low-input experiments) to a flexible discriminative model that can be used to denoise new data. We believe that a similar approach can be used in other biological assays, e.g., ATAC-seq and DNase-seq (Buenrostro et al., 2013; Crawford et al., 2006), where it is near impossible to analytically characterize all types of technical noise or the overall data distribution but possible to generate noisy versions of high-quality samples through experimental or computational perturbation. This can significantly reduce cost while maintaining or even improving quality, especially in high-throughput settings or when dealing with limited amounts of input material (e.g., in clinical studies).

An important caveat to our work is that the performance of Coda depends strongly on the similarity of the noise distributions and underlying data distributions in the test and training sets. For example, Coda expects that the relationships between different histone marks should be conserved between the test and training set. Thus, applying a set of trained Coda
models to data that is very different from what it was trained on is unlikely to work. We also assume that the noise parameters in the test data are known in advance, e.g., the sequencing depth, the number of input cells, or the level of ChIP enrichment. In some cases (e.g., the low-sequencing-depth and low-cell-input settings) this is true, but in others (e.g., the low-enrichment setting) it is not always possible. An important direction for future work is to make Coda more robust, for example, training a single model over various settings of the noise parameters and various cell types could improve the generalizability of the models.

To further improve performance, more complex neural network architectures could also be explored: for example, using recurrent neural networks (Sutskever et al., 2014) to explicitly model long-range spatial correlations in the genome; multi-tasking across output marks instead of training separate models for each mark; or using deeper networks.

Another avenue for future work is exploring using more than just the noisy histone ChIP-seq data at test time. In this work, we use only the noisy data at test time, training our models to transform it into high-quality data. In reality, at test time we might have access to other data; for example, we might also have the DNA sequence of the test sample or access to high-quality ChIP-seq data on a closely related cell type. Other work has used DNA sequence to predict transcription factor binding (Alipanahi et al., 2015; Zhou and Troyanskaya, 2015), chromatin accessibility (Kelley et al., 2015), and methylation status (Angermueller et al., 2016a). A natural next step would be to combine the ideas from these methods with ours, e.g., by having a separate convolutional module in our neural network that incorporates sequence information and joins with the ChIP-seq module at an intermediate layer. Others have also used high-quality ChIP-seq data from closely related cell types for imputation (Ernst and Kellis, 2015), combining this with our denoising approach could help to avoid a potential pitfall of imputation approaches, namely the loss of cell-type-specific signal, while improving the accuracy of our denoised output.

Below, we provide a link to a script that trains a model for low-sequencing-depth noise using the LCL data described above. Since the type of noise can vary from context to context, we also provide the code for the general Coda framework to allow for developers of new protocols (e.g., new low-cell-count techniques) or core facilities that have high throughput to train Coda with data specific to their context.

Data Availability and Processing

Datasets

We used the following publicly-available GEO datasets in this work:

1. GSE50893 for ChIP-seq data on LCLs (Kasowski et al., 2013)
2. GSE63323 for ULI-NChIP-seq data (Brind’Amour et al., 2015)
3. GSE65516 for MOWChIP-seq data (Cao et al., 2015)
4. GSM736620 for DNase I hypersensitive peaks (Bernstein et al., 2012)

For the low-sequencing-depth experiments, the full depth for GM12878 (training set) was 171M (million reads) for H3K4me1, 168M for H3K4me3, 328M for H3K27ac, 265M for H3K27me3, and 123M for H3K36me3. The full depth for GM18526 (test set) was 120M for H3K4me1, 136M for H3K4me3, 225M for H3K27ac, 138M for H3K27me3, and 123M for H3K36me3. For all of the above experiments, we also used data from the control experiments (no antibody) as input. Lastly, for the low-cell-input experiments, we used H3K4me1, H3K4me3, and H3K27me3 from the ULI-NChIP-seq dataset and H3K4me3 and H3K27ac from the MOWChIP-seq dataset.

Low-cell-input datasets

The ULI-NChIP-seq (Brind’Amour et al., 2015) and MOWChIP-seq (Cao et al., 2015) papers provided several datasets corresponding to different numbers of input cells used. For each protocol, we used the datasets with the lowest number of input cells as the noisy input data (ULI-NChIP-seq: 10^3 cells for H3K9me3 and H3K27me3, 5x10^3 cells for H3K4me3, MOWChIP-seq: 10^2 cells) and the datasets with the highest number of input cells as the gold-standard, high-quality data (ULI-NChIP-seq: 10^6 cells for H3K9me3, 10^5 cells for H3K4me3 and H3K27me3, MOWChIP-seq: 10^4 cells). The ULI-NChIP-seq data had matching low- and high-input experiments only for a single cell type, so we divided it into chr5-19 for training, chr4-4 for validation, and chr1-2 for testing.

Code, data, and browser track availability

Our code is available on Github at https://github.com/kundajelab/coda, including a script that downloads pre-processed
data and replicates the low-sequence-depth experiments described above, as well as code for processing new data.

The figures of browser tracks (Figures 4, 5, and 6) shown above were taken from the Wash U Epigenome Browser (Zhou and Wang, 2012). Links to the entire browser tracks are as follows:

- Fig. 4, low-sequence-depth experiments on LCL GM12878: http://epigenomeweb.wustl.edu/browse?genome=hg19&session=KEDV2G8GB03&statusId=107864126
- Fig. 5, low-cell-count experiments on mouse HSCPs: http://epigenomeweb.wustl.edu/browse?genome=hg19&session=KEDV2G8GB03&statusId=1913128468
- Fig. 6, low-enrichment experiments on LCL GM12878: http://epigenomeweb.wustl.edu/browse?genome=hg19&session=KEDV2G8GB03&statusId=1913128468

Acknowledgements

We thank Jin-Wook Lee for his assistance with the AQUAS pipeline and Kyle Liu, Line Capuano, and Nasa Sinnott-Armstrong for their helpful feedback and suggestions.

Funding

EP acknowledges support from a Hertz Fellowship and an NDSEG Fellowship. This work was also supported by NIH grants DP2-GM-123485 and 1R01ES025009-01.

References

Manchester, R., Alkhas, D., Scherer, S. et al. (2013) Transcription factor EBF1 is essential for the maintenance of B cell identity and prevents of alternative fates in committed cells. Nature Immunology, 14, 867–875. URL http://www.nature.com/dofinder/10.1038/ni.2641
Denoising Genome-wide Histone ChIP-seq with Convolutional Neural Networks


