Pacific Symposium on Biocomputing 25:367-378(2020)

Machine learning algorithms for simultaneous supervised detection of peaks in multiple samples and cell types

Toby Dylan Hocking† and Guillaume Bourque

School of Informatics, Computing, and Cyber Systems, Northern Arizona University, Flagstaff, AZ 86011, USA
†*E-mail: toby.hocking@nau.edu*

McGill University and Genome Quebec Innovation Centre, Montreal, QC, Canada
E-mail: guil.bourque@mcgill.ca

Joint peak detection is a central problem when comparing samples in epigenomic data analysis, but current algorithms for this task are unsupervised and limited to at most 2 sample types. We propose PeakSegPipeline, a new genome-wide multi-sample peak calling pipeline for epigenomic data sets. It performs peak detection using a constrained maximum likelihood segmentation model with essentially only one free parameter that needs to be tuned: the number of peaks. To select the number of peaks, we propose to learn a penalty function based on user-provided labels that indicate genomic regions with or without peaks in specific samples. In comparisons with state-of-the-art peak detection algorithms, PeakSegPipeline achieves similar or better accuracy, and a more interpretable model with overlapping peaks that occur in exactly the same positions across all samples. Our novel approach is able to learn that predicted peak sizes vary by experiment type.

Keywords: Epigenome; ChIP-seq; ATAC-seq; Joint; Multi-sample; Peak Detection; Supervised; Machine Learning.

1. Introduction: Joint supervised peak detection in ChIP-seq data

Chromatin immunoprecipitation sequencing (ChIP-seq) is an experimental technique used for genome-wide profiling of histone modifications and transcription factor binding sites.[1] Each experiment yields a set of sequence reads which are aligned to a reference genome, and then the number of aligned reads are counted at each genomic position. To compare samples at a given genomic position, biologists visually examine coverage plots such as Figure 1 for presence or absence of common "peaks." In machine learning terms, a single sample over B base positions is a vector of non-negative count data $\mathbf{z} \in \mathbb{Z}_+ = \{0, 1, \dots\}^B$ and a peak detector is a binary classifier $c : \mathbb{Z}_+^B \to \{0, 1\}^B$. The positive class is peaks and the negative class is background noise. Importantly, peaks and background occur in long contiguous segments across the genome.

In this paper we use a supervised learning framework with labels from low-throughput experiments[12] or visual inspection of genome plots.[4] Previous work has shown that a relatively

small number of labeled regions/samples are required in order to learn accurate predictive models.[3,4] For each labeled genomic region $i \in \{1, \ldots, n\}$ there is a set of count data $\mathbf{z}_i$ and labels L_i ("noPeaks," "peaks," etc. as in Figure 1). These labels define a non-convex label error function

$$E[c(\mathbf{z}_i), L_i] = \text{FP}[c(\mathbf{z}_i), L_i] + \text{FN}[c(\mathbf{z}_i), L_i] \tag{1}$$

which counts the number of false positive (FP) and false negative (FN) regions, so it takes values in the non-negative integers. In this framework the goal of learning is to find a peak detection algorithm c that minimizes the number of incorrect labels (1) on a test set of data:

$$\underset{c}{\text{minimize}} \sum_{i \in \text{test}} E[c(\mathbf{z}_i), L_i]. \tag{2}$$

In practical situations we have $S > 1$ samples, and a matrix $\mathbf{Z} \in \mathbb{Z}_+^{B \times S}$ of count data. For example in Figure 1 we have $S = 4$ samples. In these data we are not only interested in accurate peak detection in individual samples, but also in detecting the differences between

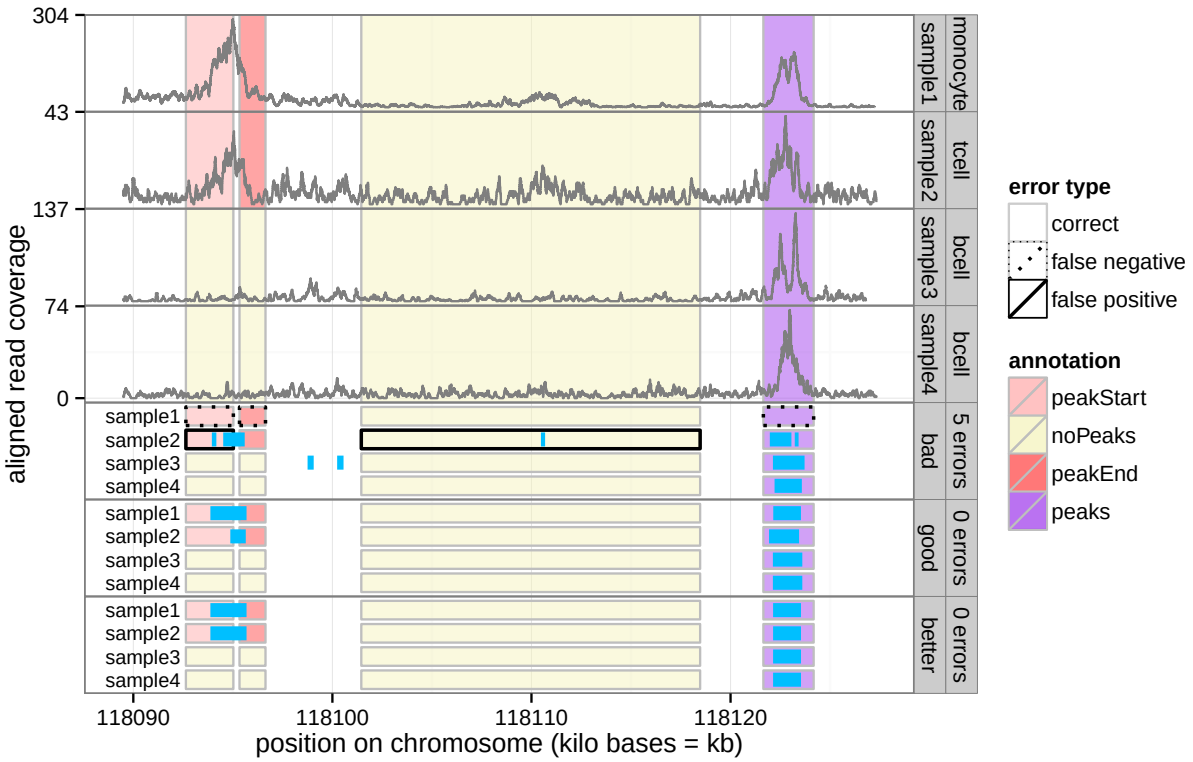

Fig. 1. Labeled ChIP-seq coverage data for $S = 4$ samples (top 4 panels) with 3 peak models (bottom 3 panels). Colored rectangles are labels which an expert genomic scientist has provided to indicate presence/absence of significant peaks: noPeaks means there should be no overlapping peaks, and peaks means there should be at least one; peakStart / peakEnd mean there should be exactly 1 peak start/end somewhere in the region. An ideal peak model minimizes the number of incorrect labels (false positives are too many peaks and false negatives are not enough peaks). The "good" model achieves 0 errors but the "better" model is more interpretable since overlapping peaks in different samples always occur in the exact same positions.

samples. In particular, an ideal model of a multi-sample data set is a **joint** peak detector with identical overlapping peak positions across samples (the "better" model in Figure 1).

1.1. *Contributions and organization*

The main contribution of this paper is PeakSegPipeline, a joint peak calling pipeline for epigenomic data sets which can be trained using supervised learning algorithms. Unlike previous methods (Section 2), it is the first joint peak detector that explicitly models any number of sample types (Section 3). A secondary contribution is the JointZoom heuristic segmentation algorithm, which efficiently infers the most likely position of an overlapping common peak in several samples (Section 4). We show results in several epigenomic data sets in Section 5 and provide a discussion in Section 6.

2. Related work

2.1. *Single-sample ChIP-seq peak detectors*

There are many different unsupervised peak detection algorithms for epigenomic data analysis.[12–14] In this paper we propose a new pipeline based on a previously described constrained maximum segmentation model which was shown to achieve state-of-the-art peak detection accuracy.[6]

In the multi-sample setting of this paper, each single-sample peak detection algorithm may be applied independently to each sample. The drawback of these single-sample methods is that the predicted peaks do not occur in the same positions across each sample, so it is not straightforward to interpret the model in terms of similarities and differences between samples.

2.2. *Methods for several data sets*

This paper is concerned with models that predict peaks in several samples (perhaps of different cell types) of the same experiment type. For example, Figure 1 shows two bcell samples, one monocyte sample, and one tcell sample (all of the H3K4me3 experiment type). As far as we know, there are no existing algorithms that can jointly model peaks in such data.

The most similar peak detectors in the bioinformatics literature model several samples of the same cell type. For example, the JAMM algorithm of Ref. 9 can analyze several samples, but is limited to a single cell type since it assumes that each sample is a replicate with the exact same peak pattern. So to analyze the data of Figure 1 one would have to run JAMM three times (once for each cell type), and the resulting peaks would not be the same across cell types. Another example is the PePr algorithm of Ref. 16, which can model either one or two cell types, but is unsuitable for analysis of three or more cell types. In contrast, we propose a new model for several samples without limit on the number of cell types.

Although not the subject of this paper, there are several algorithms designed for the analysis of data from several different ChIP-seq experiments.[2,8,15] In these models, the input is one sample (e.g. monocyte cells) with different experiments such as sharp H3K4me3 and broad H3K36me3. Because different experiments have different peak sizes (see Section 5.2),

and our propopsed joint model assumes common peak positions across samples, it should be used separately on data from each experiment. For example, our model takes as input several samples (e.g. monocyte, tcell, bcell) for one experiment such as H3K4me3.

3. Models

We begin by summarizing the single-sample constrained optimal segmentation model,[6] and then introduce the multi-sample model.

3.1. *Finding the most likely peaks in a single sample*

This section describes the single-sample constrained optimal segmentation model of Ref. 6, which was shown to provide state-of-the-art peak predictions in several broad H3K36me3 and sharp H3K4me3 data sets.[4]

Given a single sample profile $\mathbf{z} = [\, z_1 \cdots z_B \,] \in \mathbb{Z}_+^B$ of aligned read counts on B bases, and a maximum number of peaks P, we define the constrained maximum likelihood segmentation problem as follows. First, we assume that there are $K = 2P + 1$ distinct segments, with odd numbered segments modeling background noise regions, and even numbered segments modeling peaks. This assumption is reasonable for epigenomic data, in which peaks are not expected on the boundaries of contigs.

Once a given model size K is fixed, the model parameters are the segment means and the changepoint positions. For each data point i on a given segment k, we assume that the data are independent and identically distributed $z_i \sim \text{Poisson}(u_k)$ with a segment-specific mean $u_k \in \mathbb{R}_+$. We use the Poisson distribution because it is a simple one-parameter model for non-negative count data; other distributions such as Negative Binomial could be considered in future work. We furthermore define integer-valued changepoint variables $t_0, t_1, \cdots, t_K$ which control where the segments begin/end. More precisely, for each segment $k \in \{1, \ldots, K\}$ the mean parameter u_k is assigned to data points $i \in \{t_{k-1} + 1, \cdots, t_k\}$, with $t_0 = 0$ and $t_K = B$.

Formally, the optimal cost in K segments up to B data points, subject to the up-down constraint on adjacent segment means, is defined as

$$\underset{\substack{u_1,\ldots,u_K \in \mathbb{R} \\ 0=t_0<t_1<\cdots<t_{K-1}<t_K=B}}{\text{minimize}} \quad \sum_{k=1}^{K} \sum_{i=t_{k-1}}^{t_k} u_k - z_i \log u_k \tag{3}$$

$$\text{subject to} \quad u_{k-1} \leq u_k \ \forall k \in \{2, 4, \ldots\}, \tag{4}$$

$$u_{k-1} \geq u_k \ \forall k \in \{3, 5, \ldots\}. \tag{5}$$

Note that the optimization objective of minimizing the Poisson loss (3) is equivalent to maximizing the Poisson likelihood. The constraints force the mean to alternate between increasing (4) and decreasing (5) at each changepoint. Thus the even numbered segments are interpreted as peaks, and the odd numbered segments are interpreted as background noise.

The original quadratic time algorithm that was proposed for the constrained optimal changepoint model is a heuristic;[6] it is not guaranteed to compute the optimal solution to (3). The optimal solution can be computed using an algorithm which is linear in the number of segments K and log-linear in the number of data B;[7] an even faster algorithm which is only $O(N \log N \log K)$ has been recently proposed.[5]

Note that since (3) is defined for a single sample, it may be independently applied to each sample in data sets such as Figure 1. However, any overlapping peaks in different samples will not necessarily occur in the same positions. In the next section we fix this problem by proposing the more interpretable multi-sample model.

3.2. *PeakSegJoint: finding the most likely common peak in* $0, \ldots, S$ *samples*

Recall that we have assumed that there are S samples to analyze for peaks. Let $Z_{i,s} \in \mathbb{Z}_+$ be the observed count data at base i for sample s. The joint model we propose is similar to the previous single-sample model in that it starts down, jumps up to a peak, and then jumps back down to background. There are two new assumptions with respect to the single-sample problem: (1) this region has at most one peak in each sample (e.g. Figure 2), and (2) any peaks in this region have common start/end positions across samples. Although this assumption may not be appropriate for all samples (e.g. mutations in histone or transcription factor binding sites), we have observed common peak positions across samples in the vast majority of normal epigenomic data (including in samples of different cell types).

More formally, we assume that each segment $k \in \{1, 2, 3\}$ has a corresponding mean vector $\mathbf{M}_k = [\, M_{k,1} \cdots M_{k,S}\,] \in \mathbb{R}_+^S$, with an element for each sample. There are two integer-valued changepoint variables t_1, t_2 which determine the common peak start/end. The number of samples with a common peak $P \in \{0, 1, \ldots, S\}$ must be fixed before solving the optimization problem:

$$\mathcal{L}_P = \min_{\substack{\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3 \in \mathbb{R}^S \\ 0 = t_0 < t_1 < t_2 < t_3 = B}} \sum_{s=1}^{S} \sum_{k=1}^{3} \sum_{i=t_{k-1}}^{t_k} M_{k,s} - Z_{i,s} \log M_{k,s} \tag{6}$$

$$\text{subject to} \quad \sum_{i=1}^{S} I[M_{1,s} < M_{2,s} > M_{3,s}] = P \tag{7}$$

$$\sum_{i=1}^{S} I[M_{1,s} = M_{2,s} = M_{3,s}] = S - P. \tag{8}$$

The objective function (6) is the Poisson loss, same as the previous problem, but now summed over all samples s. The first constraint (7) ensures there are exactly P samples which jump up to (and down from) the common peak. The second constraint (8) ensures that the mean of the other samples remain constant at the background noise level (no peak).

The proposed multi-sample peak detection problem (6) therefore results in a sequence of models (Figure 2). At one extreme, when $P = 0$, there are no common peaks, and all samples are predicted to be only background noise in this region. At the other extreme, when $P = S$, each sample has a common peak in this region. The intermediate P values result in common peaks in some samples but not others.

We propose to choose the number of samples with a common peak P by learning a penalty function using the labels, as previously described.[11] Briefly, we compute a simple feature vector for every joint peak detection problem, which is used as the input an a supervised machine learning problem. Features include several non-linear transformations of statistics such as number of data points, variance estimates, quantiles, etc. The output is an interval of penalty

values that results in minimal label errors. Large penalty values typically result in few peaks and therefore false negatives; small penalty values result in many peaks and false positives (Figure 2). Overall the learned function predicts a penalty value, and therefore P, the number of samples with a common peak.

4. Algorithms

4.1. *Heuristic discrete optimization for joint segmentation*

The joint optimization problem has a convex objective function and non-convex constraints (6). Explicitly computing the maximum likelihood and feasibility for all $O(B^2)$ possible peak start/endpoints is guaranteed to find the global optimum, but would take too much time in genomic regions with many bases B. Instead, we propose to find an approximate solution using a new discrete optimization algorithm called JointZoom (Algorithm 1).

The main idea of the JointZoom algorithm is to first zoom out (downsample the data) repeatedly by a factor of β, obtaining a new data matrix of size $b \times S$, where $b \ll B$ (line 1). Then we solve the joint problem for S peaks via GridSearch (line 2), a sub-routine that checks all $O(b^2)$ possible peak start and end positions. Then we zoom in by a factor of β (line 4) and refine the peak positions in $O(\beta^2)$ time via SearchNearPeak (line 5). After having zoomed back in to the BinSize=1 level we return the final Peak positions, and the best Loss values that were found.

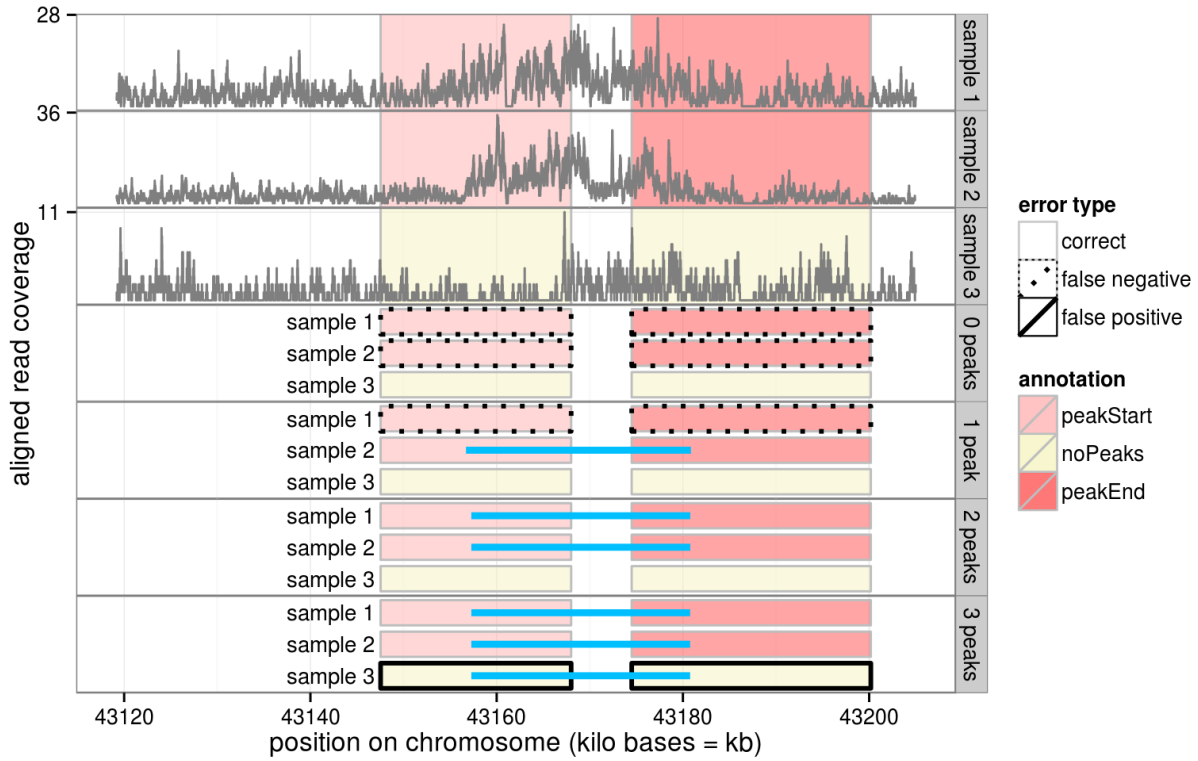

Fig. 2. A labeled data set with $S = 3$ samples (top 3 panels) and the proposed joint models for $p \in \{0, 1, 2, 3\}$ peaks (bottom 4 panels).

Algorithm 1 JointZoom, available at `https://github.com/tdhock/PeakSegJoint`

Require: count data $\mathbf{Z} \in \mathbb{Z}_+^{B \times S}$, zoom factor $\beta \in \{2, 3, \dots\}$.

BinSize $\leftarrow$ MaxBinSize(B, β).
Peak, Loss $\leftarrow$ GridSearch$(\mathbf{Z}, \text{BinSize})$.
while $1 <$ BinSize **do**
  BinSize $\leftarrow$ BinSize$/\beta$.
  Peak, Loss $\leftarrow$ SearchNearPeak$(\mathbf{Z}, \text{BinSize}, \text{Peak}, \text{Loss})$
end while
return Peak $\in \mathbb{Z}_+^2$.

For example if we fix the zoom factor at $\beta = 2$, a demonstration of the algorithm on a small data set with $B = 24$ points is shown in Figure 3. MaxBinSize returns 4, so GridSearch considers 15 models of $b = 7$ data points at bin size 4, and then SearchNearPeak considers 16 models each at bin sizes 2 and 1. In the real data set of Figure 2, there are $B = 85846$ data points, MaxBinSize returns 16384, GridSearch considers 10 models of $b = 6$ data points, and then SearchNearPeak considers 16 models each at bin sizes 8192, 4096, ..., 4, 2, 1.

Overall JointZoom with zoom factor β searches a total of $O(\beta^2 \log B)$ models, and com-

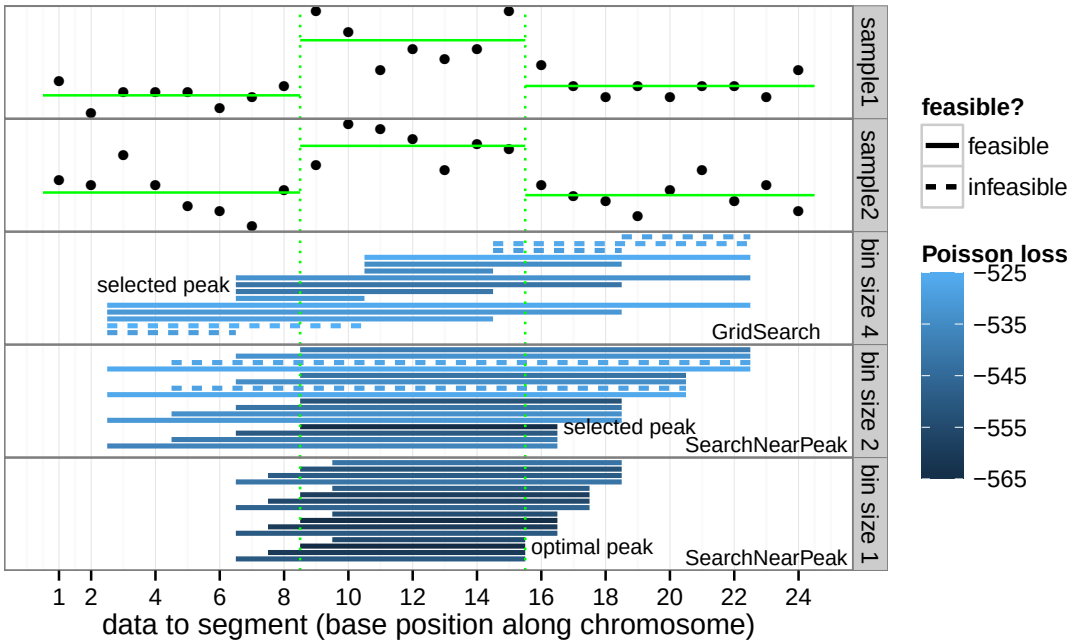

Fig. 3. Demonstration of the JointZoom segmentation algorithm. For a data set with $B = 24$ data points and $S = 2$ samples (top 2 panels), the algorithm with zoom factor of $\beta = 2$ proceeds as follows. First, the Poisson loss and feasibility is computed via GridSearch over all peak starts and ends at bin size 4. The feasible model with minimum Poisson loss is selected, the bin size is decreased to 2, and SearchNearPeak considers a new set of models around the selected peak starts and ends. We continue and return the optimal model at bin size 1 (shown in green). Interactive figure at `http://bit.ly/1AA6TgK`

puting the likelihood and feasibility for each model is an $O(SB)$ operation. JointZoom therefore returns the estimated common Peak position in worst case $O(\beta^2 SB \log B)$ time. We then compute loss values for each sample with/without that Peak in $O(SB)$ time, and sort the sample-specific loss decrease values in $O(S \log S)$ time to obtain an approximate solution path $\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_S$.

In experiments on real biological data we have observed that JointZoom can return peaks that are clearly sub-optimal if the bin factor parameter is fixed, e.g. $\beta = 2$. In practice we therefore run the algorithm with several values, by default $\beta \in \{2, \ldots, 7\}$, and for each model size we only keep the model with minimum loss.

4.2. *Overall pipeline*

We propose a practical implementation of the models and algorithms we describe in the PeakSegPipeline R package, which is available on `http://github.com/tdhock/PeakSegPipeline`. It provides a multi-step joint peak calling pipeline that can be easily run using a computer cluster to analyze epigenomic data sets. Inputs are a set of bigWig coverage data files, optionally with corresponding label files for supervised parameter learning. We suggest creating labels via visual inspection of bigWig coverage data in a genome browser, as previously described.[4]

The pipeline first runs the single-sample model on each bigWig file/contig independently, and then runs the joint model on each genomic region where the single-sample model detected at least one peak (Figure 4). In detail, the pipeline is divided into six steps:

- In parallel on each labeled data subset, determine which penalty values result in minimal label errors.
- Combine the results of the previous step, and use a supervised learning algorithm to compute a penalty function that can predict the number of peaks in each sample and genomic region.
- In parallel on each sample and genomic region, use the previously learned penalty function to predict the single-sample peaks.
- Combine the results of the previous step in order to determine which genomic regions have at least one overlapping peak.
- In each region with at least one overlapping peak, run the multi-sample joint peak detection algorithm, including supervised learning for predicting the optimal penalty / number of samples with a common peak.
- Finally combine the results of joint peak detection in each genomic region into a genome-wide joint peak prediction matrix (samples x genomic regions).

Overall the pipeline differs substantially from previous work in two respects. First, the pipeline assumes there are labels that can be used with supervised learning algorithms to choose optimal model parameters; there are no p-value thresholds or bin size parameters that need to be manually chosen by the end-user. Second, the final peak predictions are the result of the joint peak detection algorithm, which is easily interpretable in terms of similarities and differences between samples. It is thus trivial to make queries such as, "in which genomic regions do all samples have a peak except for Monocytes?"

5. Results

5.1. *Accuracy of PeakSegPipeline in labeled data sets*

We created labeled data sets in order to train and test the accuracy of our peak detection pipeline. The labels were created using visual inspection, as previously described.[4] Briefly, we created labels only in areas with clear presence/absence of significant peaks, relative to nearby genomic regions, and other samples. It is important to label a set of peak signatures that is representative of all samples and genomic regions for which predictions are desired; previous work has shown that only a few dozen labeled regions is necessary for optimal peak prediction accuracy.[4]

In previous work we have estimated peak prediction accuracy using computational cross-validation experiments in H3K36me3 and H3K4me3 data.[6] These previous experiments show that our previous constrained changepoint model provides accurate peak predictions with train/test splits over genomic regions on the same samples (e.g. train on some regions in H3K36me3 immune cell samples, predict/test on other regions on same samples). In this paper we would like to demonstrate the prediction accuracy on un-labeled test samples, possibly of other cell types. We therefore divided the data sets into two cross-validation folds: immune cell types (B cells, T cells, Monocytes) and other cell types (Skeletal muscle, Kidney, etc). We trained PeakSegPipeline on labels from one of the folds, then quantified the peak prediction accuracy using the labels in the other fold as a test set. We also ran the following baseline peak prediction algorithms, with default settings:

MACS2 is an unsupervised single-sample algorithm for sharp histone marks.[17]
DFilter is another unsupervised single-sample algorithm for broad histone marks.[10]
JAMM is a multi-sample algorithm which assumes all samples are replicates.

We observed that PeakSegPipeline is always as accurate as the baseline algorithms, and is often more accurate (Figure 5). Overall this suggests that PeakSegPipeline provides state-of-the-art peak detection accuracy in both sharp H3K4me3 and broad H3K36me3 data.

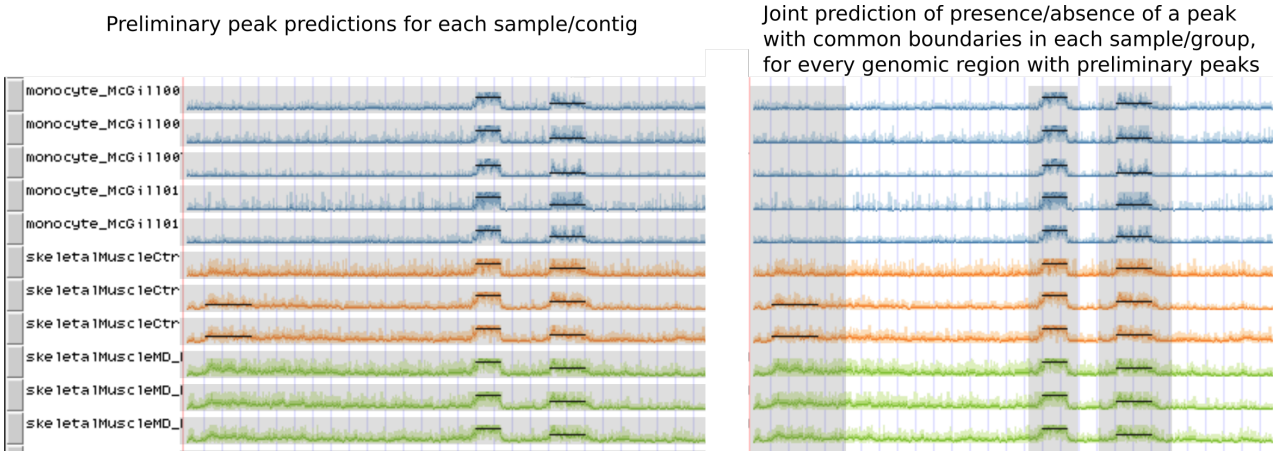

Fig. 4. PeakSegPipeline first predicts preliminary peaks for each sample/contig (left), then predicts presence/absence of a peak in each sample/group, for each genomic region with a preliminary peak (right). Colors are different cell types and peak predictions are in black.

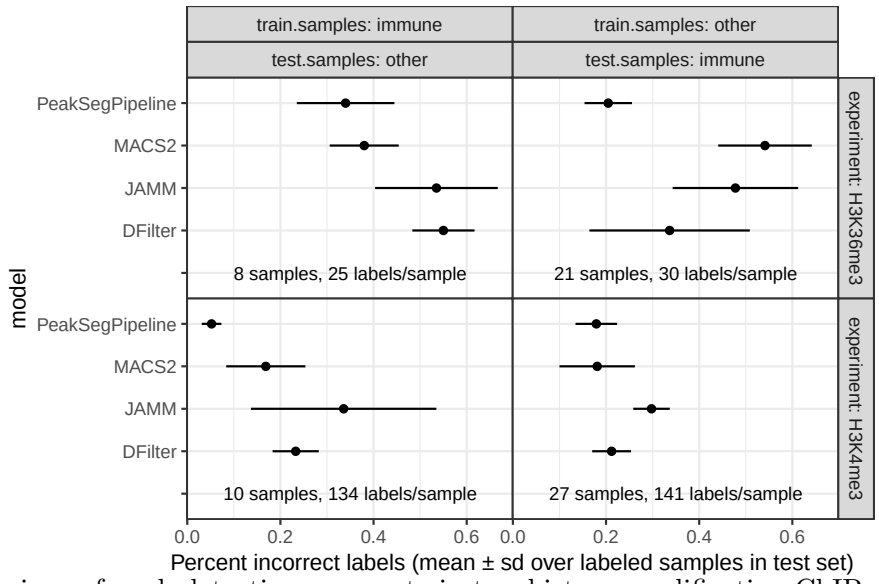

Fig. 5. Comparison of peak detection error rate in two histone modification ChIP-seq experiments (panels from top to bottom) and two sample groups (panels from left to right). MACS2, JAMM, and DFilter were run with parameters indicated by the authors for either sharp H3K4me3 or broad H3K36me3 data; parameters of PeakSegPipeline were learned using the train samples, and error rates were computed using the test samples. It is clear that the error rate of PeakSegPipeline is competitive with the other tools, and sometimes significantly lower.

5.2. *Predicted peak sizes vary with experiment type*

We also used predicted peak sizes to compare the outputs of PeakSegPipeline with the baseline MACS2 algorithm. We observed that PeakSegPipeline peak sizes depend on the experiment type (as expected), but MACS2 peak sizes depend on the parameter settings (Figure 6). This observation can be explained because MACS2 reports many small false positive peaks using default parameter settings; in contrast, PeakSegPipeline uses labels to learn experiment-specific model parameters that result in much fewer false positive peaks.

We furthermore used PeakSegPipeline to characterize the peak sizes of eight different epigenomic experiments. In each experiment we used a labeled subset of samples and genomic regions to learn a peak model. We did not manually specify any experiment-specific bin size or expected peak size parameters; rather, the peak size distribution was learned from the data and the labels. We observed that there are three distinct classes of peak size (Figure 7). The largest peaks of on average 30–50 kilobases occured in H3K36m3 data, which are known for broad peak patterns. Several histone marks (H3K9me3, H3K27me3, H3K4me1, H3K27ac, and H3K4me3) had intermediate peak sizes (1–5 kb on average). ATAC-seq and CTCF transcription factor ChIP-seq had the smallest peaks, 200–400 bases on average.

6. Discussion and Conclusions

The proposed joint model is explicitly designed for supervised, multi-sample peak detection problems such as the data sets that we considered. We observed that our proposed algorithm provides more accurate peak detection than several baselines, including the multi-sample

JAMM algorithm. Whereas JAMM was designed for replicate samples of a single cell type, PeakSegPipeline can handle any number of samples and cell types. This was advantageous in data sets such as Figure 1, which contains 3 cell types: tcell, bcell, and monocyte. The proposed joint model can be run once on all cell types, and the resulting model can be easily interpreted to find similarities and differences across samples and cell types.

Our current model is limited to separately predicting peaks in each experiment type (e.g. H3K36me3 peaks predicted separately from H3K4me3). For future work we would be interested in adapting our supervised peak learning methods for jointly analyzing several experiment types simultaneously. Our current benchmark data sets include labels that are all the same for each sample in a cell type (e.g. all T cell samples have a peak in a certain genomic region), so for future work we will be interested to create new labels in heterogeneous samples which may contain peak differences within a cell type.

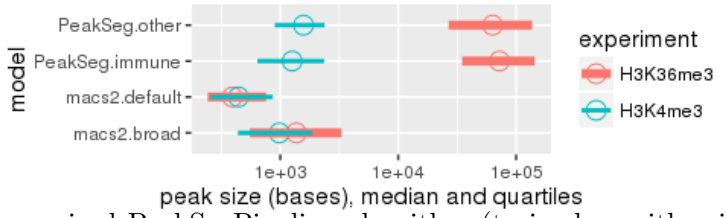

Fig. 6. Using the supervised PeakSegPipeline algorithm (trained on either immune or other samples), H3K4me3 peaks are smaller than H3K36me3 peaks, as expected. In contrast, there is little difference between H3K4me3 and H3K36me3 peak size using the unsupervised MACS2 algorithm with default parameters. MACS2 with broad parameters predicts larger peaks for both H3K4me3 and H3K36me3 data.

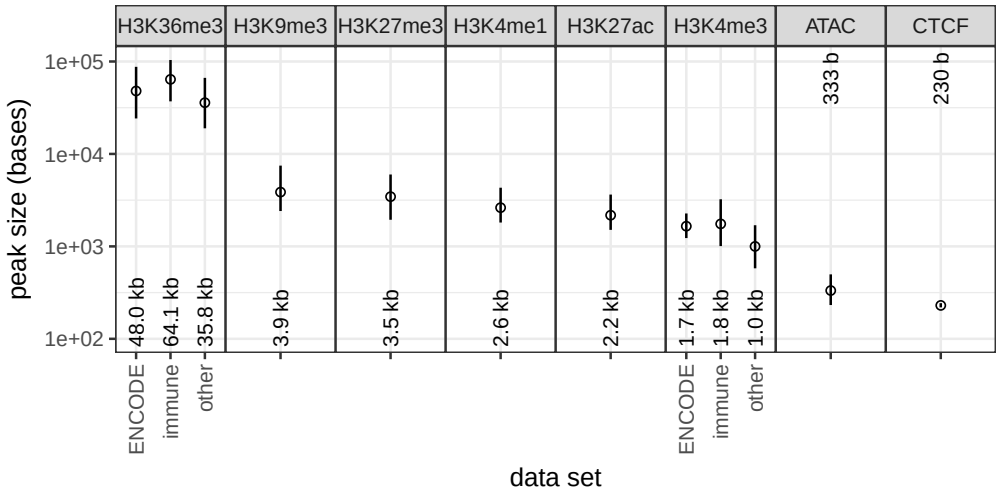

Fig. 7. Peak size varies with epigenome experiment type. PeakSegPipeline was trained on each data set, and used to predict peaks for each data set. It is clear that there are several distinct categories of peak sizes: large 30-50 kb (H3K36me3), medium 1–5 kb (H3K9me3, H3K27me3, H3K4me1, H3K27ac, H3K4me3), and small 200–400 b (ATAC, CTCF).

References

1. Timothy Bailey, Pawel Krajewski, Istvan Ladunga, Celine Lefebvre, Qunhua Li, Tao Liu, Pedro Madrigal, Cenny Taslim, and Jie Zhang. Practical guidelines for the comprehensive analysis of ChIP-seq data. *PLoS computational biology*, 9(11):e1003326, 2013.
2. Jason Ernst and Manolis Kellis. ChromHMM: automating chromatin-state discovery and characterization. *Nature Methods*, 9(3):215–216, 2012.
3. TD Hocking, V Boeva, G Rigaill, G Schleiermacher, I Janoueix-Lerosey, O Delattre, W Richer, F Bourdeaut, M Suguro, M Seto, F Bach, and JP Vert. Seganndb: interactive web-based genomic segmentation. *Bioinformatics*, 30(11):1539–46, 2014.
4. TD Hocking, Patricia Goerner-Potvin, Andreanne Morin, Xiaojian Shao, Tomi Pastinen, and Guillaume Bourque. Optimizing chip-seq peak detectors using visual labels and supervised machine learning. *Bioinformatics*, 2016.
5. TD Hocking, G Rigaill, P Fearnhead, and G Bourque. Generalized Functional Pruning Optimal Partitioning (GFPOP) for Constrained Changepoint Detection in Genomic Data. arXiv:1810.00117, 2018.
6. Toby Dylan Hocking, Guillem Rigaill, and Guillaume Bourque. PeakSeg: constrained optimal segmentation and supervised penalty learning for peak detection in count data. In *Proc. 32nd ICML*, 2015.
7. Toby Dylan Hocking, Guillem Rigaill, Paul Fearnhead, and Guillaume Bourque. A log-linear time algorithm for constrained changepoint detection. *arXiv preprint arXiv:1703.03352*, 2017.
8. Michael M Hoffman, Orion J Buske, Jie Wang, Zhiping Weng, Jeff A Bilmes, and William Stafford Noble. Unsupervised pattern discovery in human chromatin structure through genomic segmentation. *Nature Methods*, 9:473–476, 2012.
9. Mahmoud M Ibrahim, Scott A Lacadie, and Uwe Ohler. JAMM: a peak finder for joint analysis of NGS replicates. *Bioinformatics*, page btu568, 2014.
10. Vibhor Kumar, Masafumi Muratani, Nirmala Arul Rayan, Petra Kraus, Thomas Lufkin, Huck Hui Ng, and Shyam Prabhakar. Uniform, optimal signal processing of mapped deep-sequencing data. *Nature biotechnology*, 31(7):615–622, 2013.
11. Guillem Rigaill, Toby Hocking, Jean-Philippe Vert, and Francis Bach. Learning sparse penalties for change-point detection using max margin interval regression. In *Proc. 30th ICML*, pages 172–180, 2013.
12. Morten Beck Rye, Pål Sætrom, and Finn Drabløs. A manually curated ChIP-seq benchmark demonstrates room for improvement in current peak-finder programs. *Nucleic acids research*, 2010.
13. Adam M. Szalkowski and Christoph D. Schmid. Rapid innovation in ChIP-seq peak-calling algorithms is outdistancing benchmarking efforts. *Briefings in Bioinformatics*, 12(6):626–633, 2011.
14. EG Wilibanks and MT Facciotti. Evaluation of Algorithm Performance in ChIP-Seq Peak Detection. *PLoS ONE*, 5(7), 2010.
15. Xin Zeng, Rajendran Sanalkumar, Emery Bresnick, Hongda Li, Qiang Chang, and Sunduz Keles. jMOSAiCS: joint analysis of multiple ChIP-seq datasets. *Genome Biology*, 14(4):R38, 2013.
16. Yanxiao Zhang, Yu-Hsuan Lin, Timothy D. Johnson, Laura S. Rozek, and Maureen A. Sartor. PePr: a peak-calling prioritization pipeline to identify consistent or differential peaks from replicated ChIP-Seq data. *Bioinformatics*, 30(18):2568–2575, 2014.
17. Yong Zhang, Tao Liu, Clifford A Meyer, Jérôme Eeckhoute, David S Johnson, Bradley E Bernstein, Chad Nusbaum, Richard M Myers, Myles Brown, Wei Li, et al. Model-based analysis of ChIP-Seq (MACS). *Genome Biol*, 9(9):R137, 2008.