Bioinformatics, 34, 2018, i891–i900
doi: 10.1093/bioinformatics/bty612
ECCB 2018

DeepDiff: DEEP-learning for predicting DIFFerential gene expression from histone modifications

Arshdeep Sekhon, Ritambhara Singh and Yanjun Qi*

Department of Computer Science, University of Virginia, Charlottesville, VA, USA

*To whom correspondence should be addressed.

Abstract

Motivation: Computational methods that predict differential gene expression from histone modification signals are highly desirable for understanding how histone modifications control the functional heterogeneity of cells through influencing differential gene regulation. Recent studies either failed to capture combinatorial effects on differential prediction or primarily only focused on cell type-specific analysis. In this paper we develop a novel attention-based deep learning architecture, DeepDiff, that provides a unified and end-to-end solution to model and to interpret how dependencies among histone modifications control the differential patterns of gene regulation. DeepDiff uses a hierarchy of multiple Long Short-Term Memory (LSTM) modules to encode the spatial structure of input signals and to model how various histone modifications cooperate automatically. We introduce and train two levels of attention jointly with the target prediction, enabling DeepDiff to attend differentially to relevant modifications and to locate important genome positions for each modification. Additionally, DeepDiff introduces a novel deep-learning based multi-task formulation to use the cell-type-specific gene expression predictions as auxiliary tasks, encouraging richer feature embeddings in our primary task of differential expression prediction.
Results: Using data from Roadmap Epigenomics Project (REMC) for ten different pairs of cell types, we show that DeepDiff significantly outperforms the state-of-the-art baselines for differential gene expression prediction. The learned attention weights are validated by observations from previous studies about how epigenetic mechanisms connect to differential gene expression.
Availability and implementation: Codes and results are available at deepchrome.org.
Contact: yanjun@virginia.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Gene regulation is the process of controlling gene expression. The human body contains hundreds of different cell types. Although these cells include the same set of DNA information, their functions are different. Cells resort to a host of mechanisms to regulate genes differently. Many factors, especially those in the epigenome, can affect how cells express genes differently. As reviewed in Egger *et al.* (2004) and Meisner and Reif (2015), epigenomics studies how gene expression is altered by a set of chemical reactions over the chromatin that do not alter the DNA sequence.

Histone modification (HM) is one set of critical chemical reactions over the chromatin that plays a crucial role in regulating gene transcription. DNA strings are wrapped around 'bead'-like structures called nucleosomes that are composed of histone proteins. These histone proteins are prone to a variety of modifications (e.g. methylation, acetylation, phosphorylation, etc.) that can modify the spatial orientation of the DNA structure. Such modifications impact the binding behavior of transcription factor proteins (to DNA) and thus generate different forms of gene regulation. The significant role of histone modifications in influencing gene regulation was evidenced in studies like connecting anomalous histone modification profiles to cancer occurrences (Bannister and Kouzarides, 2011). Contrary to DNA mutations, such epigenetic changes are potentially reversible (Bannister and Kouzarides, 2011). This vital feature has brought histone modifications to the center stage of epigenetic therapy.

Recent advances in next-generation sequencing have allowed researchers to measure gene expression and genome-wide histone modification patterns as read counts across many cell types. These datasets have been made available through large-scale repositories, one latest being the Roadmap Epigenome Project (REMC, publicly available) (Kundaje *et al.*, 2015). REMC has released thousands of genome-wide datasets including gene expression reads (RNA-Seq datasets), and HM reads across 100 different human cells/tissues (Kundaje *et al.*, 2015). Multiple recent papers tried to understand gene regulation by predicting gene expression from large-scale HM signals. Related studies (summarized in Supplementary Table S2) have mostly focused on the formulation under a single cell condition, even though gene regulation undergoes differential changes by environmental triggers, from one tissue type to another, or under different cell development stages.

Differential gene expression, or difference in expression levels of the same gene in two cell conditions, controls functional and structural heterogeneity of cells and has also been implicated in a number of diseases, providing valuable tools for the discovery of therapeutic targets and diagnostic markers. Differential gene expression has been linked to aberrant HM profiles in the literature. For example, Gjoneska *et al.* (2015) showed the correlation between differential gene expression in different stages of Alzheimer's disease-like neurodegeneration in mice. Further, the authors observed that the changes in HM patterns associate with the differentially regulated genes. Koch *et al.* (2007) reported coordinated changes between HM profiles and differential gene expression across the lymphoblastoid cell line GM06990, K562 and HeLa-S3 cell lines. As another example, Weng *et al.* (2012) showed links between differential expression of naive T cells versus memory T cells, ascribed mainly to changes in histone modifications.

This paper proposes an attention-based deep learning architecture to learn from datasets like REMC, how different histone modifications work together to influence genes' differential expression pattern between two different cell-types. We argue that such differential analysis and differential understanding of gene regulation from HMs can enable new insights into principles of life and diseases, will allow for the interrogation of previously unexplored regulation spaces, and will become an important mode of epigenomics analysis in the future. Four fundamental data challenges exist when modeling such tasks through machine-learning:

1. Genome-wide HM signals are spatially structured and may have long-range dependency. For instance, to quantify the influence of a histone modification mark, learning methods typically need to use as input features all of the signals covering a DNA region of length 10 000 base pair (bp) centered at the transcription start site (TSS) of each gene. These signals are sequentially ordered along the genome direction. To develop 'epigenetic' drugs, it is important to recognize how an HM mark's influence varies over different genomic locations.
2. The core aim is to understand what the relevant HM factors are and how they work together to control differential expression. Various types of HM marks exist in human chromatin that can influence gene regulation. For example, each of the five standard histone proteins can be simultaneously modified with various kinds of chemical modifications, resulting in a large number of varying histone modification marks. As shown in Figure 1, we build a feature vector representing signals of each HM mark surrounding a gene's TSS position. When modeling genome-wide signal reads from multiple marks, learning algorithms should take into account the modular nature of such feature inputs, where each mark functions as a module. We want to understand how the interactions among these modules influence the prediction (differential gene expression).
3. Since the fundamental goal of such analysis is to understand how HMs affect gene regulation, it requires the modeling techniques to provide a degree of interpretability and allowing for automatically discovering what features are essential for predictions.
4. There exist a small number of genes exhibiting a significant change of gene expression (differential patterns) across two human cell types like A and B. This makes the prediction task using differential gene expression as outputs much harder than predicting gene expression directly in a single condition like A alone or B alone.

In this paper, we propose an attention-based deep learning model, DeepDiff, that learns to predict the log-fold change of a gene's expression across two different cell conditions (assuming cell type A and cell type B in the rest of the paper). Here, $\boldsymbol{X}^A \in \mathbb{R}^{M\times T}$ and $\boldsymbol{X}^B \in \mathbb{R}^{M\times T}$, where M represents the number of HM signals and T represents the number of bins. For each gene, its input signals consist of $\boldsymbol{X}^A$ (the histone modification signals from A), $\boldsymbol{X}^B$ (the histone modification signals from B) and $(\boldsymbol{X}^A - \boldsymbol{X}^B)$ (the difference matrix between HM signals of these two conditions). All three cover the gene's neighboring 10 000 base pair regions centered at the transcription start site (TSS). (i) To tackle the first challenge of modeling spatially structured input signals, we use the Long Short-term Memory (LSTM) (Section 3.4) deep-learning module that can represent interactions among signals at the different positions of a chromatin mark. Because we model multiple HM marks, resulting in multiple LSTMs learning to embed various HM marks. (ii) To handle the second challenge of modeling how HM marks work together, we use a second-level LSTM to learn complex dependencies among different marks. (iii) For the third challenge of interpretability, we borrow ideas from the AttentiveChrome (Singh *et al.*, 2017) that focuses on cell-specific predictions. We train two levels of 'soft' attention weights, to attend to the most relevant regions of a chromatin mark, and to recognize and attend to the critical marks for each differential expression prediction. Through predicting and attending in one unified architecture, DeepDiff allows users to understand how chromatin marks control differential gene regulation between two cell types. (iv) For the last challenge of difficult label situation for differential expression prediction, we design a novel multi-task framework to use the cell-type-specific prediction network as auxiliary tasks to regularize our primary task of differential expression prediction. The cell-type specific system (one for cell A and another one for cell B) also uses attention plus the hierarchical LSTMs formulation. Further, we introduce an additional auxiliary loss term that encourages the learned embeddings of HM inputs $\boldsymbol{X}^A$ and $\boldsymbol{X}^B$ to be far apart for differentially expressed genes.

In summary, DeepDiff provides the following technical contributions:

- DeepDiff uses a hierarchy of two levels of gene-specific attention mechanisms to identify salient features at both the bin level and the HM levels. It can model highly modular inputs where each module is highly structured. Attention weights enable our model to explain its decisions naturally by providing 'what' and 'where' in HM signal inputs is important for the differential gene expression output. This flexibility and interpretability make this model an ideal technique for handling large-scale epigenomic data analysis.
- We introduce an auxiliary task and auxiliary loss formulation to aid the main task of differential gene expression prediction. The proposed multitasking framework couples the two related tasks of cell-type-specific predictions and the main task of differential expression prediction. This auxiliary formulation provides the model with additional information from the auxiliary evidence, encouraging richer feature embeddings compared to

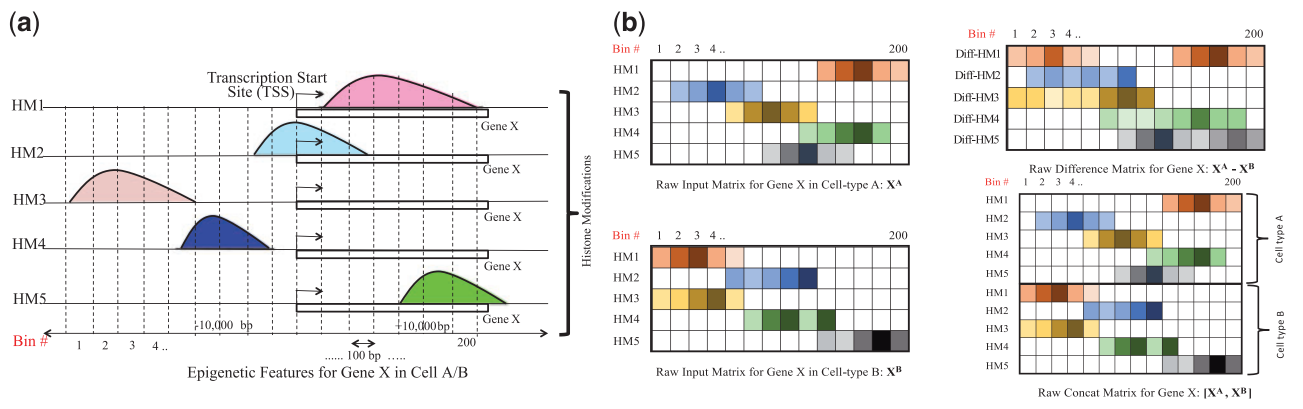
(a)
Transcription Start Site (TSS)
HM1
HM2
HM3
HM4
HM5
Gene X
Histone Modifications
Bin #
1 2 3 4 ..
-10,000 bp
+10,000 bp
200
...... 100 bp .....
Epigenetic Features for Gene X in Cell A/B
(b)
Raw Input Matrix for Gene X in Cell-type A: $\mathbf{X^A}$
Raw Input Matrix for Gene X in Cell-type B: $\mathbf{X^B}$
Diff-HM1
Diff-HM2
Diff-HM3
Diff-HM4
Diff-HM5
Raw Difference Matrix for Gene X: $\mathbf{X^A - X^B}$
Cell type A
Cell type B
Raw Concat Matrix for Gene X: $[\mathbf{X^A}, \mathbf{X^B}]$

Fig. 1. (**a**) Feature input generation for a gene in a cell-type and (**b**) Raw input feature variations to DeepDiff model for differential expression: difference and concatenated HM signals of both cell-types

only difference HM features. It helps DeepDiff to build on top of the state-of-the-art AttentiveChrome (Singh *et al.*, 2017) and can borrow auxiliary features from AttentiveChrome tasks. Further, we introduce a novel auxiliary loss term inspired by the contrastive loss (Hadsell *et al.*, 2006) of the Siamese architecture formulation. This loss term encourages the model to learn embeddings whose neighborhood structures in the model's representation space are more consistent with the differential gene expression pattern.

- To the authors' best knowledge, DeepDiff is the first deep learning based architecture for relating histone modification and differential gene expression patterns. DeepDiff provides more accurate predictions than state-of-the-art baselines. Using datasets from REMC, we evaluate DeepDiff on ten different pairs of cell types. We validate the learned attention weights using previous observations obtained from HM enrichment analysis across differentially regulated genes.

2 Previous works

Multiple computational methods have been proposed to employ HMs for predicting gene expression using large-scale histone modification datasets. Recent methods in the literature can be roughly grouped into three categories with respect to the formulation of outputs: regression, classification or ranking. (i) The regression-based models include linear regression (Costa *et al.*, 2011; Karlić *et al.*, 2010) and Support Vector Regression (SVR) (Cheng and Gerstein, 2012). Cheng and Gerstein (2012) divided the DNA regions around TSS (transcription start site) and TTS (transcription terminal site) into small bins of 100 base pairs and used a multiple bin-specific Support Vector Regression (SVR) to model HM signals for gene expression prediction. They extended this SVR model to predict differential gene expression between mouse embryonic stem cell and neural progenitor cell using the difference of the HM signals as features/inputs. This study also uses a two-layer SVR model to integrate information from multiple bins. The first layer is a bin-specific SVR model for histone modification features. A second layer takes as input the predictions from the first layer across all bin positions and predicts a single regression output for differential gene expression. (ii) Frasca and Pavesi (2013) proposed a ranking based cell type-specific model that formulates gene expression prediction as a ranking task such that high ranks correspond to high levels of gene expression and low rank corresponds to low expression. (iii) Multiple studies used classification-based formulation to model the gene expression prediction from HM inputs. This includes support vector machines (Cheng and Gerstein, 2012), random forests (Dong *et al.*, 2012; Li *et al.*, 2015a), rule-based learning (Ho *et al.*, 2015) and deep learning frameworks like DeepChrome (Singh *et al.*, 2016) and AttentiveChrome (Singh *et al.*, 2017). (Dong *et al.*, 2012) used a random forest classifier to predict genes as silent or transcribed, while the authors also used linear and multivariate regression to predict gene expression values. Li *et al.* (2015a) presented a two-step process—feature selection, then followed by prediction for differential gene expression. It used the so-called ReliefF (Kononenko *et al.*, 1997) based feature selection and Random Forest Classification. DeepChrome (Singh *et al.*, 2016) and Attentive Chrome (Singh *et al.*, 2017) are deep learning based frameworks for cell-type specific gene expression prediction. DeepChrome (Singh *et al.*, 2016) used Convolution Neural Nets (CNN). Differently, AttentiveChrome (Singh *et al.*, 2017) used a hierarchical attention-based deep learning architecture to predict gene expression from HM reads. Supplementary Table S2 compares all the aforementioned related studies for gene expression prediction.

3 Materials and methods

3.1 Background: Recurrent Neural Networks and Long Short-Term Memory (LSTM)

Recurrent neural networks (RNNs) have achieved remarkable success in sequential modeling applications like translation, image captioning, video segmentation, etc. A sequential input of RNN is normally represented by an input matrix $\mathbf{X}$ of size $n_{in} \times T$, where T represents the time steps and n_{in} represents the dimension of the features describing each time-step of the input. For an input X, an RNN produces a matrix $\mathbf{H}$ of size $D \times T$ as output, where D is the RNN embedding size. More concretely, at each timestep $t \in \{1, \ldots, T\}$, an RNN takes an input column vector $\mathbf{x}_t \in \mathbb{R}^{n_{in}}$ and the previous hidden state vector $\mathbf{h_{t-1}} \in \mathbb{R}^d$ to produce the next hidden state $\mathbf{h_t}$ by applying the following recursive operation:

$$\mathbf{h}_t = \sigma(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}) = \overrightarrow{LSTM}(\mathbf{x}_t), \tag{1}$$

where $\mathbf{W}, \mathbf{U}, \mathbf{b}$ are the trainable parameters of the model, and σ is an element-wise nonlinearity function. Due to the recursive nature, in theory, RNNs can capture the complete set of dependencies among all time steps without having to learn different parameters for each time step, like all spatial positions in a sequential sample.

A variant of the RNN, LSTM (Hochreiter and Schmidhuber, 1997), further improves upon the basic RNN [Eq. (1)] to model long-term dependencies. In addition to the hidden state-to-state recurrent component in an RNN, an LSTM layer has a recurrent cell state updating function and gating functions. The gating functions control what information needs to be added or removed from the cell state. This combination of cell state and gating functions allows the LSTM to learn long-term dependencies while avoiding vanishing and exploding gradients. Similar to a basic RNN, when given input vector $\mathbf{x}_t$ and the state $\mathbf{h}_{t-1}$ from previous time step $t-1$, an LSTM module also produces a new state vector $\mathbf{h}_t$. For our task, we call each bin position on the genome coordinate a 'time step'.

3.2 Attention-based deep-learning models

Deep neural networks augmented with attention mechanisms have obtained great success on multiple artificial intelligence topics such as machine translation (Dzmitry *et al.*, 2014), object recognition (Jimmy *et al.*, 2014; Volodymyr *et al.*, 2014), image caption generation (Xu *et al.*, 2015), question answering (Ilya *et al.*, 2014), text document classification (Zichao *et al.*, 2016), video description generation (Li *et al.*, 2015b), visual question answering (Huijuan and Kate, 2016), or solving discrete optimization (Oriol *et al.*, 2015). The idea of attention in deep learning is inspired by the properties of the human visual system. When perceiving a scene, the human vision fixates more on some areas over others, depending on the task at hand (Corbetta and Shulman, 2002). Augmenting deep learning models with attention allows them to focus selectively on only relevant features for a prediction. Different attention mechanisms have been proposed in the literature, including 'soft' attention (Dzmitry *et al.*, 2014), 'hard attention' (Xu *et al.*, 2015; Minh-Thang *et al.*, 2015), or 'location-aware attention' (Chorowski *et al.*, 2015). Soft attention (Dzmitry *et al.*, 2014) calculates a 'soft' weighting scheme over all the components of an input. These weights indicate the relative importance of each feature component for a given prediction. The weights are then used to compute a summary representation of the input as a weighted combination of the components. The magnitude of an attention weight correlates highly with the degree of significance of the corresponding component to the prediction. This property is particularly ideal for adapting deep learning to biology tasks, as it gives users interpretable information regarding how features contribute to a prediction. Recently, AttentiveChrome (Singh *et al.*, 2017) introduced two levels of attention at the bin and the histone-modification level, enabling the user to get information about which features were responsible for each prediction at the sample level.

3.3 Input generation

We focus on the predictive modeling of differential gene expression given the histone modification profiles of a gene in two cell-types. We formulate the output as the log fold change in expression given the histone modification profiles for the two cell-types under consideration. Similar to DeepChrome (Singh *et al.*, 2016) and AttentiveChrome (Singh *et al.*, 2017), we divided the 20 000 base-pair (bp) DNA region ($\pm$10 000 bp) around the transcription start site (TSS) of each gene into bins of length 100 bp. Each bin includes 100 bp long adjacent positions flanking the TSS of a gene.We consider five core histone modification marks that have been uniformly profiled across multiple cell types in the REMC database (Kundaje *et al.*, 2015). Supplementary Table S4 summarizes the 5 HMs we use and their associated functional regions on the genome. Figure 1(a) summarizes our input matrix generation strategy. More concretely, our input for each gene includes two 5×200 matrices, each matrix corresponding to each of the two cell types under consideration [depicted in Fig. 1(b)]. Columns and rows in each matrix represent bins and histone modifications, respectively. Thanks to the capability of neural networks for learning meaningful representations, we do not perform any feature selection before feeding the matrices in Figure 1(b) to the proposed DeepDiff model variations, a hierarchical attention-based DNN. Compared to previous studies (Supplementary Table S2), DeepDiff does not need to explore the best-bin, averaged or other feature selection strategies. As an end-to-end strategy (raw features to predictions), DeepDiff eliminates the need to evaluate different feature selection strategies.

3.4 DeepDiff: learning meaningful representations through two levels of embedding and attention modules

Notations: Our training set consists of N_{samp} gene samples in the form of $(\mathbf{X}_{(n)}, y_{(n)})$ pairs, where $n \in \{1., .., N_{samp}\}$. Given two cell-types A and B, and a gene g under consideration, the HM profile of gene g in A and B is denoted as $\boldsymbol{X}^A$ and $\boldsymbol{X}^B$, respectively. We consider $M=5$ HM marks for each gene. Each HM signal across the $T=200$ bins in cell-type A is represented by a row vector in $\boldsymbol{X}^A$. Similarly, for cell-type B, each HM signal is represented by a row vector in $\boldsymbol{X}^B$.

We do not perform any feature selection on the raw histone modification features. Instead, we use deep learning modules to learn sensible features. Our raw features have two important properties (as depicted in Fig. 1): (i) In addition to the raw HM signals from the two cell types under consideration, we use difference and concatenation of the raw HM signals. This results in modular raw input features with four possible input matrices, rows corresponding to HM vectors [as shown in Fig. 1(b)]. (ii) These HM vectors are spatially structured along the genome coordinate. Considering the spatially structured raw features and their modular property, we use two levels of basic embedding modules: Level I and Level II embedding units coupled with two levels of attention modules. These basic modules used in DeepDiff variations are illustrated in Figure 2. Now, we explain how we use deep learning to learn the representation of each matrix in Figure 1(b).

Level I embedding (f1): The Level I Embedding module consists of a bin-level LSTM for learning the embedding of every HM, followed by a bin level attention mechanism. The bin level LSTM sequentially models the signal at each bin position. We name LSTMs, for all input HMs, put together, as the Level I Embedding module. The Level I Embedding module f_1 consists of multiple bidirectional LSTMs, one for each input HM. The $LSTM_j$ corresponding to HM j, takes as input the jth HM, i.e, jth row vector in matrix $\boldsymbol{X}$, $\boldsymbol{X}_j = [\boldsymbol{x}_{j1}, \ldots, \boldsymbol{x}_{jt}, \ldots, \boldsymbol{x}_{jT}]$. A bidirectional LSTM has one LSTM in each direction. The forward LSTM models dependencies in $\boldsymbol{x}_j$ in the direction 1 to T, i.e. $\overrightarrow{\boldsymbol{h}}_{jt} = \overrightarrow{LSTM}_j(\boldsymbol{x}_{jt})$ where $\overrightarrow{\boldsymbol{h}}_{jt}$ is of hidden state size D. The backward LSTM models the dependencies from T to 1: that is $[\boldsymbol{x}_{jT}, \ldots, \boldsymbol{x}_{jt}, \ldots, \boldsymbol{x}_{j1}]$. Hence, $\overleftarrow{\boldsymbol{h}}_{jt} = \overleftarrow{LSTM}_j(\boldsymbol{x}_{jt})$. The output of the bidirectional LSTM is a concatenation of the hidden state output of the forward and backward LSTMs at each t position: $\boldsymbol{h}_{jt} = [\overrightarrow{\boldsymbol{h}}_{jt}, \overleftarrow{\boldsymbol{h}}_{jt}]$, where $\boldsymbol{h}_{jt}$ is of size $2 \times D$.

Bin level attention: Attention in Deep Learning is a powerful tool used to highlight features that are important for a given prediction. The hidden state at each step of the LSTM produces an embedding for that bin position $\boldsymbol{h}_{jt}$. To get a cumulative embedding to represent all the bin positions, one strategy could be to sum the embeddings across all bin positions. However, not all the bin

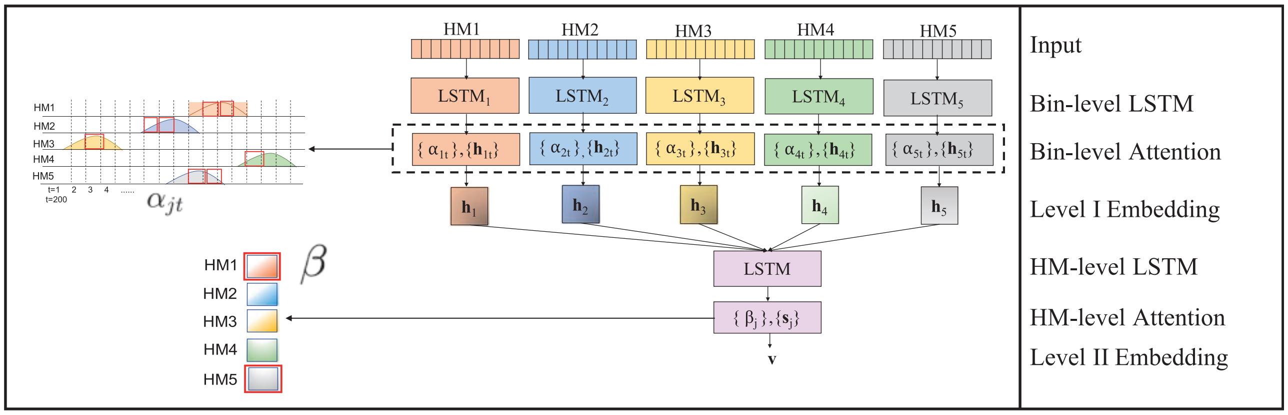

Fig. 2. Two level attention mechanism used in DeepDiff variations for meaningful feature representation: α_{jt} represents the bin level attention for HM *j* and bin *t* obtained from the Level I Embedding module attention mechanism, indicating the relative importance for bin *t* in HM *j*. β_j represents the HM-level attention for HM *j* obtained from the Level II Embedding module's attention mechanism, representing the relative importance of HM *j*

positions are equally relevant. For example, in certain HM patterns, bin positions near the transcription start site are more important than the ones away from it. To learn and encode such important information into the embeddings, we use a soft attention mechanism that automatically discovers which are the important bin positions as part of the training process. The attention-augmented LSTMs result in learning a weighting representation for the bin embeddings, such that more important bin positions get a higher weight. In detail, this is done using a context weight vector, denoted by $\boldsymbol{Wb}_j$ of dimension $2 \times D$ for each HM *j*. An attention weight α_{jt}, corresponding to bin position *t* for the *jth* HM is obtained by

$$\alpha_{jt} = \frac{exp(\boldsymbol{h}_{jt} \cdot \boldsymbol{Wb}_j)}{\Sigma_{k=1}^{T}(exp(\boldsymbol{h}_{jk} \cdot \boldsymbol{Wb}_k))} \tag{2}$$

where $\boldsymbol{Wb}_j$ is learned through training and $\cdot$ indicates dot product. These attention weights are then used to weigh the embedding vectors of all bins to get a summary embedding: $\boldsymbol{h}_j = \Sigma_{t=1}^{T}(\alpha_{jt} \times \boldsymbol{h}_{jt})$. Essentially, this 'summary' representation for each HM represents a bin importance weighted sum of all bins in the HM under consideration. The attention tells us where in this HM is important for prediction.

Level II embedding (f2): To efficiently represent the combinatorial dependencies between the various HMs, we use another LSTM as a second level embedding module. This LSTM takes as input the Level I Embedding output $\boldsymbol{h}_j$ where $j \in \{1, \dots, M\}$. In detail, the *jth* HM embedding from Level I Embedding module $\boldsymbol{h}_j$ is used as input to the *jth* time step in a bidirectional LSTM. The LSTM will generate an embedding vector for *jth* time step: $s_j = LSTM(\boldsymbol{h}_j)$.

HM level attention: To combine the outputs from all *M* HMs in an informative way, we use a second level attention mechanism to learn attention weights β_j, representing the importance of the *jth* HM. To get these HM level attention scores β_j where $\{j \in 1, \dots, M\}$, we learn an HM-level context vector $\boldsymbol{W}_h$ to calculate an attention score as

$$\beta_j = \frac{exp(s_j \cdot \boldsymbol{W}_h)}{\Sigma_{l=1}^{M}(exp(\boldsymbol{s}_l \cdot \boldsymbol{W}_h))} \tag{3}$$

This attention weight β_j intuitively represents the relative contribution of the HM $\boldsymbol{x}_j$ to the summary representation of the whole matrix $\boldsymbol{X}$. To get a summarized embedding of the HMs, we use the outputs at all time steps of the HM level LSTM weighted by its attention score as the final embedding of describing $\boldsymbol{X}$ i.e. $\boldsymbol{v} = \Sigma_{j=1}^{M}(\beta_j \times \boldsymbol{s}_j)$. Including the bin level as well as HM-level attention weights representing *X* allows us to interpret which bins in which HMs were relatively more important for the current prediction.

3.5 DeepDiff main task: an end-to-end deep-learning architecture for regression

We formulate the differential gene expression prediction (our main task) as a regression task. The target label for a gene is the log fold change of its expression in Cell-type A versus its expression in Cell-type B. In DeepDiff, the learned representation vector **v** from Level II Embedding is fed into a multi-layer perceptron (MLP) module to learn a regression function, a mapping from HM profiles to the target real value representing differential pattern. In detail, this prediction module $f_{mlp}(.)$ comprises a standard, fully connected multilayer perceptron network with multiple alternating linear and nonlinear layers. Each layer learns to map its input to a hidden feature space, and the last output layer learns the mapping from the hidden space to the output label space.

We use $f(\mathbf{X})$ to denote the whole end-to-end network in DeepDiff mapping raw HM profile to differential output. The parameters learned during training of function $f(.)$ will be denoted as Θ. Θ consists of all learnable parameters of the LSTMs as well as the context vectors in both Level I and Level II Embedding modules, and the parameters of the aforementioned f_{mlp}. When training this deep model, parameters are randomly initialized first and input samples are fed through the network. The output of this network is a prediction associated with each sample. The difference between each prediction output $f(\mathbf{X})$ and true label y is fed back into the network through a 'back-propagation' step. The parameters (Θ) are updated in order to minimize a loss function which captures the difference between true labels and predicted values. The loss function ℓ_{Diff}, on the entire training set of size N_{samp}, is defined as:

$$\ell_{Diff} = \frac{1}{N_{samp}} \sum_{n=1}^{N_{samp}} loss(f(\mathbf{X}_{(n)}), y_{(n)}) \tag{4}$$

To train our regression function for the main differential task, we select the squared error loss as the loss function which is defined per-sample as:

$$loss(f(\mathbf{X}^{(n)}), y^{(n)}) = (y^{(n)} - f(\mathbf{X}^{(n)}, \Theta))^2 \tag{5}$$

This means we use mean squared error (MSE) ℓ_{Diff} for training. The back-propagation step essentially uses stochastic gradient descent (SGD) method to train parameters (Léon, 2004). For a set of training samples, instead of calculating true gradient of the objective on all training samples, SGD calculates gradient and updates accordingly on each training sample. This means a gradient descent step is applied to update network parameters Θ as follows:

$$\Theta \leftarrow \Theta - \eta \frac{\partial loss(f(\mathbf{X}^{(n)}), y^{(n)})}{\partial \Theta} \tag{6}$$

where η is the learning rate parameter and $\frac{\partial loss(f(\mathbf{X}^{(n)}), y^{(n)})}{\partial \Theta}$ is the gradient. We use the optimizer Adaptive Moment Estimation (ADAM) (Omony, 2014) to train our models. In comparison to SGD in Eq. (7), ADAM computes adaptive learning rates η, instead of fixed η during training, using estimates of the first and second moments of the gradients. Details of ADAM optimizer are in Supplementary Section S1.1.

3.6 DeepDiff with multitasking: learning better representations with auxiliary tasks

We then extend the basic DeepDiff mentioned above into a multi-task learning formulation, in order to learn better joint representations informed by auxiliary tasks (details below). Multi-task learning (Multitasking) was initially proposed by Caruana (1997) to find common feature representations across multiple relevant tasks. Most of the multitasking studies have focused on neural networks (Caruana, 1997), where some hidden layers are shared between various tasks. If different tasks are sufficiently related, multitasking can lead to better generalization. In this paper, we consider two such related tasks as auxiliary tasks for multi-task learning with our DeepDiff main task:

Cell-Specific Auxiliary – (Auxiliary-Task-A + Auxiliary-Task-B): We posit the cell-type specific gene expression prediction in each of the two cell-types (A and B) as the Cell-Specific Auxiliary tasks to regulate the main DeepDiff in the training phase. We call these two tasks in cell-types A and B as Auxiliary-Task-A and Auxiliary-Task-B, respectively. Since our main DeepDiff task uses the log-fold change of the counts of gene expression as the target y, we use the log of counts of gene expression in cell-type A and cell-type B as the target value in Auxiliary-Task-A and Auxiliary-Task-B, respectively. To handle zero values of expression (log 0 gives NA), we add 1 to all counts. Besides, we also evaluate using binarized cell-type specific gene expression as the label for the auxiliary tasks (i.e. as classification as opposed to regression. See details in Supplementary Material). For Auxiliary-Task-A, $\mathbf{X}^A$ is passed through two levels of embedding and attention modules that are specific for cell-type A. The output embedding of the Level II Embedding unit $\boldsymbol{v}_A$ is passed through an MLP layer $f^A_{mlp}(\boldsymbol{v}_A)$. $f^A_{mlp}(\boldsymbol{v}_A)$ is used to map $\boldsymbol{v}_A$ (from $\boldsymbol{X}^A$) to the target value for the gene in Cell-type A. Similarly, for the Auxiliary-Task-B, $\boldsymbol{v}_B$, obtained by passing $\mathbf{X}^B$ through two levels of embedding modules, is passed through an MLP layer $f^B_{mlp}(\boldsymbol{v}_B)$ for Cell-type B specific gene expression prediction. By jointly training cell-type specific gene expression with differential gene expression, we can improve the main DeepDiff task performance (see section 4 for experimental results). We train both of these auxiliary tasks using the sum of the MSE loss averaged over the training set (Eq. (6)) between the gene expression target and predicted values for both cell-types. We denote this cumulative Cell-Specific Auxiliary loss for both cell-types as $\ell_{CellAux}$.

Siamese Auxiliary – with Contrastive Siamese loss: We use a second type of auxiliary task to regulate the neighborhood structure of the learned embedding space. This auxiliary task encourages the model to learn embeddings whose neighborhood structures in the model representation space are more consistent with the differential gene expression pattern. It is achieved through a contrastive loss term inspired by the Siamese architecture formulation (Hadsell *et al.*, 2006). In detail, a Siamese architecture consists of two identical networks with shared parameters which accept distinct inputs but are joined by a similarity metric at the output. This similarity metric, used in the output loss of the Siamese network, coupled with shared weights encourages 'similar' inputs to map to nearby points in the output representation space and 'dissimilar' inputs to map to distant points in the representation space. We extend this notion of similarity and dissimilarity to the differential gene expression case. We consider the histone modification profiles $\boldsymbol{X}^A$ and $\boldsymbol{X}^B$ of two differentially expressed genes (upregulated or downregulated) to be 'different' and 'similar' for genes not differentially expressed. We denote 'dissimilar' with label $S = 1$ and 'similar' with label $S = 0$. If the log change of differential gene expression ≤ -2 (downregulated) or ≥ 2 (upregulated), we label it as ($S = 1$). Otherwise, we label the training sample with $S = 0$. For this auxiliary task, we use Level I Embedding modules as the twin networks of Siamese module (see Figure 3 describing DeepDiff variations). $\mathbf{X}^A$ is passed through two levels of embedding and attention modules specific to cell-type A. Similarly, $\mathbf{X}^B$ is passed through two levels of embedding and attention modules. In the Siamese contrastive loss, we use the Level I embeddings for Cell-type A and B: f^A_1 and f^B_1. The siamese contrastive loss (Hadsell *et al.*, 2006), denoted as $\ell_{Siamese}$, uses the following Eqn. 7, at the output of the Level I Embedding units f^A_1 and f^B_1 to train this auxiliary task:

$$\ell_{Siamese} = (1 - S) \times \frac{1}{2} \times R + S \times \frac{1}{2} \max(0, m - R)^2 \tag{7}$$

Here, R represents:

$$R = ||(f^A_1(\boldsymbol{X}^A) - f^B_1(\boldsymbol{X}^B))||_2 \tag{8}$$

Many possible variations of DeepDiff exist through the various combinations of the auxiliary tasks in the multi-tasking framework as well as the variations of raw HM features. For example, we can use all auxiliary tasks to multi-task with the main task. This means the sum of the main and auxiliary task losses is used as part of the training objective: (i) Differential expression prediction loss for the main task (ℓ_{Diff}), (ii) Auxiliary task loss ($\ell_{CellAux}$) from the Cell-specific prediction tasks and (iii) Contrastive Siamese Loss ($\ell_{Siamese}$) from the Siamese-Auxiliary Task. The network is trained using similar steps outlined in the main DeepDiff task with the sum of these losses as training objective.

In our experiments, we have evaluated the following DeepDiff variations:

- (**Raw: d**) Raw Difference Features;
- (**Raw: c**) Concatenation of Raw HM features;
- (**Raw**) Concatenation and difference of raw HM features;
- (**Aux**) Auxiliary Embeddings as Features;
- (**Raw+Aux**) Concatenation and Difference of HMs + Embeddings from Auxiliary tasks;
- (**Aux+Siamese**) Auxiliary Features with Siamese Contrastive Loss;
- (**Raw+Aux+Siamese**) Raw and Auxiliary Features with Siamese Contrastive Loss

Figure 3 and Supplementary Table S1 show the different auxiliary tasks and resulting different DeepDiff variations through

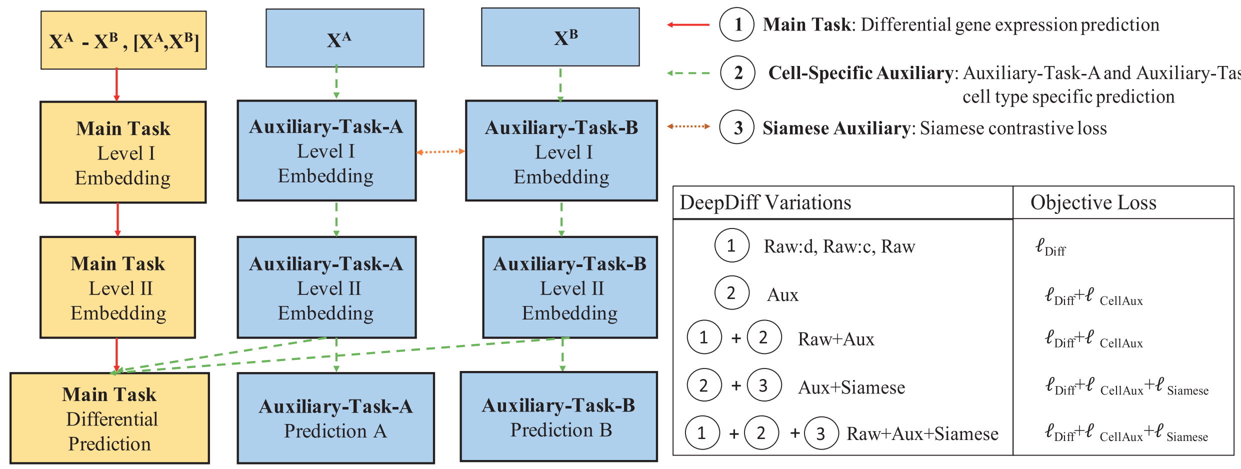
$X^A - X^B$, $[X^A, X^B]$
X^A
X^B
Main Task Level I Embedding
Auxiliary-Task-A Level I Embedding
Auxiliary-Task-B Level I Embedding
Main Task Level II Embedding
Auxiliary-Task-A Level II Embedding
Auxiliary-Task-B Level II Embedding
Main Task Differential Prediction
Auxiliary-Task-A Prediction A
Auxiliary-Task-B Prediction B
1 Main Task: Differential gene expression prediction
2 Cell-Specific Auxiliary: Auxiliary-Task-A and Auxiliary-Task-B cell type specific prediction
3 Siamese Auxiliary: Siamese contrastive loss
DeepDiff Variations
Objective Loss
1 Raw:d, Raw:c, Raw
ℓ_{Diff}
2 Aux
$\ell_{Diff}+\ell_{CellAux}$
1 + 2 Raw+Aux
$\ell_{Diff}+\ell_{CellAux}$
2 + 3 Aux+Siamese
$\ell_{Diff}+\ell_{CellAux}+\ell_{Siamese}$
1 + 2 + 3 Raw+Aux+Siamese
$\ell_{Diff}+\ell_{CellAux}+\ell_{Siamese}$

Fig. 3. We implement multiple variations of DeepDiff through deep learning based multitasking. Our system includes a set of auxiliary tasks shown as units in this Figure. In details, we use two types of auxiliary tasks coupled with the main DeepDiff task of differential gene expression. The *Cell-specific Auxiliary* tasks, denoted by *Auxiliary-Task-A* and *Auxiliary-Task-B* cell-type specific gene expression prediction. The second *Siamese-Auxiliary* task uses the siamese contrastive loss at the Level I Embedding outputs. The DeepDiff variations are indicated as combinations of the main task, auxiliary tasks and variations of the input

various combinations with the raw HM features. Due to space limitation, we have the detailed description of each variation in Supplementary Section S1.3. In the rest of the paper, we use the short names enclosed in parantheses above to refer to each variation.

4 Experimental setup and results

4.1 Dataset

We downloaded gene expression and HM signal data of five core histone modification signals for ten different cell types from the REMC database (Kundaje *et al.*, 2015). Supplementary Table S5 summarizes the IDs and information of the ten cell-types we use that have been extensively profiled by the Roadmap Epigenomics Project. Supplementary Table S4 describes the five core histone modifications marks we use along with their known important roles in gene regulation. For the regression labels, we use the log fold change of raw counts in the two cell-types under consideration as the target label for the main task and the log of the raw counts in each cell type for the two cell-specific auxiliary tasks. In total, we apply DeepDiff variations and baselines (see Section 4.2) on ten pairs of cell-types from REMC. Supplementary Table S6 provides a list of the ten cell-type pairs. For each cell type pair, we have a sample set of total 18 460 genes. This set was divided into 3 separate folds: training (10 000 genes), validation (2360 genes) and test (6100 genes) folds.

4.2 Baselines

We compare DeepDiff to two variations of the Support Vector Regression (SVR) (Cheng and Gerstein, 2012), the ReliefF based feature selection followed by Random Forest Regression (Li *et al.*, 2015a) and the AttentiveChrome (Singh *et al.*, 2017). In details:

- *Single Layer SVR (*Cheng and Gerstein, 2012*):* The authors selected 160 bins from regions flanking each gene TSS and TTS. Each bin uses a separate SVR, resulting in 160 different bin-specific SVR models. The radial basis kernel is used for the SVR. Cheng and Gerstein (2012) proposed to use the best-bin strategy. Therefore, by using cross-validation, we pick the best bin based on Pearson Correlation Coefficient (PCC) used by Cheng and Gerstein (2012). The best bin model is then used for prediction on the test set.
- *Two Layer SVR (*Cheng and Gerstein, 2012*):* The two-layer model in Cheng and Gerstein (2012) seeked to combine the signals of all HMs across all the 160 bins. In the first layer, it predicts expression levels in each of the bins using a bin-specific SVR model in each individual bin. Then the expression levels predicted by each bin are combined in the second layer using another SVR model to make a final prediction. The radial basis kernel is used for both layers of the SVR.
- *ReliefF Feature Selection + Random Forest (*Li *et al.*, 2015a*):* We implement the best performing combination in Li *et al.* (2015a): ReliefF algorithm for feature selection followed by random forest. While Li *et al.* (2015a) treat the problem as a binary classification with two classes of upregulated versus downregulated genes, we used this baseline for our regression formulation. The number of features for ReliefF is selected from the set of $\{50, 70, 100\}$. The number of trees for Random Forest is selected from the set of $\{10, 50, 100, 150, 200\}$.
- AttentiveChrome (Singh *et al.*, 2017): We also compare our models to the differential patterns derived from predictions made by AttentiveChrome. We train AttentiveChrome for cell type specific gene expression as a regression task. Then we calculate the differential gene expression prediction as the log fold change between the predicted expression from the two trained models that are specific to the two cell-types under consideration. In detail, for each pair of cell-types, we train two cell-specific AttentiveChrome models independently from each other. We then use the cell type specific predictions of each model to calculate differential gene expression. Clearly, this baseline is not an end-to-end solution for differential gene expression prediction.

We implemented SVR and Random Forest baselines using the scikit-learn (Pedregosa *et al.*, 2011) package. We implemented AttentiveChrome and DeepDiff models in Pytorch. We use Pearson Correlation Coefficient to evaluate our models. We train all the variations of DeepDiff (summarized in Supplementary Table S1) were trained using our training set and tune the hyperparameter on the validation set. The best performing models were then

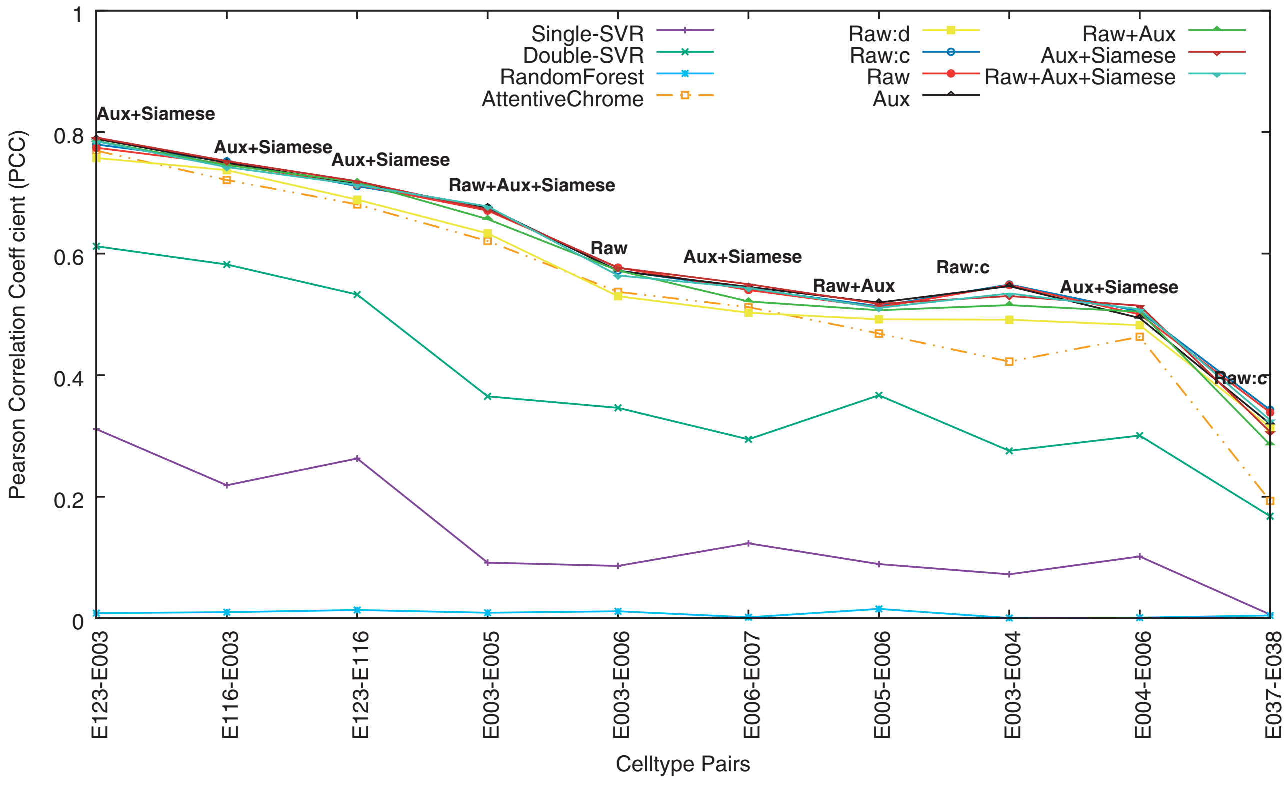

Fig. 4. Pearson Correlation Coefficient (PCC) for all DeepDiff variations along with the baselines for each cell-type pair (x-axis). The best performing DeepDiff model variation is indicated by the text label for each case (cell-type pair)

evaluated on the test set. The details about evaluation metric and hyperparameters for DeepDiff and baselines are in the Supplementary Section S1.5.

4.3 Performance evaluation

Figure 4 shows the Pearson Correlation Coefficient (y-axis) for all DeepDiff variations versus the baselines. The x-axis shows the ten cases (cell-type pairs) in our experiments. The deep learning based models outperform both the SVR as well as Random Forest baselines. The two-layer SVR model performs better than one layer SVR. When comparing DeepDiff variations to the AttentiveChrome baseline, DeepDiff also outperforms, indicating the need for modeling differential gene expression prediction. Among the DeepDiff variations, the *Raw: d* model performs the worst in 9 out of the 10 cases. This indicates that simply taking the difference of the HM signals is not enough to model the combinatorial interactions of HMs for differential regulation. Instead, using all HM features, instead of only the difference, is clearly helpful, as indicated by the higher PCC of *Raw: c* in comparison to *Raw: d* across all 10 cases. The results of *Raw+Aux* show that adding cell-type specific gene expression prediction in the two cell-types as auxiliary tasks clearly helps in improving the prediction performance. Combining *Aux* and *Raw* features gives better performance than only *Aux* and *Raw* in 7 and 6 out of the 10 cell type pairs, respectively. *Aux+Siamese* is the best performing model in 5 out of the 10 cases. This shows that adding the Siamese-based contrastive loss improves the prediction performance. Table 1 shows the mean and median of the relative performance (%) with respect to PCC when being compared to the best performing baselines: two-layer SVR and AttentiveChrome. For example, as shown in Table 1, when averaged across the 10 cases, combining the raw and auxiliary features (*Raw+Aux* variation) results in a relative PCC with respect to the two-layer SVR baseline of 162.23%.

4.4 Interpreting differential regulation using attention

Finally, we analyze the attention weights of Level II Embedding for one of the best performing case: cell type pair E116 and E123. Here, E116 represents 'normal' blood cell (GM12878) whereas E123 represents the leukemia cell (K562). We aim to validate that the learned attention weights can provide some insights into the differential gene regulation across these two selected cell types: a normal and a diseased (cancer) cell state. We only use the attention weights obtained on the test set for this analysis. First, we get a list of down-regulated genes (with log fold change < -8.0) and a list of up-regulated genes (with log fold change >8.0) from the test set. Figure 5 plots the average attention weights of each of the 5 HMs when considering all of our selected up-regulated (black bars) and down-regulated genes (gray bars). We observe that for the top upregulated genes, H3K4me1 (enhancer associated) and H3K4me3 (promoter associated) get the highest weights (i.e. contributing more importance). This is consistent with observations by Grégoire *et al.* (2016) that upregulated genes are more enriched for H3K4me1 and H3K4me3 when under cancer condition.

For top down-regulated genes (Fig. 5), H3K27me3 (Polycomb Repression associated) gets a comparatively higher weight (the second highest attention weight). Grégoire *et al.* (2016) also reported this trend showing that down-regulated genes are more enriched with the repressive H3K27me3 under the cancer condition whereas up-regulated genes do not show variation of this HM between normal and cancer conditions. Figure 5 shows that H3K9me3 (heterochromatin linked) gets low attention weights for both up-regulated and down-regulated genes. Grégoire *et al.* (2016) also reported this trend

Table 1. Mean and Median of the relative performance (%) with respect to Pearson Correlation Coefficient (PCC) when comparing DeepDiff models to the best-performing baselines: two-layer SVR and AttentiveChrome across all ten cell-type pairs

Method	Two-layer SVR		AttentiveChrome	
	Mean	Median	Mean	Median
Raw: d	153.72	156.81	108.90	102.14
Raw: c	163.59	166.64	115.75	107.64
Raw	162.94	166.55	115.26	107.71
Aux	157.64	166.58	111.20	106.13
Raw+Aux	162.23	164.73	114.61	106.62
Aux+Siamese	161.85	168.90	114.19	107.87
Raw+Aux+Siamese	161.95	166.07	114.44	107.56

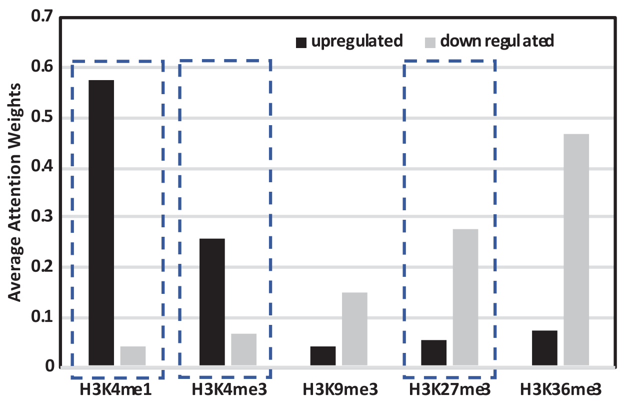

Fig. 5. HM attention weights at Level II Embedding for one of the best performing cell type pair—E116 and E123. Here, E116 represents 'normal' blood cell (GM12878) whereas E123 represents 'cancer' or leukemia cell (K562). We plot the average attention weights for all 5 HMs across down-regulated (log fold change < -8.0) and up-regulated genes (log fold change >8.0). H3K4me1 and H3K4me3 get the highest weights in upregulated genes and comparatively lower weights in downregulated genes, indicating the importance of these histone modifications for upregulation. Similarly, H3K27me3 gets a comparatively higher weight (second highest weight) in downregulated genes than the same histone modification in upregulated genes. Both these observations have been confirmed previously by Grégoire *et al.* (2016) through HM enrichment analysis across the two cell types

that the divergence of expression is less likely due to this HM. We want to emphasize again that our model learns the importance of HMs for the differentially expressed genes in an end-to-end manner.

5 Conclusion

We have presented DeepDiff, a deep learning framework for differential gene expression prediction using histone modifications (HMs). DeepDiff is an attention-based deep learning architecture designed to understand how different HMs work together to influence changes in expression patterns of a gene between two different cell types. DeepDiff uses a modular architecture to represent the spatially structured and long-range HM signals. It incorporates a two-level attention mechanism that gives it the ability to find salient features at the bin level as well as the HM level. Additionally, to deal with fewer differentially expressed genes between two cell types, we design a novel multi-task framework to use the cell-type-specific prediction network as auxiliary tasks to regularize our primary task of differential expression prediction. We also incorporate a Siamese contrastive loss term to further improve the learned representations. For the future work, we will evaluate the performance and the attention scores of DeepDiff on more cell type pairs. We will incorporate additional epigenomic signals that may relate to differential gene expression. We would also like to explore different ways to interpret and validate the attention weights. In summary, leveraging deep learning's ability to extract rich representations from data can enhance our understanding of gene regulation by HMs, thus enabling insights into principles of gene regulation through epigenetic factors.

Conflict of Interest: none declared.

Funding

This work was partly supported by the National Science Foundation under NSF CAREER award No. 1453580. Any Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

References

Bannister,A.J. and Kouzarides,T. (2011) Regulation of chromatin by histone modifications. *Cell Res.*, **21**, 381–395.

Caruana,R. (1997) Multitask learning. *Mach. Learn.*, **28**, 41–75.

Cheng,C. and Gerstein,M. (2012) Modeling the relative relationship of transcription factor binding and histone modifications to gene expression levels in mouse embryonic stem cells. *Nucleic Acids Res.*, **40**, 553–568.

Chorowski,J.K. *et al.* (2015) Attention-based models for speech recognition. In: *Advances in Neural Information Processing Systems, Proceeding NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems*, **vol. 1**. Montreal, Canada, December 07–12, 2015, pp. 577–585.

Corbetta,M. and Shulman,G.L. (2002) Control of goal-directed and stimulus-driven attention in the brain. *Nat. Rev. Neurosci.*, **3**, 201–215.

Costa,I.G. *et al.* (2011) Predicting gene expression in t cell differentiation from histone modifications and transcription factor binding affinities by linear mixture models. *BMC Bioinformatics*, **12**, S29–S21.

Dong,X. *et al.* (2012) Modeling gene expression using chromatin features in various cellular contexts. *Genome Biol.*, **13**, R53.

Dzmitry,B. *et al.* (2014) Neural machine translation by jointly learning to align and translate. *arXiv Preprint arXiv*, **1409**, 0473.

Egger,G. *et al.* (2004) Epigenetics in human disease and prospects for epigenetic therapy. *Nature*, **429**, 457.

Frasca,M. and Pavesi,G. (2013) A neural network based algorithm for gene expression prediction from chromatin structure. In: *The 2013 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Gjoneska,E. *et al.* (2015) Conserved epigenomic signals in mice and humans reveal immune basis of Alzheimer's disease. *Nature*, **518**, 365.

Grégoire,L. *et al.* (2016) The transposable element environment of human genes is associated with histone and expression changes in cancer. *BMC Genomics*, **17**, 588.

Hadsell,R. *et al.* (2006) Dimensionality reduction by learning an invariant mapping. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **vol. 2**, IEEE, pp. 1735–1742.

Ho,B.H. *et al.* (2015) Combinatorial roles of DNA methylation and histone modifications on gene expression. In: Dang,Q.A. *et al.* (eds) *Some Current Advanced Researches on Information and Computer Science in Vietnam*. Springer, Cham, pp. 123–135.

Hochreiter,S. and Schmidhuber,J. (1997) *Long Short-Term Memory*. **vol. 9**. MIT Press, Cambridge, Massachusetts, pp. 1735–1780.

Huijuan,X. and Kate,S. (2016) Ask, attend and answer: exploring question-guided spatial attention for visual question answering. In: *ECCV*.

Ilya,S. *et al.* (2014) Sequence to sequence learning with neural networks. In: *Advances in Neural Information Processing Systems*, pp. 3104–3112.

Jimmy,B. *et al.* (2014) Multiple Object Recognition with Visual Attention. arXiv preprint arXiv:1412.7755.

Karlić,R. *et al.* (2010) Histone modification levels are predictive for gene expression. *Proc. Natl. Acad. Sci. USA*, **107**, 2926–2931.

Xu,K. *et al.* (2015) Show, attend and tell: neural image caption generation with visual attention. In: *ICML*, **volume 14**, pp. 77–81.

Koch,C.M. *et al.* (2007) The landscape of histone modifications across 1% of the human genome in five human cell lines. *Genome Res.*, **17**, 691–707.

Kononenko,I. *et al.* (1997) Overcoming the myopia of inductive learning algorithms with relieff. *Appl. Intell.*, **7**, 39–55.

Kundaje,A. *et al.* (2015) Integrative analysis of 111 reference human epigenomes. *Nature*, **518**, 317–330.

Léon,B. (2004) Stochastic learning. In: Bousquet,O. *et al* (eds) *Advanced Lectures on Machine Learning*. Springer, pp. 146–168.

Li,J. *et al.* (2015a) Using epigenomics data to predict gene expression in lung cancer. *BMC Bioinformatics*, **16**, S10.

Li,Y. *et al.* (2015b) Describing videos by exploiting temporal structure. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE.

Minh-Thang,L. *et al.* (2015) Effective approaches to attention-based neural machine translation. In: Màrquez,L. *et al.* (eds) *Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Lisbon, Portugal, pp. 1412–1421.

Meisner,M. and Reif,D.M. (2015) Computational methods used in systems biology. In: *Systems Biology in Toxicology and Environmental Health*. Elsevier, Cambridge, Massachusetts, pp. 85–115.

Omony,J. (2014) Constrained stochastic space search method for parameter estimation in biological networks. *Adam Method Stochastic Optim.*, **4**, 952.

Oriol,V. *et al.* (2015) Pointer networks. In: Cortes,C. *et al.* (eds.) *Advances in Neural Information Processing Systems*, **vol. 28**. Curran Associates, Inc., pp. 2692–2700.

Pedregosa,F. *et al.* (2011) Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.*, **12**, 2825–2830.

Singh,R. *et al.* (2016) Deepchrome: deep-learning for predicting gene expression from histone modifications. *Bioinformatics*, **32**, i639–i648.

Singh,R. *et al.* (2017) Attend and Predict: Understanding Gene Regulation by Selective Attention on Chromatin. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA*, 4-9 December 2017, pp. 6785–6795.

Volodymyr,M. *et al.* (2014) Recurrent models of visual attention. In: *Advances in Neural Information Processing Systems*, pp. 2204–2212.

Weng,N.-P. *et al.* (2012) The molecular basis of the memory t cell response: differential gene expression and its epigenetic regulation. *Nat. Rev. Immunol.*, **12**, 306.

Zichao,Y. *et al.* (2016) *Hierarchical Attention Networks for Document Classification*. Association for Computational Linguistics, San Diego, pp. 1480–1489.