Sequence Analysis

EPIP: A novel approach for condition-specific enhancer-promoter interaction prediction

Amlan Talukder 1, Samaneh Saadat 1, Xiaoman Li 2,* and Haiyan Hu 1, *

1Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA and
2Burnett School of Biomedical Science, College of Medicine, University of Central Orlando, FL 32816, USA.

* To whom correspondence should be addressed. Tel: +1 407 882 0134; Fax: +1 407 823 5835; Email: haihu@cs.ucf.edu. Correspondence may also be addressed to Xiaoman Li. Tel: +1 407 823 4811; Fax: +1 407 823 5835; Email: xiaoman@mail.ucf.edu.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: The identification of enhancer-promoter interactions (EPIs), especially condition-specific ones, is important for the study of gene transcriptional regulation. Existing experimental approaches for EPI identification are still expensive, and available computational methods either do not consider or have low-performance in predicting condition-specific EPIs.

Results: We developed a novel computational method called EPIP (Enhancer Promoter Interaction Prediction) to reliably predict EPIs, especially condition-specific ones. EPIP is capable of predicting interactions in samples with limited data as well as in samples with abundant data. Tested on more than eight cell lines, EPIP reliably identifies EPIs, with an average area under the receiver operating characteristic curve of 0.95 and an average area under the precision-recall curve of 0.73. Tested on condition-specific enhancer-promoter interactions, EPIP correctly identified 99.26% of them. Compared with two recently developed methods, EPIP outperforms them with a better accuracy.


Contact: xiaoman@mail.ucf.edu or haihu@cs.ucf.edu

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

Enhancers play important regulatory roles. They control the expression patterns of their target genes by directly interacting with the promoter regions of those genes (Mora et al., 2015). Even though enhancers may be tens of kilobases (kb) away from their target genes, they get in direct contact with the promoters of their target genes via chromatin looping (Dekker et al., 2002; De Laat et al., 2013; Cai et al., 2010; Zheng et al., 2014). Because of the long range of possible distances (1 kb to several megabases (Mb)) between enhancers and their targeted promoters, it is challenging to predict enhancer-promoter interactions (EPIs) (De Laat et al., 2013). To date, the majority of EPIs under specific experimental conditions have not been discovered yet (Corradin et al., 2014).

Experimental approaches for identifying EPIs are mainly based on chromosome conformation capture (3C) and its variants such as chromatin interaction analysis with paired-end tag sequencing (ChIA-PET) and high throughput genome-wide chromosome conformation capture (Hi-C) (Dekker et al., 2002; Fullwood et al., 2009; Rao et al., 2014). These experimental techniques determine the relative frequency of direct physical contacts between genomic regions and have successfully identified EPIs and other long-range interactions (He et al., 2014). However, the ChIA-PET method still has a low signal-to-noise ratio and most available Hi-C data have a low resolution (Fullwood et al., 2009; Rao et al., 2014). In addition, since certain EPIs are condition-specific, experimental EPI data in one sample cannot always be directly applied to infer EPIs in other samples. Here, a “sample” refers to a cell type, a cell line, or a tissue sample under a specific experimental condition. An EPI is called condition-specific, if the interaction only occurs in a specific sample.

As most experimental procedures are expensive, computational methods have been indispensable alternatives for identifying EPIs. These methods employ available genomic and/or epigenomic data to predict EPIs in an inexpensive way. Early methods considered the closest promoter as the only target of an enhancer. However, a study demonstrated that only 40% of enhancers regulate their nearest promoters and one enhancer...
may regulate multiple genes (Andersson et al., 2014). Later, several computational approaches were developed based on the correlation of epigenomic signals in enhancers and those in promoters (Corradin et al., 2014; Andersson et al., 2014; Thurman et al., 2012; Ernst et al., 2011). One challenge of using these methods is to find a proper threshold of correlations to reduce false EPI predictions (Roy et al., 2015; Whalen et al., 2016). Recently, supervised learning based methods have been developed, such as IM-PET (He et al., 2014), PETModule (Zhao et al., 2016), RIPPLE (Roy et al., 2015), and TargetFinder (Whalen et al., 2016). These methods commonly use genomic and epigenomic data such as those from DNase I hypersensitive sites sequencing (DNase-seq) and histone modification based chromatin immunoprecipitation followed by massive parallel sequencing (ChIP-seq) to extract features for EPI predictions. IM-PET, RIPPLE, and PETModule utilize random forests as their classifier, while TargetFinder is based on boosted trees. These methods either do not consider or have low-performance on condition-specific EPI predictions (Roy et al., 2015).

Here we proposed a computational method for predicting condition-specific EPIs called EPIP. EPIP stands for “Enhancer-Promoter Interaction Prediction”. It is a supervised learning-based approach that utilizes functional genomic and epigenomic data to build a robust model to predict shared and condition-specific EPIs. EPIP can work with missing data, different types of datasets, and even a dataset with a partial list of features. Tested on experimental data from more than eight samples, EPIP reliably predicted condition-specific EPIs and shared EPIs in different samples with the average area under the receiver operating characteristic curve (AUCROC) about 0.95, and the average area under the precision-recall curve (AUPR) about 0.73. In addition, we compared EPIP with two state-of-the-art computational methods for predicting EPIs and showed that EPIP outperformed both.

2 Materials and Methods

2.1 Enhancers and promoters

We obtained all 32693 enhancers annotated by FANTOM from http://slidebase.binf.ku.dk/human_enhancers/results (Andersson et al., 2014). We chose this set of enhancers because this was arguably the largest set of enhancers that were defined with the same criteria and supported by experiments. We next overlapped the FANTOM enhancers with the computationally predicted ChromHMM enhancers (Ernst et al., 2011) for samples that had the ChromHMM data available (GM1278, HeLa, HMEC, HUVEC, IMR90 and NHEK). These ChromHMM enhancers were defined with fifteen hidden states (https://genome.ucsc.edu/cgi-bin/hgTrackUi?g=wgEncodeBroadHmmb&db=hg19). We considered both strong and weak enhancers (states 4–7) as valid ChromHMM enhancers (Ernst et al., 2011). The FANTOM enhancers overlapping with at least one ChromHMM enhancer in a sample were considered as the enhancers for that sample in the following analyses. Since KBM7 does not have any annotated ChromHMM enhancer, all FANTOM enhancers were used for KBM7. An enhancer was considered “active” in a sample if it overlapped with the H3K27ac ChIP-seq peaks in this sample. The H3K27ac peaks were downloaded from ENCODE (Dunham et al., 2012). With no H3K27ac data available for KBM7, all obtained enhancers were considered as “active” enhancers. In this way, we obtained 7023 to 32693 enhancers and 4888 to 32693 active enhancers in a sample (Table S1).

We obtained all annotated transcription start sites (TSSs) from GENCODE V 19 (Harrow et al., 2012) and considered the regions between 1 kb upstream and 100 base pairs downstream of the TSSs as “promoters”. This resulted in 57783 promoters. For samples with RNA-Seq data (Dunham et al., 2012) (GM1278, HeLa, HUVEC, IMR90, K562 and NHEK), we defined promoters as “active” if the corresponding genes had at least 0.30 reads per kb of transcript per million mapped reads with the irreproducible discovery rate of 0.1, similarly as previously (Whalen et al., 2016). For samples without RNA-Seq data (HMEC and KBM7), all promoters were considered as active promoters (Table S2).

2.2 Training data

To train EPIP, we defined positive and negative enhancer-promoter pairs (EPI-pairs) (i.e. interacting and non-interacting EPI-pairs) using the normalized Hi-C contact matrices, which were generated with the Knight and Ruiz normalization vectors by Rao et al. (Rao et al., 2014). Rao et al. inferred these matrices for the following seven samples: GM1278, HMEC, HUVEC, IMR90, K562, KBM7 and NHEK (GSE63525). They also extracted significant intra-chromosomal chromatin interactions called “looplists” in the above seven samples and the HeLa sample. The number of EPI-pairs from the looplists defined at the highest resolution for these samples was too small to train the EPIP model well. We thus defined the positive and negative EPI-pairs from their normalized Hi-C contact matrices, as previously (Li et al., 2016; Zhao et al., 2016) (Table S3).

In brief, if an “active” enhancer and an “active” promoter overlapped with a pair of regions that were supported by at least 30 normalized Hi-C reads, we considered this EPI-pair as a positive EPI-pair. Similarly, an EPI-pair was considered as negative if it did not overlap with any pair of regions that were supported by 5 or more normalized Hi-C reads (Figure 1A). The cutoffs, 5 and 5, were chosen based on our test results with different cutoffs (Table S3). In this way, we defined positive and negative EPI-pairs for the above seven samples with contract matrices. To train EPIP, we used both balanced and unbalanced models. We randomly chose 30% of positive EPI-pairs and the same number of negative EPI-pairs in each of the above seven samples. We then combined these positives and negatives from different samples to train a balanced prediction model. We also combined 30% of positive EPI-pairs and ten times randomly-chosen negative EPI-pairs in each sample to train an unbalanced prediction model. We then combined the two models into the final EPIP model, which predicts an EPI-pair as a “negative” pair only when both models predict this pair as a negative pair and predicts an EPI-pair as a positive pair otherwise. This strategy was based on the observation that the balanced model had a high sensitivity and the unbalanced model had a high specificity when tested on the training data by cross-validation. For simplicity’s reason, in the remaining of the paper, we called this final EPIP model as “EPIP”.

2.3 Testing data

We tested EPIP on a variety of data (Figure 1B). We tested it on the remaining 70% of positive EPI-pairs, together with the same number of randomly selected negative pairs that were not used for training (balanced test data). We also tested it on the remaining 70% of positive EPI-pairs together with ten times randomly selected negative pairs that were not used for training (unbalanced test data). We tested EPIP on all EPI-pairs within 2 Mb that were not used for training as well. Moreover, we tested EPIP on the positive EPI-pairs defined with normalized Hi-C contact matrices under the cutoffs 10, 20, 30, 50, and 100. Finally, we tested EPIP on EPI-pairs collected in other studies (Li et al., 2012; Jin et al., 2012; Rao et al., 2014), which were obtained from the strictly-defined interacting regions by the original studies and represented more strictly-defined EPI-pairs.

2.4 Features of EPI-pairs considered

EPIP considers three common features of EP-pairs in every sample. These features are the distance between the enhancer and the promoter in an EP-pair, the conserved synteny score that measures the co-conservation of an
EPIP pair in five other vertebrate genomes (chicken gaGal, chimpanzee panTro, frog xenTro3, mouse mm10 and zebrafish zv9) and the correlation of epigenomic signals in the enhancer region and that in the promoter region of an EP-pair across ENCODE Tiers 1 and 2 samples (Zhao et al., 2016). For simplicity’s sake, in the following, these features are called “distance”, “css”, and “corr”, respectively.

In addition, depending on the types of data available in a sample, EPPI considers features from fourteen additional types of data. These include DNase-seq data. ChiP-seq data for nine types of histone modifications (H3K4me1, H3K4me2, H3K27ac, H3K27me3, H3K36me3, H3K79me2, H3K9ac and H4K20me1) and four types of chromatin factors (CTCF, POL2, RAD21 and SMC3). These data are shown to provide important indicators for predicting EPIs (Roy et al., 2015). For each of the fourteen types of data, EPPI generates two features that correspond to its signals in enhancer regions and its signals in promoter regions (Table S4). The value of a feature for a region corresponds to the “peak strength” value of this feature in its signal peak that overlapped with this region. If a region overlaps with multiple peaks of a feature signal, we considered the average signal value in these overlapping peaks as the feature value for this region.

For instance, when H3K4me1 ChiP-seq data is available for an enhancer in a sample, the H3K4me1 feature value for this enhancer is the average peak strength of all H3K4me1 ChiP-seq peaks overlapping with this enhancer: the feature signal peaks and their signal strength are downloaded from ENCODE (Dunham et al., 2012). Due to the difference of available types of data in different samples, we could consider 31, 25, 27, 31, 3 and 27 features in GM12878, HMEC, HUVEC, IMR90, K562, KBM7 and NHEK, respectively, including the three common features (Table S4).

2.5 Partitioning feature space to handle missing data

EPPI groups features into eleven partitions or overlapping feature sets (Figure 2A, Table S5). Partitions with overlapping features are used, because in this way, (i) EPPI can be trained and tested on various samples, no matter whether the samples have data for a large or small number of features; (ii) such partitions enable more samples to be used to train each partition and thus likely produce more accurate predictors; and (iii) the trained EPPI model can be used to make predictions in more samples. For instance, samples with a large number of features will benefit from a large number of partitions, while EPPI can still make predictions for samples with a small number of features.

In brief, EPPI considers the three common features: distance, css, and corr, as a partition, which shows the static genomic information of EPIs. Moreover, EPPI considers the above three common features together with features from each of the following two feature groups as a different partition: H3K4me1; DNase-seq; H3K4me1 and H3K27ac; DNase-seq and H3K27ac; H3K4me1 with H3K27ac and H3K4me3; DNase-seq with H3K27ac and H3K4me3; H3K4me1-3 together with H3K27ac and DNase-seq (Table S5). Note that for every histone modification mark, EPPI considers its signal values in enhancers and in promoters. Therefore, the above partitions have two features for every histone modification mark (such as H3K4me1_P and H3K4me1_E for the mark H3K4me1 in promoters and enhancers, respectively). Finally, EPPI considers the largest sets of features that are not considered above and occur in at least one training sample as additional partitions (Figure 2B).

2.6 An ensemble approach to predict condition-specific EPIs under a variety of conditions

We developed an ensemble method called EPPI to distinguish positive from negative EP-pairs (Figure 2A). The method uses the training data to train the model iteratively, based on the idea of AdaBoost (Polikar et al., 2000). The details are in the following paragraphs.

With the above partitions of features, EPPI trains an incremental learner for every partition. An incremental learner consists of a number of weak learners trained on the training data. The weak learners in EPPI are decision tree classifiers. We set a maximum allowed depth of ten for the decision trees to avoid overfitting, after testing on several depth options. For a given partition and a corresponding sample, EPPI trains 200 weak learners, because 200 is the smallest number that gives EPPI the highest AUROC and AUPR scores (Figure S1). With weak learners trained on data from all samples with the required features by the corresponding partition, the corresponding incremental learner combines predictions from all its weak learners through a majority voting. EPPI then combines the predictions by the trained incremental learners from all partitions by a majority voting (Figure 2A).

EPPI trains weak learners iteratively. Given a partition and a sample with the required features by this partition, the first weak learner is trained
We did not compare them on the Ripple data, as (i) the Ripple data is TargetFinder data in three shared samples (GM12878, HeLa, and K562). Compared EPIP with Ripple by running them on EPIP data and the above were based on the 5C (GSE39510) and Hi-C (GSE63525) datasets. We condition-specific interactions in a new sample. Its training and test data RIPPLE is trained on multiple samples and is capable of predicting data such as DNase-seq, ChIP-seq, and RNA-Seq data to extract features. multi-task learning framework (Roy et al., 2015). It uses different types of data such as DNase-seq, ChIP-seq, and RNA-Seq data to extract features. RIPPLE is trained on multiple samples and is capable of predicting condition-specific interactions in a new sample. Its training and test data were based on the SC (GSE39510) and Hi-C (GSE63525) datasets. We compared EPIP with Ripple by running them on EPIP data and the above TargetFinder data in three shared samples (GM12878, HeLa, and K562). We did not compare them on the Ripple data, as (i) the Ripple data is balanced, which does not represent the reality well, where we often have much more negatives than positive EP-pairs; (ii) the resolution of the Ripple data is low, where a promoter within 2.5 kb of a pair of interacting regions may be considered as the targets of one of the regions; and (iii) the data barely overlap with any FANTOM enhancer.

We used the 10-fold cross-validation method to train and test EPIP, TargetFinder and Ripple, similarly as that in the TargetFinder study and in the Ripple study. We used the generate_train.py in TargetFinder to generate TargetFinder features for EP-pairs. Then we used the procedure mentioned in its readme file to apply the 10-fold cross-validation on the training data using GradientBoostingClassifier (GBM). In terms of Ripple, we used the genFeatures tool in Ripple to generate features for EP-pairs. Then we used its runAllFeatures_crosscellline.m Matlab code to apply 10-fold cross-validation on the training data.

### 3 Results

#### 3.1 EPIP reliably predicts EPIs

We tested EPIP on the balanced test data, unbalanced test data, all EP-pairs within 2.5 kb to 2 Mb, EP-pairs defined with different normalized Hi-C contact number cutoffs, and EP-pairs from other studies. EPIP reliably predicted untrained EPIs in all datasets, with a high AUROC, AUPR and/or F1 score (Table 1, Table S6-S8). The AUROC, AUPR and the F1 score predicted untrained EPIs in all datasets, with a high AUROC, AUPR and/or F1 score (Table 1, Table S6-S8). The AUROC, AUPR and the F1 score calculated using the scikit-learn libraries (Pedregosa et al., 2011).

We studied the performance of EPIP on the balanced test data, the unbalanced test data, and all EP-pairs within 2.5 kb to 2 Mb (Material and Methods). No EP-pair in these test data was used for training. With five sets of randomly chosen training data and the corresponding test data, on average, EPIP had an AUROC of 0.96, 0.96, and 0.95; an AUPR of 0.96, 0.92, and 0.73; an F1 score of 0.99, 0.95, and 0.51 for the balanced, unbalanced, and all EP-pairs within 2.5 kb to 2 Mb test data, respectively (Table 1 and S6). Note that the F1 score on the third test data was not bad, given the fact that the number of negatives was around 13 times the number of positives here. In this test dataset, the recall in all samples was higher than 0.92, although KBM7 had no epigenomic data. The average precision was 0.34 in these samples, with the largest precision in GM12878, where the Hi-C sequencing depth was the highest. The much higher precision and F1 scores may be underestimated in other samples, as the lower sequencing
Table 1. The performance of EPiP on all pairs within 2.5kb and 2Mb, balanced and unbalanced test data.

<table>
<thead>
<tr>
<th>Cell line</th>
<th>AUROC</th>
<th>AUPR</th>
<th>F1</th>
<th>Precision</th>
<th>Sensitivity/Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>GM12878</td>
<td>0.9907 (0.9944,0.9961)</td>
<td>0.8502 (0.9939,0.9951)</td>
<td>0.9337 (0.9942,0.9962)</td>
<td>0.8777 (0.9979,0.9978)</td>
<td>0.9979 (0.9906,0.9936)</td>
</tr>
<tr>
<td>HMEC</td>
<td>0.9872 (0.9962,0.9981)</td>
<td>0.9308 (0.9963,0.9992)</td>
<td>0.4287 (0.9893,0.9415)</td>
<td>0.2733 (0.9873,0.8955)</td>
<td>0.9938 (0.9913,0.9925)</td>
</tr>
<tr>
<td>HUVEC</td>
<td>0.9842 (0.9943,0.9934)</td>
<td>0.8904 (0.9952,0.9807)</td>
<td>0.3081 (0.9844,0.9369)</td>
<td>0.1824 (0.9871,0.9878)</td>
<td>0.9915 (0.9818,0.9796)</td>
</tr>
<tr>
<td>IMR90</td>
<td>0.9963 (0.9985,0.9991)</td>
<td>0.9894 (0.9998,0.9995)</td>
<td>0.6365 (0.9933,0.9577)</td>
<td>0.4472 (0.9921,0.9247)</td>
<td>0.9985 (0.9994,0.9992)</td>
</tr>
<tr>
<td>K562</td>
<td>0.9947 (0.9985,0.9991)</td>
<td>0.9874 (0.9988,0.9993)</td>
<td>0.5599 (0.9941,0.9686)</td>
<td>0.3587 (0.9900,0.9141)</td>
<td>0.9958 (0.9950,0.9959)</td>
</tr>
<tr>
<td>KBM7</td>
<td>0.9856 (0.9856,0.9861)</td>
<td>0.9285 (0.9876,0.9590)</td>
<td>0.5317 (0.9664,0.9328)</td>
<td>0.3629 (0.9876,0.9895)</td>
<td>0.994 (0.9852,0.9793)</td>
</tr>
<tr>
<td>NHEK</td>
<td>0.9942 (0.9979,0.9981)</td>
<td>0.9394 (0.9982,0.9939)</td>
<td>0.4485 (0.9892,0.9406)</td>
<td>0.2897 (0.9899,0.8969)</td>
<td>0.9926 (0.9883,0.9887)</td>
</tr>
</tbody>
</table>

The scores for all pairs within 2.5kb to 2Mb are shown as the first value. The scores for balanced and unbalanced data are shown in parentheses.

We hypothesized that the low performance in GM12878 was likely due to the fact that the Hi-C sequencing depth was much higher in GM12878 than in other samples. In other words, the quality of the EP-pairs in other samples was different from that in GM12878. To test this hypothesis, we applied the same EPiP model trained on the five other samples based on the cutoffs 30 and 5 to test condition-specific EP-pairs defined with the cutoff 100 in GM12878. We found that EPiP correctly predicted 2396 (78.69%) of 3045 condition-specific EP-pairs in GM12878 (Table S7). Therefore, EPiP indeed can reliably predict condition-specific EP-pairs in new samples, with an accuracy of 91.00% (7894 out of 8675 condition-specific EP-pairs in seven samples except for GM12878). EPiP predicted only 31.77% of condition-specific EP-pairs in GM12878 (Table 2).

We compared EPiP with two recently published methods, TargetFinder and Ripple. We compared them on the TargetFinder data and the EPiP all EP-pair test data within 2 kb to 2 Mb. EPiP showed better performance than TargetFinder and Ripple (Table 3). First, we compared EPiP with TargetFinder and Ripple on the data from TargetFinder (Tables 3 and S10). For the six samples in TargetFinder data (GM12878, HeLa, HUVEC, IMR90, K562 and NHEK), on average, EPiP had an AUROC, AUPR, F1, precision, recall and specificity of 0.95, 0.84, 0.96, 0.09, 0.45 and 1.00, respectively. TargetFinder had an AUROC, AUPR, F1, precision, recall and specificity of 0.92, 0.59, 0.50, 0.72, 0.39 and 0.99 respectively (Table 3). Ripple had an AUROC, AUPR, F1, precision, recall and specificity of 0.75, 0.19, 0.02, 0.75, 0.01 and 1.00 respectively (Table 3). The poor performance of Ripple may be due to the fact that Ripple could not deal with unbalanced data well, although the real data are always balanced in practice.

Next, we compared EPiP with TargetFinder and Ripple on all EP-pairs test data within 2.5 kb to 2 Mb (Tables 3 and S11). For the five samples shared with the TargetFinder study (GM12878, HUVEC, IMR90, K562, NHEK), on average, EPiP had an AUROC, AUPR, F1, precision, recall and specificity of 1.00, 0.98, 0.99, 0.99, 1.00 and 1.00, respectively. Based
A. Talukder et al.

Fig. 3. The overall performance of EPIP on external datasets.

Table 2. The performance of cell specific EPIP models on all pairs within 2.5kbp to 2Mb.

<table>
<thead>
<tr>
<th>Cell line</th>
<th>AUROC</th>
<th>AUPR</th>
<th>F1</th>
<th>Precision</th>
<th>Sensitivity/Recall</th>
<th># of cell specific EPIs</th>
<th>% of predicted cell specific EPIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>GM12878</td>
<td>0.9873</td>
<td>0.9385</td>
<td>0.9385</td>
<td>0.9946</td>
<td>0.8885</td>
<td>58765</td>
<td>0.9975</td>
</tr>
<tr>
<td>HMEC</td>
<td>0.9951</td>
<td>0.9905</td>
<td>0.7664</td>
<td>0.6256</td>
<td>0.9889</td>
<td>42</td>
<td>0.8333</td>
</tr>
<tr>
<td>HUVEC</td>
<td>0.9920</td>
<td>0.9816</td>
<td>0.6955</td>
<td>0.5387</td>
<td>0.9810</td>
<td>73</td>
<td>0.8767</td>
</tr>
<tr>
<td>IMR90</td>
<td>0.9979</td>
<td>0.9960</td>
<td>0.8714</td>
<td>0.9938</td>
<td>0.9960</td>
<td>2781</td>
<td>0.9960</td>
</tr>
<tr>
<td>K562</td>
<td>0.9955</td>
<td>0.9903</td>
<td>0.8421</td>
<td>0.7377</td>
<td>0.9881</td>
<td>1178</td>
<td>0.9677</td>
</tr>
<tr>
<td>K562</td>
<td>0.9892</td>
<td>0.9826</td>
<td>0.8431</td>
<td>0.7377</td>
<td>0.9836</td>
<td>2004</td>
<td>0.9880</td>
</tr>
<tr>
<td>NHEK</td>
<td>0.9942</td>
<td>0.9791</td>
<td>0.7675</td>
<td>0.6257</td>
<td>0.9879</td>
<td>102</td>
<td>0.8235</td>
</tr>
</tbody>
</table>

The number and percentage of cell specific EP-pairs predicted are also shown in the last two columns.

The overall performance of EPIP on external datasets.

Table 2. The performance of cell specific EPIP models on all pairs within 2.5kbp to 2Mb.

<table>
<thead>
<tr>
<th>Cell line</th>
<th>AUROC</th>
<th>AUPR</th>
<th>F1</th>
<th>Precision</th>
<th>Sensitivity/Recall</th>
<th># of cell specific EPIs</th>
<th>% of predicted cell specific EPIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>GM12878</td>
<td>0.9873</td>
<td>0.9385</td>
<td>0.9385</td>
<td>0.9946</td>
<td>0.8885</td>
<td>58765</td>
<td>0.9975</td>
</tr>
<tr>
<td>HMEC</td>
<td>0.9951</td>
<td>0.9905</td>
<td>0.7664</td>
<td>0.6256</td>
<td>0.9889</td>
<td>42</td>
<td>0.8333</td>
</tr>
<tr>
<td>HUVEC</td>
<td>0.9920</td>
<td>0.9816</td>
<td>0.6955</td>
<td>0.5387</td>
<td>0.9810</td>
<td>73</td>
<td>0.8767</td>
</tr>
<tr>
<td>IMR90</td>
<td>0.9979</td>
<td>0.9960</td>
<td>0.8714</td>
<td>0.9938</td>
<td>0.9960</td>
<td>2781</td>
<td>0.9960</td>
</tr>
<tr>
<td>K562</td>
<td>0.9955</td>
<td>0.9903</td>
<td>0.8421</td>
<td>0.7377</td>
<td>0.9881</td>
<td>1178</td>
<td>0.9677</td>
</tr>
<tr>
<td>K562</td>
<td>0.9892</td>
<td>0.9826</td>
<td>0.8431</td>
<td>0.7377</td>
<td>0.9836</td>
<td>2004</td>
<td>0.9880</td>
</tr>
<tr>
<td>NHEK</td>
<td>0.9942</td>
<td>0.9791</td>
<td>0.7675</td>
<td>0.6257</td>
<td>0.9879</td>
<td>102</td>
<td>0.8235</td>
</tr>
</tbody>
</table>

The number and percentage of cell specific EP-pairs predicted are also shown in the last two columns.

4 Discussion

Identifying EPIs is important for the study of gene transcriptional regulation. Although several computational methods are available to predict EPIs, they often cannot predict condition-specific EPIs and their performance is still not satisfactory. We thus developed a computational method, EPIP, to learn the patterns of EPIs and to predict condition-specific EPIs. We demonstrated that, on average, EPIP correctly predicts 99.99% of condition-specific EPIs in five common samples shared with the TargetFinder study. TargetFinder predicted only 83.91% of these condition-specific EPI-pairs (Table S11). In the two common samples shared with the Ripple study, EPIP predicted 99.99% of the condition-specific EP-pairs, while Ripple only could predict 27.07% of them (Table S11).

Finally, we compared EPIP with TargetFinder and Ripple on condition-specific EPIs in TargetFinder data. EPIP predicted 51.36% of the 8471 condition-specific EP-pairs in the six samples, while TargetFinder predicted 38.5% of the 8471 condition-specific EP-pairs (Table S10). Ripple predicted only 0.53% of the 5787 condition-specific EP-pairs in the three samples shared by the Ripple study and the TargetFinder study (GM12878, HeLa and K562), while EPIP predicted 54.42% of the same 5787 EP-pairs (Table S10). The accuracy of EPIP on condition-specific EP-pairs here was much lower compared with that on EPPI test data, which may be because the TargetFinder data were not in good quality. For instance, enhancers and promoters used by TargetFinder were from computational predictions (Ernst et al., 2012; Hoffman et al., 2012), which were prone to errors. Moreover, almost 50% of their enhancers and promoters overlap with their promoters and enhancers, respectively. In addition, the negative EP-pairs in TargetFinder data were problematic. TargetFinder labeled an EP-pair “negative”, if it did not overlap with Rao et al. looplists of any resolution. Due to the limited sequencing resolution and the limitation of the algorithms to analyze raw Hi-C reads to generate looplists, EP-pairs not identified as looplists are not necessarily negative pairs (Forcato et al., 2017).

Forcato et al. (2017).
learning approach in EPIP provides the opportunity to efficiently train the model when new data become available. EPIP is trained with different samples. This means that data from different samples are fed to the training model in a specific order. To investigate whether the order of the samples in training has an impact on the performance of EPIP, we considered HUVEC as the testing sample and trained the EPIP model on the remaining six samples in all possible 120 orders. We observed that the order of the samples used in training EPIP does not significantly impact the final performance, as the standard deviation of the AUROC and the F1 score was 0.001 and 0.002, respectively, for all 120 different orders of training in these experiments.

We used FANTOM enhancers to define EP-pairs in this study. The number of FANTOM enhancers is small compared with the known and predicted enhancers in various studies (Ernst et al., 2012). However, FANTOM enhancers arguably represent the largest set of enhancers we have so far that are defined with the same criteria and supported by experiments. We further overlapped FANTOM enhancers with ChromHMM enhancers and H3K27ac ChIP-seq peaks to define active enhancers, which is likely to generate more reliable enhancers and more reliable training data. However, the choice of the FANTOM enhancers may have prevented us from testing EPIP more generally, since we only tested EPIP on EP-pairs based on FANTOM enhancers. When there is a larger and more reliable set of experimentally determined enhancers available in the future, it is necessary to test EPIP on the EP-pairs based on the new set of enhancers to make sure that it performs similarly.

We tried to train EPIP on EP-pairs from Rao et al. looplists, which generated suboptimal models due to the much smaller size of training data (Tables S12 and S13). Instead, we used the cutoffs 30 and 5 to define positive and negative EP-pairs. We selected this combination of cutoffs based on our testing with different cutoffs and our previous studies (Li et al., 2016; Zhao et al., 2016). Note that these EP-pairs defined with the cutoffs 30 and 5 (Tables S12 and S13). Although EPIP has better performance compared with the state-of-the-art methods, there is still room for improvement. For instance, different experimental methods are available to identify EPs, such as Hi-C, ChIA-PET, and 5C. We used Hi-C for extracting training data. It is worth studying how the performance of EPIP improves if we train EPIP with EPs from other sources together with Hi-C. Moreover, we used chromatin loops extracted by Rao et al. to define significant interactions. Developing a method that is able to extract interactions from raw Hi-C data may help to improve the performance of EPIP. In addition, like all most all existing methods, EPIP considers one EP-pair at a time to predict interacting EP-pairs while considering multiple EP-pairs may help the discovery of true positive EP-pairs, as shown in a previous study (Zhao et al., 2016). In the future, we will work on these directions together with others to further improve the accuracy of the EPI prediction

### Funding

This work has been supported by the National Science Foundation [grants 1356524, 1661414 and 1149955] and the National Institute of Health [grant R15GM123407]. Funding for open access charge: The National Science Foundation grant 1149955.

### References


