Mutation-Convolution-Max Layers Enhance Deep Learning of DNA Motifs

Abstract

Convolutional neural networks are widely used in many applications including functional genomics. Convolutional filters are capable of capturing sequence motifs (e.g. TF binding motifs) while tolerating some substitutions. However, standard convolutional filters do not allow insertion/deletion (indels); indeed any *shift* in a part of the motif can completely alter the activation of the filter. This can be a severe limitation because many biological functions are approximately preserved by short indels and we would like the architecture to respect this invariance. We propose an alternative architecture, the mutation-convolution-max (MCM) layer, which flexibly models the phenomenon that sequences that are within a short edit distance from one another, allowing for indels, may have similar biological performance. We propose a theoretical metric, the *informedness*, by which we are able to evaluate and compare convolution and MCM layers. On synthetic experiments, the MCM layer outperforms the convolution layers in settings where the training set is similar to the test set, and also in domain-adaptation settings where the test set consists of more mutations than training set. While conv-nets are designed to capture Hamming distance properties of the input data, MCM is a new framework capable of modeling the more complex edit distance and could have many applications in genomics and other domains.

1. Introduction: Why Are Convolutional Nets Not Enough?

In recent years, a variety of deep learning-based algorithms have been proposed to analyze and infer genomic regulation from high-throughput, feature-rich datasets (Park and Kellis, 2015; Quang and Xie, 2016; Angermueller et al., 2016). In particular, convolutional neural networks (CNNs) have gained popularity as a tool to predict the relationship between regulatory proteins and their target DNA or RNA molecules.

CNNs are a natural choice for the architecture of neural networks for regulatory genomics because they consist of sets of weights, referred to as filters, that are convolved across the input sequences, effectively serving as spatially-invariant "motif detectors" along the input DNA sequence. The output of a convolution filter is a continuous value that is a function of the dot product between the weights and parts of the input sequence, encoded as a one-hot vector (see Alipanahi et al. (2015) for example). This continuous output permits slight mismatches (i.e. a small number of substitutions) between the set of weights learned by the network and the nucleotide bases appearing in the input DNA sequence. This is useful because biological function is often preserved under small differences in sequence, and thus input DNA sequences that are within a small *Hamming distance* (Geisler, 2005) from each other might be presumed to have similar function.

However, in many genomic settings the relevant biological functions are more closely related to the *edit distance* between input sequences (Robertson et al., 2006; Kankainen et al., 2012) rather than Hamming distance. This is because short insertions or mutations, may not significantly affect the binding of a regulator protein. A single insertion or deletion only increases the edit distance between the original and mutated sequences by 1, but can significantly affect the Hamming distance between the sequences. This is illustrated in Fig. 1. As a result, CNNs that search for motifs that bind to regulatory proteins often require a large number of independent filters (each with independent parameters that must be trained) to detect sequences that simply mutated versions of one another.

In this paper, we propose a variation of the standard convolution layer, which we term the ***mutation-convolution-max*** (MCM) layer that uses a local data augmentation approach to search for nucleotide sequences that differ up to an insertion or deletion. By sharing parameters between similar sequences explicitly, we are better able to detect motifs. To analyze this idea, we propose the notion of ***layer informedness***, which allows us to compare the effectiveness of different kinds of

*Equal contribution and listed alphabetically . **AUTHORERR: Missing \icmlcorrespondingauthor.**

layers in a neural network. We then compare the effectiveness of neural networks with standard convolution layers as well as MCM layers on synthetic data. We find that the MCM layer outperforms the convolution layer in settings where the training set is similar to the test set, and also in domain-adaptation settings where the test set consists of more mutations than training set. A general limitation of convolution filters is the implicit assumption that the Hamming distance is the relevant metric of similarity. We have developed an alternative architecture, the MCM layer, which learns edit distance-based similarities, which may have broad applications in deep learning.

2. Method: The Mutation-Convolution-Max (MCM) Layer

The classical convolution filter is typically applied in the early layers of a neural network, to detect repeated motifs in the raw DNA sequence. In this sense, each convolution filter can be thought of as a "motif-detector" that is trained to maximally *activate* when it is multiplied against the part of the input sequence containing the motif. However, convolutional filters do not explicitly model the fact that biological systems could be somewhat insensitive to a small number insertions or deletions, and it is possible in principle that a frame-shift mutation can significantly affect the output activation of a convolution filter.

This leads to two problems: one during the training of convolution networks and the other during test time. During training, because each filter only learns an individual set of weights, there may need to be many channels to effectively learn the weights of different shifted versions of a motif, even though they are all similar to each other. This leads to an unnecessarily large number of parameters and the requirement that there be enough training samples consisting of each specific mutuation to be learned. An even bigger problem can occur during test time: if the test dataset has significantly more (or new kinds of) mutations than the training set, standard convolution filters will suffer from domain adaptation issues, because the structure of the CNN is not flexible enough to detect shifted versions of the motif after the weights of the filters have been trained.

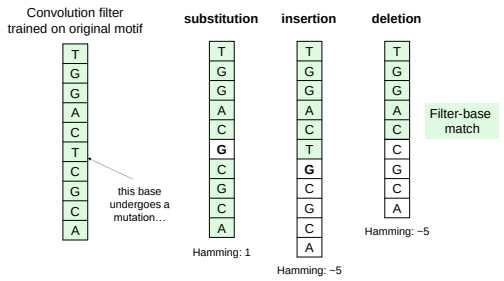
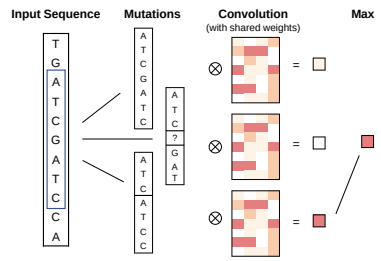


Figure 1. Here, we show that convolution filter that is trained to detect a particular motif (shown on the left) will be mostly activated even when the motif undergoes a substitution mutation, since the Hamming distance between the original and mutated sequences is just 1. But if an insertion or deletion occurs within the motif, the filter is no longer aligned with part of the input sequence, leading to a significantly reduced activation. The activation is most reduced if the insertion or deletion occurs near the middle of the motif, as the convolution "slides" along the input, finding the largest contiguous portion of the input that matches the filter.

To overcome these problems, we introduce the notion of a mutation-convolution-max (MCM) layer. The MCM layer, illustrated schematically in Fig. 2(a), performs three operations as it "slides" across the input sequence, in a manner similar to the standard convolution filter. First, for each input within the window of the filter, the layer first generates "mutated" versions of the input by (i) inserting a random base at the center of the input, (ii) deleting the base at the center of the input, or (iii) doing nothing. Other mutations can be performed, but as we show in Section 3, there is a tradeoff to performing each mutation. Then, as in a convolutional filter, a set of shared linear weights is convolved with each of the mutated versions of the input. The maximum value of the convolution is then passed through an non-linear activation function, and passed onto the neurons in the next convolution or fully-connected layer.

The purpose of the MCM layer is to produce a non-linear transformation that is not only spatially invariant[1] (as is the traditional convolution layer), but insensitive to small changes in the edit distance of the input. This is illustrated in the table Fig. 2(b), where we show how a series of mutations that are all within 1 edit distance of a true motif can have significantly different activations from a convolution filter, but similar activations from an MCM layer. The MCM layer, without introducing any additional parameters, activates for a wider range of input sequences, and thus more closely reflect the functional significance of these sequences.

[1]Technically, convolution layers (and MCM layers) are spatially *covariant*, as translations in the input are carried over to the output, but a convolution/MCM layer followed by a pooling layer enforces a degree of spatial invariance.



(a)

Filter matches...	Convolution filter		MCM filter	
ATCGATC	?TCGATC	ATCGTC?	?TCGATC	ATCGTC?
	A?CGATC	ATCGAC?	A?CGATC	ATCGAC?
	AT?GATC	ATCGAT?	AT?GATC	ATCGAT?
	ATC?ATC	?TCGATC	ATC?ATC	?TCGATC
	ATCG?TC	A?TCGAT	ATCG?TC	A?TCGAT
	ATCGA?C	AT?CGAT	ATCGA?C	AT?CGAT
	ATCGAT?	ATC?GAT	ATCGAT?	ATC?GAT
	TCGATC?	ATCG?AT	TCGATC?	ATCG?AT
	ACGATC?	ATCGA?T	ACGATC?	ATCGA?T
	ATGATC?	ATCGAT?	ATGATC?	ATCGAT?
	ATCATC?		ATCATC?	

Filter-base match: strong, weak

(b)

Figure 2. (a) A schematic of the mutation-convolution-max (MCM) layer. As the MCM layer slides across the input, it generates mutated versions of the input, convolves them with shared weights, and then takes the maximum resulting product. This operation is differentiable, allowing the shared weights to be learned by standard backpropagation. (b) A table of motifs that are 1 edit away from a base motif (which the filter is trained to recognize). The middle column shows that a convolution filter that is trained to recognize the base motif will not activate for certain sequences that are generated by insertions or deletions near the center of the base motif. However, as shown in the rightmost column, all of these sequences activate an MCM layer trained to identify the same motif.

3. Theoretical Analysis: *Informedness* of the Layer

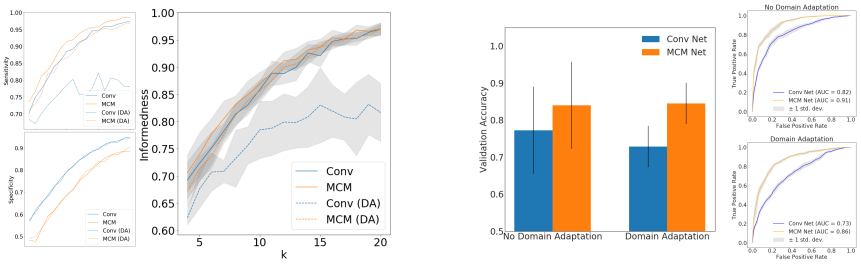
In Section 2, we demonstrated that the MCM layer can detect a larger set of *biologically-relevant* input sequences than the standard convolution layer. However, we would also expect that the MCM layer detects sequences which are *not* biologically-relevant, simply because the layer is activated by more sequences. How do we quantify this trade-off between the increased sensitivity of the layer and the decreased specificity? To understand this trade-off in isolation, we introduce in this paper the notion of ***layer informedness***. The layer informedness measures how the set of *positive* input sequences, those that should lead to a positive binding prediction (i.e. $X_p \equiv \{x_i | y_i = 1\}$ for input features x_i and labels y_i) activate the neurons in the layer, as compared to the set of *negative* input sequences, $X_n \equiv \{x_i | y_i = 0\}$. In particular, if the layer is a parametrized function f_θ that maps an input x to a real number in $[0, 1]$ (e.g. a convolution/MCM layer followed by a sigmoid), then we define the layer informedness as:

$$I_\alpha(f, \theta, X_p, X_n) = \arg\max_\theta \underbrace{\prod_{x:x\in X_p} f_\theta(x)^{\frac{\alpha}{|X_p|}}}_{\text{sensitivity}} \cdot \underbrace{\prod_{x:x\in X_n} (1 - f_\theta(x))^{\frac{1-\alpha}{|X_n|}}}_{\text{specificity}} \tag{1}$$

This expression consists of two terms, which can be interpreted as the *sensitivity* and *specificity* respectively. The first term captures how well positive samples activate the layer, and the second term measures how well negative samples do *not* activate the layer. The parameter α quantifies the relative importance of sensitivity and specificity in a particular analysis. Although this expression should not be taken to be an exact measure of the discriminative power of the layer (since the output of the layer gets affected in all sorts of non-linear ways by further layers), the informedness is an intuitive measure of the tradeoff between different layer architectures that can be optimized in isolation from the rest of the network.

We can estimate the informedness of a layer quite easily by generating a set of positive examples $\{X_p\}$ and negative examples $\{X_n\}$ then running gradient descent to optimize the parameters of the layer, θ. We can also simulate the scenario of domain adaptation by optimizing the parameters over one set of examples, but evaluating the expression in (1) on a different set of examples. In Fig. 3(a), we plot, as a function of the motif length k, the sensitivity, specificity, and informedness of a standard convolution layer and an MCM layer when the X_p consists of $3k$ k-mers, a third of which are identical copies of a base motif, a third are the result of random insertions, and a third the result of random deletions that are all within 1 edit of the base motif. For the negative set, we randomly sample 5,000 motifs that are at least 2 edits from the base motif. For the domain adapation scenario, we evaluate the parameters over the same training set, but optimize on a positive set that consists only of identical copies of a base motif. X_n is unchanged. For the informedness, we choose $\alpha = 0.9$ (i.e. it is quite important to detect most of the positive sequences, even if it involves detecting some false positives).

We observe that in the case where the training and evaluation datasets are the same, the sensitivity of MCM layer is higher than that of the convolution layer, but the specificity is worse, because the MCM layer detects false positives: those k-mers



(a) (b)

Figure 3. (a) The sensitivity (top left), specificity (bottom left), and informedness (right) of the convolution layer and the MCM layer are compared for the settings with and without domain adaptation (DA). These results are averaged over 10 trials, and the gray shaded area on the right panel reflects plus/minus 1 standard deviation. (b) Here, we compare the accuracy of two neural networks, one with a convolution layer and the other with a MCM layer. The accuracies of the networks are averaged compared for the settings with and without DA (left). More detailed information can be found by examining the ROC curves for each classifier (right) on a particular synthetic dataset: the curves on the top represent the setting without DA, while the curves on the bottom represent the setting with DA. Gray areas represent plus/minus 1 stadard deviation bounds, generated by bootstrapping the evaluation examples.

that are more than 1 edit away from the original motif, but nevertheless activate the MCM layer. On the other hand, the sensitivity of the MCM layer is significantly higher than the sensitivity of the convolution layer in the case of domain adaptation. These results are combined in the plot on Fig. 3(a, right), where the informedness is plotted. We see that the informedness of the MCM layers is at least as good as that of the convolution layers.

4. Results: Synthetic Experiments

Armed with some intuition for the MCM layer, we proceed to design a neural network that can allow us to directly compare the MCM layer and standard convolution layer. We design a 2-layer neural network where the first layer is either the convolution *or* the MCM layer with 4 filters (followed by a max pooling with a window size of 2), and the second layer is a 256-neuron fully-connected layer (with 0.95 dropout) that is followed by a 2-class softmax layer. Each neural network is fed 1,000 training examples, each example being 100 nucleotide bases long, with roughly half containing a (possibly-mutated) 16-base motif at a random position. The composition of the datasets is the same as in section 3: in the first dataset, a third of the positive sequences in both the training set and test set include copies of the base motif and the rest include mutated copies of the motif (equally likely to be random insertions or deletions) that are all within 1-edit distance of the base motif. In the second dataset, where we simulate domain adaptation, the test set is the same, but the positive sequences in the training set include only the base motif.

The results are shown in Fig. 3(b). On the left, we show the validation accuracy averaged over 10 randomly generated datasets. We find that the accuracy of the MCM network is higher than that of the conv-net, both in the setting where the training set is similar to the test set, and also in domain-adaptation setting. By plotting the ROC curves for one trial, we see that the network with the MCM layer is able to achieve higher true positive rates, as well as lower false positive rates.

5. Conclusion

We have proposed a new architecture, the mutation-convolution-max (MCM) layer, which can be incorporated into existing deep learning models for functional genomics. The MCM layer explicitly models the fact that DNA motifs that have undergone a single insertion or deletion likely bind to the same regulatory proteins as those that have not undergone the mutation. Our theoretical analysis of the informedness as well as empirical results on synthetic experiments show that the MCM layer outperforms the convolution layer, particularly in settings where the test set includes significantly more mutations than the training set. As next steps, we will be testing MCM nets on large-scale genomics datasets, where we aim to identify conditions the MCM layers offer improved ability of neural networks to learn the regulatory code of the genome. While we have proposed MCM as an algorithm for functional genomics, it is a generalization of the standard convolution layer to any setting where the edit distance represents the relevant similarity metric, and thus may have broader applicability beyond genomics.

References

Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015.

Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. Deep learning for computational biology. *Molecular systems biology*, 12(7):878, 2016.

Jonathan Geisler. Error detection & correction. *Taylor University, Computer & System Sciences Dept*, 2005.

Matti Kankainen, Teija Ojala, and Liisa Holm. Blannotator: enhanced homology-based function prediction of bacterial proteins. *BMC bioinformatics*, 13(1):33, 2012.

Yongjin Park and Manolis Kellis. Deep learning for regulatory genomics. *Nature biotechnology*, 33(8):825–826, 2015.

Daniel Quang and Xiaohui Xie. Danq: a hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic acids research*, 44(11):e107–e107, 2016.

Gordon Robertson et al. cisred: a database system for genome-scale computational discovery of regulatory elements. *Nucleic Acids Research*, 34(suppl$_1$) : $D68 - -D73, 2006$.