FILTER SHAPING FOR CONVOLUTIONAL NEURAL NETWORKS

Xingyi Li, Fuxin Li, Xiaoli Fern & Raviv Raich
School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, OR 97331, USA
{lixin,lif,xfern,raich}@eecs.oregonstate.edu

ABSTRACT

Convolutional neural networks (CNNs) are powerful tools for classification of visual inputs. An important property of CNN is its restriction to local connections and sharing of local weights among different locations. In this paper, we consider the definition of appropriate local neighborhoods in CNN.We provide a theoretical analysis that justifies the traditional square filter used in CNN for analyzing natural images. The analysis also provides a principle for designing customized filter shapes for application domains that do not resemble natural images. We propose an approach that automatically designs multiple layers of different customized filter shapes by repeatedly solving lasso problems. It is applied to customize the filter shape for both bioacoustic applications and gene sequence analysis applications. In those domains with small sample sizes we demonstrate that the customized filters achieve superior classification accuracy, improved convergence behavior in training and reduced sensitivity to hyperparameters.

1 INTRODUCTION

In recent years, deep learning approaches have seen tremendous successes. They have redefined the state-of-the-art for a variety of challenging domains including computer vision and natural language processing. A particularly appealing property of deep learning is that it eliminates the need for feature engineering on complex inputs such as natural images and raw texts. For the task of image classification, deep convolutional neural networks (CNN) have rapidly become the standard approach. CNN builds upon many sequential convolutional and pooling layers. They are capable of automatically extracting highly discriminative features from raw pixel inputs without prior domain knowledge, leading to impressive performances even surpassing that of humans on the challenging ImageNet dataset (He et al., 2016). One of the key reasons for the success of CNNs is that the network is only connected locally. Traditional CNN designs use small square filters, which have been quite successful in analyzing and classifying natural images. Also, translation invariance in natural images allow the local filter weights to be shared among image locations. These two constructs greatly reduce the number of parameters and make the learning process easier.

The success of CNN inspires us to consider the application of CNN to other domains that do not necessarily resemble natural images, and do not have millions of labeled training examples to learn from. Such domains are abundant in natural and biomedical sciences, where the cost of collecting and annotating data can be high and small-sample performance is important. In order to extend CNN to such domains, we start by examining the definition of locality and the use of square filters in CNN. We introduce a novel correlation analysis methodology that analyzes the cross-correlation between neighboring pixels across the whole dataset. A theoretical justification based on Gaussian complexities is provided to demonstrate why such statistics can lead to a natural design methodology of filter shapes for CNN. When applied to natural images, it justifies the use of square filters in image classification tasks.

We believe that multiple layers in a deep network do not have to use the same filter shapes since they may capture different types of information. Spectrograms present one such example, where local filters tend to capture variations within one frequency region, while global filters could capture the

relationships between different harmonics and syllables (each syllable indicates a single utterance of a bird). We propose a data-driven filter design algorithm that automatically derives multiple levels of different convolutional filter shapes based on the correlation analysis of the data. We believe that the generalization capability of CNNs, especially on small training samples, would improve if those customized data-dependent filter designs are adopted rather than standard square filters, since they capture more of the correlation structure in the data.

We evaluate our proposed method in two diverse domains, bioacoustic spectrograms and gene sequence data. Empirical results suggest that by customizing the filter designs, the proposed approach not only significantly outperforms traditional CNN approaches in classification accuracy, but also demonstrates superior robustness to hyperparameter tuning, especially with a small training set size.

While we have only considered two application domains, we expect the contribution to go beyond these specific application domains. In particular, the proposed method is generally applicable to a variety of other data analysis applications where data is scarce and demonstrate spatial autocorrelation patterns that differ from natural images.

2 CORRELATION ANALYSIS

2.1 THE CORRELATION ANALYSIS METHODOLOGY AND NATURAL IMAGES

We focus on examining the use of locality in CNN. It is well-known that locality makes strong intuitive sense in natural images – the computer vision domain is dependent on building from low-level to high-level features, where the low-level features, such as edges and corners, can be determined within a local neighborhood. Intuitively, local patterns such as edges and corners are discriminative even in a 3×3 box in the image, because normally, the values between each pixel and its neighboring pixels are highly correlated. If a pixel is white, it is very likely that all the pixels in its 8-neighborhood are also white. This makes boundaries significant patterns as some of the pixels behave significantly differently from such a strong prior.

In other words, we could think of such strong correlation as a natural description of a neighborhood: a CNN neighborhood can be defined as the pixels that have the highest correlations with each pixel. To capture this intuition, we introduce the following *correlation analysis* procedure:

Let $\mathcal{X} = [X_1, \ldots, X_N]$ be a dataset with N examples where the dimensions of each $X_i, i = 1, \ldots, N$ has an integer lattice structure, i.e., $X_i = \{X_i(\mathbf{k}), \mathbf{k} \in \mathbb{Z}^p\}$ where $\mathbf{k}$ is a p-dimensional index vector. Let $X_i[\mathbf{k}]$ denote X_i shifted $\mathbf{k}$ indices, i.e. $X_i[\mathbf{k_1}] = \{X_i(\mathbf{k} + \mathbf{k_1}), \mathbf{k} \in \mathbb{Z}^p\}$ then

$$\mathbf{C}(\mathbf{k}) = \frac{\sum_{i=1}^{N} cov(X_i[\mathbf{k}], X_i)}{\sum_{i=1}^{N} cov(X_i, X_i)} \tag{1}$$

where $cov(A, B) = \sum_{\mathbf{i} \in \mathbb{Z}^p} (A(\mathbf{i}) - \mu_A)(B(\mathbf{i}) - \mu_B)$ is the covariance function, and μ_A, μ_B is the average value across the corresponding dataset. In practice, we collect the tensor only with $\mathbf{k} \in R^p \subset \mathbb{Z}^p$ with R being a short range (e.g. $\{-50, 50\}$). We also exclude data that represent obvious noise in the computation, such as background white noise in the spectrogram data. These white noise do not contain signals yet show strong correlation within local regions, hence can bias $\mathbf{C}$ towards a regular ball shape.

Fig. 1 shows the results of such correlation analysis on $1,000$ images from the PASCAL VOC dataset (Everingham et al.). It can be seen that each pixel is correlated the most with the pixels in its 4-neighborhood, followed by ones in its 8-neighborhood and bigger. This corresponds well with the common filter designs in natural images such as 3×3 (Simonyan & Zisserman, 2014), 5×5 (Szegedy et al., 2014) and 7×7 (Krizhevsky et al., 2012).

2.2 THEORETICAL JUSTIFICATION OF THE CORRELATION ANALYSIS APPROACH

The insight we developed in the previous section has justifications from machine learning theory. The capability of generalization from neural networks can be characterized using Rademacher complexities (Bartlett & Mendelson, 2002):

Theorem 1. *(Theorem 18, (Bartlett & Mendelson, 2002)) Suppose that $\sigma : \mathbb{R} \mapsto [-1, 1]$ has Lipschitz constant L and satisfies $\sigma(0) = 0$. Define the class computed by a two-layer neural network with 1-norm weight constraints as:*

$$F = \left\{ \mathbf{x} \mapsto \sum_i v_i \sigma(\mathbf{w}_i \cdot \mathbf{x}) : \|\mathbf{v}\|_1 \leq 1, \|\mathbf{w}_i\|_1 \leq B \right\}. \tag{2}$$

For any $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$, we have

$$\hat{G}_N(F) \leq \frac{cLB(\ln d)^{1/2}}{N} \max_{j,j'} \sqrt{\sum_{i=1}^{N} (x_{ij} - x_{ij'})^2}, \tag{3}$$

where $\hat{G}_N(F) = \mathbb{E}\left[\sup_{f \in F} \frac{2}{N} \sum_{i=1}^{N} g_i f(x_i)\right]$ is the empirical Gaussian complexity of the function class F, with g_i being independent standard normal random variables. The Gaussian complexity (and the closely-related Rademacher complexity) measures the capacity of the function class and is linked to the generalization capability of the learner. The learner from a function class F with a smaller Gaussian complexity is potentially able to generalize better (Bartlett & Mendelson, 2002), but has less capacity to fit the training data well. Thus a trade-off is required for practical learners.

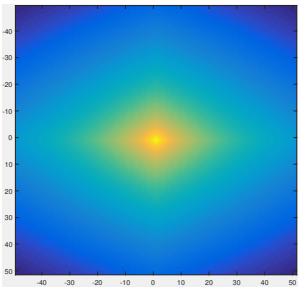


Figure 1: Covariance analysis on natural images

Convolutional neural networks are neural networks with locally-defined connections hence similar results as above hold for CNNs, with the change that $\max_{j,j'}$ is now defined within the range of each single filter, as well as a change from the L_1 norm to the L_2 norm:

Theorem 2. *Suppose that $\sigma : \mathbb{R} \mapsto \mathbb{R}$ is a contraction mapping. Define the class computed by a two-layer convolutional neural network with one convolutional layer and one fully-connected layer with 2-norm weight constraints as:*

$$F = \left\{ \mathbf{x} \mapsto \sum_i v_i \sigma(\mathbf{w}_i * \mathbf{x}) : \|\mathbf{v}\|_2^2 \leq 1, \|\mathbf{w}\|_1 \leq B \right\}. \tag{4}$$

For any $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$, we have

$$\hat{G}_N(F) \leq \frac{cB(\ln d)^{1/2}}{N} \max_{\mathbf{j}-\mathbf{j}' \in \mathcal{N}} \sqrt{\sum_{i=1}^{N} \|\mathbf{x}_i(\mathbf{j}) - \mathbf{x}_i(\mathbf{j}')\|^2}, \tag{5}$$

where $\mathcal{N} \subset \mathbb{Z}^p$ defines the shape of the convolution filter, i.e.

$$(\mathbf{w}_i * \mathbf{x})(\mathbf{k}) = \sum_{\mathbf{j} \in \mathcal{N}} \mathbf{w}_{i,\mathbf{j}} \mathbf{x}[\mathbf{k}](\mathbf{j}) \tag{6}$$

The proof can be found in the appendix.

This result suggests that in order to minimize $\hat{G}_N(F)$ of the learner, one needs to select a convolutional filter that minimizes $\max_{\mathbf{j} \in \mathcal{N}, \mathbf{j}' \in \mathcal{N}} \sqrt{\sum_{i=1}^{N} \|\mathbf{x}_i(\mathbf{j}) - \mathbf{x}_i(\mathbf{j}')\|^2}$. This requires to select only dimensions that are highly correlated to each other (ones that maximize $cov(\mathbf{x}_i(\mathbf{j}), \mathbf{x}_i(\mathbf{j}'))$) and avoid having dimensions that are less correlated in the same convolutional filter. In natural images, this justifies selecting 3×3, 5×5 or 7×7 squares in natural images since those are neighborhoods that maximize the correlation within them.

3 Customize the Shape of CNN Filters

3.1 Filter Shaping Methodology

Our intuition for filter design is that the filters for a CNN framework should be capable of representing the correlation image obtained from the covariance analysis, hence trading-off between learner

capacity and its Gaussian complexity. If the CNN filter allows data dimensions with little correlation in the same filter, it will make the bound in (5) worse, hence increasing the Gaussian complexity and negatively impact generalization, especially in small-sample scenarios.

In one layer, the solution can be obtained via the following LASSO optimization problem:

$$\min_{f_1} \|\mathbf{C} - f_1 * \mathbf{C}_0\|_F^2 + \lambda ||f_1||_1 \tag{7}$$

where $\mathbf{C}$ is the correlation tensor computed from the analysis in (1), $\mathbf{C_0}$ is an initial correlation tensor with a 1 at the center location and 0s at all other locations, and λ is a nonnegative regularization parameter trading off the representation error with the sparsity of the filters.

The LASSO optimization will attempt to convolve the $\mathbf{C}_0$ tensor so that it is most similar to $\mathbf{C}$. With this setup, the solution would be selecting the locations with the highest correlations to the center location, i.e. finding the $\Delta\mathbf{j}$s maximizing $\sum_{i=1}^{N} \mathbf{x}_i(\mathbf{j} + \Delta\mathbf{j})\mathbf{x}_i(\mathbf{j})$, directly corresponding to the theorem. By controlling λ, one can control how many nonzeros will be in f_1. Since CNN is more appropriate than the unsupervised LASSO for learning the exact filter weights, we will not keep the weights learned by the LASSO. Instead, only the *shape* of f_1 will be used, in the sense that only the locations with values in f_1 that are above a threshold will be accepted in the shape of the filter, and the rest will have their weights fixed to 0.

With more than one layer, the theory is less clear about what would be the shape of the convolutional filter. We propose a simple approach that repeatedly uses the previous convolution result to reconstruct $\mathbf{C}$. We assume that the next layers CNN filter f_i convolve with $f_{i-1} * (f_{i-2} \dots f_1(\mathbf{C}_0))$ would be more similar to the correlation tensor. Hence, we can solve again the same LASSO problem to approximate the full correlation tensor. Repeatedly applying this would amount to solve:

$$\min_{f_i} \|\mathbf{C} - f_i * \mathbf{C}_{i-1}\|_F^2 + \lambda \|f_i\|_1 \tag{8}$$

where $\mathbf{C}_{i-1} = f_{i-1} * (f_{i-2} \dots f_1(\mathbf{C}_0))$ is the solution from all the previous filters. The filter shape is then determined by the same approach as before. With appropriate parameters, this methodology can fully recover multiple levels of 3×3 squared filters in natural images.

3.2 Filter Shaping for Discrete Data

The above theory and computation works for continuous data. For discrete data where the minus operator is not defined, we could use an analogous approach to (8) with group LASSO instead of LASSO. Suppose again the dimensions of each X_i has an integer lattice structure and each element comes from a categorical distribution, i.e. $X_i(\mathbf{k}) \in \{1, \dots, K\}, \mathbf{k} \in \mathbb{Z}^p$.
In this case, the discrete correlation tensor is defined as

$$C(A, B, \mathbf{k}) = \frac{\mathbb{E}_{i \in \{1,\dots,N\}, \mathbf{j} \in \mathbb{Z}^p}[\mathbb{I}(X_i[\mathbf{k}](\mathbf{j}) = A)\mathbb{I}(X_i(\mathbf{j}) = B)] - P(A)P(B)}{\sqrt{P(A)(1 - P(A))P(B)(1 - P(B))}} \tag{9}$$

where $P(A)$ is the probability of any entry in the dataset attaining a value of A.

After collecting a similar covariance tensor as in the continuous case, one can solve a group LASSO problem instead of a LASSO problem in order to handle the discrete entries in the covariance:

$$\min_{f_i} \|\mathbf{C} - f_i * \mathbf{C}_{i-1}\|_F^2 + \lambda \|f\|_{2,1} \tag{10}$$

where f_i is a $K \times K \times \mathbb{Z}^p$ filter, $\|f\|_{2,1} = \sum_{\mathbf{j} \in \mathbb{Z}^p} \sqrt{\sum_{A=1}^{K} \sum_{B=1}^{K} f(A, B, \mathbf{j})^2}$ regularizes on all the entries at position $\mathbf{j}$. The optimization allows the optimization to set all entries $f(A, B, \mathbf{j})$ to 0 for a certain $\mathbf{j}$, which is then equivalent with filter shaping in the continuous case. In the discrete case, the optimization starts with $C_0(A, B, \mathbf{0}) = 1, \forall A = B \in \{1, \dots, K\}$ and other entries equal to 0.

3.3 Accounting for Max-Pooling by Average Pooling

Max-pooling layers are essential to the performance of deep neural networks. Therefore, we need an approach to account for max-pooling layers in our filter shaping algorithm. However, directly

max-pooling on the solved correlation will destroy the symmetry of the $\mathbf{C}_i$ matrix. Therefore, as a compromise we utilize average pooling in $\mathbf{C}$ instead of max-pooling. Suppose a max-pooling layer is after f_i. We will resize both $\mathbf{C}_i$ and $\mathbf{C}$ according to the max-pooling size reduction, and populate the values with bilinear interpolation (average over all the items that max-pooling is supposed to consider). One potential justification of using average pooling instead of max is that, each small box considered by max pooling in each example may attain the maximal value at a separate location. One can assume that the chance of each location attaining the max is equal, hence the expectation of max-pooling will be average pooling to resize $\mathbf{C}_i$ and $\mathbf{C}$.

4 Problem Domains

4.1 Bio-Acoustics

One special problem domain we consider is bird bio-acoustics, where the images we analyze are spectrograms. A spectrogram is a 2-dimensional image where the x-axis indicates time and the y-axis indicates frequency. The brighter a "pixel" is, the higher the energy at this time and frequency. Spectrograms are commonly used for sound classification (Stowell & Plumbley, 2011) (Ranjard & Ross, 2008) (Lasseck, 2013). Fig. 2 shows an example spectrogram of a recording of hummingbird wingbeats. Our goal is to predict the species of the bird based on in-situ recording of wingbeats (for hummingbirds) or bird-songs (for song birds). Consider Fig. 2(a), we notice that the wingbeats are reflected as a collection of bright wavy lines at the low frequency range. The lines are regularly spaced, revealing a clear harmonic structure. From the figure, it is reasonable to conclude that the pixels that are horizontally closer to each other tend to be more similar when viewed on the local scale. On a global scale, there are strong correlations between harmonics even they are separated about $25Hz$ apart. Such correlation patterns appear to be highly different from natural images.

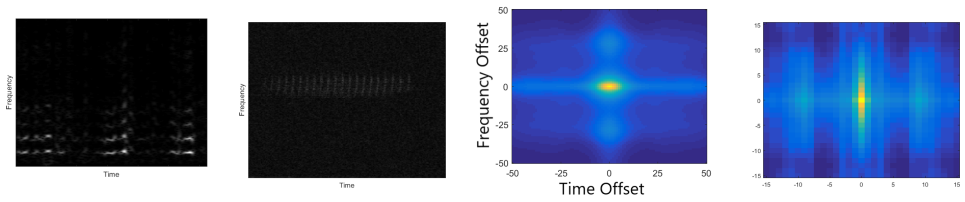


Figure 2: From left to right: (a) Sample spectrogram of bird wingbeats; (b) Sample spectrogram of bird song; (c) Correlation image of bird wingbeats; (d) Correlation image of bird song.

We performed the correlation analysis on wingbeat spectrograms and bird-song spectrograms separately based on formula 1. Fig. 2(c) and 2(d) are resultant correlation images for wingbeats and bird-song respectively. One can see significant differences w.r.t. the pattern in natural images.

4.2 Gene Sequence analysis: rSNP prediction

In this domain, our data are gene sequences, each describing a genetic variation in human genome called Single-Nucleotide Polymorphism (SNP). Human genome contains millions of SNPs and most of them reside in non-coding region of the genome and there is limited biological knowledge about their functionality. There is great interest in the biomedical community to use machine learning to help identify what non-coding SNPs are regulatory. Given a database of SNPs that have been annotated by experts based on prior biological knowledge as regulatory or non-regulatory, the goal is to learn a model to predict whether a SNP serves any regulatory function based on the gene sequence around it. Each SNP is represented as a 2001 base-pair gene sequence centered at the SNP location.

Because gene sequences are discrete data, we use formula 9 to calculate the correlations. The computed C_{gene} is shown in Fig. 3. It indicates that there are strong correlations in every other column near the center of the gene covariance. Those patters will disappear after the first pooling layer. In other words, the customized filter CNN only differ with the baseline model in the first convolutional layer.

5 RELATED WORK

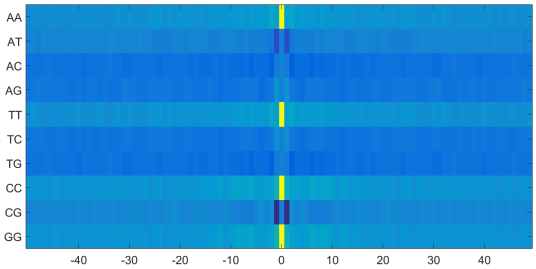


Figure 3: The correlation image for gene sequences.

There has been significant work on CNN network design.For example, AlexNet (Krizhevsky et al., 2012), VGG (Simonyan & Zisserman, 2014), GoogLeNet (Szegedy et al., 2014) and ResNet (He et al., 2016) all involved significant redesign process of the network structure. However, they all work on natural images hence their filter designs are mainly selecting the size for the square filters. Xie et al. (2016) proposed a customized shape for CNN filters by making a 3 by 3 hole in a 7 by 7 convolutional filter to tackle the structured labeling problems that focus on modeling object and context separately. This specific filter shape is designed manually and not learned from the data. (Bruna et al., 2014; Henaff et al., 2015) explored defining CNN on a graph via a hierarchical clustering approach and a harmonic analysis approach. Their approach can be used to recover a CNN on a lattice, however its sparsity is mostly in the frequency domain, and usually cannot achieve sparsity in the original domain as our approach. (Wen et al., 2016) adopted group lasso methodology to make the convolutional layer sparse. Our experiments showed that $L1$ regularization could improve the performance due to the sparsity but not as significant as our proposed model.

Convolutional neural networks are suitable for a wide range of application problem domains beyond natural images. Kalchbrenner et al. (2014) proposed a Dynamic Convolutional Neural Network (DCNN) on the problems of semantic modelling of sentences, which reduced at least 25% error reduction comparing with the strongest baseline framework. Also in the domain of Natural Language Processing (NLP), Ma et al. (2015) built a dependency tree between words within a sentence in order to capture the words that highly correlated with each other but located further apart within the sentence. Instead of concatenating successive words within a sequence, they concatenated words based on the dependency tree which redefined the state-of-the-art performance in both sentiment analysis and question classification. Shen et al. (2014) used convolutional neural network to learn semantic representations for search queries and Web documents, and the proposed frame work outperforms the previous state-of-the-art by a significant margin. Zbontar & LeCun (2015) successfully applied convolutional neural network to predict the degree that two image patches match for stereo matching. CNNs have also seen tremendous success in analyzing human speech (Abdel-Hamid et al. (2014)). Recently, there are increasing interest in applying CNNs for analyzing bird songs for the species prediction problem (Goëau et al., 2016). But to the best of our knowledge, there has been no prior study that focuses on classifying hummingbird species based on the sound of wingbeats.

In R-SNP prediction for gene sequences, a related work that we know of utilizes CNN is by Zhou & Troyanskaya (2015). They use CNN to extract high level features from gene sequences based on relevant domain knowledge in order to perform functional-variant predictions. We aim to predict the R-SNP directly from sequence-based data without integrating any domain knowledge.

In other related work, (Zhang & LeCun, 2015) uses the similar Rademacher complexity as in this work to characterize the sample complexity of CNN, but focus on a different research problem that creates an abstain option for the classifiers.

6 EXPERIMENTS

6.1 FILTER SHAPES AND NETWORK DESIGN

Experiments are conducted on three different tasks, two in the bird-bioacoustics domain and one for sequence-based rSNP prediction.

The first task is to identify the species of hummingbirds based on recorded wingbeats. This dataset contains 434 labeled recordings from a total of 16 species, where each recording is $20-30$ seconds long. Since the wingbeat signals almost exclusively reside in the low frequency range of the spectrogram, we only consider the frequency range $0Hz-300Hz$. Note that although every

recording contains some wingbeats, the signal may only occupy part of the recording. We apply a pre-processing step to extract wingbeat regions with high energy in the low frequency range. We then apply a sliding window to the resulting spectrograms and each window is considered as a separate instance. This process results in a total of 5922 labeled instances.

For the second task, our goal is to recognize the species of song-bird based on recorded bird songs. In this case, our data contains a total of 122 segments that have been manually extracted from spectrograms and are labeled with species. Each segment is a spectrogram in a short interval which contains some vocalization from a single bird and is labeled with its species. Similarly, a sliding window is applied to the segment and each window is treated as a separate instance for classification. In total we have 14 species and 1177 labeled instances.

The last task is to predict whether a non-coding Single-Nucleotide Polymorphism (SNP) serves any regulatory function. Our data contains 4300 SNPs that have been annotated as either positive (serving regulatory function) or negative (no known regulatory function). The data contains roughly the same number of positive and negative instances.

The covariance analysis procedure is applied to all datasets. We do not directly work with the regularization parameter λ in (7). Instead, we specify the maximum number of nonzero elements (namely $DFMax$) in the Lasso solution. For all datasets, we selected the DFmax values from 9, 11 and 13, and picked the smallest value that allowed for non-degenerate filters across all layers. The selected value is 13 for the wingbeat data, 9 for the bird-song data and the gene sequencing data. A stability study for a wider range of $DFMax$ values (3-15) was conducted and the results were shown in the appendix. Fig. 4 shows the filter shapes learned from the wingbeat data where eight convolutional layers are used. The first 4 filters primarily capture local horizontal patterns, whereas the last 4 filters capture more of the global harmonic structure in the data. More details and filter shapes in the other two datasets can be found in the appendix.

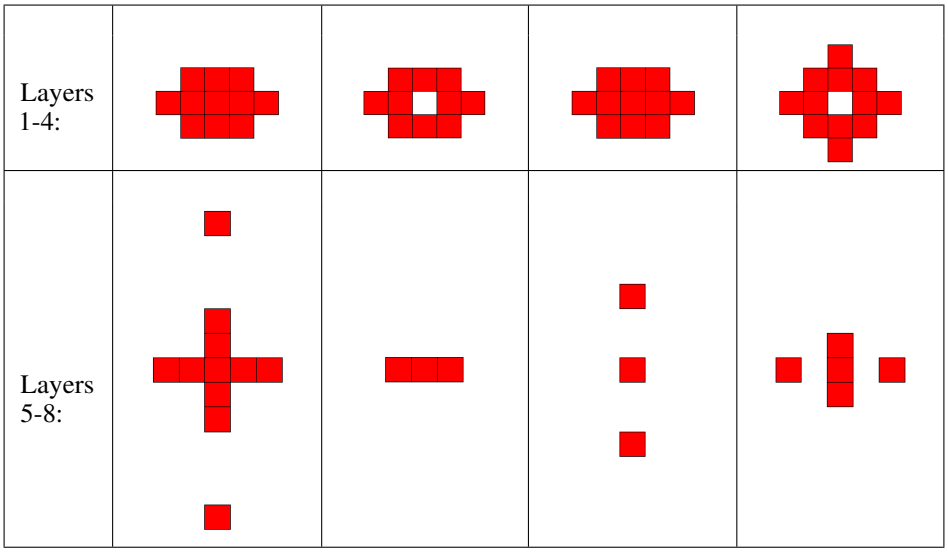


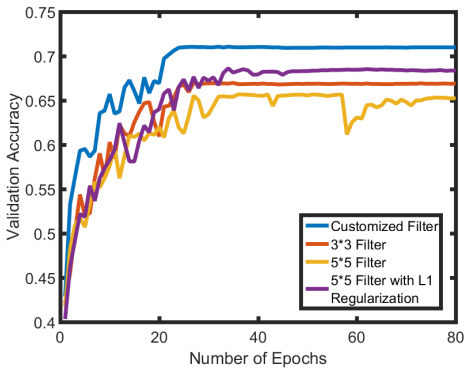
Figure 4: The derived filter shapes for the wingbeat data

6.2 WINGBEATS CLASSIFICATION RESULTS

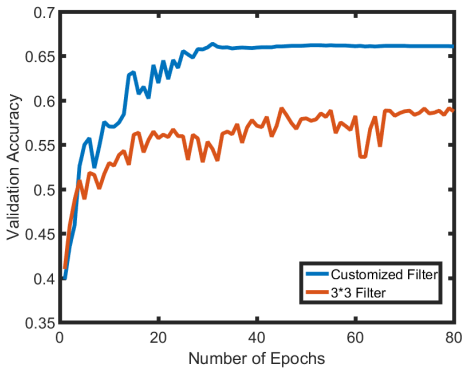
We consider three baselines: $3 \times 3, 3 \times 5, 5 \times 5$ convolutional filters. These are selected since 3×3 filter have roughly equal number of parameters compared to our customized filters, and 3×5 covers the domain of the customized filter in the first 3 levels. In the baseline of 5×5 filters, we added L_1 regularization on the weights, in order to test whether the same effect of filter shaping can be learned directly from sparsity priors on the weights. The swipe range for the L_1 regularization parameter is from 10^{-3} to 10^{-10}, and we selected the one that generated best performance. 5-fold cross-validation is conducted at the recording level. Thus, no validation example comes from the same recording as any of the training examples. For training, we set the batch size to 20 for all designs and the learning

rate to 10^{-4} for the customized filters and 10^{-5} and 10^{-6} for the 3×3 and 3×5 filters respectively. Fig. 5(a) shows the cross-validated performance as a function of training epochs. As can be seen from the figures that the cross-validation accuracy of our customized filter significantly outperforms that of the traditional filters, and L_1 regularization cannot achieve the same effect as filter shaping.

In order to further illustrate the robustness of our model in small-sample situations, we conducted an experiment with half of the instances from each recording, resulting 2961 instances in total. As shown in Fig. 5(b), the performance gap between the customized filter and in 3×3 widened significantly. More results on parameter sensitivity can be found in the appendix.



(a) 5-fold validation accuracy

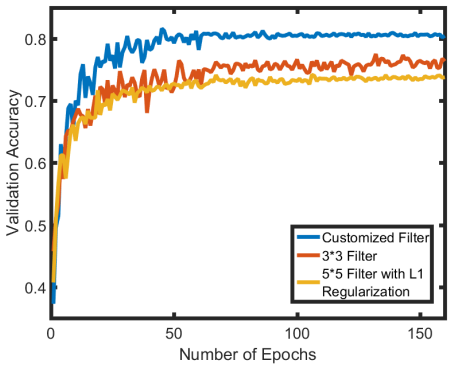


(b) 5-fold validation accuracy on reduced data (50%)

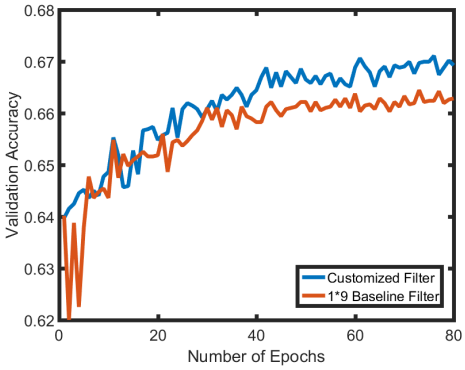
Figure 5: Accuracy on the wingbeats dataset

6.3 Bird song classification results

For the bird song dataset, we compare to a baseline of 3×3 filter as well as a 5×5 filter with L_1 regularization, and the swipe range for the L_1 regularization parameter is from 10^{-3} to 10^{-10} as well. 5-fold recording-level cross validation is performed and the averaged cross-validation accuracies are plotted in Fig. 6(a) as a function of the number of training epochs. From the figures, we observe that the customized filter achieves (5%) higher validation accuracy compared to the 3×3 filter. L_1 regularization on the weights negatively impacted performance for the 5×5 filter, perhaps because that there are not enough data to learn the sparsity structure of the weights.



(a) 5-fold validation accuracy for bird song



(b) 5-fold validation accuracy for rSNP

Figure 6: Validation accuracy on the birdsong and gene datasets

6.4 rSNP Prediction Results

For the gene dataset, we compare with a baseline that is similar with the framework used by Zhou & Troyanskaya (2015), which contains several layers of CNN with 1×9 shape filters. 5-fold cross-validation was performed and the results are shown in Fig. 6(b). The customized model was 0.8% better than the baseline. The performance gap is not as significant as in the bird-bioacoustics data, mainly because that the solved filters are not different with traditional ones after max-pooling (see

appendix for details). Due to the fact that we apply no domain knowledge, our performance may not be comparable with some recent papers which have used additional domain knowledge. However, it demonstrates the strength of the customized filter design over traditional ones.

7 CONCLUSION

In this paper, we introduced a theoretically justified correlation analysis approach to reveal the neighborhood structure in image-like domains. We proposed an algorithm that automatically designs multiple levels of filters in a CNN by solving repeated lasso problems. Experiments show that our approach significantly outperforms traditional fixed-size filters and exhibit higher robustness to parameters, especially on smaller datasets. In future work we would like to extend this methodology to other domains such as computational astronomy and biomedical imaging, as well as work on the theory in the discrete and multi-layer cases.

ACKNOWLEDGMENTS

This work is partially supported by the National Science Foundation grants IIS-1464371, CCF-1254218, and DBI-1356792 and IIS-1055113.

REFERENCES

Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(10):1533–1545, 2014.

P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014.

M. Everingham, L. Van Gool, Chris Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2012. www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Hervé Goëau, Hervé Glotin, Willem-Pier Vellinga, Robert Planqué, and Alexis Joly. Lifeclef bird identification task 2016: The arrival of deep learning. In *Working Notes of CLEF 2016-Conference and Labs of the Evaluation forum, Évora, Portugal, 5-8 September, 2016.*, pp. 440–449, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

Mario Lasseck. Bird song classification in field recordings: winning solution for nips4b 2013 competition. In *Proc. of int. symp. Neural Information Scaled for Bioacoustics, sabiod. org/nips4b, joint to NIPS, Nevada*, pp. 176–181, 2013.

Michel Ledoux and Michel Talagrand. *Isoperimetry and Processes in Probability in Banach Spaces*. Springer, 1991.

Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou. Dependency-based convolutional neural networks for sentence embedding. *arXiv preprint arXiv:1507.01839*, 2015.

Louis Ranjard and Howard A Ross. Unsupervised bird song syllable classification using evolving neural networks. *The Journal of the Acoustical Society of America*, 123(6):4358–4368, 2008.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pp. 373–374. ACM, 2014.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Dan Stowell and Mark D Plumbley. Birdsong and c4dm: A survey of uk birdsong and machine recognition for music researchers. 2011.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2074–2082, 2016.

Saining Xie, Xun Huang, and Zhuowen Tu. Convolutional pseudo-prior for structured labeling. In *European Conference on Computer Vision*, 2016.

Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1592–1599, 2015.

Xiang Zhang and Yann LeCun. Universum prescription: Regularization using unlabeled data. *arXiv preprint arXiv:1511.03719*, 2015.

Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning-based sequence model. *Nature methods*, 12(10):931–934, 2015.

Appendices

A PROOF OF THEOREM

The proof of the theorem follows from various Slepian-type variance comparison theorems. We state a couple of them here without proof. For proofs see Bartlett & Mendelson (2002); Ledoux & Talagrand (1991).

Theorem 3. *Ledoux & Talagrand (1991) Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be Gaussian random vectors in* $\mathbb{R}^d$ *such that for every* i, j*,*

$$\mathbb{E}|Y_i - Y_j|^2 \leq \mathbb{E}|X_i - X_j|^2 \tag{11}$$

, then for every non-negative convex increasing function F *on* $\mathbb{R}_+$*,*

$$\mathbb{E}F(\max_{i,j}|Y_i - Y_j|) \leq \mathbb{E}F(\max_{i,j}|X_i - X_j|). \tag{12}$$

Lemma 1. *Bartlett & Mendelson (2002) For* $\mathbf{x} \in \mathbb{R}^d$*, define*

$$F_1 = \{\mathbf{x} \mapsto \mathbf{w}^\top \mathbf{x} : \mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_1 \leq 1\}.$$

where $\cdot$ *represents an inner product. For any* $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$*, we have*

$$\hat{G}_n(F_1) \leq \frac{c}{n}(\ln d)^{1/2} \max_{j,j'} \left(\sum_{i=1}^{n} (x_{ij} - x_{ij'})^2 \right)^{1/2}. \tag{13}$$

where $\hat{G_n}(F)$ *is the Gaussian complexity of the function class* F*,* c *is a constant.*

This lemma characterizes the Gaussian complexity of simple inner product functions. Convolutional operators are also inner product functions, albeit they share weights over the entire domain (image) they are defined. Hence, this lemma can be used to bound the Gaussian complexity of each individual convolutional filter. In the proof, we convert the result of the convolutional operation into a number of inner product operations, and utilize Lemma 1 to bound the Gaussian complexity of each inner product operation.

Proof. (of Theorem) Without loss of generality, the result of the convolution operator with parameters $\mathbf{W} = [\mathbf{w_1}, \ldots, \mathbf{w_D}] * \mathbf{x}_i$ can be written as

$$f_{\mathbf{W}}(\mathbf{x}_i) = \mathbf{W} * \mathbf{x}_i = \left[\mathbf{w}_1 \cdot \mathbf{x}^{i1}, \mathbf{w}_1 \cdot \mathbf{x}^{i2}, \ldots, \mathbf{w}_1 \cdot \mathbf{x}^{im}, \ldots, \mathbf{w}_D \cdot \mathbf{x}^{i1}, \mathbf{w}_D \cdot \mathbf{x}^{i2} \ldots \mathbf{w}_D \cdot \mathbf{x}^{im}\right] \tag{14}$$

where $\mathbf{x}^{ij}$ represents a part in $\mathbf{x}_i$ equivalent to the size of the convolutional filter and m represents the number of such parts (size of the image). Let

$$X_{\mathbf{vW}} = \sum_{i=1}^{N} g_i \left(\mathbf{v} \cdot \sigma(f_{\mathbf{W}}(\mathbf{x}_i))\right) \tag{15}$$

where $g_i \sim \mathcal{N}(0, 1)$ are Gaussian random variables, so that

$$\|X_{\mathbf{vW}} - X_{\mathbf{v},\mathbf{W}'}\|^2 \leq \|\mathbf{v}\|^2 \|\sum_{i=1}^{N} g_i \sigma(f_{\mathbf{W}}) - \sum_{i=1}^{N} g_i \sigma(f_{\mathbf{W}'})\|^2 \tag{16}$$

$$\leq \|\mathbf{v}\|^2 \sum_{k=1}^{D} \|\mathbf{w}_k \cdot \sum_{i=1}^{N} g_i \sum_{j=1}^{m} \mathbf{x}^{ij} - \mathbf{w}'_k \cdot \sum_{i=1}^{N} g_i \sum_{j=1}^{m} \mathbf{x}^{ij}\|^2 \tag{17}$$

The first inequality is Cauchy-Schwarz, and the second inequality is because σ is a contraction mapping Ledoux & Talagrand (1991).

Now, if we define $Y_{\mathbf{w}_k} = \mathbf{w}_k \cdot \sum_{i=1}^{N} g_i \sum_{j=1}^{m} \mathbf{x}^{ij}$ then according to Slepian's lemma Bartlett & Mendelson (2002); Ledoux & Talagrand (1991) there exist constants c and B so that

$\mathbb{E}[\sup_{\mathbf{v},\mathbf{W}}(X_{vw})] \leq cB\sum_{i=1}^{k}\mathbb{E}[\sup_f(Y_{w_k})]$, and according to Lemma 1,

$$\mathbb{E}[\sup_f(Y_{w_k})] \quad \leq \quad \frac{c}{N}(\ln d)^{1/2}\max_{t,t'}\left(\sum_{i=1}^{N}\left(\sum_{j=1}^{m}x_t^{ij}-\sum_{j=1}^{m}x_{t'}^{ij}\right)^2\right)^{1/2} \tag{18}$$

Putting those together and noting $\hat{G}_N(F) = \mathbb{E}[\sup_{\mathbf{v},\mathbf{W}} X_{\mathbf{vW}}]$ (F as defined in Theorem 2) finishes the proof of the theorem.

□

B Filter Shapes and Network Structure for Birdsong and Gene Expression

Fig. 4 and Fig. 7 show the filter shapes that are learned from the covariance images of the wingbeat data and the birdsong data respectively.

Similarly, for the birdsong data, filters in Fig.7(1) to 7(4) are long in the vertical direction and capture local vertical patterns, whereas filters in Fig.7(5) to 7(8) are stretched along the horizontal direction, which accounts for longer temporal patterns that occur at a more global scale.

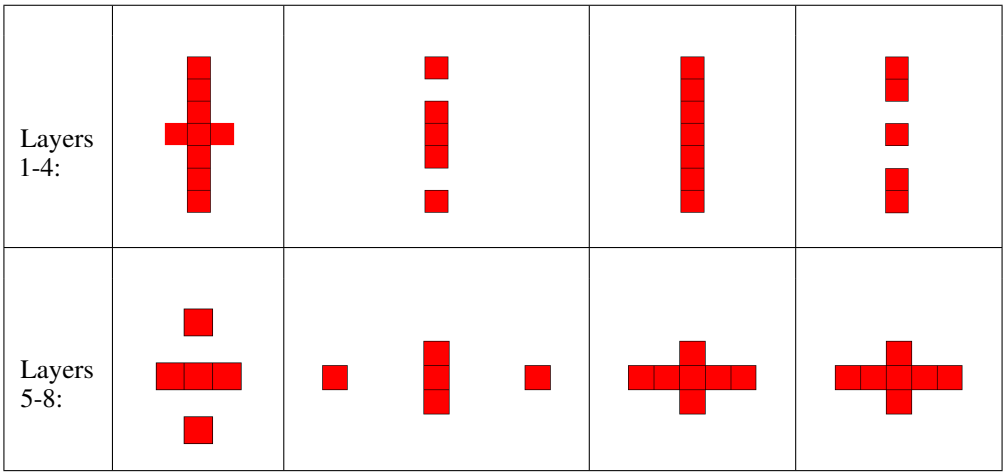


Figure 7: The derived filter shapes for the bird song data

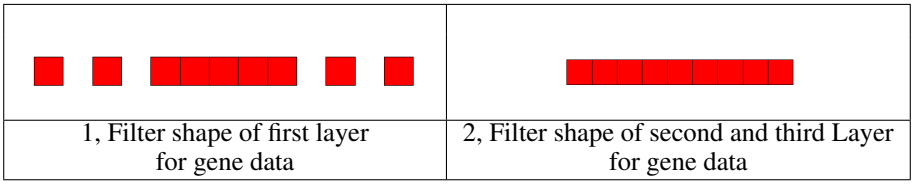


Figure 8: The derived filter shapes for gene data

Tables 1, 2 and 3 describes the network structure for the wingbeats, birdsong and gene data, respectively. Every convolutional layer is followed by a ReLU layer.

Layer name	Layer description
Input	307×49 spectrogram
Conv1	Fig. 4(1) (or 3×3) conv. 64
Conv2	Fig. 4(2) (or 3×3) conv. 64
Maxpool	2×2 stride $(2, 2)$
Conv3	Fig. 4(3) (or 3×3) conv. 128
Conv4	Fig. 4(4) (or 3×3) conv. 128
Maxpool	2×2 stride $(2, 2)$
Conv5	Fig. 4(5) (or 3×3) conv. 256
Conv6	Fig. 4(6) (or 3×3) conv. 256
Maxpool	2×2 stride $(2, 2)$
Conv7	Fig. 4(7) (or 3×3) conv. 512
Conv8	Fig. 4(8) (or 3×3) conv. 512
FC-1024	fully connected layer
FC-1024	fully connected layer
Softmax layer	

Table 1: The structure of the CNN for the wingbeat dataset.

Layer name	Layer description
Input	256×46 spectrogram
Conv1	Fig. 7(1) (or 3×3) conv. 32
Conv2	Fig. 7(2) (or 3×3) conv. 32
Maxpool	2×2 stride $(2, 2)$
Conv3	Fig. 7(3) (or 3×3) conv. 64
Conv4	Fig. 7(4) (or 3×3) conv. 64
Maxpool	2×2 stride $(2, 2)$
Conv5	Fig. 7(5) (or 3×3) conv. 128
Conv6	Fig. 7(6) (or 3×3) conv. 128
Maxpool	2×2 stride $(2, 2)$
Conv7	Fig. 7(7) (or 3×3) conv. 256
Conv8	Fig. 7(8) (or 3×3) conv. 256
FC-512	fully connected layer
FC-512	fully connected layer
Softmax layer	

Table 2: The structure of the CNN for the bird song dataset.

Layer name	Layer description
Input	4×2001 one-hot gene sequence
Conv1	4×9 conv. 64
Maxpool	1×4 stride $(1, 4)$
Conv2	1×9 conv. 128
Maxpool	1×4 stride $(1, 4)$
Conv3	1×9 conv. 256
Maxpool	1×4 stride $(1, 4)$
Sigmoid layer	

Table 3: The structure of the CNN baseline for the gene dataset.

Layer name	Layer description
Input	4×2001 one-hot gene sequence
Conv1	Fig. 8(1) conv. 64
Maxpool	1×4 stride $(1, 4)$
Conv2	Fig. 8(2) conv. 128
Maxpool	1×4 stride $(1, 4)$
Conv3	Fig. 8(2) conv. 256
Maxpool	1×4 stride $(1, 4)$
Sigmoid layer	

Table 4: The structure of the CNN for the bird song dataset.

C SENSITIVITY TEST ON PARAMETERS

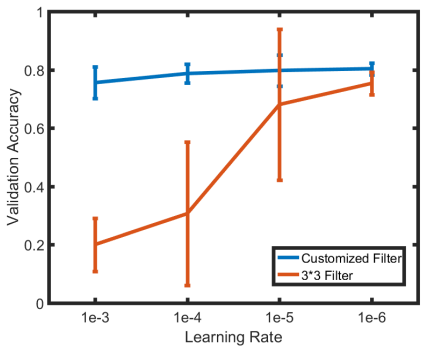


Figure 9: Sensitivity to the learning rate on the bird song data.

In our experiments, we have observed that CNNs with customized filters are more robust to the choices of hyperparameter compared to those using 3×3 filters, especially on smaller datasets. To illustrate this effect, a separate experiment is designed where we train different networks with the learning rate varying from 10^{-3} to 10^{-6}. To measure the performance of a network with each learning rate, we randomly sample 80% of the data for training and test on the remaining 20%. We repeat this five times and report the average testing accuracy achieved by CNN with customized filter and the traditional 3×3 filter. Fig. 9 shows the results on the smaller bird song dataset, and it can be seen that with the customized filter we are able to obtain good learning results with a much wider range of learning rates. On bigger datasets such as wingbeats such trends also exist, but not as significant as in a smaller dataset such as birdsong.

D DFMAX STABILITY

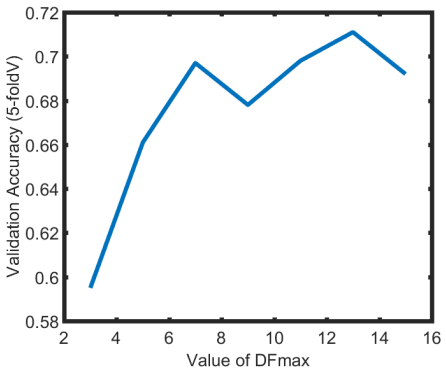


Figure 10: Stability of DFMax

Instead of controlling the parameter λ in LASSO, we introduce another parameter $DFMax$ to limit the maximum number of nonzero solutions in LASSO. It is shown that different $DFMax$ values can generate different sets of customized convolutional filters. In order to investigate the relation between $DFMax$ and performance, we swipe $DFMax$ from 3 to 15 and perform 5-fold cross validation on wingbeats dataset. Fig. 10 shows that the performance of our framework is stable for a wide range of $DFMax$ values. When $DFMax$ equals to 3, all customized convolutional filters are in shape 1×3, it is reasonable for its performance is not as high as the framework with higher $DFMax$ values.