Image and Vision Computing 77 (2018) 45–59

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Structured deep Fisher pruning for efficient facial trait classification ☆

Qing Tian*, Tal Arbel, James J. Clark

Centre for Intelligent Machines & ECE Department, McGill University, 3480 University Street, Montreal, QC, Canada

ARTICLE INFO

Article history:
Received 16 September 2017
Received in revised form 4 June 2018
Accepted 21 June 2018
Available online 5 July 2018

Keywords:
Neural network pruning
Fisher LDA
Facial trait classification

ABSTRACT

High efficiency is desirable for many interactive biometrics tasks, including facial trait recognition. Although deep convolutional nets are effective for a multitude of classification tasks, their high space and time demands make them impractical for PCs and mobile devices without a powerful GPU. In this paper, we propose a structured filter-level pruning approach based on Fisher LDA [1], which boosts efficiency while maintaining accuracy for facial trait classification. It starts from the last convolutional layer where we find filter activations are less correlated. Through Fisher's LDA, we show that this decorrelation makes it safe to discard directly filters with high within-class variance and low between-class variance. The pruning goes on by tracing deconvolution based dependency over layers. Combined with light classifiers, the reduced CNNs can achieve comparable accuracies on example facial traits from the LFWA(+) and CelebA datasets, but with large reductions in model size (96%–98% for VGG-16, 81% for GoogLeNet) and computation (as high as 80% for VGG-16, 61% for GoogLeNet).

1. Introduction

Deep artificial neural networks have revolutionized many computer vision areas, in large part due to effective algorithms for large network training [2-5] and the huge amount of data that has become available. In particular, the effectiveness of convolutional neural networks (CNN), a concept from the 1980s [6], has been fully revealed in a wide variety of visual recognition tasks, including visual biometrics recognition. From the 'very' deep VGG-16 net [7] and GoogLeNet [8] to 'extremely' deep Microsoft ResNets [9], the competition for higher accuracy with ever larger depths is strong. However, with increased depth comes more space and computational burden, which renders deep nets' wide deployment on mobile and VR devices impractical. Memory, speed, and power can still be major constraints, even with powerful GPUs.

In this paper, we explore ways to greatly prune very deep networks while maintaining or even improving on their classification accuracy. Our motivation stems from the current popular practice where, rather than start from scratch, algorithm developers adopt a pre-trained general network model and fine-tune it for a particular task at hand on a smaller dataset. However, there is usually no theory to justify the 'inherited' structure. Therefore, chances are that some structures from the pre-trained model are not fully adapted for the current task. Our premise is that less useful structures (together with possible redundancies) could be pruned away in order to increase computational efficiency. Although our approach is generally applicable to many vision-based real-time biometrics problems, here we explore its effectiveness in the context of facial trait classification. This paper extends our previous work [10] by including more facial traits, the ability to prune module-based deep structures (e.g. GoogLeNet [8]), as well as detailed pruning and efficiency analysis (in terms of number of parameters and FLOPs). We will also briefly discuss our approach's potential for facial identification tasks where data is limited for general face embedding learning.

Deep convolutional networks are generally considered to be composed of: (1) convolutional (conv) layers as feature extractors and (2) fully connected (FC) layers as a final classifier[1]. Deep nets outperform many traditional computer vision algorithms largely because, given enough valid training data, the first component does well in learning the compositionality of the real world. It does this by constructing very complicated features based on primitive ones. More often than not, such features learned for a particular task are superior to handcrafted features designed with limited domain knowledge. The

☆ This paper has been recommended for acceptance by Vitomir ?truc.

* Corresponding author.
E-mail addresses: qing.tian@mail.mcgill.ca (Q. Tian), arbel@cim.mcgill.ca (T. Arbel), clark@cim.mcgill.ca (J.J. Clark).

[1] In this paper, conv and FC layers are used in a general sense.

https://doi.org/10.1016/j.imavis.2018.06.008
0262-8856/

FC layers can be considered as a traditional Multi Layer Perceptron (MLP) with softmax in the output layer. It should be noted that although FC layers dominate the size of most popular deep CNN models, it is usually the conv layers that account for most computations (mainly due to weight sharing). In addition to bringing down the number of parameters and computations, decreasing conv layer complexity reduces the size of input/output feature maps. This leads to fewer memory accesses which tend to consume non-negligible amount of energy. Therefore, in this paper, our pruning approach mainly focuses on the CNN features. By investigating the firing patterns of conv layer filters through Fisher's Linear Discriminant Analysis (Fisher LDA) [1], we discover that final conv layer filter activations are highly decorrelated, which permits directly discarding of a large number of less informative filters without loss of information. To increase the efficiency for the second final classifier component, we can replace FC layers with light alternatives. For *in-the-wild* cases, Bayesian QDA is found to perform slightly better than the widely used SVMs, when combined with our pruned features. Unlike unstructured pruning [11], our structured pruning approach makes direct space and computational savings possible. On LFWA and CelebA, our method shows a reduction in space of over 95% for VGG-16 and 81% for GoogLeNet and computational savings as high as 80% for VGG-16 and 61% for GoogLeNet.

The remainder of the paper is structured as follows: The relevant literature is reviewed in Section 2. In Section 3, our structured Fisher's LDA based pruning approach is introduced. Section 4 describes our experimental validation and compares our modified nets to their originals as well as other pruned/shallow structures in terms of accuracy and efficiency. In Section 5, our contribution and possible future directions are discussed. Finally, Section 6 concludes the paper.

2. Related work

2.1. Facial trait classification

The classification of facial traits, especially those considered as biometrics, has attracted much attention from the computer vision community over the years. Traditional approaches are based on hand-engineered features that can be grouped to be either global [12,13] or local [14-18]. Perez et al. [19] used a combination of both kinds of features together with mutual information. O'Toole et al. [20] showed that depth information can also be helpful. Most of the above methods were tested on highly controlled benchmarks such as FERET [21] where near-perfect accuracies have been achieved [22,19]. In an attempt to deal with in-the-wild cases, Shan [23] employed Local Binary Patterns (LBP) and reported satisfactory results on a subset of the Labelled Faces in the Wild (LFW) dataset [24]. However, many images in that subset contain only frontal or near frontal faces. In terms of classification, SVMs are widely used alone [25] or with boosting algorithms such as Adaboost [26,23]. For example, in the FaceTracer system, Kumar et al. [27] constructed local SVMs based on a rich set of local feature options and utilized Adaboost to automatically select a linear combination of them. FaceTracer can produce comparably high accuracies on relatively simple datasets. When it comes to more challenging cases, such as when many occlusions and view changes are present, Toews and Arbel [28] demonstrated Bayesian classifiers' superiority to SVM when using a multiple local scale-invariant features based Bayesian framework. We refer our readers to [29,30] for more traditional works on both feature extraction and classification for facial trait classification. Although they can do well in certain well controlled cases, the main problem with handcrafted features based classification approaches is that they require domain knowledge and may not generalize well (e.g. on more challenging datasets in the wild).

Over the past decade, deep neural networks have become state-of-the-art. 'Features and classifiers' learned by them usually lead to better performance given enough valid data. As a matter of fact, artificial neural networks, for use in classification tasks, have been around for almost half a century. In the 1990s, they began to be employed for facial trait classification [31-33]. However, the shallow structures of early neural networks constrained their performance and applicability. It was not until Krizhevsky et al. [34] won the 2012 ImageNet Recognition Challenge with a ConvNet (AlexNet, similar to [6]) that neural networks regained attention. In the following years, various deep nets were successfully applied to a variety of visual recognition tasks including facial attribute classification. Verma et al. [35] showed that the learned CNN filters correspond to similar features that neuroscientists identified as cues used by human beings to recognize facial traits such as gender. Inspired by the dropout technique in training deep nets, Eidinger et al. [36] trained a SVM with random dropout of some features and achieved promising results on their relatively small Adience dataset, on which Levi and Hassner [37] later trained and tested a not-very-deep CNN. Instead of training on entire face images, Mansanet et al. [38] trained relatively shallow nets using local patches and reported better accuracies than whole image based nets of similar depths. To better utilize shape information, Li et al. [39] proposed tree-structured CNNs with shape adaptive kernels for facial trait classification. Zhang et al. [40] modeled deep attributes by combining pose-normalized CNNs (PANDA) and achieved satisfactory results. Liu et al. [41] leveraged multiple CNNs in a sequential pipeline to detect facial regions (using LNets) and then recognize facial traits (using ANet). They reported state-of-the-art accuracies on the CelebA and LFWA datasets. Given enough training data and reasonable layer structures, larger depths generally lead to higher accuracies [7,9]. However, the number of parameters, computations, and the amount of energy consumption explode with an increase of network depths. What's more, when labeled data is limited, larger networks are more prone to over-fitting. Few works have properly addressed this issue.

Finally, there are works that address facial trait classification based on identity features learned for recognition/verification purposes (e.g. [42]). Although it works well for identity-related facial traits, it might not work, or at least not directly, for non-identity facial traits. To classify facial traits efficiently, it is desirable to directly disentangle useful structures from possibly redundant and less useful ones.

2.2. Deep neural networks pruning

Early works in this direction include magnitude-based biased weight decay [43], Hessian based Optimal Brain Damage [44] and Optimal Brain Surgeon [45]. Since they targeted shallow nets, assumptions such as diagonal Hessian in [44] do not necessarily hold for deep nets. In this section, we focus on approaches designed for deep nets. For earlier work, we refer our readers to [46].

In the past 5 years, neural networks tend to go deeper and deeper. With more layers, not only come possible higher accuracies [7], but also comes more complexity. This re-ignited research in network pruning. However, most pruning approaches today are local (ignoring relationships within filters or across layers) and/or with a suboptimal utility measure (e.g. equating large weight/activation magnitudes to high importance). Han et al. [11] pruned away small weights and achieved satisfactory compression rates. In terms of implementation, small values are set to zero and masks are required to freeze/disregard such weights during retraining. Although compression schemes, such as weight quantization and Huffman encoding [47], can make the model smaller for storage and transferring, after being deployed, the actual model size and computation do not change much without specific hardware and software optimizations.

Other approaches that sparsify networks by setting weights to zero include [48-53].

To better utilize pruning's computational advantages, Anwar et al. [54] located pruning candidates using particle filters in a more structured way. Recently, with the increasing depths and complexity of deep nets, filter or channel level pruning [55,56,10] gained popularity because they can directly lead to space and computational savings without incurring much overhead. However, most of these works' utility measure is not accurate enough and, usually, each pruning candidate is weighted separately. It is also worth mentioning that some approaches control storage and computational complexity by reducing weight bitwidth (e.g. XNOR-Net and BWN [57]) at the expense of possible accuracy loss. Another way is to adopt compact modules/layers with fewer weights (e.g. Network-in-Network [58], GoogLeNet [8], SqueezeNet [59], MobileNet [60], and ResNet [61]).

Unlike traditional approaches, we treat network pruning as a structured and supervised dimensionality reduction problem in the learned deep feature space. It is worth mentioning that the pruning algorithm in this paper can be combined with other techniques to further boost efficiency, such as weight sharing and Huffman coding [47], bitwidth reduction [57,62,63], decomposition of filters [64,65], feature maps pruning [66], and student-teacher learning [67].

3. Fisher LDA based structured pruning

3.1. Inspirations for structured filter level pruning

In CNNs, weights in a filter join forces to respond to a particular pattern, and we believe that it is beneficial to consider all weights in a filter as a whole. Structured filter level pruning is promising because it is able to directly result in both filter-wise and channel-wise savings (current layer's filter outputs correspond to next layer's input channels). However, as mentioned previously, many pruning approaches are unstructured. Unstructured sparsity produced, if any, cannot directly result in much space and time savings. Furthermore, the relationship of inner filter parameters is usually overlooked, which runs the risk of breaking their aggregate impact. This is especially true when the pruning percentage is large (Section 4.2.2). For example, in a conv filter, there could be one large positive weight and several negative weights of slightly smaller magnitudes. Given a uniform positive input map, the joint effect of the smaller weights leads to a negative output (0 after ReLU). If pruning is according to individual weight magnitudes, all the negative weights would be abandoned first. At a certain point during pruning, the filter output sign would change to positive. Such pruned filters could possibly adversely affect feature extraction and final classification.

We also draw inspiration for (supervised) structured filter level pruning from neuroscience findings. Although there are numerous neuron (filter) connections in the brain cortex, each neuron (filter) typically only receives inputs from a small task-specific set [68]. Mountcastle [69,70] hypothesized the existence of functional columns in the cortex. It has been demonstrated anatomically [71-73] and functionally [69,74-76] that minicolumns have accompanying functionalities, which only becomes clear when seen on the higher macrocolumn level. According to [77,78], the macrocolumn level activations tend to be sparse. Such high level structured sparsity indicates the possibility of pruning on the filter (function unit) level in artificial networks.

3.2. Dimension reduction in the last conv layer

In our work, we fully train the network (using cross entropy loss with L2 regularization and dropout) before pruning the CNN features starting from the last conv layer and replacing FC layers with light alternatives. The last conv layer is chosen as the starting point because its filters are observed experimentally to fire more uncorrelatedly than other conv layers (according to our analysis, from bottom to top, the filter activations become progressively more decorrelated). This, as will be seen, is critical for our LDA-based approach. Moreover, unlike FC layers, last conv layer preserves location information, and is not restricted by input dimension. In fact, many works such as [79,80] have demonstrated the last conv layer's possible superiority over FC layers in facial traits classification and image retrieval.

We define the maximum activation value of a filter as its firing score. Then for each input image an N-dimensional firing vector can be obtained in the last conv layer ($N = 512$ for VGG-16, $N = 1024$ for GoogLeNet), which we call a firing instance or observation. By stacking all of these observations extracted from a set of images, the firing data matrix X for that set is obtained. In our experiments, X is normalized as a pre-processing step. The benefits of abandoning less useful dimensions in X are twofold: (1) compressing features (i.e. potential for pruning), (2) helping reveal critical patterns buried in high dimension (even when the available data is limited), thus simplifying classification and possibly boosting accuracy. In contrast to label-blind unsupervised dimensionality reduction techniques with general utility measure (e.g. high variance for PCA, the reconstruction power for autoencoders), we draw our inspirations from Fisher's LDA [1] and its applications on face images [13,81-84], and adopt the intra-class correlation (ICC) to better measure information utility for facial trait classification:

$$ICC = \frac{s^2(b)}{s^2(b) + s^2(w)} \tag{1}$$

where $s^2(w)$ is the variance within each class, $s^2(b)$ is the variance between classes, and the sum of the two is the overall variance across all samples from all classes. When reducing dimensions, we are trying to maximize *ICC*, which has an equal effect of maximizing between-class variance while minimizing within-class variance. The direct multivariate generalization of it is:

$$W_{opt} = \arg\max_W \frac{|\hat{S}_b(W)|}{|\hat{S}_w(W)|} = \arg\max_W \frac{|W^T S_b W|}{|W^T S_w W|} \tag{2}$$

where

$$S_w = \sum_i \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T \tag{3}$$

$$S_b = \sum_i N_i (\mu_i - \mu)(\mu_i - \mu)^T \tag{4}$$

x_k is a firing instance of the last conv layer, μ is the mean firing vector, and i indicates the class. W is the orthonormal transformation matrix projecting the data X from its original space to a new space with the columns in W being the new space's coordinate axes. Through analyzing S_w and S_b for both the Labeled Faces in the Wild (LFW) dataset and the CelebFaces Attributes (CelebA) dataset [41] in our experiments, we find that most off-diagonal values in S_w and S_b are (near) zero, which is to say, the firing of different filters in the last conv layer is highly uncorrelated. This is because high dimensional coincidences can hardly occur by chance. It is also intuitive given the fact that, unlike common primitive features, higher layers capture high-level abstractions of various aspects (e.g. car wheels, dog ears, and flower pedals). The chances of them firing simultaneously are much lower than primitive filters in an early layer. Fig. 1 shows some examples of the last conv layer patterns in a VGG-16 network trained for gender classification on CelebA.

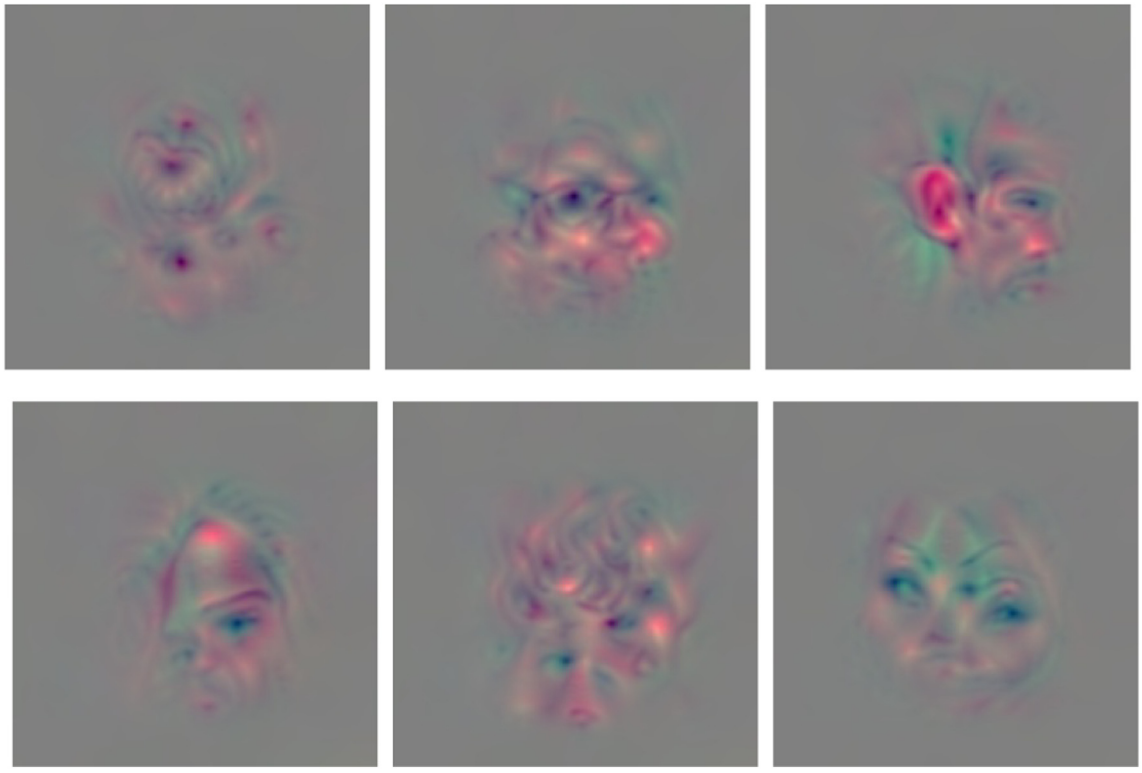

Fig. 1. Sample VGG-16 conv5_3 filter patterns (trained for gender on CelebA). From left to right, top to bottom, they fire for goatees/mustaches, glasses, ears, hairline, curly hair, and noses respectively. Each pattern is synthesized via a regularized optimization algorithm [85] and can be interpreted as the pattern the corresponding filter fires most on in the input image.

With most off-diagonal values (near) zero, the numerator and denominator in Eq. (2) (inside the norm sign) can then be viewed as eigen decompositions of the between-class scatter $\hat{S}_b$ and the within-class scatter $\hat{S}_w$. Since columns in W^T are eigenvectors of $\hat{S}_w$ and $\hat{S}_b$, the columns in W are eigenvectors of S_w and S_b (under the constraint in [84], which imposes low redundancy for our pruning while help resolve possible under-sampled problems). It follows that W columns are the standard basis vectors (the new uncorrelated Fisher LDA axes are aligned with original filter axes). When reducing dimensionality, maximizing between-class scatter while minimizing within-class scatter simply becomes selecting the filter/neuron dimensions of low within-class variance and high between-class variance. For instance, in Fig. 1, although both the goatee/mustache filter and the glasses filter have high variances (that PCA prefers), given the specific task of gender recognition, the goatee/mustache dimension has a higher chance to be selected by Fisher's LDA due to its higher ICC. This corresponds to intuition, as most females do not have goatee/mustache while many males do. The direct abandonment of certain last conv layer filters greatly facilitates pruning at all other layers.

3.3. Dependency aware layerwise pruning

Last conv layer dimensionality reduction along filter/neuron dimensions serves as the start for cross-layer filter level pruning. With the removal of a filter, the dependencies of this filter on others in previous layers are also eliminated. When all large dependencies on a filter from higher layers are removed, this filter can be discarded. Take Fig. 2 for example. The remaining feature maps (filter outputs) in a layer are colored in cyan. Corresponding useful depths of a next layer filter are colored in green (e.g. useful depths of each remaining $C3$ filter is represented by the small green block in column $C2$). The remaining cyan feature maps (overlapped with the green useful depths of a next layer filter) depend only on those cyan feature maps in the previous layer. Non-colored filter parts and feature maps can thus be discarded, contributing to channel-wise and filter-wise savings, respectively. In this particular example, when 106 $C2$ filters (small blocks in column $C1$) are thrown away, not only the $C2$ convolution computations with $C1$ output data are reduced by 106/128, but also $C3$ filters' depth is reduced by the same ratio (as shown in green in Column $C2$). The same applies when other layer filters are discarded. In total, 151,938 conv layer parameters are pruned away.

In this paper, the dependency of a filter on others in previous layers is calculated using deconvolution (deconv) [86,87], a technique mapping a max activation through lower layers all the way to the pixel level. As a mirrored version of the feed forward process, the deconv procedure consists of series of unpooling (utilizing stored max location switches), rectification, and reversed convolution (using a transpose of the filter). Instead of capturing a certain order dependency (e.g. 1st order gradient), we employ deconv to reconstruct the contributing sources of useful disentangled firing patterns. Additionally, we mainly care about the maximum activation of each filter. Deconv is more robust to noise activations. It is also worth noting that the dependency here is learned by pooling over training samples, which simulates the biology fact that multiple exposures can strengthen relevant synapses in the brain (the Hebbian theory [88]). Our approach is superior to local utility based pruning because we consider both within filter and across layer relationships that directly contribute to the final classification. The important aspect here is the combined effect of weights/filters on the final classification. On the other hand, local weight/activation magnitudes/variances are not necessarily as important. No optimization procedure is perfect, some weights/activations may stay/become irrelevant to the output classification (this is especially true when the

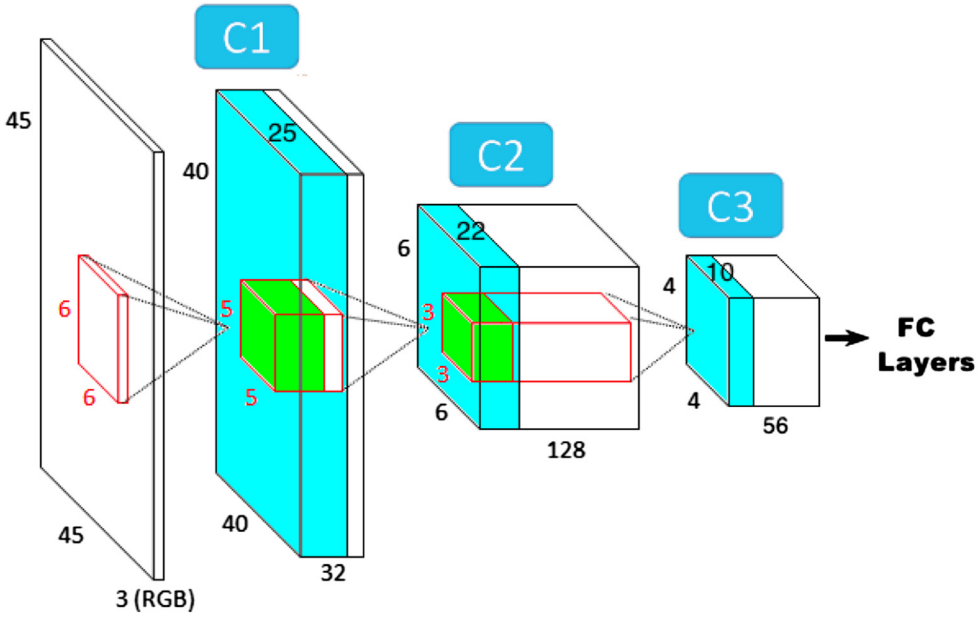

Fig. 2. Demonstration of pruning CNN on the filter level. Cyan indicates remaining data (feature maps), green represents the surviving part of a remaining next layer filter.

network is pre-trained for a different task and fine-tuning data is limited). Regardless of their magnitudes, irrelevant weights/activations are of little use. When pruning, the filters with a (max) LDA-deconv dependency smaller than a threshold is deleted. In our experiments, such a threshold is not difficult to set. Except for the first few conv layers, deconv dependencies in most other layers tend to be sparse. When the threshold is smaller than a certain value t_0, an accuracy plateau is reached, beyond which point the accuracy does not change too much with the decrease of the threshold (Fig. 5). t_0 can then be selected as the final threshold with a purpose of deriving a compact structure without obvious accuracy loss. That said, the threshold can be set differently according to different needs such as highest accuracy or even lighter structures with possibly more accuracy loss. To recover high accuracy, one-time retraining (from the surviving parameters without re-initializing) is needed after pruning.

3.4. Alternative classifiers on top of CNN features

FC layers can be considered as a final classifier on top of extracted CNN features. Even though FC layers are not as computationally intensive as conv layers, in many cases, they dominate the model size and may lead to over-fitting for small datasets. As such, this leads to the possibility that by replacing these layers with lighter classifiers, a reduction in parameters and/or an increase in accuracy become possible.

In fact, a wide variety of methods have combined CNN features with SVMs and have been met with some success in this regard [89,90,80]. Specifically for face attributes recognition, Zhong et al. [80] found that linear SVM, together with CNN features, is able to achieve higher mean prediction accuracy than FC layers. That said, not many other classic classification methods have been tested with deep net features. Before the deep learning era, Toews and Arbel [28] demonstrated that Bayesian classifiers possess the potential to outperform SVMs for facial trait recognition in the wild (where a wide variety of occlusions and view changes are present). Moreover, rather than a binary decision, Bayesian classifiers have a nice probabilistic interpretation and are usually easier to train than SVM. In spite of these possible advantages, without the naive dimensions' independence assumption, Bayesian classifiers are vulnerable to the usual high dimensionality curse of deeply learned features. Therefore, relatively few approaches bother to combine the two naturally ill-matched pair. However, as will be shown later, our Fisher LDA pruning manages to avoid their conflicts and take advantage of both methods by preserving enough discriminating power well before the dimensionality curse kicks in (Figs. 3 and 4). In this paper, Bayesian quadratic discriminant analysis (QDA) is used in addition to SVMs for facial traits recognition on the large CelebA and in-the-wild LFWA(+) datasets.

4. Experiments and results

4.1. Experimental setup

Although our pruning approach is generally applicable, in our experiments, we choose two well-known deep structures, i.e. VGG-16 [7] and GoogLeNet (a.k.a. Inception V1 [8]), as representative examples for conventional CNNs and module-based CNNs, respectively. The VGG-16 is pre-trained on the ImageNet and IMDB-WIKI [91] data. The GoogLeNet is pre-trained solely on ImageNet. As for fine-tuning and testing, the following two datasets are used in this paper.

The Labeled Faces in the Wild (LFW) dataset [24] is a popular face recognition/verification dataset that covers a large range of pose and background clutter variations. It contains over 13,000 web-collected face images of 5749 identities, each with an identity name. The LFWA dataset is a richly labeled version of LFW and has 40 facial traits labels for each image. We obtain the images, labels, and training/testing splits from [41]. Since there is no explicit validation split provided, we select identities with last name starting from 'R' to 'Z' for validation purposes. In an extended version of LFWA, Liu et al. [41] provide 30 extra facial attributes including race, which we use as a multi-category trait example to test our pruning approach's efficacy on GoogLeNet.

Another dataset used in this paper is the CelebFaces Attributes Dataset (CelebA) [41], which is a larger dataset containing 202,599 images of 10,177 identities with the same 40 facial attribute labels

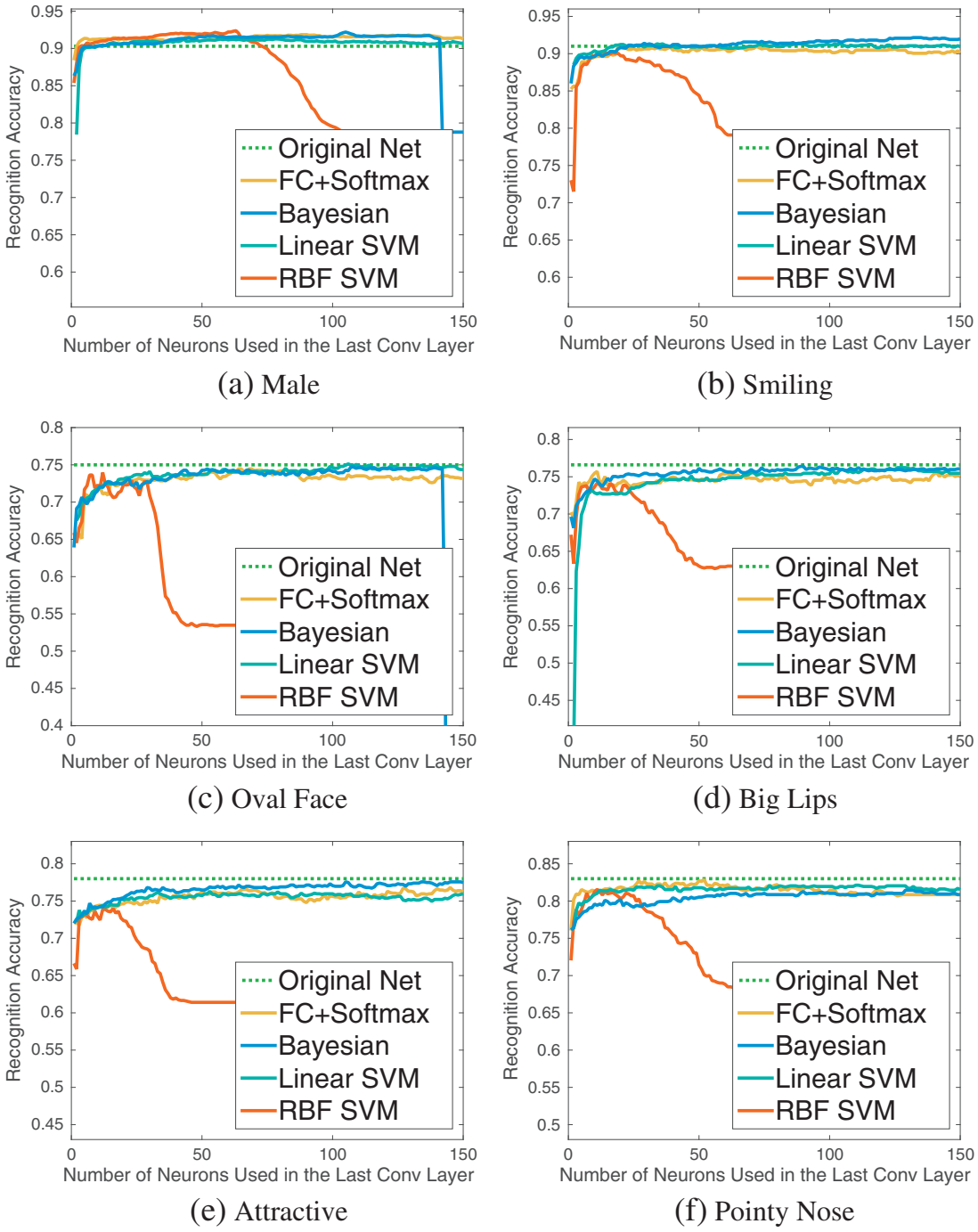

Fig. 3. Accuracy comparison of alternative classifiers using LDA-pruned CNN features on the LFW dataset. Horizontal axis: number of filters/neurons preserved in the last conv layer, vertical axis: recognition accuracy.

as LFWA for each image. Despite its relatively large size, most of its images are celebrity portrait photos against simple backgrounds. The train/validation/test splits suggested in [41] are adopted. In our experiments, all images are pre-sized to a dimension of 224*224. Our programs are based on the Caffe [92] library and include more functions such as filter-level pruning and LDA-deconv dependency calculation.

In the following three sections, we will demonstrate and discuss our pruning approach's efficacy. Section 4.2 focuses on pruning conventional deep models (e.g. VGG-16) for binary facial trait classification. Section 4.3 will be dedicated to pruning deep modular structures for recognizing multi-category facial traits. In Section 4.4, we will briefly discuss our pruned facial trait structure's potential for the related task of facial identification with limited data.

4.2. VGG-16 on LFWA and CelebA

In this section, we test our pruning approach on the well-known VGG-16 structure. From the LFWA and CelebA datasets, we select example facial traits that cover both global and local, identity related

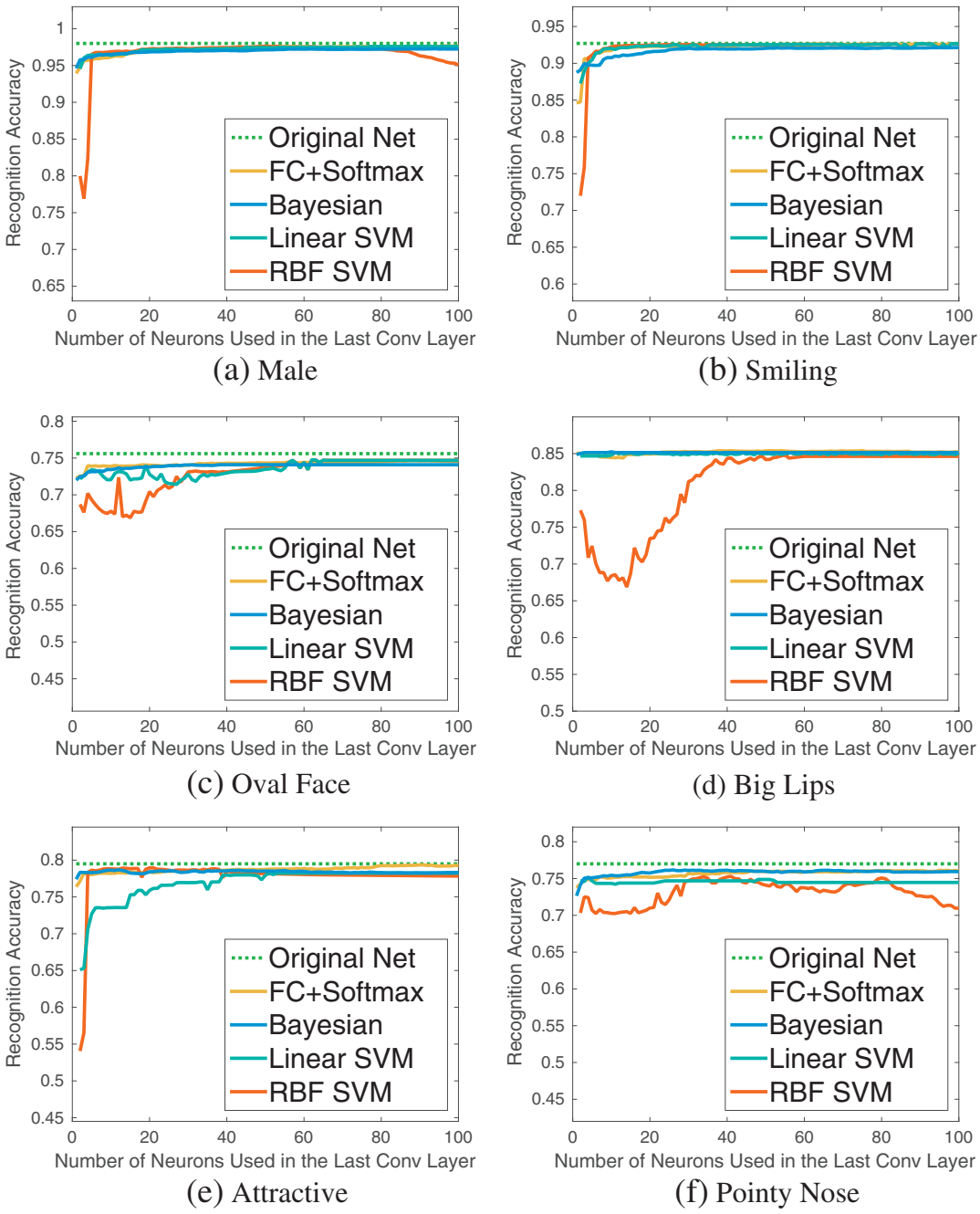

Fig. 4. Accuracy comparison of alternative classifiers using LDA-Pruned CNN features on the CelebA dataset. Horizontal axis: number of filters/neurons preserved in the last conv layer, vertical axis: recognition accuracy.

and non-related categories (i.e. gender, smile, big lips, oval face, pointy nose, and attractiveness). In Section 4.2.1 and 4.2.2, we compare our pruning approach with others in terms of accuracy and its change w.r.t. conv layer pruning rate. Section 4.2.3 provides a detailed layerwise complexity analysis of our models.

4.2.1. Recognition accuracy

To get an idea of our approach's efficacy in selecting filters/neurons, we show, in Figs. 3 and 4, accuracy changes with the number of Fisher's LDA selected last conv layer filters/neurons for different classifiers (on validation data). For comparison, the results of the original deep net and unpruned[2] FC + Softmax layers are also included.

According to Figs. 3 and 4, combined with alternative light classifiers, only a small subset of Fisher's LDA selected last conv layer neurons are discriminative enough to achieve comparable accuracies to the original net on the two datasets. There is no significant accuracy improvement with additional neurons. Occasionally, more neurons even lead to accuracy loss. This is consistent with our hypothesis that

[2] except for the first FC layer

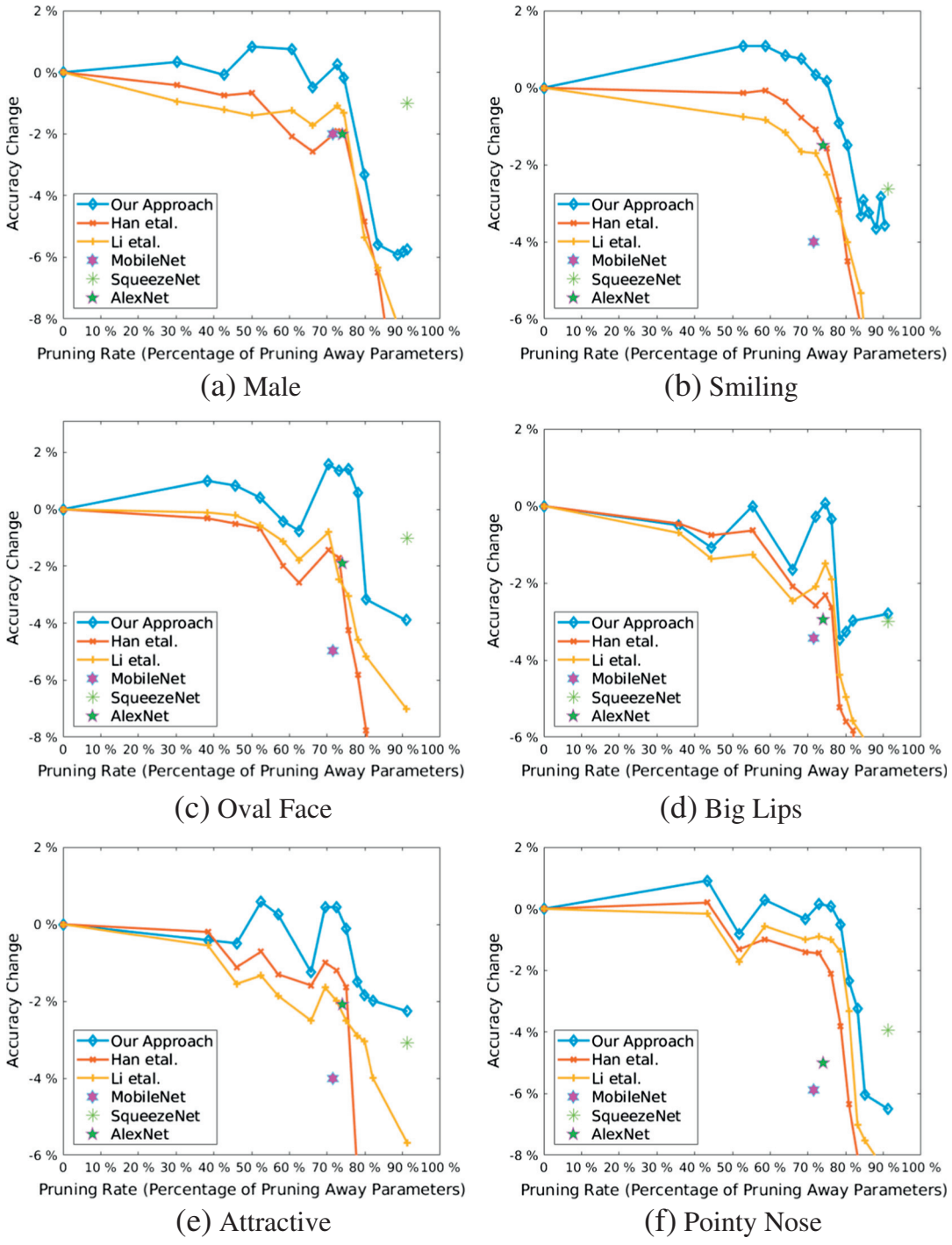

Fig. 5. Accuracy change vs. conv layers pruning rate. Comparisons: Han et al. [11], Li et al. [56], MobileNet: [60], SqueezeNet: [59], and AlexNet [34].

fine-tuned deep nets have many ineffectual and redundant structures. In general, the Bayesian classifier achieves a slightly higher accuracy than others especially on the challenging in-the-wild LFW dataset. This is because when there is significant uncertainty and noise, the Bayesian classifier can capture more information than just the margins. Although RBF SVM performs slightly better in occasional cases when there are only a few neurons, its performance is not stable. One possible reason is that after the conv layers, deep net features lie in a space in which simpler classifiers such as linear SVM and Bayesian perform well enough. With more dimensions added, the complicated RBF SVM suffers from overfitting. This is less obvious on CelebA because it is more difficult to 'memorize' a larger dataset (definition of overfitting). Since usually a few neurons are enough to obtain a comparably high accuracy, the dimensionality curse is not a problem here for the Bayesian QDA classifier. What's more, if training efficiency is one critical concern, the Bayesian classifier may be the best choice as it has fewer parameters to set than SVMs and FC+Softmax.

We select the filters/neurons (and also corresponding classifiers) that lead to a comparably high accuracy on the validation

Table 1
Performance comparison of facial traits prediction. FaceTracer: [27], ANet: [41], PANDA: [40] (with landmark position & bounding box labels), MobileNet: [60], SqueezeNet: [59], Han et al.: [11] (non-iterative), Li et.al. [56]. The (original) parameters for ANet, PANDA, VGG-16, MobileNet, and SqueezeNet are about 32 M, 84 M (all poselets), 138 M, 4.2 M, and 1.2 M and the computational complexities of them are 443 M, 30 B, 31 B, 1.1 B, 1.6 B, respectively. Our models' parameters (shared by [56,11]) and FLOPs complexities (shared by [56]), are included in the parentheses (FLOPs defined as in [11]). O.Face, B.Lips, Attra., P.Nose stand for Oval Face, Big Lips, Attractive, and Pointy Nose. For fair comparison, ground truth alignment is used (not LNets) for ANet [41] and our ANet implementation has a pooling stride of 2 and 512 first FC neurons.

Dataset	Methods	Accuracies					
		Male	Smile	O.Face	B.Lips	Attra.	P.Nose
LFWA	FaceTracer	84%	78%	66%	68%	71%	74%
	ANet	91%	84%	68%	68%	75%	74%
	PANDA	92%	89%	72%	73%	78%	76%
	VGG-16	91%	91%	74%	78%	77%	84%
	MobileNet	89%	87%	69%	72%	75%	78%
	SqueezeNet	90%	88%	71%	73%	76%	79%
	Han [11]	91%	90%	70%	75%	76%	80%
	Li [56]	91%	89%	71%	76%	76%	82%
	Ours	92%	92%	74%	77%	77%	83%
	(Param#)	(3.5 M)	(3.2 M)	(2.8 M)	(3.4 M)	(3.6 M)	(3.1 M)
	(FLOPs)	(9.4 B)	(8.0 B)	(6.6 B)	(7.5 B)	(8.3 B)	(7.7 B)
CelebA	FaceTracer	91%	89%	64%	64%	78%	68%
	ANet	95%	92%	66%	66%	80%	69%
	PANDA	97%	92%	65%	67%	81%	71%
	VGG-16	98%	93%	75%	72%	82%	77%
	MobileNet	95%	91%	73%	70%	81%	74%
	SqueezeNet	96%	91%	73%	71%	81%	74%
	Han [11]	96%	90%	71%	71%	79%	75%
	Li [56]	97%	92%	71%	72%	79%	76%
	Ours	98%	93%	74%	72%	81%	76%
	(Param#)	(2.7M)	(2.6M)	(2.5M)	(2.5M)	(2.6M)	(2.4M)
	(FLOPs)	(6.2B)	(5.9B)	(5.4B)	(5.2B)	(5.6B)	(5.6B)

set and report accuracies on the test set in Table 1. In this process, we control the size of our pruned nets so that their accuracy changes lie within $-1\% \sim +1.5\%$ of the original net's. For comparison, results of our original net as well as some previously mentioned approaches in Section 2 are included. We divide all the approaches into two large categories: (1) traditional methods and unpruned nets (e.g. FaceTracer: [27], ANet: [41], PANDA: [40], VGG-16 [7]) and (2) well-known pruned/compact structures (e.g. the 22-layer light SqueezeNet [59], the 30-layer MobileNet [60] of depth-wise separable convolutions, the weight-magnitude-based pruning approach [11], and a simple filter-level pruning approach [56] that, unlike ours, simply throws away filters of small absolute weights sum). For fair comparison, Han et al. [11], Li et al. [56] and our net have the same number of conv layer parameters for each facial trait case, which is specified in the first parentheses row for each dataset. It is also worth mentioning that in our implementation of [56], we keep the number of pruned filters the same as ours in each layer (rather than empirically determined). Therefore, Li et al. [56] and ours also have the same computational complexities, which are specified in the second parentheses row. Due to [11]'s unstructured nature, its computational complexities, loosely speaking, are the same with the original net. The parameter numbers and computational complexities of other deep models can also be found in Table 1 (caption).

As we can see from Table 1, compared to FaceTracer [27], ANet [41] and PANDA nets [40], deeper nets (pruned or unpruned), generally speaking, have higher accuracies. This is consistent with the claim that increased depth usually leads to higher accuracies [7]. The differences are especially obvious in physical shape related traits (i.e. oval face, big lips, pointy nose). The reason may be that pre-trained larger nets have more abundant primitive features in the early layers. In the category of pruned/compact structures, our approach is able to outperform Han [11] (with the same conv parameter numbers), MobileNet (similar in size), and SqueezeNet (roughly 3 times smaller), and enjoys accuracies comparable to the original VGG-16. To our surprise, SqueezeNet, being approximately 4× lighter, outperforms MobileNet in most cases on LFWA. On CelebA, their performances are similar. It is worth mentioning that lighter versions of pruned nets are possible if the accuracy is decreased from what is reported here (more details in Fig. 5). With comparably high accuracies, our pruned structures are far more efficient than the original VGG-16 in terms of space and computation. We will provide a detailed complexity analysis of our above pruned structures in the following subsections.

4.2.2. Accuracy change vs. pruning rate

In this subsection, we analyze the relationship of parameter pruning rate and accuracy change, and compare our pruned nets with the above-mentioned pruned or compact structures, i.e. Han et al. [11], Li et al. [56], MobileNet, SqueezeNet, and another conventional CNN - AlexNet [34] (without filter grouping). Fig. 5 demonstrates the comparisons on the LFWA dataset. It is worth noting that since MobileNet and SqueezeNet do not have FC layers and this paper focuses on the conv layers, the pruning rates here apply to the conv layers only.

As can be seen from Fig. 5, our approach has higher accuracies than [11] and [56] across most pruning rates for all six facial traits. Our better performance mainly stems from accounting for each neuron's contribution to the final discriminating power. Generally speaking, the performance difference is more obvious when the pruning rate is large. As mentioned before, the reason is that [11] completely ignores the relationship among weights. Effects of small weights can accumulate both within filters and across layers or be enlarged by large weights and inputs. When the pruning rate is large, the joint forces of weights become more vulnerable. Pruning away small but important weights at relatively early stages explains the early degradations of [11]'s performance. This also explains why [56] performs a little better than [11] when the pruning rate is large. In addition, accuracy can even improve during pruning using our approach, which shows our method's potential as a way to design

optimal structures for a given task. In general, about a quarter of conv layer weights are enough to maintain a comparable discriminating power. When the original net is pruned to about the same size as AlexNet [34] (conv layers) or MobileNet, our approach always enjoys higher accuracies. In the next subsection, we will provide more insight into our pruning method's space and computational efficiency.

4.2.3. Complexity analysis

This subsection offers a detailed layerwise space and computation complexity analysis of our CNN structures used in Table 1 (take the LFWA cases for example). The structural complexities are shared by [56]. As for [11], there are hardly any direct structured savings in space and computation.

4.2.3.1. Space complexity. Fig. 6 demonstrates the layerwise space complexity. It can be seen that, most parameters in the middle conv layers (Conv2_2 to Conv4_1) do not help with our task. Compared to later layers, the first three layers have relatively low reduction rates. This is easy to understand given the observation that earlier layers contain more generic features such as edge and color blob detectors that could be useful to all classes. However, it is worth noting that even though the numbers of filters in a particular layer are similar for some traits, the corresponding patterns can be very different (especially in the last few layers). Through analyzing

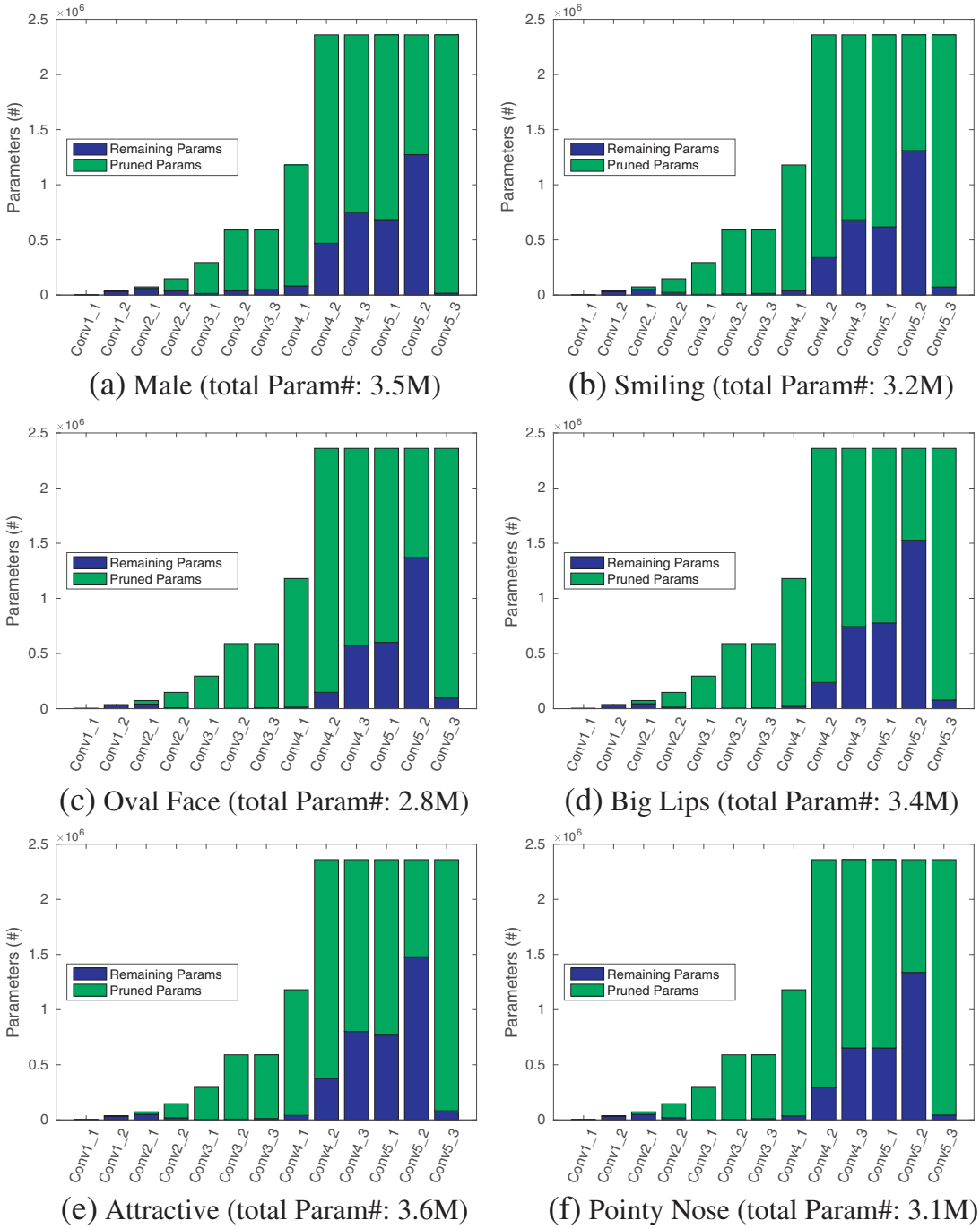

Fig. 6. Demonstration of layerwise structure complexity reduction. Green indicates parameters pruned away, blue represents remaining parameters.

discarded neurons, we find that some stand for less relevant concepts (e.g. texts, water, sky) to our fine-grained recognition tasks. Our pruning approach abandons all their weights regardless of their magnitudes. Furthermore, unlike [11], our approach's high pruning rate can directly contribute to low memory requirements because it enables us to discard (rather than disregard by setting to 0) filter weights. Besides pruning the conv layers, it's worth mentioning that replacing FC layers plays a critical role in bringing the model size down. For the Bayesian classifier, the storage overhead can be ignored when only a small set of neurons are used (even when all neurons are utilized in the last conv layer of VGG-16, the extra space required is just about 2 MB). For SVM, the extra storage needed depends on the number of trained support vectors. In our experiments, it is only about 30 KB. Since our LDA-deconv-pruned CNNs focus on the highest activations, they are more robust to noise (no accuracy loss is incurred when we use the FP16 precision in our experiment). Compared to the original deep models of 500 MB, our pruned models with comparable accuracies are very light and usually take up at most 15 MB in space (7.5 MB when using the FP16 precision). Given the fact that most of today's latest cellphone models have only 1–2 GB RAM and that most PCs have only less than 20 MB CPU caches, the low space requirements of our pruned nets are

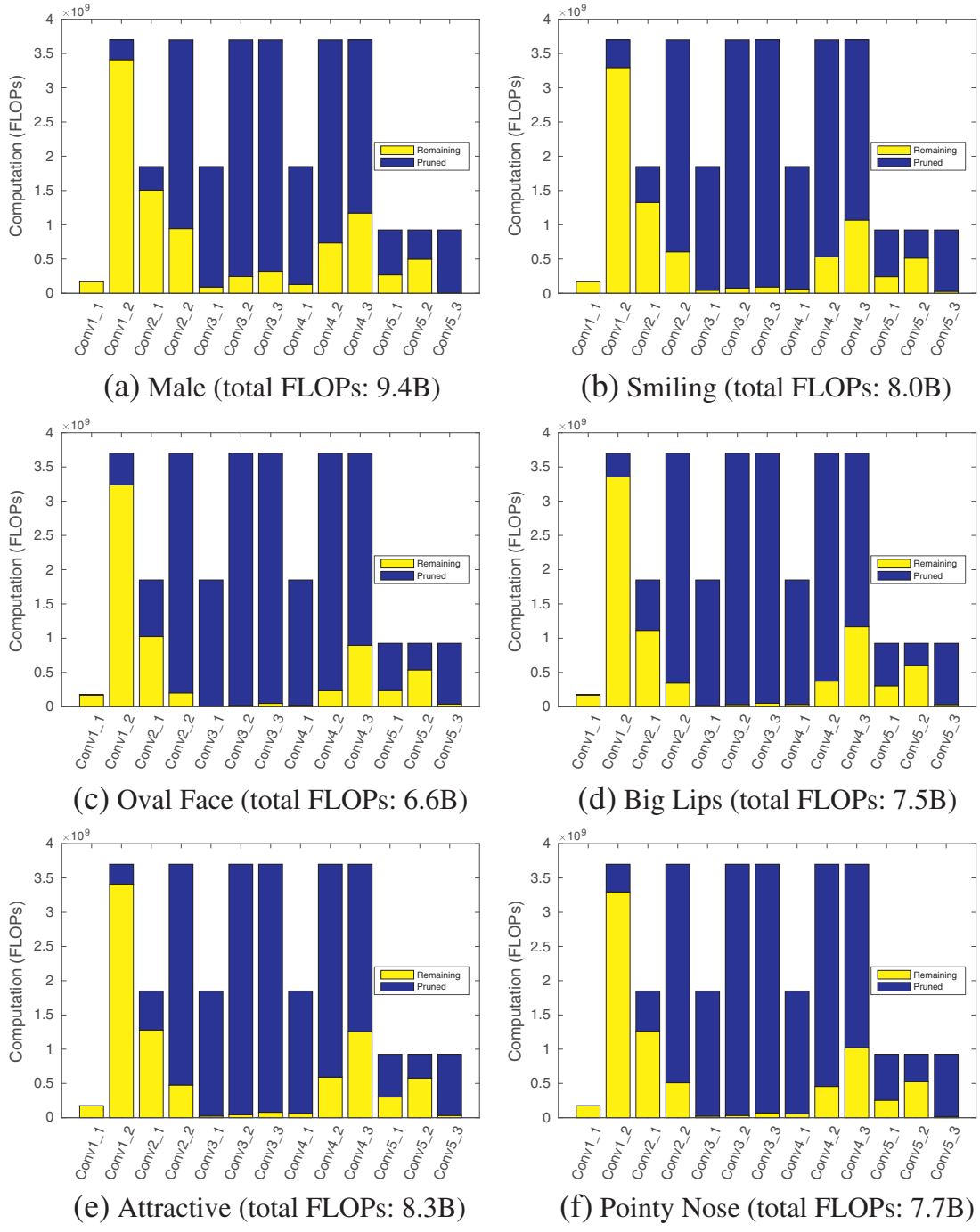

Fig. 7. Demonstration of layerwise computation (FLOP) savings. Blue indicates computations saved by pruning, yellow represents remaining computations.

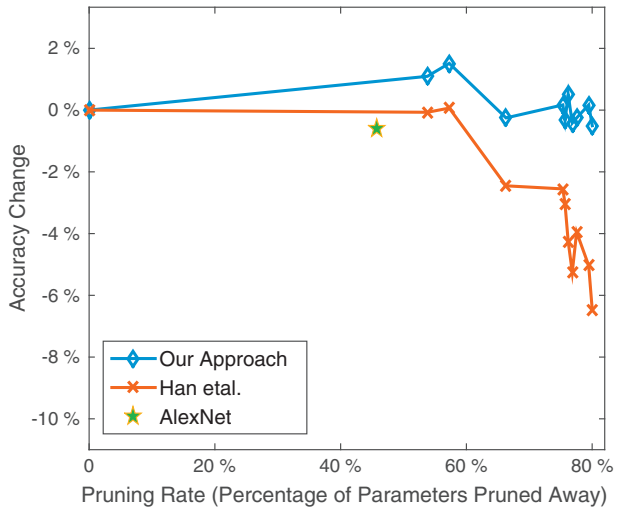

Fig. 8. Accuracy vs. conv pruning rate of GoogleNet on the multi-category race trait.

critical if we want to boost performance significantly by going from off-chip to on-chip.

4.2.3.2. Computation complexity. In this part, we adopt the number of floating-point operations (FLOPs) to measure computational complexity. As in [11], both multiplication and addition count for one FLOP in our calculation [3]. FLOP is a generic measure that, compared to direct timing, is less dependent on hardware specifics and implementation details, such as base frequency, cache structures, memory scheduling, and even temperature. Fig. 7 demonstrates FLOP savings across all conv layers. As we can see, our LDA-Pruned CNNs require much fewer computations than the original net (over 30B FLOPs) across the layers. As a pruning approach that considers cross-layer dependencies, ours preserves most primitive features in early layers and helps discover the middle 'bottleneck' layers. It is worth noting that the light alternative classifiers to FC layers (e.g. linear SVM and Bayesian QDA) are computationally negligible. Their computations are three orders of magnitude fewer than the original FC layers (which are already computationally efficient).

4.3. GoogLeNet on extended LFWA

In this section, we will show our pruning method's efficacy on modular deep structures. They are different from conventional deep nets (e.g. VGG-16) in that each module is composed of different filter sizes. By pruning such module-based nets, we can select both the types of filters and the filter number of each kind. Our pruning idea is the same, except now we need to do the deconv dependency tracing and pruning along each module branch. Additionally, multiple branches in each module share the same input and their outputs are combined as input to the next module. Although our approach is generally applicable to various modular structures, we take GoogLeNet (a.k.a. Inception V1) as an example (the state-of-the-art ResNet can be pruned similarly because summation at a module end can be considered as concatenation followed by convolution). In contrast to the binary traits in Section 4.2, we focus on the four-category race trait (White, Black, Asian, and others) from the extended LFWA. Fig. 8 and Table 2 show how validation accuracy changes with the remaining conv parameters and FLOPs. Figs. 9 and 10 show the layerwise

[3] some works have different measurements of FLOPs. For example, in [9], one multiplication and one accumulation together count for one FLOP.

Table 2
Accuracy change v.s. left params and FLOPs (all: 6 M params, 3.2B FLOPs).

Param#	FLOPs	Acc change
2.55 M	2.02 B	+1.52%
2.01 M	1.72 B	−0.25%
1.47 M	1.42 B	+0.17%
1.42 M	1.38 B	+0.05%
1.38 M	1.35 B	−0.42%
1.22 M	1.24 B	+0.16%
1.19 M	1.21 B	−0.51%

parameter and FLOP complexities of the second-to-lightest model in Table 2 that performs comparably to the unpruned model on the test set.

4.4. Pruned facial trait features for facial identification with limited data

Most big facial identity datasets, which are indispensable to effectively learning a general face embedding, are the property of large corporations (e.g. DeepFaces [93], FaceNet [94]) and are not publicly available. In this paper, we consider facial identity as a group of identity-related facial traits (e.g. race) and use our structure pruned for facial trait classification to assist facial identification where data is limited (w.r.t. both images per identity and total images). After all, it is easier to collect 10,000 images of Asian people than to collect 10,000 images of one identity. As an simple example, we choose our pruned structure in Figs. 9 and 10 (for race recognition) to train and classify some unseen identities from LFW (subjects with over 50 images in the test split [41]). All identities here have not been seen before during previous training and pruning since there is no identity overlap between different LFWA splits [41]. We select the images with serial numbers greater than 10 for training while use others for testing. There are altogether 910 training images and 59 testing images from 6 identities. We compare our net with MobileNet, SqueezeNet, and the original GoogLeNet. The results are shown in Table 3.

As we can see, net structure does have an impact on performance. When trained from scratch (random initialization), our structure (derived via pruning for the identity-related facial trait) is clearly better than the other three designed for generic object (ImageNet) recognition. The margins are especially large between ours and the compact MobileNet and SqueezeNet, which highlights the value of our task-specific pruning and demonstrates that our pruned net structure for the identity-related facial trait is helpful for the facial identification task. GoogLeNet performs better than MobileNet and SqueezeNet because of its larger capacity. When the training is from pre-trained initializations, the performance differences between the nets are reduced. The competing compact nets get significantly better but their accuracies are still a few percent worse than ours. Although the three compared structures are designed for ImageNet recognition, models with larger capacities tend to 'digest' the facial trait information better. For the smallest SqueezeNet, the 'knowledge' pre-trained from ImageNet is more helpful than the facial trait knowledge even though the latter is more related to the current task of facial identity recognition.

5. Discussion and future directions

Structured pruning like ours can result in direct space and computational savings, which is critical for biometrics-in-the-wild algorithms because they are usually designed for mobile devices without a powerful GPU and large memory. Even for desktop applications, it greatly alleviates the pain of waiting for large nets to be loaded (from

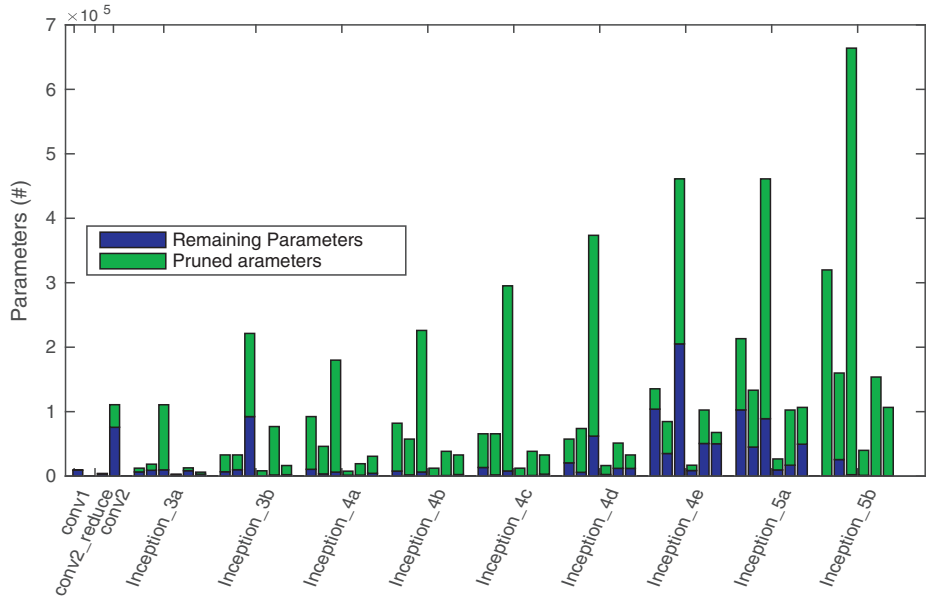

Fig. 9. Layerwise param complexity reduction of GoogLeNet. Green: parameters pruned away, blue: remaining parameters. Each Inception module has six conv layers. From left to right, they are: 1*1, 1*1 followed by 3*3, 1*1 followed by 5*5, 1*1 (after pooling) conv layers.

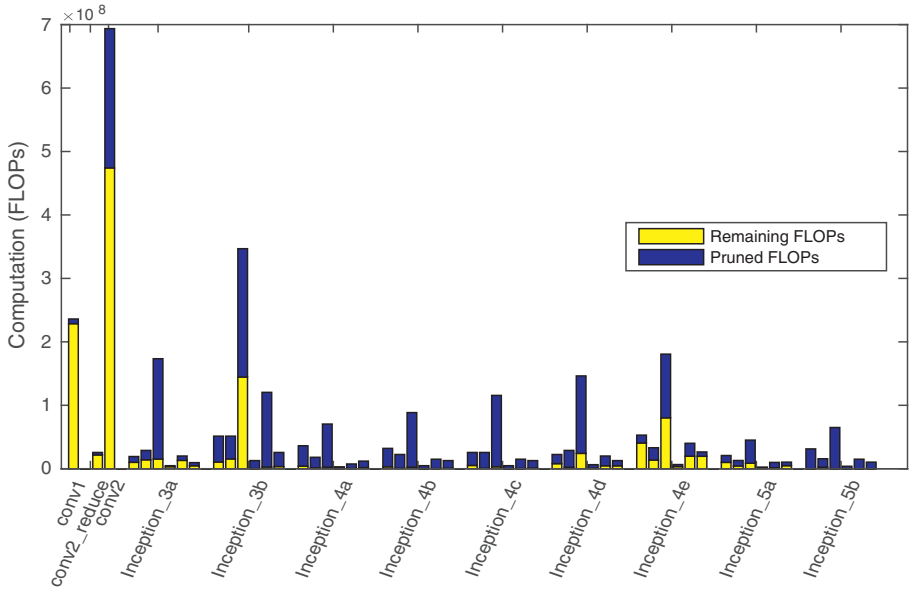

Fig. 10. Layerwise computation (FLOP) savings of GoogleNet. Blue: FLOPs pruned, yellow: remaining FLOPs. Each Inception module has six conv layers. From left to right, they are 1*1, 1*1 followed by 3*3, 1*1 followed by 5*5, 1*1 (following pooling) conv layers.

either a hard drive or the Internet) and deployed on the memory. In addition, unlike generic compact nets that can hardly be optimal for all tasks, our pruning approach provides a feasible and flexible way to search for optimal network architectures given a task while being mindful of both parameter and computational complexities. Many compact modular nets reduce dimensionality by employing a random set of 1*1 filters, but we achieve informed dimensionality reduction in the feature (map) space via filter-level pruning with a final classification related utility measure.

6. Conclusion

In this paper, we have proposed a Fisher's LDA based CNN pruning approach that can boost efficiency while maintaining or even improving accuracy for facial trait classification. It is found that most last conv layer neurons tend to fire uncorrelatedly. Through Fisher LDA, the neurons in dimensions that have low ICC were safely

Table 3
Performance comparison of competing nets in the task of facial identity prediction. Parameter# and FLOPs are of conv layers.

Net	(Param#, FLOPs)	Pre-trained		From scratch
		FacialTrait	ImageNet	
MobileNet	(3.21 M, 1.14 B)	96.6%	96.6%	62.7%
SqueezeNet	(0.75 M, 1.55 B)	93.2%	94.9%	74.6%
GoogLeNet	(5.97 M, 3.16 B)	98.3%	96.6%	88.1%
Our pruned	(1.22 M, 1.24 B)	100%	–	91.5%

discarded, thereby greatly pruning the whole network (through deconv tracing) and significantly increasing efficiency. As the result, the approach can be useful in contexts where fast and accurate performance is desirable but where expensive GPUs are not available (e.g. embedded systems). Our LDA based pruning is superior to many state-of-the-art approaches (e.g. [11,56]) in that we take into account both within filter and across layer relationships that are related to final classification. Unlike fixed compact MobileNet or SqueezeNet, our approach provides a way to design such structures. By combining with alternative classifiers, the pruning approach is shown to achieve comparable accuracies to the original net on the LFW and CelebA datasets, but with huge space savings (96%–98 % for VGG-16, 81% for GoogLeNet) and large computational reductions (as high as 80% for VGG-16, 61% for GoogLeNet).

Acknowledgment

The authors gratefully acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and McGill MEDA Award. We would also like to acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research.

References

[1] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugenics 7 (2) (1936) 179–188.
[2] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[3] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks, Adv. Neural Inf. Proces. Syst. 19 (2007) 153.
[4] C.P. MarcAurelio Ranzato, S. Chopra, Y. LeCun, Efficient learning of sparse representations with an energy-based model, Proceedings of NIPS, 2007.
[5] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R.R. Salakhutdinov, Improving Neural Networks by Preventing Co-adaptation of Feature Detectors, arXiv preprint arXiv:1207.0580, 2012.
[6] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[7] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, Proceedings of the International Conference on Learning Representations (ICLR) 2015, 2015.
[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. pp. 1–9.
[9] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, 2015.arXiv preprint arXiv:1512.03385.
[10] Q. Tian, T. Arbel, J.J. Clark, Deep LDA-pruned nets for efficient facial gender classification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshop on Biometrics), IEEE. 2017, pp. 512–521.
[11] S. Han, J. Pool, J. Tran, W. Dally, Learning both weights and connections for efficient neural network, Advances in Neural Information Processing Systems, 2015. pp. 1135–1143.
[12] M.A. Turk, A.P. Pentland, Face recognition using eigenfaces, Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on, IEEE. 1991, pp. 586–591.
[13] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, Eigenfaces vs. fisherfaces: recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.
[14] T. Ahonen, A. Hadid, M. Pietikäinen, Face recognition with local binary patterns, European Conference on Computer Vision, Springer. 2004, pp. 469–481.
[15] T. Ahonen, A. Hadid, M. Pietikainen, Face description with local binary patterns: application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 2037–2041.
[16] D.G. Lowe, Object recognition from local scale-invariant features, Computer vision, 1999. The Proceedings of the Seventh IEEE International Conference on, vol. 2, Ieee. 1999, pp. 1150–1157.
[17] N. Kumar, A.C. Berg, P.N. Belhumeur, S.K. Nayar, Attribute and simile classifiers for face verification, Computer Vision, 2009 IEEE 12th International Conference on, IEEE. 2009, pp. 365–372.
[18] V. Štruc, N. Pavešić, Gabor-based kernel partial-least-squares discrimination features for face recognition, Informatica 20 (1) (2009) 115–138.
[19] C. Perez, J. Tapia, P. Estévez, C. Held, Gender classification from face images using mutual information and feature fusion, Int. J. Optomechatronics 6 (1) (2012) 92–119.
[20] A.J. O'toole, T. Vetter, N.F. Troje, H.H. Bülthoff, Sex classification is better with three-dimensional head structure than with image intensity information, Perception 26 (1) (1997) 75–84.
[21] P.J. Phillips, H. Wechsler, J. Huang, P.J. Rauss, The FERET database and evaluation procedure for face-recognition algorithms, Image Vis. Comput. 16 (5) (1998) 295–306.
[22] I. Ullah, M. Hussain, G. Muhammad, H. Aboalsamh, G. Bebis, A.M. Mirza, Gender recognition from face images with local WLD descriptor, 19th International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE. 2012, pp. 417–420.
[23] C. Shan, Learning local binary patterns for gender classification on real-world face images, Pattern Recogn. Lett. 33 (4) (2012) 431–437.
[24] G.B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments, Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.
[25] B. Moghaddam, M.-H. Yang, Learning gender with support faces, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 707–711.
[26] Y. Freund, R.E. Schapire, et al. Experiments with a new boosting algorithm, Icml, vol. 96, 1996. pp. 148–156. Bari, Italy.
[27] N. Kumar, P. Belhumeur, S. Nayar, Facetracer: a search engine for large collections of images with faces, European conference on computer vision, Springer. 2008, pp. 340–353.
[28] M. Toews, T. Arbel, Detection, localization, and sex classification of faces from arbitrary viewpoints and under occlusion, IEEE Trans. Pattern Anal. Mach. Intell. 31 (9) (2009) 1567–1581.
[29] E. Makinen, R. Raisamo, Evaluation of gender classification methods with automatically detected and aligned faces, IEEE Trans. Pattern Anal. Mach. Intell. 30 (3) (2008) 541–547.
[30] D. Reid, S. Samangooei, C. Chen, M. Nixon, A. Ross, Soft Biometrics for Surveillance: An Overview, Machine Learning: Theory and Applications, Elsevier. 2013, 327–352.
[31] B.A. Golomb, D.T. Lawrence, T.J. Sejnowski, SEXNET: A Neural Network Identifies Sex From Human Faces., NIPS, vol. 1, 1990. pp. 2.
[32] B. Poggio, R. Brunelli, T. Poggio, HyberBF Networks for Gender Classification, 1992.
[33] S. Gutta, H. Wechsler, Gender and ethnic classification of human faces using hybrid classifiers, Neural Networks, 1999. IJCNN'99. International Joint Conference on, vol. 6, IEEE. 1999, pp. 4084–4089.
[34] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in Neural Information Processing Systems, 2012. pp. 1097–1105.
[35] A. Verma, L. Vig, Using convolutional neural networks to discover cognitively validated features for gender classification, Soft Computing and Machine Intelligence (ISCMI), 2014 International Conference on, IEEE. 2014, pp. 33–37.
[36] E. Eidinger, R. Enbar, T. Hassner, Age and gender estimation of unfiltered faces, IEEE Trans. Inf. Forensics Secur. 9 (12) (2014) 2170–2179.
[37] G. Levi, T. Hassner, Age and gender classification using convolutional neural networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015. pp. 34–42.
[38] J. Mansanet, A. Albiol, R. Paredes, Local deep neural networks for gender recognition, Pattern Recogn. Lett. 70 (2016) 80–86.
[39] S. Li, J. Xing, Z. Niu, S. Shan, S. Yan, Shape driven kernel adaptation in convolutional neural network for robust facial traits recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. pp. 222–230.
[40] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, L. Bourdev, Panda: pose aligned networks for deep attribute modeling, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014. pp. 1637–1644.
[41] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, Proceedings of International Conference on Computer Vision (ICCV), 2015.
[42] Y. Sun, X. Wang, X. Tang, Deeply learned face representations are sparse, selective, and robust, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. pp. 2892–2900.
[43] L.Y. Pratt, Comparing Biases for Minimal Network Construction with Backpropagation, vol. 1. Morgan Kaufmann Pub. 1989.
[44] Y. LeCun, J.S. Denker, S.A. Solla, R.E. Howard, L.D. Jackel, Optimal brain damage., NIPs, vol. 2, 1989. pp. 598–605.
[45] B. Hassibi, D.G. Stork, Second Order Derivatives for Network Pruning: Optimal Brain Surgeon, Morgan Kaufmann. 1993.
[46] R. Reed, Pruning algorithms-a survey, IEEE Trans. Neural Netw. 4 (5) (1993) 740–747.
[47] S. Han, H. Mao, W.J. Dally, Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding, 2015.CoRR, abs/1510.00149 2.
[48] S. Srinivas, R.V. Babu, Data-free Parameter Pruning for Deep Neural Networks, 2015.arXiv preprint arXiv:1507.06149.
[49] Z. Mariet, S. Sra, Diversity Networks, ICLR, 2016.
[50] X. Jin, X. Yuan, J. Feng, S. Yan, Training Skinny Deep Neural Networks with Iterative Hard Thresholding Methods, 2016.arXiv preprint arXiv:1607.05423.
[51] Y. Guo, A. Yao, Y. Chen, Dynamic network surgery for efficient DNNs, Advances In Neural Information Processing Systems, 2016. pp. 1379–1387.
[52] H. Hu, R. Peng, Y.-W. Tai, C.-K. Tang, Network Trimming: A Data-driven Neuron Pruning Approach Towards Efficient Deep Architectures, 2016.arXiv preprint arXiv:1607.03250.
[53] V. Sze, T.-J. Yang, Y.-H. Chen, Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning, 2017, 5687–5695.

[54] S. Anwar, K. Hwang, W. Sung, Structured Pruning of Deep Convolutional Neural Networks, 2015.arXiv preprint arXiv:1512.08571.
[55] A. Polyak, L. Wolf, Channel-level acceleration of deep face representations, IEEE Access 3 (2015) 2163–2175.
[56] H. Li, A. Kadav, I. Durdanovic, H. Samet, H.P. Graf, Pruning Filters for Efficient Convnets, 2016.arXiv preprint arXiv:1608.08710.
[57] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, Xnor-net: Imagenet classification using binary convolutional neural networks, European Conference on Computer Vision, Springer. 2016, pp. 525–542.
[58] M. Lin, Q. Chen, S. Yan, Network in Network, ICLR, 2014.
[59] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, K. Keutzer, SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and< 0.5 MB Model Size, 2016.arXiv preprint arXiv:1602.07360.
[60] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv preprint arXiv:1704.04861, 2017.
[61] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. pp. 770–778.
[62] M. Courbariaux, Y. Bengio, J.-P. David, Binaryconnect: training deep neural networks with binary weights during propagations, Advances in Neural Information Processing Systems, 2015. pp. 3123–3131.
[63] J. Wu, C. Leng, Y. Wang, Q. Hu, J. Cheng, Quantized convolutional neural networks for mobile devices, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. pp. 4820–4828.
[64] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, D. Shin, Compression of deep convolutional neural networks for fast and low power mobile applications, ICLR, 2016.
[65] B. Liu, M. Wang, H. Foroosh, M. Tappen, M. Pensky, Sparse convolutional neural networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. pp. 806–814.
[66] B. Reagen, P. Whatmough, R. Adolf, S. Rama, H. Lee, S.K. Lee, J.M. Hernández-Lobato, G.-Y. Wei, D. Brooks, Minerva: enabling low-power, highly-accurate deep neural network accelerators, Proceedings of the 43rd International Symposium on Computer Architecture, IEEE Press. 2016, pp. 267–278.
[67] J. Ba, R. Caruana, Do deep nets really need to be deep? Advances in Neural Information Processing Systems, 2014. pp. 2654–2662.
[68] L.G. Valiant, A quantitative theory of neural computation, Biol. Cybern. 95 (3) (2006) 205–211.
[69] V.B. Mountcastle, et al. Modality and topographic properties of single neurons of cats somatic sensory cortex, J. Neurophysiol. 20 (4) (1957) 408–434.
[70] V.B. Mountcastle, The columnar organization of the neocortex., Brain 120 (4) (1997) 701–722.
[71] P.S. Goldman, W.J. Nauta, Columnar distribution of cortico-cortical fibers in the frontal association, limbic, and motor cortex of the developing rhesus monkey, Brain Res. 122 (3) (1977) 393–413.
[72] J. Lübke, A. Roth, D. Feldmeyer, B. Sakmann, Morphometric analysis of the columnar innervation domain of neurons connecting layer 4 and layer 2/3 of juvenile rat barrel cortex, Cereb. Cortex 13 (10) (2003) 1051–1063.
[73] V. Egger, T. Nevian, R.M. Bruno, Subcolumnar dendritic and axonal organization of spiny stellate and star pyramid neurons within a barrel in rat somatosensory cortex, Cereb. Cortex 18 (4) (2008) 876–889.
[74] T.A. Woolsey, H. Van der Loos, The structural organization of layer IV in the somatosensory region (SI) of mouse cerebral cortex: the description of a cortical field composed of discrete cytoarchitectonic units, Brain Res. 17 (2) (1970) 205–242.
[75] D.H. Hubel, T.N. Wiesel, Uniformity of monkey striate cortex: a parallel relationship between field size, scatter, and magnification factor, J. Comp. Neurol. 158 (3) (1974) 295–305.
[76] T.D. Albright, R. Desimone, C.G. Gross, Columnar organization of directionally selective cells in visual area MT of the macaque, J. Neurophysiol. 51 (1) (1984) 16–31.
[77] G.J. Rinkus, A cortical sparse distributed coding model linking mini-and macrocolumn-scale functionality, Front. Neuroanat. 4 (17). (2010)
[78] G.J. Rinkus, Sparsey: event recognition via deep hierarchical sparse distributed codes, Front. Comput. Neurosci. 8 (2014)
[79] A. Babenko, V. Lempitsky, Aggregating local deep features for image retrieval, Proceedings of the IEEE international conference on computer vision, 2015. pp. 1269–1277.
[80] Y. Zhong, J. Sullivan, H. Li, Leveraging mid-level deep representations for predicting face attributes in the wild, Image Processing (ICIP), 2016 IEEE International Conference on, IEEE. 2016, pp. 3239–3243.
[81] Y. Li, J. Kittler, J. Matas, Effective implementation of linear discriminant analysis for face recognition and verification, Computer Analysis of Images and Patterns, Springer. 1999, pp. 234.
[82] A. Jain, J. Huang, S. Fang, Gender identification using frontal facial images, 2005 IEEE International Conference on Multimedia and Expo, IEEE. 2005, pp. 4–pp.
[83] J. Bekios-Calfa, J.M. Buenaposada, L. Baumela, Revisiting linear discriminant techniques in gender recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (4) (2011) 858–864.
[84] Z. Jin, J.-Y. Yang, Z.-S. Hu, Z. Lou, Face recognition based on the uncorrelated discriminant transformation, Pattern Recogn. 34 (7) (2001) 1405–1416.
[85] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, H. Lipson, Understanding Neural Networks through Deep Visualization, arXiv preprint arXiv:1506.06579, 2015.
[86] M.D. Zeiler, G.W. Taylor, R. Fergus, Adaptive deconvolutional networks for mid and high level feature learning, 2011 International Conference on Computer Vision, IEEE. 2011, pp. 2018–2025.
[87] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, European Conference on Computer Vision, Springer. 2014, pp. 818–833.
[88] D.O. Hebb, The Organization of Behavior: A Neuropsychological Theory, Psychology Press. 2005.
[89] Y. Tang, Deep Learning using Linear Support Vector Machines, 2013.arXiv preprint arXiv:1306.0239.
[90] A. Sharif Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014. pp. 806–813.
[91] R. Rothe, R. Timofte, L.V. Gool, Deep expectation of real and apparent age from a single image without facial landmarks, Int. J. Comput. Vis. (IJCV) (2016)
[92] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, Proceedings of the 22nd ACM International Conference on Multimedia, ACM. 2014, pp. 675–678.
[93] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Deepface: closing the gap to human-level performance in face verification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014. pp. 1701–1708.
[94] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: a unified embedding for face recognition and clustering, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. pp. 815–823.