On the Size of Convolutional Neural Networks and Generalization Performance

Maya Kabkab*†, Emily Hand*, Rama Chellappa*†
Center for Automation Research, UMIACS*
Department of Electrical and Computer Engineering†
University of Maryland, College Park
Email: {mayak, emhand, rama}@umiacs.umd.edu

***Abstract*—While Convolutional Neural Networks (CNNs) have recently achieved impressive results on many classification tasks, it is still unclear why they perform so well and how to properly design them. In this work, we investigate the effect of the convolutional depth of a CNN on its generalization performance for binary classification problems. We prove a sufficient condition —polynomial in the depth of the CNN— on the training database size to guarantee such performance. We empirically test our theory on the problem of gender classification and explore the effect of varying the CNN depth, as well as the training distribution and set size.**

I. Introduction

Convolutional Neural Networks (CNNs) are now widely used for classification problems due to their state-of-the-art performance (see, e.g., [1], [2]). However, one important challenge, which remains an open problem, is how to size them appropriately. When designing a CNN, the most common approach is to experiment with the depth (and many other parameters), until a suitable model is found. It is known that if the CNN is *too shallow*, then it may not correctly represent the underlying relationship between the input and its corresponding class (i.e., under-fit). If it is *too deep*, however, it may follow irrelevant properties of the dataset on which it is trained (i.e., over-fit). In this paper, we try to address this problem by investigating the relationship between the depth of a CNN and its generalization performance using approaches from statistical learning theory.

Recently, CNNs have drawn much needed attention, and a lot of empirical work has attempted to understand why they perform so well [3], [4] as well as how to properly design them [5], [6]. However, from a theoretical perspective, CNNs are still not completely understood. While theoretical results on deep architectures exist [7]–[10], they are almost always restricted to feedforward neural networks.

In this paper, we investigate the effect of CNN depth on its generalization performance. Specifically, we ask the question of how to pick a suitable CNN depth given a training database size. We assume that the examples are drawn according to an arbitrary, fixed, probability distribution, and that the learning algorithm will produce a CNN which will correctly predict on a substantial fraction of the training set. We are concerned with how the same CNN will perform on unseen (testing) samples, drawn from the same, or a slightly different, distribution. Our work is based on the VC dimension, which was first introduced in [11], [12] and provided a mathematical foundation for answering such questions. We follow an approach similar to [7], which is specific to feedforward networks, but extend it for the case of CNNs. We restrict our study to the problem of binary classification in which the set of possible labels contains only two elements, e.g., 0 and 1.

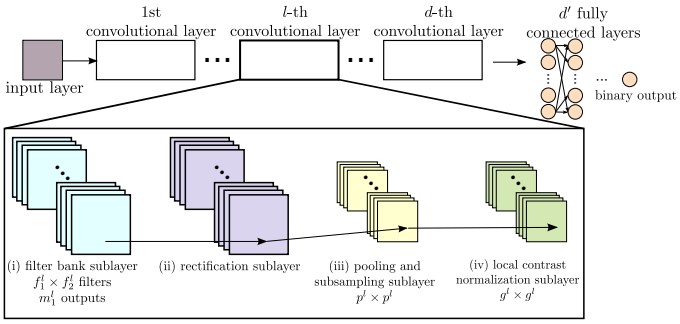

Fig. 1. Model architecture of a CNN with d convolutional layers and d' fully connected layers.

We show that, if the training and testing sampling distributions are the same, a sufficient condition to guarantee valid generalization is for the CNN training set size to be some constant times d^4 where d is the depth of the convolutional layers. We also show how to generalize the condition for different training and testing distributions. We empirically demonstrate that these conditions are sufficient but often not necessary, and examine the behavior of the testing error as we vary the CNN depth, the training distribution, and set size.

The paper is organized as follows: section II introduces the CNN model architecture under consideration, section III develops the mathematical framework as well as the theoretical results, and finally, section IV provides experimental results on the binary problem of gender classification.

II. Network architecture

In this work, we consider an architecture similar to the one presented in [13]. As shown in Figure 1, a CNN of depths (d, d') consists of d convolutional layers and d' fully connected layers. The l-th layer of a CNN is composed of the following:

(i) *a filter bank sublayer*, which takes as input $\mathbf{x}^l$, a 3D array with n_1^l 2D feature maps of size $n_2^l \times n_3^l$ each, and outputs a 3D array with m_1^l 2D feature maps of size $m_2^l \times m_3^l$ each. The size of the output maps is determined by the size $f_1^l \times f_2^l$ of the convolution filters and is given by $m_2^l = n_2^l - f_1^l + 1$ and $m_3^l = n_3^l - f_2^l + 1$. Filter k_{ij}^l

connects the i-th input feature map $\mathbf{x}_i^l$ to the j-th output feature map: $\mathbf{y}_j^l = a_j^l \cdot \tanh\left(\sum_i k_{ij}^l * \mathbf{x}_i^l\right)$. The filters and the coefficients $\{a_j^l\}$ are trainable parameters.

(ii) *a rectification sublayer*, which only retains positive inputs: $\bar{y}_{ijk}^l = \max\{0, y_{ijk}^l\}$.

(iii) *a pooling and subsampling sublayer*, which keeps the maximum (or the average) from each $p^l \times p^l$ window and outputs $\bar{\bar{\mathbf{y}}}^l$.

(iv) *a local contrast normalization sublayer*, which performs the following operations: $v_{ijk}^l = \bar{\bar{y}}_{ijk}^l - \sum_{i,p,q} w_{pq} \cdot \bar{\bar{y}}_{i,j+p,k+q}^l$, where w is a Gaussian window of size $g^l \times g^l$. Then, $\bar{\bar{\bar{y}}}_{ijk}^l (= x_{ijk}^{l+1}) = \frac{v_{ijk}^l}{\max\left\{\mu^l, \sigma_{jk}^l\right\}}$, where $\sigma_{jk}^l = \sum_{ipq} w_{pq} \cdot (v_{i,j+p,k+q}^l)^2$ and $\mu^l = \text{mean}(\sigma_{jk}^l)$.

The d' fully connected layers have a fixed structure and trainable weights $\mathbf{W}_f$. In the rest of the paper, we will assume that d' is fixed and study the effect of varying d on the classifier's generalization performance. As mentioned earlier, we restrict our study to binary classification, i.e., CNNs which implement a function that maps samples from the input domain I, to a boolean value in $\{0, 1\}$.

III. Relationship between depth and generalization performance

A. Problem formulation

In this paper, we are interested in characterizing how the depth of a CNN affects its generalization performance. Formally, we let $\mathcal{C}^d$ be the set of convolutional neural networks with d convolutional layers, for some fixed values of $\{n_1^l, n_2^l, n_3^l, m_1^l, f_1^l, f_2^l, p^l, g^l\}_{l=1}^d$, as defined in section II above. This set includes all such CNNs realized by varying the learned parameters $\{a_j^l, k_{ij}^l\}_{i,j,l} \cup \{\mathbf{W}_f\}$. As with any supervised learning algorithm, a CNN learning algorithm starts with a training set $S = \{x_1, x_2, \ldots, x_{|S|}\} \subseteq I$, assumed to be drawn at random according to a fixed but arbitrary distribution $\mathcal{D}_S$ on the input domain I. The aim of the algorithm is to find a suitable CNN $c \in \mathcal{C}^d$ which agrees with the *ground truth*, or target, hypothesis $h^* : I \to \{0, 1\}$ as much as possible. It is assumed that the true labels of the training samples, i.e., $h^*(x_1), h^*(x_2), \ldots, h^*(x_{|S|})$, are known. The resulting CNN c will have an empirical training error given by:

$$\hat{e}_S(c) \triangleq \frac{1}{|S|} \sum_{i=1}^{|S|} 1\left(h_c(x_i) \neq h^*(x_i)\right), \tag{1}$$

where $1(.)$ is the indicator function and $h_c(.)$ is the boolean function implemented by the CNN c. Clearly, $\hat{e}_S(c)$ is a random variable since the set S is chosen at random. However, if the learning algorithm is designed properly, $\hat{e}_S(c)$ will tend to be small. This does not, however, provide any guarantee as to how the CNN classifier will perform on test samples. We assume that testing samples are drawn at random according to a distribution $\mathcal{D}_T$. We are thus interested in the average performance of c on these new samples:

$$e_T(c) \triangleq \Pr_{\mathcal{D}_T}\left[h_c(x) \neq h^*(x)\right], \tag{2}$$

where x is a random sample picked according to $\mathcal{D}_T$.

B. Same training and testing distribution

We first look at the case when the training and testing sampling distributions are the same, i.e., $\mathcal{D}_S \sim \mathcal{D}_T$. As previously stated, a CNN c (or its corresponding boolean function $h_c(.)$), which is accurate on the training set (i.e., has small $\hat{e}_S(c)$), might not necessarily be accurate on new examples which are not in the training set, even if the new examples are drawn from the same distribution. In this case, we are interested in performance guarantees on $e_T(c) = e_S(c)$, whenever $\hat{e}_S(c)$ is small. To this end, we first state Lemma 1 which computes the *VC dimension* of CNNs of convolutional depth d. The VC dimension of a set of binary functions, is the maximum number m of vectors which can be separated into two classes in all 2^m ways using functions from the set [14].

Lemma 1 *Let $\mathcal{H}^d \triangleq \left\{h_c : I \to \{0,1\} \mid c \in \mathcal{C}^d\right\}$ be the set of boolean functions implementable by all CNNs in $\mathcal{C}^d$, and $q(d) \triangleq \sum_{l=1}^d m_1^l \cdot (n_2^l - f_1^l + 1) \cdot (n_3^l - f_2^l + 1) \cdot (n_1^l n_2^l n_3^l + m_1^l (g^l)^2 + (p^l)^2)$. Then, the VC dimension of the class of CNNs defined in section III-A above satisfies*

$$\text{VCdim}\left(\mathcal{H}^d\right) \leq \alpha \left(d \cdot q(d)\right)^2, \tag{3}$$

for some constant α.

Proof Sketch: A parametrized class of functions with parameters in $\mathbb{R}^t$ that is computable in no more than p operations has a VC dimension which is $\mathcal{O}\left(t^2 p^2\right)$ (see [15, Theorems 5, 8] for allowable operations). We have: $t = \sum_{l=1}^d (m_1^l + n_1^l m_1^l f_1^l f_2^l) + |\mathbf{W}_f|$, where $|\mathbf{W}_f|$ is the number of trainable weights in the fully connected layers. The computational complexity of the l-th convolutional layer of a CNN is at most $\mathcal{O}(m_1^l \cdot (n_2^l - f_1^l + 1) \cdot (n_3^l - f_2^l + 1) \cdot (n_1^l n_2^l n_3^l + m_1^l (g^l)^2 + (p^l)^2)$. This result, together with the fact that we have assumed d' to be fixed, proves the lemma. An exact expression for the VC dimension bound which does not introduce a constant α can be derived. It is omitted here for clarity of presentation. ■

We now state the following theorem on the CNN generalization performance guarantees:

Theorem 1 *For any $0 < \delta < 1$, $\epsilon > 0$, $0 < \gamma \leq 1$, if S is chosen at random according to the distribution $\mathcal{D}_S$, such that*

$$|S| \geq \frac{8}{\gamma^2 \epsilon} \max\left\{\ln \frac{8}{\delta},\ 2\alpha \left(d \cdot q(d)\right)^2 \ln \frac{16}{\gamma^2 \epsilon}\right\}, \tag{4}$$

then, with probability at least $1 - \delta$, for every $c \in \mathcal{C}^d$, one of the following will hold:

(i) $\hat{e}_S(c) > (1 - \gamma)\epsilon$,

(ii) $e_T(c) = e_S(c) \leq \epsilon$, $\hat{e}_S(c) \leq (1 - \gamma)\epsilon$.

Proof Sketch: The proof can be derived using Lemma 1 and [16, Theorem A3.1]. Note that this result is not restricted to the exact architecture given in section II and any activation function can be used as long as it can be computed using the operations listed in [15, Theorems 5, 8]. ■

Theorem 1 implies that if condition (4) is met, and if the trained CNN c is such that $\hat{e}_S(c)$ is as small as desired,

then we know that, with high probability, c will exhibit good generalization performance. Let $M = \max_{l=1,\ldots,d} \{m_1^l \cdot (n_2^l - f_1^l + 1) \cdot (n_3^l - f_2^l + 1) \cdot (n_1^l n_2^l n_3^l + m_1^l (g^l)^2 + (p^l)^2)\}$, then $q(d) \leq M \cdot d$. From (4), we see that, for proper generalization, the training sample size should be larger than $M' \cdot d^4$ where $M' = M^2 \alpha \cdot \frac{16}{\gamma^2 \epsilon} \cdot \ln \frac{16}{\gamma^2 \epsilon}$. Conversely, when designing a CNN, given a fixed training set size $|S|$, we know that the CNN is very likely to exhibit good generalization performance if the depth of the convolutional layers is less than $\sqrt[4]{\frac{|S|}{M'}}$. We also state a converse to Theorem 1 (the proof of which is based on [17, Theorem 1]):

Theorem 2 *For any learning algorithm which uses a training sample set S of size*

$$|S| \leq \frac{\mathrm{VCdim}\left(\mathcal{H}^d\right) - 1}{2e\epsilon} \tag{5}$$

(where e denotes the base of the natural logarithm), there exists a CNN $c \in \mathcal{C}^d$ and a distribution $\mathcal{D}$ such that the expected error of c (w.r.t. $\mathcal{D}$) is at least ϵ.

C. Different training and testing distributions

In section III-B above, we addressed the question of when a CNN is expected to generalize from $|S|$ training examples chosen according to an arbitrary probability distribution $\mathcal{D}_S$, assuming that test examples are drawn from the same distribution. In this section, we relax this assumption and allow the training and testing distributions to be different, $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. To this end, we define the variation divergence between the two distributions [18]:

$$\tau \triangleq 2 \sup_{B \in \mathcal{B}} \left| \Pr_{\mathcal{D}_S}[B] - \Pr_{\mathcal{D}_T}[B] \right|, \tag{6}$$

where $\mathcal{B}$ is the set of measurable subsets under $\mathcal{D}_S$ and $\mathcal{D}_T$. While we allow the two distributions to be different, our hope is that they are not *too* different so that learning from $\mathcal{D}_S$ is still somehow relevant for testing on $\mathcal{D}_T$. We now reformulate Theorem 1 for the case when $\tau \neq 0$:

Theorem 3 *Let $0 < \delta' < 1$, $\epsilon' > \tau$, $0 < \gamma' \leq 1$. If the training and testing sets are chosen independently at random according to the distributions $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively, such that*

$$|S| \geq \frac{8}{\bar{\gamma}^2 (\epsilon' - \tau)} \max \left\{ \ln \frac{16}{\delta'}, 2\alpha \left(d \cdot q(d)\right)^2 \ln \frac{16}{\bar{\gamma}^2 (\epsilon' - \tau)} \right\}, \tag{7}$$

where

$$\bar{\gamma} = \gamma' \cdot \left(1 + \frac{\tau}{\epsilon' - \tau}\right) - \frac{\tau}{\epsilon' - \tau}, \tag{8}$$

then, with probability at least $1 - \delta'$, for every $c \in \mathcal{C}^d$, one of the following will hold:

(i) $\hat{e}_S(c) > (1 - \gamma')\epsilon'$,

(ii) $e_T(c) \leq \epsilon'$, $\hat{e}_S(c) \leq (1 - \gamma')\epsilon'$.

Proof: We define the following event: $A = \{$ For every $c \in \mathcal{C}^d$, one of (i) or (ii) holds $\}$. We show that the probability that A does not occur is less than δ':

$$\begin{aligned}
\Pr\left[\bar{A}\right] &= \Pr\left[\exists c : e_T(c) > \epsilon', e_S(c) \leq \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\quad + \Pr\left[\exists c : e_T(c) > \epsilon', e_S(c) > \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\leq \Pr\left[\exists c : e_T(c) > \epsilon', e_S(c) \leq \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\quad + \Pr\left[\exists c : e_S(c) > \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\leq \Pr\left[\exists c : \epsilon' - \tau < e_S(c) \leq \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\quad + \Pr\left[\exists c : e_S(c) > \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \qquad (\star) \\
&\leq \Pr\left[\exists c : e_S(c) > \epsilon' - \tau, \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\quad + \Pr\left[\exists c : e_S(c) > \epsilon', \hat{e}_S(c) \leq (1-\gamma')\epsilon'\right] \\
&\leq \frac{\delta'}{2} + \frac{\delta'}{2} = \delta'. \qquad (\star\star)
\end{aligned}$$

where $(\star)$ follows from the fact that, from [18, Theorem 1], $e_T(c) \leq e_S(c) + \tau$, and $(\star\star)$ is an application of Theorem 1 with $\delta = \delta'/2$, $\epsilon = \epsilon' - \tau$, and $\gamma = \bar{\gamma}$. ■

Note that Theorem 3 requires that $\epsilon' > \tau$. As mentioned earlier, we are interested in the case when τ is small so that the learning is still useful. If $\tau \ll \epsilon'$, then $\bar{\gamma} \approx \gamma'$ and (4) and (7) are very close. When τ increases, so does the lower bound on $|S|$. This is to be expected, as we are looking at learning from and testing on two very different distributions.

IV. Experimental results

While section III gives some insight as to how to design CNNs which exhibit desirable generalization performance, it has been shown that neural networks tend to perform well with training sets which are smaller than required by the VC dimension bounds [9]. We therefore attempt to gain a better and more practical understanding of the problem by designing experiments for gender classification of face images. To this end, we use three different datasets: Images of Groups (GROUPS) [19], Labeled Faces in the Wild (LFW) [20], and Facetracer [21]. We resize face images to 64x64 and normalize them using histogram equalization. We then use mean-subtraced normalized face images to train CNNs of convolutional depths 3, 4, and 5. Once the CNN is trained, we classify new face images by resizing and normalizing them, then applying the learned model to them. We use the Caffe framework [22] to train and test the CNNs.

A. Method

For each depth $d = 3, 4, 5$, we select uniform random subsets of varying sizes from each training dataset. Since, as noted in Section III-B, a sufficient training sample size which guarantees good generalization is proportional to d^4, we choose the random training subsets to have sizes $|S| = \beta \cdot d^4$ for different values of β. Then, for each depth, dataset, and training subset size, we train a CNN (starting from a random weight initialization) until we reach a training error $\hat{e}_S(c) < 0.05$. We then test the resulting CNN on a testing set T in order to estimate $e_T(c)$. For the case when the testing and training distributions are the same, we perform 5-fold cross-validation using the protocol specified in [23] for LFW and GROUPS, and five random splits for Facetracer. We also perform cross-dataset testing, training on subsets of one dataset and testing on the other two.

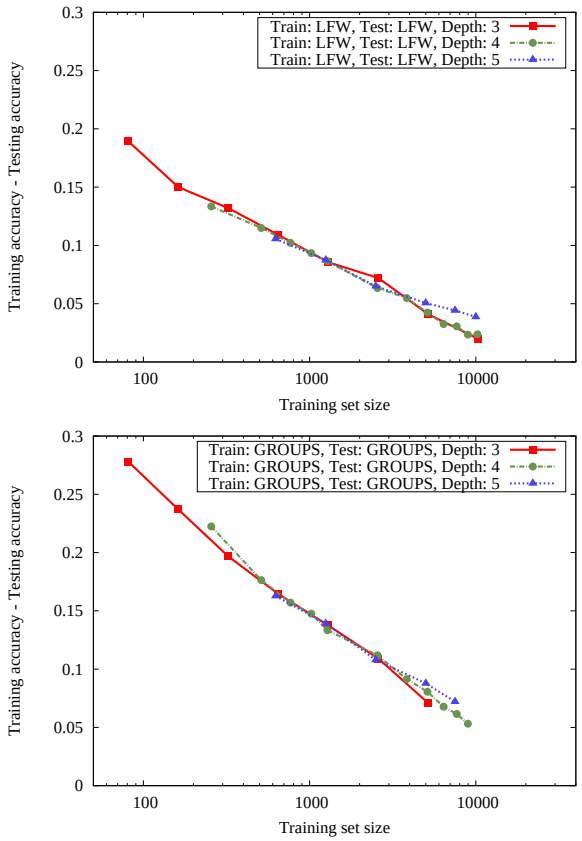

Fig. 2. Generalization performance of CNNs trained and tested on LFW (top) and GROUPS (bottom) for varying training set sizes. For a fixed generalization performance, deeper networks require more training data.

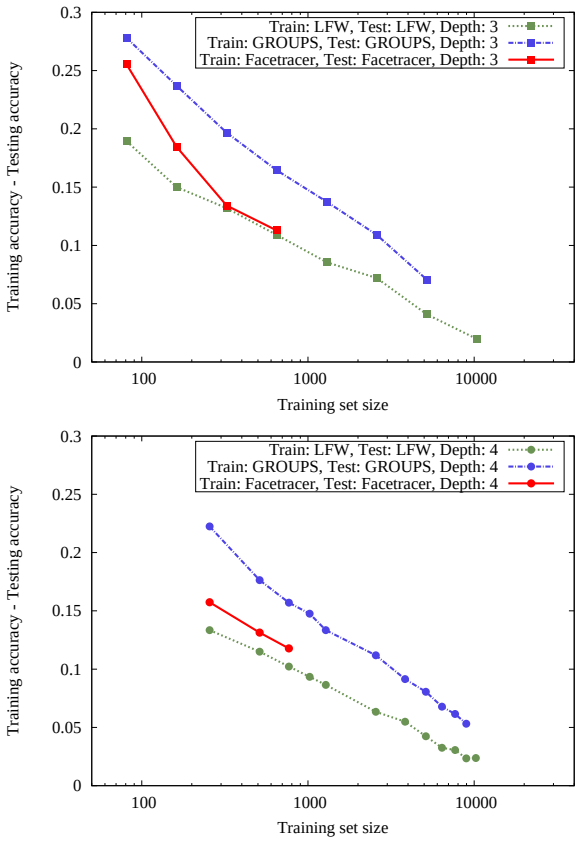

Fig. 3. Generalization performance of CNNs of depths 3 (top) and 4 (bottom) trained and tested on the same datasets. Other than the training set size, factors such as the training distribution affect generalization performance.

B. Architectures

As mentioned in section II, each convolutional layer of the CNN is composed of a filter bank sublayer, a rectification sublayer, a pooling and subsampling sublayer, and a local contrast normalization sublayer. All pooling sublayers are max-pooling and use 3×3 windows. All local contrast normalization sublayers use 5×5 windows, except for the first one, which uses 7×7 windows. The first layer's filter bank sublayer consists of a 15×15 convolution mask applied every 3 pixels, resulting in 96 feature maps. The second filter bank sublayer has 5×5 convolution filters with 256 output maps. The third (and, when needed, fourth and fifth) sublayer uses 3×3 kernels with 384 feature map outputs.The convolutional layers are followed by three fully connected layers. The first two have 4096 outputs and are each followed by rectification and a 50% dropout. The last fully connected layer has two outputs. We do not attempt to optimize the architecture of the CNNs and keep it fixed in the experiments, only varying the convolutional depth d.

C. Results

Since the designed CNNs have different training errors, comparing their testing accuracies would not be very informative. Instead, we consider the difference between the testing and training errors. When dataset D1 is used for training and dataset D2 for testing, we denote this difference by $\Delta_{\text{D1,D2}}$.

1) Same training and testing distribution: In the case of the same training and testing distribution, we take the average across the five cross-validation tests. In general, and as expected, we notice that $\Delta_{\text{D1,D1}}$ decreases with the training set size. For instance, Figure 2 plots, for depths $d = 3, 4, 5$, $\Delta_{\text{LFW,LFW}}$ and $\Delta_{\text{GROUPS,GROUPS}}$ vs. the training set size $|S|$ (in logscale). We note that, when plotted against $|S|$, $\Delta_{\text{LFW,LFW}}$ behaves similarly for depths 3 and 4, and the CNNs actually achieve good generalization performance for relatively small training set sizes. For example, to have $\Delta_{\text{LFW,LFW}} \leq 0.05$, $|S|$ should only be greater than about 1500. This is much smaller than the bound given in Theorem 1 which is actually very large (in fact, even for $d = 1$, $q(1)$ is larger than the total number of images in GROUPS and Facetracer). It also seems to be the same for both $d = 3$ and $d = 4$, which is contrary to what was expected. For $d = 5$, slight over-fitting seems to take place, and larger training set sizes are needed to achieve similar generalization performance as in shallower networks. As seen in the bottom plot, we observe a similar behavior with GROUPS but the over-fitting is apparent starting from $d = 4$. As previously mentioned, while shown to be tight in Theorem 2, bounds based on the VC dimension tend to be very large as they provide generalization performance guarantees regardless of the underlying probability distribution on the training and testing examples, and of the training algorithm used [15]. In fact, Figure 3 shows that while the CNN performance does generally improve with larger training sets, other aspects,

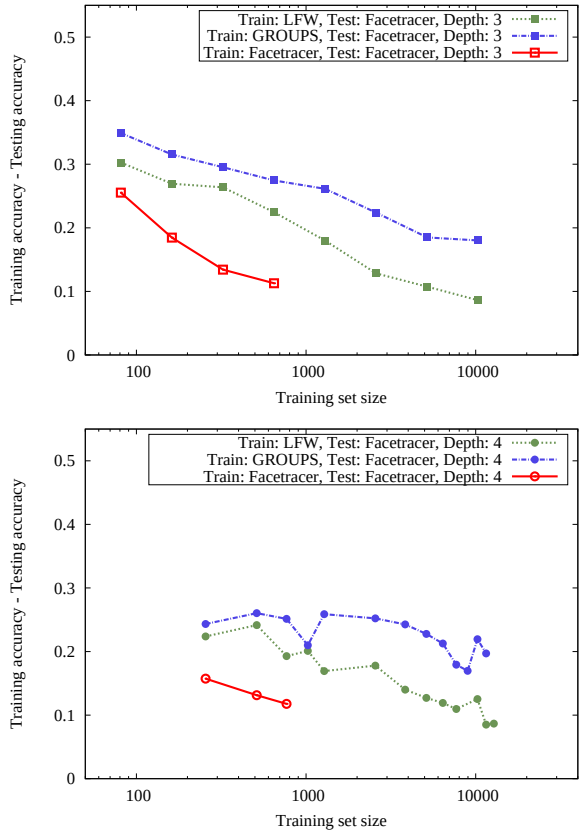

Fig. 4. Generalization performance of CNNs of depths 3 (top) and 4 (bottom) tested on Facetracer and trained on different datasets. For a fixed performance, more training samples are needed for cross-dataset testing.

especially the sample distribution, have a considerable effect. For example, the results seem to suggest that CNNs perform better on LFW gender classification than on GROUPS gender classification. The training algorithm is also important as it can restrict the set of realizable CNNs to a subset of $\mathcal{C}^d$. Our training uses dropout in the fully connected layers. Dropout is a very well known technique to reduce overfitting in deep neural networks [24]. However, CNNs with dropout and without dropout have the same VC dimension and therefore share the same bounds in Theorem 1. Since dropout has become almost standard in state-of-the-art CNN implementations, we chose to only carry out experiments using it. However, we naturally expect the over-fitting behavior to be much more prominent for deep CNNs which do not use dropout.

2) Different training and testing distributions: Theorem 3 suggests that more training samples are needed for cross-dataset testing in order to achieve the same generalization performance compared to when the training and testing samples have the same distribution. This is shown to be clearly the case in Figure 4. In the top figure, we see that, for depth 3, to achieve $\Delta_{\text{D1,D2}} < 0.2$, for D2 = Facetracer, we need $|S|$ to be greater than 105, 1000, and 1300, for D1 = Facetracer, LFW, and GROUPS, respectively. Figure 4 also shows that training using the LFW dataset seems to be more "relevant" for testing on Facetracer. This suggests that the variation divergence τ between the underlying distributions of Facetracer images and LFW images could be smaller than that between the distributions of Facetracer and GROUPS images. However, τ cannot be accurately estimated from finite samples of distributions [18]. We therefore seek a different approach to quantifying the distance between the distributions. We consider the method proposed in [25] to estimate the KL divergence between distributions based on k-th nearest neighbor distances. The KL divergence is a non-symmetric measure of the difference between two probability distributions. According to [25], given $\{X_1, \ldots X_m\}$ and $\{Y_1, \ldots, Y_p\}$ n-dimensional samples drawn according to two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively, the KL divergence estimate is given by:

$$\hat{D}(\mathcal{D}_1||\mathcal{D}_2) = \frac{n}{m}\sum_{i=1}^{m} \ln \frac{\nu_k(i)}{\rho_k(i)} + \ln \frac{p}{m-1}, \qquad (9)$$

where $\nu_k(i)$ is the distance between X_i and its k-th nearest neighbor in $\{Y_j\}$, and $\rho_k(i)$ is the distance between X_i and its k-th nearest neighbor in $\{X_j\}_{j\neq i}$. The choice of k trades off bias and variance. While it is true that the number of images available is relatively small compared to their dimension (64x64x3) and therefore, the KL divergence estimates are not very accurate, we notice that both $\hat{D}(\mathcal{D}_{\text{Facetracer}}||\mathcal{D}_{\text{LFW}})$ and $\hat{D}(\mathcal{D}_{\text{LFW}}||\mathcal{D}_{\text{Facetracer}})$ are consistently smaller (by a factor of around 3) than $\hat{D}(\mathcal{D}_{\text{Facetracer}}||\mathcal{D}_{\text{GROUPS}})$ and $\hat{D}(\mathcal{D}_{\text{GROUPS}}||\mathcal{D}_{\text{Facetracer}})$ for different values of k ranging from 1 to 10. This difference could explain why, when tested on Facetracer, CNNs trained using LFW perform better than those trained using GROUPS.

In the bottom plot of Figure 4, we notice an over-fitting trend for the cross-dataset case at depth 4. This is in contrast with the findings when the training and testing samples have the same distribution. We investigate this on a different dataset (LFW) and across depths 3, 4 and 5. The results are shown in Figure 5. In the top figure, we see that the generalization performance tends to become worse as the depth increases, especially for models trained on GROUPS. In the bottom figure, the x-axis is changed to β (where $|S| = \beta \cdot d^4$) and we notice that for large β (> 10), models trained on GROUPS behave similarly across depths. This means that, if to achieve a certain generalization performance, a training set size $\beta \cdot 3^4$ is needed for CNNs of depth 3, then approximately $\beta \cdot 4^4$ and $\beta \cdot 5^4$ training samples are needed to achieve the same level of performance for CNNs of depths 4 and 5, respectively. It seems that, in this case, the number of samples needed for good generalization scales with d^4 as predicted by the theoretical bound (albeit with a smaller multiplicative constant). We found similar trends when testing on the GROUPS and Facetracer datasets but the plots are omitted due to space limitations.

V. Conclusion

In this paper, we extended various statistical learning theorems to characterize the relationship between the depth of a CNN, the size of the training set, and the generalization performance. We proved that whenever the training and testing distributions are the same, if the training set size is some constant times d^4, then the CNN will, with high probability, exhibit good generalization. We also showed that this bound increases when the training and testing distributions are different, and characterized it as a function of the variation divergence between the distributions. We then implemented deep CNNs for the problem of gender recognition on three well-known

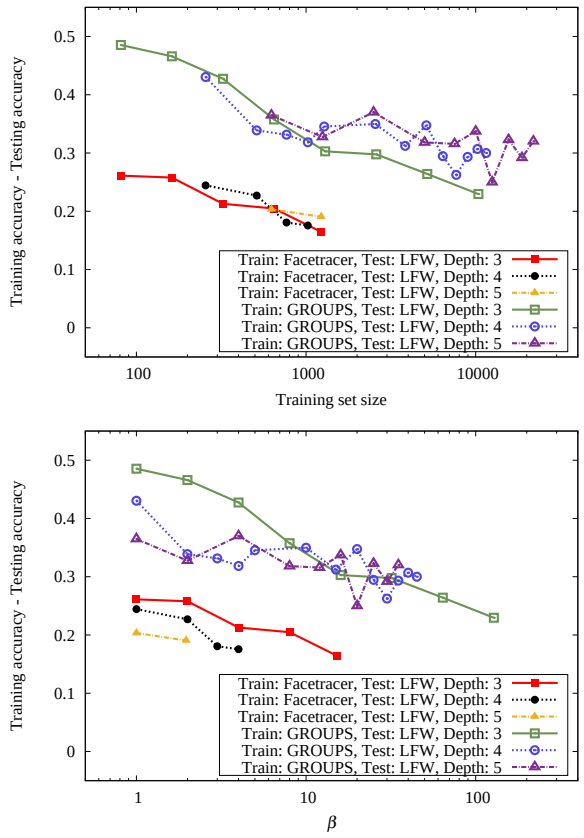

Fig. 5. Generalization performance of CNNs tested on LFW and trained on different datasets. The generalization performance tends to become worse as the depth increases. The d^4 relationship described in Theorem 1 is apparent in the bottom figure.

datasets. We empirically demonstrated that over-fitting tends to occur for very deep networks, which require larger training sets to achieve generalization performance similar to shallower versions. This is especially the case when the training and testing distributions are different. In our future work, we plan to further develop our theory and experiments to study the effects of other CNN parameters, extend to multi-class classification problems, as well as investigate the impact of other factors such as the underlying distribution and the training algorithm.

ACKNOWLEDGEMENTS

This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

REFERENCES

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE CVPR*, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[3] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *ICLR Workshops*, 2014.

[4] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[5] D. Eigen, J. Rolfe, R. Fergus, and Y. LeCun, "Understanding deep architectures using a recursive convolutional network," in *ICLR Workshops*, 2014.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[7] E. B. Baum and D. Haussler, "What size net gives valid generalization?" *Neural computation*, vol. 1, no. 1, pp. 151–160, 1989.

[8] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *NIPS*, 2014.

[9] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.

[10] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE Transactions on Neural Networks*, vol. 25, no. 8, pp. 1553–1565, 2014.

[11] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, no. 3, pp. 326–334, 1965.

[12] V. N. Vapnik and A. Y. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability & Its Applications*, vol. 16, no. 2, pp. 264–280, 1971.

[13] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *IEEE ICCV*, 2009.

[14] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2000.

[15] P. L. Bartlett and W. Maass, "Vapnik-Chervonenkis dimension of neural nets," *The handbook of brain theory and neural networks*, pp. 1188–1192, 2003.

[16] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, "Learnability and the Vapnik-Chervonenkis dimension," *Journal of the ACM*, vol. 36, no. 4, pp. 929–965, 1989.

[17] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant, "A general lower bound on the number of examples needed for learning," *Information and Computation*, vol. 82, no. 3, pp. 247–261, 1989.

[18] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[19] A. C. Gallagher and T. Chen, "Understanding images of groups of people," in *IEEE CVPR*, 2009.

[20] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," *Technical Report 07-49, University of Massachusetts, Amherst*, vol. 1, no. 2, 2007.

[21] N. Kumar, P. Belhumeur, and S. Nayar, "Facetracer: A search engine for large collections of images with faces," in *ECCV*, 2008.

[22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[23] P. Dago-Casas, D. González-Jiménez, L. L. Yu, and J. L. Alba-Castro, "Single- and cross-database benchmarks for gender classification under unconstrained settings," in *IEEE ICCV Workshops*, 2011.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] Q. Wang, S. R. Kulkarni, and S. Verdú, "Divergence estimation for multidimensional densities via-nearest-neighbor distances," *IEEE Transactions on Information Theory*, vol. 55, no. 5, pp. 2392–2405, 2009.