Complexity of Multiverse Networks and their Multilayer Generalization

Etai Littwin and Lior Wolf
The Blavatnik School of Computer Science
Tel Aviv University

***Abstract*—Multiverse networks were recently proposed as a method for promoting more effective transfer learning. While an extensive analysis was proposed, this analysis failed to capture two main aspects of these networks: (i) the rank of the representation is much lower than the rank predicted by the analysis; and (ii) the contribution of increased multiplicity in such networks diminishes quickly. In this work, we propose additional analysis of multiverse networks which addresses both deficits. A major contribution of our work is quantifying the Rademacher complexity of the multiverse network. It is shown that the complexity upper bound of multiverse networks is significantly lower than that of conventional networks, and diminishes by a factor of $\sqrt{k}$, k being the multiplicity. In addition, we generalize the notion of multiverse networks to multilayer multiverse networks. We derive the Rademacher complexity formula to such networks and present experimental results.**

I. Introduction

A major aspect in the recent success of deep neural networks is their effectiveness in tasks involving transfer learning. However, only a few contributions have focused on directly promoting this property. The Multiverse networks [7] have not only shown experimentally to significantly increase the effectiveness of transfer learning in comparison to conventional networks, but are also, in the context of the Joint Bayesian algorithm [2], theoretically justified.

Multiverse networks are a variant of neural networks in which the representation layer is linked to the predictions through multiple sets of pairwise orthogonal classifiers. Without explicitly enforcing anything beyond this orthogonality, two desirable properties emerge. First, the dimensionality of the learned representation is reduced and a low-rank representation is obtained. Second, the representation contains more discriminative directions, as is measured by the fisher score, i.e., by the generalized Rayleigh quotient of the Between- and Within-class covariance matrices. Both these properties are shown by a series of theorems in [7].

Even more remarkably, the multiverse network is essentially parameter free: as the multiplicity k is increased, the rank of the representation converges quickly to a "natural dimensionality". This dimensionality is stable and is much lower than what is predicted by the theoretical analysis of [7]. Therefore, unlike most dimensionality reduction and related methods, when employing multiverse networks, the data provides us with a clear signal as to the desired representation independently of our choice of parameters.

In the current work, we address this unique property. First, we measure the richness of the class of multiverse networks as a function of the multiplicity parameter k. Second, we explain the root cause for an emerging dimensionality that is much lower than what is predicted by [7].

In addition, we generalize the notion of multiverse networks. In [7], multiverse networks were studies at the level of what is called the "representation layer", i.e., the activations of the layer just below the top classification layer. However, as we show here, the multiverse principle can also readily be applied to other layers. In the following sections, multilayer multiverse networks are defined, a suitable optimization algorithm is proposed, their functional complexity is derived and initial experimental results for employing such networks are presented.

II. Recapitulating multiverse networks

Multiverse networks as introduced in [7] were shown to improve transfer learning in various verification metrics, as well as to encourage some desirable properties in the emerging representation such as low rank and better discriminability between classes. In its core, the multiverse network is simply a neural network with multiple orthogonal classifiers for each class, see Fig. 1(b). During training, the cross entropy loss of each vector of softmax probabilities produced by the different classifiers are summed over, with an added term that encourages orthogonality between the different classifiers of the same class. Specifically, we consider a simple model of an L layer feed forward neural net with multiple outputs and relu activation functions $\mathcal{N} : \mathbb{R}^d \rightarrow \mathbb{R}^c$. We denote[1] by $W_j \in \mathbb{R}^{d_{j-1} \times d_j} = [w_{j1}, w_{j2}...w_{jd_j}]$ the weights matrix connecting layer $j-1$ to layer j where d_j is the dimensionality of layer j (notice that $d_L = c$), and by $\sigma()$ the relu activation function acting element wise on its input. The output of a neural net with l layers acting on input x is therefore:

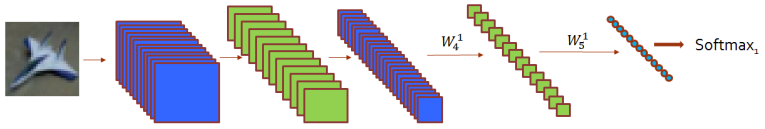
$$\mathcal{N}(x) = W_L^\top ...\sigma(W_3^\top \sigma(W_2^\top \sigma(W_1^\top x))) \tag{1}$$

We denote by $\mathcal{N}_l(x) \in \mathbb{R}^{d_l}$ the output of the l'th layer, and $\mathcal{N}_l^n(x) \in \mathbb{R}$ the output of the n'th neuron in the l'th layer, so that:

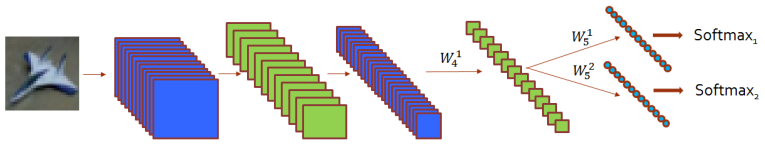
$$\mathcal{N}_l^n(x) = w_{ln}^\top \mathcal{N}_{l-1}(x) \tag{2}$$

where w_{ln} is the n'th column of W_l. A k-multiverse network has k column-wise orthogonal weight matrices $W_L^1...W_L^k$ connecting layer $L-1$ and layer L. We index the orthogonal weight vectors of the multiverse network by a superscript such

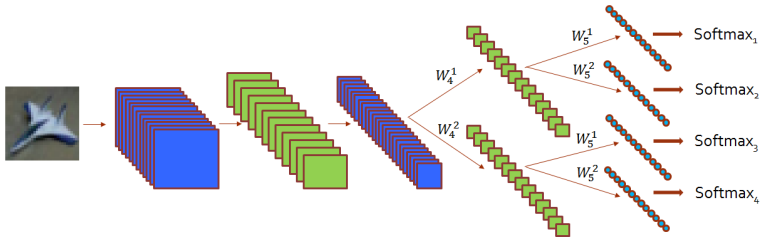
[1]The notations used in this work are summarized in Table I.



(a)



(b)



(c)

Fig. 1. An illustration of the multiverse concept. (a) In a conventional neural network, there is one weight matrix that connects each layer to the next one. This leads to a single vector of softmax pseudo-probabilities on which the training loss is computed. (b) In a conventional multiverse network [7], there are multiple, alternative, weight matrices connecting the layer before the last to the last layer. In the illustration, the multiplicity parameter k is 2. (c) In multilayer multiverse networks, there are multiple layers for which alternative weight matrices exist. For a number of split-layers p, and a given multiplicity k, the number of possible paths is k^p. In the illustration $p = k = 2$.

that: $\forall j, i \neq i' \quad w_{Lj}^{i\top} w_{Lj}^{i'} = 0$. When training, given input pairs $x = (x_1, y_1)...(x_m, y_m)$, the loss is computed as follows:

$$L(x) = -\frac{1}{k}\sum_{t=1}^{k}\sum_{i=1}^{m} log(\frac{e^{w_{Ly_i}^{t\top}\mathcal{N}_{L-1}(x_i)}}{\sum_{j=1}^{Y} e^{w_{Lj}^{t\top}\mathcal{N}_{L-1}(x_i)}}) + \lambda\sum_{t=1}^{k}\sum_{t'=t+1}^{k}\sum_{j=1}^{Y}|w_{Lj}^{t\top} w_{Lj}^{t'}| \quad (3)$$

At test time, the probability of $x_i \in y_i$ is computed as follows:

$$p(y_i|x_i) = \frac{1}{k}\sum_{t=1}^{k}\frac{e^{w_{Ly_i}^{t\top}\mathcal{N}_{L-1}(x_i)}}{\sum_{j=1}^{Y} e^{w_{Lj}^{t\top}\mathcal{N}_{L-1}(x_i)}} \quad (4)$$

It was shown in [7] that instead of the averaging of probabilities in Eq. 4 it is enough to select one probability vector (fix t), since all k probability vectors are identical (this is due to the fact the the projection of each classifier in the set of orthogonal classifiers on the data turns out to be the same, and orthogonality is achieved by the low rank representation). In addition, it was shown that multiverse networks encourage desirable properties in the emerging representation, such as a more balanced, low rank distribution and fisher spectrum. It was further shown that these properties allow for more effective transfer learning using verification metrics, as demonstrated on tasks such as face recognition.

Symbol	
x	A set of input data.
Y	Number of classes.
m	Number of data points, typically indexed by i.
y	$m \times 1$ vector of labels. Each label is y_i.
$\mathcal{N}(x_i)_l^n$	The output of the n'th neuron in the l'th layer of a neural net $\mathcal{N}(x_i)$ evaluated on input x_i.
$\mathcal{N}_{mult}$	A network with one or more multiverse layers.
L	Number of layers in a network.
d_l	Dimensionality of the representation in layer l.
W_l	A $d_{l-1} \times d_l$ matrix of weights connecting layer $l-1$ and layer l.
w_{ln}	Column n in W_l.
c_l	A bound on the norm of w_{ln}.
k	Multiplicity parameter in a multiverse network.
p	Number of multiverse layers in a multiverse network.
$\mathcal{R}(\mathcal{N}_\mathcal{C})$	The rademacher complexity of network $\mathcal{N}$.
R	A $d_{L-1} \times m$ matrix of features (i.e representation in layer $L-1$ of data x).
K	$RR^\top$.
r_i	Column i of R.
U	Kernal (Null space) of R.

TABLE I
SUMMARY OF NOTATIONS.

III. Multilayer multiverse networks

In this work, we extend the notion of multiverse networks to hidden layers, where the exact same column wise orthogonality principle applies to the weights of single or multiple hidden layers, as illustrated in Fig. 1(c). In the multilayer case, in order to compute the loss for some batch, we would need to forward propagate an input sample through exponentially many possible paths, and thus sum over exponentially many softmax probability vectors. For example, given that both layers L and $L-1$ have k orthogonal copies, the output of the network for sample x_i depends on the path $u = [u_1, u_2], u_i \in [1, 2...k]$:

$$\mathcal{N}_L(x_i, u) = W_L^{u_1\top}\sigma(W_{L-1}^{u_2\top}...\sigma(W_2^\top\sigma(W_1^\top x))) \quad (5)$$

Since it is not feasible to forward propagate through exponentially many paths during training, we adopt a more stochastic approach where the path u is uniformly sampled for each batch. The loss for some path u and input x_i is therefore:

$$L(x,u) = -\sum_{i=1}^{m} log(\frac{e^{\mathcal{N}(x_i,u)_L^{y_i}}}{\sum_{j=1}^{Y} e^{\mathcal{N}(x_i,u)_L^{y_j}}}) + \lambda\sum_l\sum_{t\neq u_l}\sum_j |w_{lj}^{u_l\top}w_{lj}^t| \quad (6)$$

where the index l implies summing over "multiverse" layers. In order to classify a new sample at test time, exponentially many forward propagations would still need to be computed and averaged out, which may be impractical in many real world applications. However, such a computation is unnecessary, since similar to the conventional multiverse network, the pseudo probability vectors for each path turn out to be the same, and hence a single forward propagation of any random path is sufficient.

IV. Rademacher complexity of multivrse networks

Let $X = [x_1...x_m], x \in \mathbb{R}^d$ denote some input set drawn from some unknown distribution, and let $F : x \to \mathbb{R}$ denote a class of functions. We denote the random variable $\hat{\mathcal{R}}_m$ as:

$$\hat{\mathcal{R}}_m(F, x) = E_\rho\Big[\sup_{f\in F}(\frac{1}{m}\sum_{i=1}^{m}\rho_i f(x_i))|x\Big] \quad (7)$$

Where ρ_i are uniform independent $-1, 1$ valued random variables. The Rademacher complexity of F is defined as [1]:

$$\mathcal{R}(F) = E_x\hat{\mathcal{R}}_m(F) \quad (8)$$

For simplicity, we will consider a model of a feed forward neural net with a single output (d_L = 1, $w_L \in \mathcal{R}^{d_{L-1}\times 1}$), which we denote by the superscript 1, e.g., $\mathcal{N}_L^1$, is an L layer neural network with a univariate output. In addition, we apply L_2 regularization on the weights by restricting the L_2 norm of each weight vector $\forall li \quad \|w_{li}\|_2 < c_l$. We define the Rademacher complexity of a k-multiverse network $\mathcal{N}_{mult,L}^1$ as:

$$\mathcal{R}(\mathcal{N}_{mult,L}^1) = E_x E_\rho\Big[\sup_W(\frac{1}{km}\sum_{i=1}^{m}\rho_i\sum_{t=1}^{k}w_L^{t\top}\mathcal{N}_{L-1}(x_i))|x\Big] \quad (9)$$

The following theorem bounds the Rademacher complexity of such class of functions.

Theorem 1: The Rademacher complexity of feed forward neural networks with a single output, relu activation functions and k orthogonal classifiers is bounded by:

$$\mathcal{R}(\mathcal{N}_{mult,L}^1) \leq \frac{\sqrt{d_{L-1}}c_L}{\sqrt{k}}\mathcal{R}(\mathcal{N}_{L-1}^1) \quad (10)$$

Proof:

$$\mathcal{R}(\mathcal{N}_{mult,L}^1) = E_x E_\rho\Big[\sup_W(\frac{1}{km}\sum_{t=1}^{k}w_L^{t\top}\sum_{i=1}^{m}\rho_i\mathcal{N}_{L-1}(x_i))|x\Big] \quad (11)$$

For any vectors u, v, it holds that $\|u\|_1\|v\|_\infty > u^\top v$, and so:

$$\mathcal{R}(\mathcal{N}_{mult,L}^1) \leq E_x E_\rho\Big[\sup_W(\frac{\|\sum_{t=1}^{k}w_L^t\|_1}{km}\sup_n\Big|\sum_{i=1}^{m}\rho_i\mathcal{N}_{L-1}^n(x_i))\Big||x\Big] \quad (12)$$

It holds that $\|\sum_{t=1}^{k}w_L^t\|_1 \leq \sqrt{d_{L-1}}\|\sum_{t=1}^{k}w_L^t\|_2$. Using the orthogonality constraint between $w_L^1...w_L^k$, we have that $\|\sum_{t=1}^{k}w_L^t\|_2 \leq c_L\sqrt{k}$. And so:

$$\begin{aligned}\mathcal{R}(\mathcal{N}_{mult,L}^1) &\leq \\ \frac{\sqrt{d_{L-1}}}{k}E_x E_\rho\Big[\sup_W(\frac{\|\sum_{t=1}^{k}w_L^t\|_2}{m}\sup_n\Big|\sum_{i=1}^{m}\rho_i\mathcal{N}_{L-1}^n(x_i))\Big||x\Big] &\leq \\ \frac{\sqrt{d_{L-1}}c_L}{\sqrt{k}}E_x E_\rho\Big[\sup_W(\frac{1}{m}\sup_n\Big|\sum_{i=1}^{m}\rho_i\mathcal{N}_{L-1}^n(x_i))\Big||x\Big] &= \frac{\sqrt{d_{L-1}}c_L}{\sqrt{k}}\mathcal{R}(\mathcal{N}_{L-1}^1) \quad (13)\end{aligned}$$

■

In order to compare the upper bound with that of a regular neural network with a single classifier, we notice that substituting $w_L^1 = w_L^2 = ...w_L^k = w_L$ in Eq. 9 results in the standard Rademacher complexity definition of neural networks with a single classifier. Following the same derivation, we get the upper bound for the universe case:

$$\mathcal{R}(\mathcal{N}_L^1) \leq \sqrt{d_{L-1}}c_L\mathcal{R}(\mathcal{N}_{L-1}^1) \quad (14)$$

Theorem 1 gives a clear indication as to the regularization power of the multiverse architecture, as the model overall complexity is reduced by a factor of $\sqrt{k}$. Theorem 1 can also be extended to the middle layer multiverse using the Rademacher bounds for the composition of functions. Specifically, we will use the following lemma [1]:

lemma 1: Let F be a a class of real valued functions, and let $\phi : \mathcal{R} \to \mathcal{R}$ be a Lipschitz function with a Lipschitz constant L_ϕ and $\phi(0) = 0$. Then:

$$\mathcal{R}(F \circ \phi) \leq 2L_\phi \mathcal{R}(F) \tag{15}$$

Lemma 1 directly applies to networks with relu activations, since the relu function is a Lipschitz function with a constant $L_\phi = 1$. We now extend Theorem 1 to a general layer in the network.

Theorem 2: The Rademacher complexity of feed forward neural networks with a single output, relu activation functions and k column wise orthogonal matrices in some layer l is bounded by:

$$\mathcal{R}(\mathcal{N}^1_{mult,L}) \leq \frac{2^{L-l+1}\prod_{j=l}^{L}\sqrt{d_{j-1}}c_j}{\sqrt{k}}\mathcal{R}(\mathcal{N}^1_{l-1}) \tag{16}$$

Proof: This result is immediate using theorem 1, lemma 1 and equation 14. ∎

We now extend the the complexity bound to multilayer multiverse nets. In the general case, where p layers have k orthogonal weight matrices, an input has k^p possible net configurations to traverse. In the following analysis, we consider the family of simple feed forward networks, where the top p hidden and output layers have k orthogonal copies. Let $u = [u_1, u_2...u_p], u_i \in [1...k]$ index the weight copies in the top p layers, the output of $\mathcal{N} \in H$ for input x and weight configuration u is therefore:

$$\mathcal{N}^1_L(x,u) = W_L^{u_1\top}...\sigma(W_{L-p+1}^{u_{p-1}\top}\sigma(W_{L-p+1}^{u_p\top}\mathcal{N}_{L-p}(x))) \tag{17}$$

We define the Rademacher complexity $R(\mathcal{N}^1_{mult,L})$:

$$\mathcal{R}(\mathcal{N}^1_{mult,L}) = E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^p m}\sum_{i=1}^{m}\rho_i\sum_{u_1...u_p=1}^{k}\mathcal{N}^1_L(x_i,u)\Big)|x\Big] \tag{18}$$

In the following theorem, we bound the Rademacher complexity of $\mathcal{N}^1_{mult,L}$.

Theorem 3: The Rademacher complexity of $\mathcal{N}^1_{mult,L}$ is bounded by:

$$\mathcal{R}(\mathcal{N}^1_{mult,L}) \leq \frac{2^{p-1}\prod_{j=1}^{p}\sqrt{d_{L-j}}c_{L-j+1}}{k^{\frac{p}{2}}}\mathcal{R}(\mathcal{N}^1_{L-p}) \tag{19}$$

Proof: We have:

$$\begin{aligned}
&\mathcal{R}(\mathcal{N}^1_{mult,L}) = \\
&E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^p m}\sum_{i=1}^{m}\rho_i\sum_{u_1...u_p=1}^{k}\mathcal{N}^1_L(x_i,u)\Big)|x\Big] = \\
&E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^p m}\sum_{u_1=1}^{k}w_L^{u_1\top}\sum_{i=1}^{m}\rho_i\sum_{u_2...u_p=1}^{k}\mathcal{N}_{L-1}(x_i,u)\Big)|x\Big] \\
&\leq \|\sum_{u_1=1}^{k}w_L^{u_1}\|_2 E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^p m}\sup_n... \\
&\Big|\sum_{i=1}^{m}\rho_i\sum_{u_2...u_p=1}^{k}\mathcal{N}^n_{L-1}(x_i,u)\Big)\Big||x\Big] \\
&\leq \frac{\sqrt{d_{L-1}}c_L}{\sqrt{k}}E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^{p-1}m}\sup_n... \\
&\Big|\sum_{i=1}^{m}\rho_i\sum_{u_2...u_p=1}^{k}\mathcal{N}^n_{L-1}(x_i,u)\Big)\Big||x\Big]
\end{aligned} \tag{20}$$

Using lemma 1:

$$\begin{aligned}
&\mathcal{R}(\mathcal{N}^1_{mult,L}) \leq \\
&\frac{2\sqrt{d_{L-1}}c_L}{\sqrt{k}}E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^{p-1}m}\sup_n\Big|\sum_{u_2=1}^{k}w_{L-1,n}^{u_2\top}\sum_{i=1}^{m}\rho_i... \\
&\sum_{u_3...u_p=1}^{k}\mathcal{N}^n_{L-2}(x_i,u)\Big)\Big||x\Big] \leq \\
&\frac{2\sqrt{d_{L-2}d_{L-1}}c_L c_{L-1}}{k}E_x E_\rho\Big[\sup_W\Big(\frac{1}{k^{p-2}m}\sum_{i=1}^{m}\rho_i... \\
&\sum_{u_3...u_p=1}^{k}\mathcal{N}^1_{L-2}(x_i,u)\Big)|x\Big]
\end{aligned} \tag{21}$$

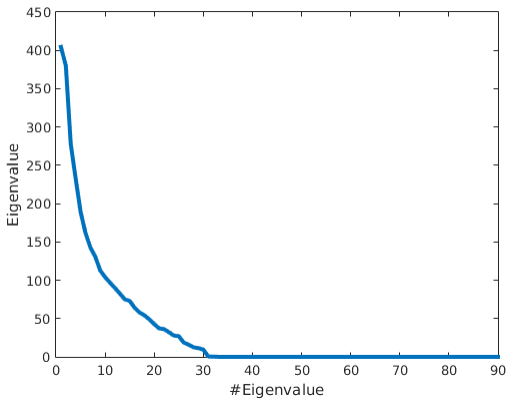
Applying this procedure recursively, we get the result:

$$\mathcal{R}(\mathcal{N}^1_{mult,L}) \leq \frac{2^{p-1}\prod_{j=1}^{p}\sqrt{d_{L-j}}c_{L-j+1}}{k^{\frac{p}{2}}}\mathcal{R}(\mathcal{N}^1_{L-p}) \tag{22}$$

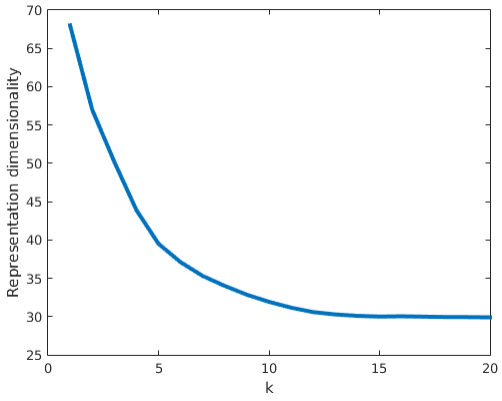
∎

V. The effect of regularization on multiverse networks

An especially evident property that emerges in the representation of a multiverse network is the low rank property. The spectrum of the representation covariance matrix seems to drop abruptly to zero in a manner not consistent with the familiar exponential tail that characterizes the representation of regular networks, as illustrated in Fig. 2. It was suggested in [7] that enforcing multiple orthogonal classifiers essentially enforces multiple solutions to a linear set of equations, effectively enforcing a non-zero kernal on the representation. This, however, does not explain the magnitude of the drop in the effective rank, since even a slight drop in rank can theoretically allow multiple orthogonal solutions. Formally, we denote the network representation in layer $L-1$ of input $x = [x_1...x_m]$ by



(a)



(b)

Fig. 2. (a) Spectrum of the representation covariance when using multiplicity k=15. An effective 30-dimensional representation is acquired, without the familiar exponential tail in the spectrum following this dimension. (b) Effective rank as a function of k. The dimensionality converges to a constant dimensionality as a function of k is shown.

$R \in \mathbb{R}^{d_{L-1} \times m} = [r_1 ... r_m] = [\mathcal{N}_{L-1}(x_1) ... \mathcal{N}_{L-1}(x_m)]$, and the representation kernal as $K = RR^\top$. Given some classifier w^1 and a representation kernal K such that K has a kernal of rank s $U \in \mathbb{R}^{d_{L-1} \times s} : U^\top K = 0$, it is possible to construct a second orthogonal classifier $w^2 = w^1 + U\gamma$ which gives rise to the same vector of scores for each input:

$$w^{1\top} R = (w^1 + U\gamma)^\top R \tag{23}$$

where $\gamma \in \mathbb{R}^s$ is a vector of coefficients. Due to the orthogonality constraint, it holds that $(w^1 + U\gamma)^\top w^1 = 0 \rightarrow \gamma^\top U^\top w_1 = -\|w^1\|_2^2$, and so:

$$\|\gamma\|_2^2 \geq \frac{\|w^1\|_2^2}{\|U^\top w^1\|_2^2} \tag{24}$$

Therefore, if U is a kernal of rank 1 (i.e., K has rank $d_{L-1} - 1$), then an orthogonal solution is possible by construction, providing that $\|U^\top w^1\|_2 \neq 0$. However, if U is a kernal of rank 1, then for a multiclass classification problem, it is highly likely that for some class j, the term $\|U^\top w_j^1\|_2$ is close to zero, resulting in a classifier w of high norm. In general, the larger the kernal U, the smaller $\|\gamma\|_2$ which, in turn, reduces the norm of the weights. The regularization of the weights norm therefore directly contributes to the dramatic decrease in the rank of K.

Layer	Filter/Stride	#Channel	#Filter
Conv11	5×5 / 1	3	192
Conv12	1×1 / 1	192	160
Conv13	1×1 / 1	160	96
Pool1	3×3 / 2	96	–
Dropout1-0.5	–	–	–
Conv21	5×5 / 1	96	192
Conv22	1×1 / 1	192	192
Conv23	1×1 / 1	192	100
Pool2	3×3 / 2	192	–
Dropout1-0.5	–	–	–
Conv31	3×3 / 1	192	192
Conv32	1×1 / 1	192	192
Conv33	1×1 / 1	192	100
Avg Pool	7×7 / 1	100	–
FC	1×100 / 1	100	100

TABLE II
THE MODIFIED NIN [6] MODEL USED IN THE CIFAR-100 EXPERIMENTS. THE NETWORK STARTS WITH A COLOR INPUT IMAGE OF SIZE $3 \times 32 \times 32$ PIXELS, AND RUNS THROUGH 3 CONVOLUTIONAL BLOCKS INTERLEAVED WITH RELU AND MAX POOLING LAYERS. FOLLOWING A SPATIAL AVERAGE POOLING AT THE END OF THE PROCESS, A REPRESENTATION OF SIZE 100 IS OBTAINED. A FC LAYER OF SIZE 100 WAS ADDED TO THE ARCHITECTURE FOR REASONS OF IMPLEMENTATION CONVENIENCE.

VI. EXPERIMENTS

We present experiments demonstrating the use of multilayer multiverse networks. The experiments are performed on the CIFAR-100 dataset [5] and on the CASIA face recognition dataset [9], where transfer is shown on LFW [3].

In our experiments, we employ two network architectures. For the CIFAR-100 experiments, we use the architecture of network in network [6]; for the face recognition experiments, we use the scratch architecture [9]. The networks were trained from scratch at each experiment, using the MatConvNet framework [8].Both networks are fully convolutional, and we added a hidden linear layer on top of the networks (no relu) in order to apply our method on top of a vector of activations. Furthermore, for the standard multiverse networks (only top layer), the learning rate for each layer excluding the top layer was divided by the multiplicity parameter k, effectively averaging the gradients from the k losses. The architectures used are given, for completeness, in Tab. II and Tab. III for the network in network and scratch networks respectively.

The CIFAR-100 [5] contains 50,000 32×32 color images, split between 100 categories. Throughout our experiments, the first 90 classes (class ids 0 to 89) are used as the source domain, and the last 10 as the target domain.

Our experiments compare four architectures: a baseline with one cross entropy loss ($k = 1$); two monolayer multiverse architectures with 3 and 5 such losses, and a multilayer multiverse network where both the last convolution layer and fully connected layer are multiverse layers with $k = 2$. For the multilayer model, the weight matrices in the multiverse layers were averaged out at test time, which slightly improved its overall performance.

Layer	Filter/Stride	#Channel	#Filter
Conv11	3×3 / 1	1	32
Conv12	3×3 / 1	32	64
Max Pool	2×2 / 2	64	–
Conv21	3×3 / 1	64	64
Conv22	3×3 / 1	64	128
Max Pool	2×2 / 2	128	–
Conv31	3×3 / 1	128	96
Conv32	3×3 / 1	96	192
Max Pool	2×2 / 2	192	–
Conv41	3×3 / 1	192	128
Conv42	3×3 / 1	128	256
Max Pool	2×2 / 2	256	–
Conv51	3×3 / 1	256	160
Conv52	3×3 / 1	160	320
Avg Pool	6×6 / 1	320	–
Dropout1-0.3	–	–	–
FC	1×320 / 1	320	320

TABLE III
THE SCRATCH MODEL BY THE AUTHORS OF [9], WHICH IS THE FACE RECOGNITION NETWORK IN OUR EXPERIMENTS. THE NETWORK STARTS WITH A GRAY SCALE INPUT IMAGE OF SIZE $1 \times 100 \times 100$ PIXELS, AND RUNS THROUGH 10 CONVOLUTIONAL LAYERS INTERLEAVED RELU AND MAX POOLING LAYERS. FOLLOWING A SPATIAL AVERAGE POOLING AT THE END OF THE PROCESS, A REPRESENTATION OF SIZE 320 IS OBTAINED. A FC LAYER OF SIZE 320 WAS ADDED TO THE SCRATCH ARCHITECTURE FOR REASONS OF IMPLEMENTATION CONVENIENCE.

Domain	Source	Target (transfer)
Metric	Val error	Accuracy
Conventional (k=1,p=1)	0.342	0.785
Multiverse (k=3,p=1)	0.344	0.805
Multiverse (k=5,p=1)	0.343	0.812
Multilayer Multiverse (k=2,p=2)	0.345	0.815

TABLE IV
CIFAR-100 RESULTS. A CONVENTIONAL NETWORK IS COMPARED TO MONOLAYER MULTIVERSE NETWORKS OF VARYING MULTIPLICITY AND TO A MULTILAYER MULTIVERSE NETWORK. THE NUMBERS INDICATE EITHER THE VALIDATION ERROR OR THE SAME/NOT-SAME ACCURACY IN THE TARGET DOMAIN.

We report the validation error during training on the source domain, and verification accuracy on the target domain. For measuring verification accuracy, we used the cosine distance to predict same/not-same on sampled pairs in the target domain. Note that the cosine distance is unsupervised, and hence, no training was done in the target domain. For the same/not-same evaluation on the CIFAR dataset, 3000 matching and 3000 non-matching pairs were randomly sampled from the 10 classes of the target domain.

As can be seen in Tab. IV, the multilayer multiverse method outperforms the baseline and the monolayer architectures on the target domain. Although we are mostly concerned with performance in the target domain, it is also evident that the validation error on the source domain is hardly affected.

As mentioned above, for the face recognition experiments, we use the scratch model [9]. The networks are trained on the CASIA dataset [9]. The LFW dataset [3] is used as the target domain.

Models are evaluated in the source domain by measuring the classification accuracy on the CASIA dataset, which we split to 90% training and 10% validation. For the target domain, the LFW benchmark in the unrestricted mode [4] is used (we do not use person ID from LFW, but do use the IDs of the CASIA dataset). The LFW results are mean accuracy estimated over the fixed ten cross-validation splits. The cosine distance is used to measure verification accuracy on the LFW splits.

Domain	Source	Target (transfer)
Metric	Casia Val error	LFW accuracy
Conventional (k=1,p=1)	0.080	0.963
Multiverse (k=3,p=1)	0.083	0.970
Multiverse (k=5,p=1)	0.084	0.972
Multilayer Multiverse (k=2,p=2)	0.084	0.971

TABLE V
FACE RECOGNITION RESULTS. SEE TAB. IV FOR DETAILS.

In the LFW experiments, we also performed the monolayer ($p = 1$) $k = 1$ (baseline), $k = 3$, and $k = 5$ experiments as well as a multilayer multiverse ($k = 2$,$p = 2$) experiments. As can be seen in Tab. V both the monolayer and the multilayer multiverse outperform the baseline method.

VII. DISCUSSION AND FUTURE WORK

While we have opened the door for the study of multilayer multiverse networks, the experiments presented only include a small fraction of their universe. There is no principled reason to focus, for example, on the very top layers. We are curious, for example, in observing the outcome of applying multiplicity to the first encoding layers in order to increase their robustness, e.g., to image noise.

The most remarkable property of multiverse networks is the emergence of a parameter independent dimensionality that is dependent solely on the data itself. We are unaware of any other methods to measure the effective dimensionality that does not depend on an underlying parameter. For multiverse networks, it seems that the dimensionality, given large enough k, is extremely stable, and does not change with k or with the regularization parameter λ. Moreover, this dimensionality is task dependent since it is obtained in a supervised network.

REFERENCES

[1] P. L. Bartlett, and S. Mendelson. Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. Journal of Machine Learning Research 3 463-482, 2002.

[2] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *European Conf. Computer Vision*, 2012.

[3] 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, October 2007.

[4] G. B. Huang and E. Learned-Miller. Labeled faces in the wild: Updates and new reporting procedures. UM-CS-2014-003, 2014.

[5] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Master's thesis, 2009.

[6] M. Lin, Q. Chen, and S. Yan. Network in network. In *International Conference on Learning Representations (ICLR)*, 2013.

[7] E. Littwin and L. Wolf. The Multiverse Loss for Robust Transfer Learning. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. Preprint as arXiv:1511.09033.

[8] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. 2015.

[9] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *CoRR*, abs/1411.7923, 2014.