Approximate Empirical Bayes for Deep Neural Networks

Han Zhao[*], **Yao-Hung Hubert Tsai**[*], **Ruslan Salakhutdinov**, **Geoff Gordon**
{hanzhao, yaohungt, rsalakhu, ggordon}@cs.cmu.edu
Machine Learning Department
Carnegie Mellon University

Abstract

We propose an approximate empirical Bayes framework and an efficient algorithm for learning the weight matrix of deep neural networks. Empirically, we show the proposed method works as a regularization approach that helps generalization when training neural networks on small datasets.

1 Introduction

The empirical Bayes methods provide us a powerful tool to obtain Bayesian estimators even if we do not have complete information about the prior distribution. The literature on the empirical Bayes methods and its applications is abundant [2, 6, 8, 9, 10, 15, 21, 23]. Existing studies on parametric empirical Bayes methods focus on the setting where the likelihood function and the prior are assumed to have specific forms, e.g., exponential family distribution and its conjugate prior, so that the marginal distribution of data has a closed form from which an estimator of the hyperparameter in the prior distribution can be obtained. While such assumption helps to simplify the setting in order to demonstrate the power of the empirical method, it restricts us from using more expressive and rich models.

Motivated by the success of the empirical Bayes method in the Gaussian settings, in this paper we explore extending it to expressive nonlinear models using deep neural networks. Although deep neural networks have been widely applied in various domains [14, 16, 18, 19], its parameters are learned via the principle of maximum likelihood in both classification and regression settings, hence its success crucially hinges on the availability of large scale datasets [25]. While different kinds of regularization approaches have been studied and designed for neural networks, e.g., weight decay [17], early stopping [4], Dropout [22] and the more recent DeCov [5] method, in this paper we propose a regularization approach for the weight matrix in neural networks through the lens of the empirical Bayes method. We aim to address the problem of overfitting when training large networks on small dataset. Our key insight stems from the famous argument by Efron [6]: It is beneficial to learn from the experience of others. Specifically, from an algorithmic perspective, we argue that the connection weights of neurons in the same layer (row/column vectors of the weight matrix) will be correlated with each other through the backpropagation learning, and hence by learning from other neurons in the same layer, a neuron can "borrow statistical strength" from other neurons to reduce overfitting.

As an illustrating example, consider the simple setting where the input $\mathbf{x} \in \mathbb{R}^d$ is fully connected to a hidden layer $\mathbf{h} \in \mathbb{R}^p$, which is further fully connected to the single output $\hat{y} \in \mathbb{R}$. Let $\sigma(\cdot)$ be the nonlinear activation function, e.g., ReLU [20], $W \in \mathbb{R}^{p \times d}$ be the connection matrix between the input layer and the hidden layer, and $\mathbf{a} \in \mathbb{R}^p$ be the vector connecting the output and the hidden layer. Without loss of generality ignore the bias term in each layer, we have: $\hat{y} = \mathbf{a}^T \mathbf{h}, \mathbf{h} = \sigma(W\mathbf{x})$. Consider using the usual ℓ_2 loss function $\ell(\hat{y}, y) = \frac{1}{2}|\hat{y} - y|^2$ and take the derivative of $\ell(\hat{y}, y)$ w.r.t. W. We obtain the update formula in backpropagation as follows:

$$W \leftarrow W - \alpha(\hat{y} - y)(\mathbf{a} \odot \mathbf{h}')\, \mathbf{x}^T,$$

where $\mathbf{h}'$ is the componentwise derivative of $\mathbf{h}$ w.r.t. its input argument, and $\alpha > 0$ is the learning rate. For any fixed $\mathbf{a} > 0$ (or $\mathbf{a} < 0$), realize that $(\mathbf{a} \odot \mathbf{h}')\, \mathbf{x}^T$ is a rank 1 matrix, and furthermore the component of $\mathbf{h}'$ is either 0 or 1. Hence the update for each row vector of W is linearly proportional to $\mathbf{x}$. Note that the observation holds for any input pair $(\mathbf{x}, y)$, so the update formula implies that the row vectors of W are correlated with each other. Similar analysis also holds for the column vectors of W. The above observation leads us to the following question: *can we define a prior distribution over W that captures the correlations between its row vectors and column vectors?*

[*] Equal contribution

2 Preliminary

In this section, we first introduce the notation used throughout the paper and then give a brief discussion on the empirical Bayes method [1, 7, 11] and its corresponding estimator [8, 9].

2.1 Notation and Setup

We use lowercase letter, such as y, to represent scalar and lowercase bold letter, such as $\mathbf{x}$, to denote vector. Capital letter, e.g., X, is reserved for matrix. Calligraphic letter, such as $\mathcal{D}$, is used to denote set. We write $\text{tr}(A)$ as the trace of a matrix A and $\text{vec}(A)$ as A's vectorization (by stacking its column vectors on top of one another). $[n]$ is used to represent the set $\{1, \ldots, n\}$ for any integer n. $\mathbf{a}_1 \odot \mathbf{a}_2$ stands for the Hadamard product (elementwise product) between two vectors $\mathbf{a}_1$ and $\mathbf{a}_2$.

Suppose we have access to a training set $\mathcal{D}$ of n pairs of data instances $(\mathbf{x}_i, y_i), i \in [n]$. We consider the supervised learning setting where $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y_i \in \mathcal{Y}$. For a regression problem, $\mathcal{Y} = \mathbb{R}$; for a binary classification problem, $\mathcal{Y} = \{1, -1\}$. Let $p(y \mid \mathbf{x}, \mathbf{w})$ be the conditional distribution of y given $\mathbf{x}$ with parameter $\mathbf{w}$. The parametric form of the conditional distribution is assumed be known. In this paper, we consider a Bayesian setting that the model parameter $\mathbf{w}$ is sampled from a prior distribution $p(\mathbf{w} \mid \theta)$ with hyperparameter θ. On the other hand, given $\mathcal{D}$, the posterior distribution of $\mathbf{w}$ is denoted by $p(\mathbf{w} \mid \mathcal{D}, \theta)$. From a Bayesian perspective, given an unseen instance $\mathbf{x}$, the goal is to infer the predictive distribution:

$$p(y \mid \mathbf{x}, \mathcal{D}, \theta) = \int p(y \mid \mathbf{x}, \mathbf{w}) \cdot p(\mathbf{w} \mid \mathcal{D}, \theta)\, d\mathbf{w}, \quad (1)$$

from which we can compute the mean, or the median, or other statistic (depending on the choice of the loss function) as our estimator of the unseen target variable y.

2.2 The Empirical Bayes Method

To solve the Bayesian inference in Equation (1), we need access to the value of the hyperparameter θ. However, complete information about the hyperparameter θ is usually not available in practice. Considering this issue, empirical Bayes method [8, 21] proposes to estimate θ from the data directly using the marginal distribution:

$$\begin{aligned} \hat{\theta} &= \arg\max_{\theta}\ p(\mathcal{D} \mid \theta) \\ &= \arg\max_{\theta} \int p(\mathcal{D} \mid \mathbf{w}) \cdot p(\mathbf{w} \mid \theta)\, d\mathbf{w}. \end{aligned} \quad (2)$$

Under specific choice of the likelihood function $p(\mathbf{x}, y \mid \mathbf{w})$ and the prior distribution $p(\mathbf{w} \mid \theta)$, e.g., exponential family distribution and its corresponding conjugate prior, we can solve the integral in (2) in closed form to obtain an analytic solution of $\hat{\theta}$, which can be subsequently plugged in (1) to obtain a Bayesian estimator for the model parameter $\mathbf{w}$ or to compute the predictive distribution of an unseen instance $\mathbf{x}$.

At a high level, by learning the hyperparameter θ in the prior distribution directly from data, the empirical Bayes method provides us a principled and convenient way to obtain an estimator of the model parameter $\mathbf{w}$. In fact, when both the prior and the likelihood functions are normal, it has been formally shown that the empirical Bayes estimators, e.g., the James-Stein estimator [15] and the Efron-Morris estimator [9], dominate the classic maximum likelihood estimator (MLE) in terms of quadratic loss for every choice of the model parameter $\mathbf{w}$. At a colloquial level, the success of empirical Bayes estimators can be attributed to the effect of *"learning from the experience of others"* [6], which also makes it a powerful tool in recent studies of multitask learning [26] and meta-learning [12].

3 Empirical Bayes Regularization

3.1 Approximate Empirical Bayes

When the likelihood function $p(\mathcal{D} \mid \mathbf{w})$ is implemented as a neural network, the marginalization in (2) over model parameter $\mathbf{w}$ cannot be computed exactly. Nevertheless, instead of performing expensive Monte-Carlo simulation, we can use point estimate of $\mathbf{w}$ to approximate the integral as follows:

$$\int p(\mathcal{D} \mid \mathbf{w}) \cdot p(\mathbf{w} \mid \theta)\, d\mathbf{w} \approx p(\mathcal{D} \mid \hat{\mathbf{w}}) \cdot p(\hat{\mathbf{w}} \mid \theta). \quad (3)$$

Note that the above approximation will be accurate if the likelihood under $\hat{\mathbf{w}}$ dominates the likelihoods under other model parameters.

Given an estimate $\hat{\mathbf{w}}$, by maximizing the R.H.S. of (3), we can obtain $\hat{\theta}$ as an approximator of the maximum marginal likelihood estimator of θ. As a result, we can use $\hat{\theta}$ to further refine the estimate $\hat{\mathbf{w}}$ by maximizing the posterior distribution as follows:

$$\hat{\mathbf{w}} \leftarrow \max_{\mathbf{w}}\ p(\mathbf{w} \mid \mathcal{D}) = \max_{\mathbf{w}}\ p(\mathcal{D} \mid \mathbf{w}) \cdot p(\mathbf{w} \mid \hat{\theta}). \quad (4)$$

The maximizer of (4) can in turn be used to better approximate the integral in (3). Formally, we can define the following optimization problem that characterizes our framework of the approximate empirical Bayes (AEB) method:

$$\max_{\mathbf{w}} \max_{\theta}\ \log p(\mathcal{D} \mid \mathbf{w}) + \log p(\mathbf{w} \mid \theta) \quad (5)$$

It is worth to connect the optimization problem (5) to the classic maximum a posteriori (MAP) inference and also discuss their difference: if we drop the inner optimization over the hyperparameter θ in the prior distribution, then for any fixed value $\hat{\theta}$, (5) reduces to MAP with the prior defined by the specific choice of $\hat{\theta}$, and the maximizer $\hat{\mathbf{w}}$ corresponds to the mode of the posterior distribution given by $\hat{\theta}$. From this perspective, the optimization problem in (5) actually defines a series of MAP inference problems, and the sequence $\{\hat{\mathbf{w}}_j(\hat{\theta}_j)\}_j$ defines a solution path towards the final approximate empirical Bayes estimator.

On the algorithmic side, the optimization problem (5) also suggests a natural block coordinate ascent algorithm where we alternatively optimize over $\mathbf{w}$ and θ until the convergence of the objective function. We will detail the discussion of the algorithm in next section where we give a specific prior distribution over the parameters of neural networks.

3.2 The Model and Algorithms

Inspired by the observation from Sec. 1, we propose to define a matrix-variate normal distribution [13] over the connection weight matrix W:

$$W \sim \mathcal{MN}(0_{p\times d}, \Sigma_r, \Sigma_c), \tag{6}$$

where $\Sigma_r \in \mathbb{S}^p_{++}$ and $\Sigma_c \in \mathbb{S}^d_{++}$ are the row and column covariance matrices, respectively. The probability density function $p(W \mid \Sigma_r, \Sigma_c)$ is given by:

$$p(W \mid \Sigma_r, \Sigma_c) = \frac{\exp\left(-\mathrm{tr}(\Sigma_r^{-1} W \Sigma_c^{-1} W^T)/2\right)}{(2\pi)^{pd/2} \det(\Sigma_r)^{d/2} \det(\Sigma_c)^{p/2}}$$

Equivalently, one can understand the matrix-variate normal distribution over W as a multivariate normal distribution with a Kronecker product covariance structure over $\mathrm{vec}(W)$: $\mathrm{vec}(W) \sim \mathcal{N}(0_{p\times d}, \Sigma_c \otimes \Sigma_r)$. It is then easy to check that the marginal prior distributions over the row and column vectors of the weight matrix W are given by:

$$W_{i:} \sim \mathcal{N}(\mathbf{0}_d, [\Sigma_r]_{ii} \cdot \Sigma_c), \quad W_{:j} \sim \mathcal{N}(\mathbf{0}_p, [\Sigma_c]_{jj} \cdot \Sigma_r)$$

We point out that the Kronecker product structure of the covariance matrix exactly captures our prior about the connection matrix W: the fan-in/fan-out weights of neurons in the same layer (row/column vectors of W) are correlated with the same correlation matrix, and they only differ at the scales (variances). Alternatively, if we treat learning each row/column vector as a separate task, then the distribution (6) corresponds to the common prior over different task vectors [26].

Define $\Omega_r := \Sigma_r^{-1}$ and $\Omega_c := \Sigma_c^{-1}$ to be the corresponding precision matrices and plug in the prior distribution (6) into the general approximate empirical Bayes framework (5). After routine algebraic simplifications, we reach the following concrete optimization problem that corresponds to the model we described in Sec. 1:

$$\begin{aligned} \min_{W,\mathbf{a}} \min_{\Omega_r,\Omega_c} \quad & \frac{1}{2}|\hat{y}(W,\mathbf{a}) - y|^2 + \lambda||\Omega_r^{1/2} W \Omega_c^{1/2}||_F^2 \\ & - \lambda\left(d \log\det(\Omega_r) + p \log\det(\Omega_c)\right) \\ \text{subject to} \quad & uI_p \preceq \Omega_r \preceq vI_p,\ uI_d \preceq \Omega_c \preceq vI_d \end{aligned} \tag{7}$$

where λ is a constant that only depends on p and d, and $0 < u \le v$ are two constants that guarantee the optimization problem (7) is well-posed.

It is not hard to show that in general the optimization problem (7) is not jointly convex in terms of $\{\mathbf{a}, W, \Omega_r, \Omega_c\}$, and this holds even if the activation function is linear and we do not have the hidden layer. However, for any fixed $\mathbf{a}, W$, the partial optimization over Ω_r and Ω_c is bi-convex, and more importantly, there is an efficient algorithm that finds the optimal $\Omega_r(\Omega_c)$ for any fixed $\mathbf{a}, W, \Omega_c(\Omega_r)$ in $O(p^2 \max\{d,p\})$ time [26]. Given a pair of constants u, v, we define the following thresholding function $\mathbb{T}_{[u,v]}(x)$:

$$\mathbb{T}_{[u,v]}(x) = \begin{cases} u, & x < u \\ x, & u \le x \le v \\ v, & x > v \end{cases}. \tag{8}$$

We summarize our block coordinate descent algorithm to solve (7) in Alg. 1. In each iterative loop, Alg. 1 takes a first-order algorithm $\mathfrak{A}$, e.g., the stochastic gradient descent, to optimize the parameters of the neural network by backpropagation. It then proceeds to compute the optimal solutions for Ω_r and Ω_c using the InverseThresholding as a sub-procedure. Alg. 1 terminates when some convergence conditions on model parameters $\mathbf{w}$ are satisfied.

4 Experiments

In Sec. 3 we develop our model and algorithms based on a simple neural network with one hidden layer and a single output. However, it is straightforward to extend our AEB framework to more sophisticated and practical models with various structures. In this section we demonstrate the effect of our AEB method on regularizing deep neural networks. Specifically, we consolidate our analysis on two different tasks. The first one is a multi-class classification problem where we define a matrix-variate prior over the weight matrix in the last softmax layer. In the second experiment, we consider a multitask learning setting where neural networks are used to learn the shared representations among all the tasks [3], and we apply our AEB method on the last layer weight matrix, where each row corresponds to a separate task vector.

Multi-class Classification. We consider the MNIST dataset where the target is to classify an input 28×28

Algorithm 1 Block Coordinate Descent for Approximate Empirical Bayes

Input: Initial value $\mathbf{w}^0 := \{\mathbf{a}^{(0)}, W^{(0)}\}$, $\Omega_r^{(0)}$ and $\Omega_c^{(0)}$, first-order optimization algorithm $\mathfrak{A}$, constants $0 < u \leq v$.

for $t = 1, \ldots, \infty$ until convergence **do**
  Fix $\Omega_r^{(t-1)}$, $\Omega_c^{(t-1)}$, optimize $\mathbf{w}^{(t)}$ by backpropagation and algorithm $\mathfrak{A}$
  $\Omega_r^{(t)} \leftarrow \text{InvThresholding}(W^{(t)}\Omega_c^{(t-1)}W^{(t)T}, d, u, v)$
  $\Omega_c^{(t)} \leftarrow \text{InvThresholding}(W^{(t)T}\Omega_r^{(t)}W^{(t)}, p, u, v)$
end for

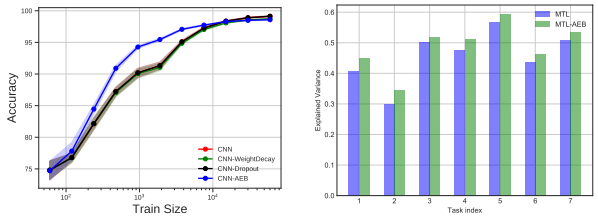
procedure INVTHRESHOLDING(Δ, m, u, v)
  Compute SVD: $Q\text{diag}(\mathbf{r})Q^T = \text{SVD}(\Delta)$
  Threshold $\mathbf{r}' \leftarrow \mathbb{T}_{[u,v]}(m/\mathbf{r})$ (see (8) for the definition of $\mathbb{T}_{[u,v]}(\cdot)$)
  return $Q\text{diag}(\mathbf{r}')Q^T$
end procedure

digit image into $\{0, \ldots, 9\}$ categories. In this experiment, we would like to show that AEB provides an effective regularization on the network parameters when the training set size is small. To this end, we use a convolutional neural network as our baseline model with the following structure: $\mathsf{CONV}_{5\times5\times1\times10}$-$\mathsf{CONV}_{5\times5\times10\times20}$-$\mathsf{FC}_{320\times50}$-$\mathsf{FC}_{50\times10}$. The notation $\mathsf{CONV}_{5\times5\times1\times10}$ denotes the convolutional layer with kernel size 5×5 from depth 1 to 10; the notation $\mathsf{FC}_{320\times50}$ denotes the fully connected layer with size 320×50. We ignore bias term for brevity. As a comparison, in our AEB model we place a matrix-variate normal prior over the weight matrix of the last softmax layer, and we use Alg. 1 to optimize both the model weights of the convolutional network and the two covariance matrices of the weight matrix in the last layer.

We compare our AEB method with classic regularization methods in the literature of deep learning, including weight decay and dropout. To show the effect of regularization, we gradually increase the training set size from 60 to 60,000 for 11 different experiments. For each training set size, we repeat the experiments for 10 times, and we show the mean along with its standard deviation as error bars in Fig. 1a.

It is clear from Fig. 1a that AEB helps generalization by reducing overfitting, and its regularization effect is particularly beneficial when the training set size is small. From the experiments we can also observe that the variance of the CNN-AEB model is significantly smaller than all the other methods, even if training set size is small.

Multitask Learning. In the second experiment we consider a dataset for multitask learning. This data relates to an inverse dynamics problem for a seven degree-of-freedom (DOF) SARCOS anthropomorphic robot arm [24]. The goal of this task is to map from a 21-dimensional input space (7 joint positions, 7 joint velocities, 7 joint accelerations) to the corresponding 7 joint torques. Hence there are 7 tasks and the inputs are shared among all the tasks. The training set and test set contain 44,484 and 4,449 examples, respectively.
Inspired by the admissible property of the empirical Bayes



(a) Different regularization methods on MNIST.

(b) Explained variance on the sarcos dataset.

Figure 1: Best viewed zoomed in and in color.

estimator, we are interested in investigating whether our AEB estimator can lead to better generalization for multiple regression problems, as empirical Bayes estimators do. To do so, we use a baseline fully-connected network (MTL) with the following structure: $\mathsf{FC}_{21\times256}$-$\mathsf{FC}_{256\times100}$-$\mathsf{FC}_{100\times7}$, where the 7 output units correspond to 7 different tasks. Again, in MTL-AEB we place a prior over the task weight matrix and we optimize for both model parameters and two covariance matrices. We plot and show the explained variance (between 0 and 1, the larger the better) of MTL and MTL-AEB in Fig. 1b. The results in Fig. 1b confirm our hypothesis: by making use of the effect "learning from the experience of ours", AEB estimator consistently reduces the test set mean squared error and improves generalization.

5 Conclusion

We propose an approximate empirical Bayes method for learning the model parameters of deep neural networks. On two datasets we demonstrate that our AEB method helps to reduce overfitting and hence improves generalization when the training set is small. One future direction is to develop a more efficient approximate solution to optimize the two covariance matrices so that our method can scale to larger networks. It is also interesting to see whether applying our AEB method to all the layers can further reduce overfitting or not.

References

[1] J. M. Bernardo and A. F. Smith. Bayesian theory, 2001.

[2] B. P. Carlin and T. A. Louis. *Bayes and empirical Bayes methods for data analysis*. Chapman and Hall/CRC, 2010.

[3] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[4] R. Caruana, S. Lawrence, and C. L. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems*, pages 402–408, 2001.

[5] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra. Reducing overfitting in deep networks by decorrelating representations. *arXiv preprint arXiv:1511.06068*, 2015.

[6] B. Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press, 2012.

[7] B. Efron and T. Hastie. *Computer age statistical inference*, volume 5. Cambridge University Press, 2016.

[8] B. Efron and C. Morris. Stein's estimation rule and its competitors—an empirical Bayes approach. *Journal of the American Statistical Association*, 68 (341):117–130, 1973.

[9] B. Efron and C. Morris. Stein's paradox in statistics. *Scientific American*, 236(5):119–127, 1977.

[10] B. Efron, R. Tibshirani, J. D. Storey, and V. Tusher. Empirical bayes analysis of a microarray experiment. *Journal of the American statistical association*, 96 (456):1151–1160, 2001.

[11] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. CRC press, 2013.

[12] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[13] A. K. Gupta and D. K. Nagar. *Matrix variate distributions*. Chapman and Hall/CRC, 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 361–379, 1961.

[16] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[17] A. Krogh and J. A. Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[18] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a backpropagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[19] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[20] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[21] H. Robbins. An empirical bayes approach to statistics. Technical report, Columbia University, New York City, United States, 1956.

[22] 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*, 15(1):1929–1958, 2014.

[23] C. Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. Technical report, Stanford University, Stanford, United States, 1956.

[24] S. Vijayakumar and S. Schaal. Locally weighted projection regression: Incremental real time learning in high dimensional space. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1079–1086. Morgan Kaufmann Publishers Inc., 2000.

[25] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[26] H. Zhao, O. Stretcu, R. Negrinho, A. Smola, and G. Gordon. Efficient multi-task feature and relationship learning. *arXiv preprint arXiv:1702.04423*, 2017.