Degrees of Freedom in Deep Neural Networks

Tianxiang Gao and **Vladimir Jojic**
Computer Science Department
University of North Carolina at Chapel Hill
Chapel Hill, NC, 27514, USA
{*tgao,vjojic*}*@cs.unc.edu*

Abstract

In this paper, we explore degrees of freedom in deep sigmoidal neural networks. We show that the degrees of freedom in these models are related to the expected *optimism*, which is the expected difference between test error and training error. We provide an efficient Monte-Carlo method to estimate the degrees of freedom for multi-class classification methods. We show that the degrees of freedom is less than the parameter count in a simple XOR network. We extend these results to neural nets trained on synthetic and real data and investigate the impact of network's architecture and different regularization choices. The degrees of freedom in deep networks is dramatically less than the number of parameters. In some real datasets, the number of parameters is several orders of magnitude larger than the degrees of freedom. Further, we observe that for fixed number of parameters, deeper networks have less degrees of freedom exhibiting a regularization-by-depth. Finally, we show that the degrees of freedom of deep neural networks can be used in a model selection criterion. This criterion has comparable performance to cross-validation with lower computational cost.

1 INTRODUCTION

Model selection is one of the key tasks in machine learning, as method's performance on training data is an optimistic estimate of its general performance. Efron [2004] provided an estimate of optimism, difference of error on test and training data, and related it to a measure of model's complexity deemed effective degrees of freedom. This result reflects Occam's razor since models with higher degrees of freedom tends to have higher optimism. Degrees of freedom, defined as parameter counts, have been frequently used in model selection. However, even in linear models, the number of parameters are not a good indicator of model's complexity. Straightforward examples of this behavior are models fit using sparsity penalties. In that context, degrees of freedom are related to the number of non-zero parameters instead of total parameter count.

Ye [1998] introduced the concept of Generalized Degrees of freedom (GDF) for complex modeling procedures with Gaussian distributed outputs. GDF is defined based on the sensitivity of the fitted values to the perturbations in observed values. Efron [2004] provided a framework for estimating degrees of freedom for modeling procedures with output in exponential family distribution. In order to estimate degrees of freedom in deep neural networks for classification problems, where the outputs can be regarded as a categorical distribution, we extend Efron's results to the context of multinomial logistic regression. Similar to Ye's GDF, the computation of the degrees of freedom involves assessing network's changes in output as a result of perturbation of the training data. The more sensitive the network's output to the perturbation, the more degrees of freedom it has.

We provide a straightforward algorithm for evaluating the degrees of freedom for any modeling procedure with outputs in categorical distribution form. This algorithm requires an additional run of the modeling procedure on the perturbed data. In the worst case, this amounts to doubling the running time of the procedure. Using this algorithm we first analyze the complexity of XOR network. This simple example highlights the fact that the degrees of freedom in a neural net is not simply equal to the total number of parameters in the network.

In our experiments, we aim to answer following questions:

1. How does the network's complexity (DoF) vary with its architecture? Specifically, how do the degrees of freedom grow with the depth of a neural network?

2. How does regularization affect network's complexity? Specifically, what is the impact of dropout, weight decay, adding noise on the degrees of freedom?

We answer these questions in the context of feed-forward sigmoidal networks employed on classification tasks on both synthetic and real datasets.

The prior work on the model complexity is rich, and we briefly review some key contributions. Bayesian Information Criterion (BIC) [Schwarz et al., 1978] and Akaike Information Criterions (AIC) [Akaike, 1974] are most commonly used techniques for model selection. Both aim to construct an estimate of the test log-likelihood by correcting the training set log likelihood with terms dependent on the number of parameters in the model in order to produce a score that is a less biased estimate of test log-likelihood. The weighting of the parameter count is different, BIC depends on the sample size, and AIC uses a constant. BIC applied to the family of models that contain the true model is consistent in the limit of the data. AIC, with some mild constraints, guarantees the selection of model with least square error, among models that do not include the true model. Crucial to the practical application of these methods is the correct count of parameters. Bayesian model selection elegantly avoids the need to specify the complexity of the network by evaluating evidence, a marginal probability of the data given the model. This approach marginalizes over all of the parameters, making models of different parameterizations comparable. The size of the parameter space directly impacts the evidence through this integration, as the prior on parameters gets spread thinly across high dimensional spaces. Unfortunately, the cost of computing such integrals is often prohibitive, but the models selected using these techniques have been shown to be very competitive. [MacKay, 2003, Neal, 1996, Guyon et al., 2004]. Kolmogorov-Chaitin complexity [Kolmogorov, 1965] describes dataset complexity in terms of a program that recapitulates the data. Generation of task-specific neural networks using algorithmically simple programs was explored by Schmidhuber [1997]. Networks whose parameters could not be captured by a simple program were avoided. A related method of Minimal Description Length reflects the desire for compact representation of the data. Its application [Hinton and Zemel, 1994] shows how the trade-off between the data and parameter compression can lead to an objective for training auto-encoders. Degrees of freedom of linear models fit with Lasso-type penalties have been analyzed, e.g. Lasso [Zou et al., 2007], Fused Lasso [Tibshirani et al., 2005] and Group Lasso [Vaiter et al., 2012]. The number of predictors and the number of degrees of freedom greatly differ due to the imposed sparsity and weight tying. Recent results on degrees of freedom for non-continuous procedures such as best subset regression and forward stagewise regression [Janson et al., 2015] highlight challenges in determining the complexity of these procedures as the estimators can be discontinuous. Research on Stein's Unbiased Risk Estimate has yielded model selection techniques [Stein, 1981] as well as algorithms for their estimation [Ye, 1998, Ra-

mani et al., 2008]. Generalization of SURE to exponential families has been proposed by Eldar [2009]. However, its focus is on estimating parameter risk instead of prediction error. In linear models, the two neatly coincide. But this does not carry over to logistic regression and more broadly sigmoidal neural networks.

2 DEGREES OF FREEDOM FOR CATEGORICAL DISTRIBUTION

In this section, we derive the definition of degrees of freedom for categorical distribution from the optimism according to Efron [2004]. Then, we introduce an efficient Monte-Carlo sampling based method [Ramani et al., 2008] to estimate degrees of freedom.

2.1 DEFINITIONS

We focus on models aimed at multi-class classification task. The data is assumed to be composed of features $\mathbf{X} \in \mathbb{R}^{n \times p}$, and output labels $\mathbf{y}$ range over k categories. We will denote categorical distribution with $\mathcal{C}(\cdot)$. Categorical distribution over k categories can be parameterized using a vector of non-negative values with a sum of 1. We treat sample label y_i as realization of categorical random variables for a specific parameter vector $\boldsymbol{\mu}_i$. Hence $y_i \sim \mathcal{C}(\boldsymbol{\mu}_i)$, where $\boldsymbol{\mu}_i = [\mu_{i1}, \mu_{i2}, \ldots, \mu_{ik}]$ is the **true probability** of sample y_i being in each class. $\mu_{ic} \in [0, 1]$ and $\sum_{c=1}^{k} \mu_{ic} = 1$. Members of exponential family follow form:

$$f(\mathbf{p}_i | \boldsymbol{\mu}_i) = r(\mathbf{p}_i) \exp\{\boldsymbol{\theta}(\boldsymbol{\mu}_i)^T \mathbf{p}_i - A(\boldsymbol{\mu}_i)\}$$

where $\mathbf{p}_i$ is the vector of sufficient statistics for sample i, $\boldsymbol{\theta}(\boldsymbol{\mu}_i)$ is the vector of natural parameters, $r(\mathbf{p}_i)$ is the base measure, $A(\boldsymbol{\mu}_i)$ is the log-partition function.

For categorical distribution with parameter $\boldsymbol{\mu}_i$, we have $\mathbf{p}_i = h(y_i) = [\delta(y_i - 1), \ldots, \delta(y_i - k - 1)]^T$, where $\delta(\cdot)$ is the Kronecker delta function, $\delta(a) = 1$ if $a = 0$, $\delta(a) = 0$ if $a \neq 0$. In other words, $\mathbf{p}_i$ is a vector of the **observations** of sample i being in each class. Base measure is $r(\mathbf{p}_i) = 1$; natural parameters are $\theta_c(\boldsymbol{\mu}_i) = \ln \mu_{ic} - \ln(1 - \sum_{l=1}^{k-1} \mu_{il})$, and log partition function $A(\boldsymbol{\mu}_i) = \ln(1 + \sum_{c=1}^{k-1} e^{\theta_c(\boldsymbol{\mu}_i)})$. Note that both $\boldsymbol{\mu}_i$ and $\mathbf{p}_i$ are of dimension $k - 1$. Let $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_n]^T$ be the matrix of observations for all sample labels $y_1, \ldots, y_n$.

2.2 OPTIMISM IN MODELS WITH CATEGORICAL DISTRIBUTION

Optimism is the difference between expected test log deviance error and training log deviance error for a model fitting procedure. It is related to the complexity of the model and degrees of freedom is derived from optimism. If the optimism for a modeling procedure can be estimated, we

can use it for model selection. Efron [2004] provides the derivations of expected optimism for the single parameter exponential family. We follow Efron's approach to derive the definition of degrees of freedom for modeling procedure with output in categorical distribution form.

Given sample input $\mathbf{x}_i$, we assume that the output label $y_i \sim \mathcal{C}(\boldsymbol{\mu}_i)$. Let $\hat{\boldsymbol{\mu}}_i = \mathcal{L}(\mathbf{p}_i)$ be the **estimated probability** for sample i from observations $\mathbf{p}_i$. The log deviance error for $\hat{\boldsymbol{\mu}}_i$ and $\mathbf{p}_i$ is:

$$
\begin{aligned}
\text{err}_i &= -2\log f(\mathbf{p}_i|\hat{\boldsymbol{\mu}}_i^T) \\
&= -2[\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i)^T \mathbf{p}_i - A(\hat{\boldsymbol{\mu}}_i)]
\end{aligned}
$$

Suppose we have another sample y_i^0 drawn from the same distribution as y_i, $y_i^0 \sim \mathcal{C}(\mu_i)$. Let $\mathbf{q}_i = h(y_i^0)$ be the vector of its observations. The expected log deviance error of $\mathbf{q}_i$ using $\hat{\boldsymbol{\mu}}_i$ is:

$$
\begin{aligned}
\text{Err}_i &= E_{y_i^0}\{-2\log f(\mathbf{q}_i|\hat{\boldsymbol{\mu}}_i)\} \\
&= -2[\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i)^T \boldsymbol{\mu}_i + A(\hat{\boldsymbol{\mu}}_i)]
\end{aligned}
$$

The definition of optimism is:

$$
\begin{aligned}
O_i &= \text{Err}_i - \text{err}_i \\
&= 2\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i)^T(\mathbf{p}_i - \boldsymbol{\mu}_i)
\end{aligned}
$$

Hence, optimism is the difference between log deviance error on the training set and expected log deviance error with respect to the true distribution.

The expected optimism over $y_i \sim \mathcal{C}(\boldsymbol{\mu}_i)$ for the estimated probability $\hat{\boldsymbol{\mu}}_i$ and true probability $\boldsymbol{\mu}_i$ is:

$$
\Omega_i = 2E_{y_i}\{\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i)^T(\mathbf{p}_i - \boldsymbol{\mu}_i).\}
$$

As we do not know the true probability $\boldsymbol{\mu}_i$, we cannot compute the expected optimism. However, we can get an approximate measurement using Taylor series expansion. We can approximate $\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i)$ by taking the Taylor series expansion at $\mathbf{p}_i = \boldsymbol{\mu}_i$ to obtain:

$$
\boldsymbol{\theta}(\hat{\boldsymbol{\mu}}_i) \approx \boldsymbol{\theta}(\mathcal{L}(\boldsymbol{\mu}_i)) + \mathbf{D}^{(i)}(\mathbf{p}_i - \boldsymbol{\mu}_i).
$$

$\mathbf{D}^{(i)}$ is the first derivative matrix where each entry $D_{jc}^{(i)} = \frac{\partial \theta_j(\mathcal{L}(\mathbf{v}))}{\partial v_c}|_{\mathbf{v}=\boldsymbol{\mu}_i}$.

Therefore, we can approximate expected optimism as:

$$
\begin{aligned}
\tilde{\Omega}_i &= 2E_{y_i}\{[\boldsymbol{\theta}(\mathcal{L}(\boldsymbol{\mu}_i)) + \mathbf{D}^{(i)}(\mathbf{p}_i - \boldsymbol{\mu}_i)]^T(\mathbf{p}_i - \boldsymbol{\mu}_i)\} \\
&= 2E_{y_i}\{\sum_{j=1}^{k-1}\sum_{l=1}^{k-1}(p_{ij} - \mu_{ij})(p_{il} - \mu_{il})D_{jl}^{(i)}\} \\
&= 2\sum_{j=1}^{k-1}\sum_{l=1}^{k-1}\text{cov}(p_{ij}, p_{il})\frac{\partial \theta_j(\mathcal{L}(\mathbf{v}))}{\partial v_l}|_{\mathbf{v}=\boldsymbol{\mu}_i}
\end{aligned}
$$

We can estimate the expected optimism by assuming $p_i \sim \mathcal{C}(\hat{\boldsymbol{\mu}}_i)$, so:

$$
\hat{\Omega}_i = 2\sum_{j=1}^{k-1}\sum_{l=1}^{k-1}\text{cov}(p_{ij}, p_{il})\frac{\partial \theta_j(\mathcal{L}(\mathbf{v}))}{\partial v_l}|_{\mathbf{v}=\hat{\boldsymbol{\mu}}_i}. \quad (1)
$$

In categorical distribution, $\text{cov}(p_{ij}, p_{il}) = -\hat{\mu}_{ij}\hat{\mu}_{il}$, if $i \neq j$. $\text{var}(p_{ij}) = \hat{\mu}_{ij}(1 - \hat{\mu}_{ij})$. Therefore, Equation (1) can be reduced to:

$$
\hat{\Omega}_i = 2\sum_{j=1}^{k-1}\frac{\partial \mathcal{L}_j(\mathbf{v})}{\partial v_j}|_{\mathbf{v}=\hat{\boldsymbol{\mu}}_i}. \quad (2)
$$

The proof is given in the supplementary material.

Equation (2) for $k = 2$ is exactly the result for Bernoulli distribution derived in [Efron, 2004]. Efron also showed that Eqn (2) gives the correct degrees of freedom for maximum likelihood estimation [Efron, 1975]. In a p-parameter curved exponential family, we have:

$$
\sum_{i=1}^{n}\frac{\partial \mathcal{L}(\mathbf{v})}{\partial v_i}|_{\mathbf{v}=\hat{\boldsymbol{\mu}}_i} = p.
$$

Here, we define the degrees of freedom for a classification model estimator $\hat{\boldsymbol{\mu}}_i = \mathcal{L}_i(\mathbf{P})$ on all the data samples $\mathbf{P}$ to be:

$$
\text{df} = \sum_{i=1}^{n}\sum_{c=1}^{k-1}\frac{\partial \mathcal{L}_{ic}(\mathbf{P})}{\partial p_{ic}}. \quad (3)
$$

This definition tells that the degrees of freedom is the sum of each sample's sensitivity of its estimated probability to the perturbations in its observation for all categories.

2.3 DEGREES OF FREEDOM FOR MODEL SELECTION

As degrees of freedom is related to the expected optimism, we can use degrees of freedom for model selection. According to Equation (2) and (3), the relationship between expected test and training log deviance errors is:

$$
\sum_{i=1}^{n}E_{y_i}\{\text{Err}_i\} = \sum_{i=1}^{n}E_{y_i}\{\text{err}_i\} + 2\text{df}. \quad (4)
$$

Euqation (4) is very similar to Akaike Information Criterions (AIC) Akaike [1974]:

$$
\text{AIC} = \sum_{i=1}^{n}\text{err}_i + 2k, \quad (5)
$$

where k is the number of parameters. We refer to 2df in Equation (4) and $2k$ in Equation (5) as "complexity correction" for training log deviance error. In simple linear

regression models, df $= k$, and the complexity corrections are the same. However, in complex models such as deep neural networks, simply counting number of parameters can result in overestimate of the expected test log deviance error. Therefore, we introduce DoFAIC for model selection:

$$\text{DoFAIC} = \sum_{i=1}^{n} \text{err}_i + 2\text{df}. \qquad (6)$$

DoFAIC uses degrees of freedom instead of the number of parameters for complexity correction. We assume that DoFAIC can produce a better criterion for model selection than Naïve AIC.

2.4 MONTE-CARLO ESTIMATE FOR DEGREES OF FREEDOM

For most practical estimators of the model's predictions with respect to the data derivatives, $\frac{\partial \mathcal{L}_{ic}(\mathbf{P})}{\partial p_{ic}}$ are not available in closed form. For example, fitting multinomial logistic regression using stochastic gradient descent with adaptive learning rates requires a fairly sophisticated derivation which accounts for changes in step-sizes as a result of data perturbation. For deep neural networks, this difficulty grows due to the use of back-propagation. In this paper, we used a sampling based method to efficiently estimate

Monte-Carlo estimation A theoretical result for a stochastic estimate of the degrees of freedom of nonlinear estimators has been proposed by Ramani et al. [2008]. We restate the key result from that paper here.

Theorem 1. *Let* b *be a zero mean i.i.d. random vector (that is independent of y) with unit variance and bounded higher moments. Then*

$$\sum_i \frac{\partial f(\mathbf{y})}{\partial y_i} = \lim_{\epsilon \to 0} E_{\mathbf{b}} \left[\mathbf{b}^T \left(\frac{f(\mathbf{y} + \epsilon \mathbf{b}) - f(\mathbf{y})}{\epsilon} \right) \right]$$

provided that f *admits a well-defined second-order Taylor expansion.*

We sketch out a proof that the prediction in a neural net via forward pass is a smooth function of the observations of training labels. We will abbreviate "differentiable with respect to observations" as d.w.r.t.o. Sigmoid and soft-max are smooth functions of their inputs. The cross-entropy loss is a multivariate function that depends on data and weights, and all of its partial derivatives exist. For simplicity, we assume that the network is trained using gradient descent. Each update of the network's parameters is a linear combination of previous weights and a gradient of the loss. Assuming that the initial weights d.w.r.t.o. and loss is smooth then the update yields weights that are d.w.r.t.o. Random initialization and pre-training both yield initializations that are independent of observations, hence the partial derivatives of the initial weights with respect to observations are

---

Algorithm 1 Monte Carlo algorithm for computing degrees of freedom of a multi-class classifier

Input: training data $\mathbf{X} \in \mathbf{R}^{n \times p}$, $\mathbf{y} \in \{1, 2, .., k\}^n$
1: Compute observations matrix $\mathbf{P} = h(\mathbf{y})$
2: Train model on $\mathbf{X}$ and $\mathbf{P}$
3: Compute estimated probabilities for each sample $\mathcal{L}(\mathbf{P})$

4: Sample entries of $\mathbf{B}^{(t)} \in \mathbf{R}^{p \times k}$ from zero-mean, unit variance normal distribution
5: Train model on $\mathbf{X}$ and $\mathbf{P}^{(t)} = \mathbf{P} + \epsilon \mathbf{B}^{(t)}$
6: Using trained model compute estimated probabilities for each sample $\mathcal{L}(\mathbf{P}^{(t)})$;
7: Repeat 4-6 for T times;
8: Calculate df from Equation (7)

---

0. By induction, gradient descent, at any iteration, yields weights that are d.w.r.t.o. Forward pass through sigmoidal network yields estimated probabilities which are smooth with respect to observations. Thus, the Taylor expansion required by the above theorem exists.

Using this theorem, we can evaluate the derivative of a function $\frac{\partial f(x)}{\partial x}$ by perturbing the inputs. We applied a modified version of the method [Ramani et al., 2008] for categorical distribution. We applied random perturbation to the observations to estimate the degrees of freedom:

$$\text{df} = \sum_{i=1}^{n} \sum_{c=1}^{k-1} \frac{\partial \mathcal{L}_{ic}(\mathbf{P})}{\partial p_{ic}}$$
$$= \lim_{\epsilon \to 0} \left\{ E_{\mathbf{B}} \left[\sum_i \sum_c b_{ic} \left(\frac{\mathcal{L}_{ic}(\mathbf{P} + \epsilon \mathbf{B}) - \mathcal{L}_{ic}(\mathbf{P})}{\epsilon} \right) \right] \right\},$$

where $\mathbf{B}$ is a zero-mean i.i.d. random matrix with unit variance and bounded higher order moments. Therefore, we can approximate df with T independent samplings of $\mathbf{B}^{(t)}$:

$$df \approx \frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{n} \sum_{c=1}^{k-1} b_{ic}^{(t)} \left(\frac{\mathcal{L}_{ic}(\mathbf{P} + \epsilon \mathbf{B}^{(t)}) - \mathcal{L}_{ic}(\mathbf{P})}{\epsilon} \right),$$
$$(7)$$

where ϵ is a small value. In our experiments, we choose $\epsilon = 10^{-5}$. To better estimate the sensitivity, we can use the average of multiple runs as the final estimation. The algorithm for estimating degrees of freedom is summarized in Algorithm 1.

Note that training on original and perturbed observations matrix can be performed in parallel. Finally, we also derived analytical derivatives for stochastic gradient descent learning which yields the same degrees of freedom as the algorithm presented above. However, this method requires maintenance of partial derivatives of each parameter with respect to each sample's observations. Such storage requirements make this method impractical for real world applications.

Variance reduction For deep neural networks, training takes a considerable amount of time. In order to estimate

degrees of freedom in a reasonable computational time, we used a variance reduction technique – common random numbers – during Monte-Carlo sampling. When comparing the degrees of freedom on a specific data, fixed $\mathbf{P}$, for several different fitting procedures, we used the same perturbation matrix $\mathbf{B}$ for all the models. We used the same random seed for all models throughout the training. For example, in deep neural network training, we use the same random seed to initialize weights and bias; during pre-training with denoising-autoencoders, we use the same random seed for drop-out and input corruptions. For stochastic gradient descent methods, we use the same mini-batches splittings during training. In our experiment, we found that we can estimate degrees of freedom well enough using just one perturbed copy of the data when using these variance reduction techniques.

2.5 DEGREES OF FREEDOM IN MULTINOMIAL LOGISTIC REGRESSION

In order to validate the above algorithm in a setting with known degrees of freedom, we perform an empirical analysis of the degrees of freedom in different multinomial logistic regression models.

We generate an i.i.d. zero mean unit variance random design matrix $\mathbf{X}$ with $n = 100$ samples and $p = 20$ features. We represent each sample with $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{ip}]$. With $k = 4$ class, we generated a random weight matrix $\mathbf{W} \in \mathbb{R}^{p \times k}$, where each entry $w_{ic} \sim \mathcal{N}(0, 1)$. We generate each label from $y_i = \operatorname{argmax}_j \mu_{ij}$, where $\boldsymbol{\mu}_i = e^{\mathbf{x}_i \mathbf{W}}$.

We fit 5 models using multinomial logistic regression. In ith model, we only use first $2i$ features in $\mathbf{X}$ to fit. Therefore, ith model only contains $2(i + 1)(k - 1)$ parameters and the degrees of freedom are equal to the number of parameters. We perform 5 Monte-Carlo degrees of freedom estimates for each model.

We plot degrees of freedom in Figure 1(a). We observed that degrees of freedom are very close to the number of parameters we used in the model. The standard error for Monte-Carlo estimate is small.

We also randomly generated 1000 samples for testing. Optimism is calculated by the difference between average testing log deviance error and training log deviance error. We plot the degrees of freedom and optimisms for all 5 models in Figure 1(b). It shows that the optimism has a linear relationship with degrees of freedom, as expected.

2.6 DEGREES OF FREEDOM OF A XOR NETWORK

We generated a small synthetic example using exclusive-or (XOR) operator, where $\text{XOR}(a, b) = 0$ if $a = b$, and $\text{XOR}(a, b) = 1$ if $a \neq b$. Given an input $x_1, x_2 \in \{0, 1\}$, the output $y = XOR(x_1, x_2)$, we hope to learn a model of
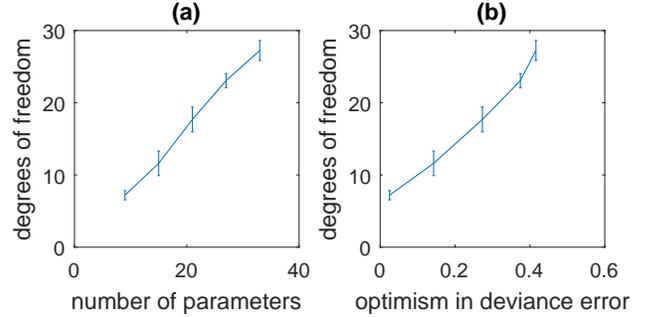
Figure 1: **(a)** Comparison between degrees of freedom estimates in multinomial logsitic regression and the true number of parameters used in the model. **(a)** Comparison between degrees of freedom estimates in multinomial logsitic regression and the optimism in log deviance error. In each plot, blue line is the mean of the five Monte-Carlo estimates. Error bar represents the standard error.

XOR operator. In general, we can build a neural network with two hidden nodes as shown in Figure 2 and weights in Table 1 to learn a perfect XOR classifier.
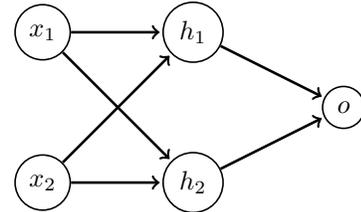
Figure 2: A Neural Network with 2 Hidden Nodes

Table 1: An XOR Network

x_1	0	1	0	1
x_2	0	0	1	1
$h_1 = \sigma(F(-0.5 + x_1 - x_2))$	0	1	0	0
$h_2 = \sigma(F(-0.5 - x_1 + x_2))$	0	0	1	0
$y = \sigma(K(-0.5 + h_1 + h_2))$	0	1	1	0

A network that trained properly should have weight matrix with form in Table 1. If x contains no noise, F, a multiplier, can be infinitely large to achieve perfect estimation. Therefore, we set y to be 0.9 instead of 1.

We train networks with different structures on XOR data using back-propagation and estimate their degrees of freedom using Monte-Carlo method. Even though there are 9 parameters in the network, we found that the degrees of freedom for all learned models is 4. We note that the symmetry in weights of the inputs to the two hidden nodes, eliminates degrees of freedom, as does implicit tying of the weights of inputs to the output node. To give an intuition

why this tying occurs, we note that the predominantly correctly labeled data drives the network to keep the weights close to each other. Hence, a small perturbation in the labels can affect multiple weights simultaneously, but does not disturb their balance. This observation encourages us to investigate deeper models.

3 DEGREES OF FREEDOM IN DEEP NEURAL NETWORKS

In this section, we investigate degrees of freedom in deep neural network models. From the XOR example, we know that the degrees of freedom in a network is not equal the number of parameters in the model. The structure of the network and different regularization techniques will impact degrees of freedom.

3.1 TERMINOLOGIES AND SETTINGS

In the following experiments, we explore deep networks trained to solve larger classification problems. Each of the networks takes real value vector $\mathbf{x}_i \in \mathbb{R}^{p \times 1}$ as input and outputs the probability $\hat{\boldsymbol{\mu}}_i$ for this sample being in one of k categories. We use sigmoid activation function for all the hidden nodes and a soft-max in the last layer. The number of hidden layers is called "**depth**" of the network. We only consider networks with an equal number of units in each hidden layer, and we call this number "**width**" of the network. Next, we investigate degrees of freedom in networks with different width and depth.

Stacked-Auto-Encoder (SdA) pre-training We used SdA [Vincent et al., 2010] to pre-train the neural network with input dataset, as unsupervised pre-training helps the network to achieve a better generalization from the training data on supervised tasks [Erhan et al., 2010]. In denoising auto-encoder, **corruption** is used in layer-wised pre-training. The corruption is introduced by zeroing out input to the auto-encoder with a certain probability. The chosen probability of corruption is called **corruption rate**. **Dropout** [Srivastava et al., 2014] is also used during the pre-training of SdA, where output of hidden units are randomly zeroed with probability, which is called **dropout rate**. We assume that increasing in corruption rate or dropout rate will reduce degrees of freedom as they provide more regularization to the network.

Weight-decay We used a weight decay penalty on the sum of the squares of all the weights in the network during both pre-training and fine-tuning stage. Adding this penalty prevents the network from over-fitting. We refer to the multiplier associated with the sum of squares as **weight decay rate**. We expect to see that the degrees of freedom drops with increasing weight decay rate.

Implementation All our code are based on Theano [Bastien et al., 2012, Bergstra et al., 2010] and we ran experiments on a cluster of machines with NVIDIA Tesla compute cards.

3.2 DATA SETS

We prepared a synthetic dataset and two real datasets MNIST and CIFAR-10 to estimate degrees of freedom.

Synthetic We build a synthetic dataset from a randomly generated network with 30 input nodes, 2 hidden layers with 30 hidden nodes in each, and 4 output nodes. We generated $n = 5000$ random zero-mean unit variance inputs with 30 dimensions. Each layer was fully connected to the previous layer, and we generated weights $w \sim \mathcal{N}(0, 5)$. We used sigmoid activation function for each layer and a soft-max on top of the network. The output sample labels y are then sampled according to the probabilities from the soft-max layer. To get the optimism, we also generated another 5000 samples for test.

MNIST [1] [LeCun et al., 1998] is a benchmark dataset that contains handwritten digit images. Each sample is a 28×28 image from 10 classes. We used 50000 samples for training.

CIFAR-10 [2] [Krizhevsky and Hinton, 2009] is a dataset contains 32×32 tiny color images from 10 classes. Each sample has 3072 features. We used 50000 samples for training.

3.3 DEGREES OF FREEDOM AND THE STRUCTURE OF THE NETWORK

To investigate the degrees of freedom for networks with different structures, we estimated the degrees of freedom for networks with width $[10, 20, \ldots, 100]$ and depth with 1,2,3 and 4, where all the hidden layers have equal widths. We used SdA to pre-train with 0.1 dropout rate and 0.1 corruption rate. We use weight decay penalty $1e - 5$ for both pre-training and fine-tuning. The estimated degrees of freedom is shown in Figure 3.

From the results, we found that networks with more width have more degrees of freedom. This is reasonable as increasing width leads to more independence between parameters. However, the degrees of freedom in deep networks is generally much less than the number of parameters it used. We see that the ratio of the parameters to degrees of freedom is on the order of 10^2. Loosely, one degree of freedom is acquired for 100 parameters. Among the models with the same number of parameters, deeper networks have

---

[1] http://yann.lecun.com/expdb/mnist/
[2] https://www.cs.toronto.edu/~kriz/cifar.html
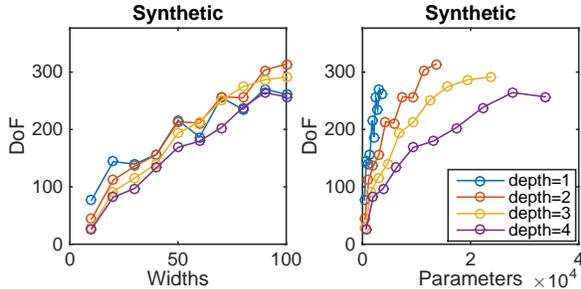
Figure 3: Degrees of freedom estimates for different models trained on synthetic data. Left: degrees of freedom vs network width. Right: degrees of freedom vs number of parameters in the network, which is linearly related to the network depth and quadratically related to the number of width. The line represent the degrees of freedom estimate from 1 Monte-Carlo run, and the color of each indicates the depth of the models.

less degrees of freedom. This observation indicates that the depth of the network has regularization on the complexity.

To further validate our assumption that deeper networks have less degrees of freedom, we also estimated degrees of freedom on MNIST and CIFAR-10 dataset. We tested networks with width $[100, 300, 500, 700]$, all other settings are the same as in the above synthetic experiment. The results are shown in Figure 4.

We observe that we can make the same conclusions hold for MNIST and CIFAR-10 as we did for synthetic data. The only difference is increasing depth results in more degrees of freedom than models trained with synthetic data. We attribute this to the differences of input data size and complexity between the real datasets, MNIST and CIFAR-10, and the much simpler synthetic datasets.

3.4 DEGREES OF FREEDOM AND REGULARIZATION TECHNIQUES

When training a deep neural network, many practical methods can be used for regularization. We investigate how the different techniques affect the degrees of freedom in the model.

We train networks using the same settings as in Section 3.3. In this experiment, we separately trained networks with different settings of penalty rates: corruption rate, dropout rate, and weight decay rate. We changed one rate at a time while keeping rest fixed.

tested the corruption rate, dropout rate, and weight decay penalty by keeping all others fixed and only changing one at a time.

For all three datasets, we trained network using corruption rate and dropout rate from $[0, 0.1, 0.2, \ldots, 0.9]$, and
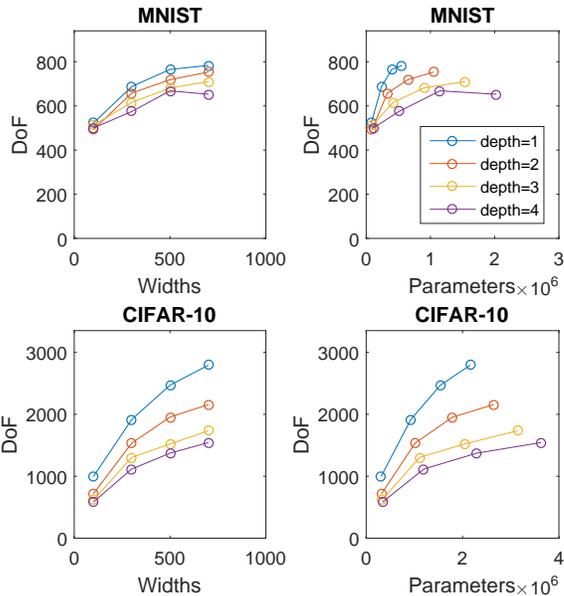
Figure 4: Degrees of freedom estimates for different models trained on MNIST and CIFAR-10. Left: degrees of freedom vs network width. Right: degrees of freedom vs the number of parameters in the network, which is linearly related to the network depth and quadratically related to the number of widths. The lines represent the degrees of freedom estimate from single Monte-Carlo sample and the color of each indicates the depth of that model.

weight decay rate from 10^{-6} to 10^{-3}. For each setting of regularization parameters, we trained a 3 layer network $[30, 30, 30]$ for synthetic data and $[300, 300, 300]$ on MNIST and CIFAR-10 data. We used one Monte Carlo sample to estimate degrees of freedom in each model. The result is shown in Figure 5.

We found that neither corruption rate nor dropout rate affected degrees of freedom drastically for synthetic data. This is because the input of the synthetic data is generated randomly. Hence, pre-training cannot learn higher level features for synthetic data. For MNIST and CIFAR-10, we found that both corruption rate and dropout rate have an impact on degrees of freedom . In CIFAR-10, the regularization effect is much larger. These results suggest that the regularization strength from dropout and corruption can be data-specific.

Weight decay penalty has a very strong effect on the degrees of freedom for all three datasets. Further, the weight decay exhibited a highly non-linear impact on the degrees of freedom, in dramatic contrast to its effect in ridge regression.[3]

---

[3]Ridge regression degrees of freedom scale with $\frac{1}{1+\lambda}$ which is non-linear but much tamer multiplier than in neural networks
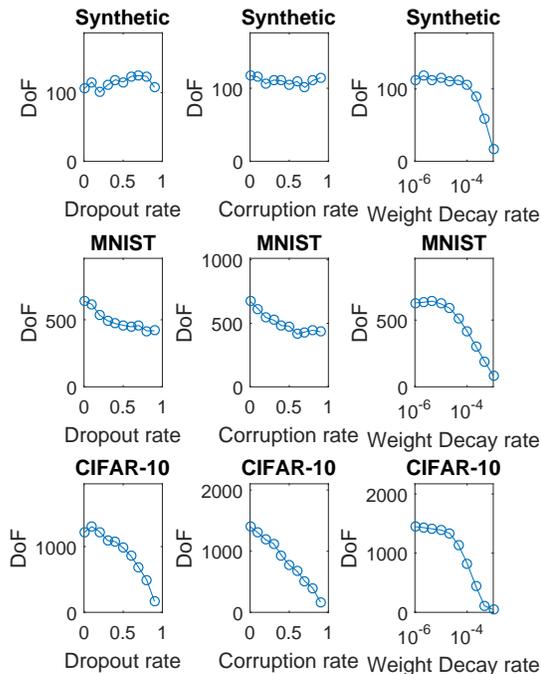
Figure 5: Degrees of freedom estimates for models trained on Synthetic data, MNIST and CIFAR-10 under different regularizations. The lines represent the degrees of freedom estimate.

Figure 6: Comparison between DoFAIC (first row) / Naïve AIC (second row) and 5-fold cross validation. Each circle in the plot represents a model with a specific structure. The x-axis is the mean cross-validation log deviance error across 5 folds.

3.5 MODEL SELECTION USING DEGREES OF FREEDOM

To validate that DoFAIC is a useful criterion for model selection, we compare it against model selection based on error estimates using cross validation. For brevity, we refer to the cross validation estimate of error as cross validation error. We performed a 5-fold cross-validation experiment for Synthetic, MNIST and CIFAR data on models with different network structures learned in Section 3.3. We calculated DoFAICs for all the models we trained using Equation (6) with the estimated degrees of freedom. We also calculated Naïve AIC using Equation (5) with the number of parameters in the network. We compared these estimates against cross-validation errors. The result is shown in Figure 6.

Further, we calculate the Spearman rank correlation between cross-validation log deviance errors and DoFAIC/Naïve AIC estimates for each dataset. The result is shown in Table 2.

We find that DoFAIC is very consistent with cross-validation error. Naïve AIC, on the other hand, exhibits negative correlation with cross validation error due to highly non-linear behavior. This is because Naïve AIC

Table 2: Spearman Rank Correlation between Cross-validation error and DoFAIC/Naïve AIC

Dataset	DoFAIC ρ	Naïve AIC ρ
Synthetic	0.9865	-0.6711
MNIST	0.9853	-0.9471
CIFAR-10	0.9941	-0.7824

overestimates the complexity of the model by using the large number of parameters in the network. The actual complexity in deeper and larger networks are much less than the number of parameters.

For all three datasets, both DoFAIC and cross-validation chose the same model. This indicates that DoFAIC can be used for model selection. We note that k-fold cross-validation, which needs at most k rounds of training, while DoFAIC only requires at most 2 rounds of training. This makes DoFAIC an efficient model selection criterion.

4 DISCUSSION

In this paper, we investigated the degrees of freedom for classification models and presented an efficient method to estimate their degrees of freedom. We showed that for simple classification models, degrees of freedom is equal to the number of parameters in the model. In deep networks, the

degrees of freedom is generally much less than the number of parameters in the model, and deeper networks tend to have less degrees of freedom. We also theoretically and empirically showed we can use DoFAIC as an efficient criterion for model selection, which has comparable performance to cross-validation.

Future work It would be interesting to investigate degrees of freedom in other deep architectures, such as Convolution Neural Network (CNN), Recurrent Neural Networks (RNN), denoising auto-encoders and contractive auto-encoders.

Acknowledgement

References

Hirotugu Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19 (6):716–723, 1974.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

Bradley Efron. Defining the curvature of a statistical problem (with applications to second order efficiency). *The Annals of Statistics*, pages 1189–1242, 1975.

Bradley Efron. The estimation of prediction error. *Journal of the American Statistical Association*, 99(467), 2004.

Yonina C Eldar. Generalized SURE for exponential families: Applications to regularization. *Signal Processing, IEEE Transactions on*, 57(2):471–481, 2009.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11: 625–660, 2010.

Isabelle Guyon, Asa Ben Hur, Steve Gunn, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems 17*, pages 545–552. MIT Press, 2004.

Geoffrey E. Hinton and Richard S. Zemel. Autoencoders, minimum description length and helmholtz free energy.

In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 3–10. Morgan-Kaufmann, 1994.

Lucas Janson, William Fithian, and Trevor J Hastie. Effective degrees of freedom: a flawed metaphor. *Biometrika*, page asv019, 2015.

Andrei N Kolmogorov. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7, 1965.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits, 1998.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996. ISBN 0387947248.

Sathish Ramani, Thierry Blu, and Michael Unser. Monte-carlo sure: A black-box optimization of regularization parameters for general denoising algorithms. *Image Processing, IEEE Transactions on*, 17(9):1540–1554, 2008.

Jürgen Schmidhuber. Discovering neural nets with low kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151, 1981.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, 67(1):91–108, 2005. URL https://ideas.repec.org/a/bla/jorssb/v67y2005i1p91-108.html.

Samuel Vaiter, Charles Deledalle, Gabriel Peyré, Jalal M. Fadili, and Charles Dossal. The Degrees of Freedom of the Group Lasso. In *International Conference on Machine Learning Workshop (ICML)*, Edinburgh, United Kingdom, 2012. URL https://hal.archives-ouvertes.fr/hal-00695292.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The*

Journal of Machine Learning Research, 11:3371–3408, 2010.

Jianming Ye. On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.

Hui Zou, Trevor Hastie, Robert Tibshirani, et al. On the "degrees of freedom" of the lasso. *The Annals of Statistics*, 35(5):2173–2192, 2007.