Non-Convex Optimization with Spectral Radius Regularization

Adam Sandler[1], Diego Klabjan[2], and Yuan Luo[3]

[1]Engineering Sciences and Applied Mathematics, Northwestern University
[2]Industrial Engineering and Management Sciences, Northwestern University
[3]Preventive Medicine (Health and Biomedical Informatics), Northwestern University

Abstract

We develop regularization methods to find flat minima while training deep neural networks. These minima generalize better than sharp minima, yielding models outperforming baselines on real-world test data (which may be distributed differently than the training data). Specifically, we propose a method of regularized optimization to reduce the spectral radius of the Hessian of the loss function. We also derive algorithms to efficiently optimize neural network models and prove that these algorithms almost surely converge. Furthermore, we demonstrate that our algorithm works effectively on applications in different domains, including healthcare. To show that our models generalize well, we introduced various methods for testing generalizability and found that our models outperform comparable baseline models on these tests.

1 Introduction

Finding flat minima solutions to optimization problems is important, especially in machine learning applications. Such models generalize better than sharp minima because the value of the loss function remains similar around flat minima if the data is shifted, distorted, or otherwise changed. Thus, in practice, optimal machine learning models near flatter optima should perform better than those near sharper minima on test data distributed differently than the original training data [Keskar *et al.*, 2017].

Here, we define flat minima as those with a small spectral radius of the Hessian of the loss function (i.e., the largest absolute eigenvalue is small) and sharp minima as those where the spectral radius is large. For flat minima, there is no direction away from the minimum in which the loss function immediately and rapidly increases or decreases. Therefore, by regularizing the optimization of models with respect to this spectral radius, we can obtain solutions that are less susceptible to errors and biases in training or test data.

However, this regularization presents certain challenges. For large neural networks, computing and storing the Hessian and the third derivative tensor (used in the gradient of the spectral radius term) are intractable; therefore, we develop methods to efficiently compute the regularization term and its gradient without computing these full quantities. We also design methods to introduce errors and biases into the data to test the generalizability of these models.

To tackle these challenges, we build methods to regularize the spectral radius while computing Hessian-vector products, rather than computing the full Hessian and then multiplying by the vector. We approximate the spectral radius and corresponding eigenvector using algorithms such as power iteration and Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG). We extend methods used for computing Hessian-vector products for neural networks and use them to efficiently compute the eigenvector and spectral radius gradient, used in our algorithms. Implementing these methods within a batch stochastic gradient descent algorithm allows us to optimize a neural network with a given loss function and our regularization term.

We also present results with different regularization parameters to show that our methodology is stable. Our contributions are as follows.

- We develop algorithms for regularizing neural networks with respect to the spectral radius of the Hessian, a novel use of a derivative measure for such regularization.
- We derive differential operators for efficient computation of Hessian-vector products for neural networks.
- We provide formal proofs of convergence and other properties of our algorithm.
- We present experimental results on multiple real-world data sets across different domains, designing specific methods to test generalizability.

In Section 2, we review existing literature related to our research. In Section 3, we derive the algorithm used for our regularization. In Section 4, we discuss convergence results and other properties of the algorithm. In

Section 5, we describe different generalizability tests and present the results of our experiments with regularization on various data sets.

2 Related Work

Existing research discussed how different learning methods affect the ability of neural networks to converge to flat minima. Keskar *et al.* [2017] observed that large-batch stochastic gradient descent (SGD) and its variants, such as adaptive moment estimation (Adam), tend to converge to sharp minima. In contrast, small-batch methods converge to flat minima. This implies that small-batch methods generalize better than large-batch methods, as the training function at sharp minima is more sensitive. Some possible causes include large-batch methods over-fitting, being attracted to saddle points, and lacking the exploratory properties of small-batch methods (i.e., they tend to converge to the minima close to the initial weights). Yao *et al.* [2018] showed that large-batch training of neural networks converges to points with a larger Hessian spectrum (both in terms of dominant and other eigenvalues), showing poor robustness. Jastrzebski *et al.* [2018]; Zhang *et al.* [2024] extended these claims by showing that a large learning rate also leads to flatter minima that generalize better than sharper minima. Baldassi *et al.* [2021] showed that wide flat minima in nonconvex neural networks arise as structures from groups of minima around locally robust configurations. Wu *et al.* [2022] showed that SGD favors flat minima but left the connection between the Hessian and generalization as an open question for future work.

Others used different ways to measure and find flat minima, including loss functions and optimization algorithms. Ma *et al.* [2020] suggested that Kronecker-Factored Approximate Curvature (K-FAC) [Martens and Grosse, 2015], an approximate second-order method, may yield generalization improvements over first-order SGD. Chaudhari *et al.* [2017]; Dziugaite and Roy [2018]; Pittorino *et al.* [2021] proposed an entropy-based loss function to find solutions in flat regions and an algorithm (called entropy-SGD) to optimize models. He *et al.* [2019] observed that at local minima of deep networks, there exist many asymmetric directions where the loss sharply increases, which they call "asymmetric valleys." They proposed stochastic weight averaging (SWA) along the SGD trajectory to bias solutions towards the flat side. Chaudhari *et al.* [2017] also noted that many neural networks, trained on various data sets using SGD or Adam, converge to a point with a large number of near-zero eigenvalues, along with a long positive tail and shorter negative tail. Our regularization method, which attempts to reduce the spectral radius of the Hessian, is tailored to avoid the eigenspectrum asymmetries described by Chaudhari *et al.* [2017] and He *et al.* [2019].

Foret *et al.* [2021] developed a Sharpness-Aware Minimization (SAM) algorithm, which minimizes the maximum loss within a neighborhood of a point. Unlike us, they focused on testing model generalization on fuzzy labels. Andriushchenko and Flammarion [2022] have raised doubts about SAM's ability to generalize in other settings. Adding SAM as a baseline comparison would require training on multiple seeds due to the stochasticity. Additionally, the SAM paper was published after the original draft of this paper was posted to ArXiv.

While Yoshida and Miyato [2017] developed a spectral norm radius regularization method, it looks solely at the spectral radius of a neural network's weight matrices rather than the spectral radius of the Hessian of the loss function. Though they experimentally showed that their regularization method has a small generalization gap (between the training and test set), their method also had a higher Hessian spectral radius than vanilla SGD, weight-decay, and adversarial methods. We believe our regularization method and generalization tests more directly address finding flat minima and measuring their generalizability.

Kaddour *et al.* [2022] compared SWA and SAM in various computer vision, natural language processing, and graph representation learning tasks. They concluded that the effectiveness of these methods is influenced by the dataset and model architecture. Flat-minima optimizers can offer asymmetric payoffs, potentially leading to slight performance decreases at worst, but significant gains at best.

Variable	Definition
w	model parameters or weights
$f(w)$	loss function
$H(w)$	Hessian of $f(w)$
$\rho(w)$	spectral radius of $H(w)$
μ	degree of regularization
K	goal of $\rho(w) < K$
$\bar{v}$	eigenvector corresponding to spectral radius

Table 1: Variable Definitions

3 Algorithm

We summarize the main variables used and their corresponding definitions in Table 1. We choose to express our problem as a regularized optimization problem rather than a constrained optimization or min-max problem, as strict adherence to our spectral radius constraint is typically unnecessary. Additionally, the regularized approach keeps the algorithm simple, while complexity is computationally taxing for large neural networks. Thus, our optimization problem is

$$\min_w f(w) + \mu \max\{0, \rho(w) - K\},$$

for weights $w \in \mathbb{R}^n$, non-convex loss function $f(w)$, spectral radius (i.e., the maximal absolute eigenvalue) $\rho(w)$ of the Hessian $H(w)$ of $f(w)$, and regularization parameters μ and K. This can also be viewed as Lagrangian relaxation of constraint $\rho(w) \leq K$. For convenience, we denote

$$g(w) := f(w) + \mu \max\{0, \rho(w) - K\}.$$

Our goal is to design efficient algorithms for solving this minimization problem, with the caveat that we cannot directly compute $H(w)$. For large neural networks of size $\mathcal{O}(n)$, computing and storing objects of size $\mathcal{O}(n^2)$ (such as the Hessian) is intractable. However, we can efficiently compute the Hessian-vector product $H(w)v$ for a given $v \in \mathbb{R}^n$ using a method discussed in Section 3.2.

In Section 3.1, we present and explain different variants of our algorithm. In Section 3.2, we discuss how to compute the regularized term and its gradient.

3.1 Algorithms

Here, we present two versions of our algorithm: a batch stochastic gradient descent power iteration algorithm (Algorithm 1) and a LOBPCG algorithm (Algorithm 2). The LOBPCG method tries to improve the run time of power iteration by using a preconditioner (a transformation used to improve numerical methods). For simplicity, we hide the w_k dependencies (where w_k is the value of weights w at iteration k) for many of the variables by defining: $f_k := f(w_k)$, $g_k := g(w_k)$, $\rho_k := \rho(w_k)$, $\nabla f_k := \nabla f(w_k)$, etc. We let the step size α_k be a predefined function of iteration k and L be the maximum number of iterations. We assume $f(w) = \sum_i \bar{f}^{(i)}(w)$ and write $\bar{f}_k^{(i)} := \bar{f}^{(i)}(w_k)$ as the value of the loss function f on sample i at iteration k. We also let $\bar{H}_k^{(i)}$ be the Hessian matrix of $\bar{f}^{(i)}$ at w_k.

Algorithm 1: Batch Stochastic Gradient Descent

1 Initialize w_1
2 **for** $k = 1, \cdots, L$ **do**
3 Select batch B_k of cardinality $\mathcal{U}$ uniformly at random
4 Compute $\nabla f_k = \frac{1}{\mathcal{U}} \sum_{i \in B_k} \nabla \bar{f}_k^{(i)}$
5 Initialize u, λ, and v
6 **while** $||u - \lambda v|| > \varepsilon_k$ **do**
7 $u = \frac{1}{\mathcal{U}} \sum_{i \in B_k} \bar{H}_k^{(i)} v$ (using $\mathcal{R}\{\cdot\}$)
8 $\lambda = u^T v$
9 $v = \frac{u}{||u||}$
10 $\rho_k = \lambda,\ v_k = v$
11 $\nabla \rho_k = \frac{1}{\mathcal{U}} \sum_{i \in B_k} v_k^T \nabla \bar{H}_k^{(i)} v_k$ (using $\mathcal{R}^2\{\cdot\}$)
12 Update $p_k = \nabla f_k + \mu \nabla \rho_k \mathbb{1}(\rho_k > K)$
13 $w_{k+1} = w_k - \alpha_k p_k$

We start with Algorithm 1, a batch stochastic gradient descent algorithm, which uses power iteration to compute ρ_k and $\nabla \rho_k$. Due to the implementation of $\mathcal{R}\{\cdot\}$ and $\mathcal{R}^2\{\cdot\}$ (see Section 3.2) during these computations, the storage requirements of $\mathcal{O}(n)$ are not onerous. Also, since the Hessian is symmetric, power iteration converges at a rate proportional to the square of the ratio between the two largest eigenvalues $\mathcal{O}\left(|\lambda_1/\lambda_2|^2\right)$, rather than the typical linear rate $\mathcal{O}\left(|\lambda_1/\lambda_2|\right)$. Note that the gradient computation in Line 4 can be done as part of the $\mathcal{R}\{\cdot\}$ in Lines 7 or 11.

Algorithm 2: LOBPCG Method

1 Initialize w_1
2 **for** $k = 1, \cdots, L$ **do**
3 Select batch B_k of cardinality $\mathcal{U}$ uniformly at random
4 **if** $k \mod b = 0$ **then**
5 Update K-FAC matrix T
6 Compute $\nabla f_k = \frac{1}{\mathcal{U}} \sum_{i \in B_k} \bar{f}_k^{(i)}$
7 Initialize r and v
8 **while** $||r|| > \varepsilon_k$ **do**
9 $u = \frac{1}{\mathcal{U}} \sum_{i \in B_k} \bar{H}_k^{(i)} v$ (using $\mathcal{R}\{\cdot\}$)
10 $\lambda = u^T v$
11 $r = u - \lambda v$
12 $w = v + \tilde{\alpha} T r$
13 $v = \frac{w}{||w||}$
14 $\rho_k = \lambda,\ v_k = v$
15 $\nabla \rho_k = \frac{1}{\mathcal{U}} \sum_{i \in B_k} v_k^T \nabla \bar{H}_k^{(i)} v_k$ (using $\mathcal{R}^2\{\cdot\}$)
16 Update $p_k = \nabla f_k + \mu \nabla \rho_k \mathbb{1}(\rho_k > K)$
17 $w_{k+1} = w_k - \alpha_k p_k$

To improve the run time and convergence of our power iteration method, we developed a LOBPCG method (Algorithm 2). This method uses a step-size $\tilde{\alpha}$ (not necessarily fixed), preconditioner T (for example, K-FAC), and update frequency b. The LOBPCG algorithm may converge faster than the power iteration algorithm with good choices for these parameters. Knyazev [2001] and Knyazev *et al.* [2007] assumed and numerically showed that T must be symmetric positive definite, with an efficient preconditioner being an approximation of H_k^{-1} (as the condition number $\kappa(TH_k)$ is low). We chose to use Martens and Grosse's [2015] K-FAC as the preconditioner, as it satisfies these conditions and is well-suited for neural networks.

3.2 Gradients of Regularization Term

The spectral radius can be expressed as $\rho(w) = \bar{v}^T H(w) \bar{v}$, where $\bar{v}$ is the eigenvector corresponding to the maximum absolute eigenvalue. To compute gradient update steps for the regularization term, we calculate $\nabla \rho$ using Lemma 3.1 from Van der Aa *et al.* [2007].

Lemma 3.1. *For distinct eigenvalues of a symmetric matrix $A(x) : \mathbb{R} \to \mathbb{R}^{n \times n}$,*

$$\frac{d\lambda_i(x)}{dx} = \bar{v}_i^T \frac{dA(x)}{dx} \bar{v}_i,$$

where $\bar{v}_i$ is the eigenvector for eigenvalue λ_i.

The expression for this derivative is more complicated with repeating eigenvalues, so we assume that the eigenvalue in question is distinct (in practice, this is usually the case).

Using this result and assumption, we express $\nabla\rho(w) = \bar{v}^T \nabla H(w) \bar{v}$. Thus, by efficiently computing $H(w)v$ and $v^T \nabla H(w) v$ for $w, v \in \mathbb{R}^n$, we can calculate $\rho(w)$ and $\nabla\rho(w)$, respectively.

Hessian-Vector Operations

In order to compute $H(w)v$ and $v^T \nabla H(w) v$ for large neural networks with $w, v \in \mathbb{R}^n$, we extend Pearlmutter's [1994] operator $v \to \mathcal{R}_v \{f; w\}$, defined as

$$\mathcal{R}_v \{f; w\} := \frac{\partial}{\partial r} f(w + rv)\bigg|_{r=0}.$$

Note that $\mathcal{R}_v \{\nabla f; w\} = H(w)v$. Thus, by applying the differential operator $\mathcal{R}_v \{\cdot\}$ to the forward and backward passes used to calculate the gradient, we can compute $\rho(w)$ efficiently.

We extend this operation to

$$\mathcal{R}_v^2 \{f; w\} := \mathcal{R}_v \{\mathcal{R}_\cdot \{f; w\}; w\}$$

by applying the differential operator $\mathcal{R}_v \{\cdot\}$ again to the forward and backwards passes. Particularly, we compute $\mathcal{R}_v^2 \{x\}$ and $\mathcal{R}_v^2 \{y\}$ during the forward pass and $\mathcal{R}_v^2 \{\nabla_y f\}$, $\mathcal{R}_v^2 \{\nabla_x f\}$, and $\mathcal{R}_v^2 \{\nabla_w f\}$ during the backward pass, where ∇_y, ∇_x, and ∇_w are the gradients with respect to output y, input x, and weights w. We derive our formulas in Appendix A. Since $\mathcal{R}_v^2 \{\nabla f; w\} = v^T \nabla H(w) v$, this allows us to efficiently compute $\nabla\rho(w)$.

These methods keep the number of stored values $\mathcal{O}(n)$, while directly computing the Hessian and third derivative tensor would require $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$ storage (which is intractable for large networks).

4 Algorithm Convergence Analysis

Here, we show that Algorithms 1 and 2 almost surely converge to a critical point, with some assumptions. While we outline our proofs here, the details are in Appendix B.

We assume that batches B are randomly selected. Note that $p_k = p_k(w_k)$. We made the following assumptions.

A1 $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^5$, $g(w)$ is bounded from below (without loss of generality, $g(w) \geq 0$).

A2 Conditions on the learning rate and tolerance:

$$\sum_{k=1}^{\infty} \alpha_k^2 < \infty, \sum_{k=1}^{\infty} \alpha_k = \infty, \sum_{k=1}^{\infty} \varepsilon_k \alpha_k < \infty.$$

A3 The moments do not grow too quickly:

$$\left|\left| \frac{1}{\mathcal{U}} \sum_{i \in B} \nabla \bar{f}^{(i)}(w) \right|\right|^j \leq A_j^{(1)} + B_j^{(1)} ||w||^j,$$

$$\left|\left| \frac{1}{\mathcal{U}} \sum_{i \in B} v^T \bar{H}^{(i)}(w) v \right|\right|^j \leq A_j^{(2)} + B_j^{(2)} ||w||^j,$$

for $j = 2, 3, 4$ and all k; positive constants $A_j^{(1)}$, $A_j^{(2)}$, $B_j^{(1)}$, $B_j^{(2)}$, any weights w and unit vector v, and any subset B of cardinality $\mathcal{U}$.

A4 The Hessian $H(w)$ is Lipschitz continuous.

A5 We have $\varepsilon_k \leq 1$ and $\varepsilon_k \to 0$ as $k \to \infty$.

A6 Outside a certain horizon, the gradient points toward the origin. There exists $D < \infty$ such that

$$\inf_{||w||^2 \geq D,\ v} w^T \sum_{i \in B} \left[\nabla \bar{f}^{(i)}(w) + \mu v^T \nabla \bar{H}^{(i)}(w) v \right] > 0,$$

for any subset B of cardinality $\mathcal{U}$. There are well-known tricks to ensure this assumption, such as adding a small linear term [Bottou, 1998].

First, the stopping criteria for computing the eigenvector is met.

Lemma 4.1. *Power Iteration (Steps 7-10 in Algorithm 1) and LOBPCG (Steps 9-14 in Algorithm 2) always finish in a finite number of iterations with $||v_k - \bar{v}_k|| \leq \varepsilon_k$, where $\bar{v}_k$ is an eigenvector corresponding to the leading eigenvalue of $H_k = \frac{1}{|B|} \sum_{i \in B} \bar{H}_k^{(i)}$.*

This follows from the proofs of power iteration convergence by Parlett and Poole [1973] and LOBPCG convergence by Knyazev [2001].

Lemma 4.2. *Given Assumptions A1-A6, $\lim_{k \to \infty} v_k^T \nabla H_k v_k = \lim_{k \to \infty} \nabla \bar{\rho}_k$, where $\nabla \bar{\rho}_k$ is the true gradient of the Hessian's spectral radius.*

We split $v_k^T \nabla H_k v_k$ into components for the true eigenvector $\bar{v}_k$ and our estimate v_k. Then, we bind it, showing that Assumption A5 is sufficient for Lemma 4.2 to hold.

Then, we show that these stochastic algorithms fit our bounds on the moments of the update term. Here, we take the expectation with respect to the choice of batch B_k, conditioned on the history

$$\mathcal{P}_k := B_1, \cdots, B_{k-1},\ w_1, \cdots, w_k,\ \alpha_0, \cdots, \alpha_k.$$

Lemma 4.3. *Given Assumptions A3-A4,*

$$\mathbb{E}_{B_k} \left[||p_k||^j \middle| \mathcal{P}_k \right] \leq A_j + B_j ||w_k||^j,$$

for $j = 2, 3, 4$, positive constants A_j and B_j, and any k.

We split p_k into its components $p_k := \nabla f_k + \mu \nabla \rho_k$. We use the Assumptions to bind each of these components. Then, we combine the results to show that the lemma holds.

Next, we show that the iterates are confined.

Lemma 4.4. *Given Assumptions A1-A6, the iterates w_k in Algorithms 1 and 2 are bounded almost surely.*

We define a sequence that is a function of w_k and show that the sum of its positive expectations is finite. Then, we apply the Quasi-Martingale Convergence Theorem and show that since the sequence converges almost surely, the norm of our weights w_k is bounded. Next, using our assumptions and Lemma 4.4, we prove almost sure convergence.

Theorem 4.1. *Given Assumptions A1-A6, in Algorithms 1 and 2 the loss function values $g(w_k)$ converge almost surely and $\nabla g(w_k)$ converge almost surely to 0.*

We use confinement of w_k to show that positive expected variations in $g(w)$ between iterates are bounded by a constant times our learning rate squared α_k^2. Using Assumption A2 and the Quasi-Martingale Convergence Theorem, we show that g_k converges almost surely. Then, we show that ∇g_k almost surely converges to zero. Our proofs of Lemma 4.4 and Theorem 4.1 are based on Bottou's [1998] proof that SGD almost surely converges.

5 Experiments

We tested our spectral radius regularization algorithms on the following data sets: forest cover types [Blackard and Dean, 1999], United States Postal Service (USPS) handwritten digits [LeCun *et al.*, 1990], and chest X-rays [Wang *et al.*, 2017]. The forest cover-type data uses cartographic data to predict which of seven tree species is planted on a plot of land. The USPS digits data includes images of digits 0-9 from scanned envelopes. The chest X-ray data uses images to identify which of the fourteen lung diseases patients were diagnosed with. We further describe these data sets in Appendix C.1.

Additionally, we trained unregularized, He *et al.*'s [2019] asymmetric valley, Chaudhari *et al.*'s [2017] entropy-SGD, and Martens and Grosse's [2015] K-FAC models, which serve as baseline comparisons. These other methods for finding flat minima were discussed in Section 2 and serve as baseline comparisons.

5.1 Setup

To test if models with lower spectral radii generalize better than those with higher spectral radii, we created test sets that are differently distributed from the training data. To accomplish this, we employed covariate shifts and image augmentation techniques and introduced new, distinct data. We provide a more detailed description of our software, parameter values, and architectures in Appendices C.2 and C.3.

For the forest cover-type data, we weighted the test plots of land to shift the mean of the features. Then, we compared the accuracy of the trained models and repeated them for one thousand shifts. These perturbations simulate test conditions with poor measurements or climate changes. This weighting method is opposite to Shimodaira [2000]; Huang *et al.* [2006]; we made the test and training data have different, rather than similar, distributions.

For USPS digits, we augmented the test set using random crops and rotations, a subset of the perturbations used by Hendrycks and Dietterich [2019] to benchmark robustness on ImageNet. These modifications simulate test conditions where digits are written on angles, cut off, or poorly scanned. We also compared how models trained on USPS data performed on MNIST [LeCun and Cortes, 2010] and images from Conditional Generative Adversarial Networks (GANs) [Mirza and Osindero, 2014]. Zhang *et al.* [2022] used performance on GAN-generated data to predict generalizability.

Model	ρ	Test Acc.	Relative Shift Acc. Mean	Relative Shift Acc. 95% CI
Unregularized	36.58	**71.74**	-2.80%	[-3.38, -2.23]
Asym. Valley	23.28	70.99	-1.48%	[-1.96, -1.01]
Entropy-SGD	6.82	69.69	-1.41%	[-1.89, -0.92]
K-FAC	58.55	70.83	-2.29%	[-2.85, -1.72]
μ=.01, K=1	**1.68**	69.71	-1.61%	[-2.10, -1.13]
μ=.01, K=0	2.16	70.39	**-0.96%**	**[-1.31, -0.61]**
μ=.005, K=1	3.09	70.97	-1.50%	[-1.97, -1.02]
μ=.001, K=5	7.15	70.67	-1.80%	[-2.28, -1.32]
μ=.001, K=0	9.03	70.87	-1.96%	[-2.44, -1.48]
LOBPCG	1.99	69.49	-2.87%	[-3.45, -2.29]

Table 2: Comparison of forest cover-type models. Performance on the shifted test data is measured relative to each model's test accuracy (accuracy on shifted data divided by accuracy on test data minus 1). Optimal values are bolded.

For the chest X-ray models, we compared performance on two similar transfer learning data sets, CheXpert [Irvin *et al.*, 2019] and MIMIC-CXR [Johnson *et al.*, 2019] (using the six conditions common to all three data sets). We kept the labeled training and validation sets separate due to differences in how the conditions were recorded. As the new chest X-ray data contains different patients with conditions not present in the training data, it tests how well these models perform in different populations. Kim *et al.* [2019]; Salehinejad *et al.* [2021] stated that the use of data from multiple geographically and temporally distinct sources is important to demonstrate the generalizability of medical image models. Zech *et al.* [2018] showed that a CheXNet model [Rajpurkar *et al.*, 2017] trained to detect pneumonia generalized poorly to data from other hospital systems and times. Since this is a multi-class, multi-label problem, we measure performance using the mean area under the curve (AUC) of the receiver operating characteristic curve over each class.

We selected different regularization parameters μ and K via an informal grid search. If μ is too large, the model will converge to a flat outlying point, predicting the same class for each sample. If μ is too small, the regularization will be ineffective.

Details on hyperparameters, network architectures, hardware, and software are in Appendix C. Data sizes are also discussed in Appendix C.1.

5.2 Results

We trained a feed-forward neural network on forest cover-type data and compared the accuracy of models on the randomly shifted test sets. We selected different regularization parameters μ and K via an informal grid search. Table 2 shows a benefit to the asymmetric valley, entropy-SGD, and power iteration regularized models over the unregularized model. The K-FAC and LOBPCG models do not significantly outperform the unregularized model. While there are some differences between the various regularized models – there is some delineation

between those with lower ρ and higher – all generalize better than the unregularized model. The relative decrease in accuracy on the shift data is less on the regularized models than on the unregularized model. Also, our spectral radius measure ρ mostly follows the regulation strictness. Our strictest regularized model with $\mu = 0.01, K = 0$ and small $\rho = 2.16$ saw the lowest decrease in accuracy on the shifted data. The confidence intervals show that models that performed worse on the shifted data also had a higher variance in their results. We further discuss the LOBPCG results in Section 5.5.

We trained a convolutional neural network on USPS digits, using various regularization and optimization methods, and compared the accuracy on multiple test sets. Per Figure 1, while the models performed comparably on the test data (all models have an accuracy of 94.47-95.91%), our regularized models (both power iteration and LOBPCG) performed significantly better than the unregularized model on both augmented test data sets (87.10-91.08% vs. 86.20% on Augmented Test 1; 65.37-69.06% vs. 63.03% on Augmented Test 2). Our regularized model with $\mu = 0.005$ and $K = 0$ was the most accurate on the USPS and augmented test sets. The model with the lowest spectral radius (1.16, $\mu = 0.05$ and $K = 1$) performed second-best on the augmented data. The asymmetric valleys model outperformed the other baseline models but was still 2.2%-3.9% worse than the $\mu = 0.005$ and $K = 0$ model on the augmented data. Also, Figure 1 shows that there is a clear relationship between our regularization parameter μ and the spectral radius ρ for our regularized models: as μ increases, ρ decreases (provided it is greater than K). Figure 2 uses 70 models with different spectral radii to show that as ρ increases, the magnitude of the generalization gap (between the test and augmented test sets) increases. This implies that models with higher spectral radii tend to perform worse on this generalization task.

We also trained two GANs on USPS data using Linder-Norén's [2021] (GAN1) and Chhabra's [2021] (GAN2) methodology. We found that the images generated by GAN1 were too similar, causing the models to classify or misclassify them in the same way. For example, the $\mu = 0.005$ and $K = 0$ model misclassified 0's as 2's, 5's as 3's, and 9's as 6's. GAN2 did not suffer from this issue. The figure on the right of Figure 1 shows the results. The model that performed best on the augmented tests ($\mu = 0.005$ and $K = 0$) performed fourth-best on the MNIST (59.80%) and GAN2 data (88.15%) but performed poorly on GAN1 (70.44%) due to the aforementioned issues. The regularized models with $\rho \approx 1.51$ performed best on the GAN data; the regularized model with $\mu = 0.03$ and $K = 0$ was 99.82% accurate on GAN1, and the $\mu = 0.1$ and $K = 2$ model was 91.35% accurate on GAN2. Entropy-SGD performed best on the MNIST data (67.12%), but regularized models were the next four best-performing models. While examining the GAN results, we realized that the generated images were abnormally distributed relative to the USPS images. To determine if these results were coincidental or due to this distribution, we constructed two data sets, Const1 and Const2, from the augmented test data to mimic the abnormal image distribution in GAN1 (see Appendix D) and found that the $\mu = 0.005$ and $K = 0$ model performed best on the constructed data. Thus, we conclude that the GAN1 results appear to be a coincidence and recommend the $\mu = 0.005$ and $K = 0$ model.

Batch Size	ρ	Test Acc.	Relative Shift Acc.	
			Mean	95% CI
32	5.25	69.39	-2.88%	[-3.46, -2.29]
64	4.05	68.84	-3.10%	[-3.69, -2.52]
128	2.16	**70.39**	**-0.96%**	**[-1.31, -0.61]**
256	1.32	69.30	-1.52%	[-2.01, -1.04]
512	**1.24**	69.16	-1.49%	[-1.98, -1.00]

Table 3: Effect of batch size on accuracy of forest cover-type models with $\mu = 0.01$ and $K = 0$.

Batch Size	ρ
32	4.82
64	3.31
128	2.16
256	1.39
512	0.93

Table 4: Computed spectral radius of forest cover-type model with $\mu = 0.01$ and $K = 0$ trained with batch size of 128.

For chest X-ray comparisons, we trained CheXNet (a 121-layer DenseNet trained on chest X-ray data, based on github.com/zoogzog/chexnet) as our baseline. Using this model as an initialization, we trained for an additional epoch with our spectral radius regularization method, comparing the mean AUC. Similarly, we used this initialization to train the entropy-SGD, K-FAC, and asymmetric valley models. For our regularized model, we employed gradient clipping to curtail an exploding spectral radius gradient. Figure 3 shows that the two models with the lowest spectral radius ρ, our regularized model ($\mu = 10^{-4}$ and $\alpha = 10^{-6}$) and entropy-SGD, performed best on the transfer learning chest X-ray data sets. Our model had a 5.34% lower ρ than entropy-SGD and a lower performance drop on 3-of-the-4 transfer learning data sets. In Appendix E, we use Grad-CAM to highlight the regions of the X-rays used to make predictions. We show that the two models with the lowest spectral radius overlap the most in explanations, signifying that their explanations generalize better too.

5.3 Batch Size

As discussed in Section 2, Keskar *et al.* [2017]; Yao *et al.* [2018]; Jastrzebski *et al.* [2018] showed that large-batch training methods yield more generalizable models. This motivated us to analyze the effect of batch size on the spectral radius and generalizability of our regularized forest cover type and USPS models. Contrary to their findings, we found (Tables 3 and 5) that smaller batch models have a larger spectral radius. Despite this, the

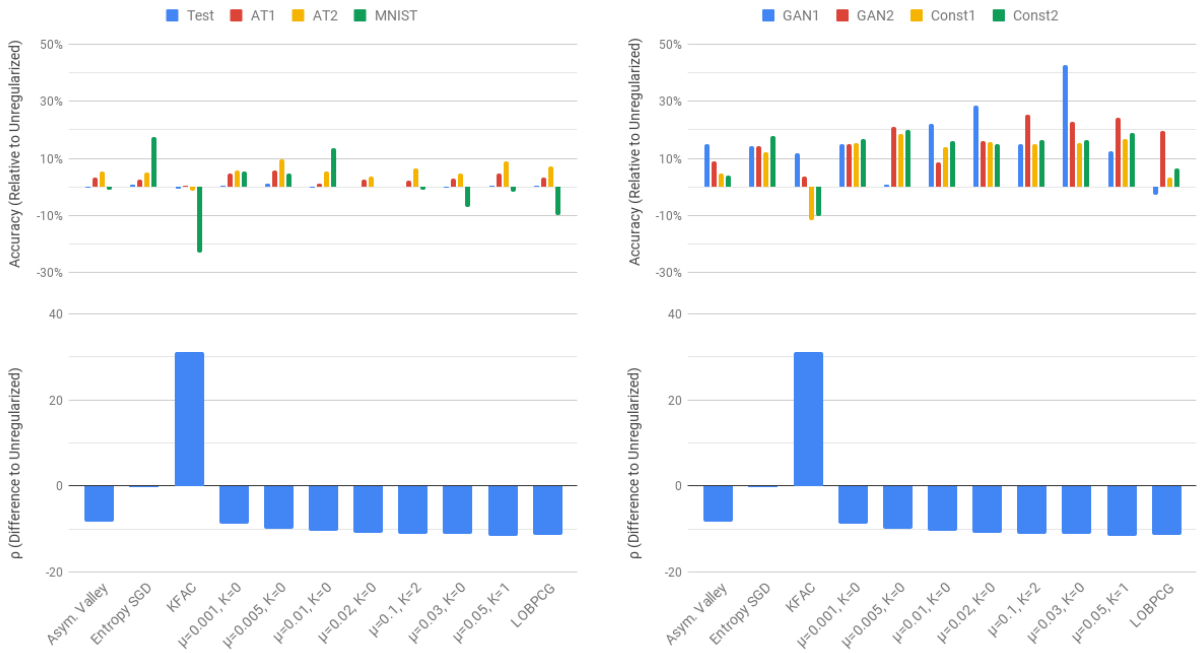


Figure 1: Accuracy of models trained on USPS data. Augmented Test (AT) 1 uses random crops of up to one pixel and random rotations of up to 15°. AT 2 uses crops of up to two pixels and rotations of up to 30°.

Batch Size	ρ	Accuracy							
		Test	AT 1	AT 2	MNIST	GAN1	GAN2	Const1	Const2
32	7.88	93.27	85.00	61.58	55.21	**90.07**	86.18.	70.47	71.67
64	4.93	94.47	87.64	65.58	59.30	80.77	86.52	74.24	73.54
128	2.69	**95.91**	**91.08**	**69.06**	**59.80**	70.44	88.15	**78.20**	**78.54**
256	2.14	94.42	88.59	67.76	57.99	73.28	82.31	75.87	76.04
512	**2.08**	94.97	86.85	64.62	50.79	73.16	**89.63**	72.27	73.12

Table 5: Effect of batch size on accuracy of USPS models with $\mu = 0.005$ and $K = 0$.

model with the original batch size (128) generally performed best on our comparison tests. However, Table 4 shows that the batch size is a major contributing factor to the computed spectral radius.

5.4 Computational Time Breakdown

Figure 4 shows that the computational time of the power iteration regularization method was relatively high but not prohibitively so. The unregularized models were the fastest to train, followed by the asymmetric valley models. The power iteration model took the longest to train on forest cover-type data and the second longest to train on USPS. LOBPCG significantly improved the training time of the forest cover-type model to the point where it was the third-fastest model. However, it only decreased the training time of the USPS model by 3%. K-FAC took the longest to train on USPS data and the third longest on forest cover type. Entropy-SGD was the second longest on forest cover-type data and the third fastest on USPS.

Figure 5 shows that three-quarters of Algorithm 1's run time is spent on power iteration (lines 5-10). About another 15% is spent on computing $\nabla \rho_k$ using $\mathcal{R}^2\{\cdot\}$ (line 11). 5-10% was spent on computing the results and other tracked statistics.

5.5 Spectral Radius Computation

As shown in Section 5.4, the LOBPCG can improve computational training time. These models also had the second-lowest spectral radius (see Section 5.2). However, given the following drawbacks, we cannot universally recommend using this method. It requires additional parameter tuning of the update frequencies b and step sizes $\tilde{\alpha}$. We could not reasonably train a LOBPCG model on the chest X-ray data since the additional memory constraints required a reduced batch size, making the run time onerous. Furthermore, we found that the residual norm ($||Hv - \rho v||$) was significantly higher for the LOBPCG method than for the power iteration method. Thus, our experiments indicate that the power iteration method generally outperformed the LOBPCG method.

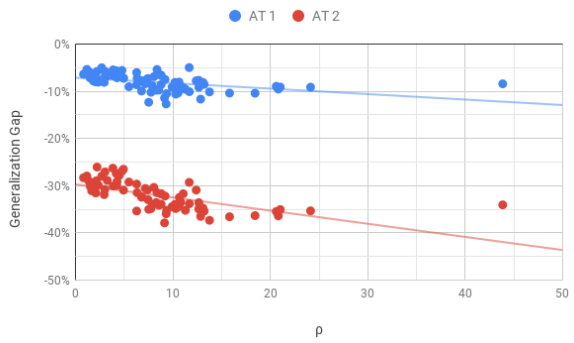


Figure 2: The generalization gap (augmented test accuracy divided by test accuracy minus 1) of USPS models tends to be larger for models with larger spectral radius ρ. A dotted linear trend line is given for reference.

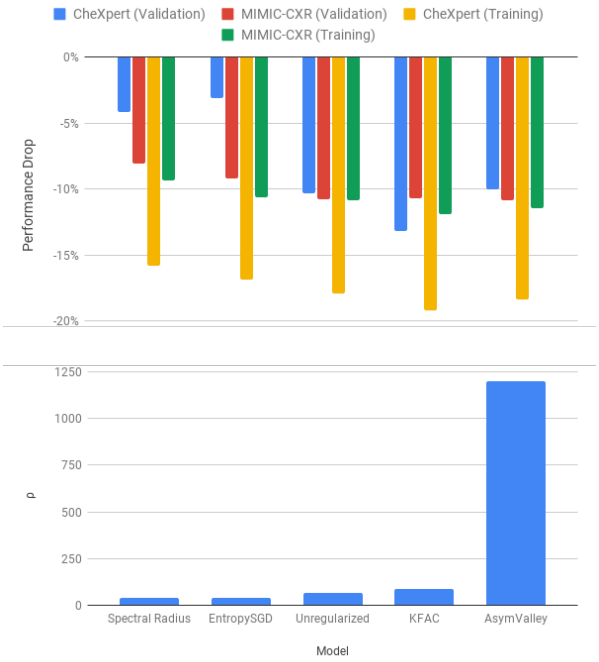


Figure 3: Chest X-ray models with low spectral radius have a lower drop in performance on distinct chest X-ray data. The performance drop was measured as the difference in mean AUC of the 6 overlapping classes from the held-out test data to the transfer learning data.

6 Conclusion

We developed algorithms for regularized optimization of neural networks, targeted at finding flat minima. Furthermore, we developed tools for calculating the regularization term and its gradient. We proved that these methods almost surely converge to a critical point. Then, we demonstrated that our regularization generalizes better than baseline comparisons on a range of applicable problems by designing unique methods.

However, optimal performance requires tuning the regularization parameters μ and K to balance the loss and spectral radius terms, a data- and model-dependent process. We observed that stricter regularization performed better until the regularization was too strict and the model would choose the majority class for all samples. Finding this point requires trial and error.

Acknowledgment

Research reported in this publication was supported, in part, by the National Library of Medicine, Grant Number T32LM012203. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 639–668. PMLR, 17–23 Jul 2022.

Carlo Baldassi, Clarissa Lauditi, Enrico M. Malatesta, Gabriele Perugini, and Riccardo Zecchina. Unveiling the structure of wide flat minima in neural networks. *Physical Review Letters*, 127:278301, Dec 2021.

Jock A. Blackard and Denis J. Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and Electronics in Agriculture*, vol.24:131–151, 1999.

Léon Bottou. Online learning and stochastic approximations. In *Online Learning in Neural Networks*, pages 9–42. Cambridge University Press, 1998.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. In *International Conference on Learning Representations*, 2017.

Sachin Chhabra. PyTorch-cGAN-conditional-GAN. github.com/sachin-chhabra/Pytorch-cGAN-conditional-GAN, 2021.

Gintare Karolina Dziugaite and Daniel Roy. Entropy-SGD optimizes the prior of a PAC-Bayes bound: Generalization properties of entropy-SGD and data-dependent priors. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1377–1386. PMLR, 10–15 Jul 2018.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

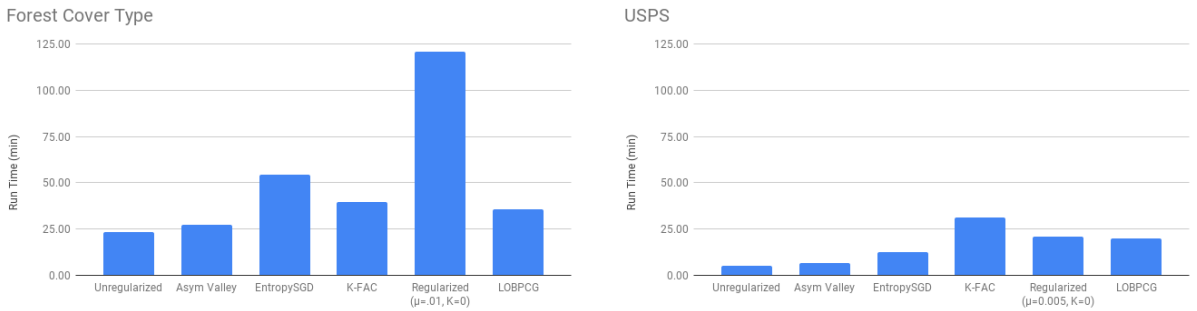


Figure 4: Computational time of models' training on forest cover-type and USPS data.

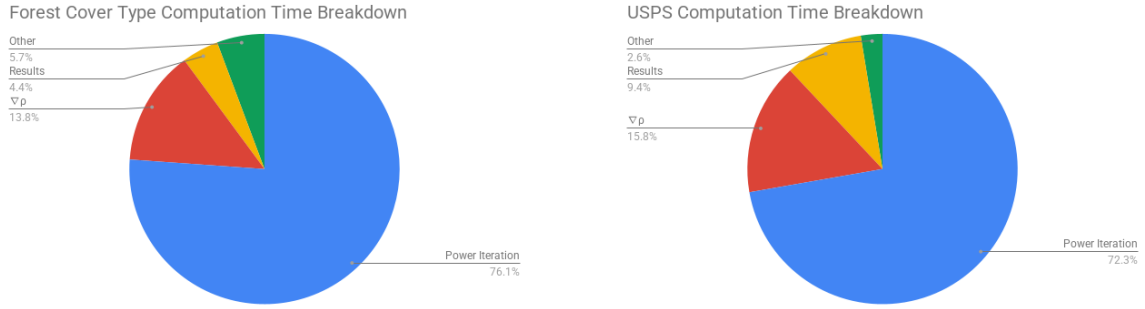


Figure 5: Breakdown of the Algorithm 1's computational time on forest cover-type and USPS data.

Jacob Gildenblat. PyTorch library for CAM methods. github.com/jacobgil/pytorch-grad-cam, 2021.

Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 2553–2564. Curran Associates, Inc., 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola. Correcting sample selection bias by unlabeled data. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn L. Ball, Katie S. Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019.

Stanisław Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Finding flatter minima with SGD, 2018.

Alistair E. W. Johnson, Tom J. Pollard, Seth J. Berkowitz, Nathaniel R. Greenbaum, Matthew P. Lungren, Chih-ying Deng, Roger G. Mark, and Steven Horng. MIMIC-CXR, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific Data*, 6(1):317, Dec 2019.

Jean Kaddour, Linqing Liu, Ricardo Silva, and Matt J Kusner. When do flat minima optimizers work? In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16577–16595. Curran Associates, Inc., 2022.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

Dong Wook Kim, Hye Young Jang, Kyung Won Kim, Youngbin Shin, and Seong Ho Park. Design characteristics of studies reporting the performance of artificial intelligence algorithms for diagnostic analysis of medical images: Results from recently published papers. *Korean Journal of Radiology*, 20:405 – 410, 2019.

Andrew V. Knyazev, Merico E. Argentati, Ilya Lashuk,

and Evgueni E. Ovtchinnikov. Block locally optimal preconditioned eigenvalue xolvers (BLOPEX) in Hypre and PETSc. *SIAM Journal on Scientific Computing*, 29(5):2224–2239, Jan 2007.

Andrew V. Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM Journal on Scientific Computing*, 23(2), 2001.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

Yann LeCun, O. Matan, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, and H. S. Baird. Handwritten zip code recognition with multilayer networks. In *Proceedings - International Conference on Pattern Recognition*, volume 2, pages 35–40. Publ by IEEE, 1990.

Erik Linder-Norén. PyTorch generative adversarial networks. github.com/eriklindernoren/PyTorch-GAN, 2021.

Linjian Ma, Gabriel Montague, Jiayu Ye, Zhewei Yao, Asghar Gholami, Kurt Keutzer, and Michael Mahoney. Inefficiency of k-fac for large batch size training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:5053–5060, 04 2020.

Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, page 1485–1488, New York, NY, USA, 2010. Association for Computing Machinery.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2408–2417, Lille, France, 07–09 Jul 2015. PMLR.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.

B. N. Parlett and W. G. Poole, Jr. A geometric theory for the QR, LU and power iterations. *SIAM Journal on Numerical Analysis*, 10(2):389–412, 1973.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Barak A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6:147–160, 1994.

Fabrizio Pittorino, Carlo Lucibello, Christoph Feinauer, Gabriele Perugini, Carlo Baldassi, Elizaveta Demyanenko, and Riccardo Zecchina. Entropic gradient descent algorithms and wide flat minima. In *International Conference on Learning Representations*, 2021.

Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Yi Ding, Aarti Bagul, Curtis Langlotz, Katie S. Shpanskaya, Matthew P. Lungren, and Andrew Y. Ng. CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning. *CoRR*, abs/1711.05225, 2017.

Hojjat Salehinejad, Jumpei Kitamura, Noah G. Ditkofsky, Amy Wei Lin, Aditya Bharatha, Suradech Suthiphosuwan, Hui-Ming Lin, Jefferson R. Wilson, Muhammad Mamdani, and Errol Colak. A real-world demonstration of machine learning generalizability in the detection of intracranial hemorrhage on head computerized tomography. *Scientific Reports*, 11, 2021.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 10 2000.

Nico P. Van der Aa, H.G. ter Morsche, and R.M.M. Mattheij. Computation of eigenvalue and eigenvector derivatives for a general complex-valued eigensystem. *Electronic Journal of Linear Algebra*, 16:300–314, 2007.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald Summers. ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3462–3471, 2017.

Chaoqi Wang. KFAC-PyTorch. github.com/alecwangcq/KFAC-Pytorch, 2019.

Lei Wu, Mingze Wang, and Weijie Su. The alignment property of sgd noise and how it helps select flat minima: A stability analysis. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 4680–4693. Curran Associates, Inc., 2022.

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W. Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 4954–4964, Red Hook, NY, USA, 2018. Curran Associates Inc.

Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning, 2017.

John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony Beardsworth Costa, Joseph J. Titano, and

Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS Medicine*, 15, 2018.

Yi Zhang, Arushi Gupta, Nikunj Saunshi, and Sanjeev Arora. On predicting generalization using GANs. In *International Conference on Learning Representations*, 2022.

Jian Zhang, Lei Qi, Yinghuan Shi, and Yang Gao. Exploring flat minima for domain generalization with large learning rates. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6145–6158, 2024.

Appendices

A Hessian-Vector Operations Derivation

The forward computation for each layer of a network with input x, output y, weights w, activation σ, bias I, error or loss measure $E = E(y)$, and direct derivative $e_k = dE/dy_k$ is given by:

$$x_k = x_k(y_{k-1}, w_k) = \sum_j w_{jk} y_{j(k-1)}$$
$$y_k = y_k(x_k, I_k) = \sigma_k(x_k) + I_k$$

The backward computation:

$$\frac{\partial E}{\partial y_k} = e_k(y_k) + \sum_j w_{jk} \frac{\partial E}{\partial x_j}$$
$$\frac{\partial E}{\partial x_k} = \sigma_k'(x_k) \frac{\partial E}{\partial y_k}$$
$$\frac{\partial E}{\partial w_{jk}} = y_k \frac{\partial E}{\partial x_j}$$

Applying $\mathcal{R}_v\{\cdot\}$ to forward pass:

$$\mathcal{R}_v\{x_k; w\} = \sum_j \left(w_{jk} \mathcal{R}_v\left\{y_{j(k-1)}; w\right\} + v_{jk} y_{j(k-1)} \right)$$
$$\mathcal{R}_v\{y_k; w\} = \mathcal{R}_v\{x_k; w\} \sigma_k'(x_k)$$

The backward computation follows as:

$$\begin{aligned}
\mathcal{R}_v\left\{\frac{\partial E}{\partial y_k}; w\right\} =& e_k'(y_k) \mathcal{R}_v\{y_k; w\} \\
&+ \sum_j \left[w_{jk} \mathcal{R}_v\left\{\frac{\partial E}{\partial x_j}; w\right\} + v_{jk} \frac{\partial E}{\partial x_j} \right] \\
\mathcal{R}_v\left\{\frac{\partial E}{\partial x_k}; w\right\} =& \sigma_k'(x_k) \mathcal{R}_v\left\{\frac{\partial E}{\partial y_k}; w\right\} \\
&+ \mathcal{R}_v\{x_k; w\} \sigma_k''(x_k) \frac{\partial E}{\partial y_k} \\
\mathcal{R}_v\left\{\frac{\partial E}{\partial w_{jk}}; w\right\} =& y_k \mathcal{R}_v\left\{\frac{\partial E}{\partial x_j}; w\right\} + \mathcal{R}_v\{y_k; w\} \frac{\partial E}{\partial x_j}
\end{aligned}$$

This yields the result found in Pearlmutter [1994]. However, we extend it one step further by applying $\mathcal{R}_v\{\cdot\}$ again, i.e., applying $\mathcal{R}_v^2\{\cdot\} = \mathcal{R}_v\{\mathcal{R}_v\{\cdot\}\}$ to the original forward pass:

$$\begin{aligned}
\mathcal{R}_v^2\{x_k; w\} =& \sum_j \Big[w_{ji} \mathcal{R}_v^2\left\{y_{j(k-1)}; w\right\} \\
&+ 2 v_{jk} \mathcal{R}_v\left\{y_{j(k-1)}; w\right\} \Big] \\
\mathcal{R}_v^2\{y_k; w\} =& \mathcal{R}_v^2\{x_k; w\} \sigma_k'(x_k) \\
&+ \left(\mathcal{R}_v\{x_k; w\}\right)^2 \sigma_k''(x_k)
\end{aligned}$$

The backward computation follows as:

$$\begin{aligned}
\mathcal{R}_v^2\left\{\frac{\partial E}{\partial y_k};w\right\} =& e_k''(y_k)\left(\mathcal{R}_v\left\{y_k;w\right\}\right)^2 \\
&+ e_k'(y_k)\mathcal{R}_v^2\left\{y_k;w\right\} \\
&+ \sum_j \left[w_{jk}\mathcal{R}_v^2\left\{\frac{\partial E}{\partial x_j};w\right\} + \right. \\
&\left. 2v_{jk}\mathcal{R}_v\left\{\frac{\partial E}{\partial x_j};w\right\}\right] \\
\mathcal{R}_v^2\left\{\frac{\partial E}{\partial x_k};w\right\} =& 2\mathcal{R}_v\left\{x_k;w\right\}\sigma_k''(x_k)\mathcal{R}_v\left\{\frac{\partial E}{\partial y_k};w\right\} \\
&+ \sigma_k'(x_k)\mathcal{R}_v^2\left\{\frac{\partial E}{\partial y_k};w\right\} \\
&+ \mathcal{R}_v^2\left\{x_k;w\right\}\sigma_k''(x_k)\frac{\partial E}{\partial y_k} \\
&+ \left(\mathcal{R}_v\left\{x_k;w\right\}\right)^2\sigma_k'''(x_k)\frac{\partial E}{\partial y_k} \\
\mathcal{R}_v^2\left\{\frac{\partial E}{\partial w_{jk}};w\right\} =& 2\mathcal{R}_v\left\{y_k;w\right\}\mathcal{R}_v\left\{\frac{\partial E}{\partial x_j};w\right\} \\
&+ y_k\mathcal{R}_v^2\left\{\frac{\partial E}{\partial x_j};w\right\} \\
&+ \mathcal{R}_v^2\left\{y_k;w\right\}\frac{\partial E}{\partial x_j}
\end{aligned}$$

The original formulation $\mathcal{R}_v\{\cdot\}$ allows us to efficiently compute $H(w)v$, which can be used to compute $\rho(w)$ and/or estimate the eigenvector $\bar{v}$ corresponding to the spectral radius (via power iteration or LOBPCG). However, the extended formulation $\mathcal{R}_v^2\{\cdot\}$ allows us to efficiently compute $v^T\nabla H(w)v$ and thus $\nabla\rho(w)$. This enables us to efficiently compute the gradient of our optimization problem for use in gradient descent methods.

B Convergence Analysis Proofs

B.1 Stochastic Gradient Descent Convergence

First, we prove the convergence of our regularization term (Lemma 4.2).

Proof. We start by splitting $v_k^T\nabla H_k v_k$ into its components

$$\begin{aligned}
v_k^T\nabla H_k v_k =& (v_k-\bar{v}_k+\bar{v}_k)^T\nabla H_k(v_k-\bar{v}_k+\bar{v}_k) \\
=& (v_k-\bar{v}_k)^T\nabla H_k(v_k-\bar{v}_k) \\
&+ 2(v_k-\bar{v}_k)^T\nabla H_k\bar{v}_k + \bar{v}_k^T\nabla H_k\bar{v}_k.
\end{aligned}$$

The last term $\bar{v}_k^T\nabla H_k\bar{v}_k = \nabla\bar{\rho}_k$ by definition. We apply the triangle inequality and bind the other terms. Given the convergence criteria on v_k and Assumptions A1 and A4 (with $||H(w)-H(\omega)|| \le L||w-\omega||,\ \forall\ w,\omega \in \mathbb{R}^n, L \ge 0$), it follows that

$$\left|\left|(v_k-\bar{v}_k)^T\nabla H_k\bar{v}_k\right|\right| \le L\left||v_k-\bar{v}_k\right|| \le L\varepsilon_k.$$

For the first term, we similarly get

$$\begin{aligned}
\left|\left|(v_k-\bar{v}_k)^T\nabla H_k(v_k-\bar{v}_k)\right|\right| \le& L\left||v_k-\bar{v}_k\right||^2 \\
\le& L\varepsilon_k^2.
\end{aligned}$$

Given Assumption A5, the limit

$$\lim_{k\to\infty} v_k^T\nabla H_k v_k = \lim_{k\to\infty}\bar{v}_k^T\nabla H_k\bar{v}_k = \lim_{k\to\infty}\nabla\bar{\rho}_k.$$

□

Next, we prove that Algorithms 1 and 2 follow the assumed update steps (Lemma 4.3):

Proof. We begin by splitting p_k into its components

$$\begin{aligned}
||p_k||^2 =& ||p_k-\nabla g_k+\nabla g_k||^2 \\
=& ||p_k-\nabla g_k||^2 + ||\nabla g_k||^2 \\
&+ 2\left(p_k-\nabla g_k\right)^T\nabla g_k.
\end{aligned}$$

Let us first assume $\rho_k > K$. By the definition of g_k and the triangle inequality,

$$||\nabla g_k||^2 \le ||\nabla f_k||^2 + 2\mu||\nabla f_k||||\nabla\rho_k|| + \mu^2||\nabla\rho_k||^2.$$

Taking the expectation (with respect to batch B_k conditioned on the history $\mathcal{P}_k$) and applying the Cauchy-Schwarz inequality yields

$$\begin{aligned}
\mathbb{E}_{B_k}\left[||p_k||^2\middle|\mathcal{P}_k\right] \le& \mathbb{E}_{B_k}\left[||\nabla f_k||^2\middle|\mathcal{P}_k\right] \\
&+ 2\mu\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^2\middle|\mathcal{P}_k\right]\right)^{\frac{1}{2}} \\
&\times\left(\mathbb{E}_{B_k}\left[||\nabla\rho_k||^2\middle|\mathcal{P}_k\right]\right)^{\frac{1}{2}} \\
&+ \mu^2\,\mathbb{E}_{B_k}\left[||\nabla\rho_k||^2\middle|\mathcal{P}_k\right].
\end{aligned}$$

Applying Hölder's inequality with $||\nabla f_k||^2$, $||\nabla\rho_k||$, $p = 3/2$, and $q = 3$ yields

$$\begin{aligned}
\mathbb{E}_{B_k}\left[||\nabla f_k||^2||\nabla\rho_k||\middle|\mathcal{P}_k\right] \le& \left(\mathbb{E}_{B_k}\left[||\nabla f_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{2}{3}} \\
&\times\left(\mathbb{E}_{B_k}\left[||\nabla\rho_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{1}{3}}.
\end{aligned}$$

Similarly, using $||\nabla f_k||^3$, $||\nabla\rho_k||$, $p = 4/3$, and $q = 4$ yields

$$\begin{aligned}
\mathbb{E}_{B_k}\left[||\nabla f_k||^3||\nabla\rho_k||\middle|\mathcal{P}_k\right] \le& \left(\mathbb{E}_{B_k}\left[||\nabla f_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{3}{4}} \\
&\times\left(\mathbb{E}_{B_k}\left[||\nabla\rho_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{1}{4}}.
\end{aligned}$$

Applying this, we obtain

$$\begin{aligned}
\mathbb{E}_{B_k}\left[||p_k||^3\middle|\mathcal{P}_k\right] \le& \mathbb{E}_{B_k}\left[||\nabla f_k||^3\middle|\mathcal{P}_k\right] \\
&+ 3\mu\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{2}{3}} \\
&\times\left(\mathbb{E}_{B_k}\left[||\nabla\rho_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{1}{3}} \\
&+ 3\mu^2\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{1}{3}} \\
&\times\left(\mathbb{E}_{B_k}\left[||\nabla\rho_k||^3\middle|\mathcal{P}_k\right]\right)^{\frac{2}{3}} \\
&+ \mu^3\,\mathbb{E}_{B_k}\left[||\nabla\rho_k||^3\middle|\mathcal{P}_k\right],
\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{B_k}\left[||p_k||^4\middle|\mathcal{P}_k\right] \leq & \mathbb{E}_{B_k}\left[||\nabla f_k||^4\middle|\mathcal{P}_k\right] \\ & + 4\mu\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{3}{4}} \\ & \times\left(\mathbb{E}_{B_k}\left[||\nabla \rho_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{1}{4}} \\ & + 6\mu^2\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{1}{2}} \\ & \times\left(\mathbb{E}_{B_k}\left[||\nabla \rho_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{1}{2}} \\ & + 4\mu^3\left(\mathbb{E}_{B_k}\left[||\nabla f_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{1}{4}} \\ & \times\left(\mathbb{E}_{B_k}\left[||\nabla \rho_k||^4\middle|\mathcal{P}_k\right]\right)^{\frac{3}{4}} \\ & + \mu^4\,\mathbb{E}_{B_k}\left[||\nabla \rho_k||^4\middle|\mathcal{P}_k\right].\end{aligned}$$

Given Assumption A3, this implies that

$$\mathbb{E}_{B_k}\left[||p_k||^j\middle|\mathcal{P}_k\right] \leq \overline{A}_j + \overline{B}_j||w_k||^j,$$

for $j = 2, 3, 4$ and some positive constants $\overline{A}_j, \overline{B}_j$. Combining this with the above results shows that the second, third, and fourth moments of the update term are bounded, as required.

If $p_k \leq K$, then $p_k = \nabla f_k$, and the statement follows from Assumption A3. □

The rest of this proof uses our assumptions and lemmas and follows Bottou's [1998] proof that SGD converges. In the next step, we prove confinement (Lemma 4.4).

Proof. Let $\varphi(x) := \begin{cases}0, & x < D,\\ (x-D)^2, & x \geq D,\end{cases}$ and $\psi_k := \varphi(||w_k||^2)$. This implies that

$$\varphi(y) - \varphi(x) \leq (y-x)\varphi'(x) + (y-x)^2,$$

for $y, x \in \mathbb{R}$. Note that this becomes an equality when $x, y > D$.

Applying this to $\psi_{k+1} - \psi_k$, we derive

$$\begin{aligned}\psi_{k+1} - \psi_k \leq & \left(-2\alpha_k w_k^T p_k + \alpha_k^2||p_k||^2\right)\psi'(||w_k||^2) \\ & + 4\alpha_k^2\left(w_k^T p_k\right)^2 - 4\alpha_k^3 w_k^T p_k||p_k||^2 \\ & + \alpha_k^4||p_k||^4.\end{aligned}$$

By the Cauchy-Schwartz inequality, we get

$$\begin{aligned}\psi_{k+1} - \psi_k \leq & -2\alpha_k w_k^T p_k\psi'(||w_k||^2) \\ & + \alpha_k^2||p_k||^2\psi'(||w_k||^2) \\ & + 4\alpha_k^2||w_k||^2||p_k||^2 + 4\alpha_k^3||w_k||||p_k||^3 \\ & + \alpha_k^4||p_k||^4.\end{aligned}$$

Taking the expectation, we have

$$\begin{aligned}\mathbb{E}_{B_k}\left[\psi_{k+1} - \psi_k\middle|\mathcal{P}_k\right] \leq & -2\alpha_k w_k^T\nabla g_k\psi'(||w_k||^2) \\ & + \alpha_k^2\,\mathbb{E}_{B_k}\left[||p_k||^2\middle|\mathcal{P}_k\right]\psi'(||w_k||^2) \\ & + 4\alpha_k^2||w_k||^2\,\mathbb{E}_{B_k}\left[||p_k||^2\middle|\mathcal{P}_k\right] \\ & + 4\alpha_k^3||w_k||\,\mathbb{E}_{B_k}\left[||p_k||^3\middle|\mathcal{P}_k\right] \\ & + \alpha_k^4\,\mathbb{E}_{B_k}\left[||p_k||^4\middle|\mathcal{P}_k\right].\end{aligned}$$

Given Assumption A2, for sufficiently large k, $\alpha_k^2 \geq \alpha_k^3 \geq \alpha_k^4$. Due to Lemma 4.3, there exist positive constants A_0, B_0 such that

$$\begin{aligned}\mathbb{E}_{B_k}\left[\psi_{k+1} - \psi_k\middle|\mathcal{P}_k\right] \leq & -2\alpha_k w_k^T\nabla g_k\psi'(||w_k||^2) \\ & + \alpha_k^2\left(A_0 + B_0||w_k||^4\right),\end{aligned}$$

and thus, there exist positive constants A, B such that

$$\begin{aligned}\mathbb{E}_{B_k}\left[\psi_{k+1} - \psi_k\middle|\mathcal{P}_k\right] \leq & -2\alpha_k w_k^T\nabla g_k\psi'(||w_k||^2) \\ & + \alpha_k^2\left(A + B\psi_k\right).\end{aligned}$$

If $||w_k||^2 < D$, then $\psi'(||w_k||^2) = 0$, and the first term on the right-hand side is zero. If $||w_k||^2 \geq D$, by Assumption A6, the first term of the right-hand side is negative. Therefore,

$$\mathbb{E}_{B_k}\left[\psi_{k+1} - \psi_k\middle|\mathcal{P}_k\right] \leq \alpha_k^2\left(A + B\psi_k\right).$$

We then transform the expectation inequality to

$$\mathbb{E}_{B_k}\left[\psi_{k+1} - (1+\alpha_k^2 B)\psi_k\middle|\mathcal{P}_k\right] \leq \alpha_k^2 A.$$

We define the sequences $\phi_k, \widetilde{\psi}_k$ as follows:

$$\phi_k := \prod_{i=1}^{k-1}\frac{1}{1+\alpha_i^2 B} \text{ and } \widetilde{\psi}_k := \phi_k\psi_k.$$

Note that $0 < \lim_{k\to\infty}\phi_k := \phi_\infty < \infty$ (this can be shown by considering $\log\phi_k$ and using the condition on the sum of the squared learning rate). By substituting these sequences into the above inequality, we obtain

$$\mathbb{E}_{B_k}\left[\widetilde{\psi}_{k+1} - \widetilde{\psi}_k\middle|\mathcal{P}_k\right] \leq \alpha_k^2\phi_{k+1}A.$$

By defining $\delta_k(u) := \left(\mathbb{E}\left[u_{k+1} - u_k\right]\right)^+$, for some process u_k, we can bound the positive expected variations of $\widetilde{\psi}_k$, as follows

$$\begin{aligned}\mathbb{E}\left[\delta_k\left(\widetilde{\psi}\right)\right] = & \mathbb{E}\left[\left(\mathbb{E}_{B_k}\left[\widetilde{\psi}_{k+1} - \widetilde{\psi}_k\middle|\mathcal{P}_k\right]\right)^+\right] \\ \leq & \alpha_k^2\phi_{k+1}A.\end{aligned}$$

Due to Assumption A2, the sum of this expectation is finite. By the Quasi-Martingale Convergence Theorem, $\widetilde{\psi}_k$ converges almost surely. And, since ϕ_k converges to $\phi_\infty > 0$, ψ_k converges almost surely. Suppose $\lim_{k\to\infty}\psi_k = \psi_\infty > 0$.

If $\{w_k\}_{k=1}^\infty$ is unbounded, then for sufficiently large $k \geq \kappa$, $||w_k||^2 > D+1$ and $\psi'(||w_k||^2) \geq c_1 > 0$. Without loss of generality, we assume this instead of dealing with a subsequence. Under these conditions, the given inequality becomes equality

$$\begin{aligned}\psi_{k+1} - \psi_k = & \left(-2\alpha_k w_k^T p_k + \alpha_k^2||p_k||^2\right)\psi'(||w_k||^2) \\ & + \left(-2\alpha_k w_k^T p_k + \alpha_k^2||p_k||^2\right)^2.\end{aligned}$$

Therefore, we can express ψ_∞ as the infinite sum

$$\begin{aligned}\psi_\infty - \psi_\kappa = & \sum_{k=\kappa}^\infty\left[\psi_{k+1} - \psi_k\right] \\ = & \sum_{k=\kappa}^\infty\Big[\left(-2\alpha_k w_k^T p_k + \alpha_k^2||p_k||^2\right)\psi'(||w_k||^2) \\ & + \left(-2\alpha_k w_k^T p_k + \alpha_k^2||p_k||^2\right)^2\Big].\end{aligned}$$

The next statements hold almost surely. We have

$$\sum_{k=\kappa}^{\infty} \left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right)^2 \leq \sum_{k=\kappa}^{\infty} \alpha_k^2 \left(A + B\psi_k\right).$$

This can be seen by expanding the square and using Cauchy-Schwarz and Lemma 4.3. Since ψ_k converges almost surely, it is bounded above by $\psi_k \leq c_2$ almost surely. Defining $c_3 := A + Bc_2$, we have

$$\sum_{k=\kappa}^{\infty} \left(-2\alpha_k w_k^T p_k + \alpha_k^2 ||p_k||^2\right)^2 \leq \sum_{k=\kappa}^{\infty} \alpha_k^2 c_3.$$

Assumption A2 implies the convergence of the series on the right-hand side. Because the terms of the sum on the left are non-negative, this sum also converges almost surely by the Monotone Convergence Theorem. Similarly, the sum $\sum_{k=\kappa}^{\infty} \alpha_k^2 ||p_k||^2 \psi'(||w_k||^2)$ converges almost surely. This uses Assumption A2, Lemma 4.3, and that ψ_k and $\psi'(||w_k||^2)$ are positive and almost surely bounded above.

Now, we subtract these convergent series from the equation for ψ_∞ to get

$$\begin{aligned}
&\psi_\infty - \psi_\kappa - \sum_{k=\kappa}^{\infty} \alpha_k^2 \|p_k\|^2 \psi'(\|w_k\|^2) \\
&\quad - \sum_{k=\kappa}^{\infty} \left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right)^2 \\
&= \sum_{k=\kappa}^{\infty} \Big[\left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right) \psi'(\|w_k\|^2) \\
&\quad + \left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right)^2 \Big] \\
&\quad - \sum_{k=\kappa}^{\infty} \alpha_k^2 \|p_k\|^2 \psi'(\|w_k\|^2) \\
&\quad - \sum_{k=\kappa}^{\infty} \left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right)^2.
\end{aligned}$$

Since the involved series almost surely converge, we can combine the terms to obtain

$$\begin{aligned}
&\psi_\infty - \psi_\kappa - \sum_{k=\kappa}^{\infty} \Big[\alpha_k^2 \|p_k\|^2 \psi'(\|w_k\|^2) \\
&\quad + \left(-2\alpha_k w_k^T p_k + \alpha_k^2 \|p_k\|^2\right)^2 \Big] \\
&= \sum_{k=\kappa}^{\infty} -2\alpha_k w_k^T p_k \psi'(\|w_k\|^2).
\end{aligned}$$

Let us consider the sum on the right-hand side. By Assumption A6, $w_k^T \nabla g_k \geq c_4 > 0$ and thus

$$\sum_{k=\kappa}^{\infty} -2\alpha_k w_k^T p_k \psi'(\|w_k\|^2) \leq \sum_{k=\kappa}^{\infty} -2\alpha_k c_4 c_1.$$

Assumption A2 implies that this sequence diverges to negative infinity. This yields a contradiction, since the left-hand side must be a finite value. Therefore, $\{w_k\}_{k=1}^{\infty}$ must be bounded. □

Next, we prove that SGD converges almost surely (Theorem 4.1).

Proof. All statements here are taken almost surely. By Assumption A1, we have $f \in \mathcal{C}^5$. From linear algebra, the Hessian of $\rho(w)$ is continuous (the largest eigenvalue is a continuous function of a parametric matrix with continuous functions). This implies that $g \in C^3$, and thus, by Lemma 4.4 it is bounded on the set of all iterates. We can bound differences in the loss criteria g_k using a first-order Taylor expansion and bounding the second derivatives with K_1.

$$|g_{k+1} - g_k + \alpha_k p_k^T \nabla g_k| \leq \alpha_k^2 ||p_k||^2 K_1.$$

This can be rewritten as:

$$g_{k+1} - g_k \leq -\alpha_k p_k^T \nabla g_k + \alpha_k^2 ||p_k||^2 K_1.$$

Taking the expectation, we get

$$\begin{aligned}
\mathbb{E}_{B_k}\left[g_{k+1} - g_k \middle| \mathcal{P}_k\right] \leq &- \alpha_k \mathbb{E}_{B_k}\left[p_k^T \nabla g_k \middle| \mathcal{P}_k\right] \\
&+ \alpha_k^2 \mathbb{E}_{B_k}\left[||p_k||^2 \middle| \mathcal{P}_k\right] K_1.
\end{aligned}$$

We decompose $p_k = \nabla g_k + (p_k - \nabla g_k)$ and bound the expectation using Lemmas 4.3 and 4.4

$$\mathbb{E}_{B_k}\left[||p_k||^2 \middle| \mathcal{P}_k\right] \leq A_2 + B_2 ||w_k||^2 \leq K_2.$$

This yields

$$\begin{aligned}
\mathbb{E}_{B_k}\left[g_{k+1} - g_k \middle| \mathcal{P}_k\right] \leq &-\alpha_k ||\nabla g_k||^2 + \alpha_k^2 K_1 K_2 \\
&-\alpha_k \mathbb{E}_{B_k}\left[(p_k - \nabla g_k)^T \nabla g_k \middle| \mathcal{P}_k\right].
\end{aligned} \tag{1}$$

Next, we apply the Cauchy-Schwarz inequality and bound the error term. From our proof to Lemma 4.2, we have

$$\left|\left|v_k^T \nabla H_k v_k - \bar{v}_k^T \nabla H_k \bar{v}_k\right|\right| \leq 2L\varepsilon_k = C_1 \varepsilon_k.$$

This implies

$$\left|\left|\mathbb{E}_{B_k}\left[p_k \middle| \mathcal{P}_k\right] - \nabla g_k\right|\right| \leq C_1 \varepsilon_k.$$

We also bound $||\nabla g_k|| \leq C_2$ using Lemma 4.4. Combining these bounds yields

$$\left|\left|\mathbb{E}_{B_k}\left[p_k \middle| \mathcal{P}_k\right] - \nabla g_k\right|\right| ||\nabla g_k|| \leq \varepsilon_k K_3. \tag{2}$$

Applying (2) to (1) gives us

$$\mathbb{E}_{B_k}\left[g_{k+1} - g_k \middle| \mathcal{P}_k\right] \leq \alpha_k^2 K_1 K_2 + \alpha_k \varepsilon_k K_3.$$

The positive expected differences are then bounded by

$$\begin{aligned}
\mathbb{E}_{B_k}\left[\delta_k(h) \middle| \mathcal{P}_k\right] = &\mathbb{E}_{B_k}\left[\delta \mathbb{E}_{B_k}\left[g_{k+1} - g_k \middle| \mathcal{P}_k\right]\right] \\
\leq &\alpha_k^2 K_1 K_2 + \alpha_k \varepsilon_k K_3.
\end{aligned}$$

By the Quasi-Martingale Convergence Theorem, g_k converges almost surely,

$$g_k \xrightarrow[k\to\infty]{\text{a.s.}} g_\infty.$$

Since g_k converges, $\sum_{k=1}^{\infty} \mathbb{E}_{B_k}\left[g_{k+1} - g_k \middle| \mathcal{P}_k\right]$ also converges. Furthermore, the series $\sum_{k=1}^{\infty} \alpha_k^2 K_1 K_2$ and

$\sum_{k=1}^{\infty} \alpha_k \varepsilon_k K_3$ converge due to Assumption A2. From (1) we have

$$\sum_{k=1}^{\infty} \alpha_k ||\nabla g_k||^2 < \infty. \tag{3}$$

We define $\theta_k = ||\nabla g_k||^2$. The differences of θ_k are bounded using the Taylor expansion, similarly to the differences of g_k

$$\theta_{k+1} - \theta_k \leq -2\alpha_k p_k^T \nabla^2 g_k \nabla g_k + \alpha_k^2 ||p_k||^2 K_4,$$

for some constant K_4. Taking the expectation, we decompose p_k and bound $||p_k||^2$ similarly to (1).

$$\begin{aligned}\theta_{k+1} - \theta_k \leq & -2\alpha_k \nabla g_k^T \nabla^2 g_k \nabla g_k + \alpha_k^2 K_2 K_4 \\ & -2\alpha_k \mathbb{E}_{B_k}\left[(p_k - \nabla g_k)^T \nabla^2 g_k \nabla g_k \middle| \mathcal{P}_k\right]\end{aligned}$$

We also bound the second derivative by $||\nabla^2 g_k|| \leq K_5/2$ and the error term using (2), yielding

$$\begin{aligned}\mathbb{E}_{B_k}\left[\theta_{k+1} - \theta_k \middle| \mathcal{P}_k\right] \leq & \alpha_k ||\nabla g_k||^2 K_5 + \alpha_k^2 K_2 K_4 \\ & + \alpha_k \varepsilon_k K_3 K_5.\end{aligned}$$

The positive expectations are bounded,

$$\begin{aligned}\mathbb{E}_{B_k}\left[\delta_k(\theta) \middle| \mathcal{P}_k\right] = & \mathbb{E}_{B_k}\left[\delta \mathbb{E}_{B_k}\left[\theta_{k+1} - \theta_k \middle| \mathcal{P}_k\right]\right] \\ \leq & \alpha_k ||\nabla g_k||^2 K_5 + \alpha_k^2 K_2 K_4 \\ & + \alpha_k \varepsilon_k K_3 K_5.\end{aligned}$$

Since the terms on the right-hand side are sums of convergent infinite sequences (due to Assumption A2 and (3)), by the Quasi-Martingale Convergence Theorem, θ_k converges almost surely. Suppose $||\nabla g_k||$ converges to a positive value $C_3 > 0$. Then for sufficiently large $k \geq \kappa$ there exists a positive constant $0 < C_4 < C_3$ such that $||\nabla g_k|| \geq C_4$. Thus, $\sum_{k=\kappa}^{\infty} \alpha_k ||\nabla g_k||^2 \geq C_4^2 \sum_{k=\kappa}^{\infty} \alpha_k$. By Assumption A2, this diverges, contradicting (3). Therefore, the limit must be zero

$$\theta_k \xrightarrow[k\to\infty]{\text{a.s.}} 0 \text{ and } \nabla g_k \xrightarrow[k\to\infty]{\text{a.s.}} 0.$$

□

C Additional Experiment Details

The code is available at https://anonymous.4open.science/r/spectral-radius/.
The algorithm is written in Python, using PyTorch [Paszke *et al.*, 2019] and TorchVision [Marcel and Rodriguez, 2010]. Forest cover-type and USPS experiments are run on a 3.1 GHz Dual-Core Intel Core i5 processor with 16 GB 2133 MHz LPDDR3 memory. Chest X-ray experiments are run on an Intel Xeon CPU E5-2650 v4 @ 2.20GHz with an NVIDIA Tesla K40c GPU.

C.1 Data Sets

The forest cover-type data [Blackard and Dean, 1999] uses cartographic data to predict the tree species (as determined by the United States Forest Service) of a 30 x 30-meter cell. This cartographic data includes elevation, aspect, slope, distance to surface water features, distance to roadways, hill-shade index at three times of day, distance to wildfire ignition points, wilderness area designation, and soil type. Seven major tree species are included: spruce/fir, lodgepole pine, Ponderosa pine, cottonwood/willow, aspen, Douglas-fir, and krummholz. In total, 581,012 samples are included, which we split 64%/16%/20% into train/validation/test data sets.

The USPS digits data [LeCun *et al.*, 1990] includes 16 x 16 pixel greyscale images from scanned envelopes to identify which digit 0-9 each image corresponds to. This data set is already split into 7,291 training and 2,007 test images. We take 1/7 of the training set as validation.

The chest X-ray data [Wang *et al.*, 2017] contains 1024 x 1024 pixel color images of patients' chest X-rays, to identify which of fourteen lung diseases each patient has. Note that this is a multi-label problem; patients can have none, one, or multiple of these conditions. A total of 112,120 patients' images are included, taken between the years 1992 and 2015, which we split 70%/10%/20% into train/validation/test data sets.

C.2 Generalization Tests

For forest cover-type data, we weight the test subjects to shift the mean of a feature or multiple features. Since we normalize the data, the weight of each test subject is determined by the ratio of the normal probability distribution function value with and without the shift. This shift adds a slight bias to the test set that is not in the original data set. We first use this shift method to increase the mean of each feature value by 0.1, compare the accuracy of our trained models, and find that certain features are problematic. Upon further examination, this is because these features are binary factors with rare classes (so our weighting of subjects emphasizes a few of them). Then, we shift each feature (except the problematic features) by a random normal amount (with mean 0 and standard deviation 0.05), compare the accuracy of our trained models, and repeat it one thousand total times.

For USPS handwritten digits data, we augment the test set: Augmented Test 1 uses random crops (with padding) of up to one pixel and random rotations of up to 15°; Augmented Test 2 uses crops of up to two pixels and rotations of up to 30°. Note that we do not augment our training set while learning our models, as a similar augmentation would yield comparable training and test sets.

For the Conditional GAN examples, we modify Linder-Norén's [2021] implementation and generate 10,000 images from the trained generator model. We train the GAN model with a batch size of 64, cosine annealing learning rate (initially 10^{-4}), $\beta_1 = 0.5$, and $\beta_2 = 0.999$. We randomly smooth the labels to be uniform between 0.0 and 0.3 for generated samples and 0.7 and 1.0 for true samples. We also swap the labels on 1%

of batches, chosen at random. We also modify Chhabra's [2021] implementation and generate 10,000 images from the trained generator model.

For the chest X-ray data, we compare performance on two similar data sets, CheXpert [Irvin *et al.*, 2019] and MIMIC-CXR [Johnson *et al.*, 2019]. For this comparison, we only consider the six conditions common to the three data sets: atelectasis, cardiomegaly, consolidation, edema, pneumonia, and pneumothorax. Additionally, we ignore any uncertain labels in the CheXpert and MIMIC-CXR classes. We keep the assigned training and validation data sets separate for each data set, as there appear to be differences in labeling. Particularly, the CheXpert validation set is fully labeled, while the training set contains uncertain and missing labels. While these are labeled "training" and "validation" sets, we solely use them as test sets. CheXpert contains 234 validation and 223,415 training images from Stanford Hospital, taken between 2002 and 2017. MIMIC-CXR contains 2,732 validation and 369,188 training images from Beth Israel Deaconess Medical Center, taken between the years 2011 and 2016.

We measure the spectral radius ρ of each model on the full training set. We use $\varepsilon = 10^{-3}$ and a maximum of 1,000 power iterations, except for the chest X-ray models, where we use $\varepsilon = 0.1$ and a maximum of 100 power iterations. These values allow the algorithm to find an accurate eigenvalue within a reasonable run time.

C.3 Implementation

For forest cover-type models, we train using stochastic gradient descent, with learning rate $\frac{0.5}{\text{epoch \#}}$, batch size of 128, and a maximum of 100 epochs. For models with batch sizes 32 and 64, we use a $\frac{0.1}{\text{epoch \#}}$ learning rate instead. The network uses 3 hidden layers with 20 hidden nodes in each layer. The learning rate and maximum number of epochs allow our algorithm to converge to an accurate model. We experiment with other feed-forward networks, learning rates, and optimizers but find that this structure works best. Our experiments with batch size are discussed in Section 5.3.

For USPS models, we use the Adam optimizer with a learning rate of 10^{-3}, a batch size of 128, and a maximum of 100 epochs. The network takes an input image and processes it through the following layers in order: convolution to 8 channels, pool, convolution to 16 channels, pool, convolution to 32 channels, pool, fully connected to 64 nodes, fully connected to 10 nodes, softmax. All convolutions are of kernel size 3, stride 1, and padding 1; all pools are max pooling with kernel size 2 and stride 2. ReLUs are used to connect the various layers. The MNIST images are re-sized to be 16 x 16 to match the model's input size. We similarly experiment with feed-forward and other convolutional networks, learning rates, and optimizers, but find that this structure performs best.

For the chest X-ray models, we resize the images to 256 x 256 and then crop them to 224 x 224. We use the Adam optimizer with a learning rate of 10^{-5}, batch size of 4, a maximum of 100 power iterations, random initialization of each power iteration, and gradient clipping (at a magnitude of 100). The crops and small batch size are necessary for the GPUs on our server to not run out of memory (12 GB). Using a CheXNet model [Rajpurkar *et al.*, 2017] (trained using their methodology) as the initialization, we train for an additional epoch, except for the Asymmetric Valley model (which we address later). We also try 5 epochs; however, the first epoch has the highest validation mean AUC in each case.

The entropy-SGD models are trained with a learning rate of 0.1 (except on chest X-ray data, where 0.001 is used), a momentum of 0.9, and no dampening or weight decay. The K-FAC models are trained using Wang's [2019] implementation, with a learning rate of 0.001 (except on chest X-ray data, where 10^{-7} is used) and a momentum of 0.9.

The Asymmetric Valley models are trained with an initial learning rate of 0.5 for 250 epochs (iterations 161-200 utilizing SWA). For chest X-ray data, we start at the SWA step, using CheXNet initialization [Rajpurkar *et al.*, 2017].

The forest cover-type LOBPCG model is trained with regularization parameters $\mu = .0028$ and $K = 1$, update frequency $b = 4$ and learning rate $\tilde{\alpha}(j) = \exp(-4j-2)$. The USPS LOBPCG model is trained with regularization parameters $\mu = .005$ and $K = 0$, update frequency $b = 4$ and learning rate $\tilde{\alpha}(j) = \exp(-4j)$.

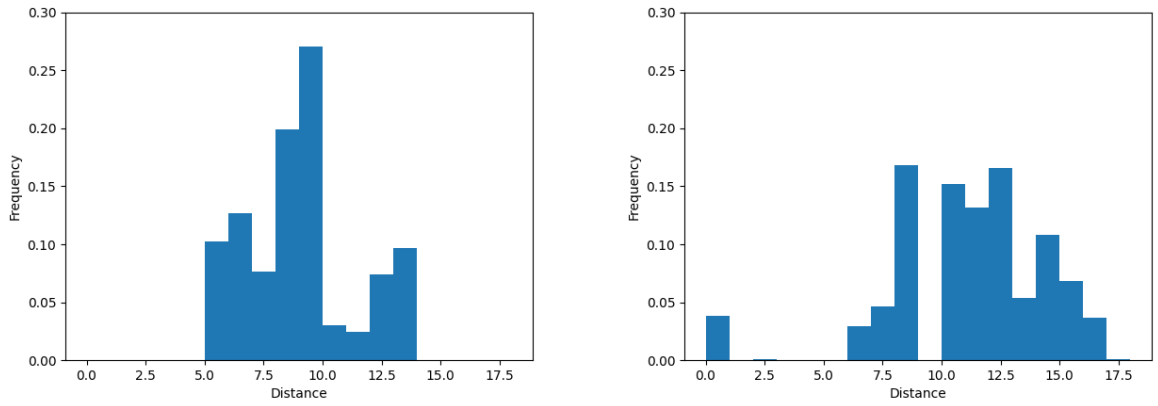
D Constructed Data Sets

During analysis of the performance of GAN1, we compute the minimum distance (L_2-norm) between each image in the GAN1 data set and the images in the USPS test data. We notice that the GAN1 images are distributed differently, relative to the USPS test data, compared to the other data sets. In particular, Figure 6 shows that the augmented data sets have a bell-curve distribution of such distances, while GAN1 has an abnormal distribution. We construct a data set, Const1, from the augmented test data sets using the following procedure.

1. Split the data with respect to distances into integer bins [0,1), [1,2), $\cdots$, [17,18).
2. Uniformly at random, select 5 bins to draw zero images from.
3. For the remaining bins, select one of the two augmented test data sets at uniform random. Add all images from the selected data set in the bin's distance range to the constructed data set.

This procedure creates a constructed data set intended to emulate the abnormal distribution of the GAN1 data.

We repeat this process with maximum cosine similarity between images and observe similar distributional abnormalities in the GAN1 data (Figure 7). We construct Const2 from the augmented data set using a similar procedure, but with bins [0.5, 0.525), [0.525, 0.55), $\cdots$, [0.975, 1.0).



(a) GAN1

(b) Const1

(c) AT1

(d) AT2

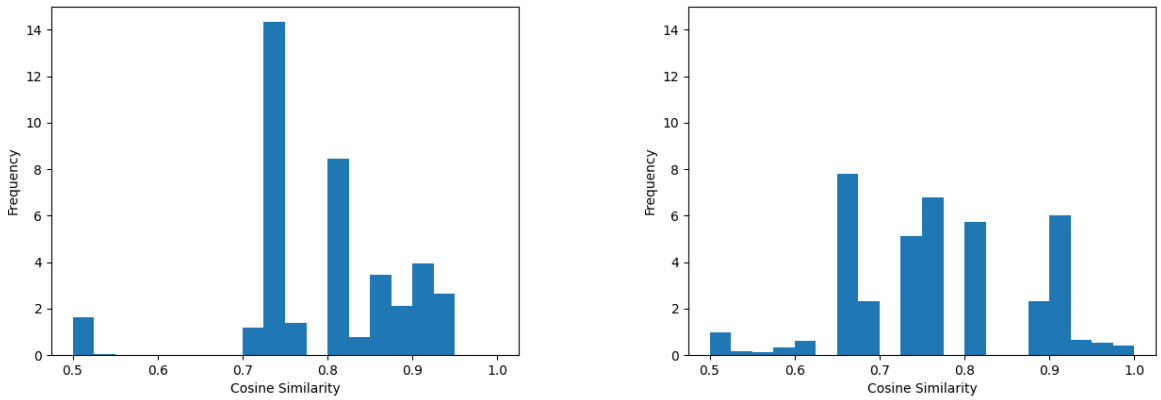
Figure 6: Histograms of Euclidean distance between data sets and USPS test data

E Grad-CAM

We use Gildenblat's [2021] Grad-CAM implementation to highlight which areas of the chest X-rays are important to our best regularized model ($\mu = 10^{-4}$ and $\alpha = 10^{-6}$) and the predictions of the baseline models. We compute the Jaccard index of the top 10% of pixels in each Grad-CAM image to compare which regions are important to each model.

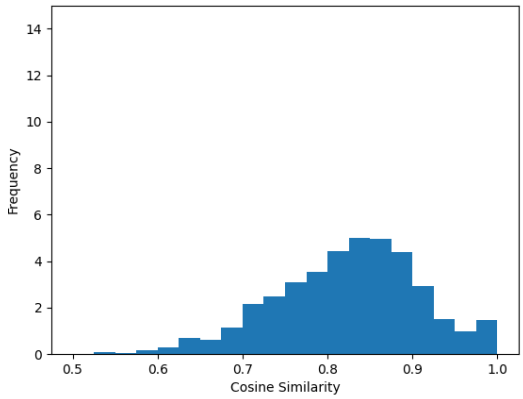
Tables 6 and 7 show that the two models with the lowest spectral radius ρ, our regularized model and entropy-SGD, have the highest overlap in explanations. These models highlight similar areas of the chest X-rays as important in making predictions. The Jaccard scores of their overlap are over .5 on the CheXpert and MIMIC-CXR validation data, the highest of any pair of models. Since these models also perform best on these transfer learning data, evidence suggests that models with low spectral radius generalize better in both their explanations and predictions. The three models with higher spectral radii have a larger dip in performance and less overlap in their explanations.

In contrast, the three models with higher spectral radius, unregularized, K-FAC, and asymmetric valley, have a higher performance drop on the transfer learning data sets and have less overlap in their explanations. These models have mean Jaccard scores of .202-.306 on the CheXpert Validation data, lower than the spectral radius regularization and EntropySGD scores of .342 and .351. The high spectral radius models have scores of .199-.249 on MIMIC-CXR Validation data, while the low spectral radius models have scores of .330 and .316.

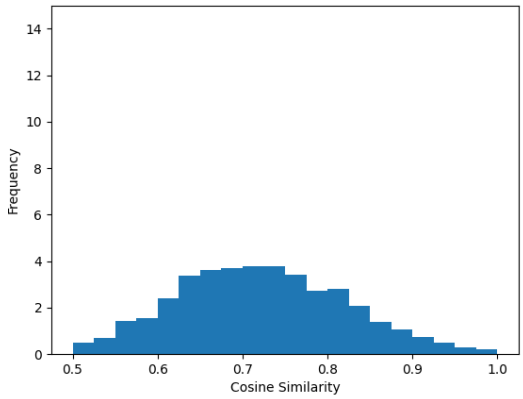


(a) GAN1

(b) Const2



(c) AT1



(d) AT2

Figure 7: Histograms of cosine similarity between data sets and USPS test data

		Jaccard Score of Overlap on CheXpert Validation Data						
Model	ρ	SpecRad	EntropySGD	UnReg	KFAC	AsymValley	Mean	PerfDrop
SpecRad	38.92	1.000	0.508	0.192	0.294	0.374	0.342	-4.15%
EntropySGD	41.11	0.508	1.000	0.181	0.280	0.435	0.351	-3.09%
UnReg	68.29	0.192	0.181	1.000	0.283	0.152	0.202	-10.30%
KFAC	84.65	0.294	0.280	0.283	1.000	0.261	0.280	-13.16%
AsymValley	1198.69	0.374	0.435	0.152	0.261	1.000	0.306	-10.06%

Table 6: There is more overlap in the explanations from the two models with low spectral radius ρ (our spectral radius regularization and entropy-SGD) on the CheXpert Validation data set.

		Jaccard Score of Overlap on MIMIC-CXR Validation Data						
Model	ρ	SpecRad	EntropySGD	UnReg	KFAC	AsymValley	Mean	PerfDrop
SpecRad	38.92	1.000	0.504	0.216	0.285	0.313	0.330	-8.10%
EntropySGD	41.11	0.504	1.000	0.172	0.229	0.358	0.316	-9.21%
UnReg	68.29	0.216	0.172	1.000	0.272	0.134	0.199	-10.80%
KFAC	84.65	0.285	0.229	0.272	1.000	0.192	0.245	-10.71%
AsymValley	1198.69	0.313	0.358	0.134	0.192	1.000	0.249	-10.84%

Table 7: There is more overlap in the explanations from the two models with low spectral radius ρ on the MIMIC-CXR Validation data set.