What is being transferred in transfer learning?

Hanie Sedghi

joint work with Behnam Neyshabur, Chiyuan Zhang

Understanding Deep Learning Phenomena

- Principled approaches to theoretically and empirically investigate deep learning phenomena, using tools from ML theory and statistics.
- To understand when and why DNNs generalize, improve training and generalization performance in state of the art deep learning models and extend the current success of our models to new domains.

Understanding Deep Learning Phenomena

- Principled approaches to theoretically and empirically investigate deep learning phenomena, using tools from ML theory and statistics.
- To understand when and why DNNs generalize, improve training and generalization performance in state of the art deep learning models and extend the current success of our models to new domains.
- Our understanding of modern neural networks lags behind their practical successes. This growing gap poses a challenge to the pace of progress.
- Although there has been some progress in this area, still we are far from answering many fundamental questions such as generalization capabilities of deep models and how to ensure successful transfer to new domains.
- I believe this understanding helps us extend beyond our current use of deep learning in a reliable way.

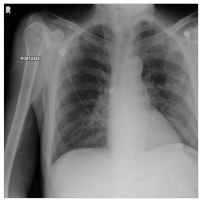
Motivation

- One desired capability of machines is to transfer their knowledge or understanding of a domain it is trained on (source domain) to another domain (target domain) where data is (usually) scarce or a fast speed of convergence is needed.
- Plethora of works using transfer learning in different applications.
- We yet do not understand:
  - what enables a successful transfer?
  - which parts of the network are responsible for that?

In this paper we address these fundamental questions and propose new tools and analysis.

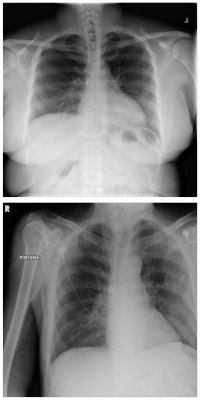
Problem Setup

- Target domains that are intrinsically different and diverse:
  - CheXpert: a medical imaging dataset of chest x-rays considering 5 different diseases.
  - DomainNet: designed to probe transfer learning for diverse visual representations. The domains range from real images to sketches, clipart and painting samples. 345 classes



Problem Setup

- Target domains that are intrinsically different and diverse:
  - CheXpert: a medical imaging dataset of chest x-rays considering 5 different diseases.
  - DomainNet: designed to probe transfer learning for diverse visual representations. The domains range from real images to sketches, clipart and painting samples. 345 classes
- Two initialization scenarios:
  - Pre-trained on ImageNet (Finetune)
  - Start from random initialization (RandInit)



Notation

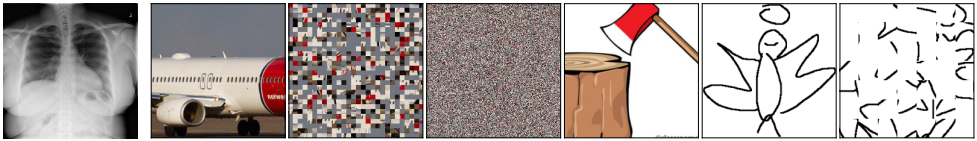
- T: trained,
- P-T: Pre-trained,
- RI: random initialization.
- Our four models are:
  - RI:random initialization
  - P: pre-trained model
  - RI-T: model trained on target domain from random initialization
  - P-T: model trained/fine-tuned on target domain starting from pre-trained weights

Role of feature reuse

- Human visual system is compositional and hierarchical.
- Modern convolutional neural networks trained on large scale visual data are shown to form similar feature hierarchies.
- The benefits of transfer learning are generally believed to come from reusing the pre-trained feature hierarchy.
- But, why in many successful applications of transfer learning, the target domain could be visually very dissimilar to the source domain?

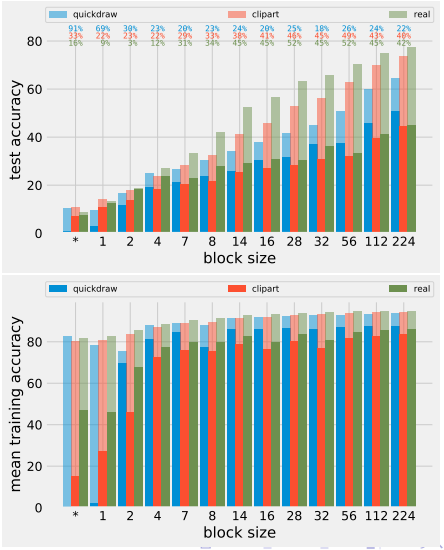
Role of feature reuse

- Comparing RI-T to P-T learning curves
  - Largest performance boost on the `real` domain, which contains natural images.
  - Even for the most distant target domains such as CheXpert and `quickdraw`, we still observe performance boosts from transfer learning.
  - The optimization for P-T also converges much faster than RI-T in all cases.
- Experiment: We partition the image of the downstream tasks into equal sized blocks and shuffle the blocks randomly. The shuffling disrupts visual features in those images.



Role of feature reuse

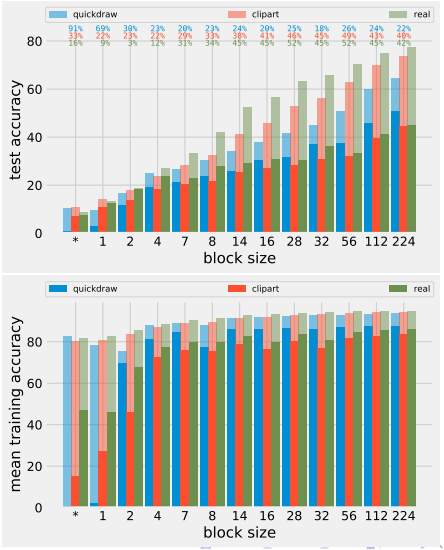
- The final performance drops for both initializations as the block size decreases, indicating the increasing difficulty of the tasks.
- The relative accuracy drop, decreases with reducing block size on both `real` and `clipart`, showing consistency with the intuition that decreasing feature reuse leads to diminishing benefits.
- On `quickdraw`, the relative accuracy drop does not show a decreasing pattern. In `quickdraw`, the input images barely contain any visual features.
- Benefits of transferred weights on optimization speed is independent from feature reuse.



Role of feature reuse

- Feature reuse plays a very important role!
  especially when the downstream task shares similar visual features with the pre-training domain.

- There are other factors at play!
  low-level statistics of the data that are not ruined in the shuffling lead to the significant benefits of transfer learning, especially on optimization speed.



Opening the model: Investigating mistakes

- Data samples where Finetune is incorrect and RandInit is correct mostly include ambiguous samples;
- Data samples where Finetune is correct, RandInit is incorrect are spread on easy samples too.
- The mistaken examples between two instances of Finetune are very similar. Two Finetune models are more similar in the feature space.

Opening the model: Feature similarity

- We use CKA [Kornblith, Norouzi, Lee, Hinton, ICML19] as a measure of feature similarity.
- Two instances of finetune model are highly similar across different layers.
- The initialization point, drastically impacts feature similarity.
- Although both networks are showing high accuracy, they are not that similar in the feature space.
- This emphasizes on role of feature reuse and that two finetune models are reusing the same features.

models/layer	conv1	layer 1	layer 2	layer 3	layer 4
P-T & P	0.6225	0.4592	0.2896	0.1877	0.0453
P-T & P-T	0.6710	0.8230	0.6052	0.4089	0.1628
P-T & RI-T	0.0036	0.0011	0.0022	0.0003	0.0808
RI-T & RI-T	0.0016	0.0088	0.0004	0.0004	0.0424

Opening the model: Distance in parameter space

- We measure ℓ_2 distance between two fine-tuned and two randinit models, both for whole network and per module.
- Randinit models are farther from each other compared to two fine-tuned models and this trend can be seen in individual modules too.
- The distance between modules increases as we move towards higher layers in the network.
- It's not just features, it's parameters too!

domain/model	2 P-T	2 RI-T	P-T & P	RI-T & P
CheXpert	200.12	255.34	237.31	598.19
clipart	178.07	822.43	157.19	811.87
quickdraw	218.52	776.76	195.44	785.22
real	193.45	815.08	164.83	796.80

Performance barriers and basins in the loss landscape

- Generalization criterion: flatness of the basin of the loss landscape near the final solution
- In a flat basin, the weights could be locally perturbed without hurting the performance,
- In a narrow basin, moving away from the minimizer would quickly hit a *barrier*, indicated by a sudden increase in the loss.
- We explore the loss landscape of P-T and RI-T.

Performance barriers and basins in the loss landscape

- Generalization criterion: flatness of the basin of the loss landscape near the final solution
- In a flat basin, the weights could be locally perturbed without hurting the performance,
- In a narrow basin, moving away from the minimizer would quickly hit a *barrier*, indicated by a sudden increase in the loss.
- We explore the loss landscape of P-T and RI-T.
- Let Θ and $\tilde{\Theta}$ be all the weights from two different checkpoints. We evaluate a series of models along the *linear* interpolation of the two weights: $\{\Theta_\lambda = (1-\lambda)\Theta + \lambda\tilde{\Theta} : \lambda \in [0,1]\}$.

Performance barriers and basins in the loss landscape

- *Any* two minimizers of a deep network can be connected via a *non-linear* low-loss path.
- Due to the non-linear and compositional structure of neural networks, the *linear* combination of the weights of two good performing models does not necessarily define a well behaved model, thus performance barriers are generally expected along the linear interpolation path.
- When the two solutions belong to the same flat basin of the loss landscape, performance barrier is absent.
- Interpolating two random solutions from the same basin could generally produce solutions closer to the center of the basin, which potentially have better generalization performance than the end points.

Definition: Basin

Given a loss function $\ell : \mathbb{R}^n \to \mathbb{R}^+$ and a closed convex set $S \subset \mathbb{R}^n$ we say that S is a (ϵ, δ)-basin for ℓ if and only if S has all following properties:

1. Let U_S be the uniform distribution over set S and $\mu_{S,\ell}$ be the expected value of the loss ℓ on samples generated from U_S.

$$\mathbb{E}_{\mathbf{w} \sim \mathsf{U}_S}[|\ell(w) - \mu_{S,\ell}|] \leq \epsilon$$

2. For any two points $w_1, w_2 \in S$, let $f(w_1, w_2) = w_1 + \tilde{\alpha}(w_2 - w_1)$, where $\tilde{\alpha} = \max_{\alpha \geq 0} : w_1 + \alpha(w_2 - w_1) \in S$. Then,

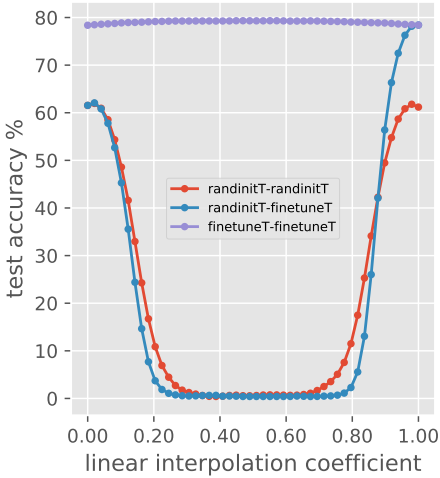
$$\mathbb{E}_{\mathbf{w_1}, \mathbf{w_2} \sim \mathsf{U}_S, \nu \sim \mathcal{N}(0, (\delta^2/n) I_n)}[\ell(f(w_1, w_2) + \nu) - \mu_{S,\ell}] \geq 2\epsilon$$

3. Let $\kappa(w_1, w_2, \nu) = f(w_1, w_2) + \frac{\nu}{\|f(w_1, w_2) - w_1\|_2}(f(w_1, w_2) - w_1)$. Then,

$$\mathbb{E}_{\mathbf{w_1}, \mathbf{w_2} \sim \mathsf{U}_S, \nu \sim \mathcal{N}(0, \delta^2)}[\ell(\kappa(w_1, w_2, |\nu|)) - \mu_{S,\ell}] \geq 2\epsilon$$

Performance barriers and basins in the loss landscape

- We evaluate a series of models along the linear interpolation of the two weights.
- Performance barriers are generally expected between two unrelated NN models.
- In a flat basin, the weights could be locally perturbed without hurting the performance.
- Finetune models reside in the same basin.
- RandInits end up in a different basin, even if starting from same random seed.



Performance barriers and basins in the loss landscape

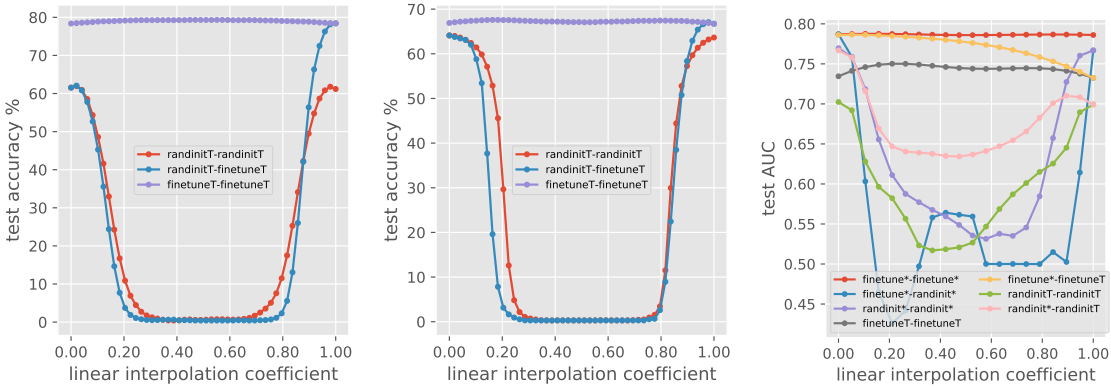


Figure: The left and middle panes show performance barrier measured by test accuracy on DomainNet `real` and `quickdraw`, respectively. The right pane shows the performance barrier measured by test AUC on CheXpert.

Performance barrier experiments with identical initialization for RI-T

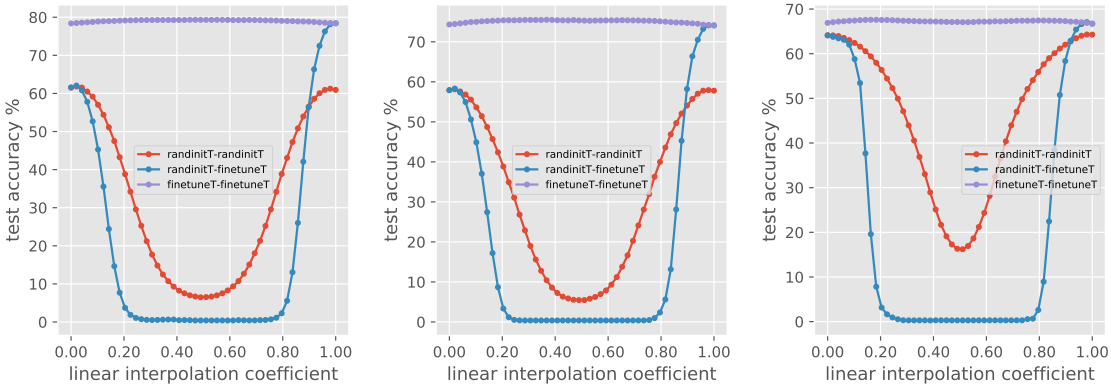


Figure: Performance barrier of `real`, `clipart`, `quickdraw`, respectively, measured by test accuracy. Like P-T, the two RI-T models are initialized from *the same* (random) weights.

Performance barrier plots with extrapolation

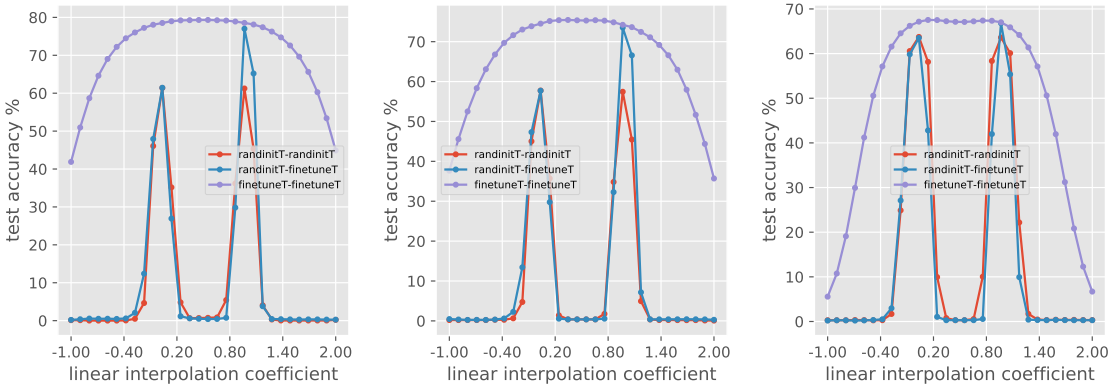
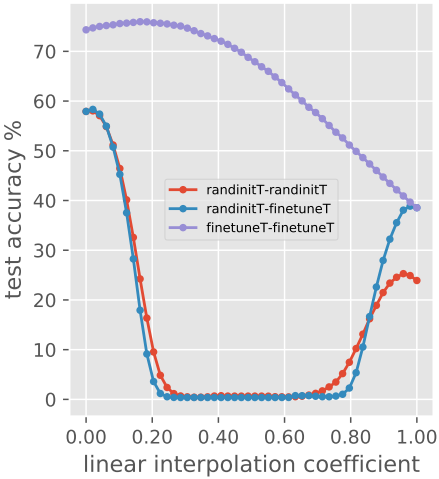


Figure: Performance barrier of `real`, `clipart`, `quickdraw`, respectively, measured by test accuracy. The linear combination of weights are extrapolated beyond $[0, 1]$ (to $[-1, 2]$).

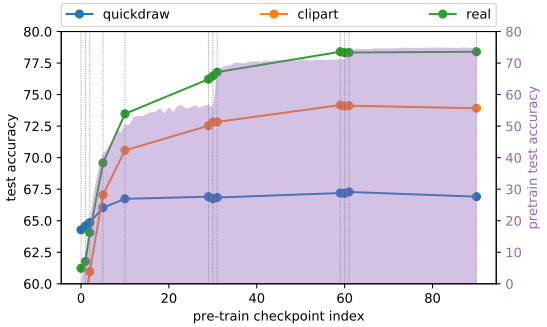
Cross-domain weight interpolation on DomainNet

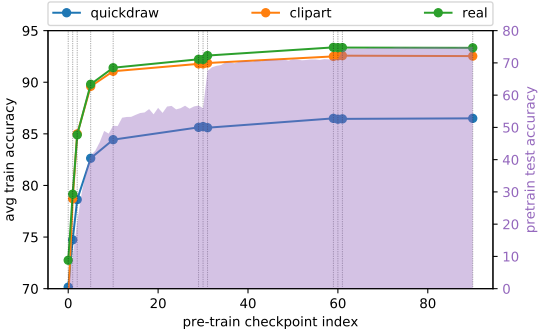
- when directly evaluated on a different domain that the models are trained from, we could still get non-trivial test performance.
- P-T consistently outperforms RI-T even in the cross-domain cases.
- when interpolating between P-T models, (instead of performance barrier) we observe performance boost in the middle of the interpolation.
- This suggests that all the trained P-T models on all domains are in one shared basin.



Which pre-trained checkpoint is most useful for transfer learning?

- Significant improvements are observed when we start from the checkpoints where the pre-training performance has been plateauing.
- Independence between the improvements on optimization speed and final performance.
- You can start from earlier checkpoints in pre-training.





Summary of Results

- Both feature-reuse and low-level statistics of the data are important.
- Finetune models make similar mistakes on target domain, they have similar features and are surprisingly close in the ℓ_2 distance in parameter space.
- Finetune models are in the same basins in the loss landscape, while models trained from random initialization are in a different basin.
- One can start from earlier checkpoints of pre-trained model without losing accuracy of the fine-tuned model.

Part II

Which part of the network is responsible for successful transfer?

Recap: Not all layers are created equal!

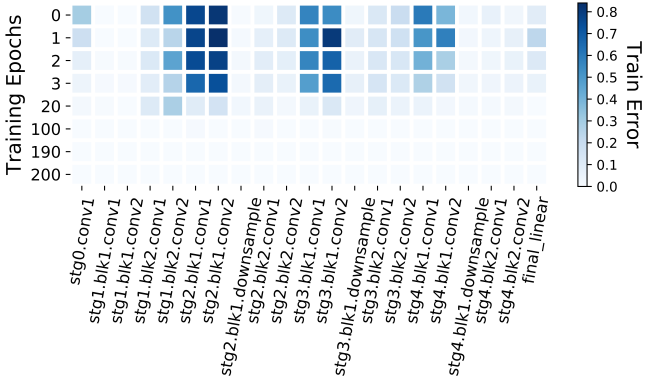
Experiment:

- Consider a deep neural network (at any training epoch).
- Pick one of the layers and rewind its value back to its value at initialization.
- Keep the value of all other layers fixed.
- Notice the change in performance.

C. Zhang, S. Bengio, Y. Singer, *Are all layers created equal?*, Feb 2019

Recap: Not all layers are created equal!

Observation: In a deep neural network, some modules are more critical than others, i.e., rewinding their parameter values back to initialization, while keeping other modules fixed at the trained parameters, results in a large drop in the network's performance.



C. Zhang, S. Bengio, Y. Singer, *Are all layers created equal?*, Feb 2019

Questions

- What implications does this have for understanding generalization in neural networks?
- Is the shape of the valley that connects the initial and final values of the module parameters informative about generalization performance of an architecture?
- Can we come up with a complexity measure based on the shape of this valley that predicts generalization performance of an architecture?

Understanding Criticality

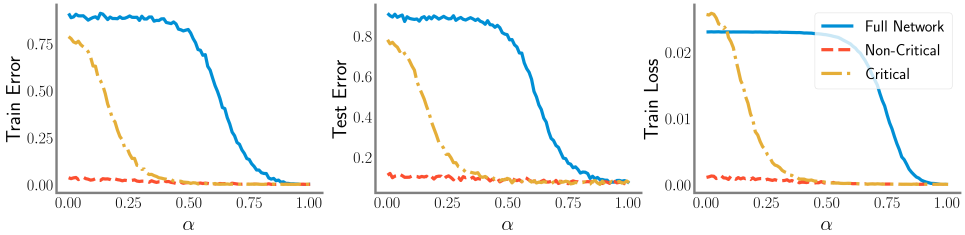
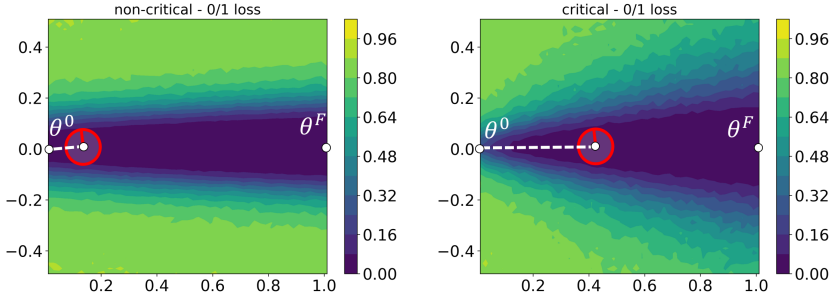


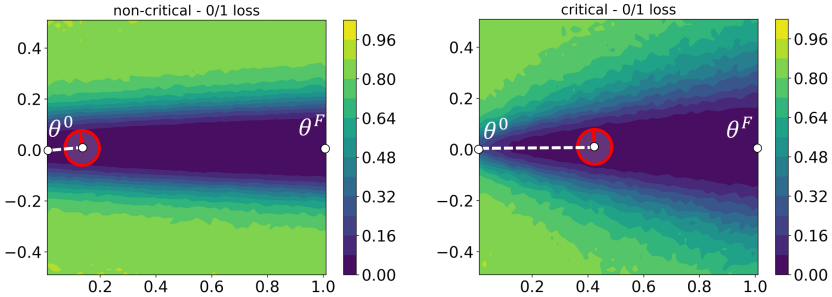
Figure: Performance degradation as we move on convex combination path from final to initial value of modules. We find that along this path the training error (as well as test error and train loss) increases monotonically from the final weights to initial weights. ResNet-18 architecture.

Module Criticality



- Loss values in the valleys that connect the initial weights θ^0 to the final weights θ^F.
- Module criticality: how far one can push the ball of radius r in the valley towards initialization divided by the radius.

Module Criticality



- Non-critical modules $\equiv$ wide valley
  Critical modules $\equiv$ sharp valley or the loss values start to increase when the ball becomes too close to the initial weight.

- Network criticality $\triangleq$ sum of the module criticalities.

Definition

Given an $\epsilon > 0$ and network f_Θ, define module criticality for module i:

$$\mu_i(\epsilon) := \min_{0 \leq \alpha_i, \sigma_i \leq 1} \left\{ \frac{\alpha_i^2 \|\theta_i^F - \theta_i^0\|_{Fr}^2}{\sigma_i^2} : \mathbb{E}_{u \sim \mathcal{N}(0, \sigma_i^2)}[L_S(f_{\theta_i^\alpha + u, \Theta_{-i}^F})] \leq \epsilon \right\},$$

$$\theta_i^\alpha = (1 - \alpha_i)\theta_i^0 + \alpha_i \theta_i^F, \alpha_i \in [0, 1]$$

σ_i is the radius of the ball around θ_i^α.
$\mathcal{L}_S$ denotes the empirical zero-one loss.

Definition

Given an $\epsilon > 0$ and network f_Θ, define module criticality for module i:

$$\mu_i(\epsilon) := \min_{0 \leq \alpha_i, \sigma_i \leq 1} \left\{ \frac{\alpha_i^2 \|\theta_i^F - \theta_i^0\|_{Fr}^2}{\sigma_i^2} : \mathbb{E}_{u \sim \mathcal{N}(0, \sigma_i^2)} [L_S(f_{\theta_i^\alpha + u, \Theta_{-i}^F})] \leq \epsilon \right\},$$

$$\theta_i^\alpha = (1 - \alpha_i)\theta_i^0 + \alpha_i \theta_i^F, \alpha_i \in [0, 1]$$

σ_i is the radius of the ball around θ_i^α.
$\mathcal{L}_S$ denotes the empirical zero-one loss.

Define the network criticality as the sum of the module criticalities $\mu_\epsilon(f_\Theta) = \sum_{i=1}^d \mu_{i,\epsilon}(f_\Theta)$.

Generalization Guarantees

- Intuitively, being able to move closer to initialization values indicate that the effective function class is smaller and hence the network should generalize better.

- Using a PAC-Bayesian analysis we show that given m samples if train error is less than ϵ then:

$$\text{Test Error} \lesssim \epsilon + \sqrt{\frac{\frac{1}{4}\mu(\epsilon) + \log\left(\frac{m}{\delta}\right)}{m}}$$

Theorem

For any data distribution D, number of samples $m \in \mathbb{N}$, for any $0 < \delta < 1$, for any $0 < \sigma_i \leq 1$ and any $0 \leq \alpha_i \leq 1$, with probability $1 - \delta$ over the choice of the training set $S_m \sim D$ the following generalization bound holds:

$$\mathbb{E}_U[\mathcal{L}_D(f_{\Theta^\alpha + U})] \leq \mathbb{E}_U[\mathcal{L}_S(f_{\Theta^\alpha + U})] + \sqrt{\frac{\frac{1}{4}\sum_{i=1}^{d} k_i \log\left(1 + \frac{\alpha_i^2 \left\|\theta_i^F - \theta_i^0\right\|_{Fr}^2}{k_i \sigma_i^2}\right) + \log\left(\frac{m}{\delta}\right) + \widetilde{\mathcal{O}}(1)}{m-1}},$$

where k_i is the number of parameters in module i. For example, for a convolution module with kernel size $q_i \times q_i$ and number of output channels c_i, $k_i = q_i^2 c_{i-1} c_i$.

Ranking architectures generalization performance

We show through extensive experiments that
module criticality is able to explain the superior generalization performance of some architectures over others, whereas earlier measures fail to do so.

The intriguing role of module criticality in the generalization of deep networks,
N. Chatterji, B. Neyshabur, H. Sedghi, Spotlight in ICLR2020

Analysis of criticality in the Transfer Learning

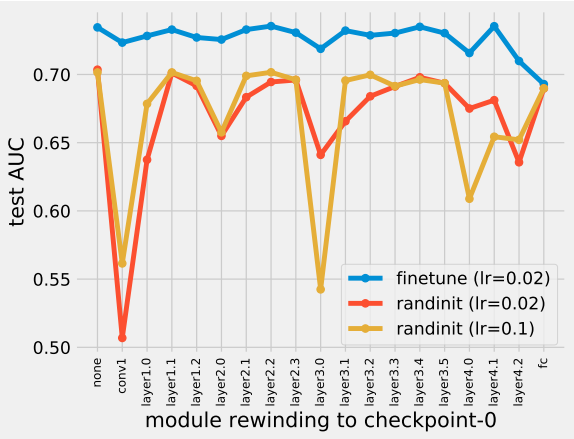


Figure: Module criticality measured on CheXpert.

Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.

Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.
- Extension of definition
  - Look at both direct path between initial and final values
  - Look at optimization path (checkpoints)

Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.
- Extension of definition
  - Look at both direct path between initial and final values
  - Look at optimization path (checkpoints)
  - Look at optimal value of weight matrices (instead of θ_i^F, we investigate θ_i^{opt})
  - Noise is proportional to Frobenius norm of the weight matrix
  - Proofs still hold.

Analysis of criticality in the Transfer Learning

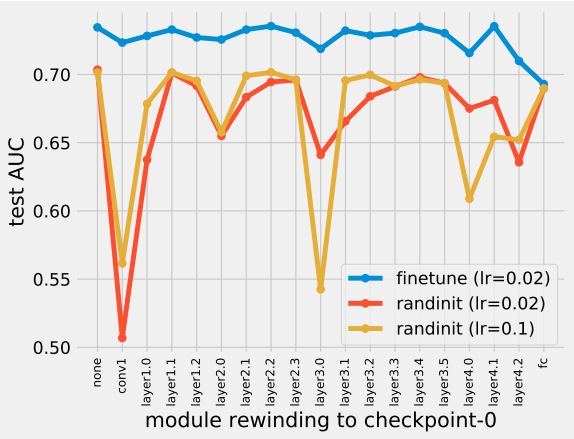


Figure: Module criticality measured on CheXpert.

Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.

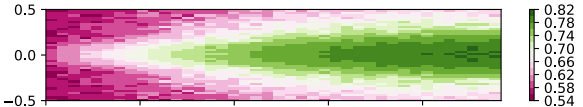
Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.
- Extension of definition
  - Look at both direct path between initial and final values
  - Look at optimization path (checkpoints)

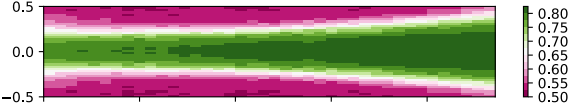
Module criticality in transfer learning regime

- Rewind analysis
  - Similar pattern to supervised learning case
  - The only difference is that the 'FC' layer becomes critical for P-T model which is expected.
- Extension of definition
  - Look at both direct path between initial and final values
  - Look at optimization path (checkpoints)
  - Look at optimal value of weight matrices (instead of θ_i^F, we investigate θ_i^{opt})
  - Noise is proportional to Frobenius norm of the weight matrix
  - Proofs still hold.

Plots for a critical and a non-critical module



(a) Module criticality Conv 1

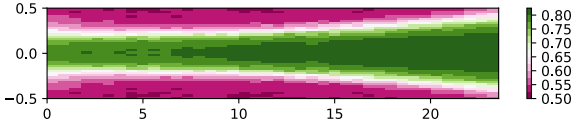


(b) Module Criticality Layer 1

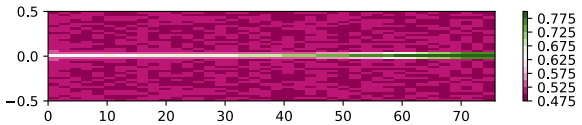
Figure: ResNet-50, transfer from ImageNet to ChexPert

Summary of observations

- As we move from the input towards the output, we see tighter valleys, i.e., modules become more critical.
- This is in agreement with observation of [Yosinski et al., 2014, Raghu et al., 2019] that lower layers are in charge of more general features while higher layers have features that are more specialized for the target domain.



(a) Module Criticality Layer1



(b) Module Criticality Layer4

Conclusion and future work

- We shed some light on what is being transferred in transfer learning and which parts of the network are at play.
- Our findings on basin in the loss landscape can be used to improve ensemble methods.
- Our observation of low-level data statistics improving training speed could lead to better network initialization methods.
- Initialization with minimum information from pre-trained model while staying in the same basin and whether this improves performance.
- Implications of these findings for parallel training and optimization.

What is being transferred in Transfer Learning?,
B. Neyshabur*, H. Sedghi*, C. Zhang*, NeurIPS 2020

References I

[1] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Advances in Neural Information Processing Systems*, pages 3342–3352, 2019.

[2] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.