On the Learning of Patterns in Deep Networks

Presenter: Jiao, Wenxiang

2020-08-06

Outline

- Memorization of random data vs. real data

A Closer Look at Memorization in Deep Networks, Yoshua Bengio, et al., ICML 2017.

- Shallow learnable data vs. deep learnable data

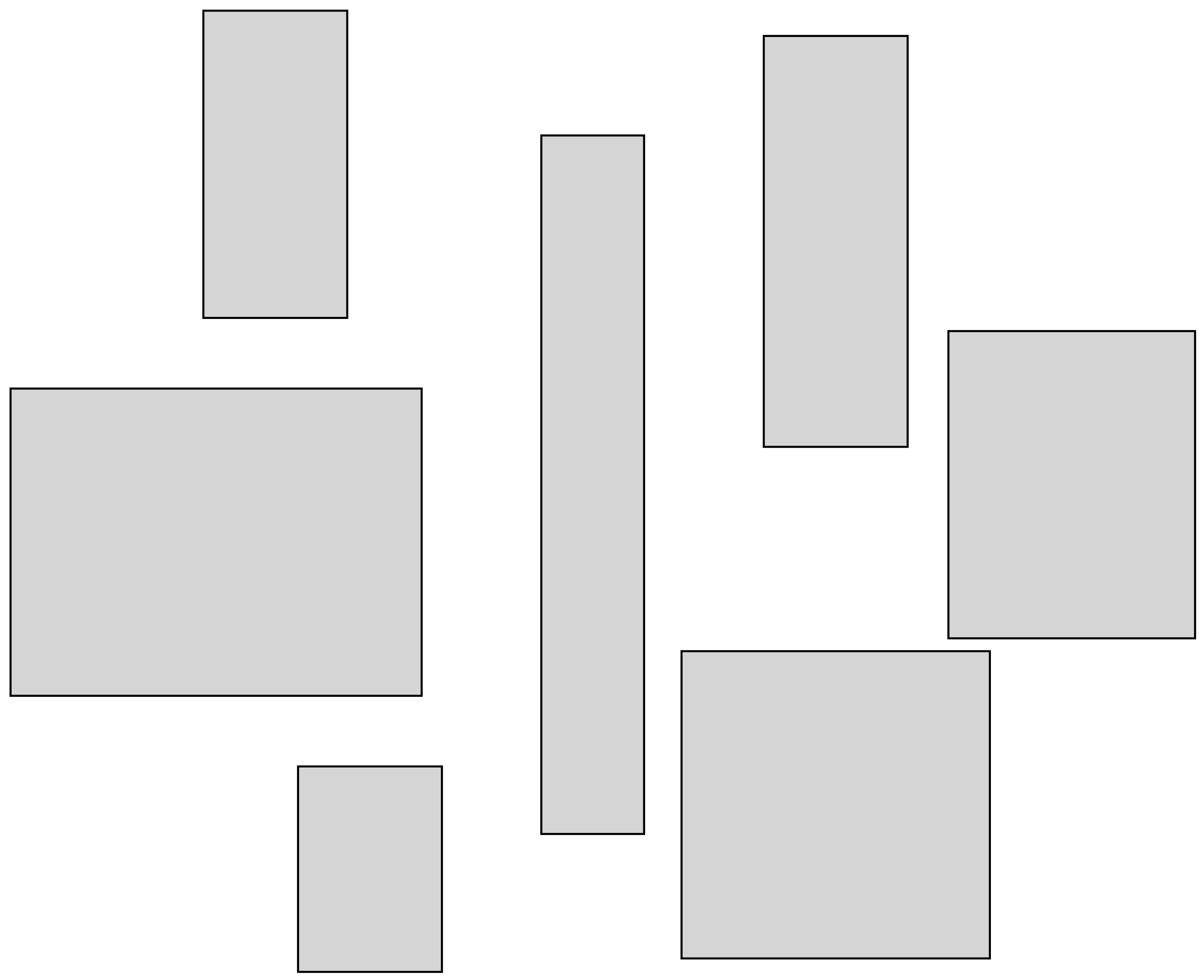
Do deep neural networks learn shallow learnable examples first? Karttikeya Mangalam and Vinay Prabhu, ICML 2019 Workshop.

A Closer Look at Memorization in Deep Networks

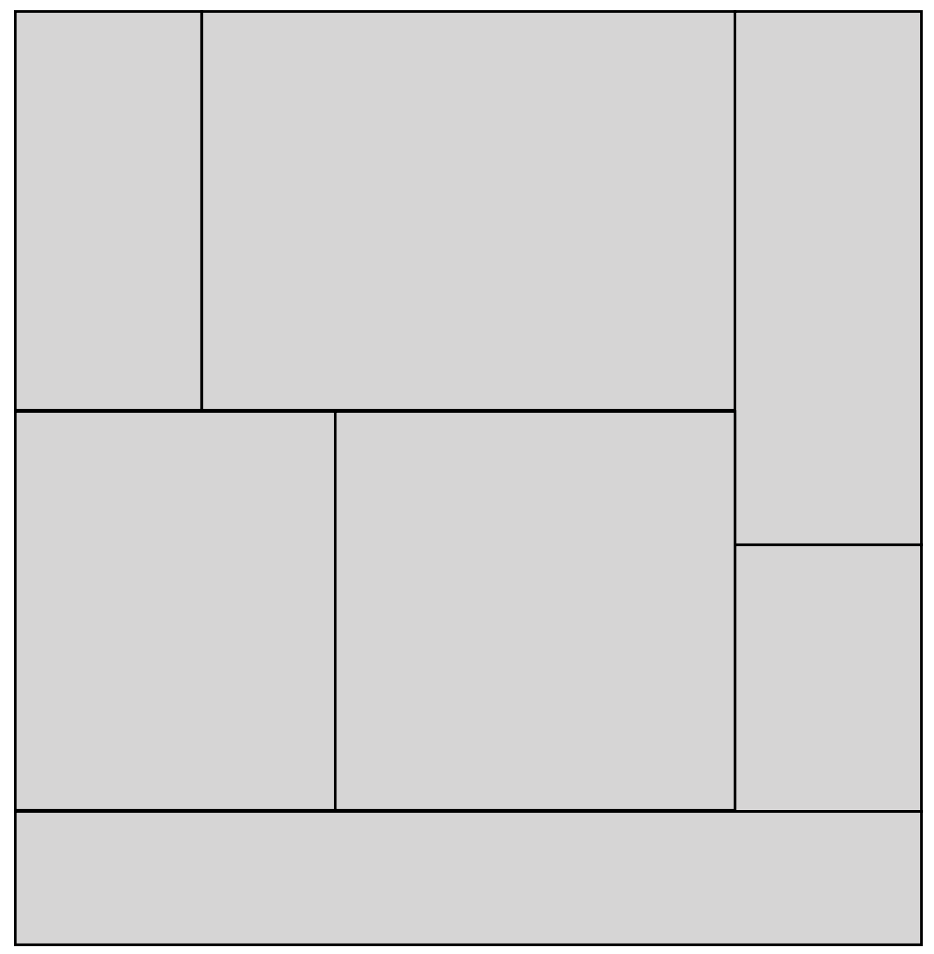
Yoshua Bengio, et al.
ICML 2017

What is memorization?

- No formal definition.
- Operational definition: behavior of DNNs trained on random data.
- Memorization does not capitalize on patterns in data (content agnostic).



Rote learning (memorization)



Meaningful learning (pattern-based)

Why is memorization important?

- Context: Understanding Deep Learning Requires Rethinking Generalization, Zhang et al., ICLR 2017.
  - Shows: DNNs can easily fit random labels.

Why is memorization important?

- Context: Understanding Deep Learning Requires Rethinking Generalization, Zhang et al., ICLR 2017.
  - Shows: DNNs can easily fit random labels.

Whether DNNs use similar memorization tactics on real data?

Why is memorization important?

- Context: Understanding Deep Learning Requires Rethinking Generalization, Zhang et al., ICLR 2017.
  - Shows: DNNs can easily fit random labels.

Whether DNNs use similar memorization tactics on real data?

OR

Are DNNs using “brute-force memorization”?

- “Brute-force Memorization”:
  - Does not capitalize on patterns shared between training examples or features.
  - Content of what is memorized is irrelevant.

Why is memorization important?

- Context: Understanding Deep Learning Requires Rethinking Generalization, Zhang et al., ICLR 2017.
  - Shows: DNNs can easily fit random labels.

Whether DNNs use similar memorization tactics on real data?

OR

Are DNNs using “brute-force memorization”?

- “Brute-force Memorization”:
  - Does not capitalize on patterns shared between training examples or features.
  - Content of what is memorized is irrelevant.
- **Data-dependent understanding on learning and generalization of DNNs!**

Main Findings

- There are qualitative differences in DNN optimization behavior on real data vs. noise. In other words, DNNs do not just memorize real data.

- DNNs learn simple patterns first, before memorizing. In other words, DNN optimization is content-aware, taking advantage of patterns shared by multiple training examples.

- Regularization techniques can differentially hinder memorization in DNNs while preserving their ability to learn about real data

Experimental Settings

- Overview of experiments:
  1. Qualitative differences in fitting noise vs. real data
  2. Deep networks learn simple patterns first
  3. Regularization can reduce memorization

- Notions:
  1. randX - random inputs (i.i.d. Gaussian)
  2. randY - random labels

1. Qualitative differences in fitting noise vs. real data

- Easy Examples as Evidence of Patterns in Real Data
  - A brute-force memorization approach to fitting data should apply equally well to different training examples.
  - If a network is learning based on patterns in the data, some examples may fit these patterns better than others.

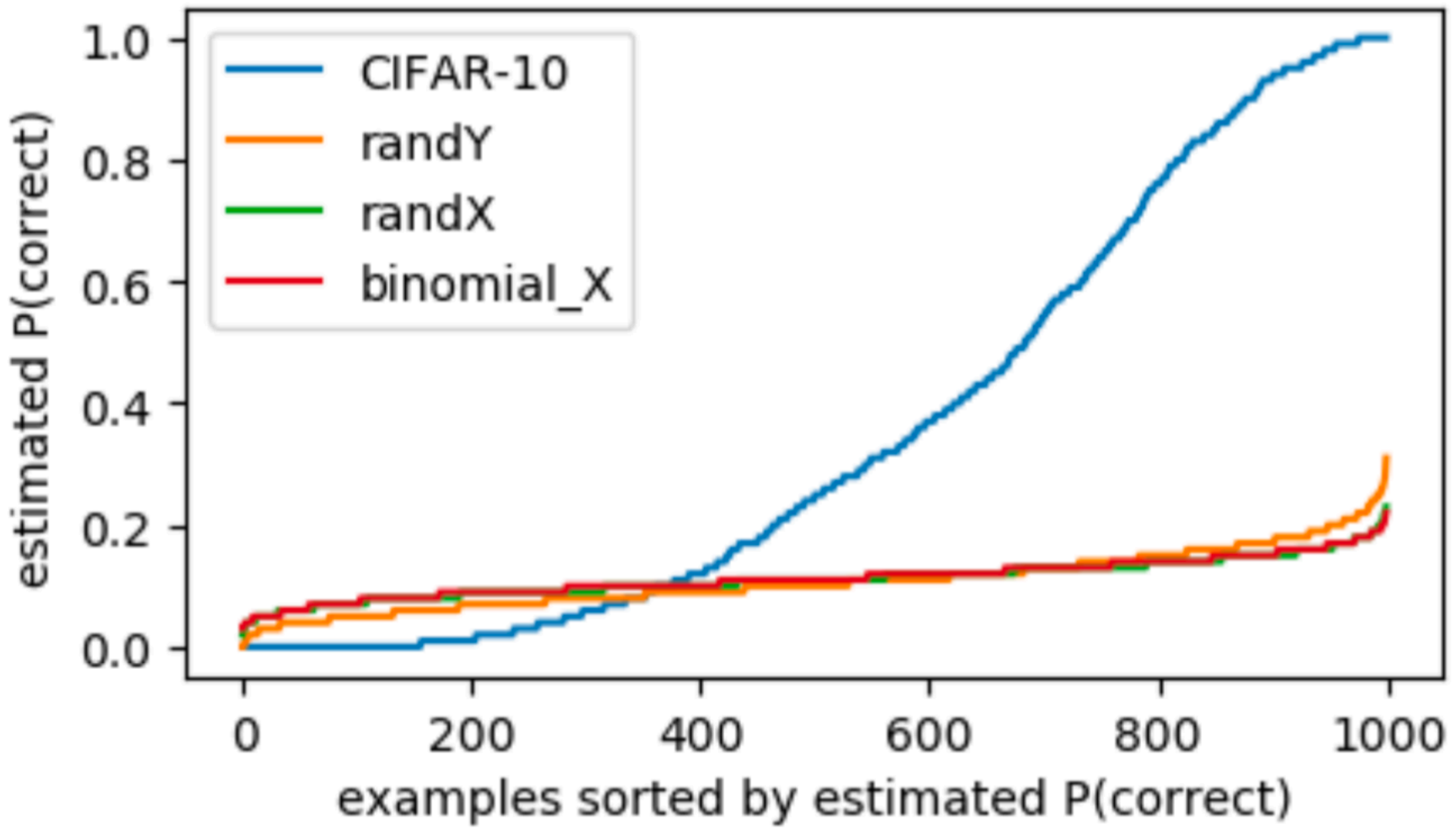


Figure 1. Average (over 100 experiments) misclassification rate for each of 1000 examples after one epoch of training. This measure of an example's difficulty is much more variable in real data.

1. Qualitative differences in fitting noise vs. real data

- Easy Examples as Evidence of Patterns in Real Data
  - A brute-force memorization approach to fitting data should apply equally well to different training examples.
  - If a network is learning based on patterns in the data, some examples may fit these patterns better than others.

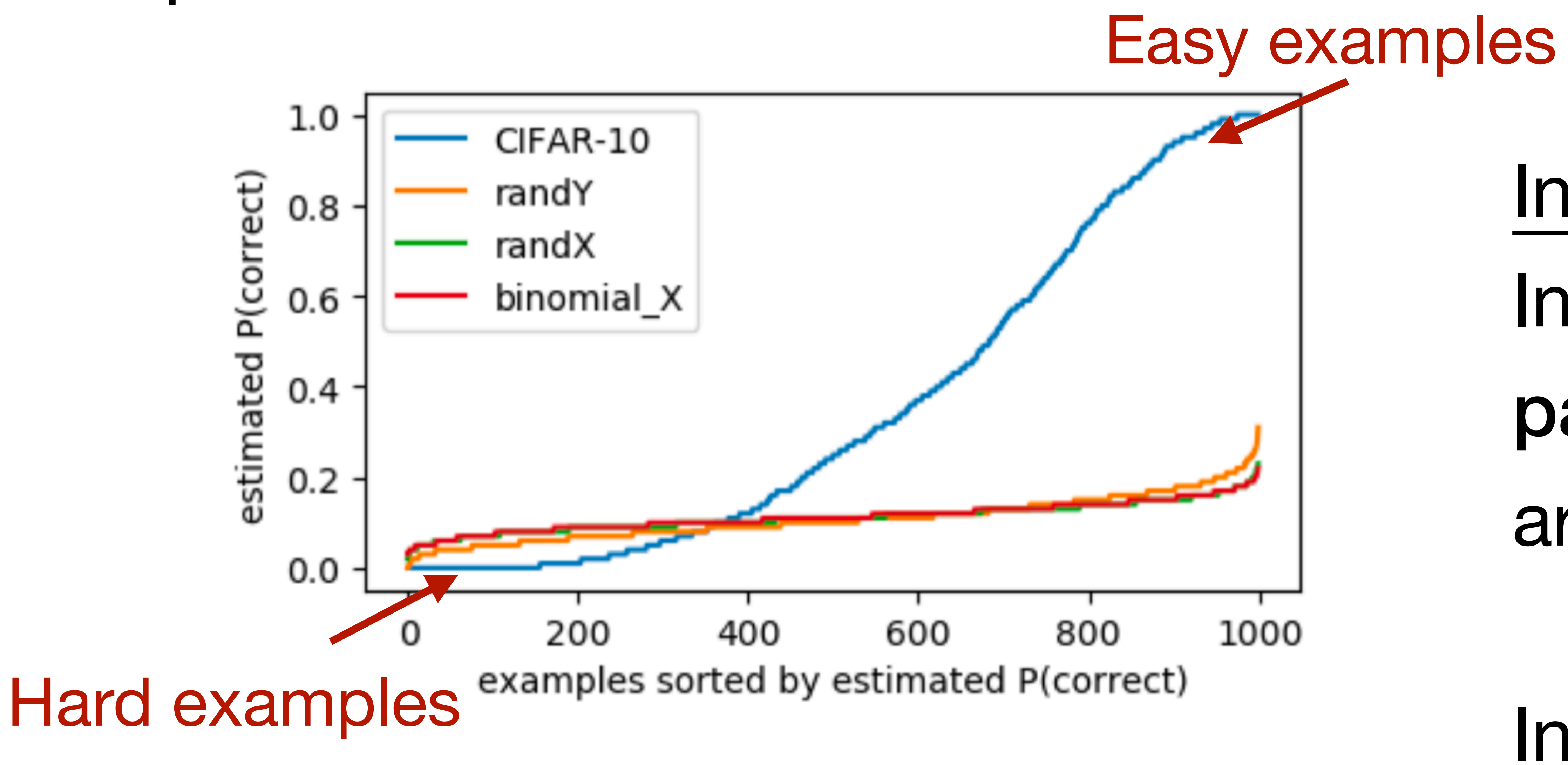


Figure 1. Average (over 100 experiments) misclassification rate for each of 1000 examples after one epoch of training. This measure of an example's difficulty is much more variable in real data.

Interpretation:

In real data, easy examples match **underlying patterns** of the data distribution; hard examples are **exceptions** to the patterns.

In random data, examples are equally hard: learning is **content agnostic.**

1. Qualitative differences in fitting noise vs. real data

- Loss Sensitivity in Real vs. Random Data
  - Cannot measure quantitatively how much each training sample x is memorized.
  - Instead, measure the effect of each sample on the average loss.

$$\bar{g}_x = \frac{\sum_t \| \partial L / \partial x \|_1}{t}$$

1. Qualitative differences in fitting noise vs. real data

- Loss Sensitivity in Real vs. Random Data
  - Cannot measure quantitatively how much each training sample x is memorized.
  - Instead, measure the effect of each sample on the average loss.

$$\bar{g}_x = \frac{\sum_t \| \partial L / \partial x \|_1}{t}$$

- Observations:
  - For real data, only a subset of the training set has high $\bar{g}_x$.
  - For random data, $\bar{g}_x$ is high for virtually all examples.

1. Qualitative differences in fitting noise vs. real data

- Loss Sensitivity in Real vs. Random Data
  - Cannot measure quantitatively how much each training sample x is memorized.
  - Instead, measure the effect of each sample on the average loss.

$$\bar{g}_x = \frac{\sum_t \| \partial L / \partial x \|_1}{t}$$

- Observations:
  - For real data, only a subset of the training set has high $\bar{g}_x$.
  - For random data, $\bar{g}_x$ is high for virtually all examples.

When being trained on real data, the neural network probably ***does not memorize***, or at least ***not in the same manner*** it needs to for random data.

1. Qualitative differences in fitting noise vs. real data

- Per-Class Loss Sensitivity in Real vs. Random Data
  - Measure the effect of examples of class i on the class j.
  -

$$\bar{g}_{i,j} = \frac{\sum_t \left\| \partial L(y=i) / \partial x_{y=j} \right\|_1}{t}$$

1. Qualitative differences in fitting noise vs. real data

- Per-Class Loss Sensitivity in Real vs. Random Data
  - Measure the effect of examples of class i on the class j.

$$\bar{g}_{i,j} = \frac{\sum_t \left\| \partial L(y=i)/\partial x_{y=j} \right\|_1}{t}$$

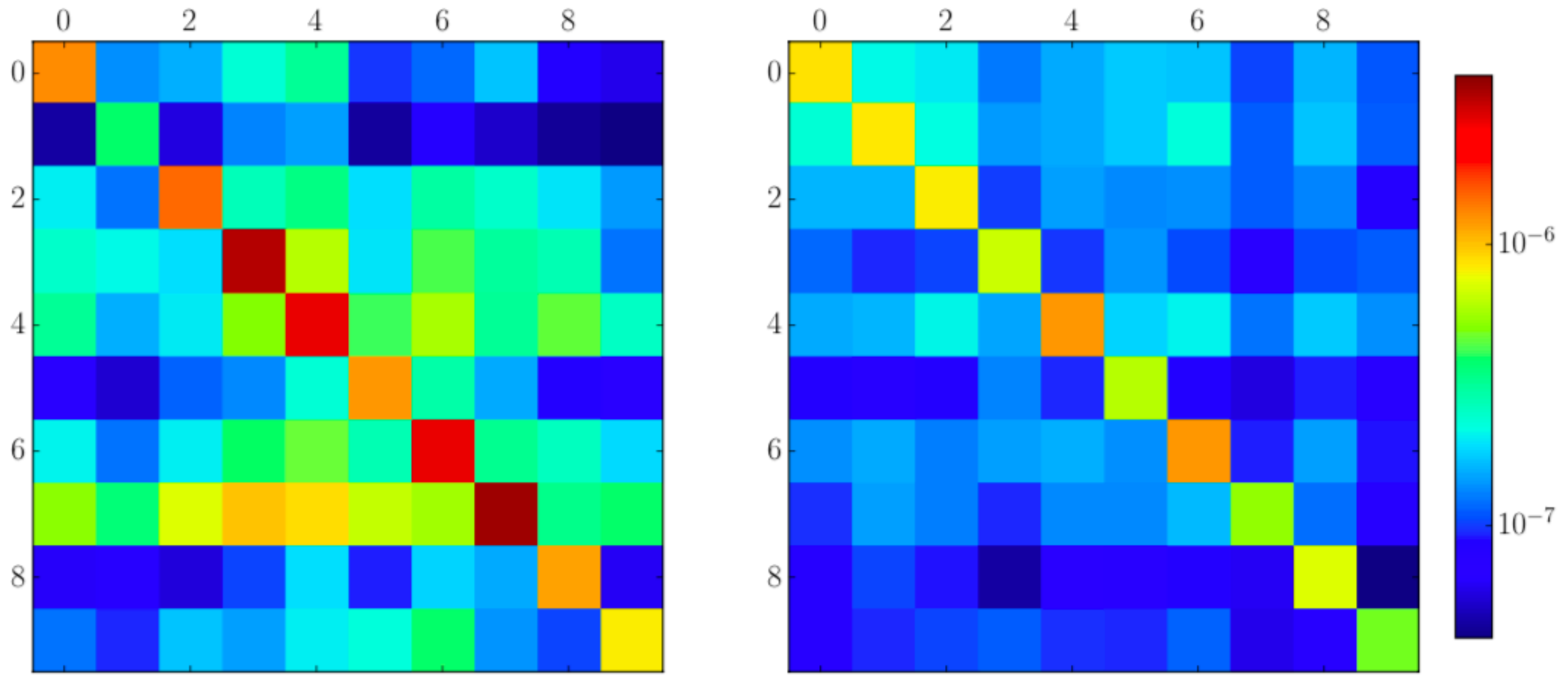


Figure 4. Plots of per-class g_x (see previous figure; log scale), a cell i, j represents the average $|\partial\mathcal{L}(y=i)/\partial x_{y=j}|$, i.e. the loss-sensitivity of examples of class i w.r.t. training examples of class j. Left is real data, right is random data.

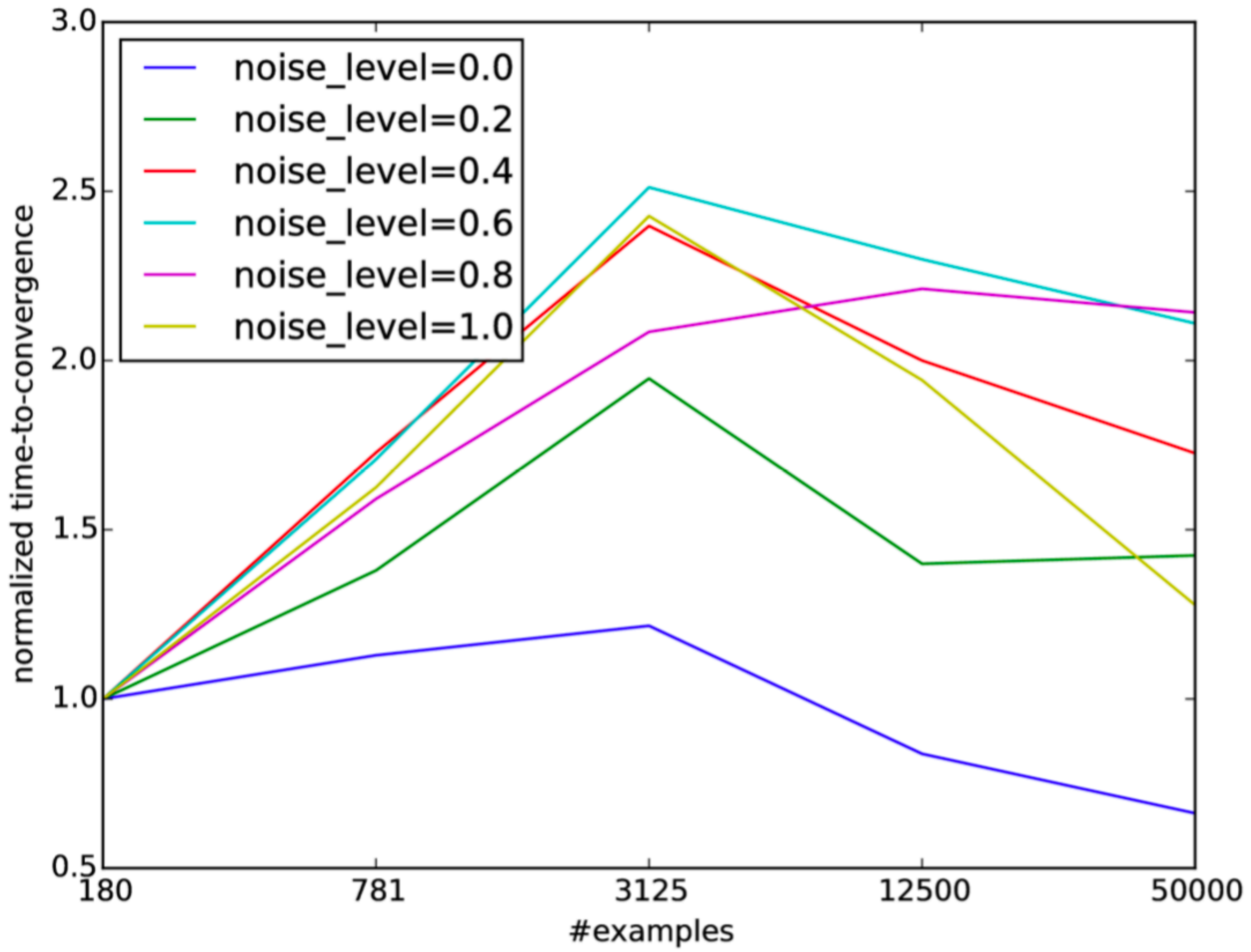
Interpretation:

In real data, more patterns (e.g. low-level features) are shared across classes.

(*This is a selling-point of deep distributed representations*)

1. Qualitative differences in fitting noise vs. real data

- Time-to-Convergence on Real vs. Random Data
  - With a limited model capacity increasing the number of examples will increase the time needed to memorize the training set.



Interpretation:
Fitting more real data examples is easier because they follow meaningful patterns.

2. DNNs learn simple patterns first

- Complexity of the hypotheses learned by DNNs on Real vs. Random Data
  - A smaller fraction of points in the proximity of a decision boundary suggests that the learned hypothesis is simpler.
  - Critical sample ratio (CSR): how many data-points have an adversarial example nearby?

$$\arg\max_i f_i(\mathbf{x}) \neq \arg\max_j f_j(\hat{\mathbf{x}})$$
$$\text{s.t. } \|\mathbf{x} - \hat{\mathbf{x}}\|_\infty \leq r$$

2. DNNs learn simple patterns first

- Complexity of the hypotheses learned by DNNs on Real vs. Random Data
  - A smaller fraction of points in the proximity of a decision boundary suggests that the learned hypothesis is simpler.
  - Critical sample ratio (CSR): how many data-points have an adversarial example nearby?

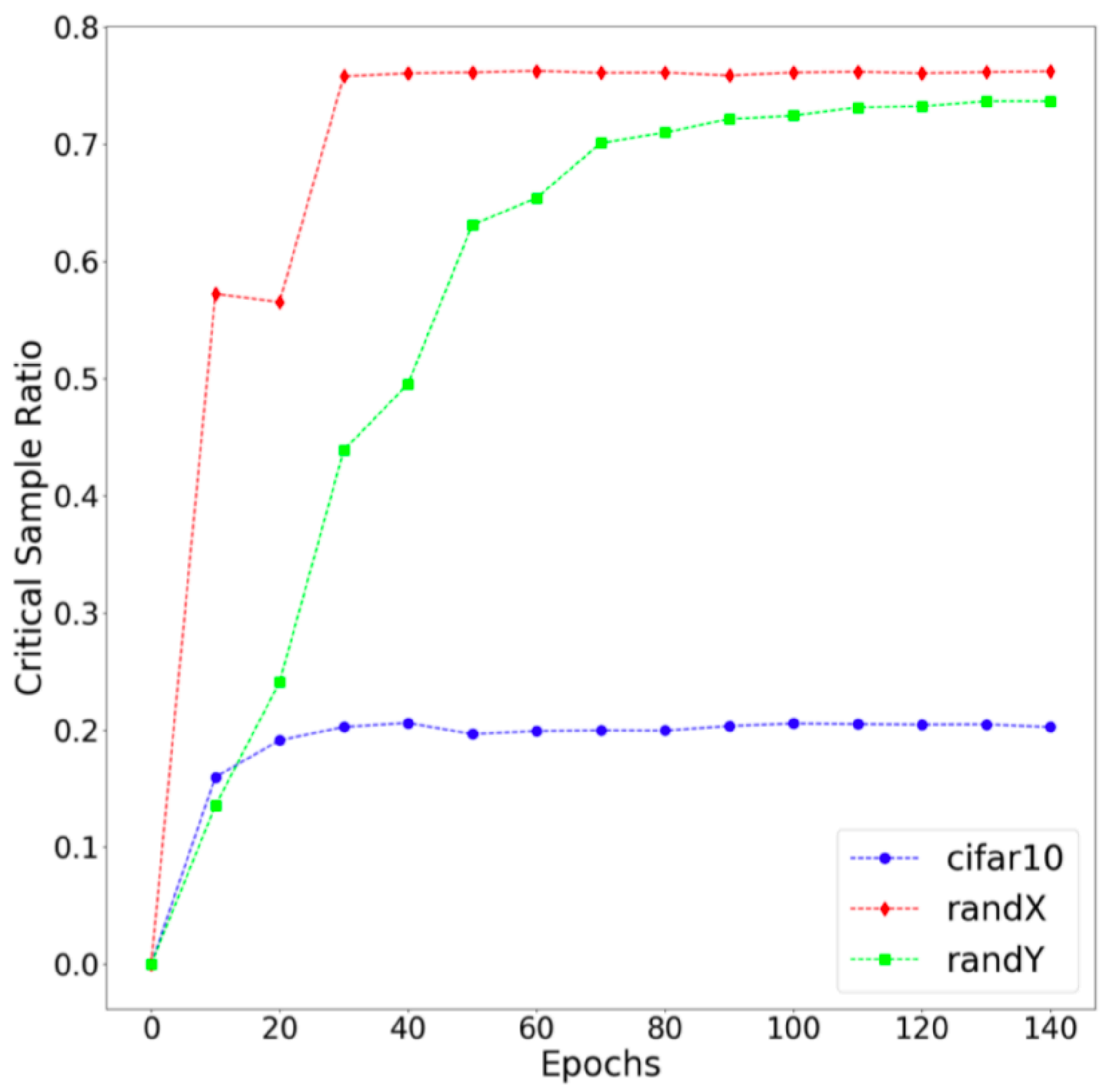


Figure 9. Critical sample ratio throughout training on CIFAR-10, random input (randX), and random label (randY) datasets.

$$\arg\max_i f_i(\mathbf{x}) \neq \arg\max_j f_j(\hat{\mathbf{x}})$$

$$\text{s.t. } \|\mathbf{x} - \hat{\mathbf{x}}\|_\infty \leq r$$

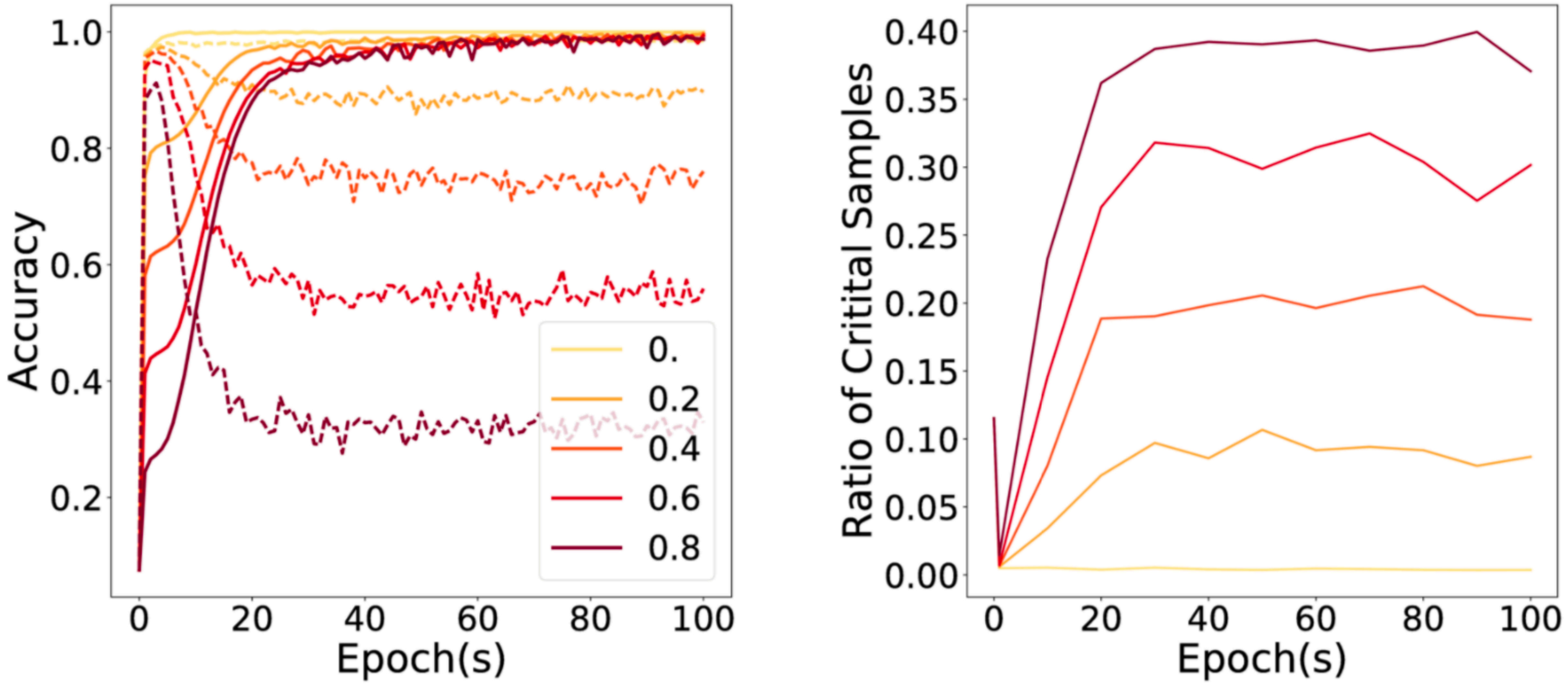
Interpretation:
Learned hypotheses are less complex for real data.

2. DNNs learn simple patterns first

- Critical Samples Throughout Training in Partially Noisy Data
  - Network achieves maximum accuracy on the validation set before achieving high accuracy on the training set.
  - As the model moves from fitting real data to fitting noise, the CSR greatly increases, indicating the need for more complex hypotheses to explain the noise.

2. DNNs learn simple patterns first

- Critical Samples Throughout Training in Partially Noisy Data
  - Network achieves maximum accuracy on the validation set before achieving high accuracy on the training set.
  - As the model moves from fitting real data to fitting noise, the CSR greatly increases, indicating the need for more complex hypotheses to explain the noise.



(b) Noise added on classification labels.

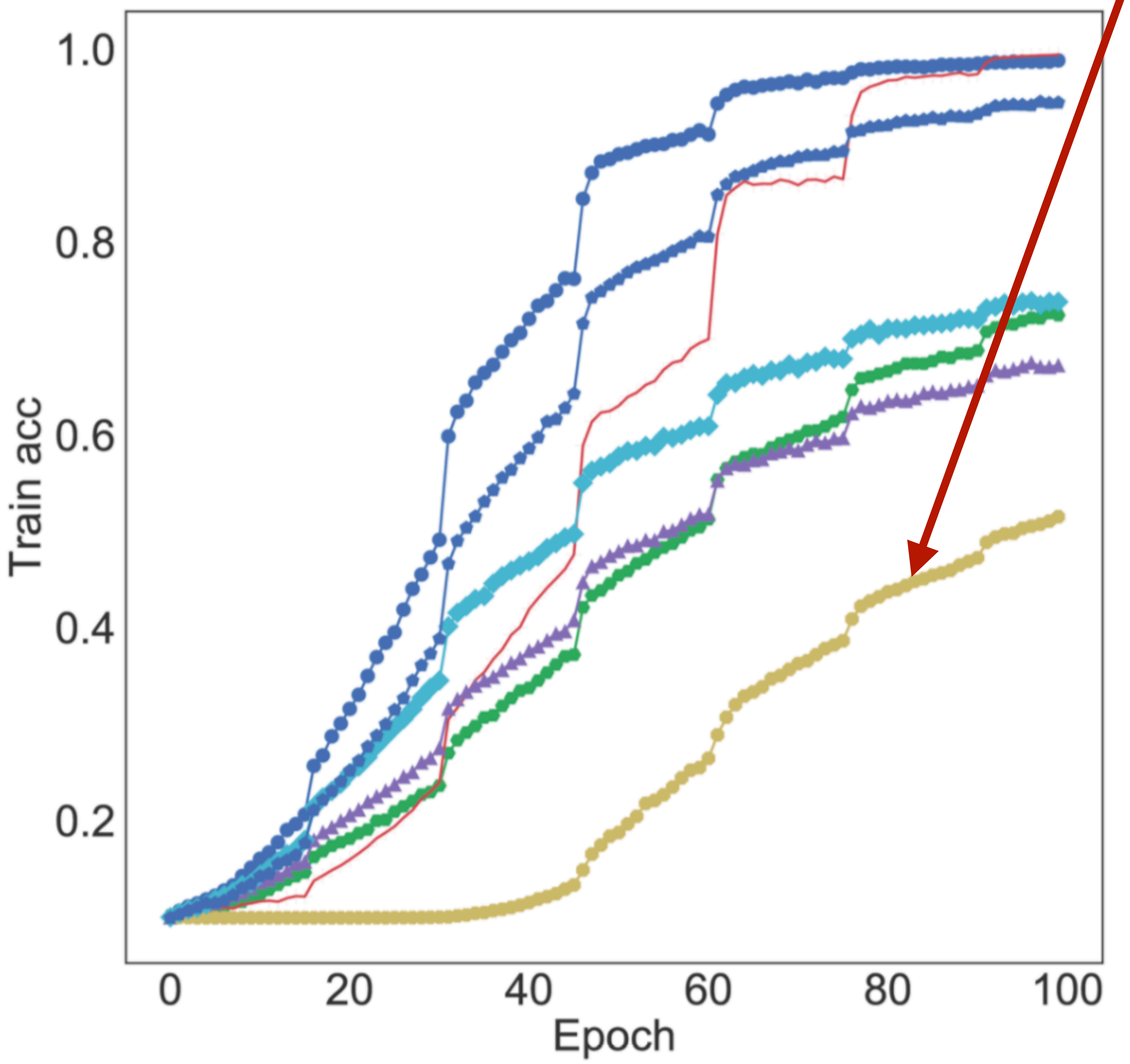
Interpretation:
DNNs fit data-points (which follow patterns) before fitting noise (which results in decreasing validation accuracy).

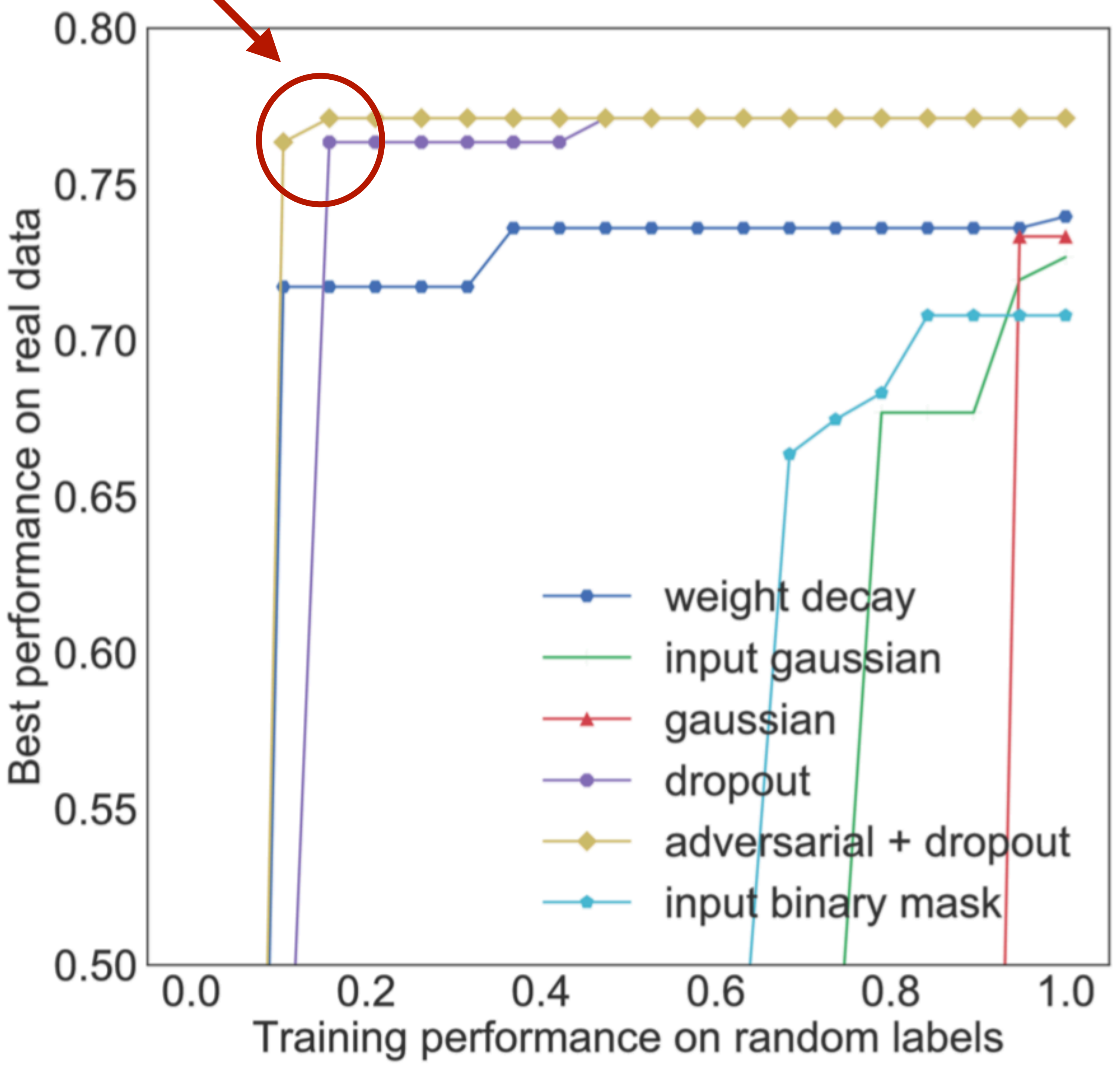
3. Regularization can reduce memorization

- Explicit Regularization on models trained on real data + randY
  - To limit the speed of memorization of noise data without significantly impacting learning on real data.

Achieve a high accuracy on real data with low memorization.

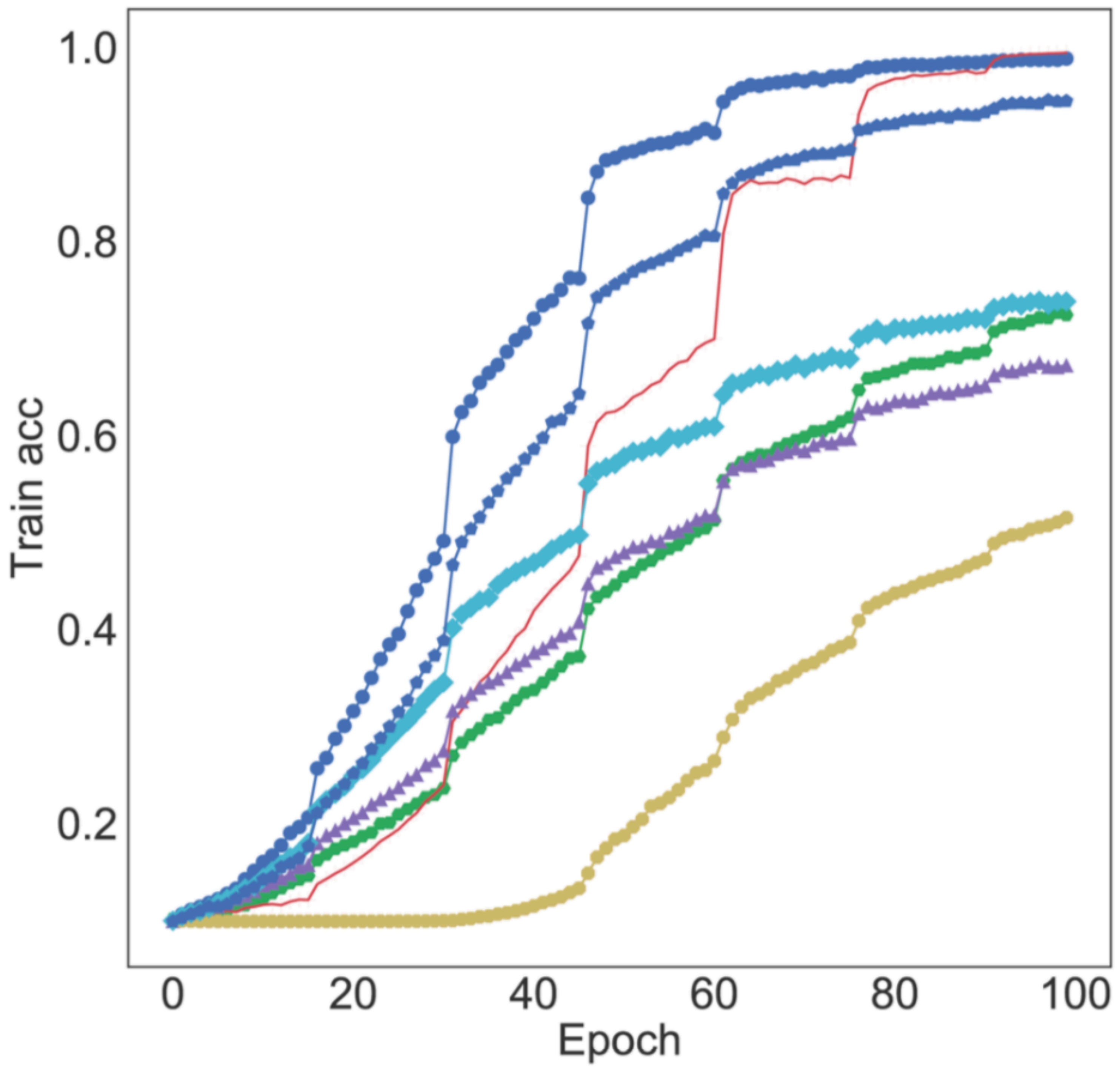
randY

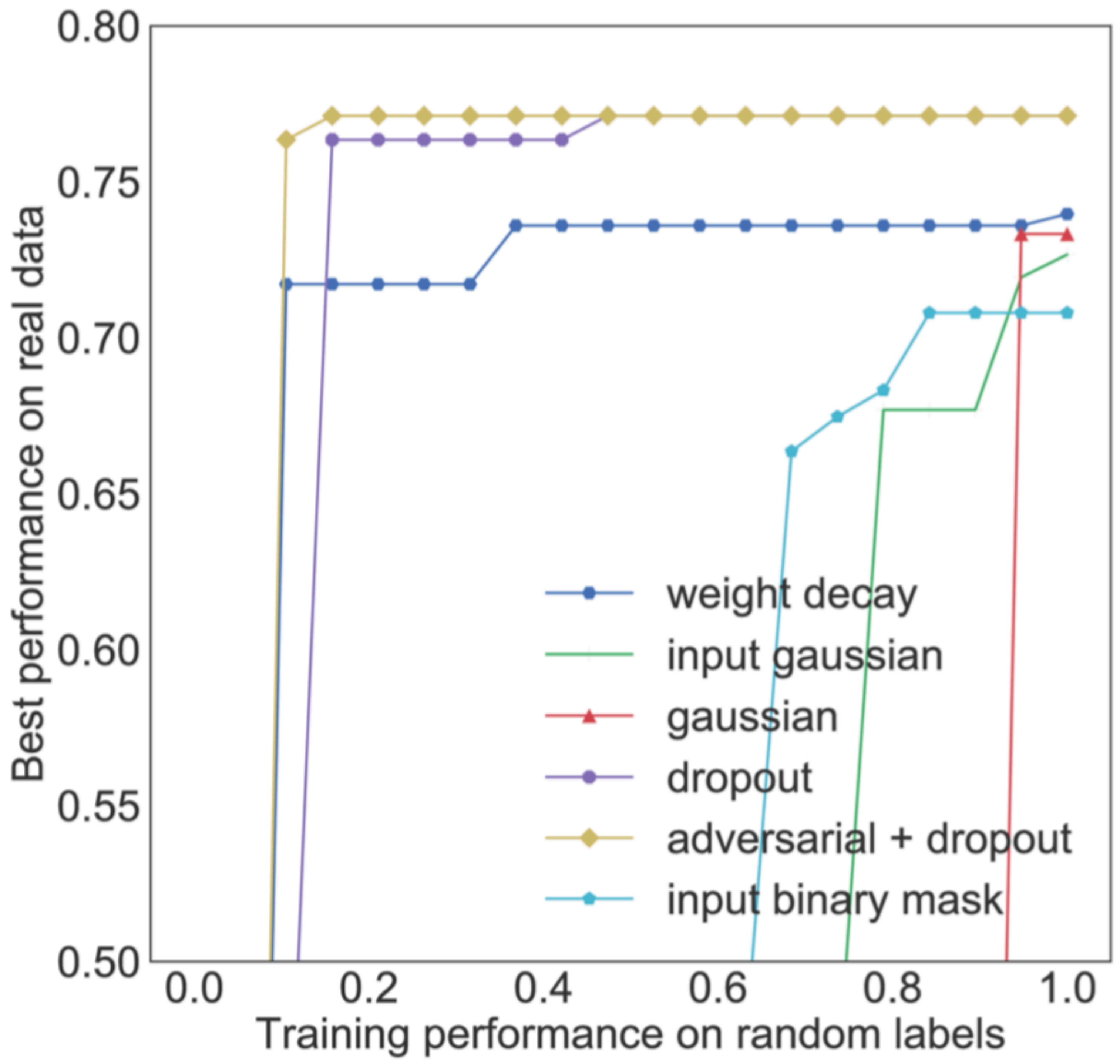




3. Regularization can reduce memorization

- Explicit Regularization on models trained on real data + randY
  - To limit the speed of memorization of noise data without significantly impacting learning on real data.





Interpretation:
We can severely limit memorization without hurting learning!

Adversarial training (+dropout) is particularly effective, supporting use of critical sample ratio to measure complexity

Summary

- DNNs do not just memorize real data as they do for random data.
- DNNs learn simple patterns first, before memorizing.
- Regularization techniques can differentially hinder memorization in DNNs while preserving their ability to learn about real data.

Do deep neural networks learn shallow learnable examples first?

Karttikeya Mangalam and Vinay Prabhu
ICML 2019 Workshop

Main Questions

- Is shallow learnability a good proxy for the easiness of an example?
  - When training DNNs, do we observe a shallow learnable to deep learnable regime change?
  - Are there examples that are shallow learnable but for some reason a DNN with a far better overall accuracy fails to classify?

Experiments

- Contingency matrix: M - Shallow models, D- Deep models

	M incorrect	**M correct**
D incorrect	T_{00}	T_{01}
D correct	T_{10}	T_{11}

- Accuracy of models M and D:

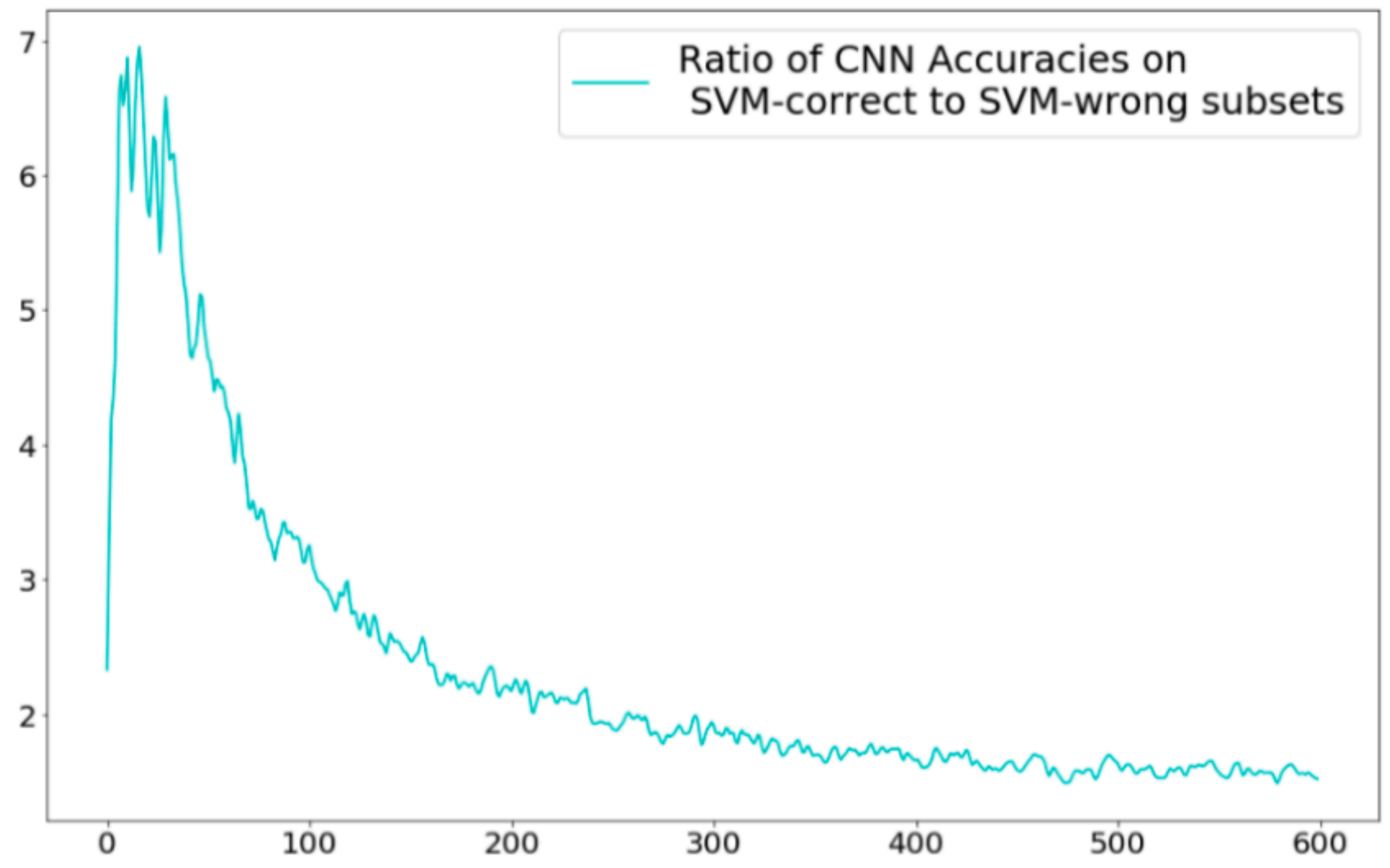
$$\text{Accuracy(M)} = \frac{T_{01} + T_{11}}{T_{11} + T_{00} + T_{10} + T_{01}} \qquad \text{Accuracy(D)} = \frac{T_{10} + T_{11}}{T_{11} + T_{00} + T_{10} + T_{01}}$$

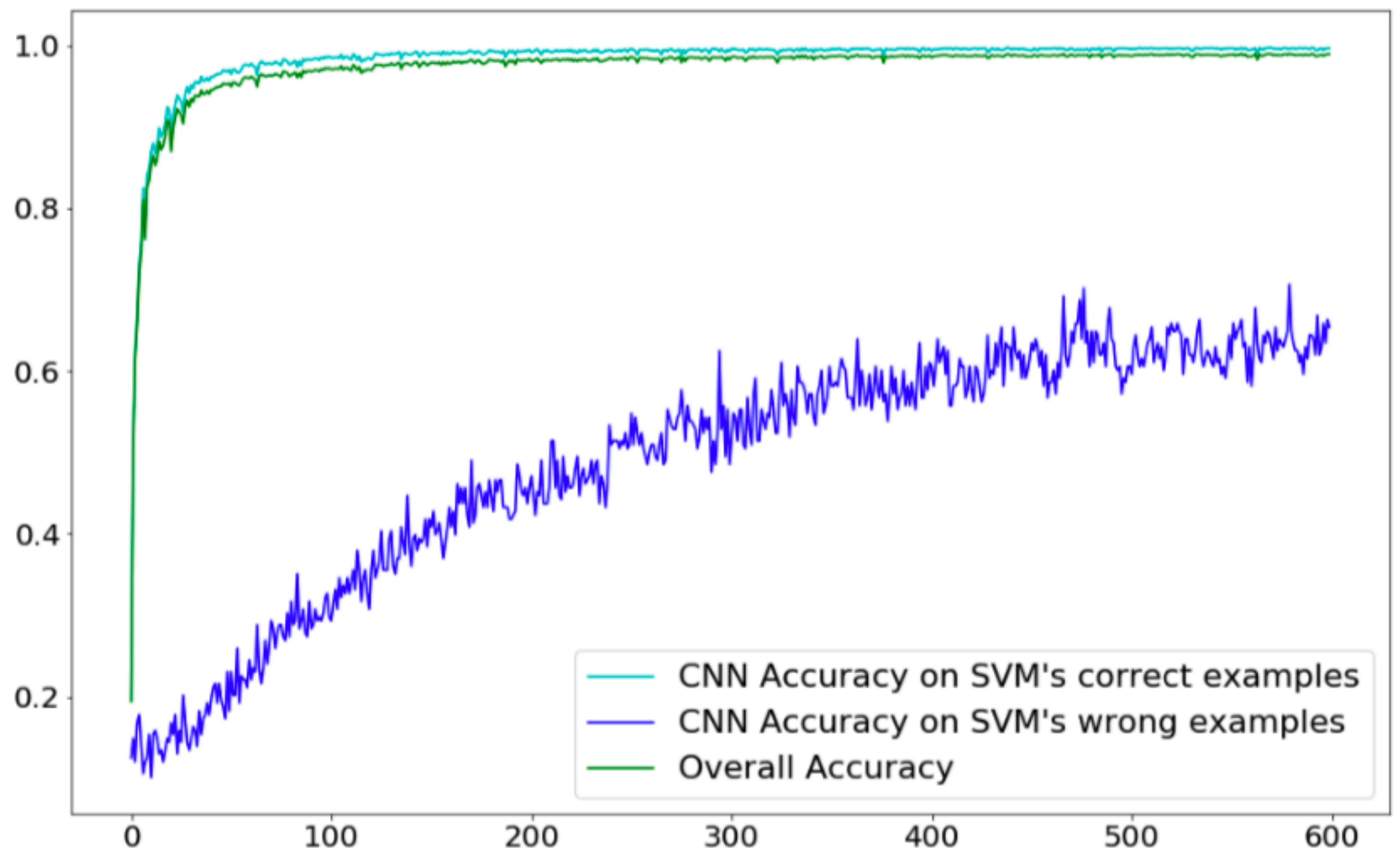
- (Marginal) Accuracy of D on subsets that M classifies correct (R_+) and incorrect (R_-):

$$R_+ = \frac{T_{11}}{T_{11} + T_{01}} \qquad R_+ = \frac{T_{10}}{T_{10} + T_{00}} \qquad R_\pm = \frac{R_+}{R_-}$$

Shallow learnable to deep learnable

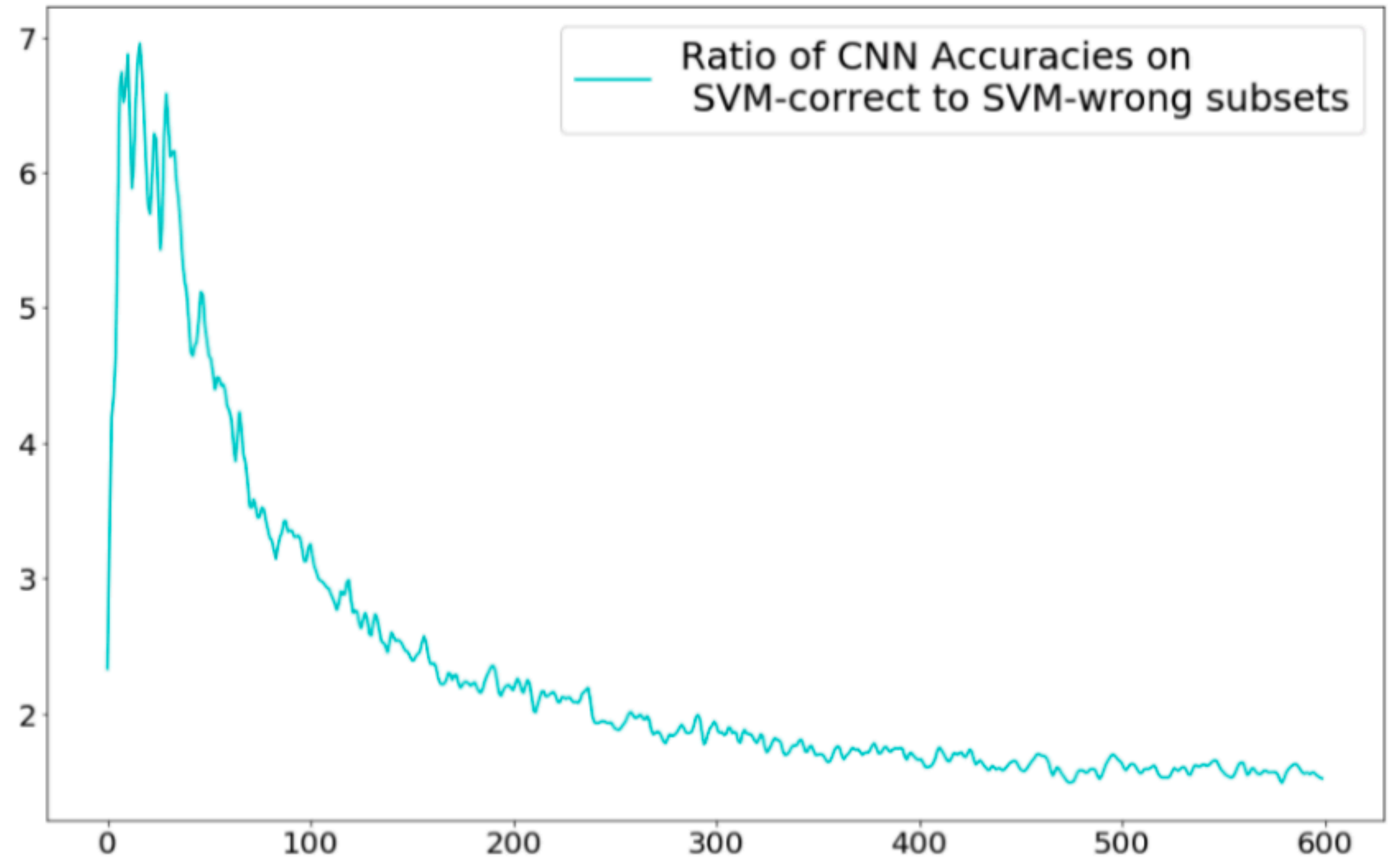
- If M+ and M- are completely irrelevant and similar for training process of D, $R_{\pm}$ should remain identically 1.

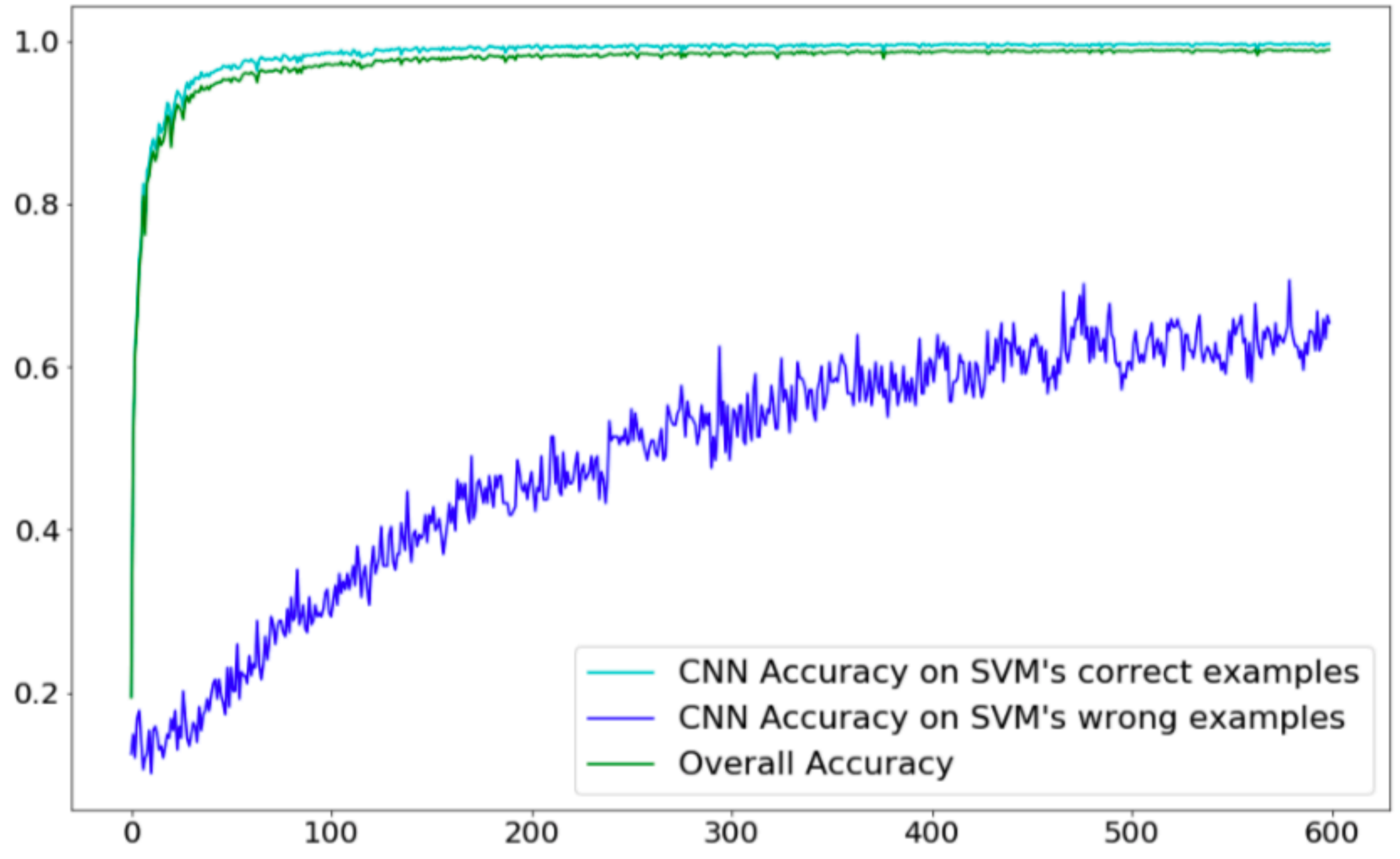




Shallow learnable to deep learnable

- If M+ and M- are completely irrelevant and similar for training process of D, $R_{\pm}$ should remain identically 1.



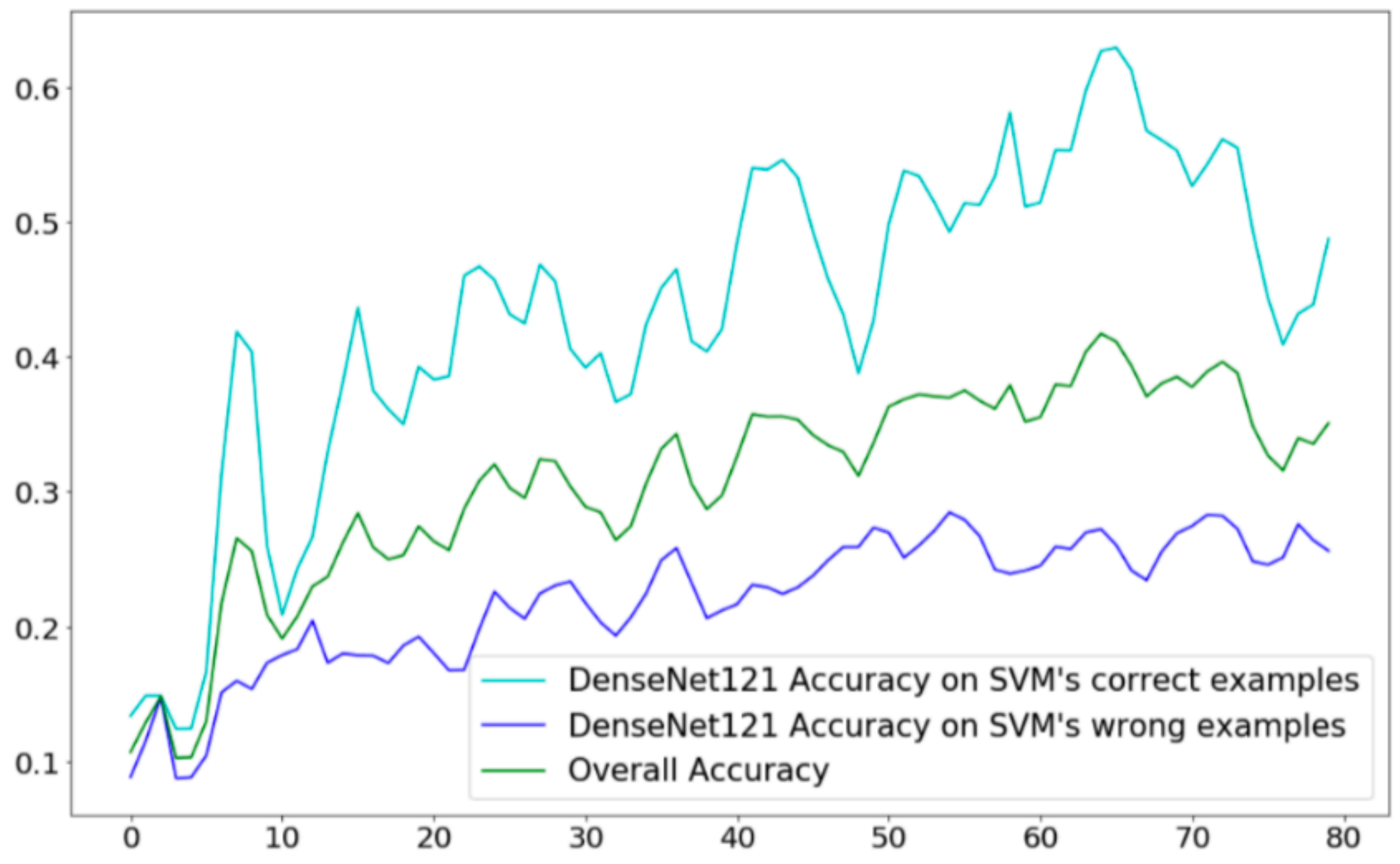


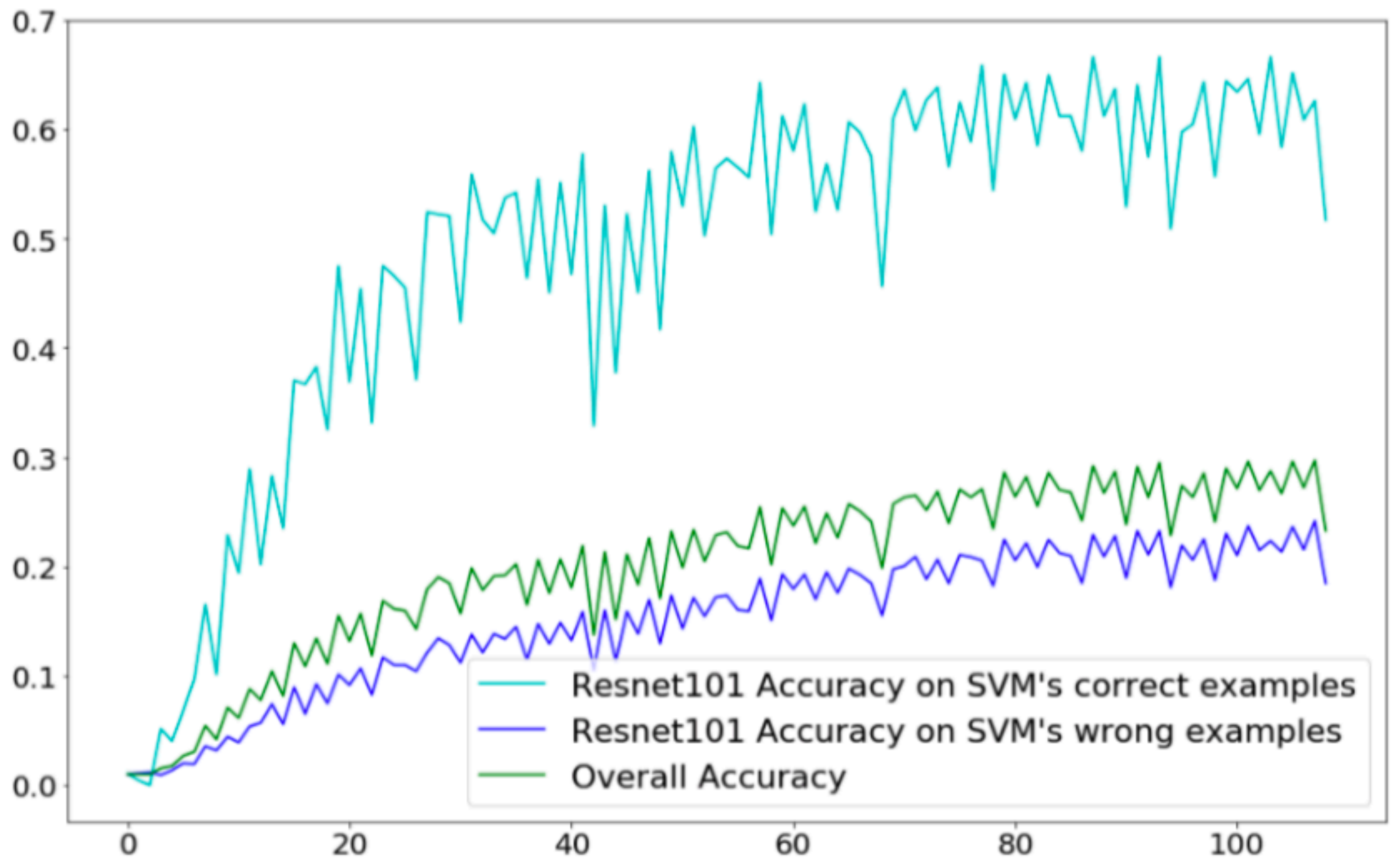
<u>Interpretation</u>:

DNNs training starts from quickly learning shallow classifiable *easy* examples and then slowly extends to the hard ones.

Shallow learnable but not deep learnable

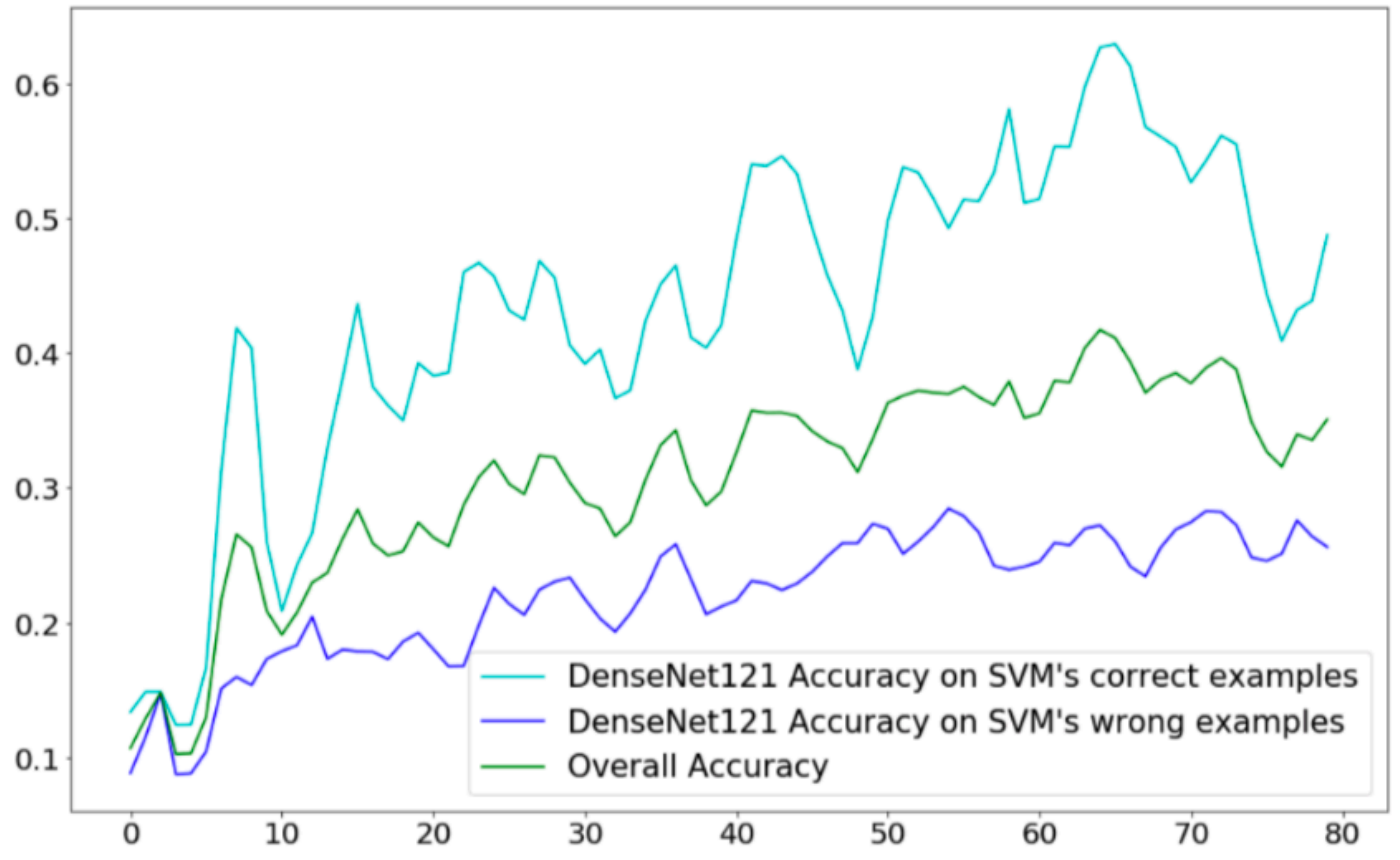
- Even after convergence, T_{01} is non-zero (i.e. there exists examples that M classifies correctly but D gets wrong).

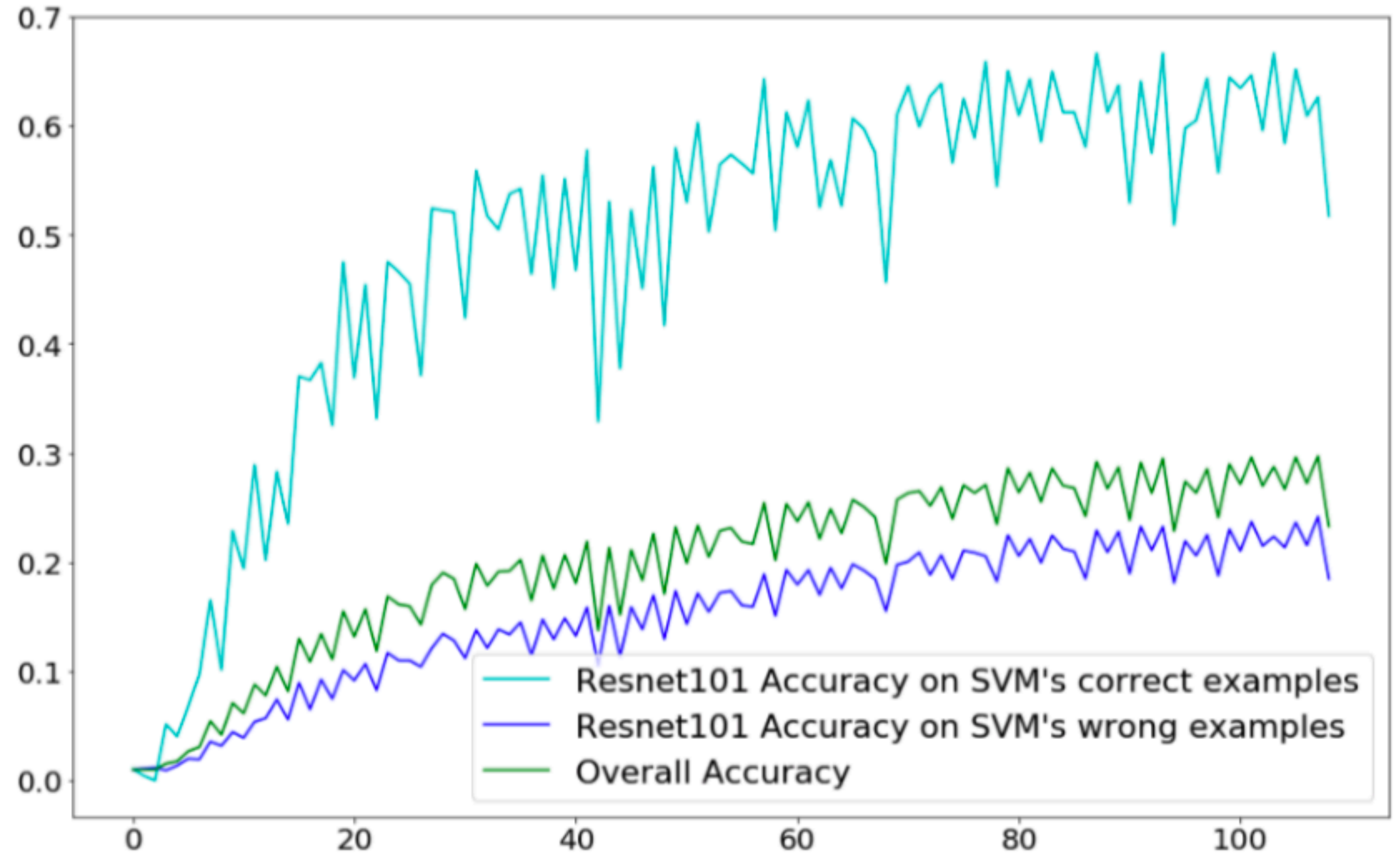




Shallow learnable but not deep learnable

- Even after convergence, T_{01} is non-zero (i.e. there exists examples that M classifies correctly but D gets wrong).





<u>Interpretation</u>:

The architecture difference between M and D may be the reason.

Summary

- Shallow models could be used for identifying easy examples in the training set.
- There could be examples that can be correctly classified by shallow models, but surprisingly cannot be correctly classified by deep models.