Improving Domain-specific Transfer Learning Applications for Image Recognition and Differential Equations

M.Sc. Thesis in Computer Science and Engineering

Candidates: Alessandro Saverio Paticchio, Tommaso Scarlatti

Advisor: Prof. Marco Brambilla – Politecnico di Milano
Co-advisor: Prof. Pavlos Protopapas – Harvard University

Agenda

INTRODUCTION

IMAGE RECOGNITION

$$\frac{\partial z}{\partial t}$$

DIFFERENTIAL EQUATIONS

CONCLUSIONS

Agenda

INTRODUCTION

IMAGE RECOGNITION

$$\frac{\partial z}{\partial t}$$

DIFFERENTIAL EQUATIONS

CONCLUSIONS

Context

Deep neural networks have become an indispensable tool for a wide range of applications.

They are extremely ***data hungry*** models and often require a lot of computational resources.

Can we reduce the training time?

↓

Transfer Learning!

Transfer Learning

A typical approach is using a pre-trained model as a starting point.
[*S. Pan and Q. Yang – 2010*]

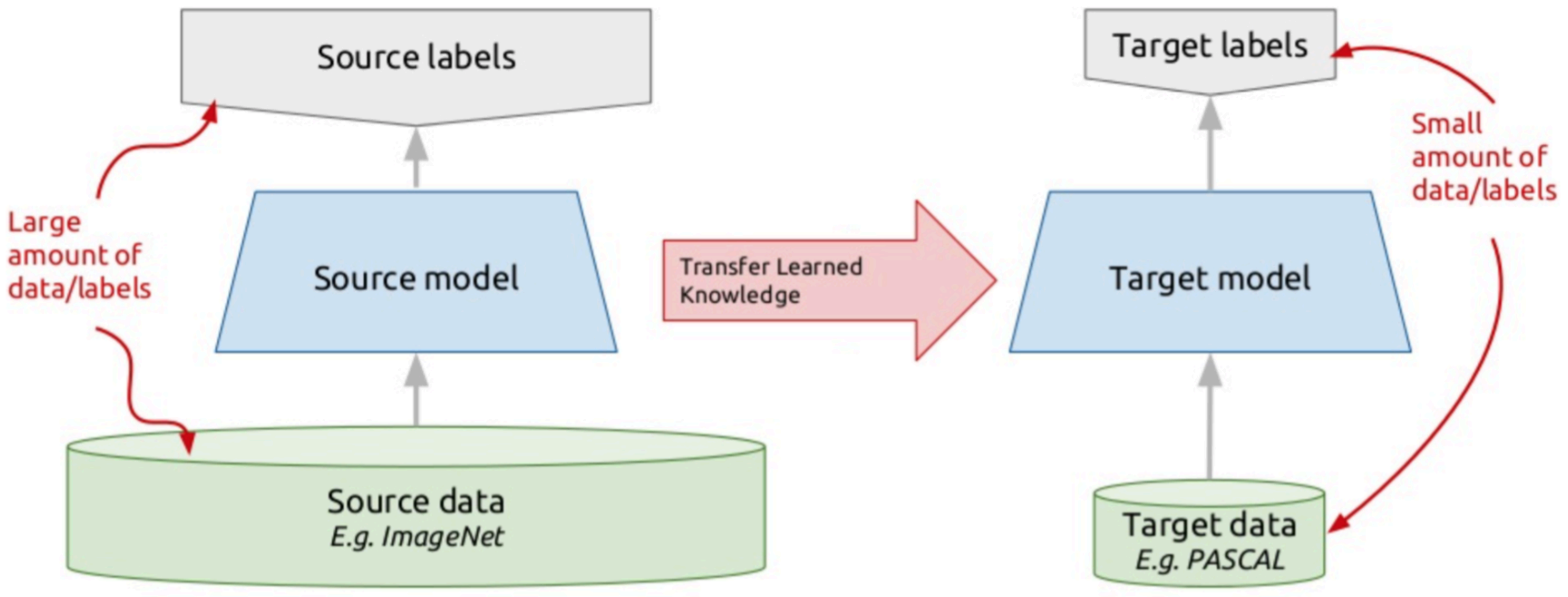
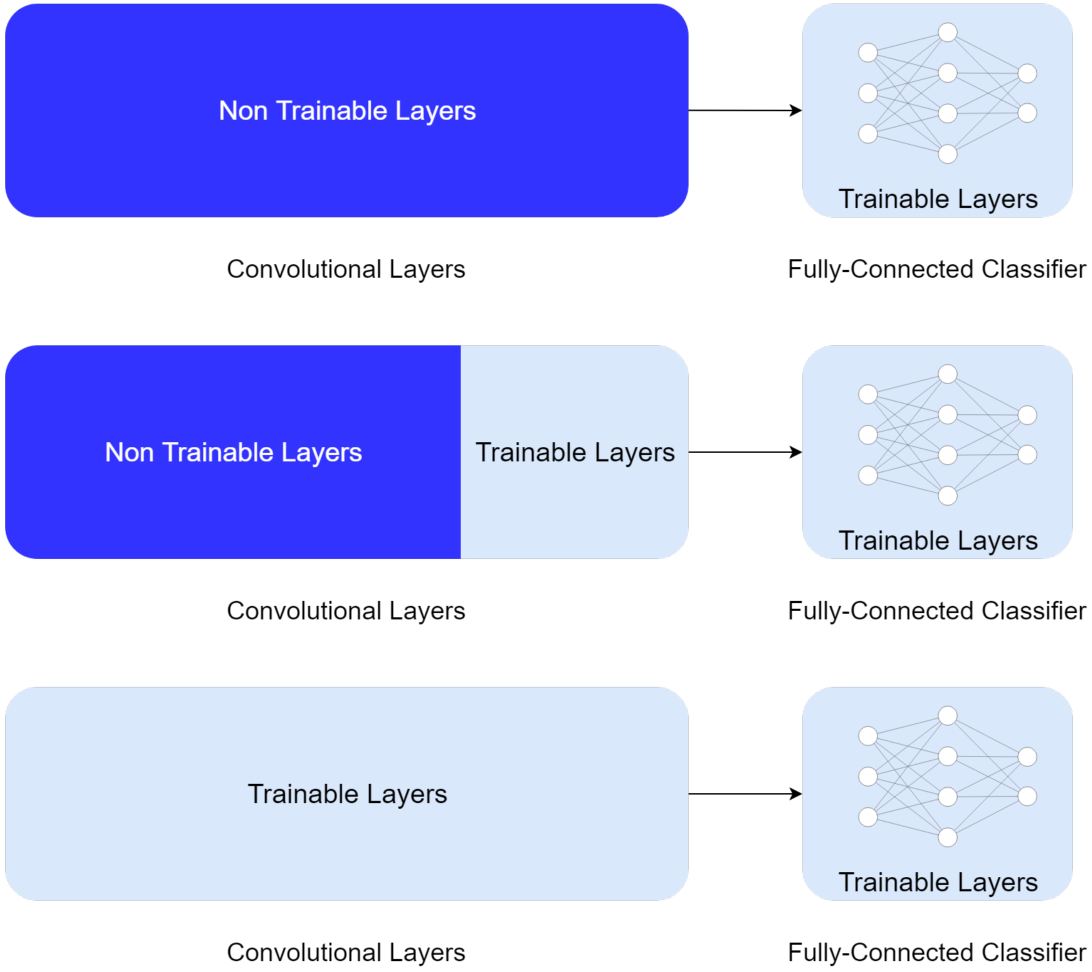


Image source: https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

Neural Networks Finetuning

- Use the weights of the pre-trained model as a starting point
- Many different variations depending on the architectures
- Layers can be frozen / finetuned



Problem statement

- Can we find **smarter techniques** to transfer the knowledge already acquired?
- Can we find a way to reduce further the computational footprint?
- Can we improve the **convergence** and the final **error** of our target model?

Proposed solution - Explore transfer learning techniques in two different scenarios:

- Image recognition
- Resolution of differential equations

Agenda

INTRODUCTION

IMAGE RECOGNITION

$$\frac{\partial z}{\partial t}$$

DIFFERENTIAL EQUATIONS

CONCLUSIONS

Image Recognition - Problem setting

It’s a **supervised** classification problem:

The model learns mapping from features x to a label y.

We analysed the problem of **covariate shift** [*Moreno-Torres et al. – 2012*], which can harm the performance of the target model:

$$P_s(y|x) = P_t(y|x)$$
$$P_s(x) \neq P_t(x)$$

Datasets and distortions

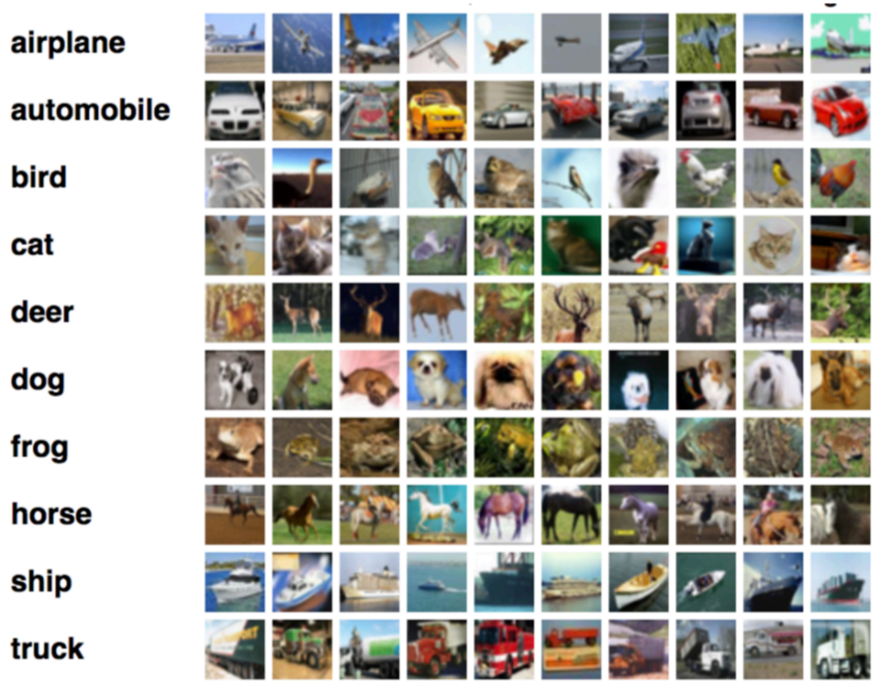
We used different types of datasets, shifts and architectures.

DATASETS

- **CIFAR-10**
- CIFAR-100
- **USPS**
- **MNIST**

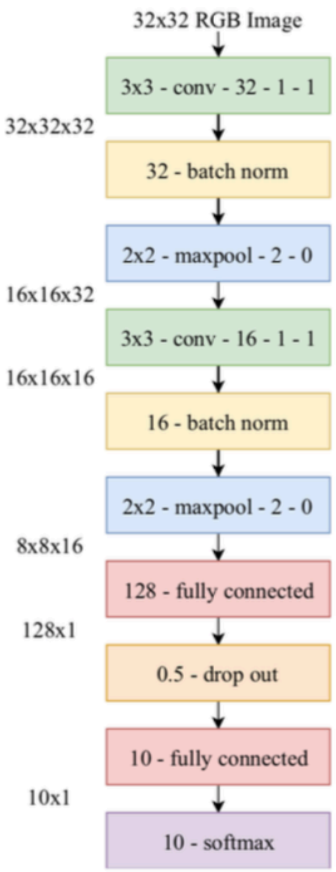
SHIFTS

- **Embedding Shift**
- Additive White Gaussian Noise
- Gaussian Blur

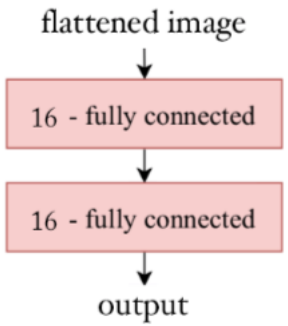


Samples images from the CIFAR-10 dataset

Architectures



Architecture for CIFAR-10 dataset



Architecture for MNIST and USPS datasets

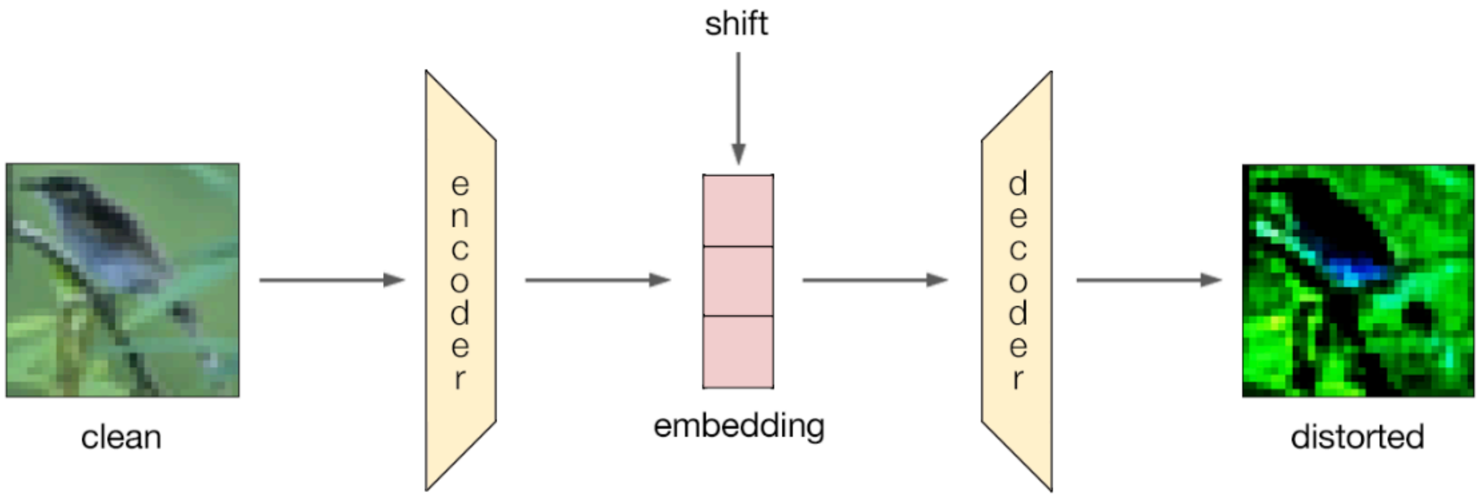
Presented scenarios

pretrained on MNIST → *finetuned on USPS*

pretrained on CIFAR-10 → *finetuned on CIFAR-10 with embedding shift*

Embedding shift

- Autoencoder learns a compressed representation of the input image called embedding;
- An additive shift is applied to each value of the embedding tensor.



Embedding shift (cont.)

- Examples of different levels of distortions applied;
- If $shift = 0$ we call it **plain** embedding shift.

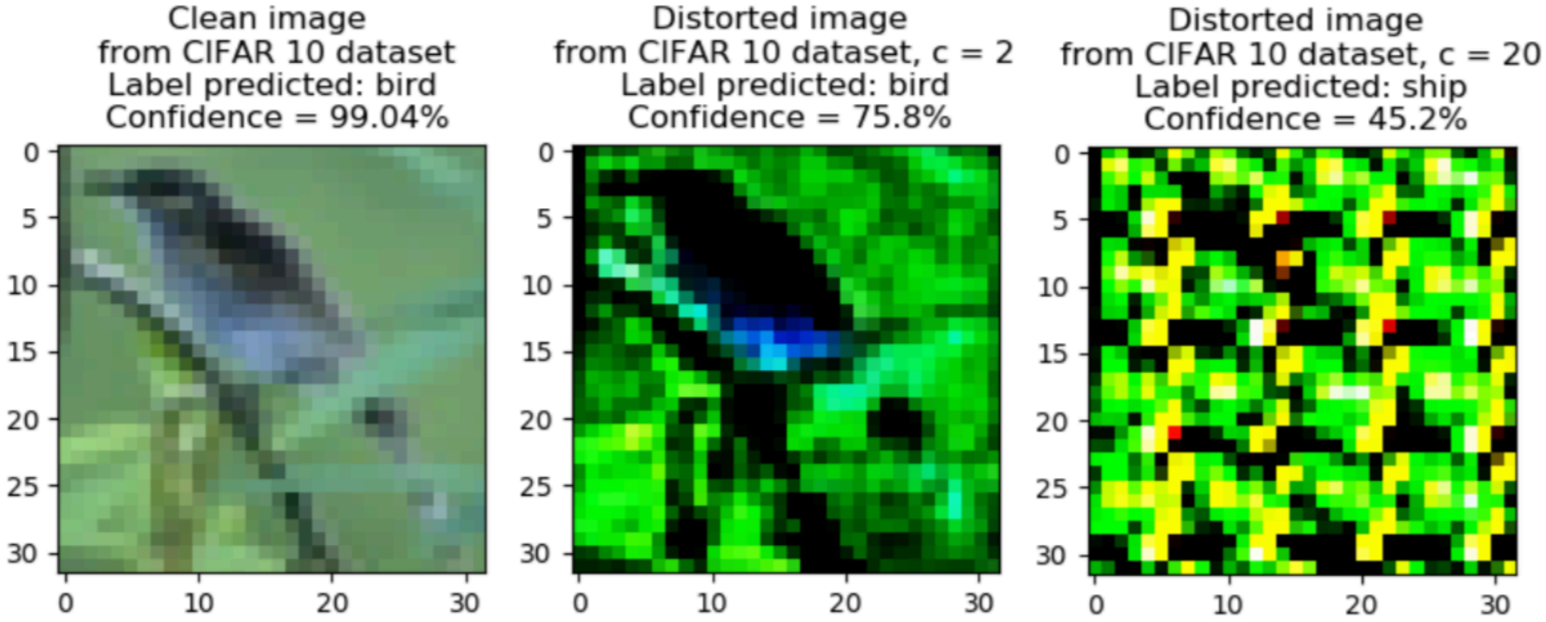


Image Recognition – Problem statement

We focused on the **data impact** in a transfer learning setting:

can we select a subset a subsample of D_t to improve finetuning?

We developed different selection criteria:

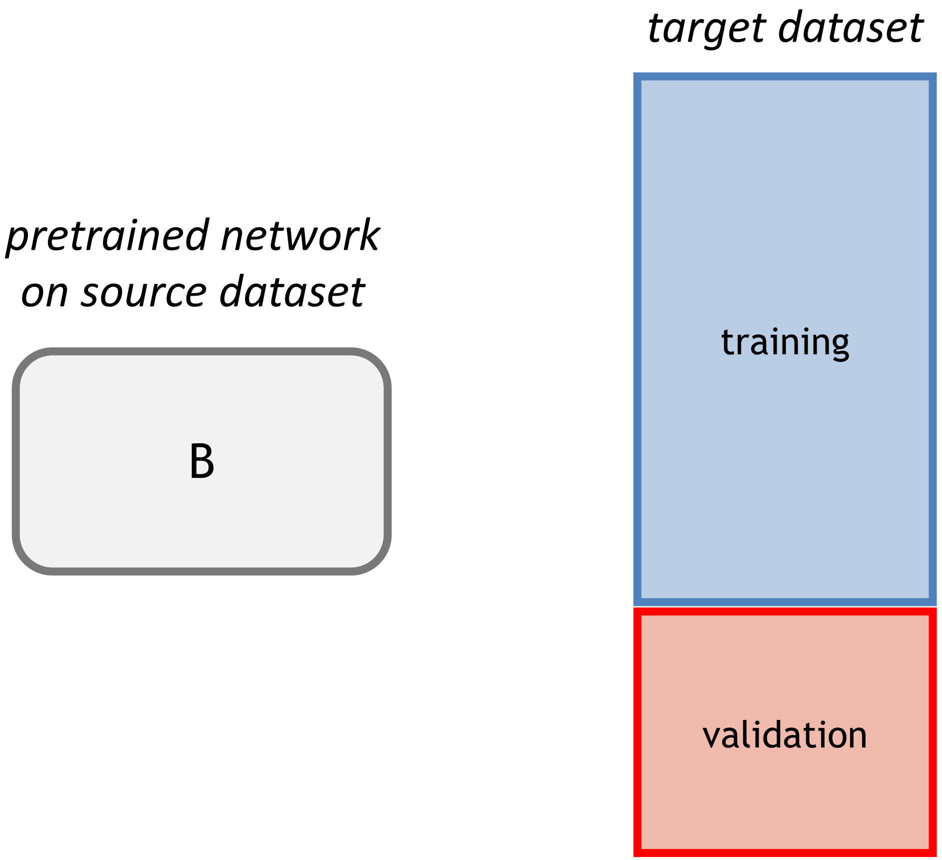
- Error-driven approach
- **Differential approach**
- **Entropy-driven approach**

Differential approach

input :

- B: pretrained network from source domain
- X_t: training set in target domain
- V_t: validation set in target domain

output : $X_t\prime$: optimized training set in target domain

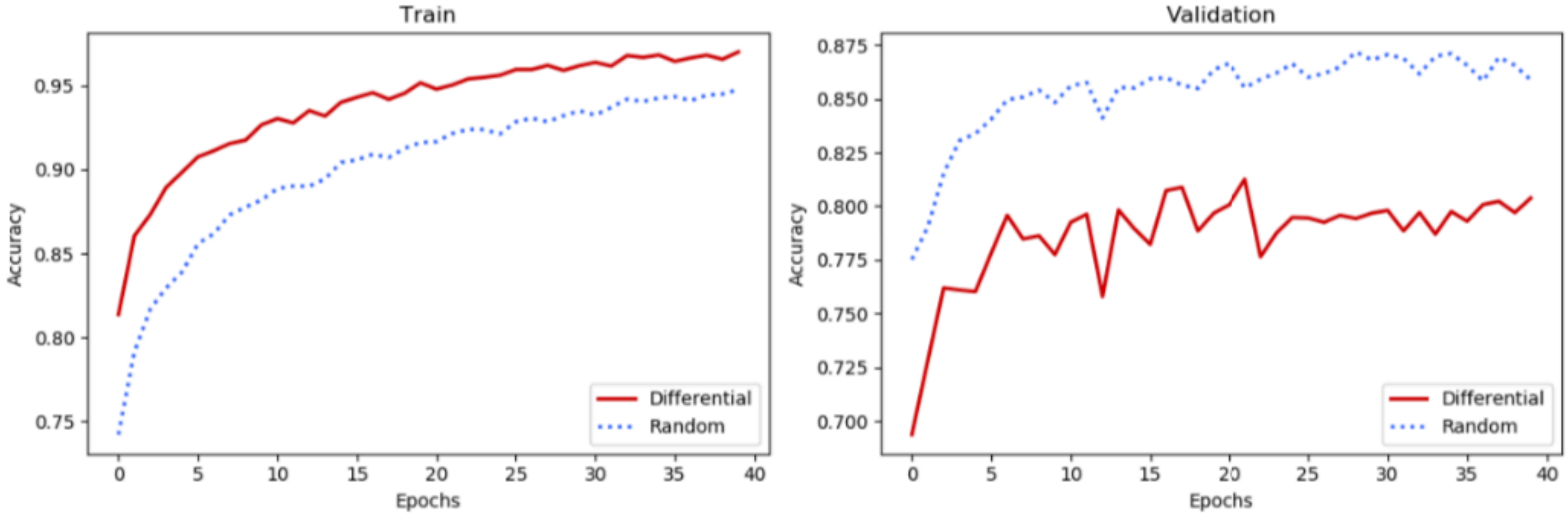


Differential approach – CIFAR-10

Leads to a result different from the expectations:

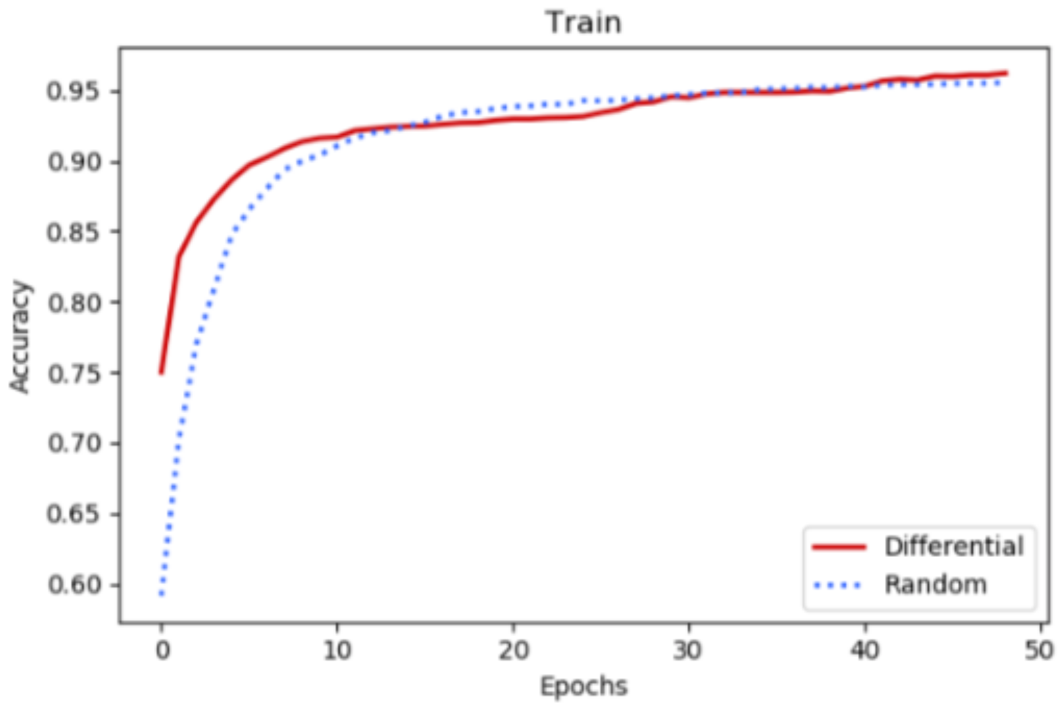
good performance on the train set, worse than random selection on the validation set.

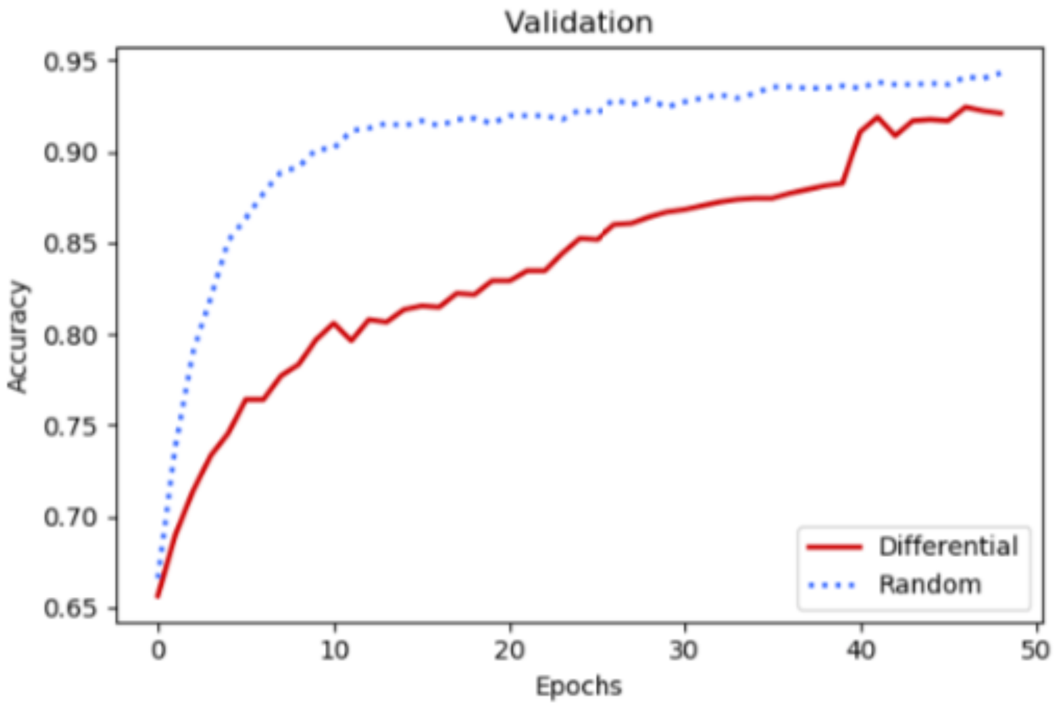
$$embedding\ shift = 2$$



Differential approach – USPS

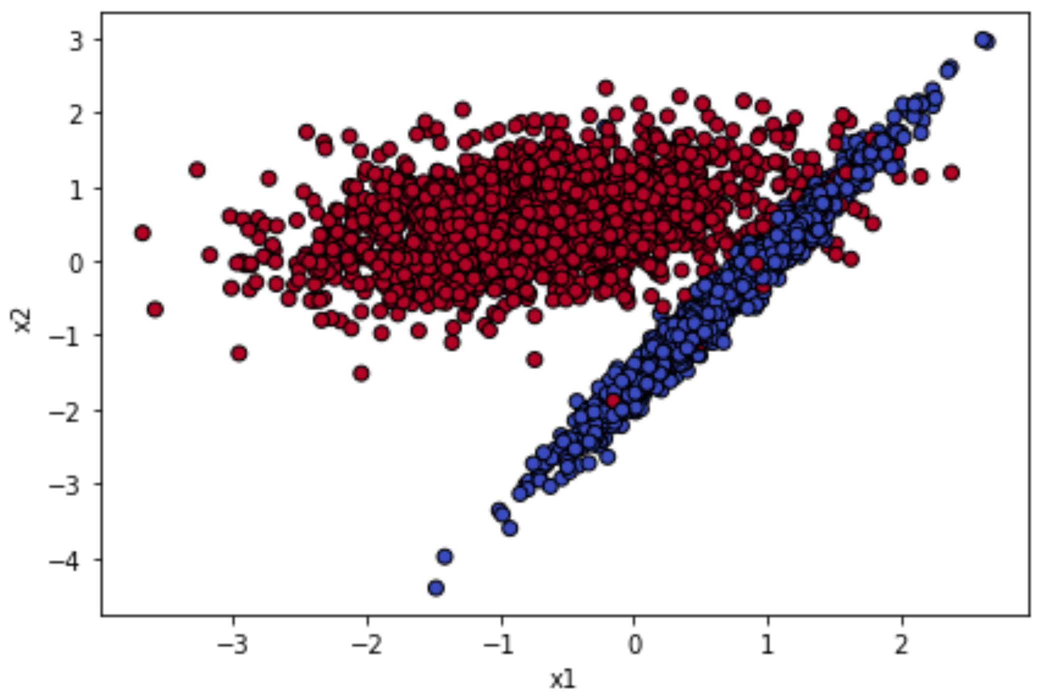
Similar results are obtained on the USPS distribution.

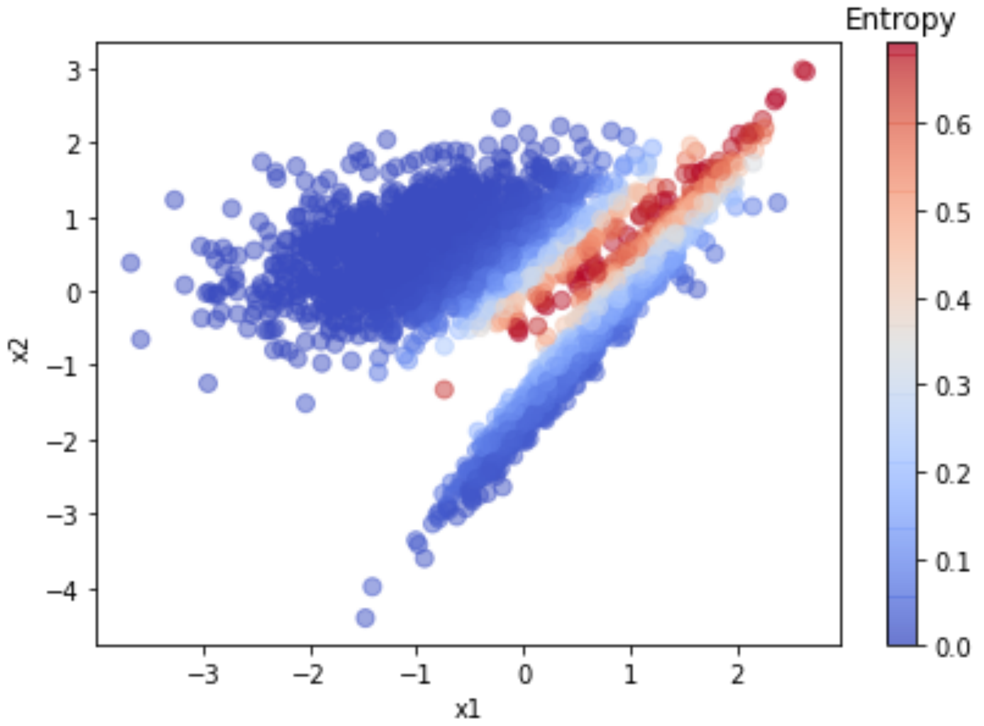




Entropy-driven approach

$$H(x) = -\sum_{m}^{M} p(y = m|x) \log p(y = m|x)$$

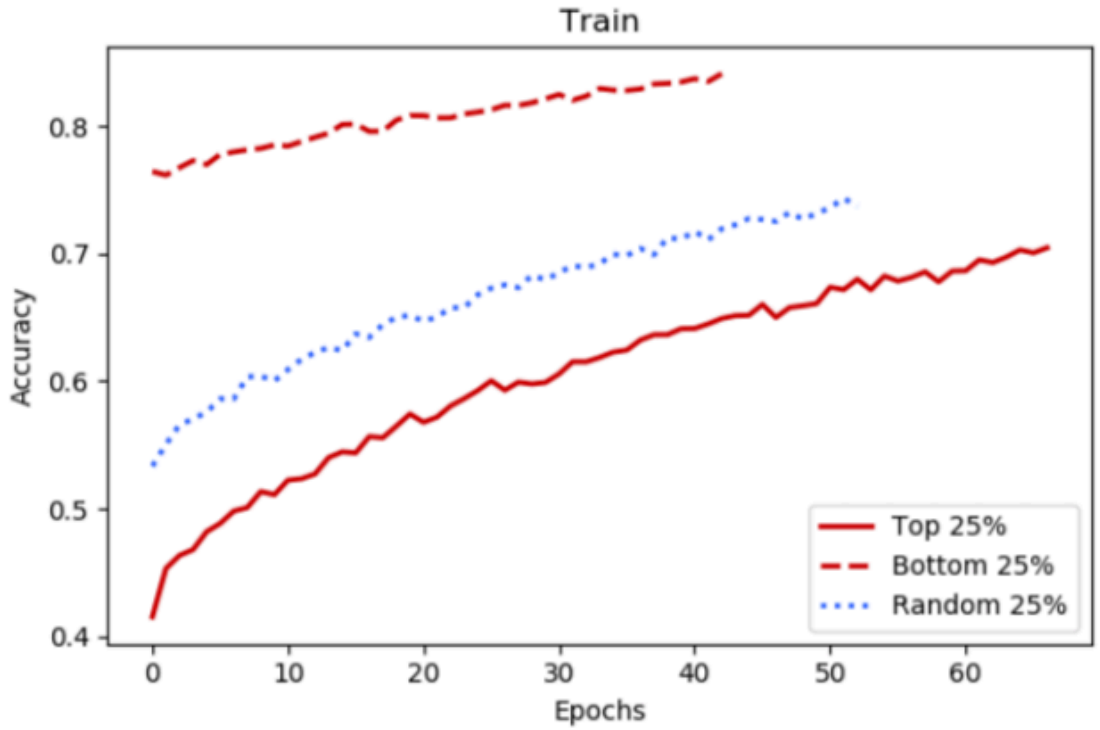


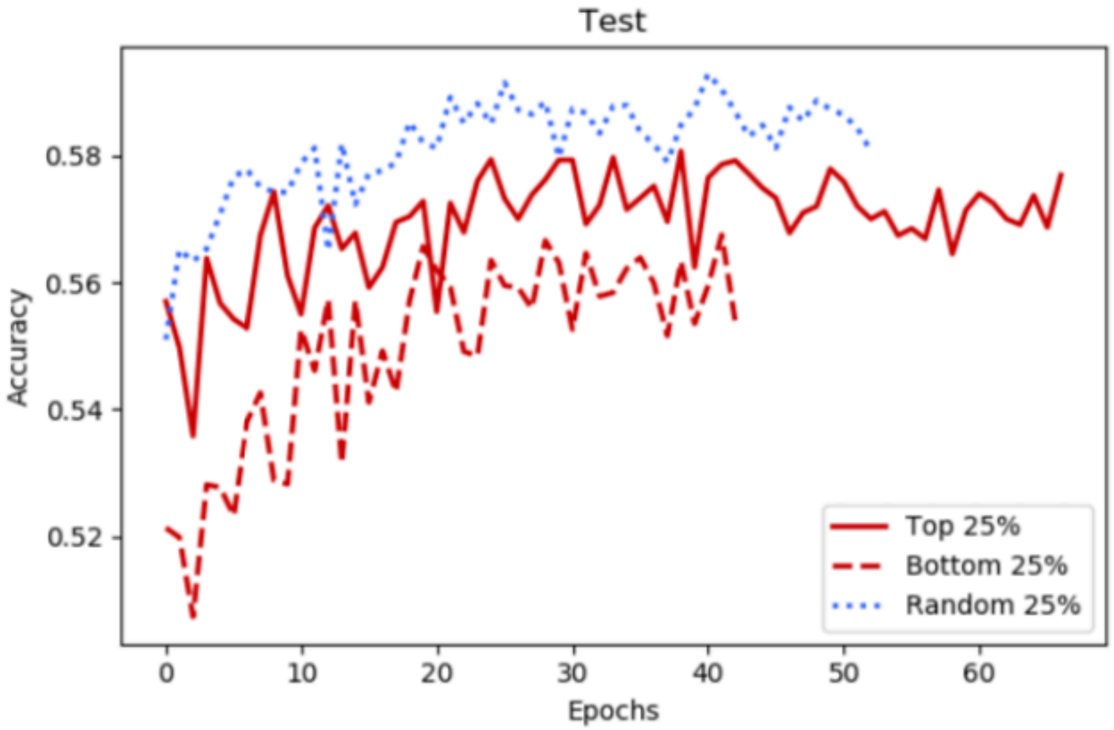


Entropy-driven approach – CIFAR-10

We compare the 25% **most/least** entropic samples with a 25% **random** selection.

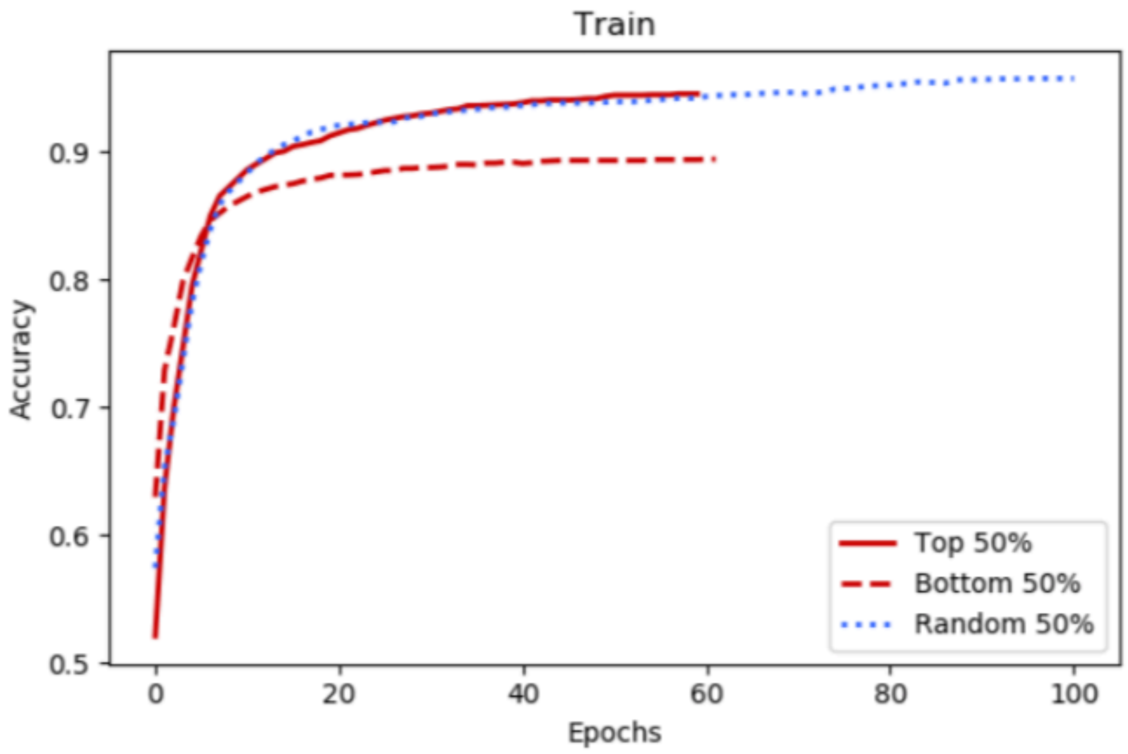
$plain\ embedding\ shift$

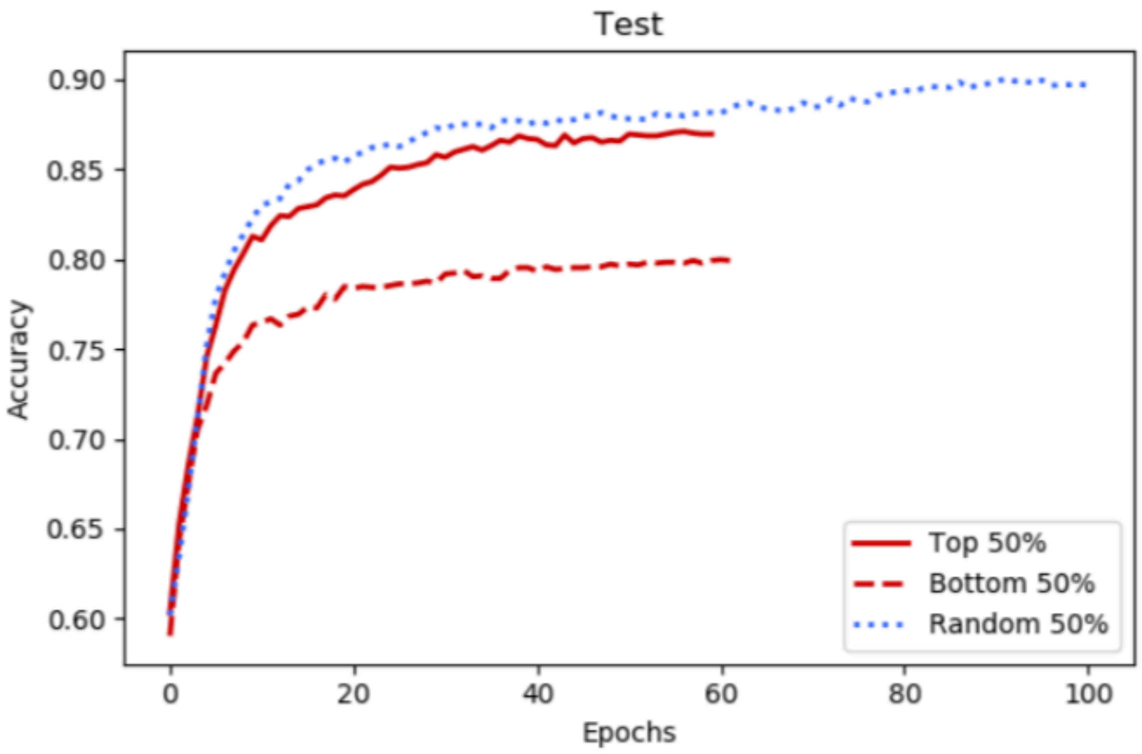




Entropy-driven approach – USPS

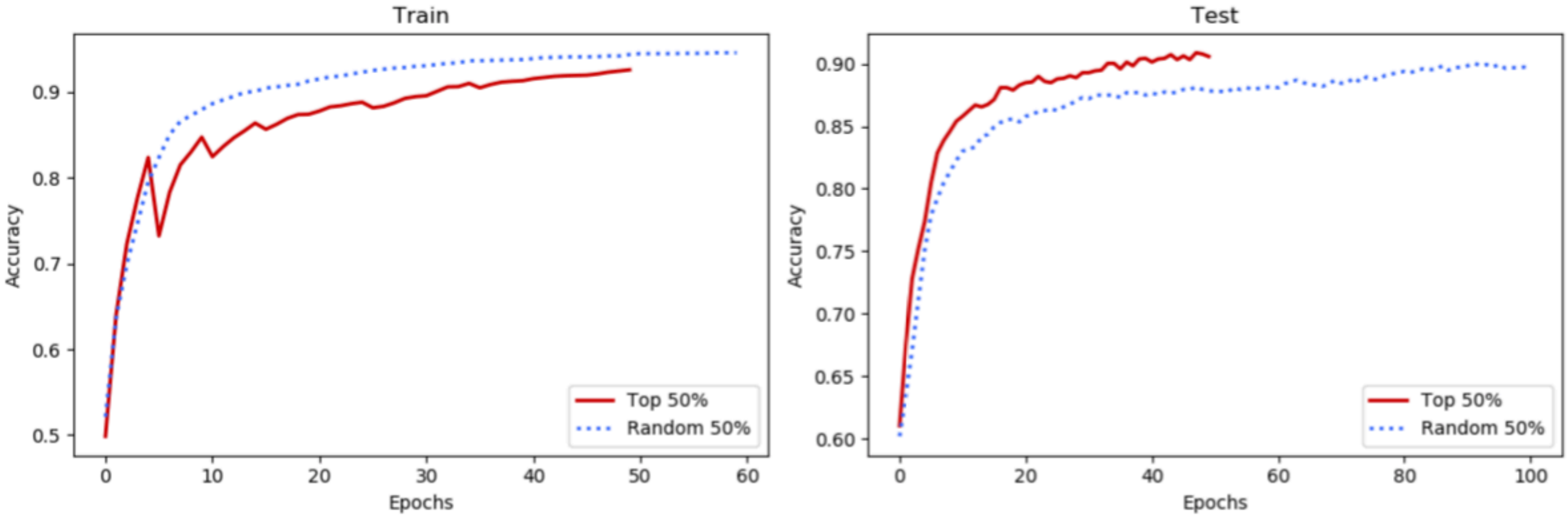
We compare the 50% **most/least** entropic samples with a 50% **random** selection.





Entropy-driven approach – USPS

We compare the 50% **most** entropic samples with a 50% **random** selection, this time we **recompute the subset** every 5 epochs.



Agenda

$$\frac{\partial z}{\partial t}$$

INTRODUCTION

IMAGE RECOGNITION

DIFFERENTIAL EQUATIONS

CONCLUSIONS

Differential Equations – Problem setting

We define the Ordinary Differential Equation as:

$$f(z(t), z'(t), z''(t)), ..., z^n(t), \theta) = 0$$

and we know that, given a differential equation:

$$z'(t) = f(t)$$

there are infinite solutions in the form:

$$z(t) = \int f(t)dt + c, c \in \mathbf{R}$$

Differential Equations – Problem setting (cont.)

If we want to find a specific solutions, we need some **initial conditions**, that defines a Cauchy Problem.

Given an initial condition $z(0)$, our goal is to find a mapping from t to $z(t)$ that satisfies:

$$f(z(t), z'(t), z''(t)), ..., z^n(t), \theta)^2 = 0 \qquad z(0) = \xi_0$$

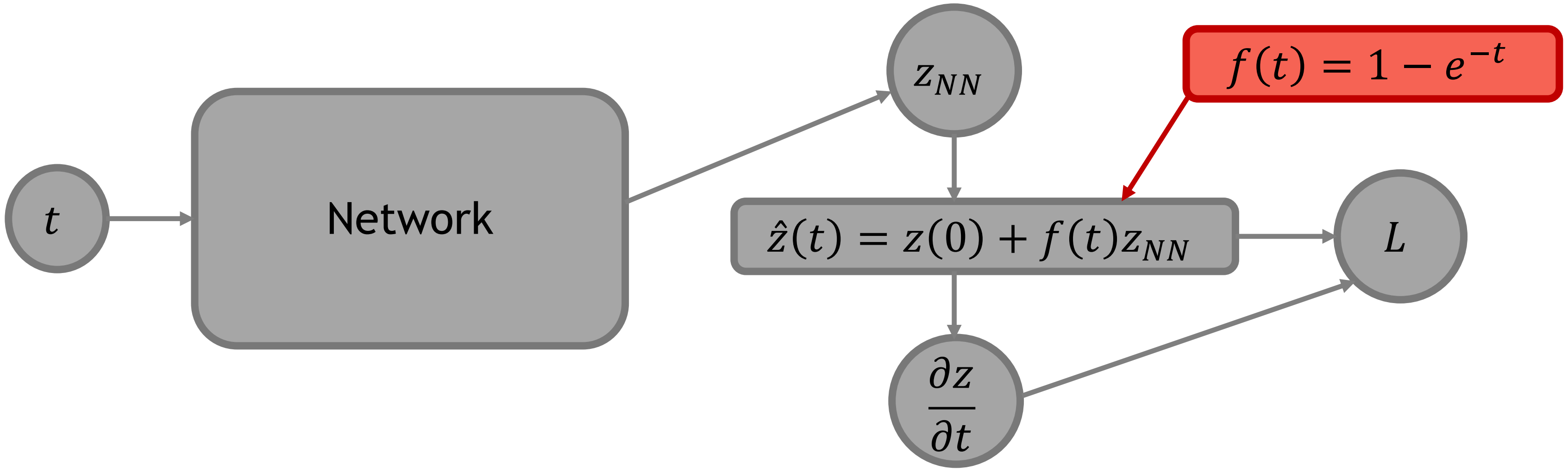
Solving DEs with Neural Networks

Find a function:

$$\hat{z}(t)$$

that minimizes a Loss function:

$$L = f(z(t), z'(t), z''(t)), \dots, z^n(t), \theta)^2$$



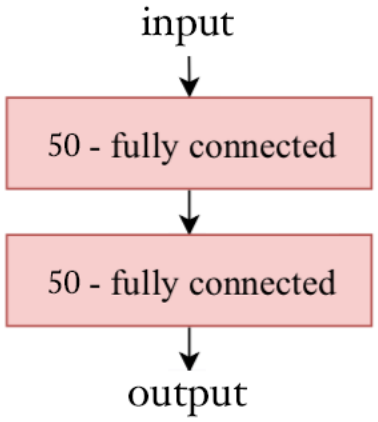
Our application: SIR model

$$\begin{cases} \dot{S} + \dfrac{\beta S I}{N} = 0 \\ \dot{I} - \dfrac{\beta S I}{N} + \gamma I = 0 \\ \dot{R} - \gamma I = 0 \end{cases}$$

S : susceptible people
I : infected people
R : recovered people

β : infection rate
γ : recovery rate

$$L = \frac{1}{K} \sum_{n=0}^{K} \left[\left(\dot{\hat{S}}^{(n)} + \frac{\beta \hat{S}^{(n)} \hat{I}^{(n)}}{N} \right)^2 + \left(\dot{\hat{I}}^{(n)} - \frac{\beta \hat{S}^{(n)} \hat{I}^{(n)}}{N} + \gamma \hat{I}^{(n)} \right)^2 + \left(\dot{\hat{R}}^{(n)} - \gamma \hat{I}^{(n)} \right)^2 \right]$$

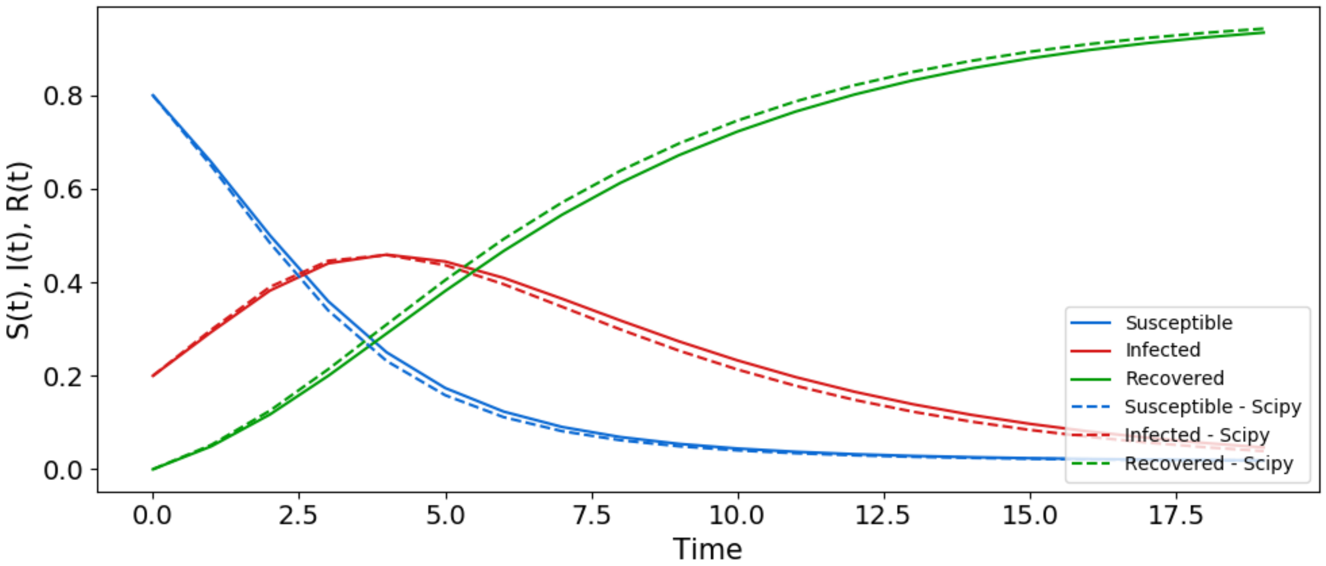


Architecture for SIR model

Example - SIR

$$S(0) = 0.80$$
$$I(0) = 0.20$$
$$R(0) = 0.00$$

$$\beta = 0.80$$
$$\gamma = 0.20$$



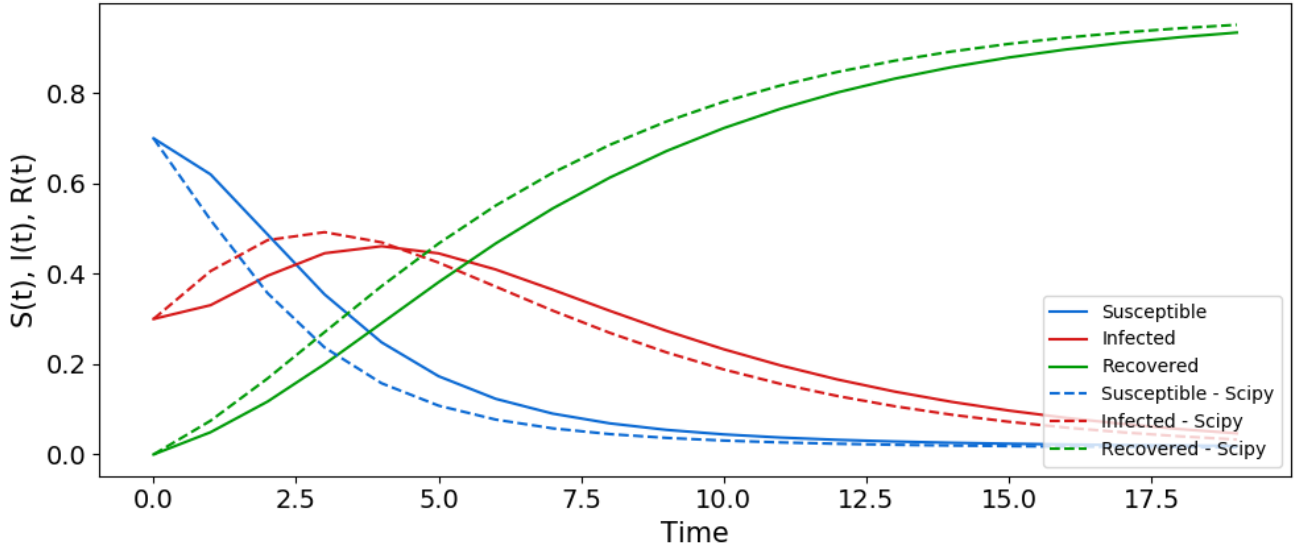
Network trained for 1000 epochs, reaching a final LogLoss $\cong -15$.

Training size: 2000 points
Time interval: $[0, 20]$

What if we perturb the initial conditions?

$S(0) = 0.70$
$I(0) = 0.30$
$R(0) = 0.00$

$\beta = 0.80$
$\gamma = 0.20$



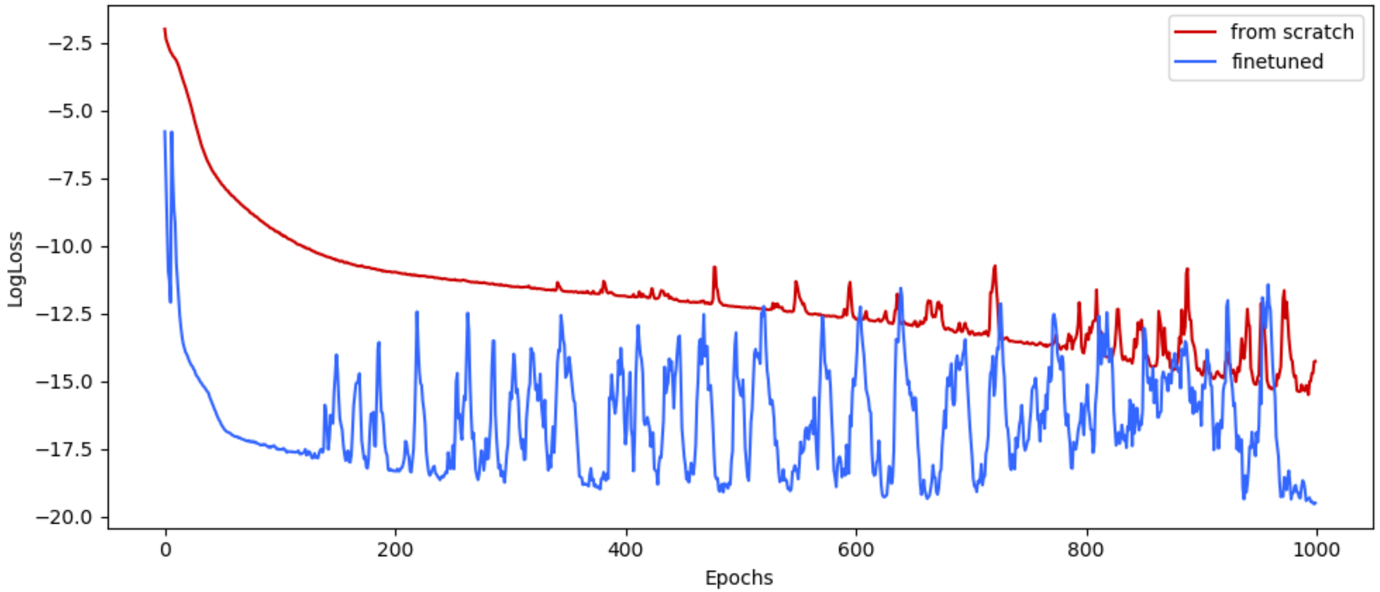
LogLoss $\cong -1.39$

Problem statement: (How) Can we leverage Transfer Learning to re-gain performance?

Fine-tuning results

$$S(0) = 0.80 \rightarrow 0.70$$
$$I(0) = 0.20 \rightarrow 0.30$$
$$R(0) = 0.00$$

$$\beta = 0.80$$
$$\gamma = 0.20$$



Can we do more?

This specific architecture allows us to solve **one** single Cauchy problem at a time.

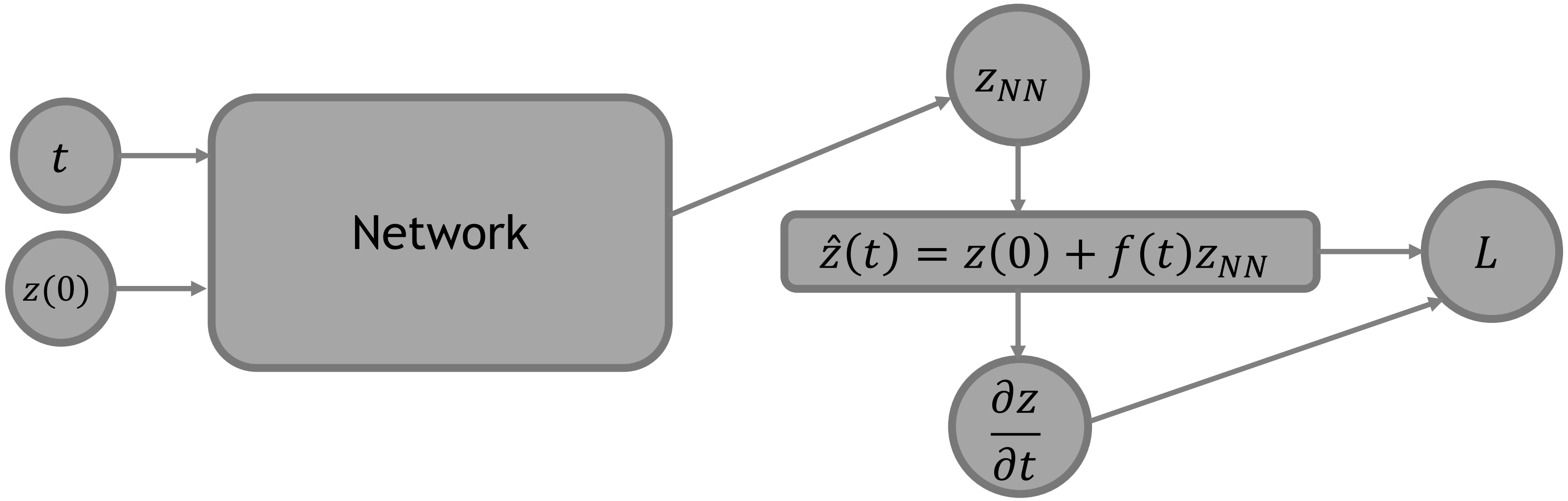
If we change the initial conditions, even by a small amount, we need to retrain.

We focused on the **architecture impact**: can we make it generalize over a **bundle** of initial conditions?

Architecture modification

We added two additional inputs to the network: the initial conditions $z(0)$.

With this modification, we are able to learn **multiple** Cauchy problems all together.



Bundle of initial conditions - Results

Training bundle

$$I(0) \in [0.10, 0.20]$$

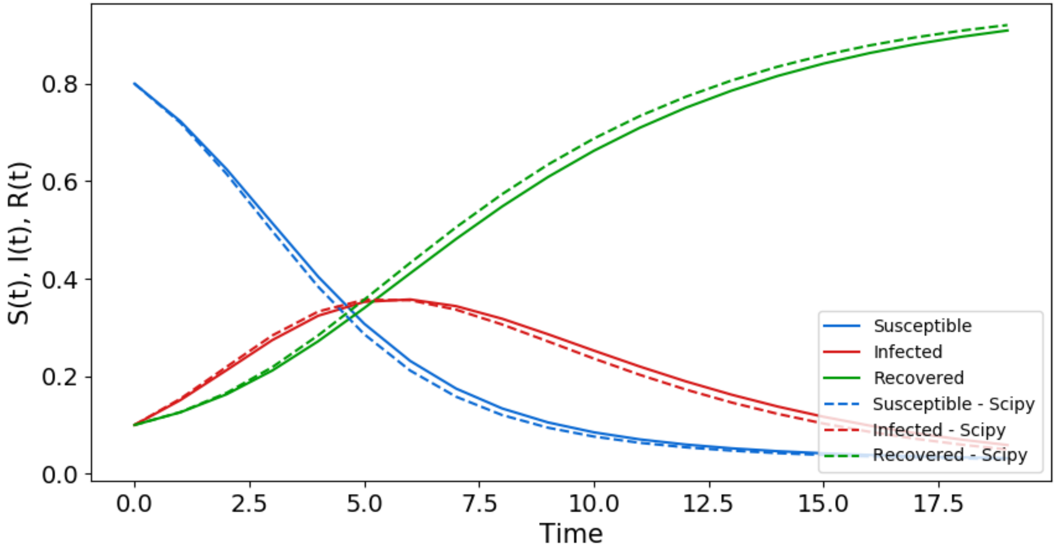
$$R(0) \in [0.10, 0.20]$$

$$S(0) = 1 - (I(0) + R(0))$$

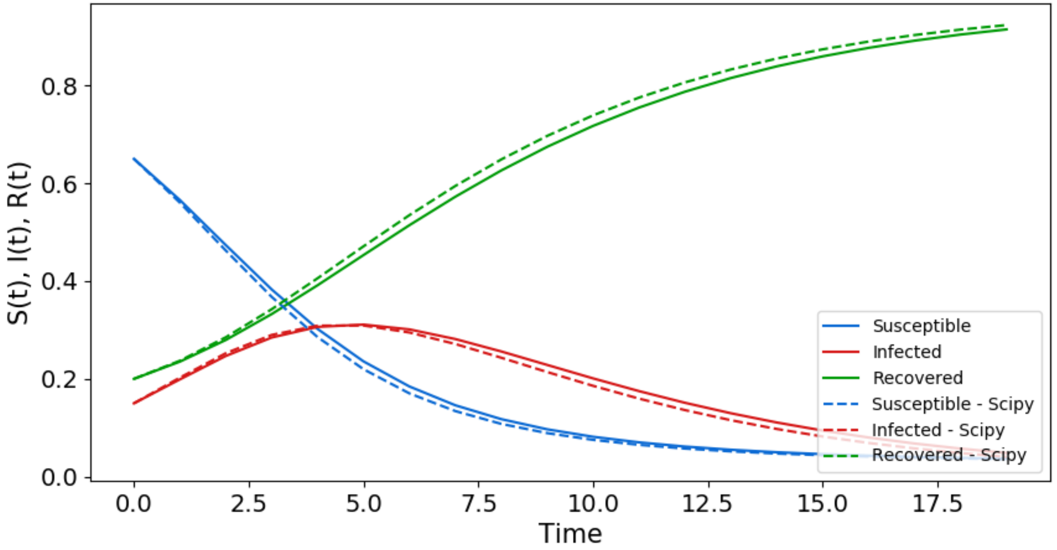
$$\beta = 0.80$$

$$\gamma = 0.20$$

$\boldsymbol{I(0) = 0.10, R(0) = 0.10}$



$\boldsymbol{I(0) = 0.20, R(0) = 0.15}$



Bundle perturbation and finetuning results

Training bundle

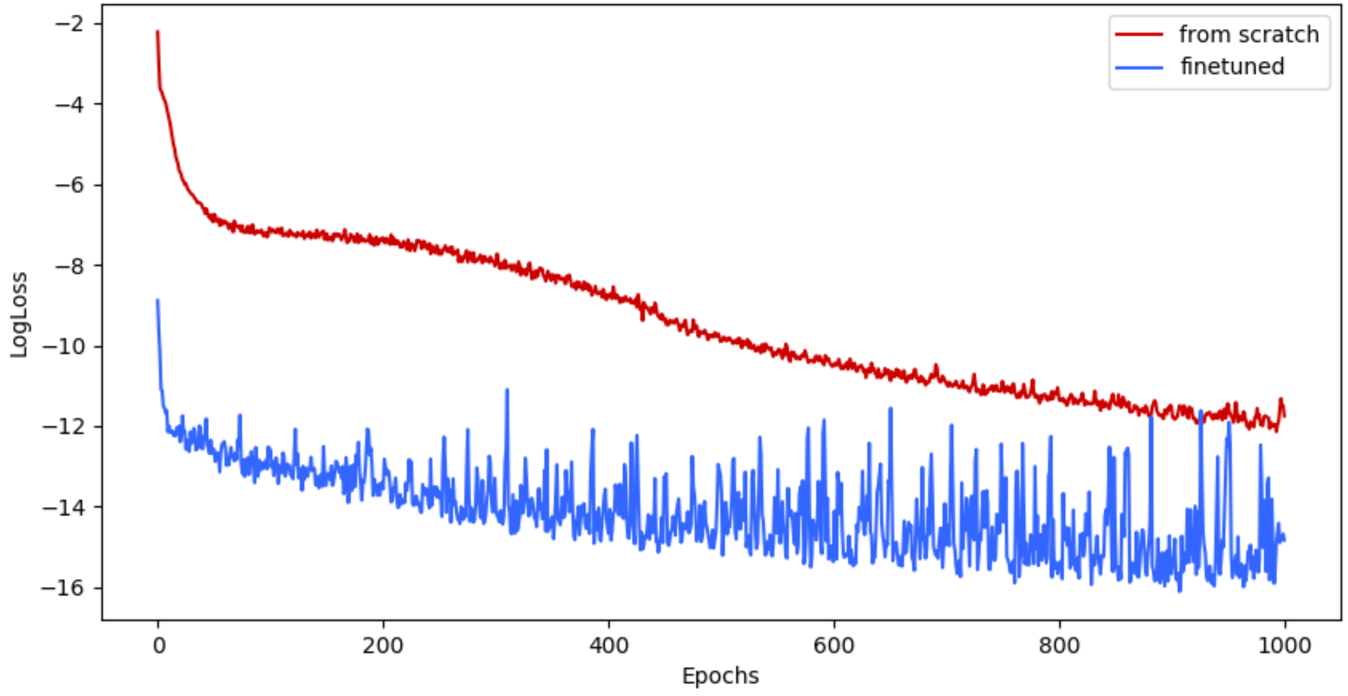
$$S(0) = 1 - (I(0) + R(0))$$

$$I(0) \in [0.10, 0.20] \rightarrow [0.30\ 0.40]$$

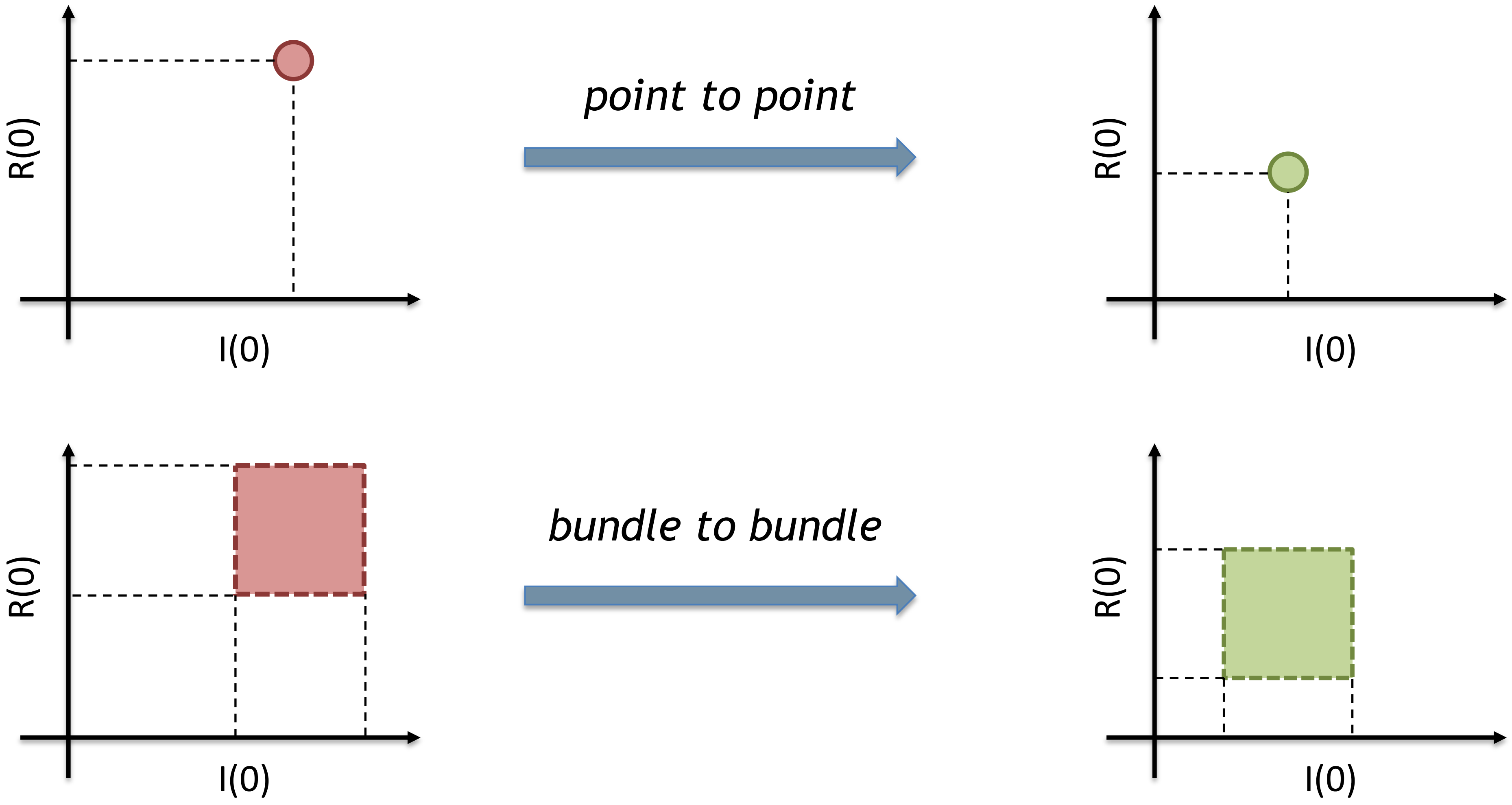
$$R(0) \in [0.10, 0.20] \rightarrow [0.30, 0.40]$$

$$\beta = 0.80$$

$$\gamma = 0.20$$

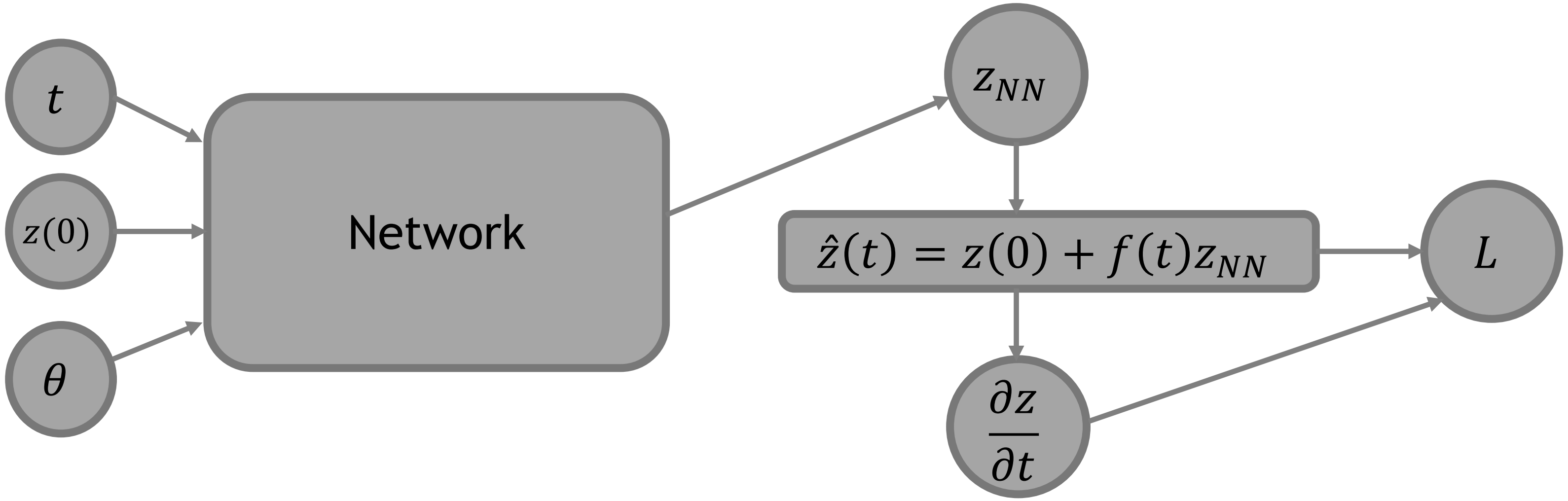


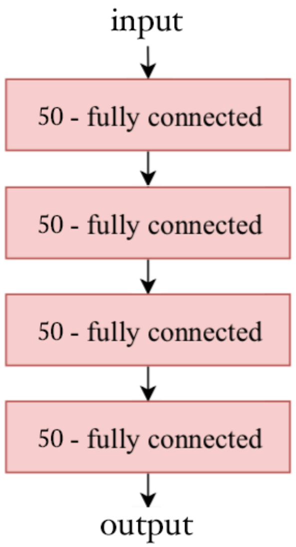
Finetuning improvements



One more input: the parameters

We gave the network full flexibility by adding as input the parameters θ.





Architecture for SIR model

Bundle perturbation and finetuning results

Training bundle

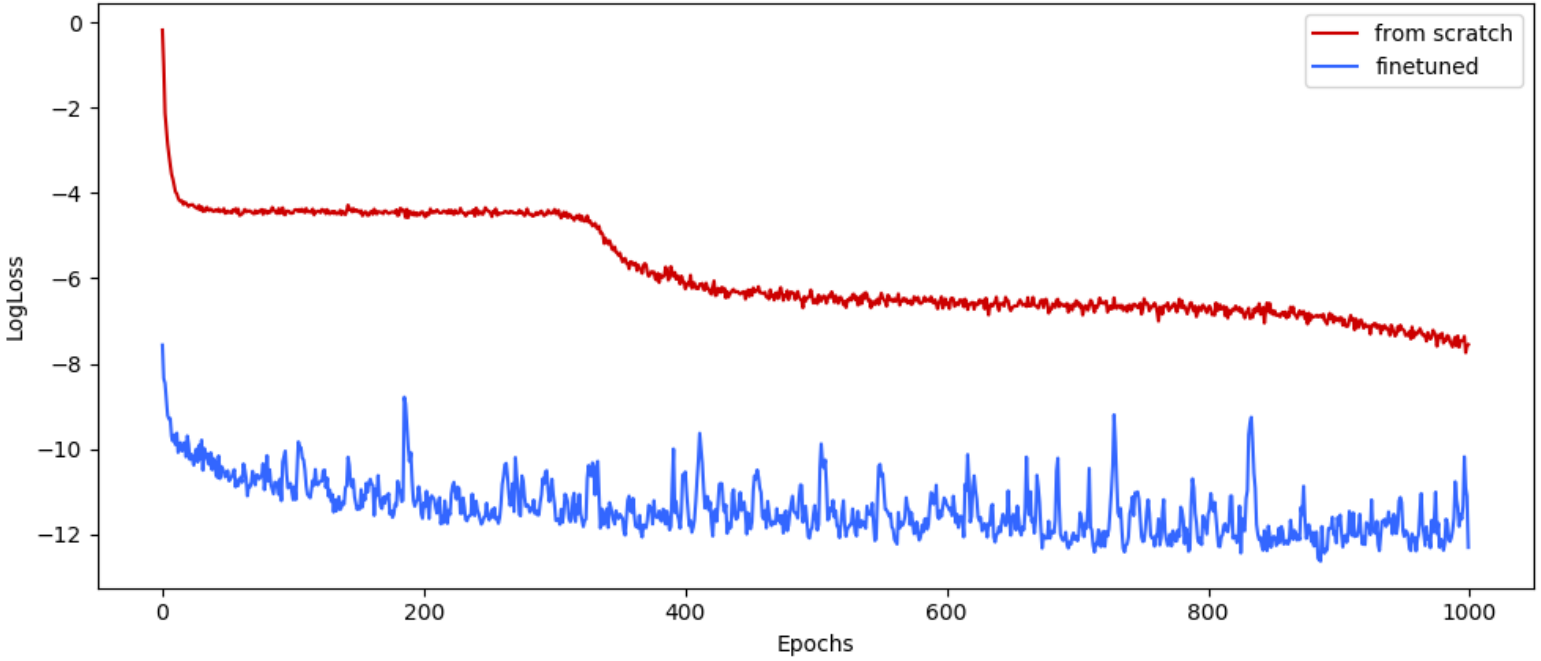
$$S(0) = 1 - (I(0) + R(0))$$

$$I(0) \in [0.20, 0.40] \rightarrow [0.30, 0.50]$$

$$R(0) \in [0.10, 0.30] \rightarrow [0.20, 0.40]$$

$$\beta \in [0.40, 0.80] \rightarrow [0.60, 1.0]$$

$$\gamma \in [0.30, 0.70] \rightarrow [0.50, 1.0]$$



Loss trend inside/outside the bundle

Training bundle

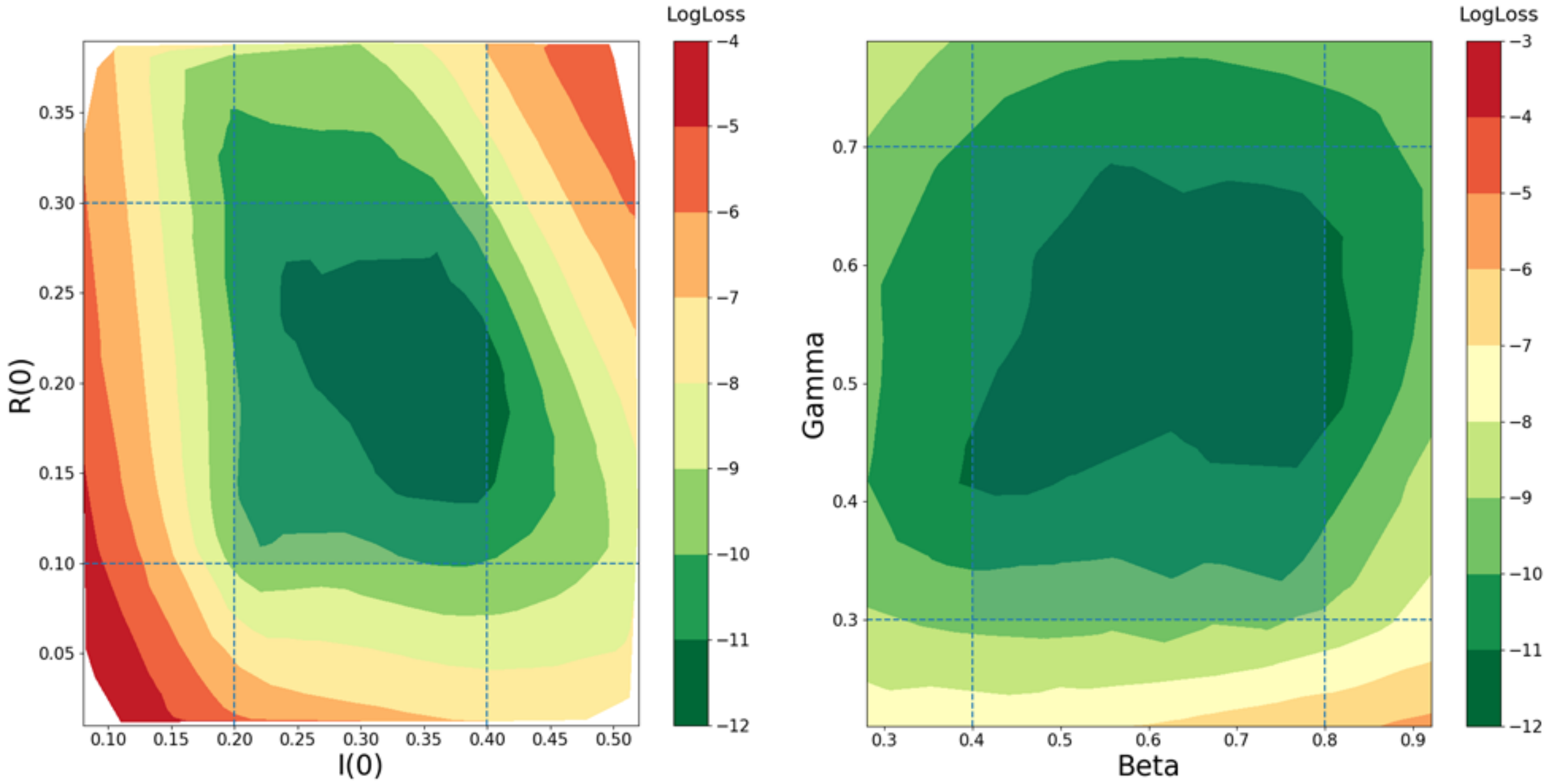
$$S(0) = 1 - (I(0) + R(0))$$

$$I(0) \in [0.20, 0.40]$$

$$R(0) \in [0.10, 0.30]$$

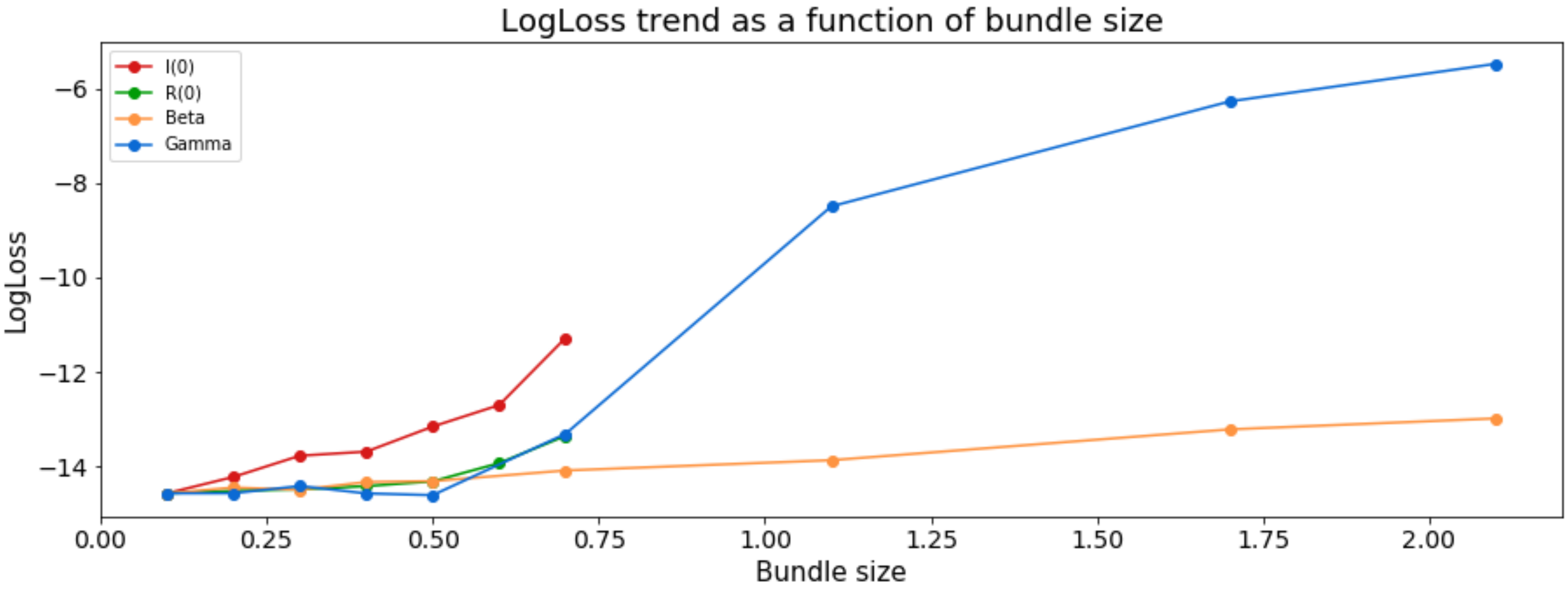
$$\beta \in [0.40, 0.80]$$

$$\gamma \in [0.30, 0.70]$$



Color represents the LogLoss of the network for a solution generated for that particular combination of $(I(0), R(0))$ or (β, γ)

How far can Transfer Learning go?



Agenda

$\frac{\partial z}{\partial t}$

INTRODUCTION

IMAGE RECOGNITION

DIFFERENTIAL EQUATIONS

CONCLUSIONS

Conclusions and Future Works

- Analysis on **data impact** and **architecture impact**

- Data-selection methods are sometimes hard to generalize
- Giving the network more flexibility helps transfer

- It would be appropriate to continue the research in the field of uncertainty sampling
- How does each bundle perturbation affects the network?

Thank you!

M.Sc. Thesis in Computer Science and Engineering

Candidates: Alessandro Saverio Paticchio, Tommaso Scarlatti

Advisor: Prof. Marco Brambilla – Politecnico di Milano
Co-advisor: Prof. Pavlos Protopapas – Harvard University