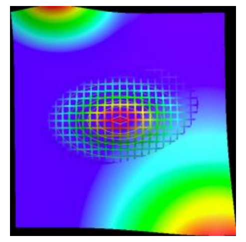


High-dimensional Hessian metric representation on GPGPUs

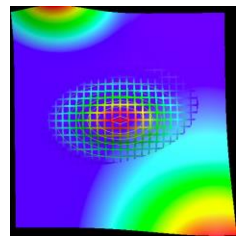
a machine learning example for sparse normalized representations

joint works with András Benczúr and Rita Aleksziev

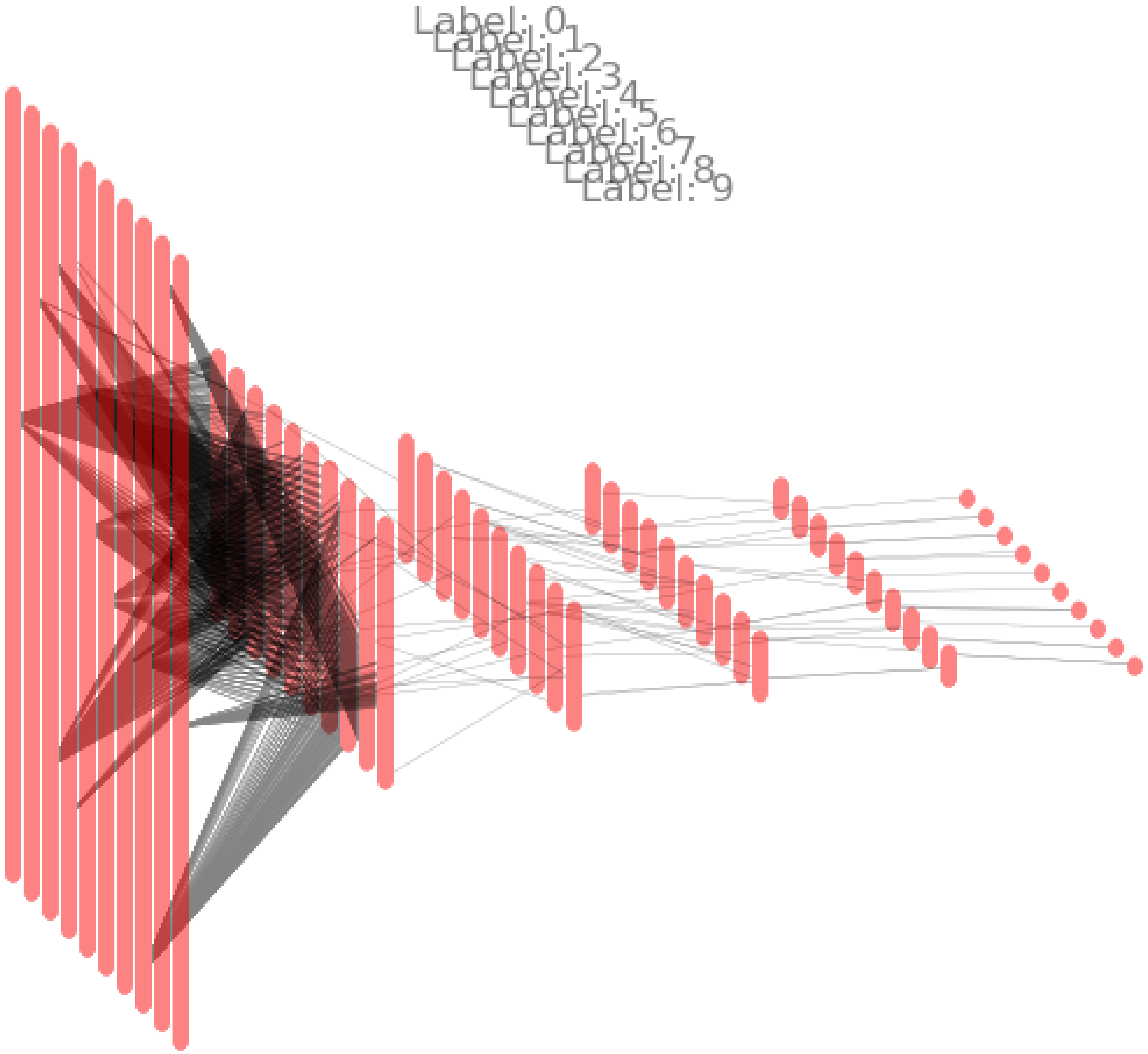
Bálint Daróczy

Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI)

GPU Day 2019
July 12 2019



Motivation: gradient structures of feed-foward networks are distinguishable



Feed-forward networks:

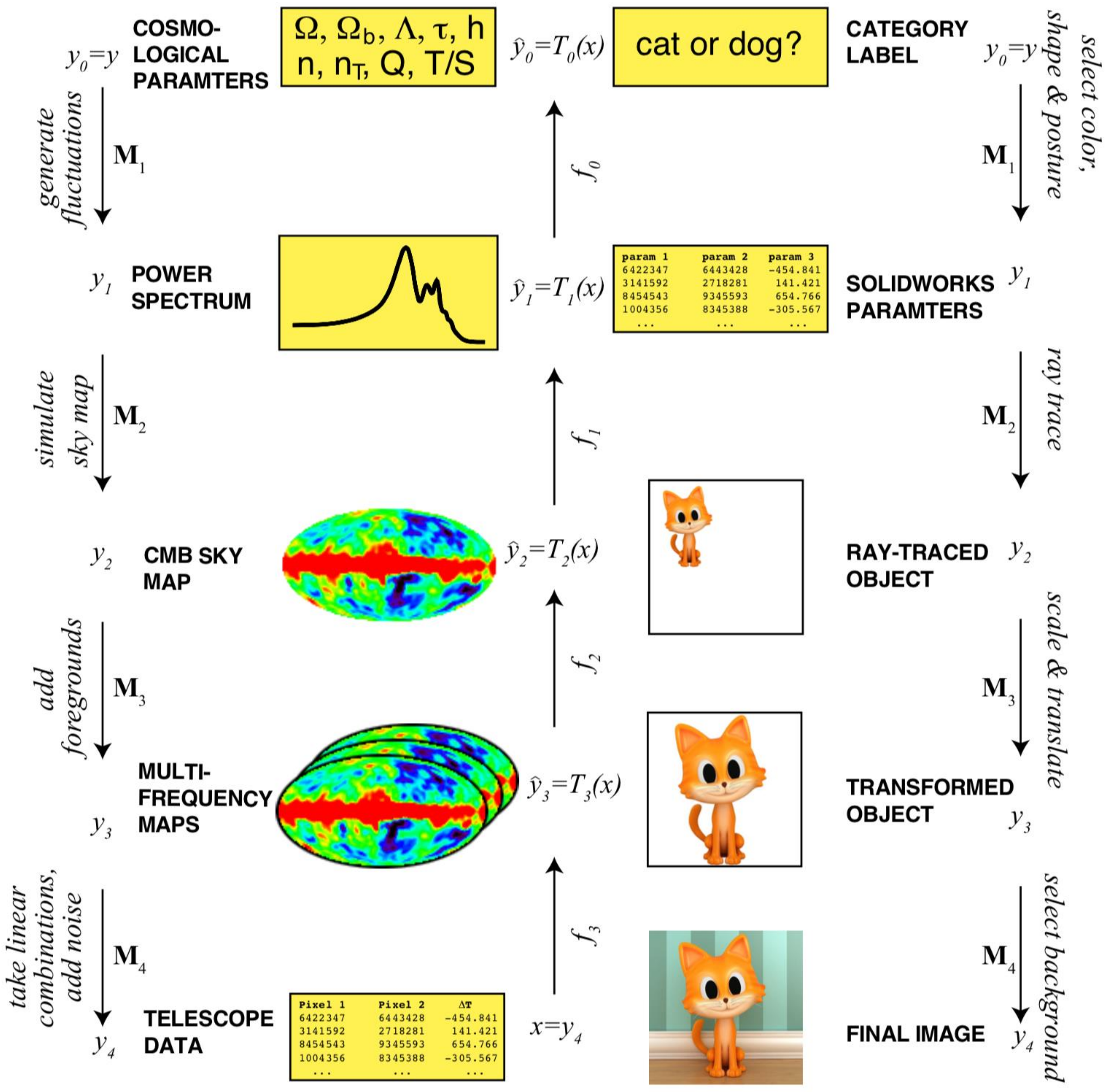
- DAG with activation functions
- ordered disjoint layers
- usually continuously differentiable
- trainable via back-propagation
- highly non-convex
- there are cases where the local minimums are close to global [Choromanska et al., 2015]

Gradient graph:
Nodes: parameters
Edges: dependencies

Sparsity?

Example: Multi-layer perceptron with 256-128-64-32-10 neurons for CIFAR10

Motivation: low degree poly structures

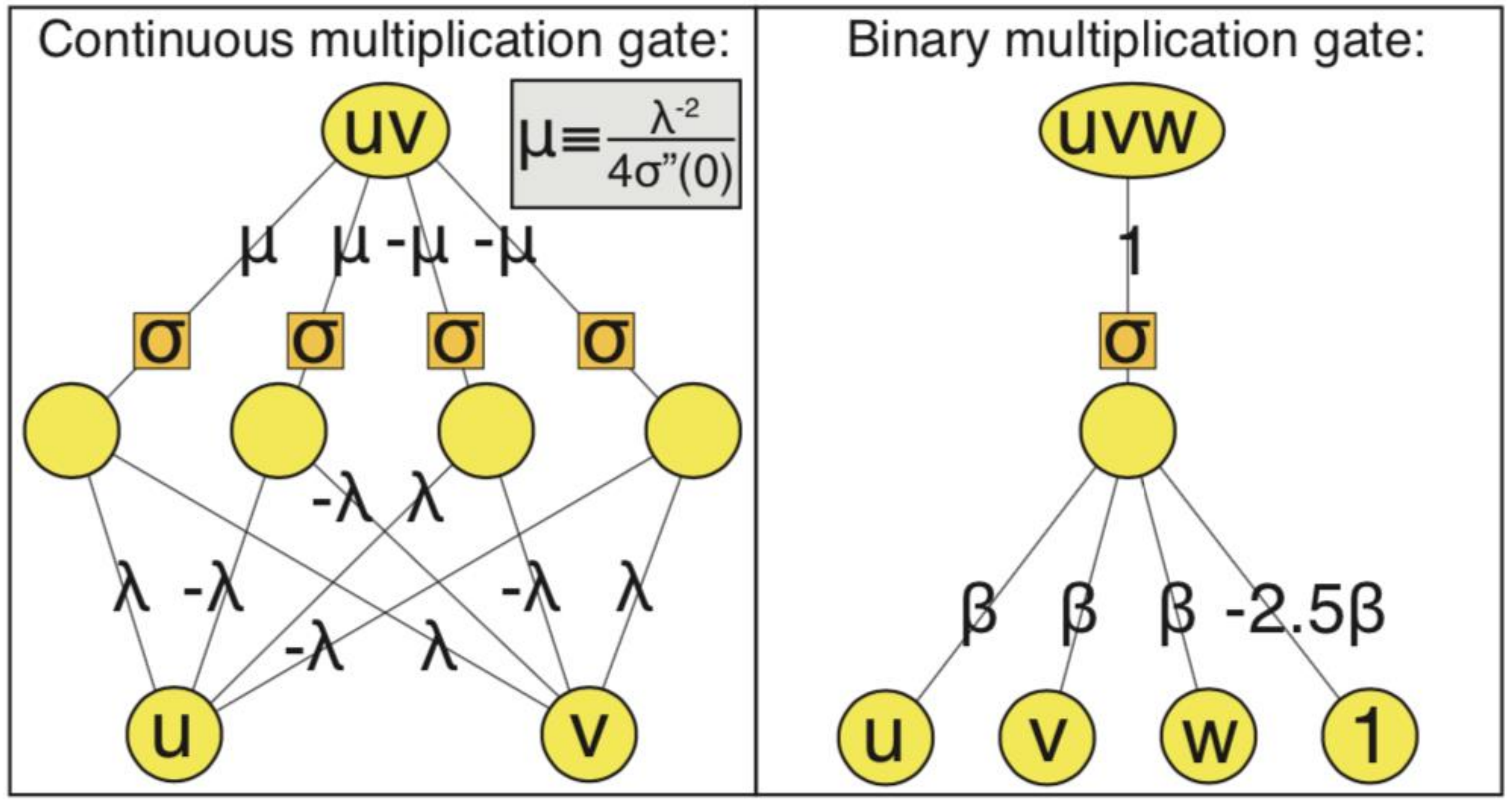


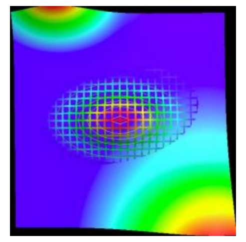
LIN, H. W., AND TEGMARK, M. Why does deep and cheap learning work so well? Journal of Statistical Physics, 2017

Hamiltonians are D<5 degree polynomials…

Why not poly networks?

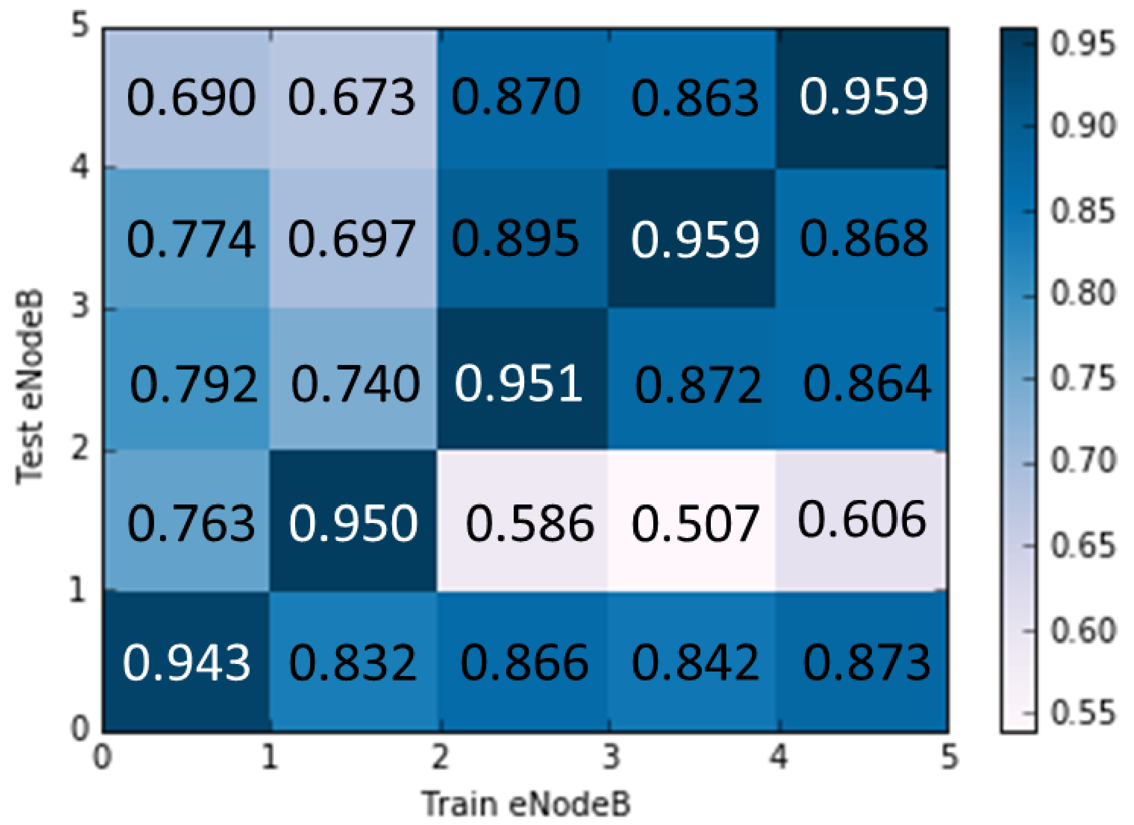
Results indicate that for known hierarchical/compositions functions deep structures can be exponentially better than shallow models.



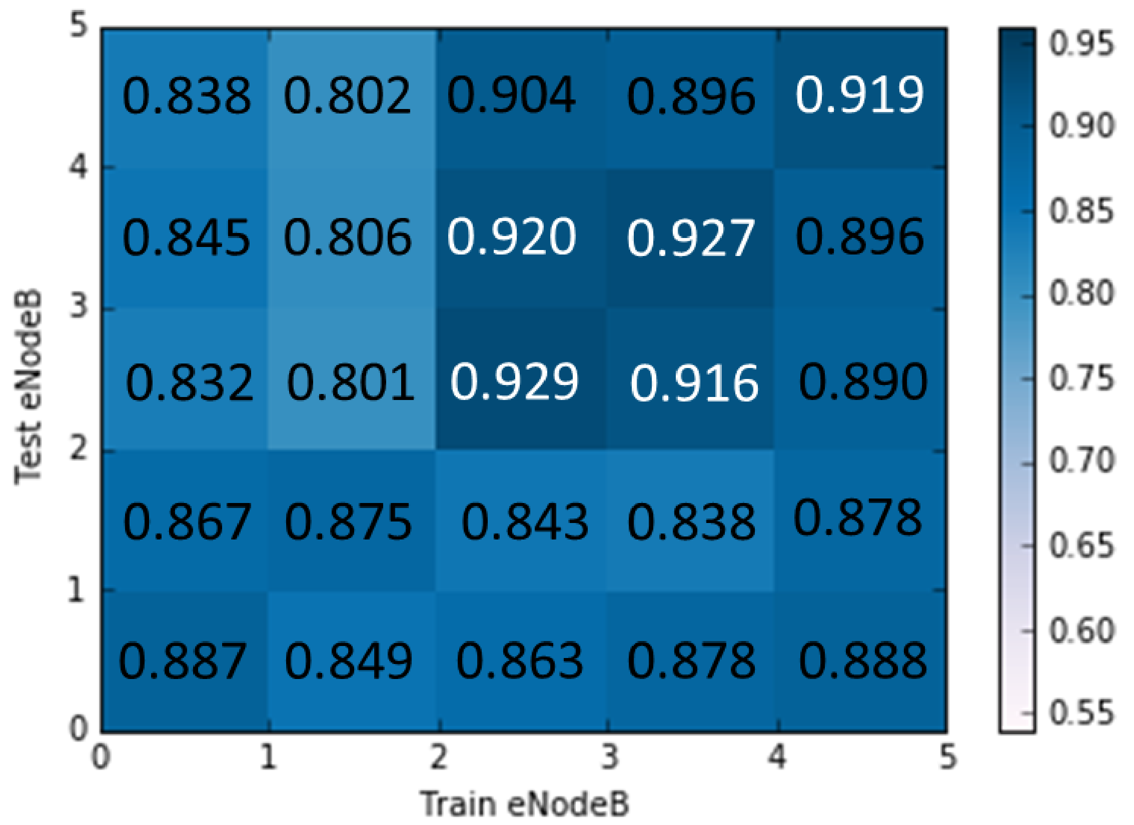


Motivation: promising results over normalized gradients

Drop prediction in LTE data connection, transfer learning between eNodeBs[1]

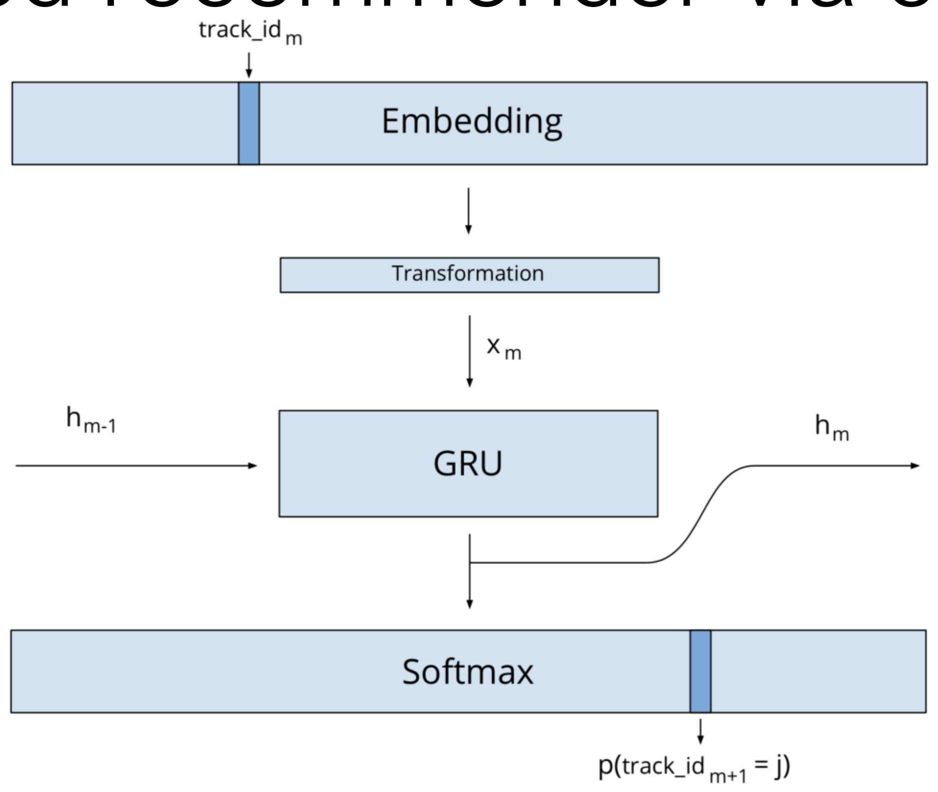


Traditional method (AdaBoost) AUC



Similarity Kernel AUC

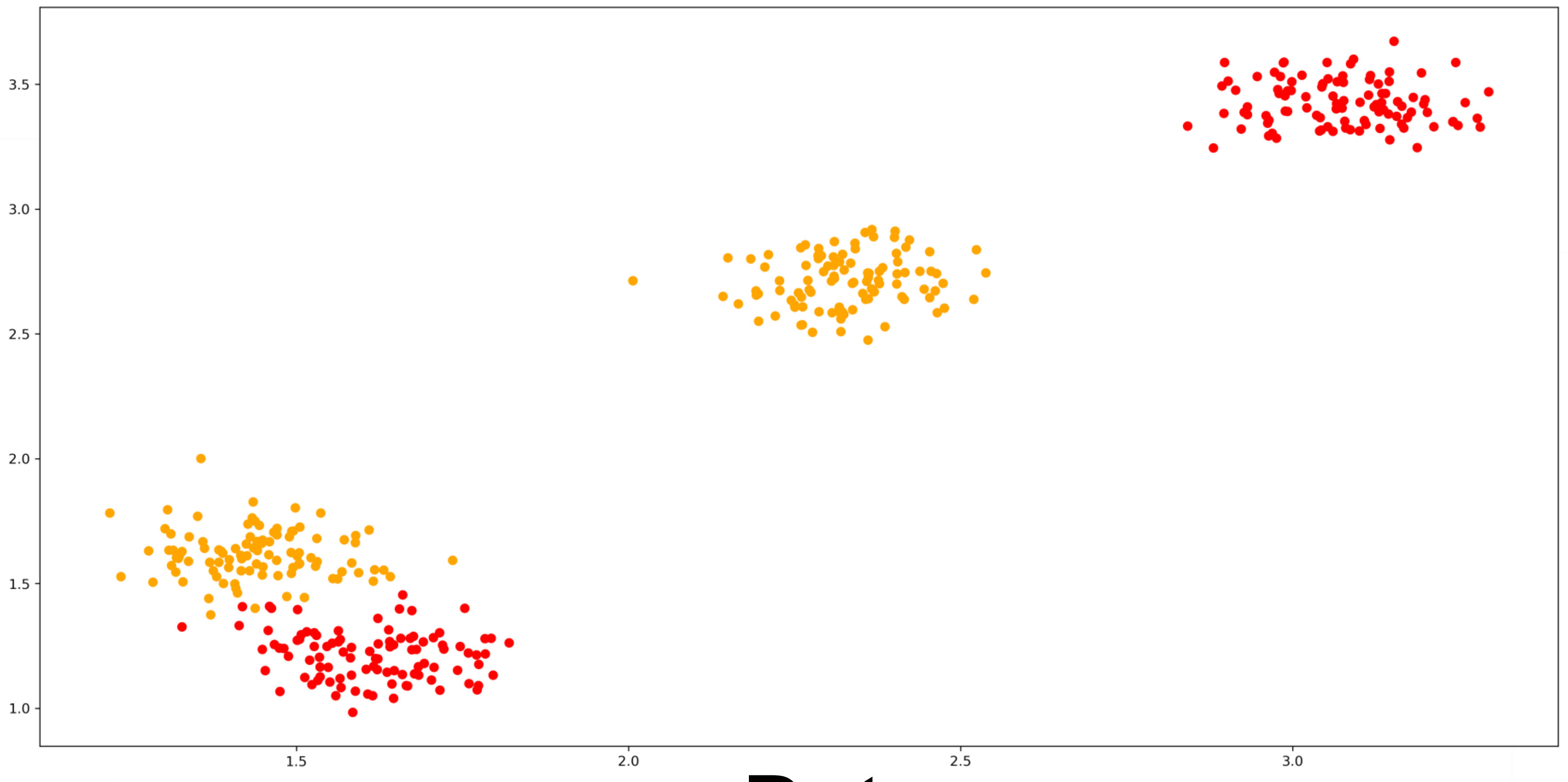
Session based recommender via conditional Fisher information normalized gradient embedding[2]



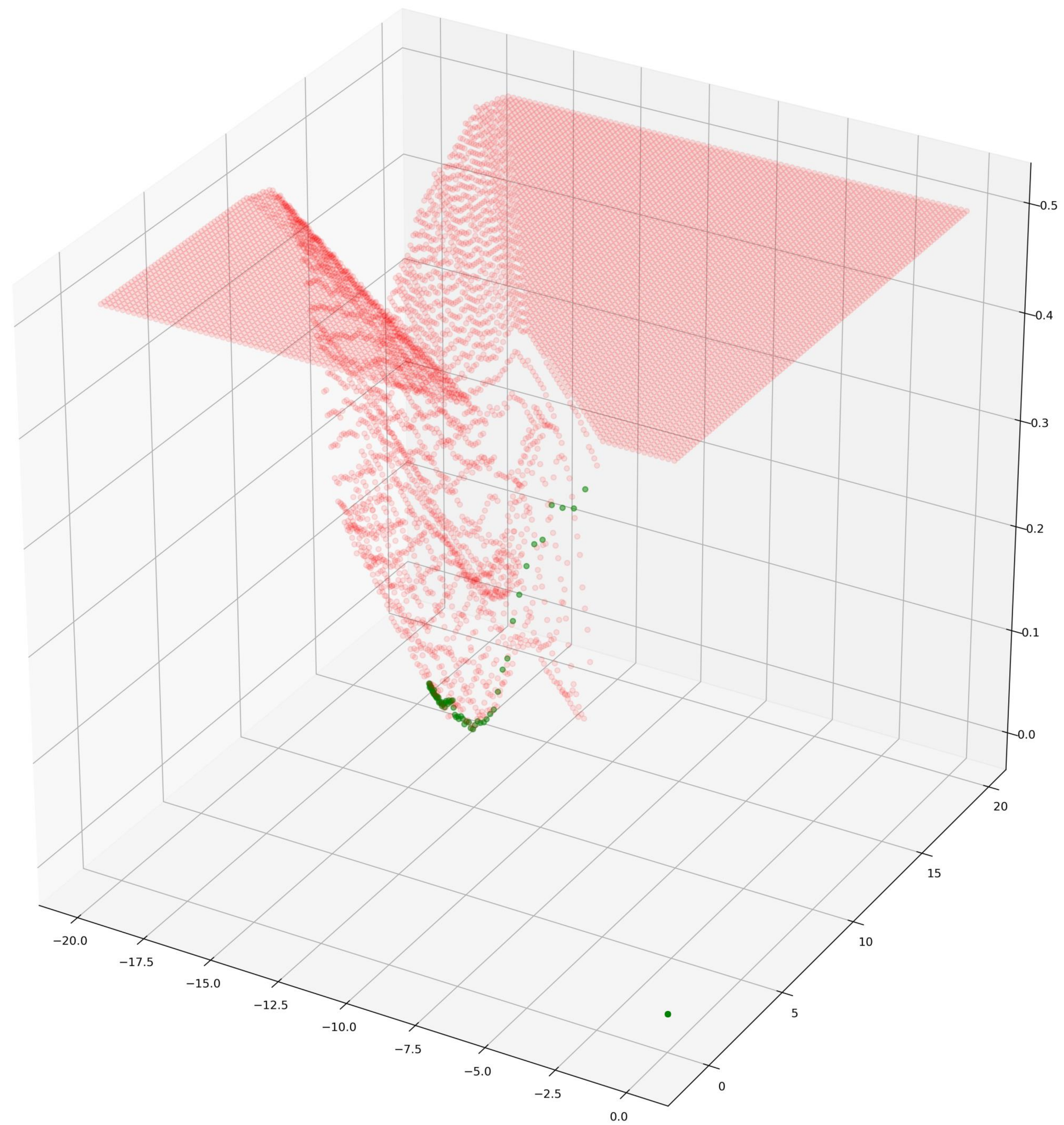
	MPR	DCG@20	Recall@20
Random embedding	0.1642	0.296	0.582
Neural embedding	0.0890	**0.466**	0.799
Feedback Jaccard based Fisher embedding	0.0853	0.437	0.794
Content based Fisher embedding	0.0985	0.405	0.757
Feedback and Content combination	**0.0809**	0.446	**0.803**

[1] Bálint Daróczy, Péter Vaderna, and András Benczúr: "Machine learning based session drop prediction in lte networks and its son aspects." In Proceedings of IWSON at IEEE 81st Vehicular Technology Conference VTC'15 Spring, Glasgow, Scotland 2015, 2015.
[2] Domokos Kelen, Bálint Daróczy, Frederick Ayala-Gómez, Anna Ország, András Benczúr: "Session Recommendation via Recurrent Neural Networks over Fisher Embedding Vectors", Sensors, under revision

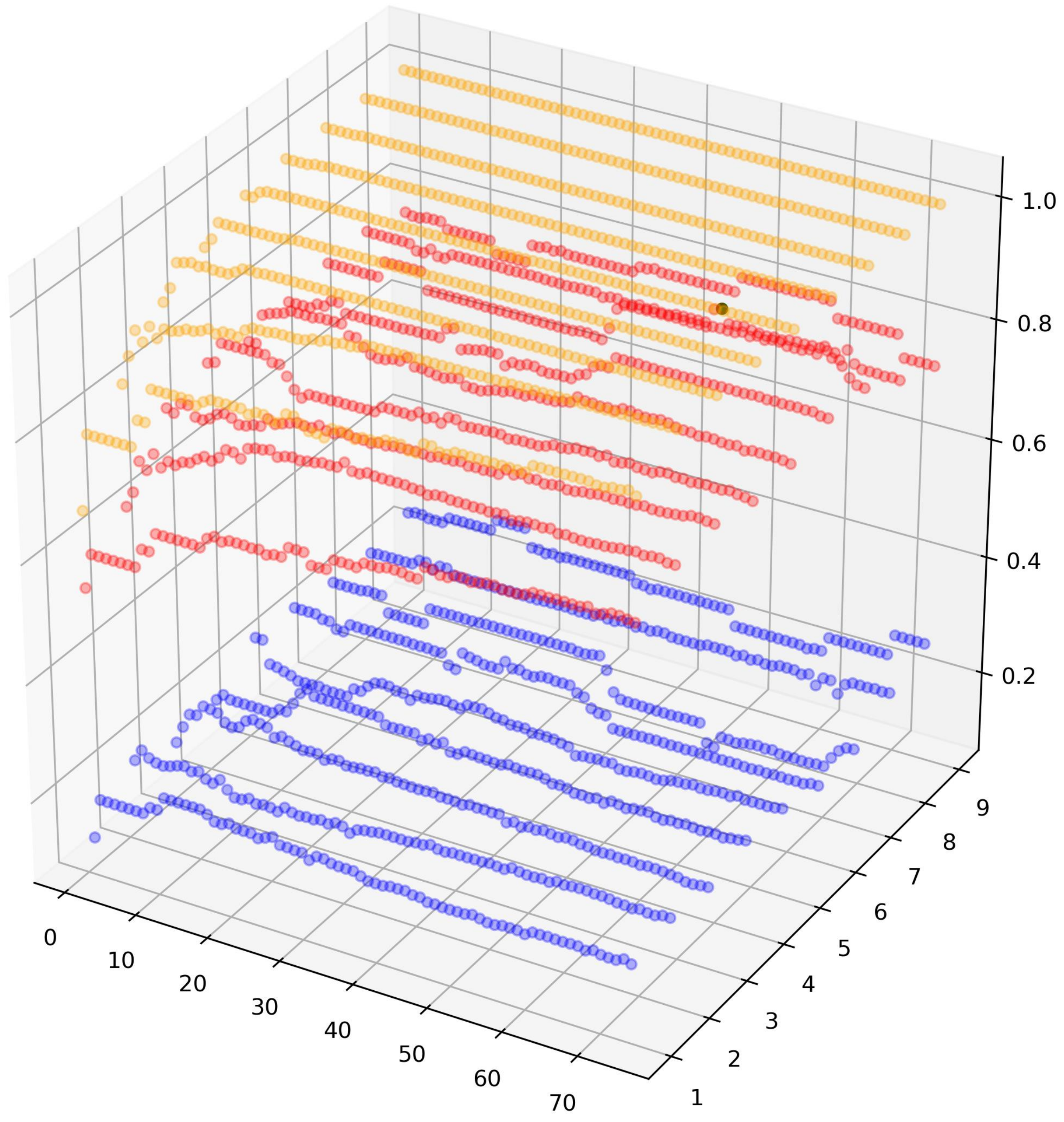
Manifolds or point clouds in machine learning ?



Data

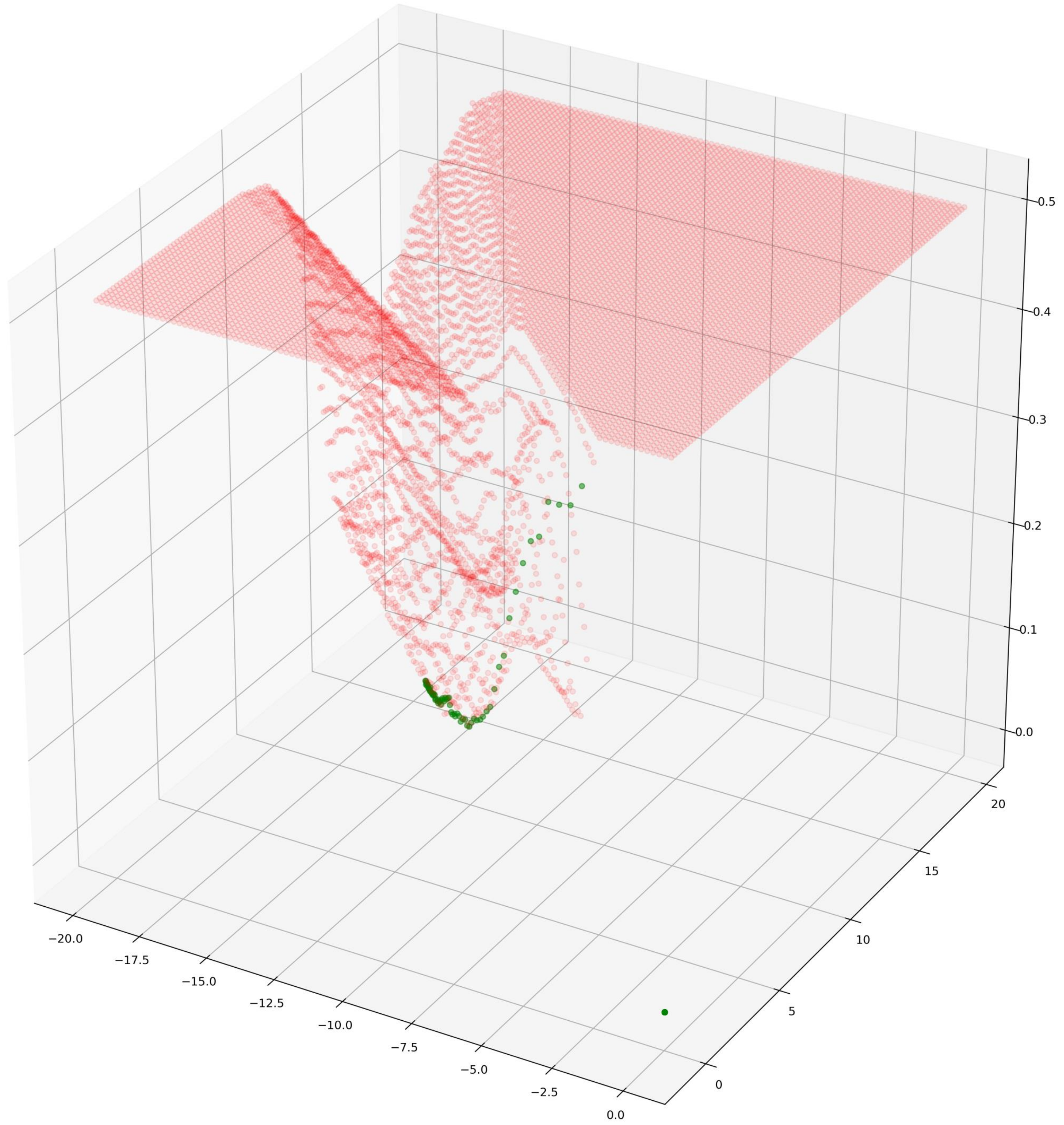


Loss surface



Hyperparameter surface

Example: linear separator in 2D with binary labels



It looks interesting :)

What about Neural Networks? Later…

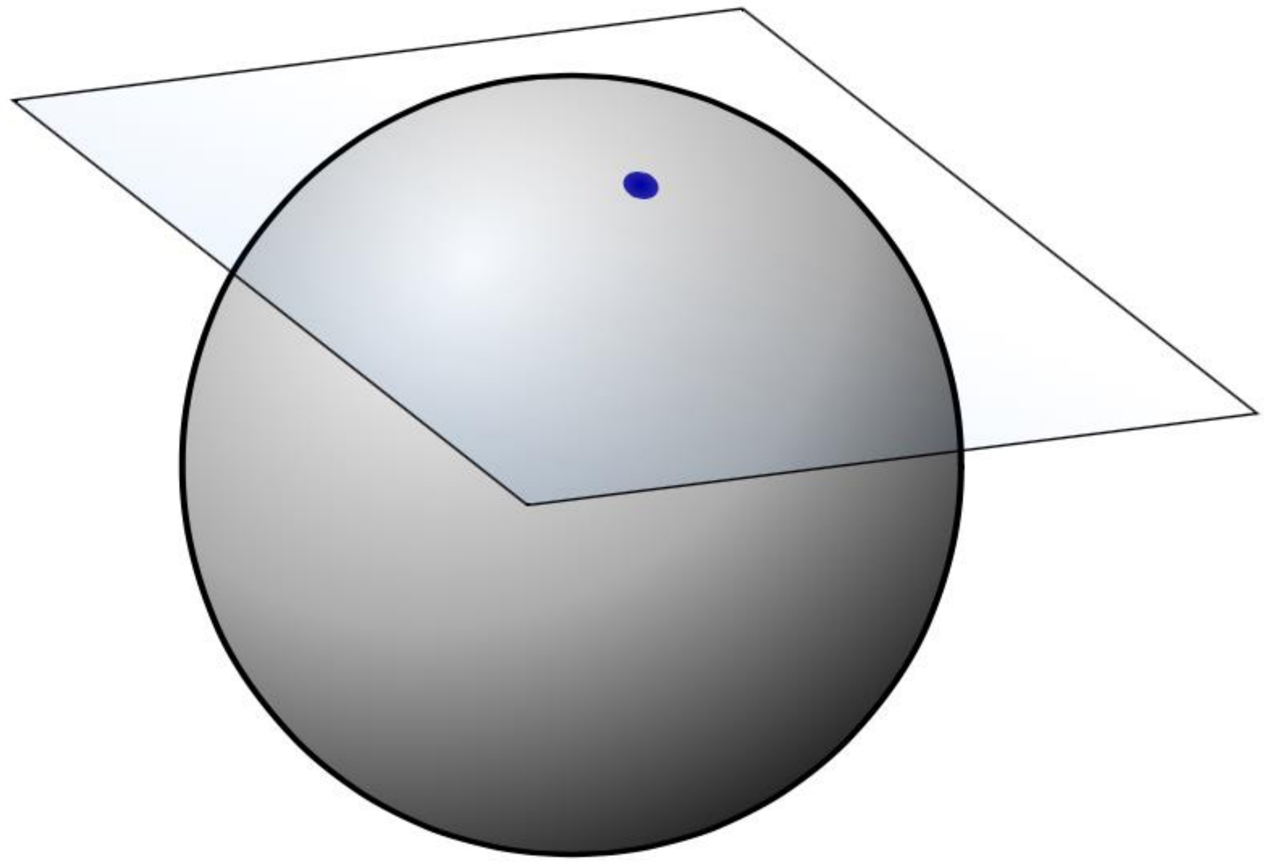
What kind of surface is that?

Spoiler: differentiable manifold

Manifolds

Manifolds are in a way surfaces

“Objects with n degree of freedom”:
The dimension of a manifold is the number of free (independent) parameters to specify a point on the manifold (not equal to the embedded dimension!).



Examples:

1. Curves embedded into 3D, x,y,z = f(t),g(t),h(t) for some continuous functions f,g,h

2. Surface of unit ball in R^3 is a 2-dimensional manifold (unit sphere in n-dim: given a starting point the parametrization in the neighbourhood of the point is only n-1 dimensional)

3. Solid ball in R^3 is a 3-dimensional manifold

Topological manifold

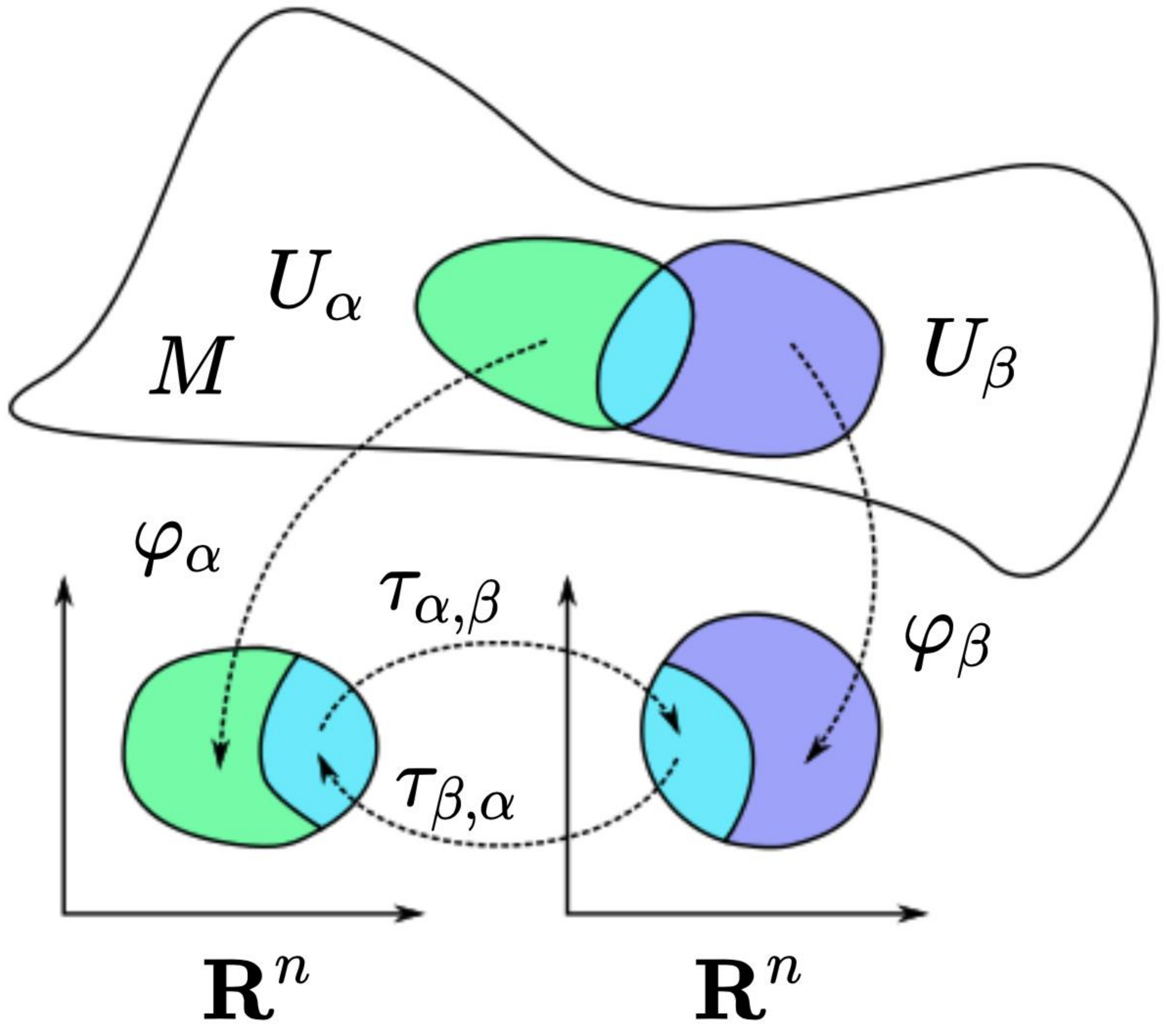
Topological manifold: topological space which locally resembles Euclidean (has a neighbourhood which is homeomorphic to R^n)

If the structure has some differential structure it is a differential manifold aka there exists some smooth function from element to element (point to point)

Connection: If the structure has some differential structure it is a differential manifold aka there exists some smooth function from element to element (point to point) on M

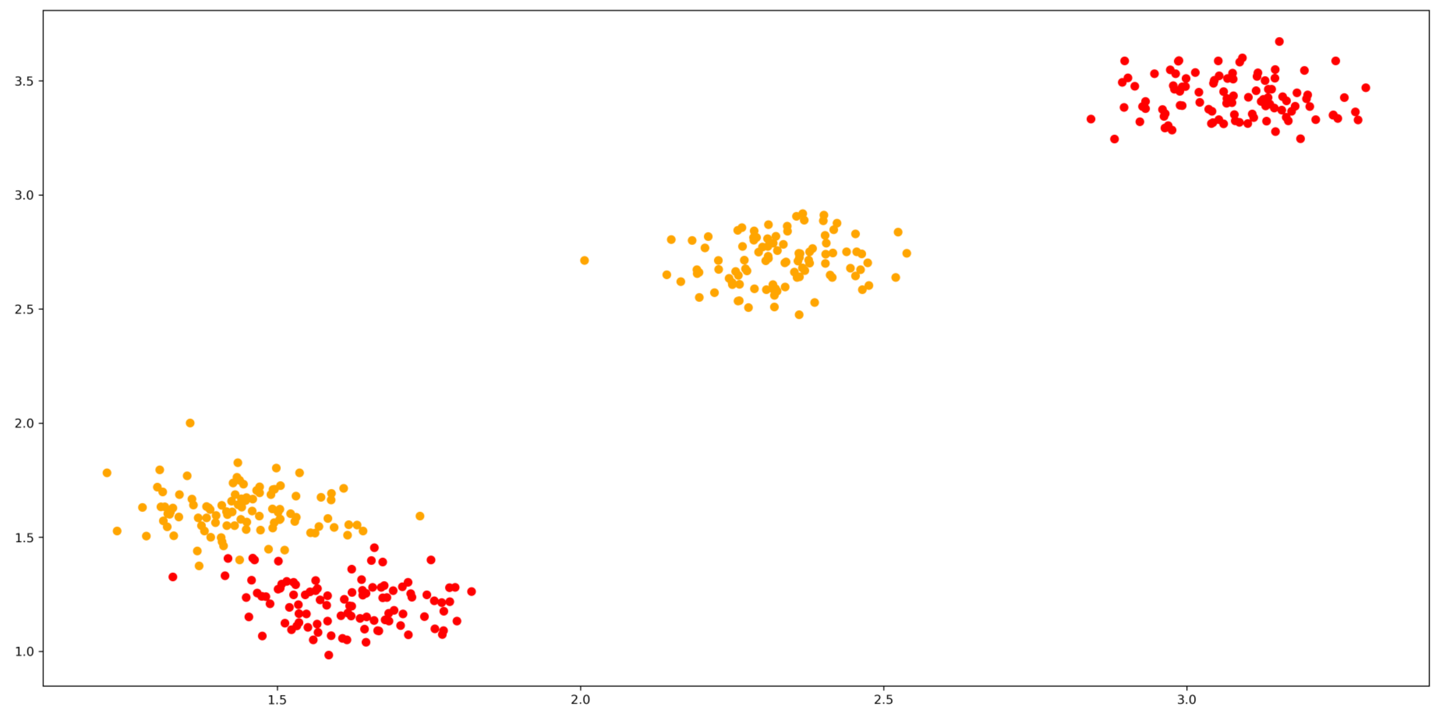
Riemann manifolds: Given a smooth (or differentiable) n-dimensional manifold M, a Riemannian metric g: TM x TM -> R on M (or TM) is a family of inner products $(\langle\bullet,\bullet\rangle_p)_{p\in M}$ on each tangent space T_pM, such that the inner product depends smoothly on p

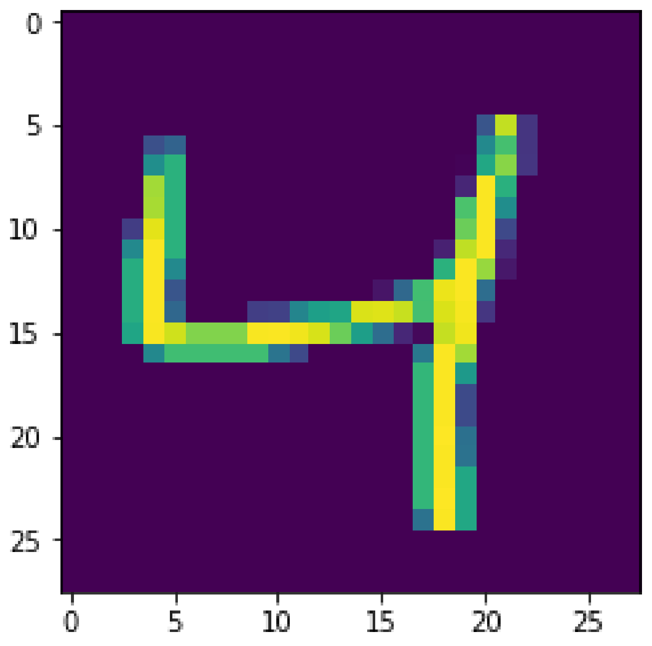
A smooth manifold M, with a Riemannian metric (RM) is called a Riemannian manifold



Manifolds or point clouds?

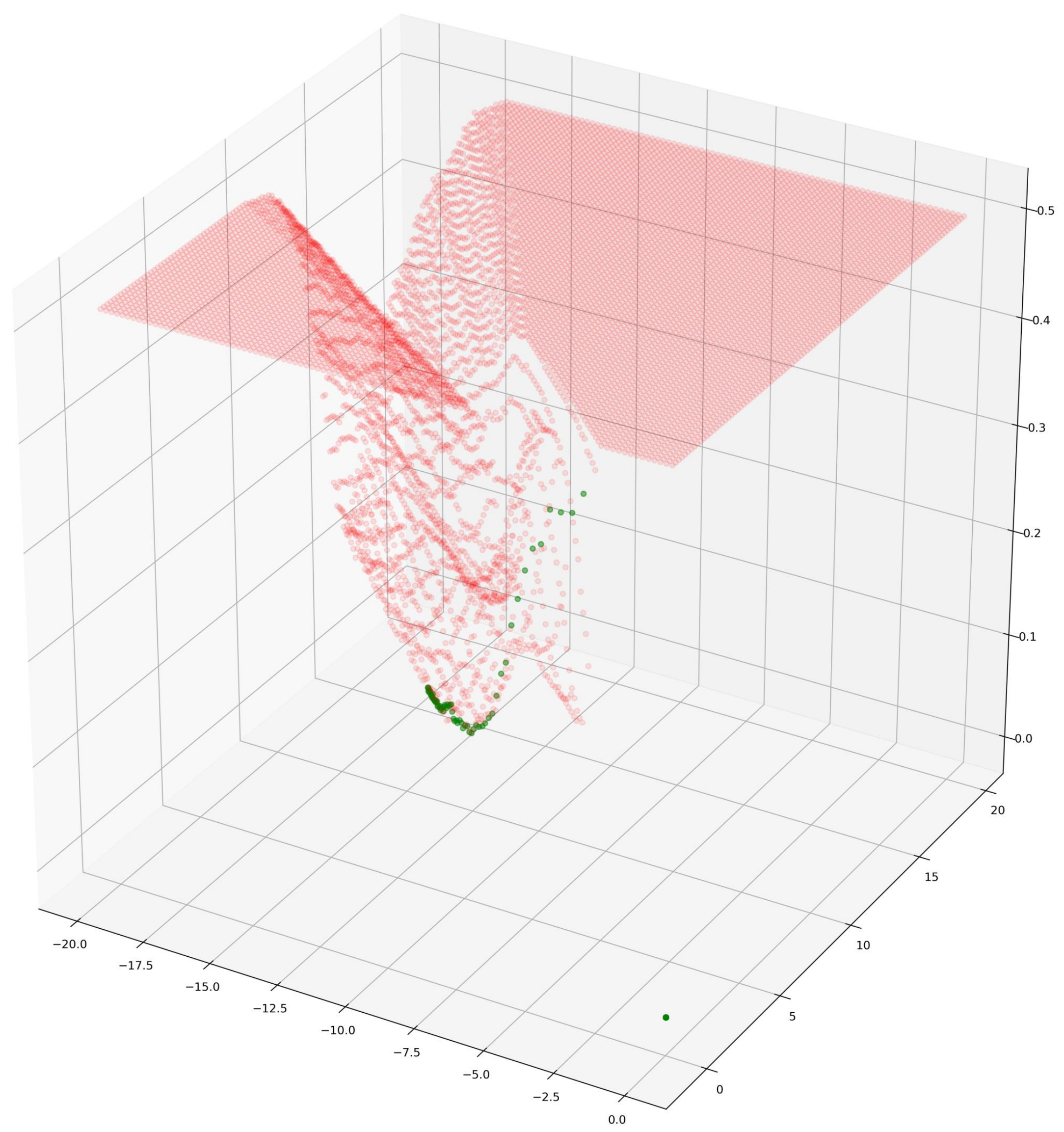
Data





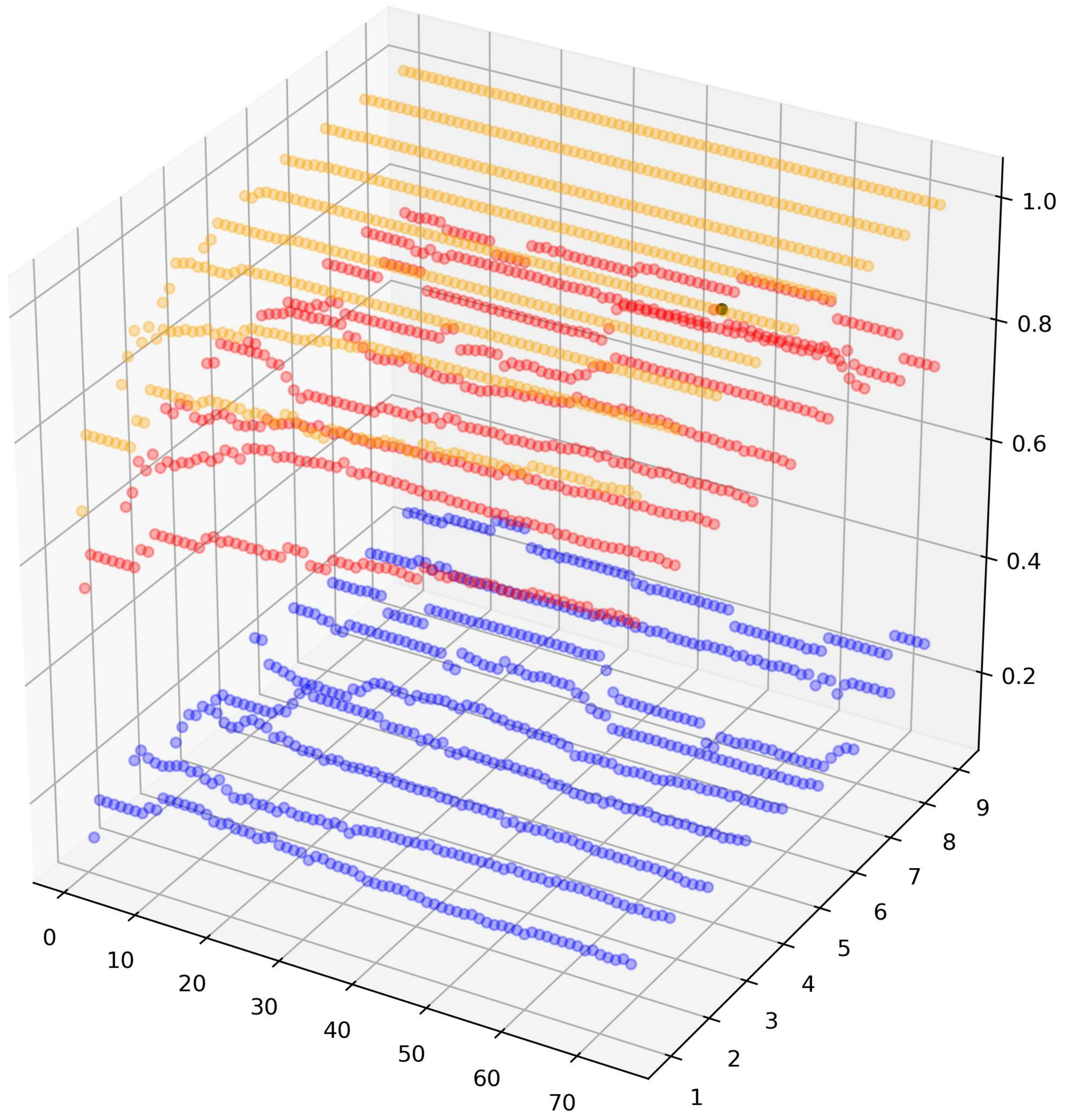
usually Not topological manifold!

Loss



usually Diff. manifold!
feed-forward NN is
[Ollivier et al, 2015,
Choromanska et al., 2015]
statistical manifolds
[Cencov, 1982,
Campbell, 1986, Amari, 1996]

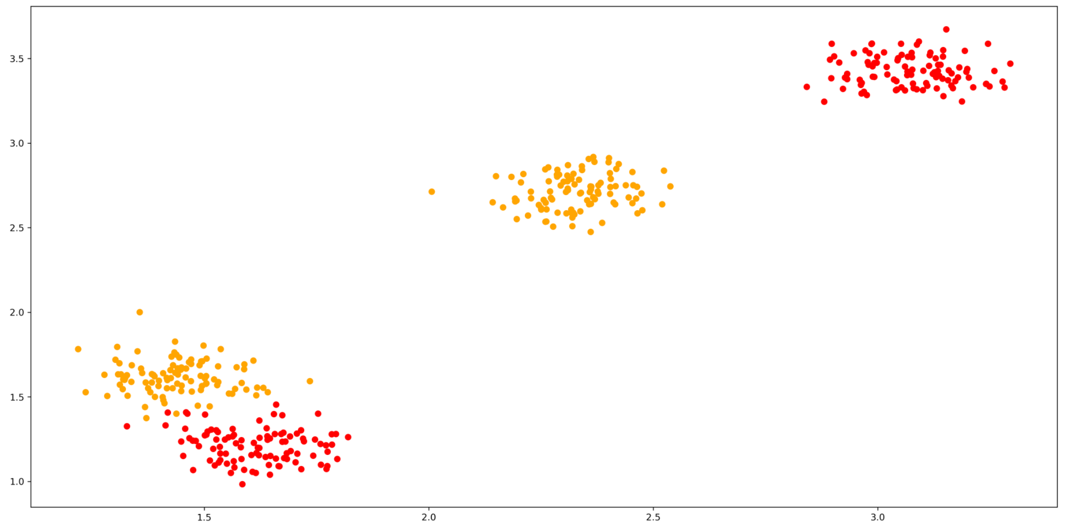
Hyperparameter surface

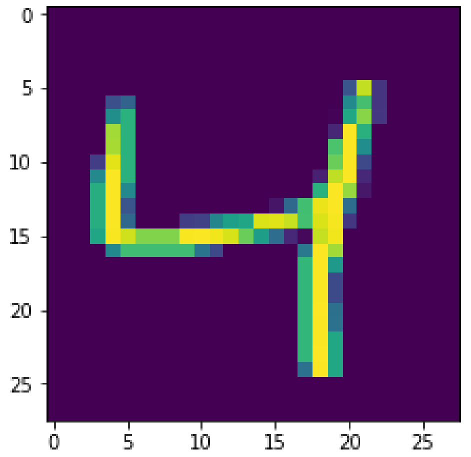


usually a set of topological manifolds but not diff.…

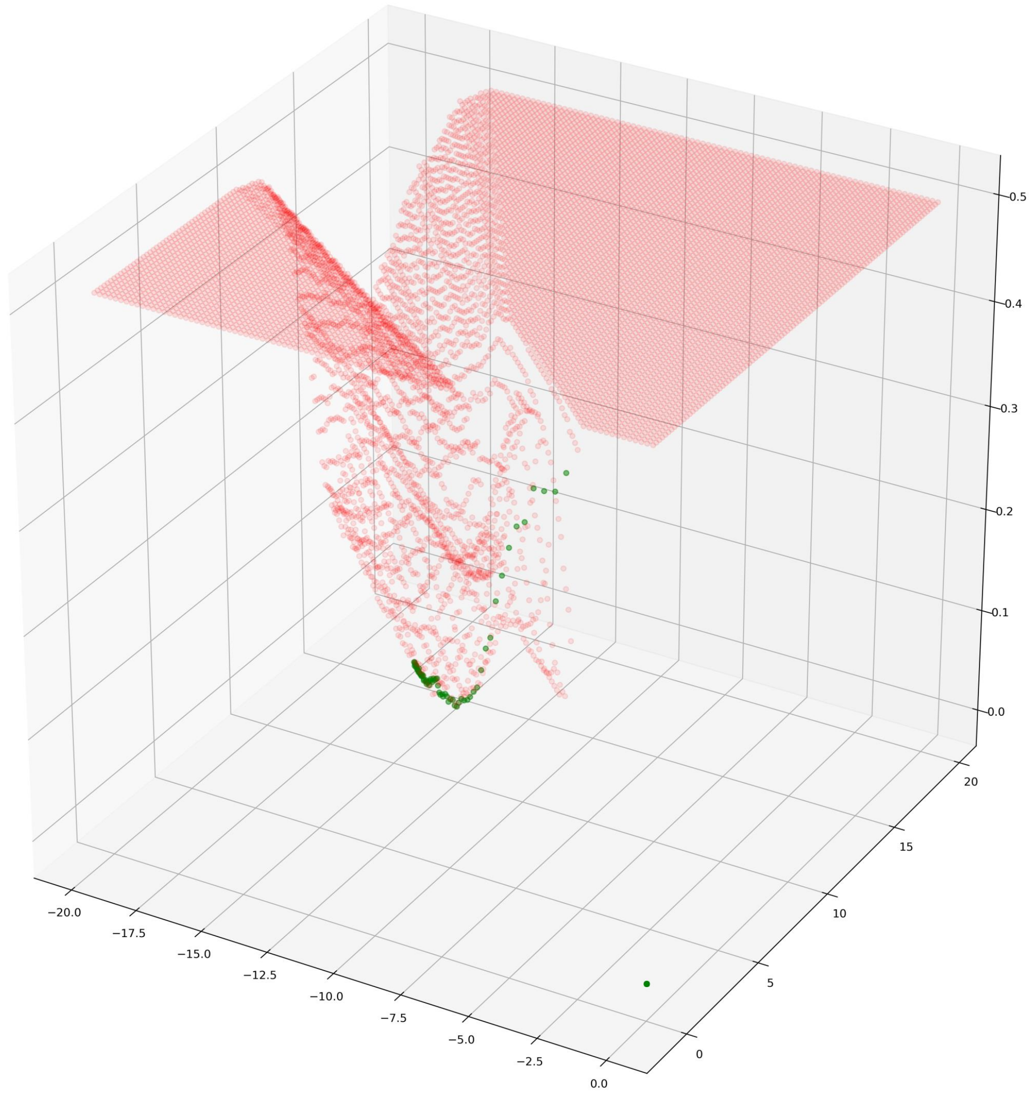
Manifolds or point clouds?

Data

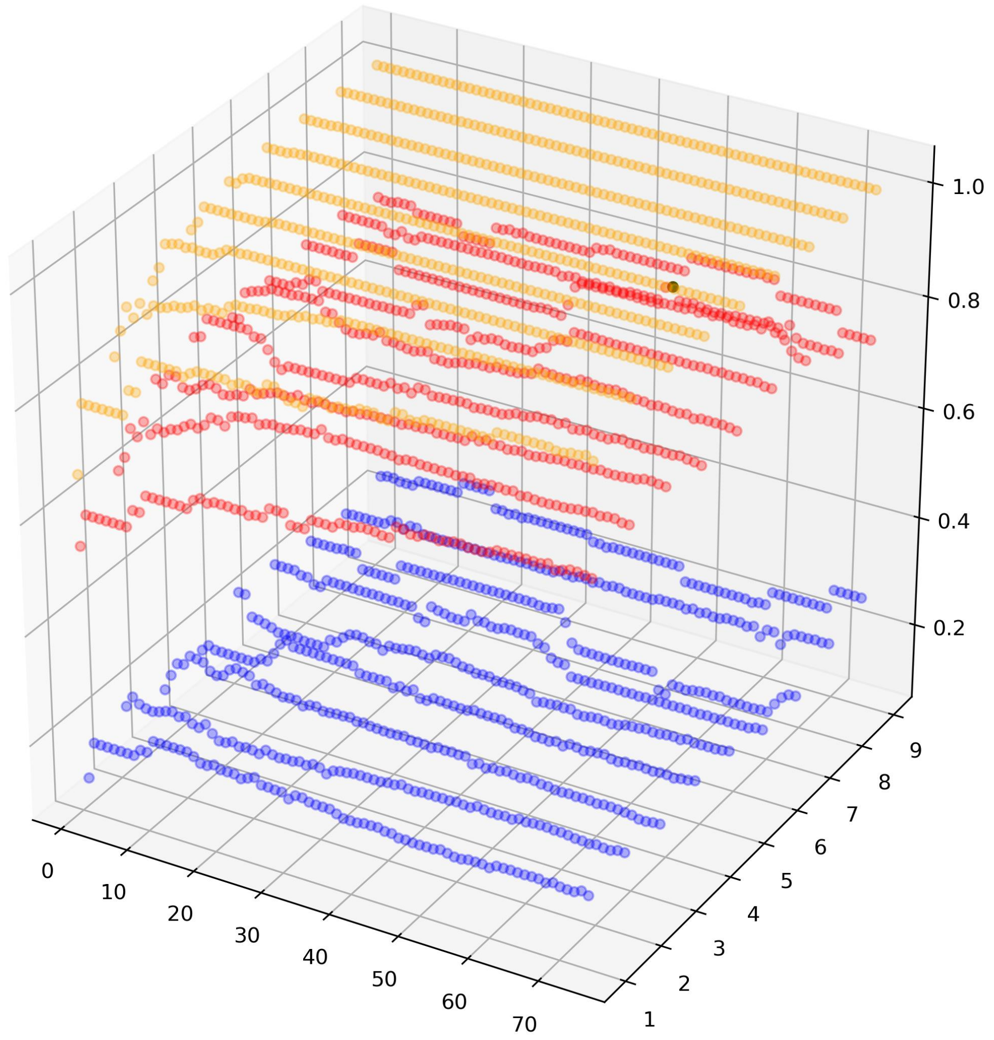




Loss



Hyperparameter surface



usually Not topological manifold!

usually Diff. manifold!

usually a set of topological manifolds but not diff.…

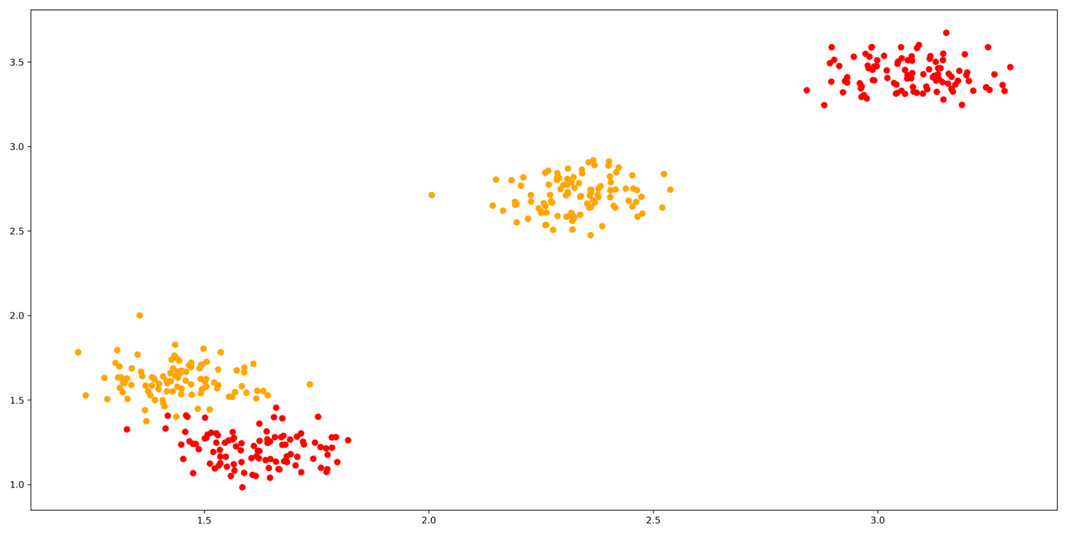
Network structure?

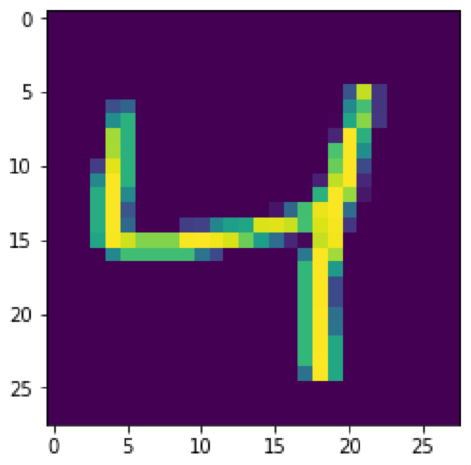
Regularization and Dropout [Hinton et al., 2012]?

Augmentation [Khrizhevsky et al., 2012]?

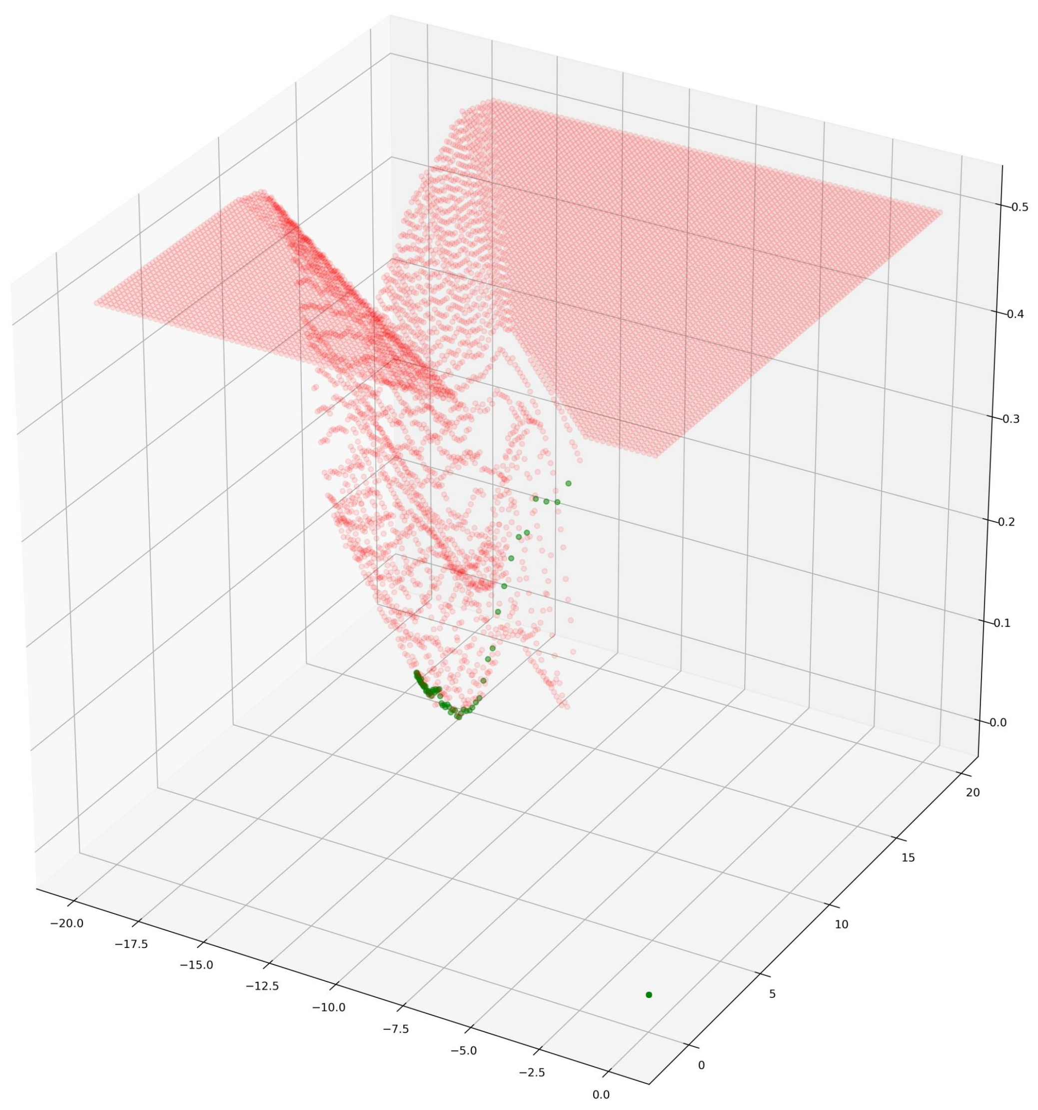
Manifolds or point clouds?

Data

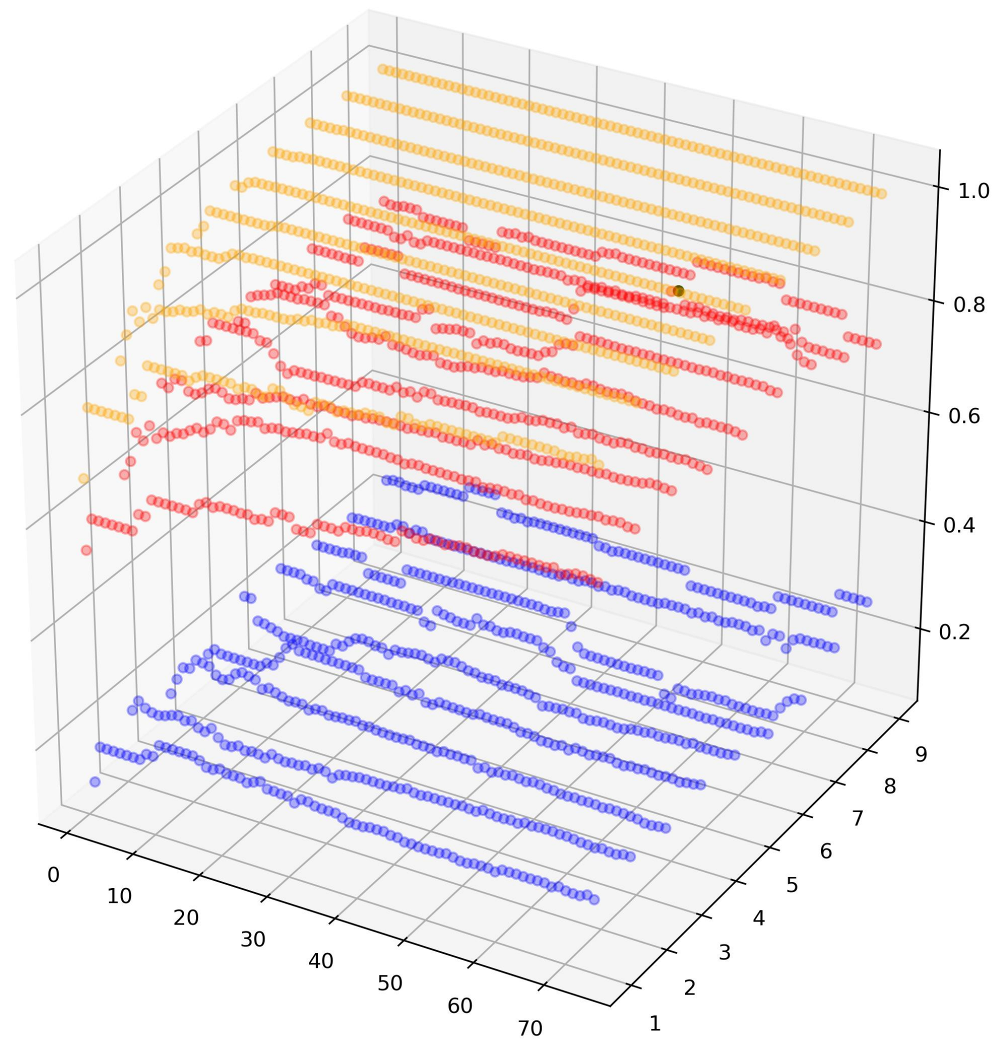




Loss



Hyperparameter surface



usually Not topological manifold!

usually Diff. manifold!

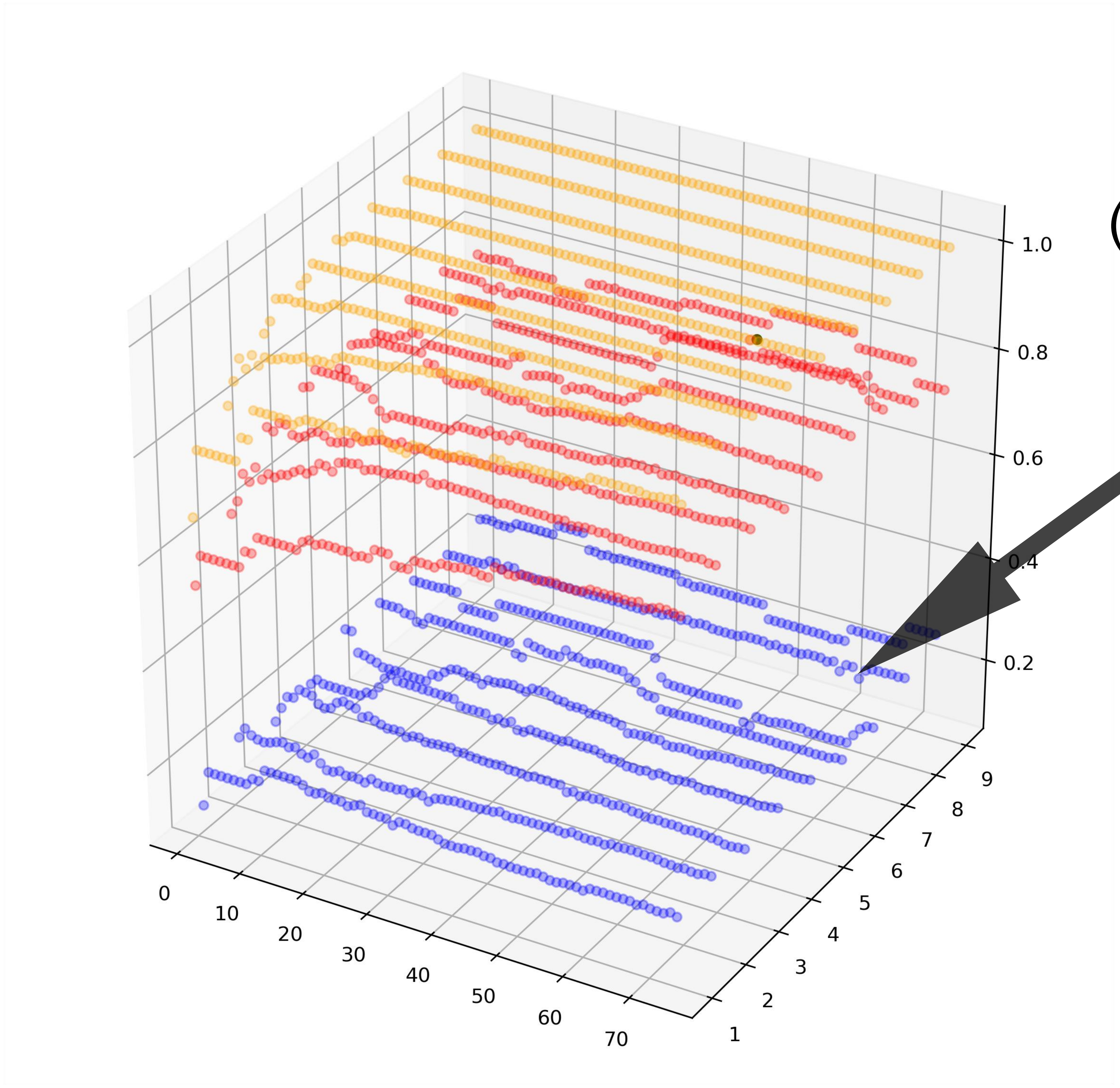
usually a set of topological manifolds but not diff.…

Augmentation [Khrizhevsky et al., 2012]

Regularization and Dropout [Hinton et al., 2012]

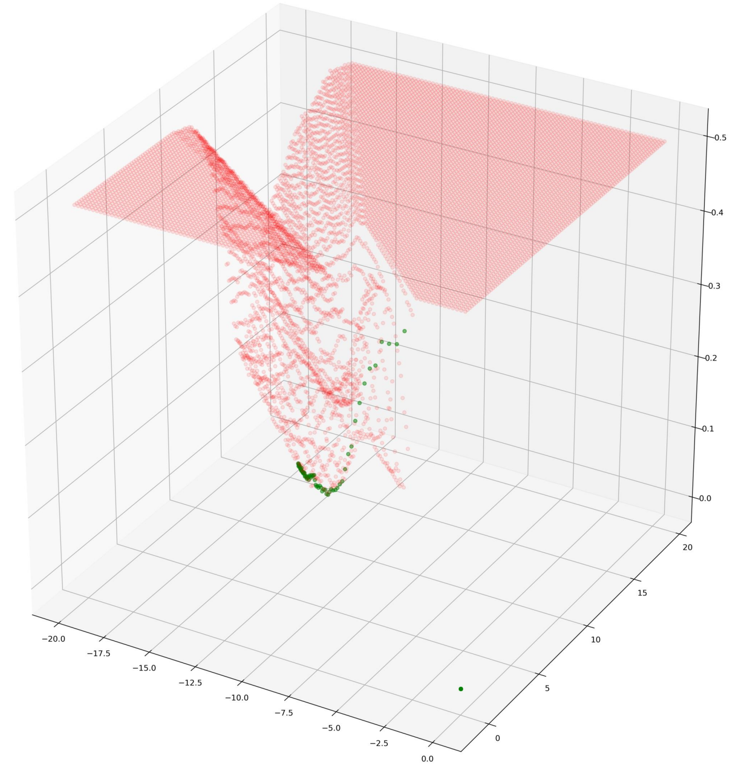
Network structure

Manifolds or point clouds?

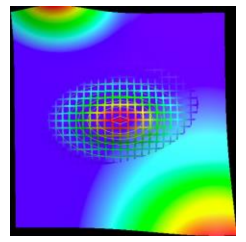


Generalization error
(difference between the empirical loss and the expected loss)
[Vapnik & Chervonenkis, 1971, Maas, 1993, Sontag, 1994, Bartlett, 2001]
as a surface?

Approx: Difference between the loss on a validation set and the loss on the training set



Now let us focus on the loss… what metric?



What metric? How to find paths on the surface?

- Simplest solution: first order GD…
  - instead of exponential maps small steps
  - momentum: e.g. [Polyak,1964], Adam [Kingma&Ba, 2014], AdaGrad [Duchi et al., 2011], RMSProp [Tieleman&Hinton, 2012], Nesterov [Nesterov, 1983]

- Semi ideal metric:
  - special metrics e.g. Fisher information [Cencov,1982, Campbell,1985, Jaakola&Haussler, 1998, Perronnin et al., 2010]
  - learn a metric to preserve some properties of the inner product locally [D. et al, 2018]

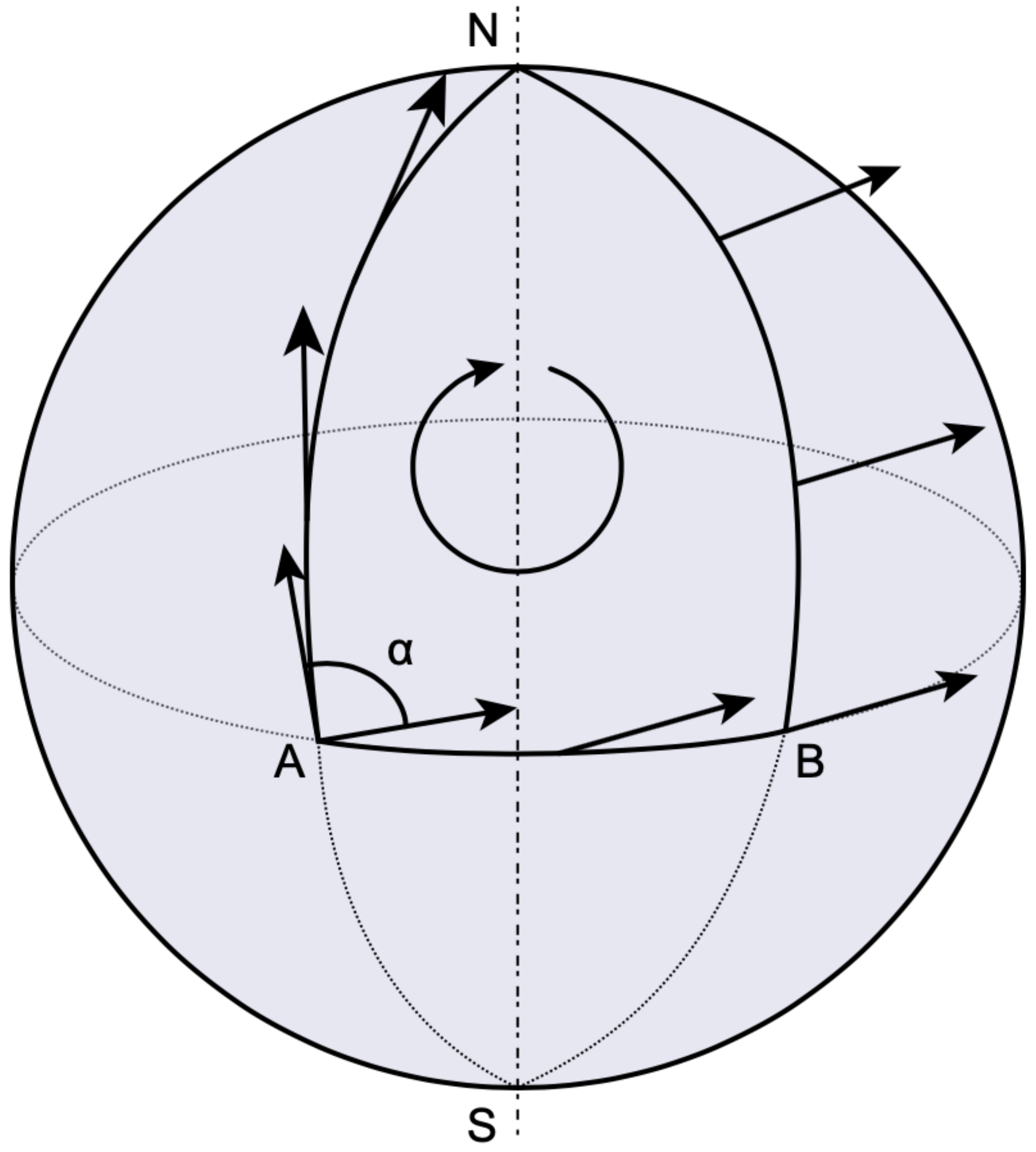


fig.: wikipedia

What metric?

- Previously determined metric

  - second order gradients?
  - Hessian metric -> pos. def.? :(

  - exponential maps
  - geodesic convexity
  [Wensing et al., 2018, Sra et al., 2018]

  - in case of loglikelihood
    - Fisher information and natural gradient:
    [Amari, 1996, Pascanu et al. 2014]

$$F(\theta) := \mathbf{E}(\nabla_\theta \log P(X|\theta) \nabla_\theta \log P(X|\theta)^T)$$

Presumption of GradNet [1]: let the loss (f) be a parametric cdf so that

- the Hessian $h_\theta(x) = H_\theta(f(x;\theta))$ exists:

$$h_{ij} = \frac{\partial^2 f(x;\theta)}{\partial\theta_i \partial\theta_j}$$

- there is a RM where g is a quasi-arithmetic mean:

$$G = g_X(h_\theta(x))$$

- and the kernel (Hessian kernel, inverse!)

$$K_\theta(x_i, x_j) = \nabla_\theta f(x_i;\theta)^T G_\theta^{-1} \nabla_\theta f(x_j;\theta)$$

satisfies the Mercer's conditions

- dot product approximation:

$$h_\theta(x) = \nabla_\theta f(x;\theta)^T \nabla_\theta f(x;\theta)$$

[1] Daróczy, B., Aleksziev, R., Benczúr, A. Sparse Hessian manifolds over feed-forward neural networks , under submission

What metric?

• Previou … hat

• sec
• Hes

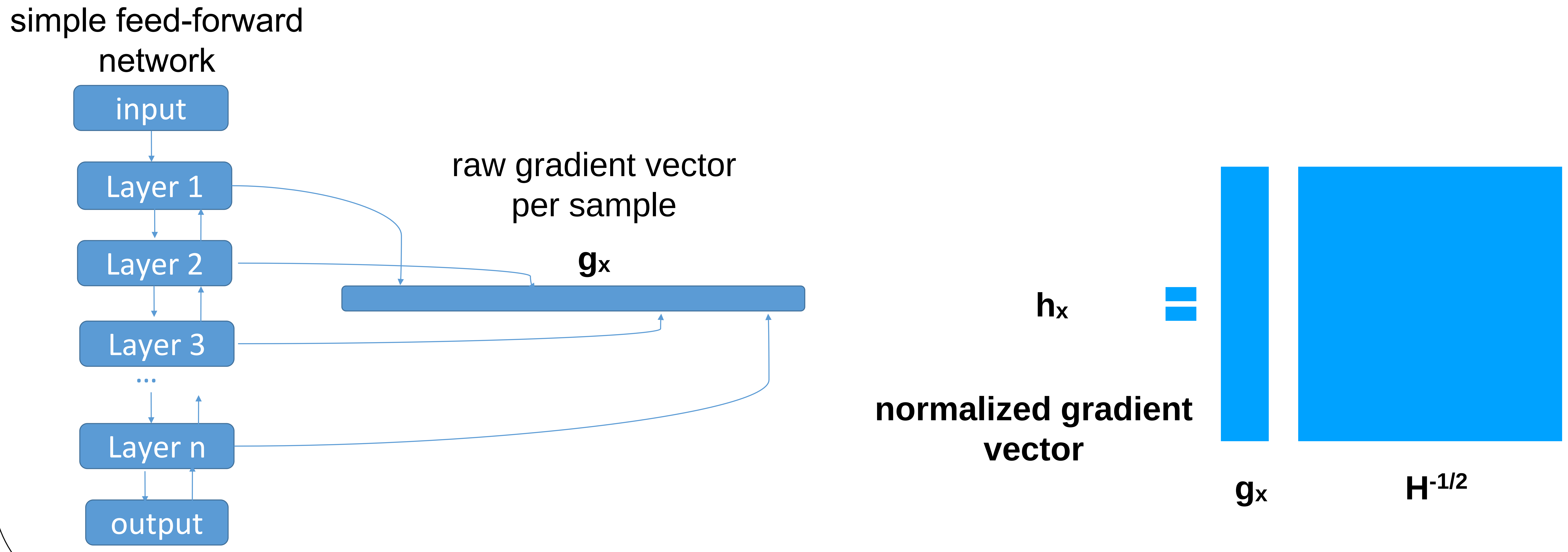
• exp … n:
• geo
[We

• in c
• F
[

$F(\theta) :=$

Dimensions:

- N = number of free parameters of the underlying model …
  - even ResNet50 has 25.2M parameters…
- dim(g_x) = N
- dim($H^{-1/2}$) = N^2 ->
  - ResNet50: in fp32 400+TB …



What metric?

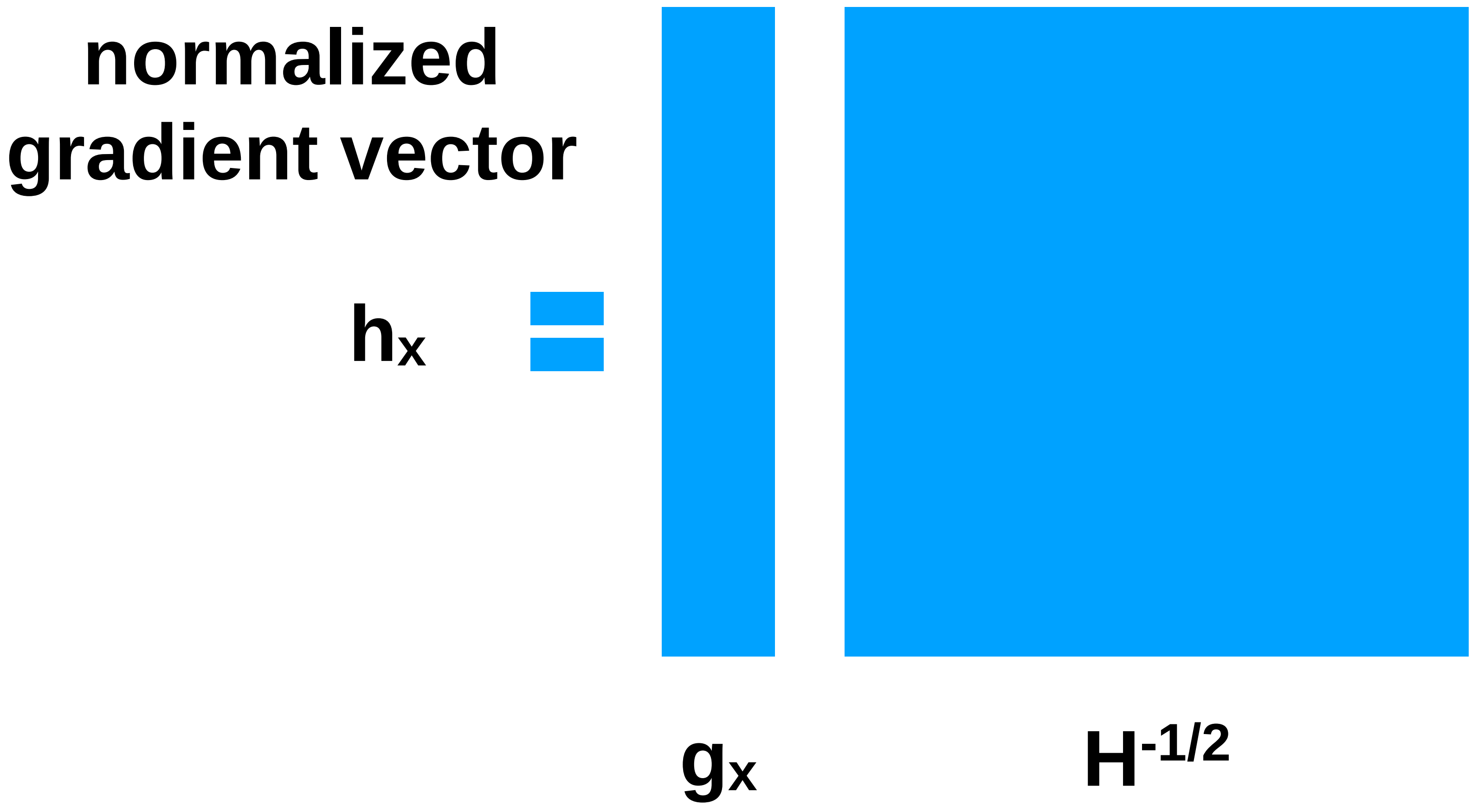
• Previou … hat

• sec
• Hes

• exp … n:
• geo
[We

• in c
• F
[

$F(\theta) :=$



Dimensions:

- N = number of free parameters of the underlying model …
  - even ResNet50 has 25.2M parameters…
- $\dim(g_x)$ = N
- $\dim(H^{-1/2})$ = N^2 ->
  - ResNet50: in fp32 400+TB…

Computational complexity:

- g_x : at every step we back-propagate per sample … existing frameworks (TF, Torch, Chainer etc.) are not suitable -> even if we have every value at some point in the GPU it will be slow over the frameworks due integrated aggregation
- H :
  - expected value (if we can store…) -> approximation via a validation set
  - inverse computation … -> diagonal approximation

or?

What metric?

- Previou … hat
  - sec
  - Hes
  - exp
  - geo
  [We
  - in c
    - F
    [

$F(\theta) :=$ … an:

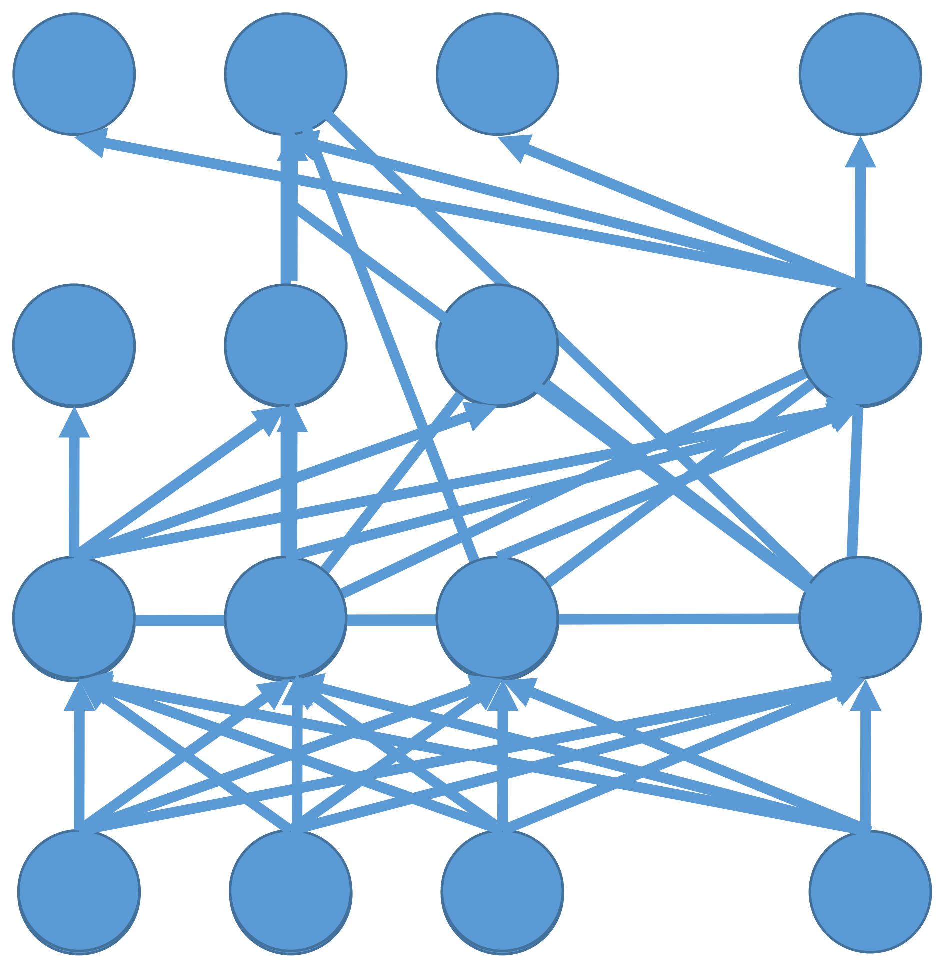
normalized gradient vector

h_x =

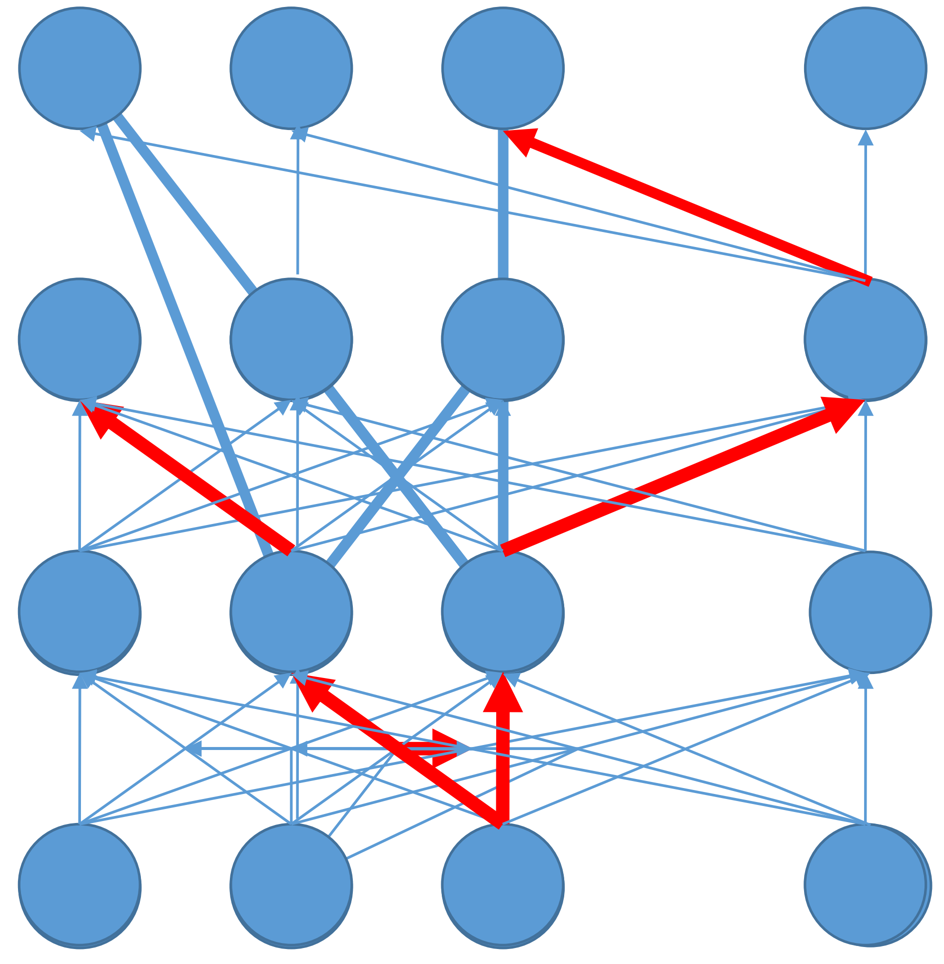
g_x **$H^{-1/2}$**

Dimensions:

- N = number of free parameters of the underlying model …
  - even ResNet50 has 25.2M parameters…
- dim(g_x) = N
- dim($H^{-1/2}$) = N^2 ->
  - ResNet50: in fp32 400+TB …



”important” parameters



“surviving” paths

Hypothesis: Sparse Hessian <-> sparsity and invariance inside the Deep NN

Identify flows -> sparse gradient

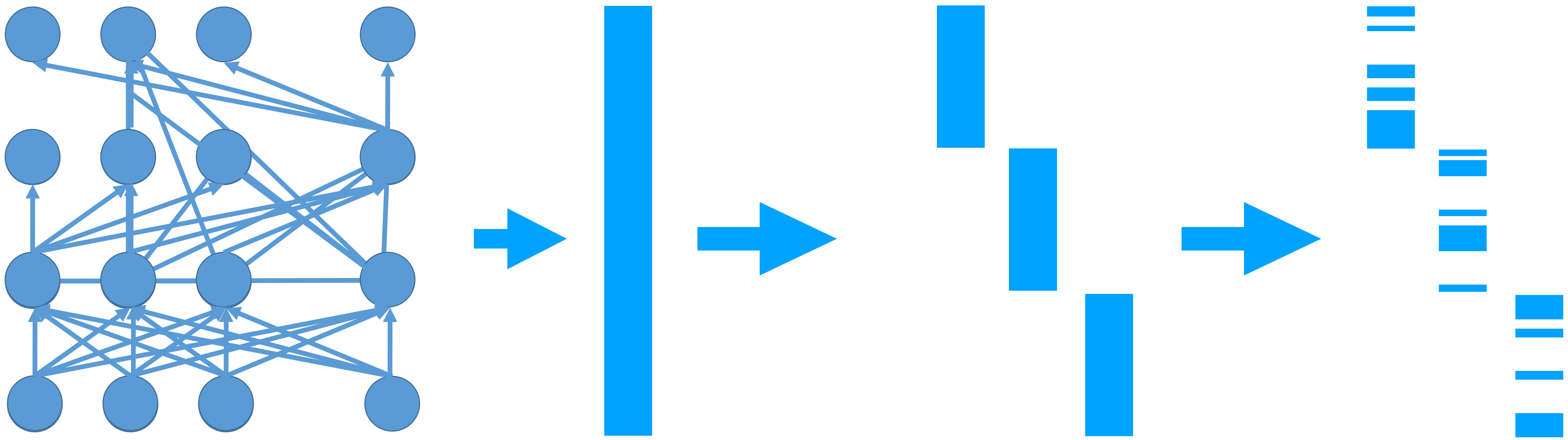
What metric?

Sparsification of the gradients:

Hierarchical nature of feed-forward networks

- Percentile rank per layer… on GPU? CuPy for Chainer
- Leave only a fraction of the gradient vectors per layer

Experiments: 15% of the gradient is sufficient



• Previou ... hat

• sec
• Hes

• exp ... an:
• geo
[Wer

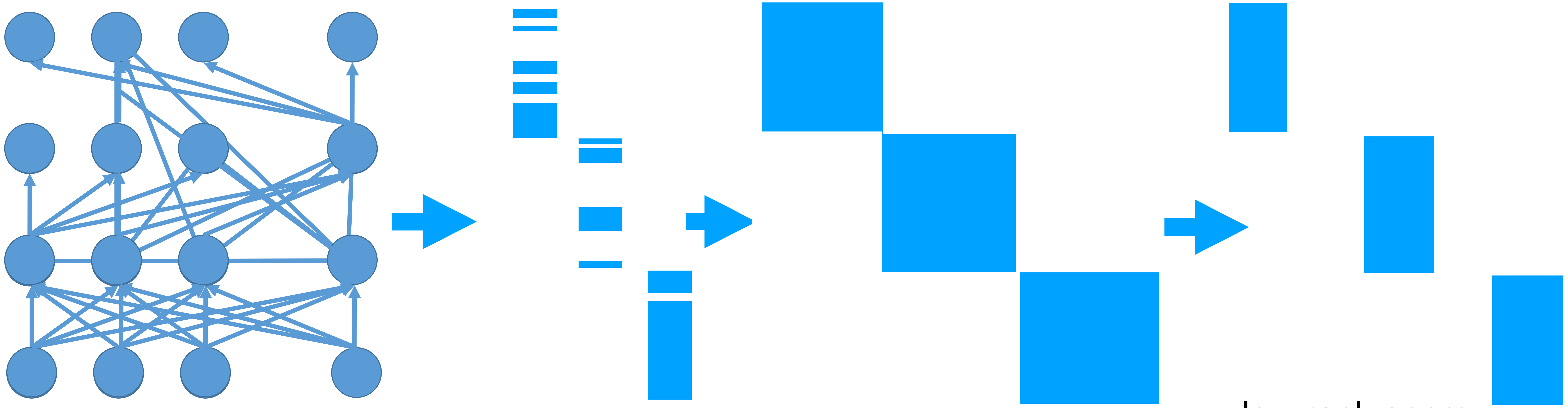
• in c
• F
[

$F(\theta) :=$

Sparsification of the metric:

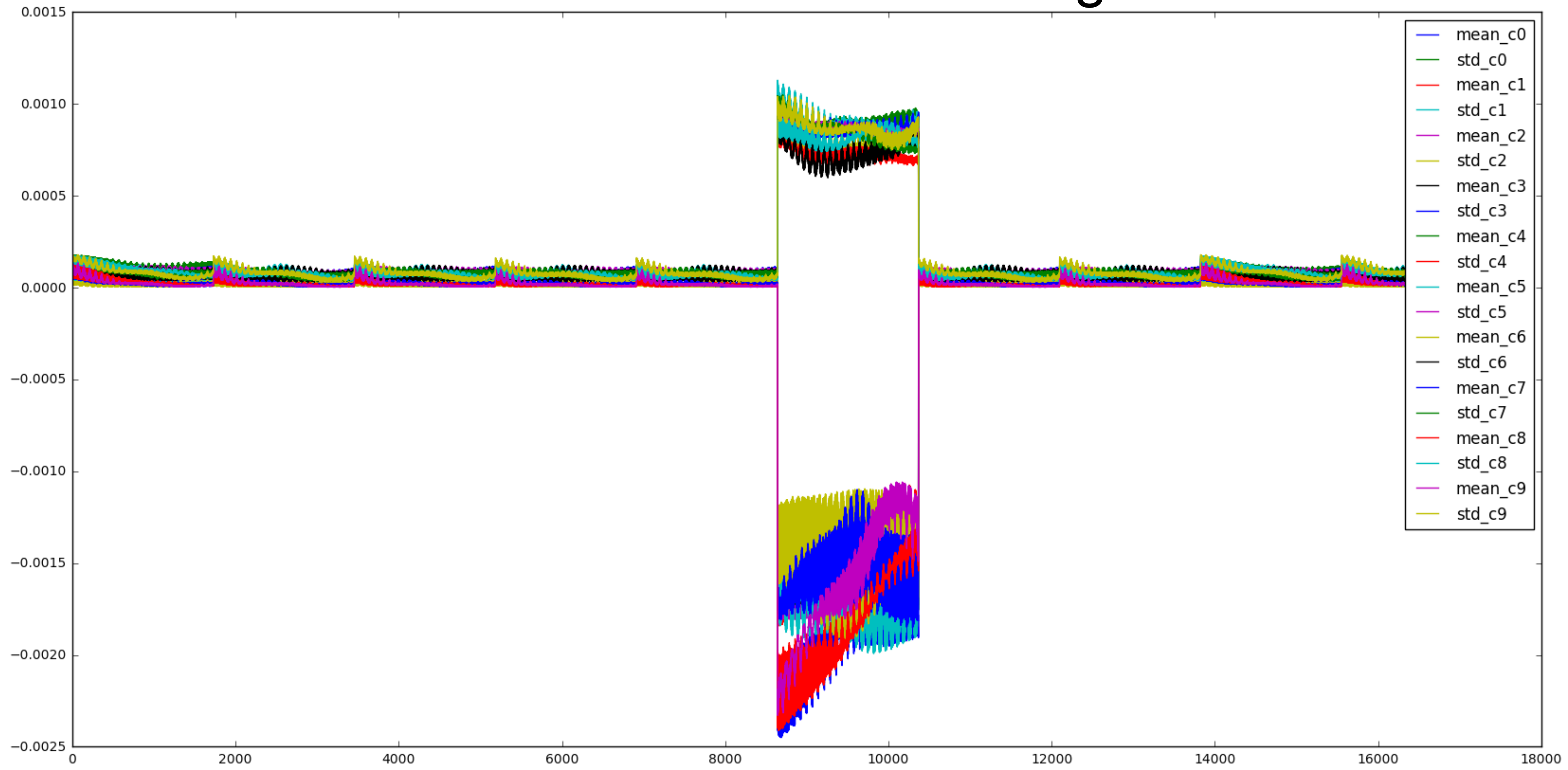
- Block nature (with proper activations) of Hessian over feed-forward networks -> important blocks
- dot product approximation and sparsification -> per sample the Hessian is highly sparse
- finite approximation via validation set and highly sparse Hessians -> fraction of the metric we need to care about
- inverse?

Hypothesis: the important blocks can be well approximated with low dimensional latent factors… -> multi-layer perceptron per layer

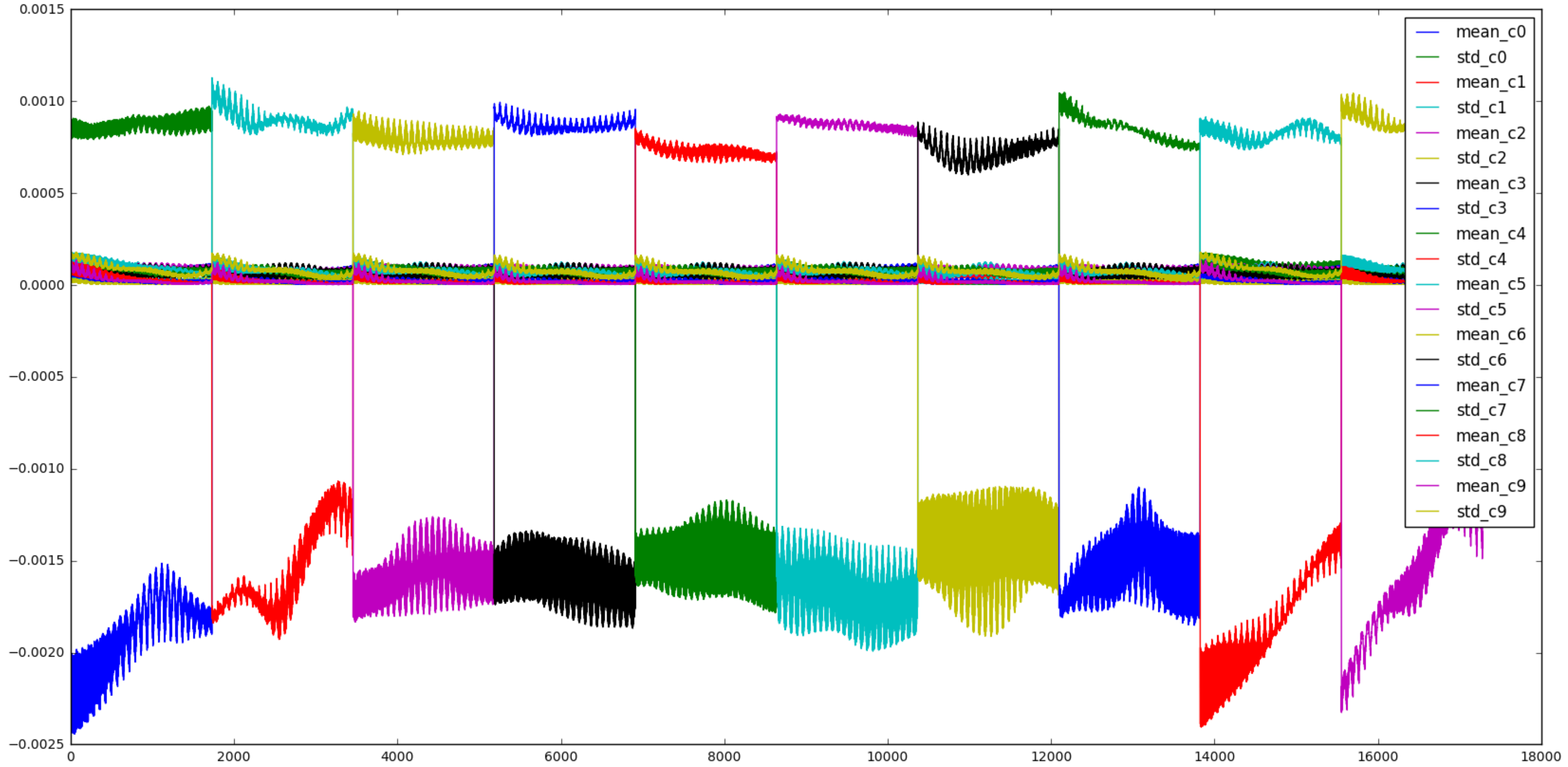


Ok, what are the partial gradients of the loss of unknown labeled sample?

Gradient based on fixed target loss

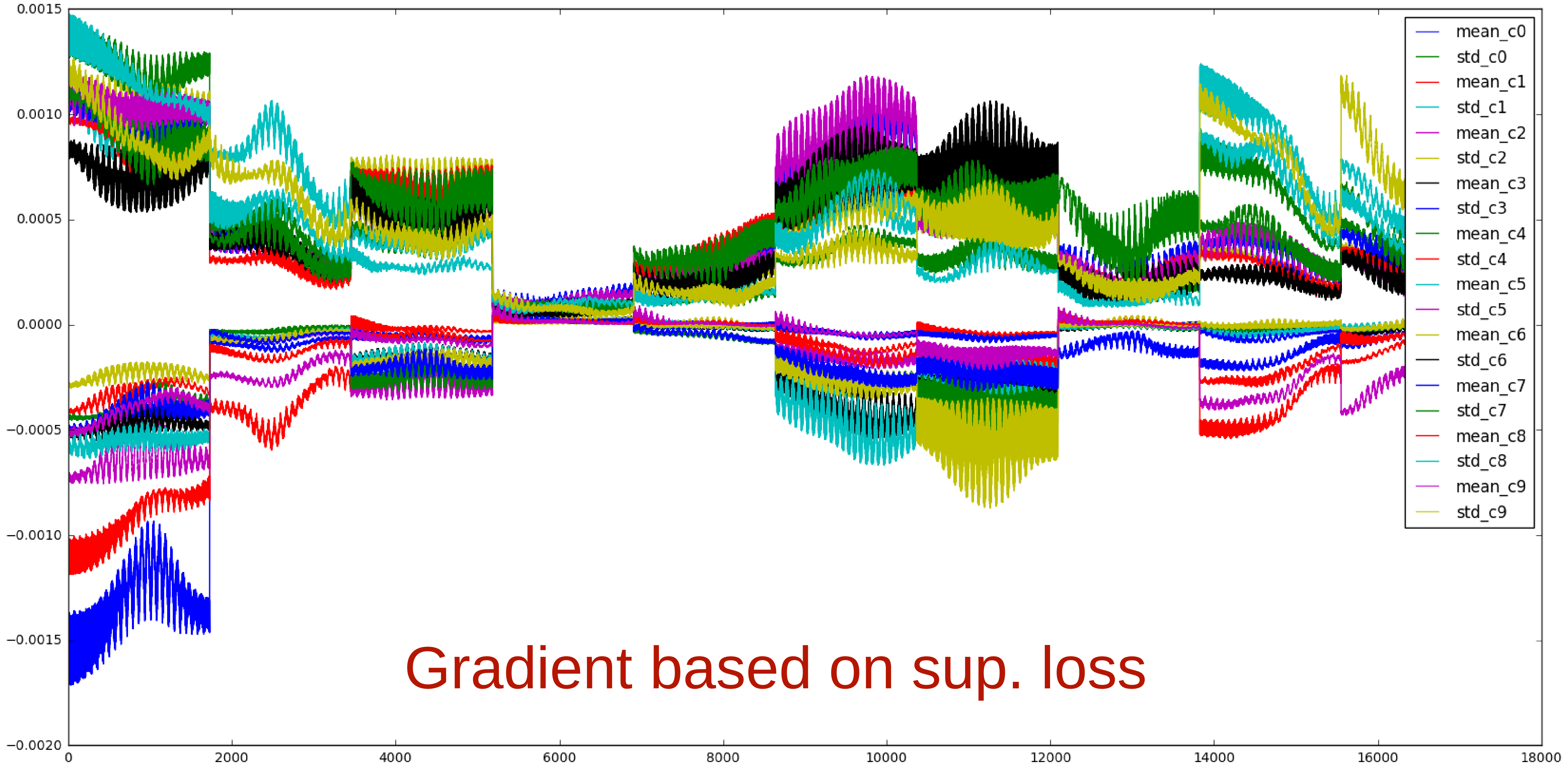


Gradient based on GT loss



Gradient representation:
what is the loss for a test sample?
We do not know the original label…

Colors indicate ground truth class
Discriminative layer on test



Gradient based on sup. loss

Sadly, the best solution is so far is the mean ….

GradNet: Experiments with a simple 3-layer CNN with 120k parameters on CIFAR and MNIST

Table 1: Performance measure of the normalized gradient based on RBM.

MNIST		
#hidden	Original	Improved
16	0.6834	0.9675
16	0.8997	0.9734
64	0.872	0.9822
64	0.9134	0.9876

Table 2: Performance measure of the improved networks.

CIFAR		
Original	Improved	Gain
0.79	0.8289	+4.9%
0.76	0.8201	+7.9%
0.74	0.8066	+9%
0.72	0.7936	+10.2%
0.68	0.7649	+12.5%
0.65	0.7511	+15.5%
0.62	0.7274	+17.3%
0.55	0.7016	+27.5%
0.51	0.6856	+34.4%
0.49	0.678	+38.3%

MNIST		
Original	Improved	Gain
0.92	0.98	+6.5%
0.96	0.9857	+2.7%
0.9894	0.9914	+0.2%

Figure 3: Optimization methods

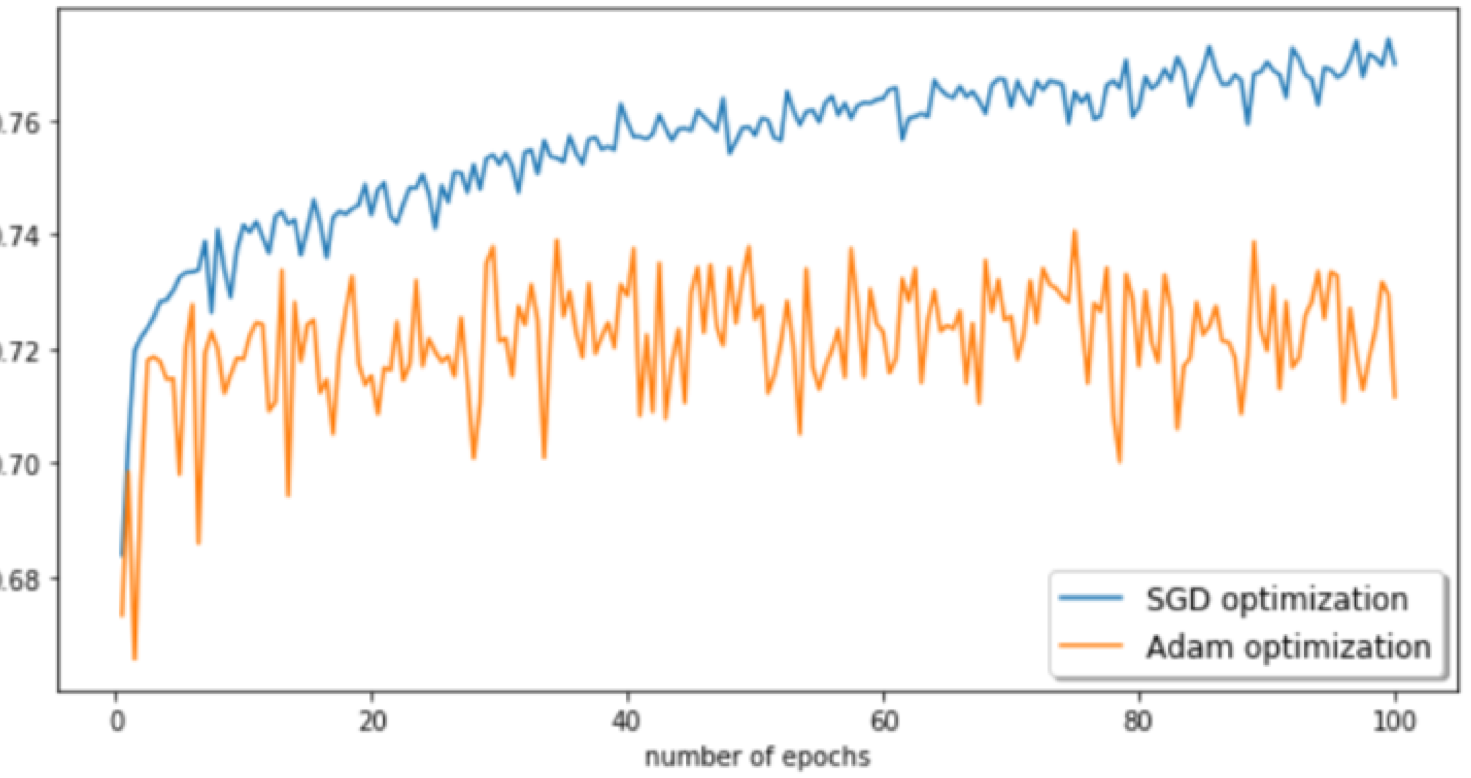
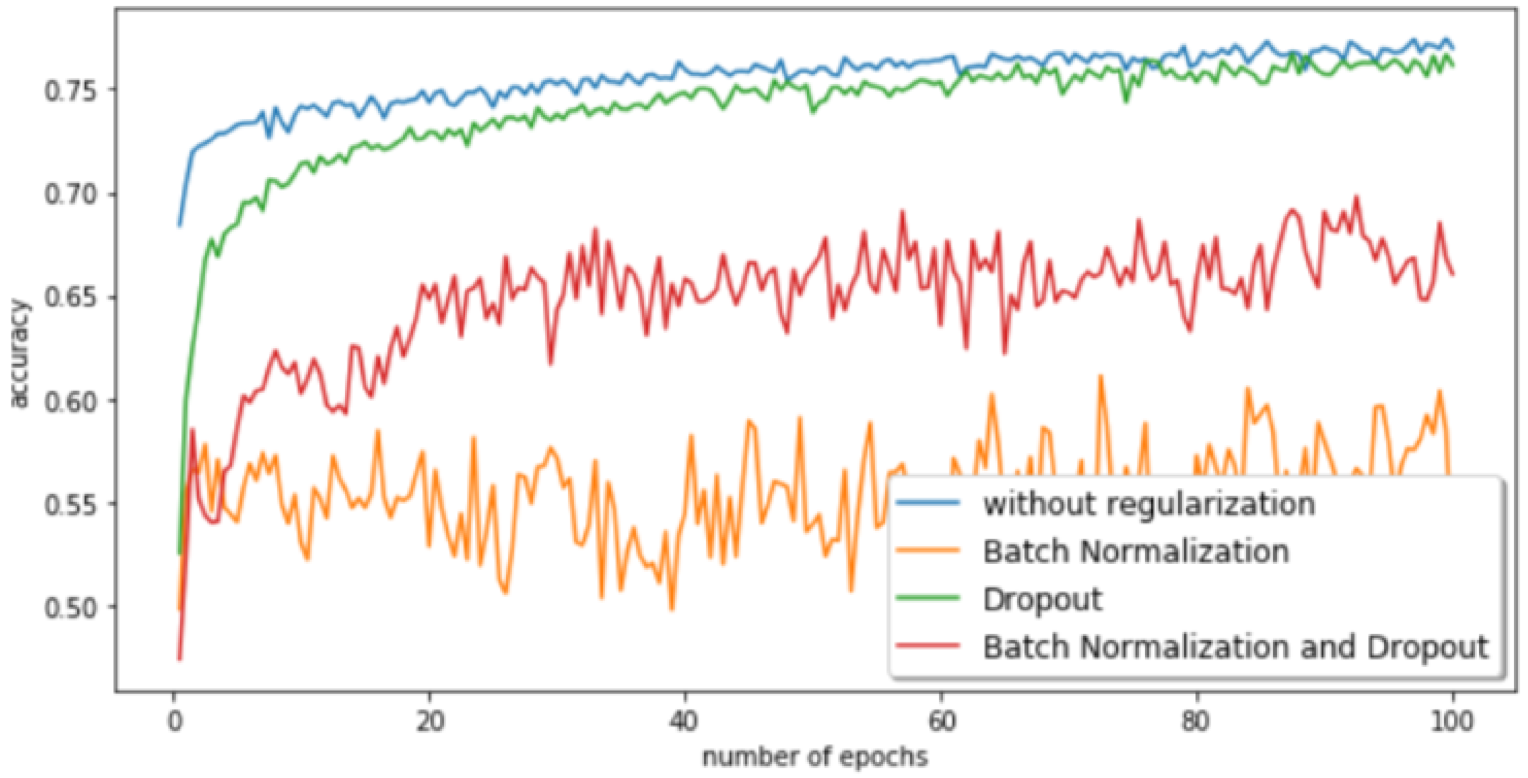
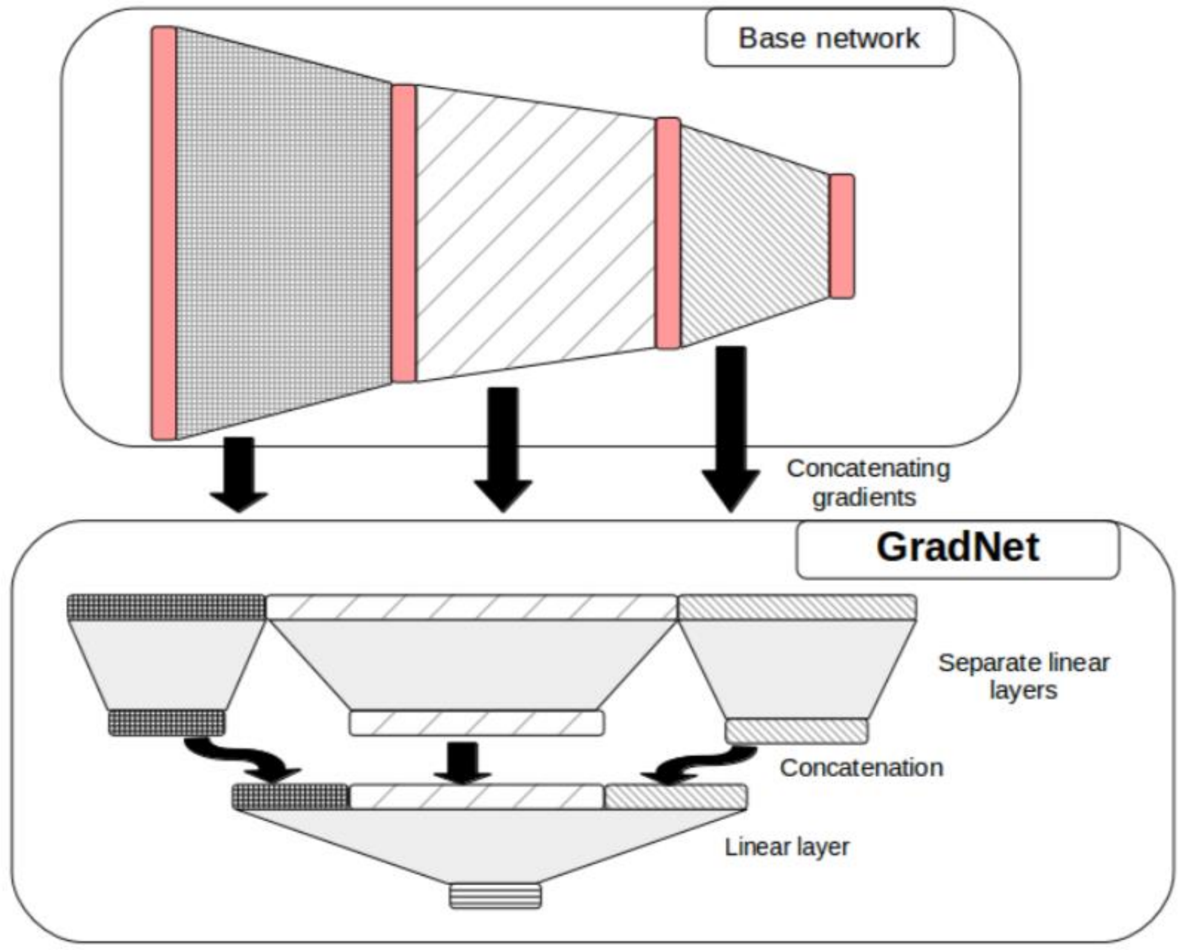


Figure 4: Regularization methods





Conclusions

- GPU memory can be really a constrain even for simple problems
- Sometimes these constraints lead to interesting results (e.g. CIFAR-10 accuracy 0.828 vs. 0.801 with and without sparsification)
- Differential geometry could play an important role in the future of ML [Hyland and Ratsch, 2016, Wisdom et al., 2016, Ox et al., 2017]:
  - Pushforward, local diffeomorphisms (lower or higher dimensional, but a sparser tangent space): -> non trivial network structures? Ensemble of structures.
  - Lie groups -> left/right/bi-invariant transformations

Thank you!