Non-Convex optimization and shape of minima

ATML Track1

Rik Sarkar

Today

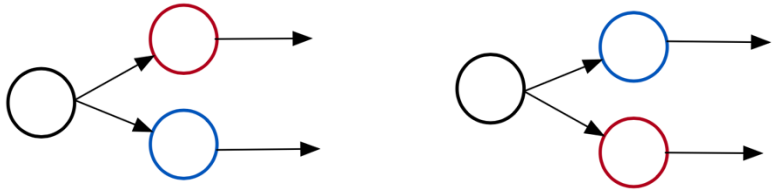
- Symmetry and non-convexity in neural network optimization
- What we know about the minima
- How SGD finds minima and intrinsic dimensions
- Stability of SGD with non-convex loss

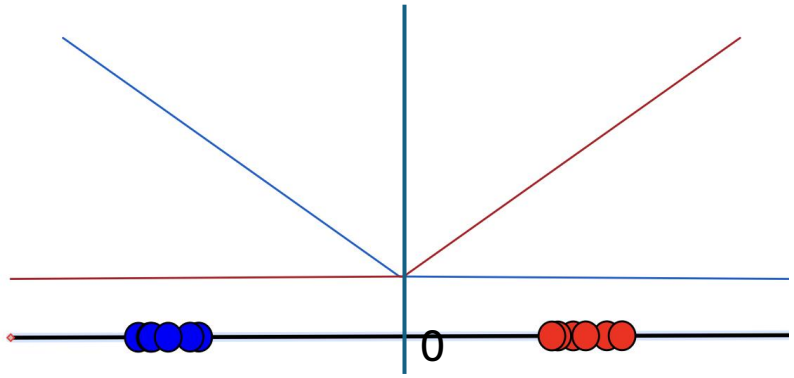
Neural networks are symmetric and have many minima

- Take any two neurons a and b in a hidden layer
- We can swap all the incoming and outgoing weights from one to another
- The modifed neural network is functionally identical to the first one
- So, a NN can have many minima obtained by swapping neurons.
  - They are all equally good!

NN are non convex – simple example

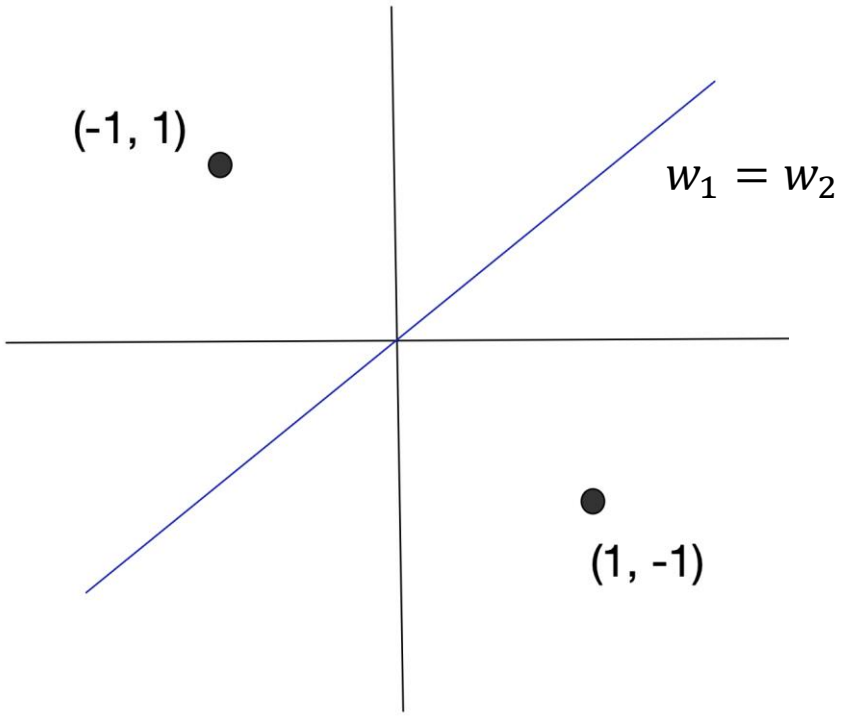
- Take a 1-D input and two classes
  - Suppose we set bias = 0
- There are two types good configurations with low loss
  - $w_1 = 1, w_2 = -1$
  - $w_2 = -1, w_1 = 1$
- But the intermediate configuration
  - $w_1 = 0, w_2 = 0$ ()
  - Is bad – high loss





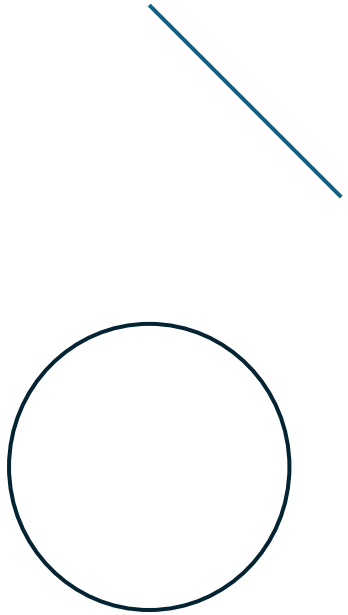
Intermediate configurations are high loss

- All configurations along $w_1 = w_2$ are bad
  - Check this for yourself. Why is it true?
- Therefore the neural network has two unconnected minima
  - Non-convex
- Connected minima can also occur in non-convex loss landscape
  - E.g. If they are not along a straight line



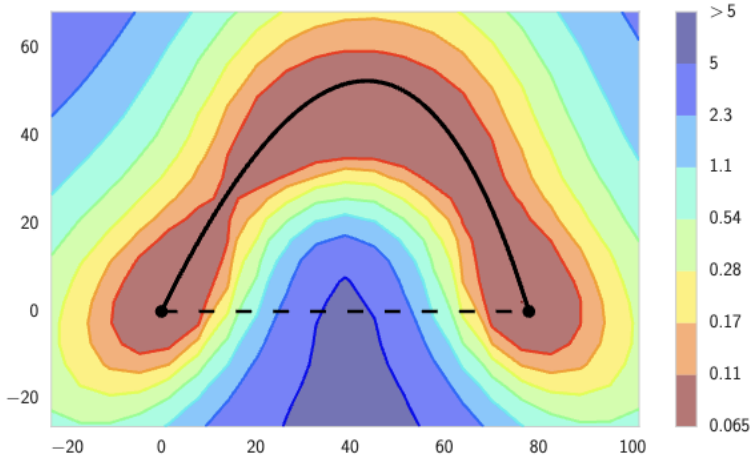
Intrinsic and Ambient Dimensions

- Intrinsict dimension at a point at an object is its dimension locally at each point
  - If this value is same everywhere, we can call it the intrinsic dimension of the object
- Ambient dimension is the dimension of the space where it is contained (embedded)
- E.g. A straight line in a plane
  - Intrinsic dimension = 1
  - Ambient dimension = 2
- Circle in a plane?



Shape of minima for neural networks

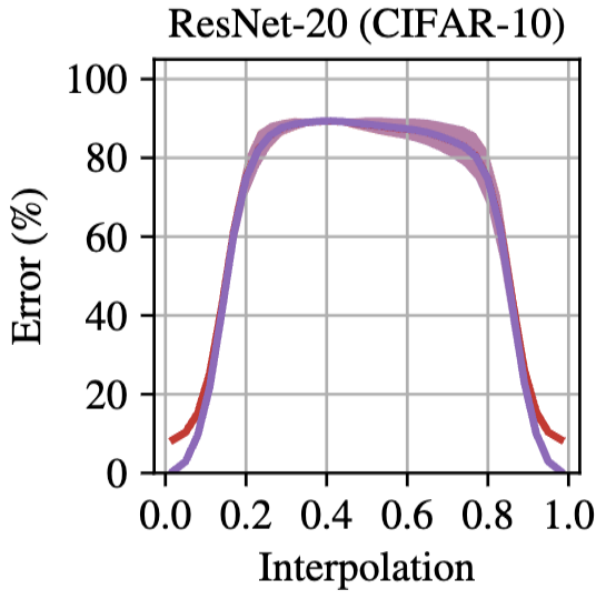
- Minima can occur in sets of connected components – connected by curved lines
  - “Minima” interpreted loosely as points with fairly low value close to the true local minima in this region
- These sets are of high intrinsic dimension,
  - But lower than the ambient dimension

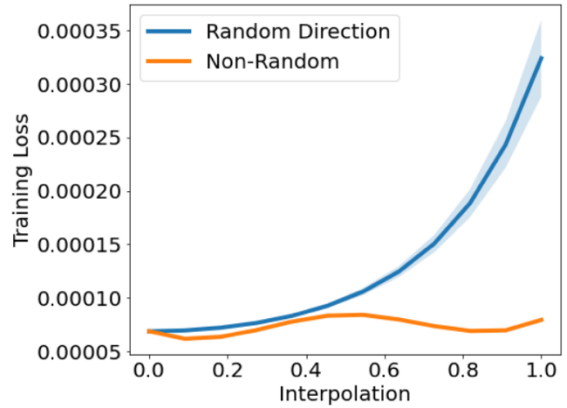


A region of min. in Resnet 164, with CIFAR-100. L2 regularized cross entropy loss. Garipov et al. 2018

More on shape of minima

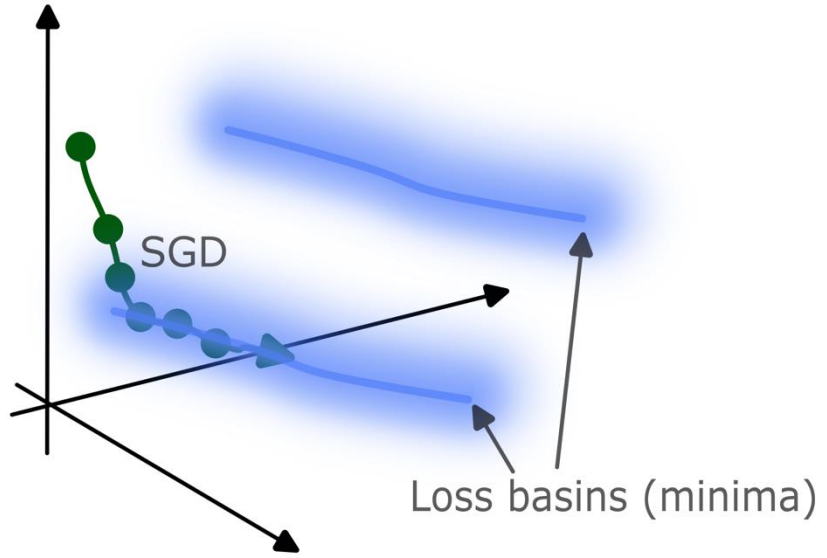
- Take two models computed by SGD from two different starting points.
- Compute the Loss or error of models along the straight line joining them
- The error rises rapidly from each side to a plateau in the middle
- At the minima, the loss rises rapidly in random directions, but stays flat along the flat min





Conclusion

- Inside the high dimensional parameter space
- Minima occur in thin subspaces like ravines (narrow basins)
  - With a flat bottom
  - Flat means the minima span several directions (dimensions)
  - But still has lower intrinsic dimension than ambient
- SGD picks early on a ravine near the starting point
  - Then tries to find a good minimum inside
- Observe: The desirable space is small part of the ambient space (due to low dimension).



Comment about subspaces and manifolds

- We referred to the relevant sets (e.g. minima) as subspaces
- In ML these are frequently called manifolds
- However, we have no guarantee that these are actually manifolds
- Manifolds have specific properties
  - E.g. same dimension everywhere
  - And we cannot be sure that these are always satisfied.

Dimension vs generalisation

- Idea: In the later stages of optimization, SGD should be exploring a low dimensional space (i.e. the space containing minima)
  - Low dimension implies good models
- If the last few points (models) found by SGD lie in a low dim. space, it is likely that it has found a good min
  - Use the dimension of space spanned by last few steps to predict generalization
- If the dimension is high, that means there are points pulling the process away from a low dimensional space – likely poor generalization

Stability of SGD

- We saw that strongly convex losses lead to stable algorithms
- Now let us see a result about non-convex losses
  - [Hardt et al. 2016. Train faster, generalize better: Stability of stochastic gradient descent]

Stability of SGD (non-convex)

- Suppose loss function f is L-Lipschitz, β-smooth
- There are n data points
- SGD takes T steps
- Step size (learning rate) is $\alpha_t \leq \frac{c}{t}$
- Then SGD has uniform stability with
- $\epsilon_{stab} \leq \frac{1+\frac{1}{\beta c}}{n-1}(2cL^2)^{\frac{1}{\beta c+1}} T^{\frac{\beta c}{\beta c+1}}$
- Omitting constant factors: $\epsilon_{stab} \leq \frac{T^{1-\frac{1}{\beta c+1}}}{n}$

General idea of stability proof/analysis

- Suppose item i is different between S and S^i
- Imagine running SGD on S and S^i with same initialization and permutation (ordering) of data
- Divide into
  - Phase 1: before i is encountered: processes are identical
  - Phase 2: after i is encountered

- Phase 1: Likely to be reasonably long
  - Expected position of i is $n/2$: unlikely that i will be encountered very early
- At end of phase 1: $\alpha_t \leq \frac{c}{t}$ is small (assuming n is large)
- Phase 2: only small steps in $\leq \alpha_t L$
- The two processes cannot diverge too far

Additional Intuition

- Remember the previous observations about SGD
- By the time the changed element is encountered, SGD has found the same loss basin thus, a single item does not change much.