

---

# MLDS HW1-3

TAs  
ntu.mldsta@gmail.com

---

# Outline

- ❖ **Timeline**
- ❖ **Task Descriptions**
- ❖ **Q&A**

# Timeline

# Three Parts in HW1

- (1-1) Deep vs Shallow:
  - Simulate a function.
  - Train on actual task using shallow and deep models.
- (1-2) Optimization
  - Visualize the optimization process.
  - Observe gradient norm during training.
  - What happens when gradient is almost zero?
- (1-3) Generalization
  - Can network fit random labels?
  - Number of parameters v.s. Generalization
  - Flatness v.s. Generalization

# Schedule

- 3/9 :
  - Release HW1-1
- 3/16 :
  - Release HW1-2
- 3/23:
  - Deadline to team-up by yourselves
  - Release HW1-3
- 3/30:
  - Deadline to team-up by TAs
- 4/6:
  - All HW1 due (including HW1-1, HW1-2 and HW1-3)

# Task Descriptions

# HW1-3: Generalization

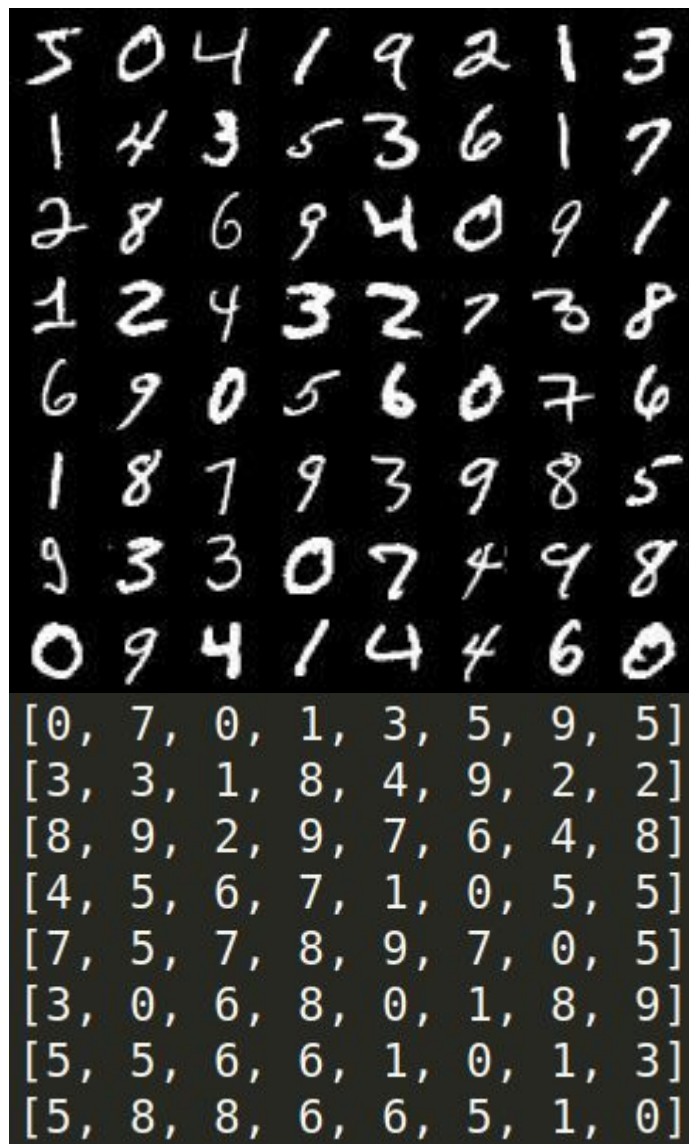
- Three subtask
  - Can network fit random labels?
  - Number of parameters v.s. Generalization
  - Flatness v.s. Generalization
- Train on MNIST or CIFAR-10...

# Can network fit random labels? <sup>1/2</sup>

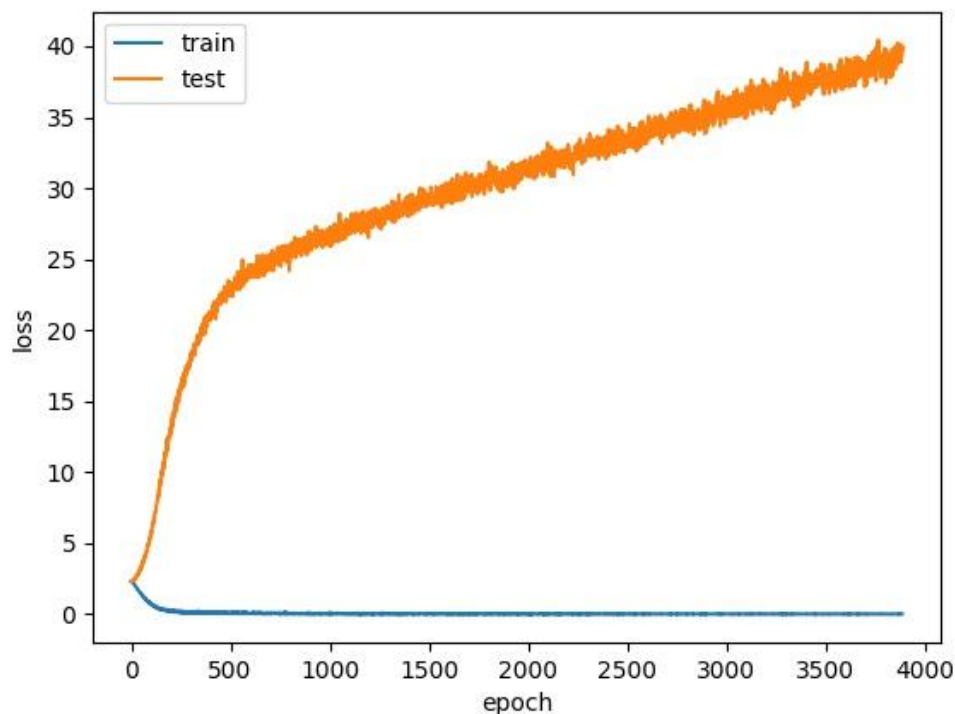
- Requirement
  - Train on MNIST or CIFAR-10
  - Randomly shuffle the label before training.
  - Try to fit the network with these random labels.



# Can network fit random labels? <sub>2/2</sub>



- MNIST
  - 3 hidden layers with 256 nodes.

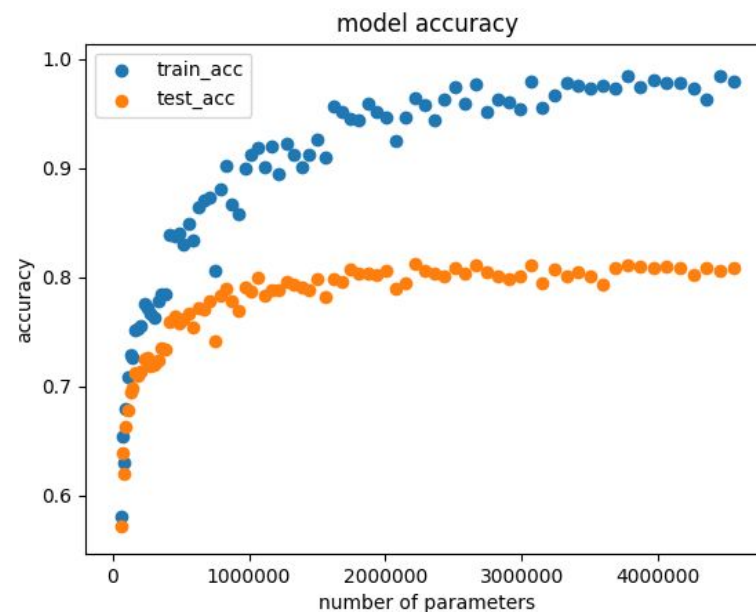
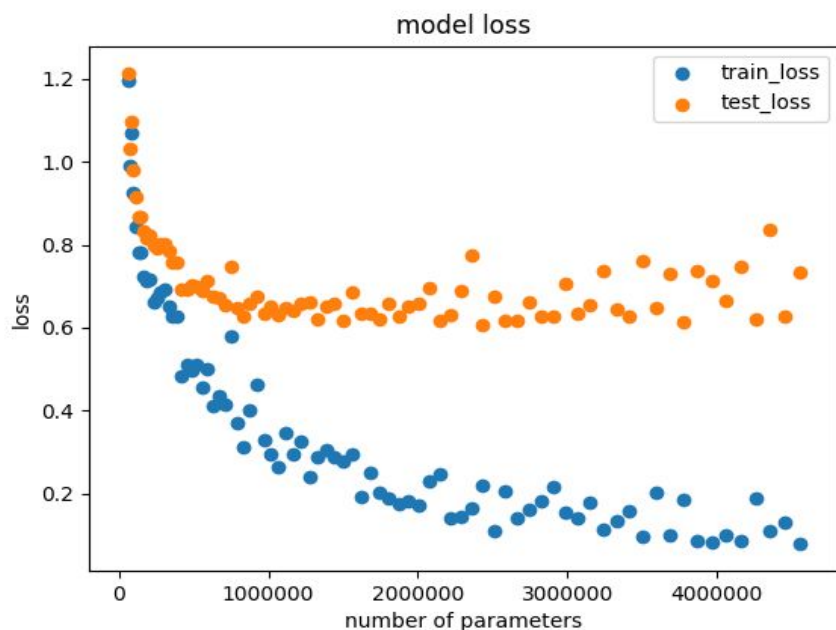


# Number of parameters v.s. Generalization <sup>1/2</sup>

- Requirement
  - Train on MNIST or CIFAR-10
  - At least 10 similar-structured models with different amount of parameters
  - Record both training and testing, loss and accuracy

# Number of parameters v.s. Generalization <sup>2/2</sup>

- CIFAR-10

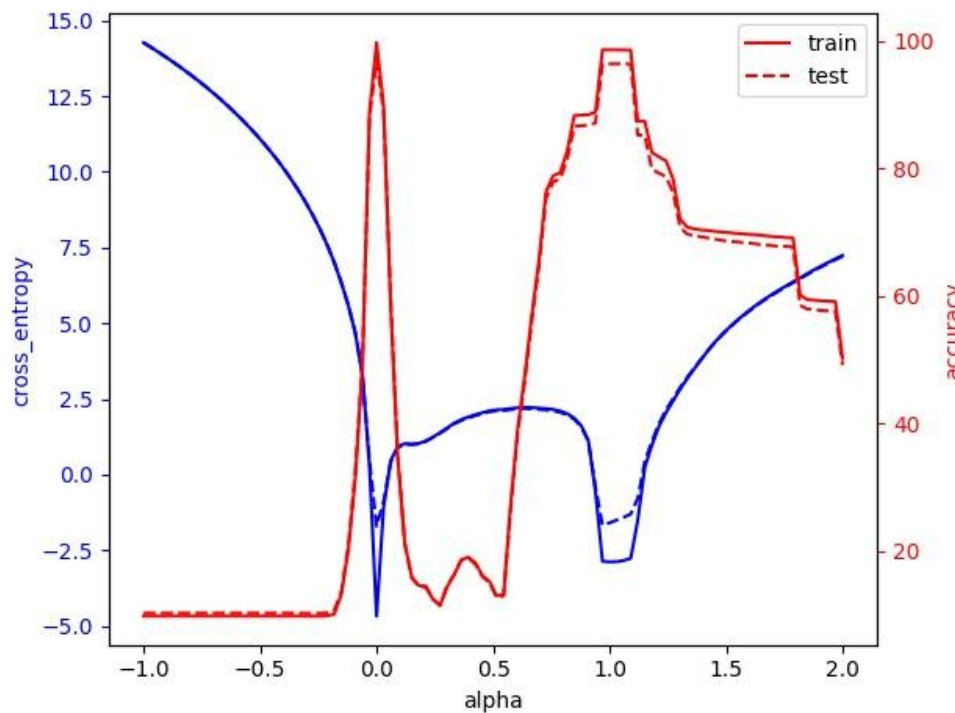
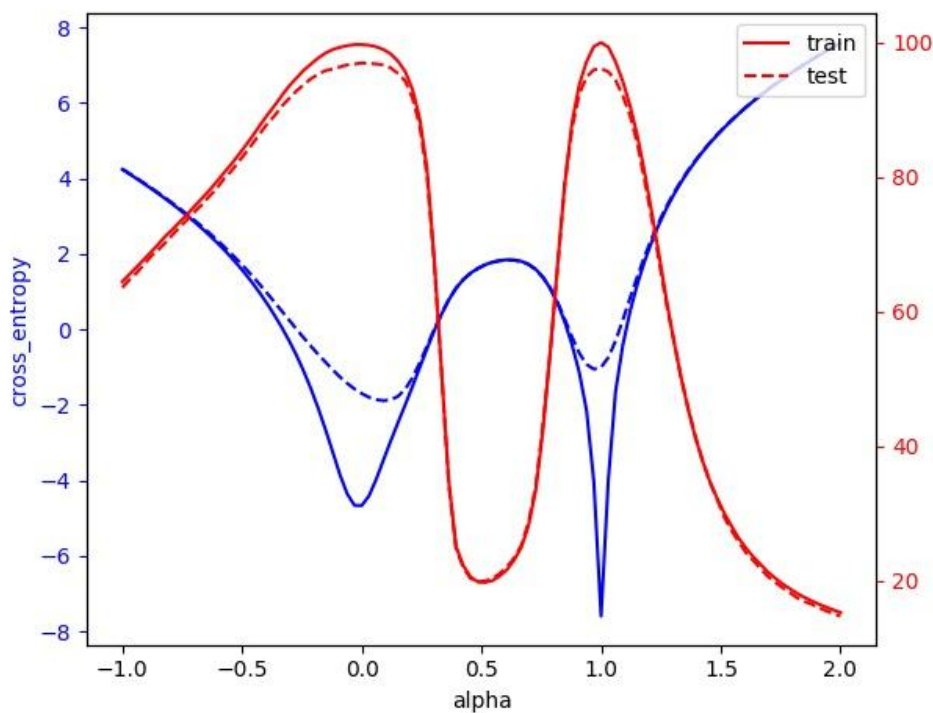


# Flatness v.s. Generalization - part1 <sub>1/2</sub>

- Visualize the line between two trained models
- Requirement:
  - Train two models (m1 and m2) with different training approach. (e.g. batch size 64 and 1024)
  - Record the loss and accuracy of the model which is linear interpolation between m1 and m2.
  - $\theta_{\alpha} = (1 - \alpha) \theta_1 + \alpha \theta_2$ , where  $\alpha$  is the interpolation ratio,  $\theta$  is the parameter of the model.

# Flatness v.s. Generalization - part1 <sub>2/2</sub>

- MNIST (The cross\_entropy is log scale)
  - batch size 64 vs. batch size 1024
  - learning rate 1e-3 vs. 1e-2



# Flatness v.s. Generalization - part2 <sup>1/</sup>

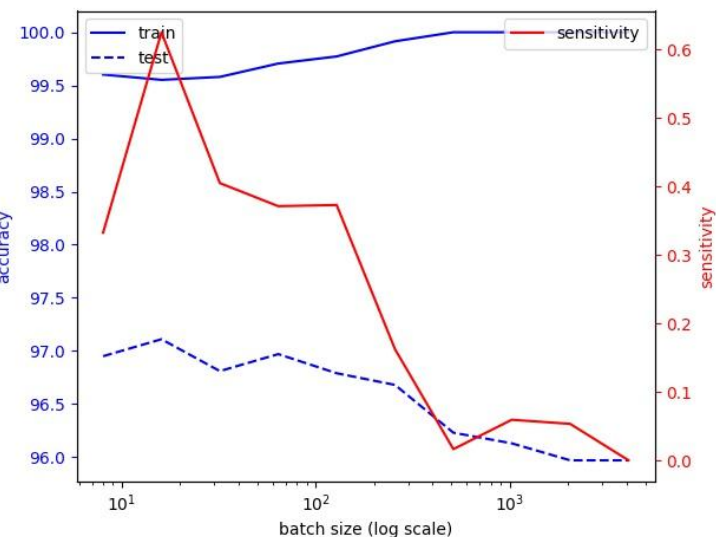
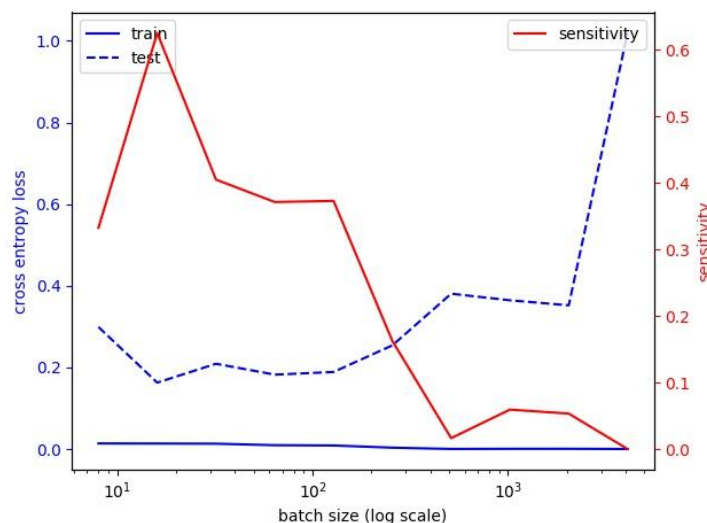
- Requirement:
  - Train at least 5 models with different training approach.
  - Record the loss and accuracy of all models.
  - Record the sensitivity of all models.

# Flatness v.s. Generalization - part2 2/

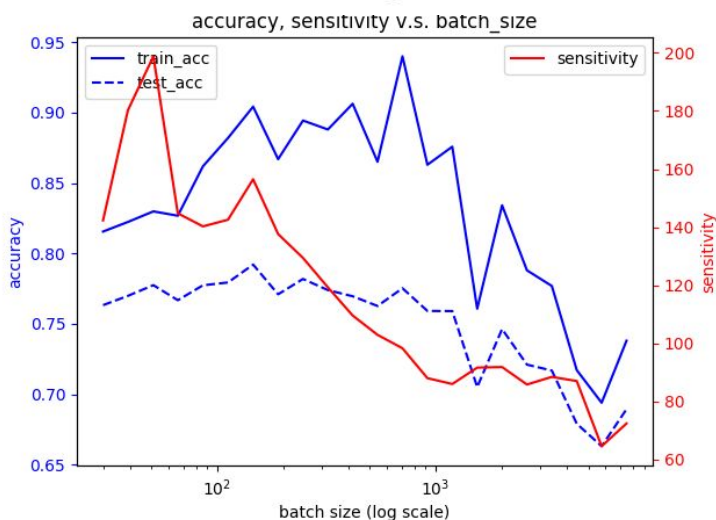
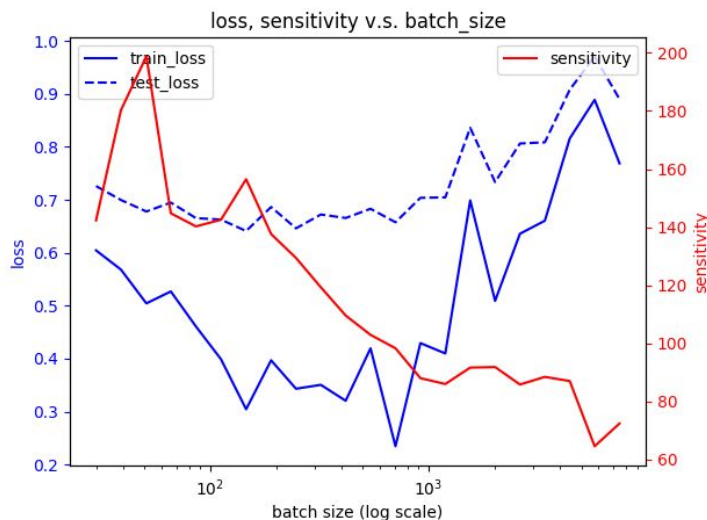
- What is sensitivity:
  - Reference: <https://arxiv.org/pdf/1802.08760.pdf>
  - Original definition:
    - Frobenius norm of Jacobian matrix of model output (class probability) to input
    - Computationally expensive for us
  - Our definition:
    - Frobenius norm of gradients of loss to input

# Flatness v.s. Generalization - part2 <sub>3/</sub>

- MNIST :



- CIFAR10 :





# HW1-3 Report Questions (10%)

- Can network fit random labels?
  - Describe your settings of the experiments. (e.g. which task, learning rate, optimizer) (1%)
  - Plot the figure of the relationship between training and testing, loss and epochs. (1%)
- Number of parameters v.s. Generalization
  - Describe your settings of the experiments. (e.g. which task, the 10 or more structures you choose) (1%)
  - Plot the figures of both training and testing, loss and accuracy to the number of parameters. (1%)
  - Comment your result. (1%)
- Flatness v.s. Generalization
  - Part 1:
    - Describe the settings of the experiments (e.g. which task, what training approaches) (0.5%)
    - Plot the figures of both training and testing, loss and accuracy to the number of interpolation ratio. (1%)
    - Comment your result. (1%)
  - Part 2 :
    - Describe the settings of the experiments (e.g. which task, what training approaches) (0.5%)
    - Plot the figures of both training and testing, loss and accuracy, sensitivity to your chosen variable. (1%)
    - Comment your result. (1%)
  - Bonus : Use other metrics or methods to evaluate a model's ability to generalize and concretely describe it and comment your results.
- Check TA's examples to know what to plot !

# Flatness v.s. Generalization - bonus example <sup>1/8</sup>

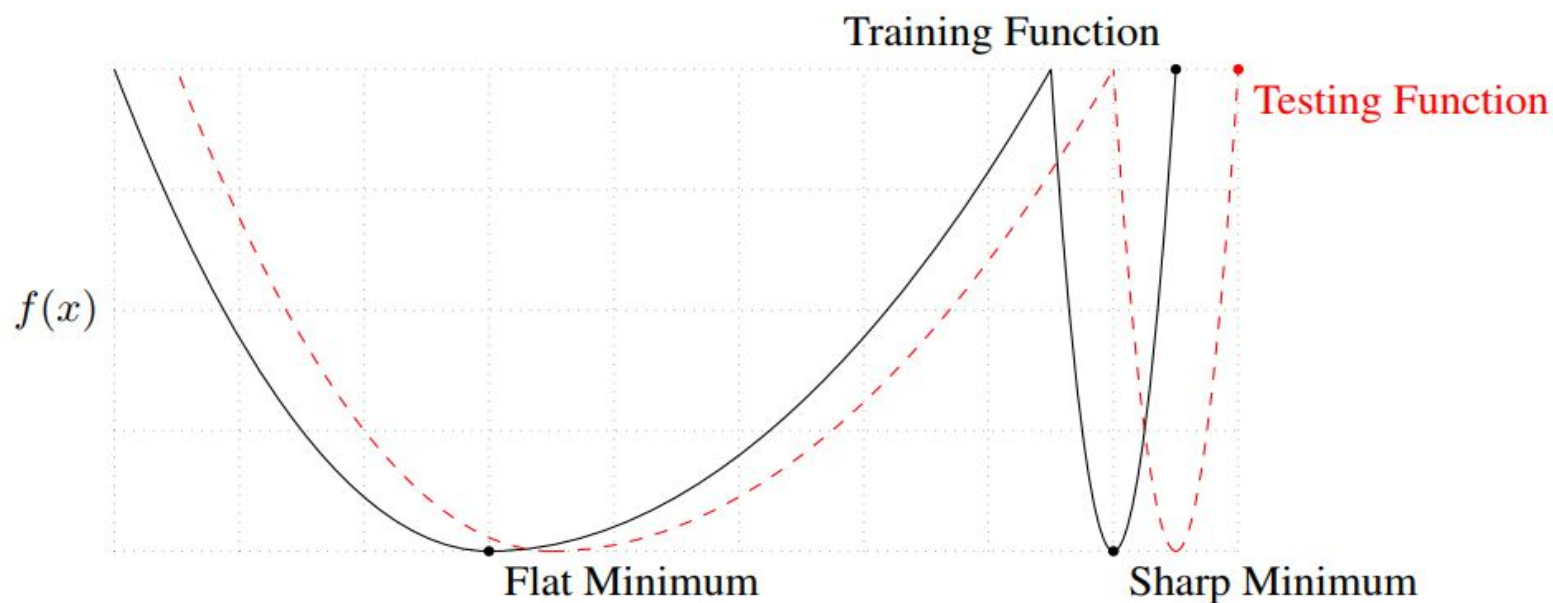
- Reference : <https://arxiv.org/pdf/1609.04836.pdf>
- Requirement
  - Train on MNIST or CIFAR-10
  - Use at least ten different approaches to train the same model
  - Calculate the sharpness of trained models
- Tips
  - Train on MNIST
  - Train with different batch size

# Flatness v.s. Generalization - bonus example <sup>2/8</sup>

- There is a generalization gap when using large-batch (LB) methods (instead of small-batch (SB) methods) for training deep learning models.
- The reasons (maybe more than these):
  - LB methods lack the explorative properties of SB methods and tend to zoom-in on the minimizer closest to the initial point.
  - SB and LB methods converge to qualitatively different minimizers with differing generalization properties (i.e. SB converges to flat minimizer, LB converges to sharp minimizer)
- We will focus on the second reason.

# Flatness v.s. Generalization - bonus example <sup>3/8</sup>

- Visually, it means that :



- How to measure sharpness (or flatness) ?

## Flatness v.s. Generalization - bonus example <sup>4/8</sup>

- Many methods can measure sharpness, but we will only utilize one in this assignment.
- Notations:
  - $\theta$  : vector of all parameters
  - $L(\theta)$  : loss of the model given the parameters
  - $B_2(\epsilon, \theta)$  : a Euclidean ball centers at  $\theta$  with radius  $\epsilon$
- Sharpness:

$$\frac{\max_{\theta' \in B_2(\epsilon, \theta)} (L(\theta') - L(\theta))}{1 + L(\theta)}$$

# Flatness v.s. Generalization - bonus example <sup>5/8</sup>

- How to calculate this :  $\frac{\max_{\theta' \in B_2(\epsilon, \theta)} (L(\theta') - L(\theta))}{1 + L(\theta)}$
- Original paper : Use L-BFGS-B to maximize  $L(\theta')$
- Around a critical point :  
$$L(\theta') = L(\theta) + \frac{1}{2}(\theta' - \theta)^T (\nabla^2 L)(\theta)(\theta' - \theta) + o(\|\theta' - \theta\|_2^2)$$
- Then :

$$\frac{\max_{\theta' \in B_2(\epsilon, \theta)} (L(\theta') - L(\theta))}{1 + L(\theta)} \longrightarrow \frac{\|(\nabla^2 L)(\theta)\|_2 \epsilon^2}{2(1 + L(\theta))}$$

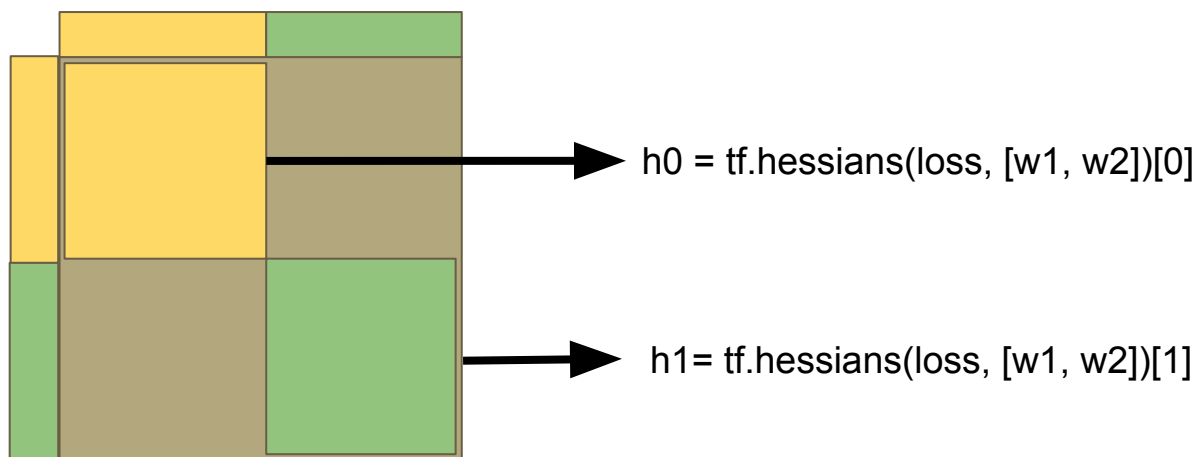
- Since 2-norm of a matrix is defined as :

$$\|A\|_2 = \max\{\|Ax\|_2 : x \in R^n \text{ with } \|x\|_2 = 1\} = \sqrt{\lambda_{\max}(A^T A)}$$

# Flatness v.s. Generalization - bonus example <sup>6/8</sup>

- How to calculate Hessian matrices efficiently:
  - Use GPU : `tf.hessians`
  - Calculate only 500 out of 60000 examples in MNIST
- But `tf.hessians` only return block-diagonal part:
  - vector of all paramters : 

W1	W2
----	----
  - Hessian matrix :



## Flatness v.s. Generalization - bonus example <sup>7/8</sup>

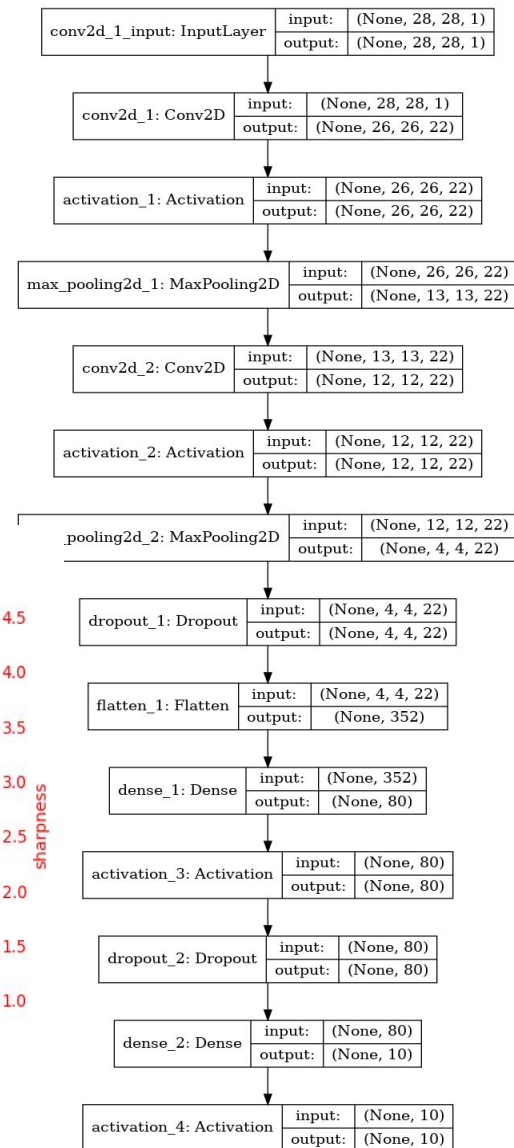
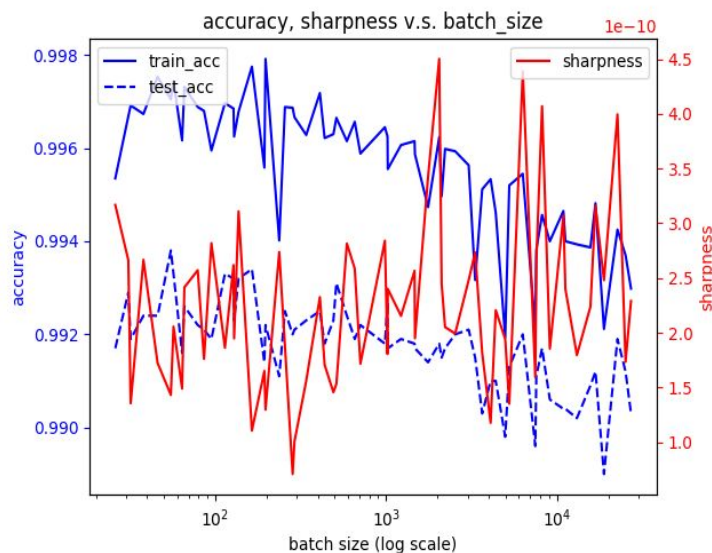
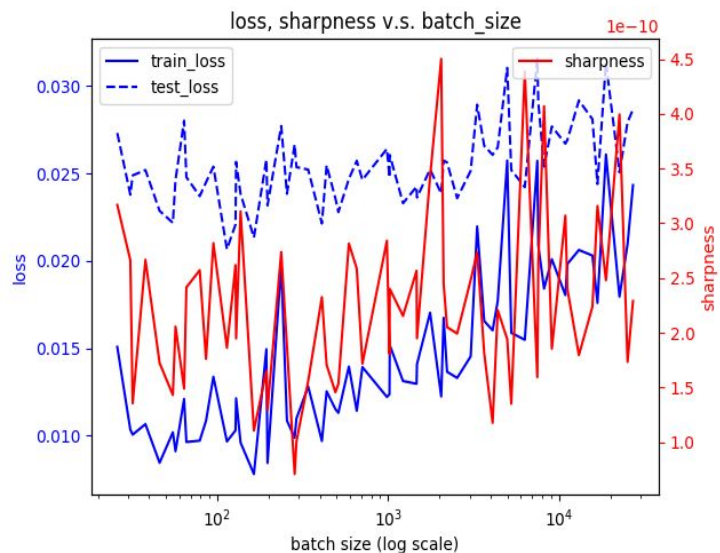
- If we assume off-block-diagonal elements is negligible:
  - Square of block-diagonal matrix is also block-diagonal.
  - Eigenvalues of a block-diagonal matrix is the list of all eigenvalues of each block submatrix.
  - Since we only want the largest eigenvalue, we can conclude that the 2-norm of a block-diagonal matrix is the 2-norm of block submatrix that contains the largest eigenvalue itself.
  - 2-norm of matrix A in tensorflow : `tf.norm(A, 2)`
  - 2-norm of matrix A in numpy : `np.linalg.norm(A, 2)`



# Flatness v.s. Generalization - bonus example 8/8

- MNIST :

- 20000~30000 parameters (in order to calculate hessian matrices while maintaining enough model capacity)
- Calculate hessian matrices as mentioned in previous slide
- epsilon :  $1e-4$



# Flatness v.s. Generalization - more possible bonus

1/2

- Reference: <https://arxiv.org/pdf/1703.04933.pdf>
- This paper shows that several metrics (including sharpness) do not indicate ability of generalization for any RELU-based deep models.
- Reparametrize:
  - $\text{relu}(x \cdot (\alpha\theta_1)) \cdot \theta_2 = \text{relu}(x \cdot \theta_1) \cdot (\alpha\theta_2)$   
if  $\alpha > 0$

# Allow Packages

- python 3.6
- TensorFlow r1.6
- PyTorch 0.3 / torchvision
- Keras 2.0.7 (TensorFlow backend only)
- MXNet 1.1.0
- CNTK 2.4
- matplotlib
- scikit-learn 0.19.1
- Python Standard Library
- Pandas
- If you want to use other packages, please ask TAs for permission first!

# Submission

- Deadline: **2018/4/6 23:59 (GMT+8)**
- Write the questions of HW1-1, HW1-2 and HW1-3 in **one** report.
- Chinese unless you are not familiar with Chinese
- At most 10 pages for HW1-1, HW1-2 and HW1-3
- Your github must have several files under directory hw1/
  - Readme.\*
  - Report.pdf
  - other code
- In your Readme, state clearly how to run your program to generate the results in your report.
- Files for training is required.
- **Check github collaborator. If you still not receive the confirmation, please invite TA's account (mldsta) again.**

# Q&A

[ntu.mldsta@gmail.com](mailto:ntu.mldsta@gmail.com)

# 組隊

## 1.組隊結果公布在

<https://docs.google.com/spreadsheets/d/1zcbUvkgjbVV4ghG0EbHfEZ2uvxWdcDmN2ddvsaAkk7o/edit?usp=sharing>

請大家確認, 之後助教幫忙組隊的名單也會公佈在這裡。

2.自行組隊將於3/23 23:59 UTC +8 (今天)截止, 之後助教將會以盡量每組湊到三個人為原則帮大家組隊

3.如果你們有兩個人已經一隊, 希望找到第三個人一組, 但是找不到, **請不要填表單並於3/23 23:59 UTC +8(今天)前寄信到助教信箱**。助教會盡量從尚未組隊的名單中找人跟你們一隊(不保證找得到, 且不保證找到的是強者), 不會把你們拆散。