Machine Learning and Having It Deep and Structured

HOMEWORK #1

Training Deep Neural Network

呂相弘　吳彥謎
r03942039@ntu.edu.tw

Outline

- DNN in Speech Recognition
- DNN
- Kaggle
- Dataset and Format
- Submission Requirements
- Grading
- GPU Acceleration
- Tips
- Baselines
- 語音導覽

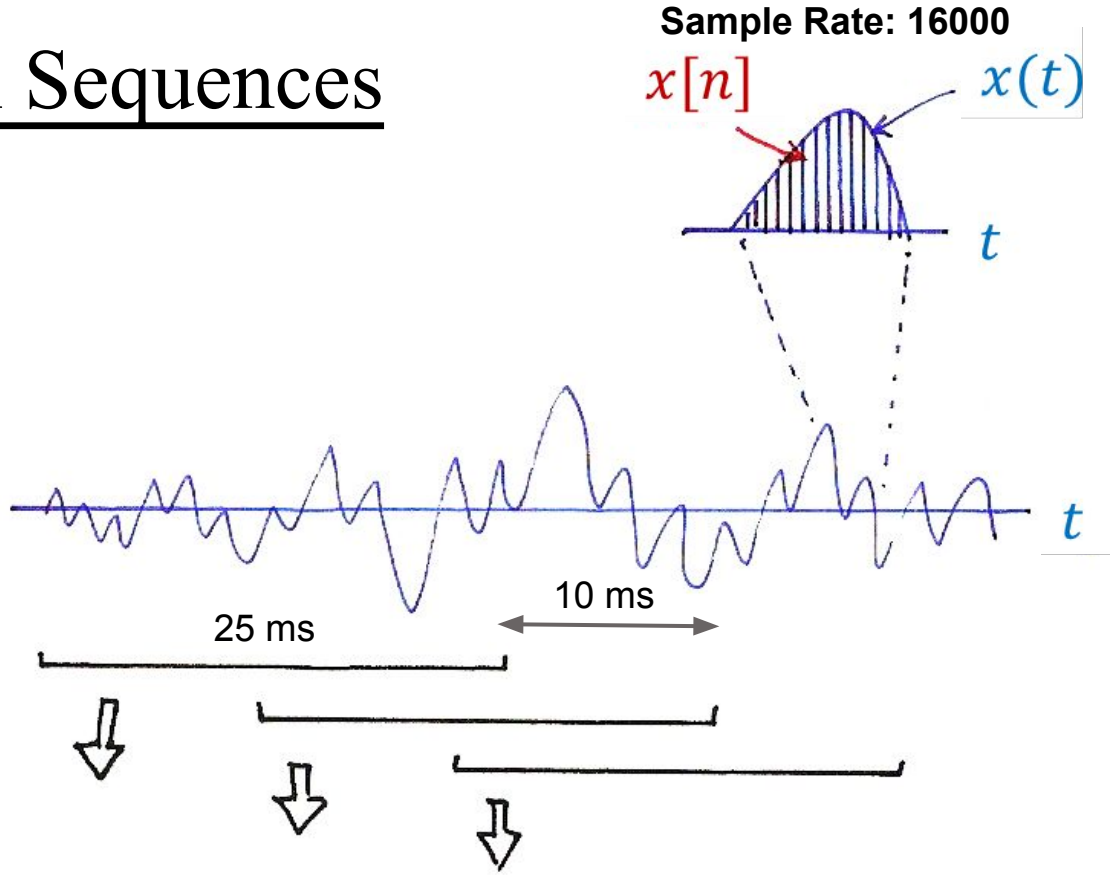
DNN IN SPEECH RECOGNITION

Speech Recognition

- In speech processing...
  - each word consists of syllables
  - each syllable consists of phonemes

    “青色” → “青(くㄧㄥ)色(ムさ、)” → ”く” (syllables)
    青:TSI --I –N (phonemes)
    色:S--@ (phonemes)
- Each time frame, with an observance (vector) mapped to a phoneme.

Observation Sequences



sliding window

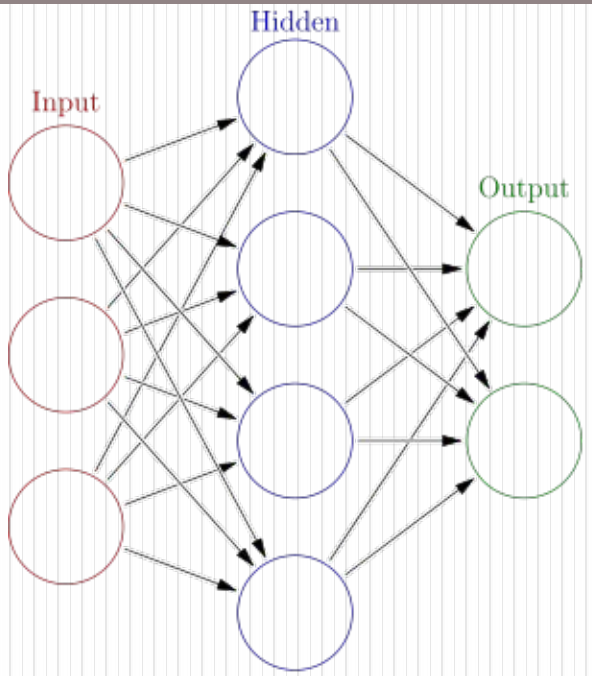
frames of features

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_D \end{bmatrix} = o_1 \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_D \end{bmatrix} = o_2 \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_D \end{bmatrix} = o_3$$

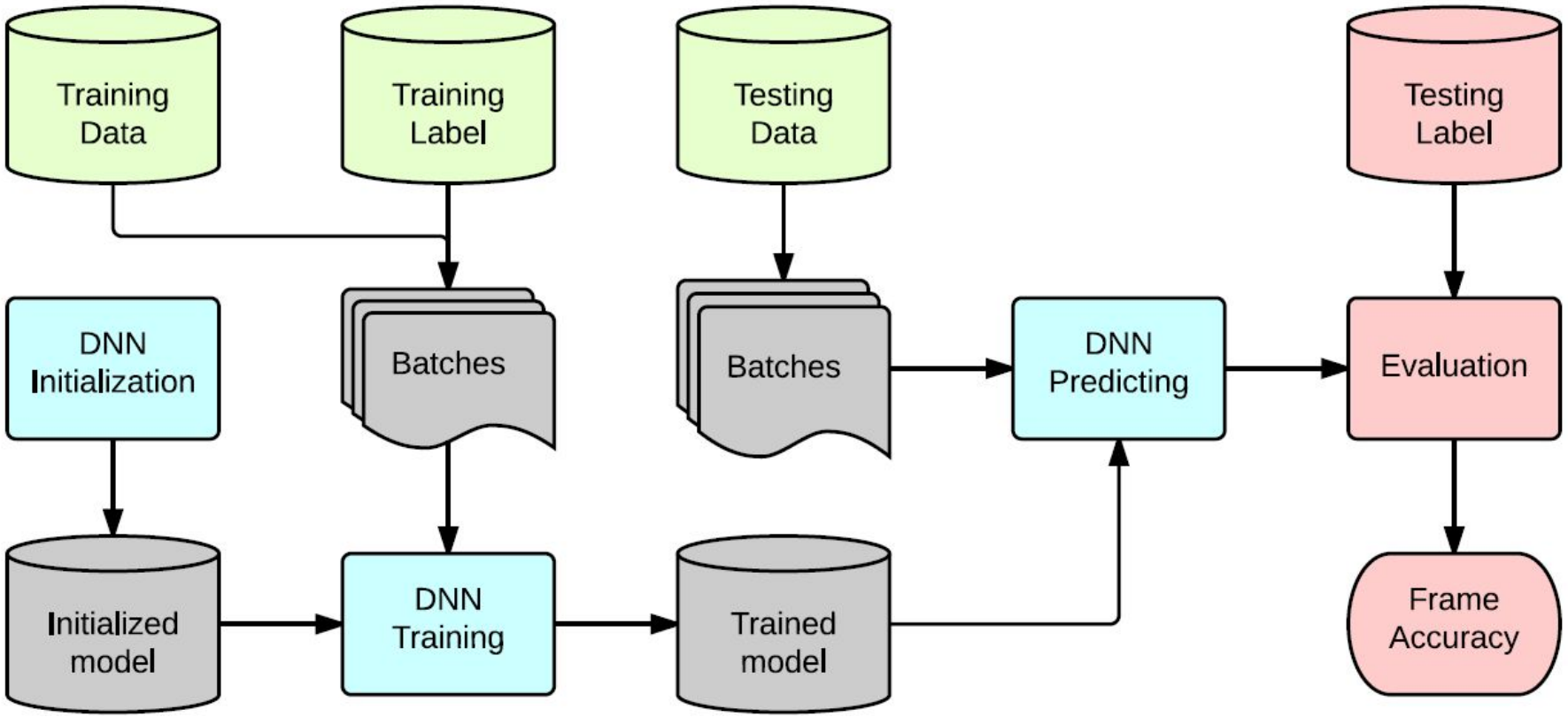
Frame 1 **Frame 2** **Frame 3**

DNN in Speech Recognition

- **Goal:** predict **phoneme** given feature in each time frame.
  - Frame-wise prediction
- **Input**: acoustic features
  - Raw waveform, MFCC, FBANK or...
- **Output**: pronunciation units
  - Phonemes or...
- To know more about Automatic Speech Recognition(ASR), please refer to http://speech.ee.ntu.edu.tw/DSP2015Spring/



Homework 1: Training Deep Neural Network

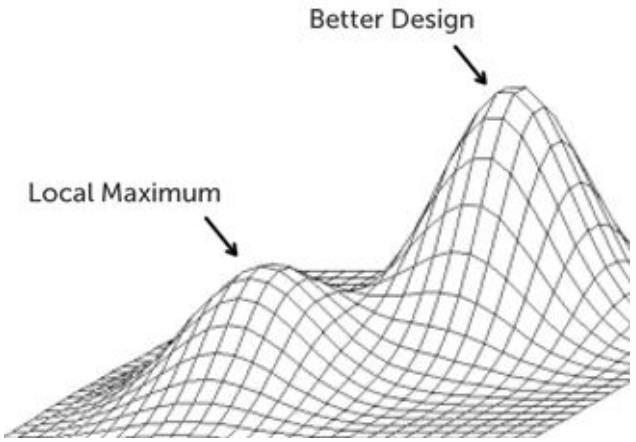

Training Data
Training Label
Testing Data
Testing Label
DNN Initialization
Batches
Batches
DNN Predicting
Evaluation
Initialized model
DNN Training
Trained model
Frame Accuracy

Main Problems in HW1

- Model initialize
- Feedforward
- Backpropagate
- Update
- Predict

Model Initialize

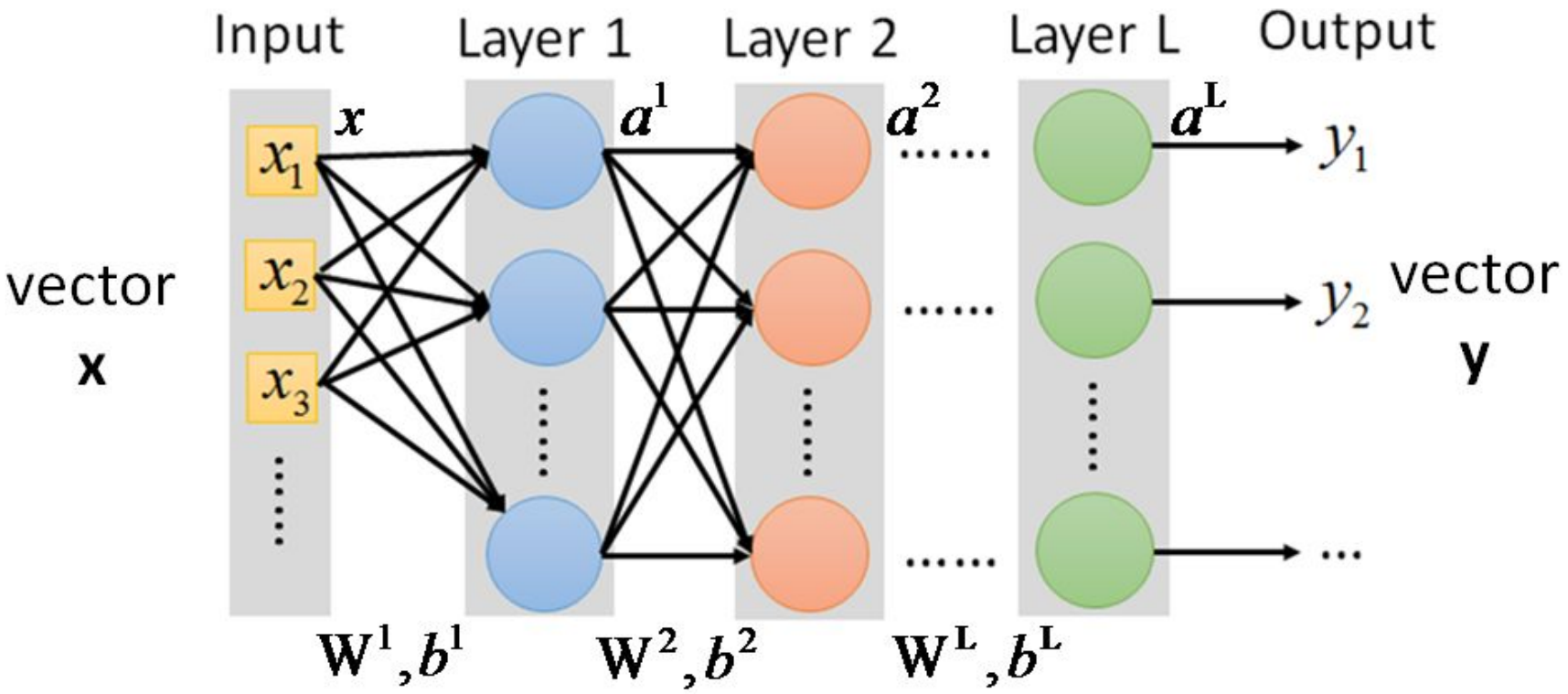
- DNN sometimes fails at local optimum problem, so **initialization matters**.
- Practically, there exists unsupervised pre-training technique on initialization.
- However, in this homework, we recommend you **initialize them randomly for the simplicity and efficiency.**



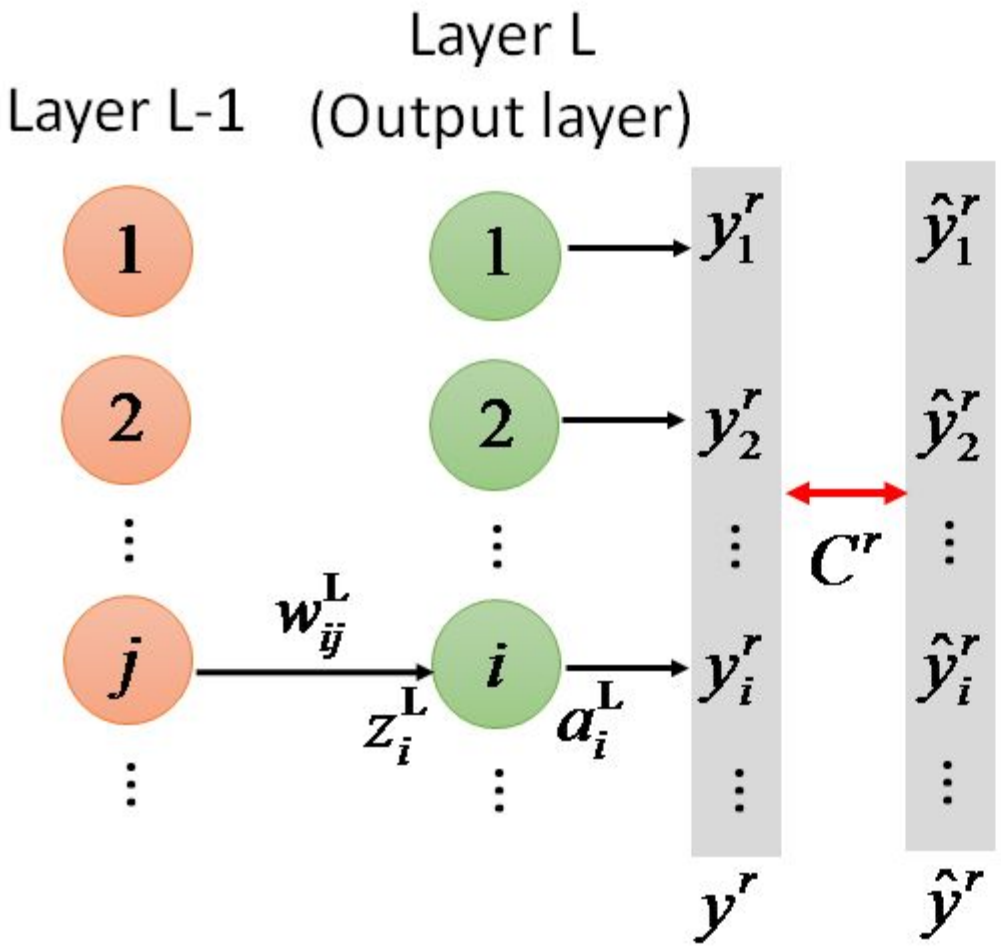
Feedforward

$$y = f(x) = \sigma\left(W^L \ldots \sigma\left(W^2 \sigma\left(W^1 x + b^1\right) + b^1\right) \ldots + b^L\right)$$

$$\sigma\left(W^L a^{L-1} + b^L\right) = a^L = y$$



Backpropagate



$$\frac{\partial C^r}{\partial w_{ij}^L} = \frac{\partial C^r}{\partial y_i^r}\frac{\partial y_i^r}{\partial z_i^L}\frac{\partial z_i^L}{\partial w_{ij}^L}$$

$$= a_j^{L-1}\sigma'\left(z_i^L\right)\left(y_i^r - \hat{y}_i^r\right)$$

Example: $\left(x^r, \hat{y}^r\right)$

Update

$$\textbf{Compute } \nabla C(\theta^0)$$

$$\theta^1 = \theta^0 - \eta \nabla C(\theta^0)$$

$$\textbf{Compute } \nabla C(\theta^1)$$

$$\theta^2 = \theta^1 - \eta \nabla C(\theta^1)$$

$$\textbf{Compute } \nabla C(\theta^2)$$

Starting Parameters

$$\theta^0 \rightarrow \theta^1 \rightarrow \theta^2 \rightarrow \cdots\cdots$$

$$C(\theta^0) > C(\theta^1) > C(\theta^2) > \cdots\cdots$$

In practice, η can simply set to be constant

η is called "***learning rate***"

η should be small enough, but should not be too small.

Predict

- DNN classifier usually output the **1-best label** as prediction.
- Practically, the DNN may preserve certain number of good candidates for further processing due to the **structure of speech.**
  - That is, the best in one frame would be unacceptable when considering the context frames.
- The sequence "fixing" procedure is actually your HW2, comparable among the world as announced.
- **In HW1, you should only predict the framewise 1-best output.**

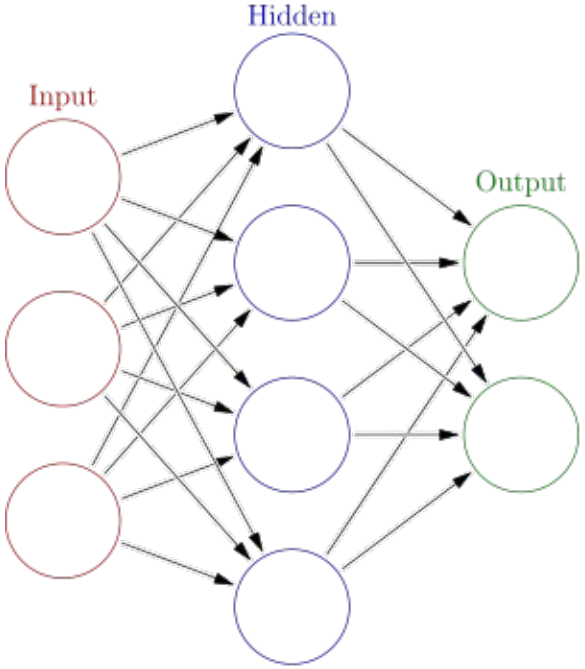
More About Input...

- We provide **3 types** of feature for the DNN input w.r.t each frame.
  - Raw Waveform(16k sampling rate)
  - MFCC(dimension = 39)
  - FBANK(dimension = 69)
- MFCC and FBANK are 2 types of feature extracted from the wave. However, **the details lie beyond the scope of this course.**
- There does exist debates over which feature is better. So the choice is yours.

More About Input…

- You could generate your own features based on the RAW/MFCC/FBANK.
- However, be sure your task can be evaluated in a frame-wise way.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_D \end{bmatrix} = o_1 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_D \end{bmatrix} = o_2$$



More About Output…

- We provide **3 types** of training target for the DNN output w.r.t each frame.
  - states(index from 0 to 1942)
  - 48 - phonemes
  - 39 - phonemes
- The difference also lies beyond the scope of this course. You could simply view them as more delicate pronunciation units at different level.

More About Output…

- Actually, the framewise labels are usually absent. There are only human annotation w.r.t each sentence.
- Practically, we forcibly align the labels to each frame by generative models other than DNN.
- The alignment results are then prepared for DNN as training targets.
- **In this homework, you do not need to bother. The alignments are transformed into framewise labels for you.**

Evaluation

- **Framewise** phoneme prediction
- Frame Accuracy

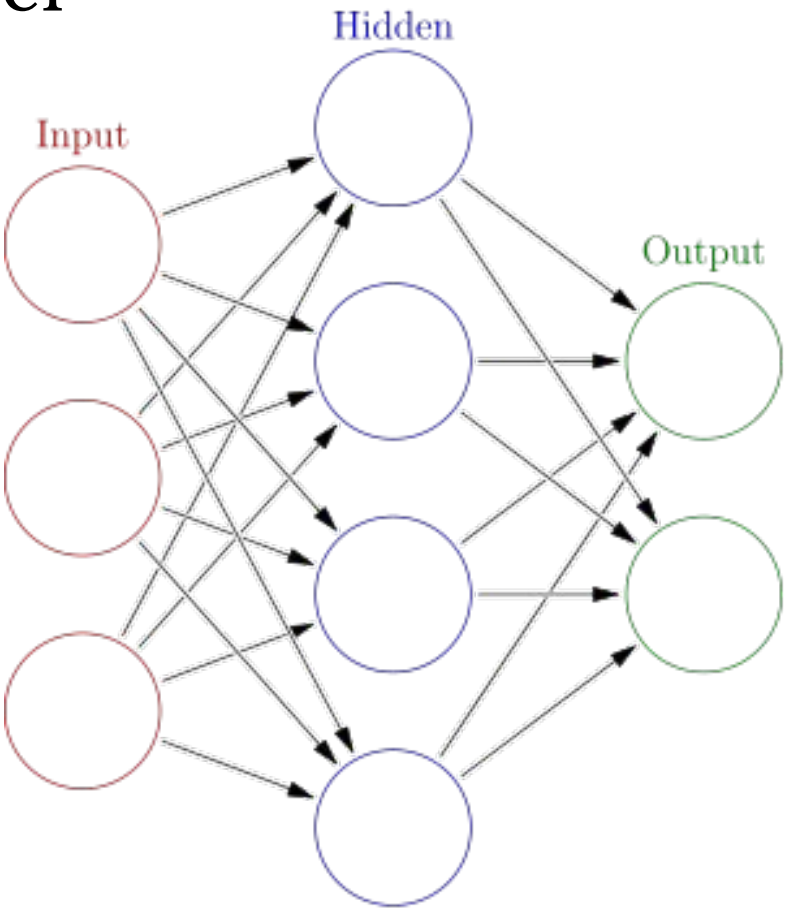
$$Frame\ Accuracy = \frac{\#\ of\ correct\ frames}{\#\ of\ total\ frames}$$

Evaluation

- Despite the type of training target you used, you have to map all of them to 39 phoneme set for evaluation.
  - Some training targets would be merged into the same phoneme.
- The frame accuracy is calculated according to the merged prediction.

Baseline

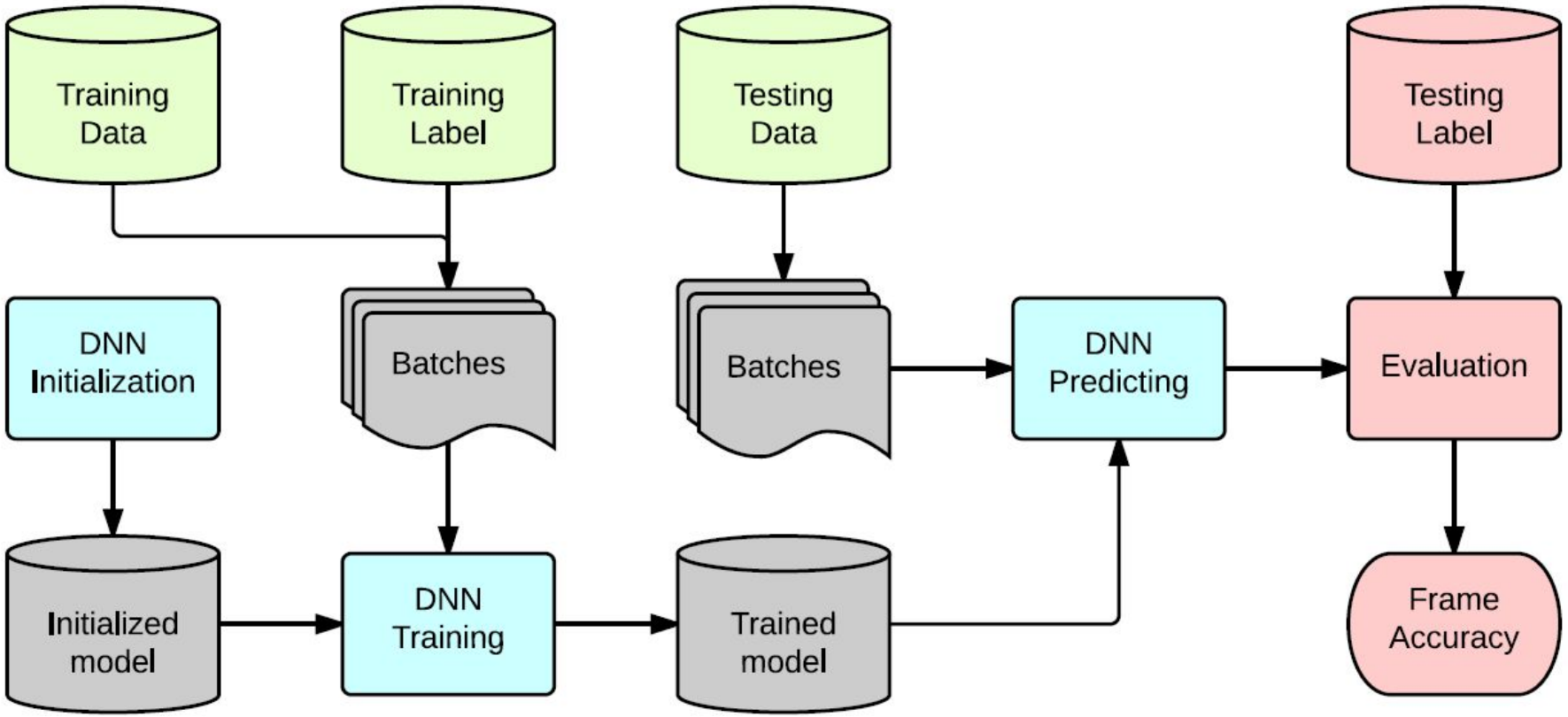
- Only one hidden layer
- 69x128x48
- 69: fbank features
- 128: 1 hidden layer
- 48: 48 phoneme set



Pseudo Code

```
dnn = DNN( STRUCTURE , LEARNING_PARAMETERS );
train = Dataset(  TRAIN_FILE , LEARNING_PARAMETERS  );
valid = Dataset(    DEV_FILE , LERANING_PARAMETERS  );
test  = Dataset(   TEST_FILE , LEARNING_PARAMETERS  );
epoch = 0;
while epoch < MAX_EPOCH
    while ( batch = train.load_batch() )
        dnn.forward(batch);
        dnn.calculate_error( ERROR_FUNCTION );
        dnn.backpropagate( );
        dnn.update();
    end
        Ein  = dnn.report_error_rate( train );
        Eval = dnn.report_error_rate( valid );
    epoch = epoch + 1;
end

while( batch = test.load_batch() )
    dnn.forward( batch )
    dnn.output_predict()
end
```


Training Data
Training Label
Testing Data
Testing Label
DNN Initialization
Batches
Batches
DNN Predicting
Evaluation
Initialized model
DNN Training
Trained model
Frame Accuracy

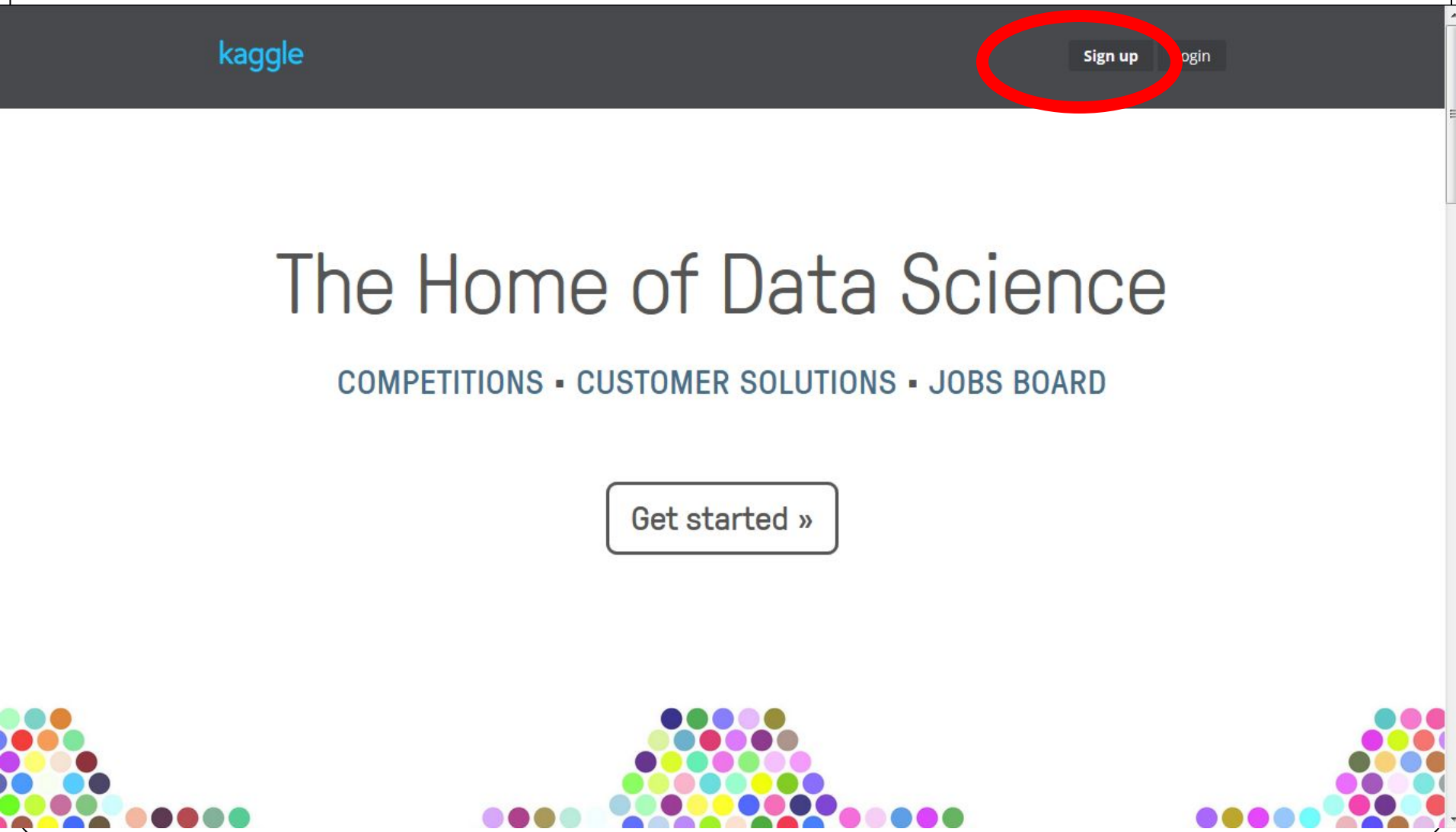
KAGGLE

Kaggle

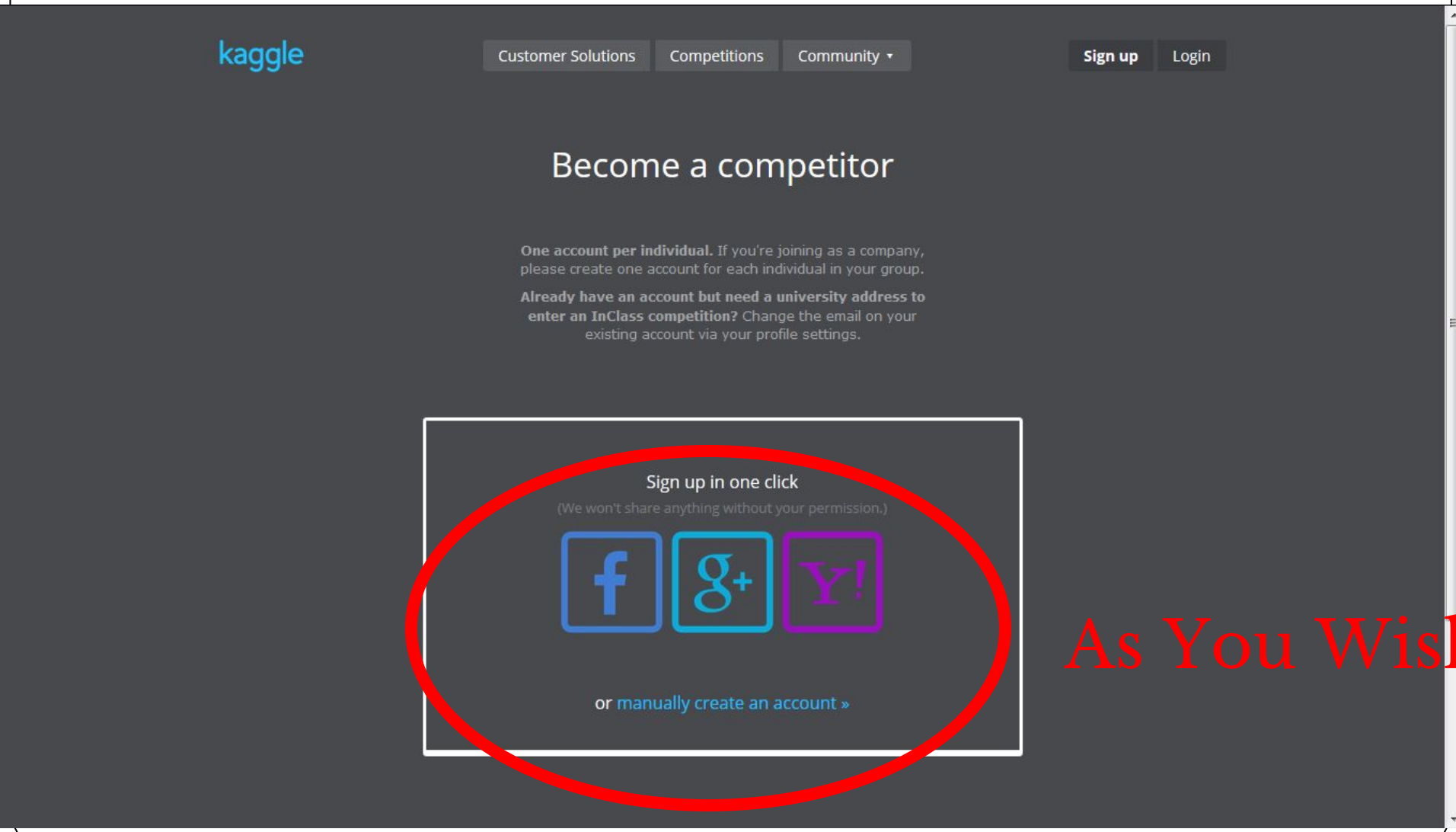
- http://www.kaggle.com/

- A competition platform used by many academic institutes.
- Evaluate your result on this website.
  - With Scoreboard that can show off your improvement.
- You could upload your predictions **only twice a day**.
  - Be careful!

Registration



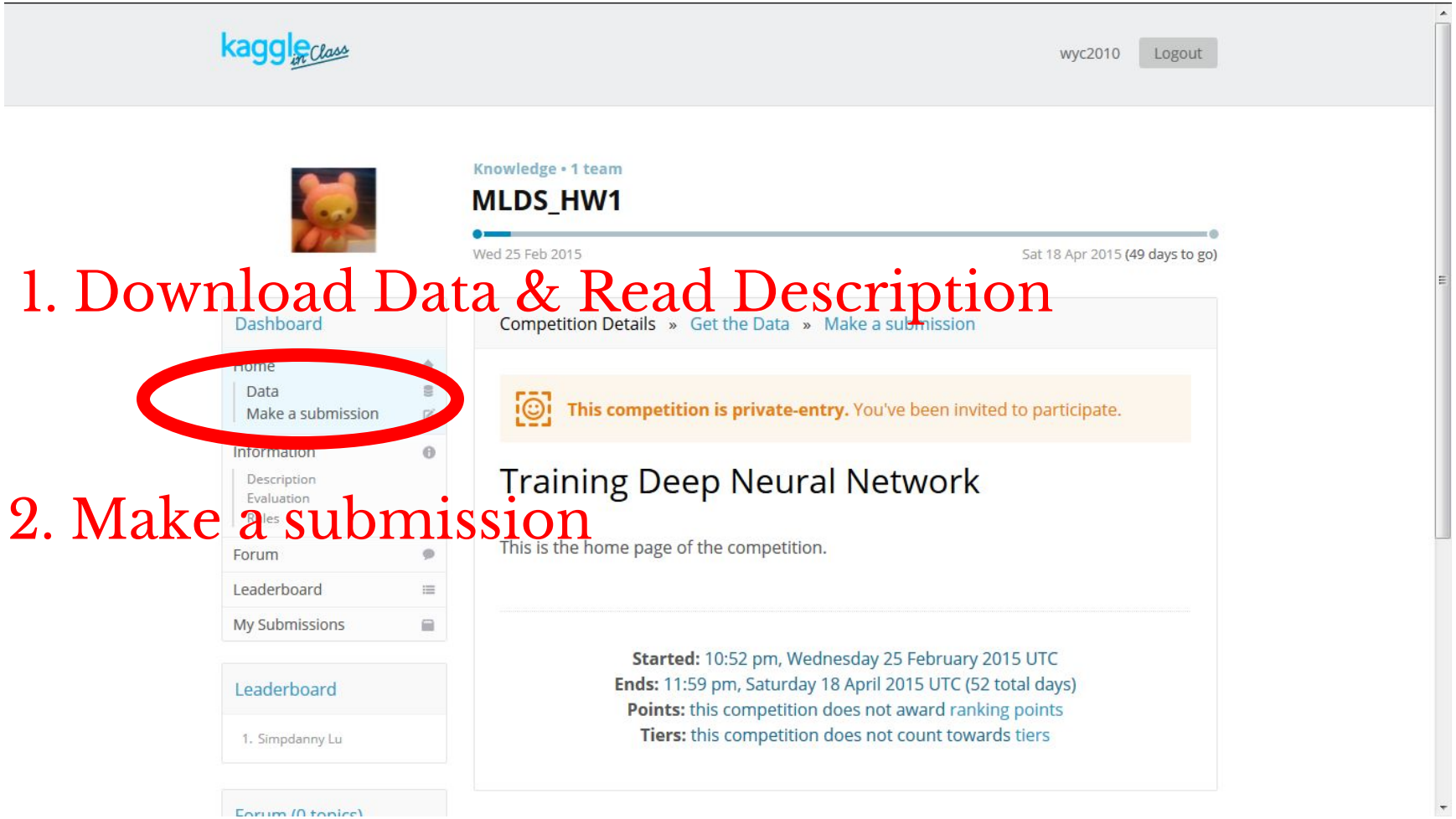
Registration



Registration

Join the Competition

- https://kaggle.com/join/mlds_hw1_kaggle



1. Download Data & Read Description

2. Make a submission

Join the Competition

1 team
1 player
1 entry

Team Mergers

Team mergers are not allowed in this competition.

Team Limits

There is no maximum team size.

Submission Limits

You may submit a maximum of 2 entries per day.
You may select up to 2 final submissions for judging.

Competition Timeline

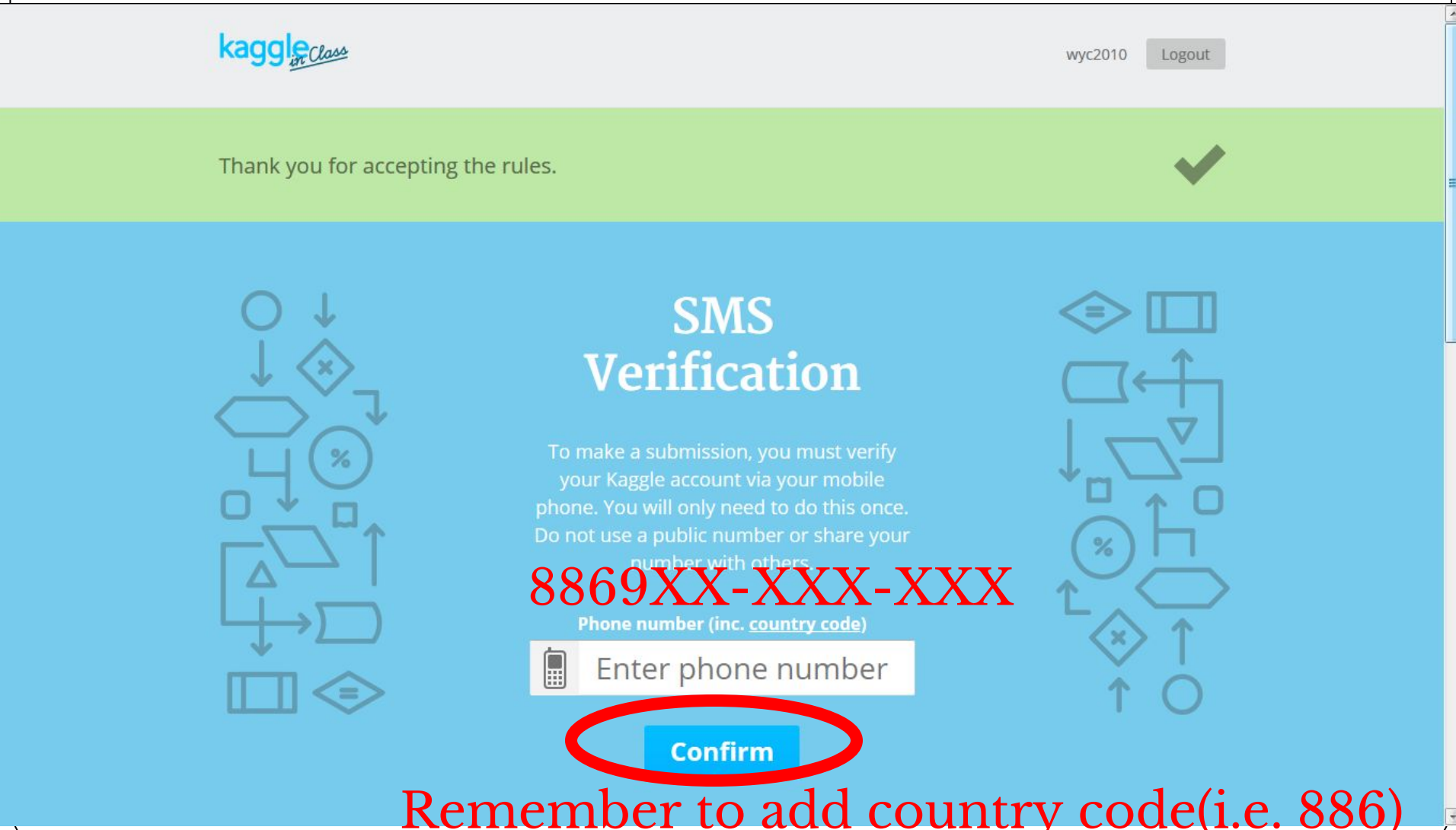
Start Date: **2/25/2015 10:52:45 PM UTC**
Merger Deadline: **None**
First Submission Deadline: **None**
End Date: **4/18/2015 11:59:00 PM UTC**

Rules Acceptance

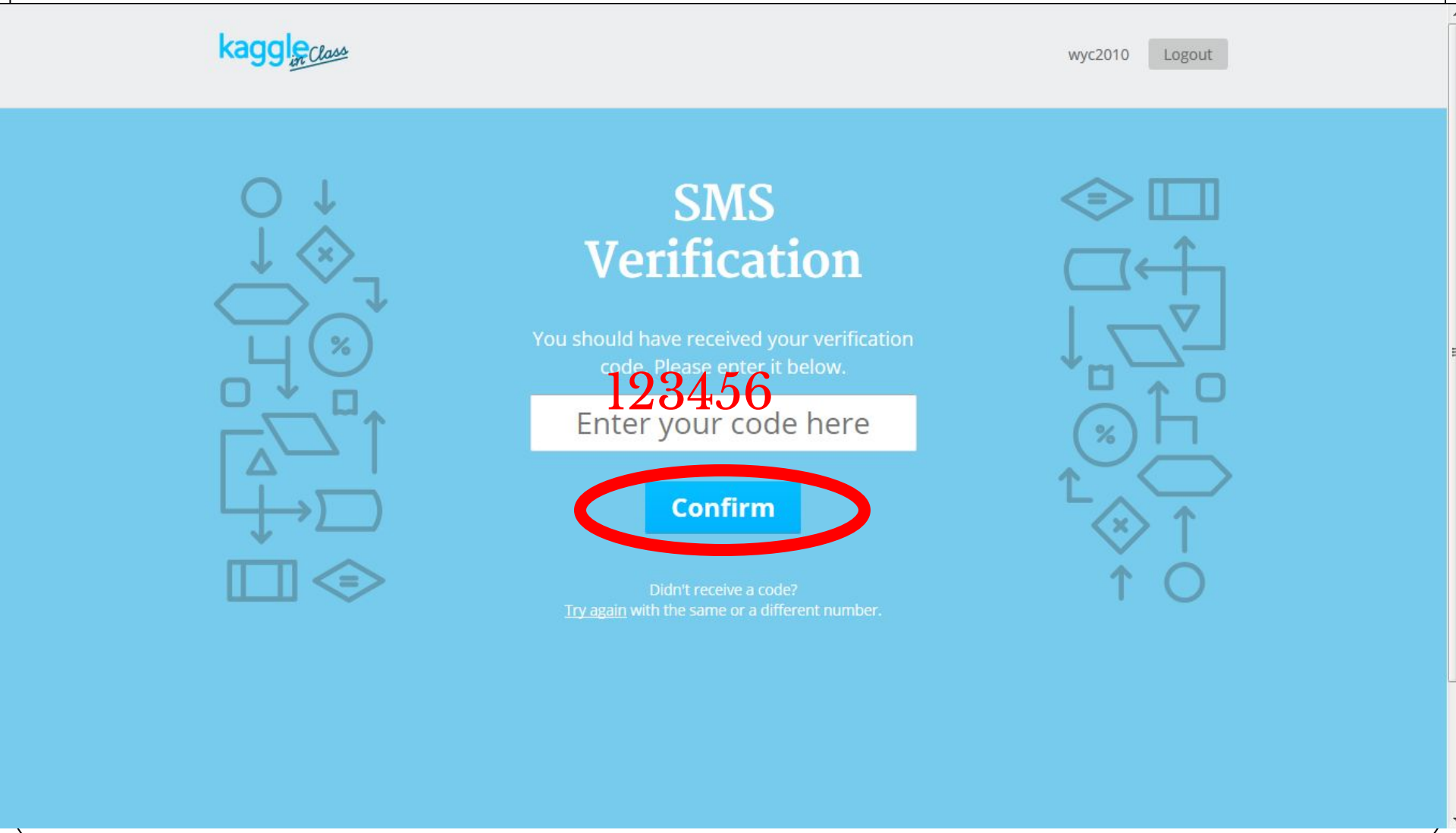
I understand and accept **I do not accept**

By clicking on the "I understand and accept" button below, you are indicating that you agree to be bound to the above rules.

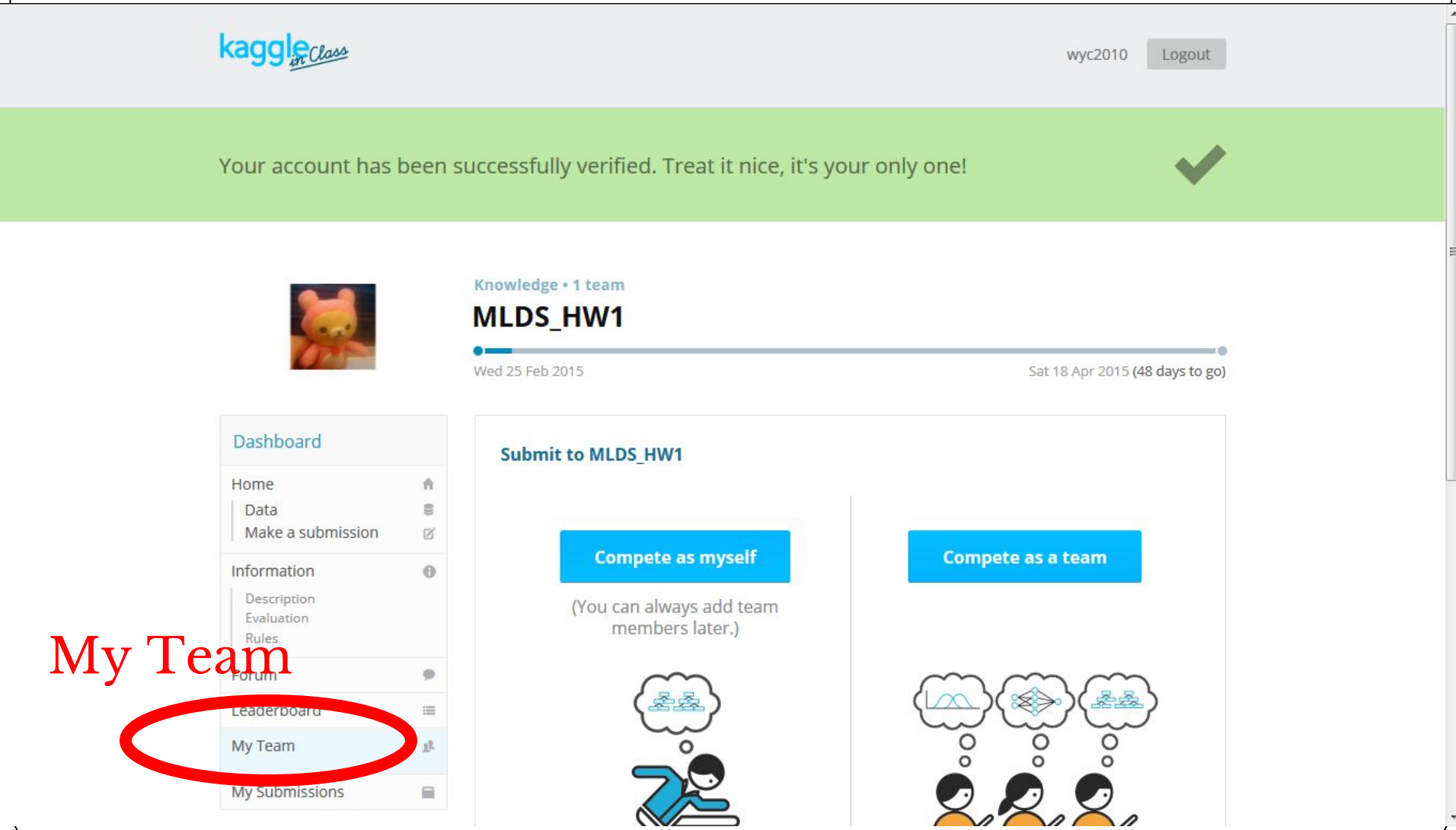
Join the Competition



Join the Competition



Join the Competition



My Team

Manage Your Team

Knowledge • 1 team

MLDS_HW1

Wed 25 Feb 2015 — Sat 18 Apr 2015 (48 days to go)

Dashboard

- Home
  - Data
  - Make a submission
- Information
  - Description
  - Evaluation
  - Rules
- Forum
- Leaderboard
- My Team
- My Submissions

Leaderboard

1. Simpdanny Lu

Forum (0 topics)

Manage Your Team

3. **Make a Submission** on behalf of your team.

1. Team Nam 飛語傳音

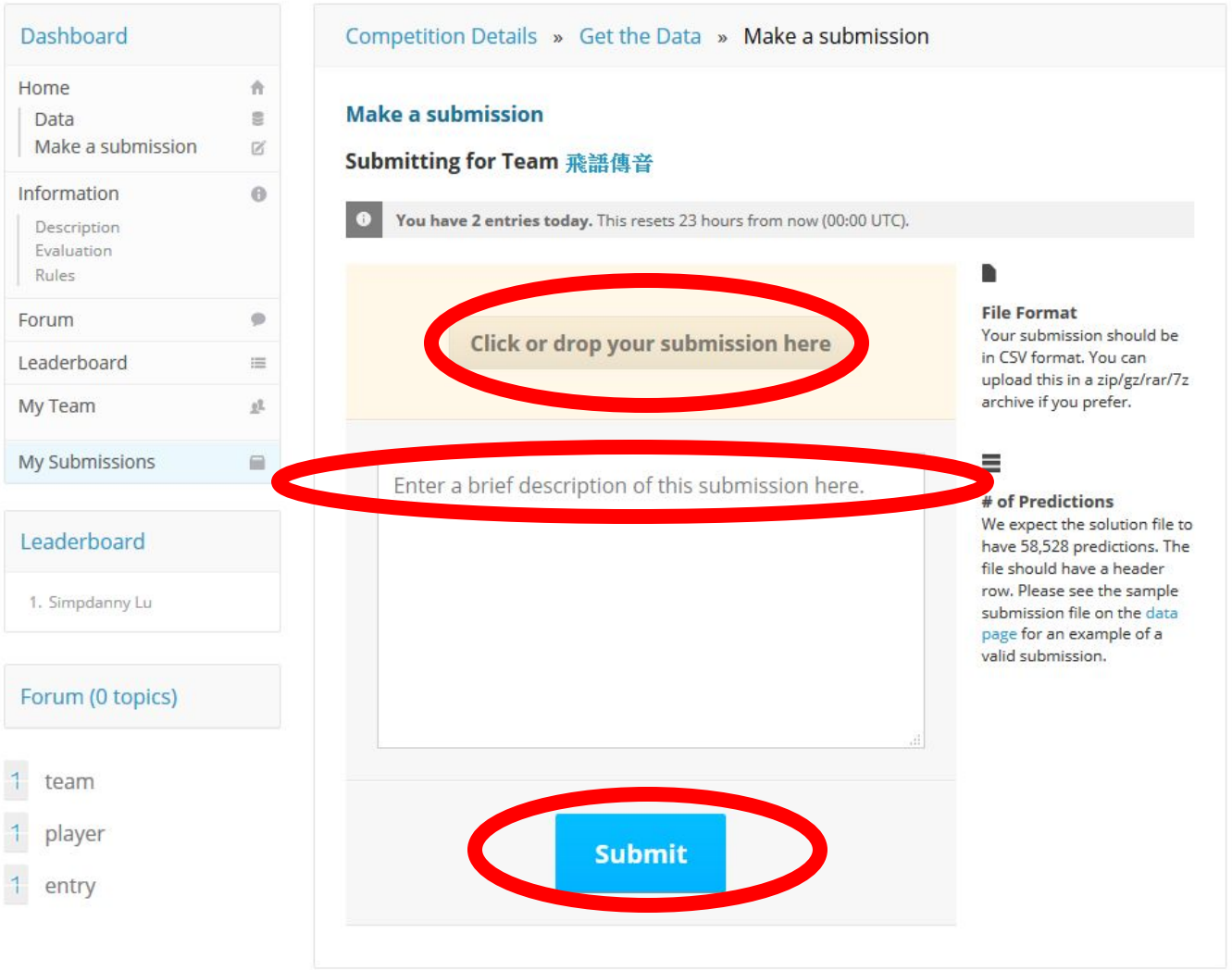
Team Members

wyc2010
Team Leader

2. Invite someone to join your team

Email Address **Send**

Submitting for Team



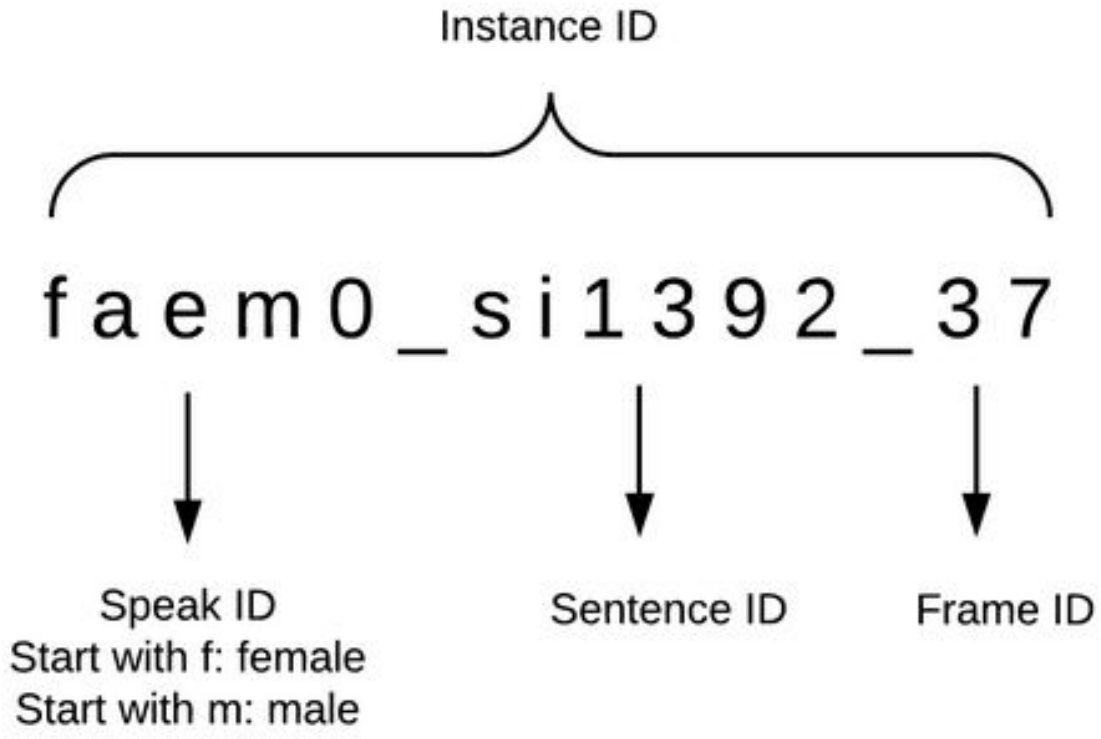
Dataset and Format

Dataset

- **TIMIT**(**T**exas **I**nstrument and **M**assachusetts **I**nstitute of **T**echnology)
- Well-transcribed speech of American English speakers of different sexes and dialects.
- Designed for the development and evaluation of ASR systems.

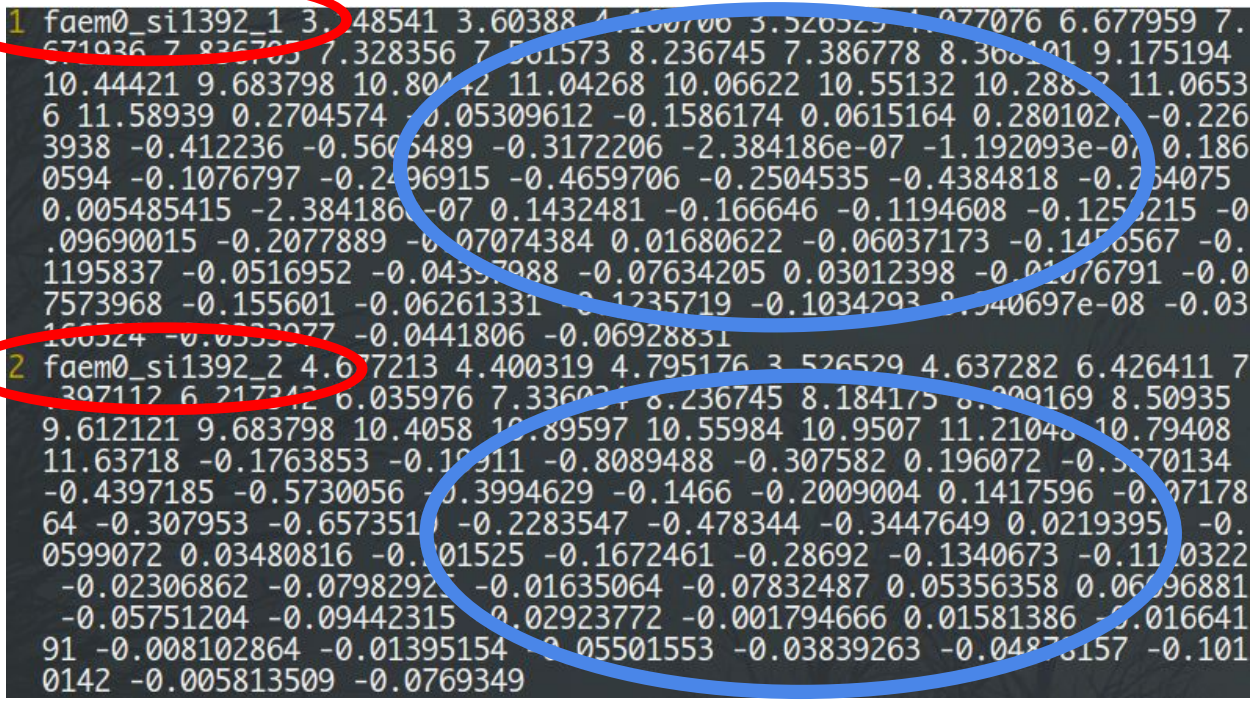
Dataset

- Each instance consists of 3 parts:
  - speaker **faem0**, sentence **si1392**, the **37th frame**



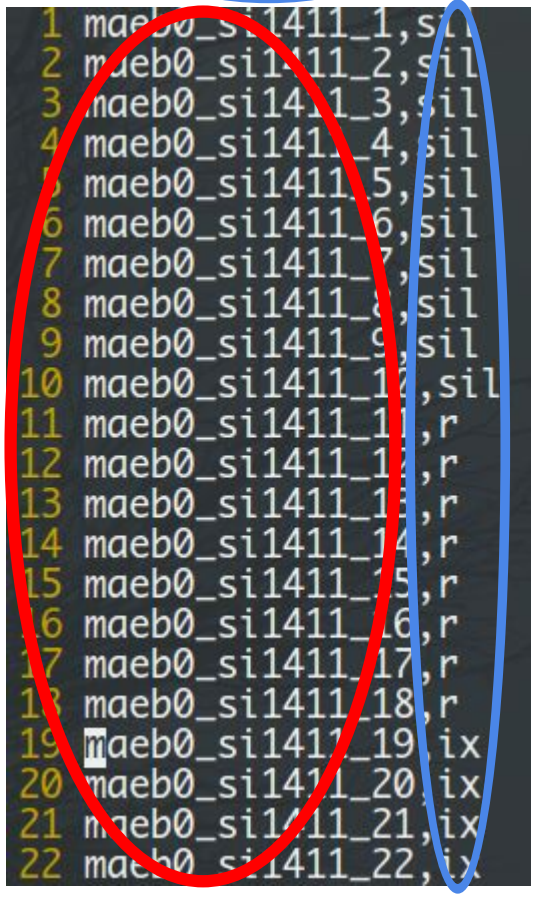
Data Format

- **WAV file: Speak-Sentence ID + .wav**
  - Check by your ear(s)
- **ARK file: Instance ID + features**



Data Format

- **LAB file: Instance ID + comma + label**
- state label and 48 phones
- you have to map them to 39 phonemes **yourselves.**



Data Format

- **MAP file: 2 mapping**
  - State ID - 48 phones - 39 phones
  - 48 phones - 39 phones
  - Delimiter: tab '\t'

0	sil	sil
1	aa	aa
2	ae	ae
3	ah	ah
4	ao	aa
5	aw	aw
6	ax	ah
7	ay	ay
8	b	b
9	ch	ch
10	cl	sil

aa	aa
ae	ae
ah	ah
ao	aa
aw	aw
ax	ah
ay	ay
b	b
ch	ch
cl	sil

Data Format

- **CSV file**: Your prediction when submission.
- **Must mapped to 39-phonemes**
- **With header row: "Id,Prediction"**
- **Instance ID + comma + 39-phonemes**

```
Id,Prediction
fadg0_si1279_1,sil
fadg0_si1279_2,sil
fadg0_si1279_3,sil
fadg0_si1279_4,sil
fadg0_si1279_5,sil
fadg0_si1279_6,sil
fadg0_si1279_7,sil
fadg0_si1279_8,sil
fadg0_si1279_9,sil
fadg0_si1279_10,sil
```

SUBMISSION REQUIREMENT

Submission(Kaggle)

- Your predict.csv.
- Twice a day.
- 50 % public score and 50% private score

#	Δ1d	Team Name	Score	Entries	Last Submission UTC (Best – Last Submission)
1	—	**Simpdanny Lu**	0.60880	2	Mon, 02 Mar 2015 15:12:21 (-4.3d)
		Simple Baseline	0.55088		
-		wyc2010	Waiting to pounce...	-	

Submission(Ceiba)

- **Your source code and documentation**
  - Usage and environment Setting
  - Package dependency
  - **NO NN-based toolkit is allowed.**
    - Only matrix manipulation acceleration package can be utilized.
- **Report**
- Upload 1 copy per group.

Report

- **Group Information 10%**
  - Group Name, Student Name, Student ID
  - Member Contribution
- **What have you done? 40%**
  - Data structure and algorithm design
  - Data preprocessing
  - Implementation tips and obstacles
  - Bug(s) and how to solve it(them).

Report

- **Experiment Setting and Results 50%**
  - How you design your experiments
  - Compare different models/techniques
- **No more than 4 pages with font 12, A4 size.**

Grading Policy

- **Accuracy 60%**
  - Simple Baseline in Kaggle (released day 7)
  - Once achieve the baseline, you can get the full credit.
  - 1 % less absolute accuracy = 1% absolute credit loss
- **Report 40%**
- **Implementation 20%.**
- **Bonus**
  - First Place 15%
  - First Runner-up 10%
  - Second Runner-up 5%

Penalty

- **Latency**
  - Half-life = 24 hrs
  - The excess time will round up to hours.

- **Usage on NN-based toolkit**
  - **You shall get 0 credit in both implementation and accuracy parts.**
  - **Please ask TAs whether you could use the package or not.**

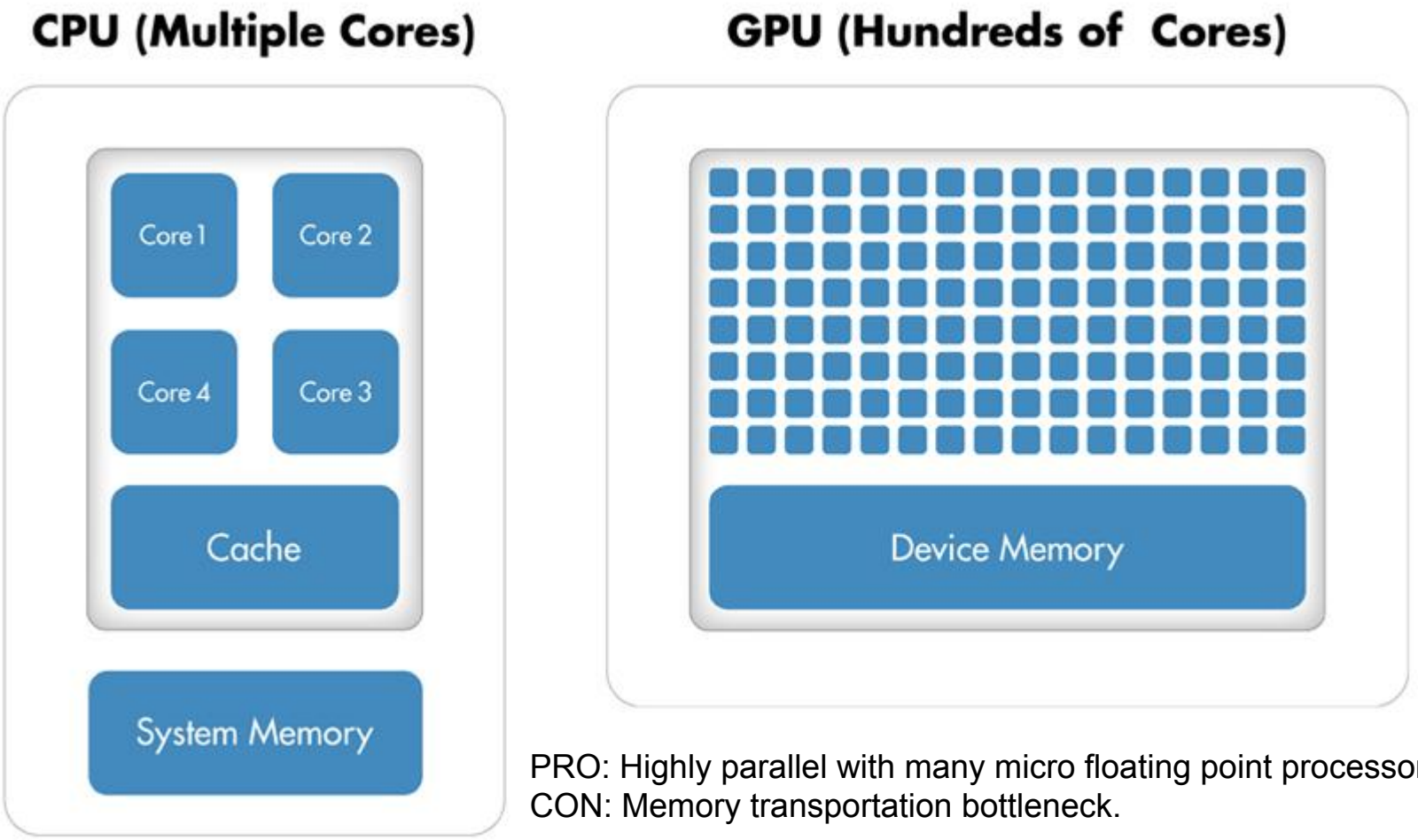
GPU Acceleration

Scalability Issue

- **GPU Acceleration**
  - C/C++: CUDA library
  - Python: Theano
  - Matlab: GPU toolbox

- **Typically, acceleration lies about 5-20 times.**
  - 20 – 80 hrs -> 4 hrs
  - You got 3 weeks to have your HW1 done.

GPU Acceleration



PRO: Highly parallel with many micro floating point processors.
CON: Memory transportation bottleneck.

GPU Acceleration - C/C++

- CUDA
- A good toolkit
- https://github.com/botonchou/libcumatrix
- Or write your toolkit! (Hard)
- https://developer.nvidia.com/cublas

GPU Acceleration - Python

- **Theano is recommended**
- **Follow the instructions of Theano configuration**
- **http://deeplearning.net/software/theano/library/config.html**

GPU Acceleration - Matlab

- **gpuArray**
- **Usage**

```
Elapsed time is 1.542150 seconds.
Elapsed time is 0.004115 seconds.
```

- **Reference**

```
% Init data
A = rand( 2000, 4000 );
gpu_A = gpuArray(A);

B = rand( 4000, 5000 );
gpu_B = gpuArray(B);

% Perform A*B
tic;
C = A * B;
toc;

tic;
gpu_C = gpu_A* gpu_B;
toc;
```

Recommendation

- **Start as early as possible.**
  - In TA's **poor** machine(Nvidia GTX 660) , 1 epoch costs about 200 seconds.
  - 10 epochs are sufficient to achieve baseline if correctly implemented.
  - Bug is everywhere.

Tips

Tip 1

- **Start-ups!**
  - **Group discussion and task assignments**
  - **Derive all formulas. Everyone.**
  - **Get familiar with Kaggle.**

Tip 2

- **Understand what happened….**
  - **Look over training data and testing data**
  - **Many decision**
  - **Data preprocessing.**
  - **Basic structure and core methods implementation.**
  - **Test on tiny set(make it yourselves!).**

Tip 3

- **Everything settled down**
  - **Start training and report testing results.**
  - **Debugging…**
  - **Can you overfit the training data?**
  - **Precision overflow? (exponential)**
  - **Matrix multiplication?**
  - **Error backpropagation?**

Tip 4

- **Basic learning techniques**
  - **Parameter selection**
    - batch size, learning rate, or…
  - **Frame shuffle?**
  - **Splice features in context window?**
  - **Regularization(weight decay)?**
  - **Learning rate decay?**

Tip 5

- **Advanced techniques**
  - **Restricted Boltzmann Machines(RBM) in DNN pre-training**
  - **Depth vs Breadth**
  - **AdaGrad**
  - **Dropout**
  - **Momentum**
  - **ReLU**
  - **Maxout**

Tip 6

- **Final Setting**
  - **Design systematic experiments**
  - **Complete report and your code documentation**
  - **Provide results and your settings.**

Baselines

Simple Baseline - **0.5285**

- Once achieve this score, **you will get full credit in the grading part of accuracy**.
- However, only pure DNN is considered. Your mix learning techniques would help you better win the competition in Kaggle, but **your DNN alone should be powerful enough**.
- You must present your experiment results with pure DNN **in the report**.

Simple Baseline - **0.5285**

Input	69	fbank features
Hidden	128	only one hidden layer
Output	48	-
Batchsize	128	-
Initial Learning Rate	0.0001	-
Learning Rate Decay	0.9999	decay with each batch updating
Initialization Variance	0.1	Model initial weights gaussian variance
Momentum	0.9	-
Training Time	about 5.5 mins	5 epochs

Real Baseline - **0.6174**

- Due to the structure of the speech, **splicing contextual frames** (say previous 4 and subsequent 4) would usually helps
- Increase the model complexity with usage of **state level** pronunciation unit.
- A **real 2-3 days** training time required.

Real Baseline - **0.6174**

Input	621	previous 4 frames + subsequent 4 frames + center 1 frame = 9 frames fbank features
Hidden	1024-1024-1024-1024	4 layers
Output	1943	state label
Batchsize	256	-
Learning Rate	0.001	-
Dropout	0.1	-
Initialization	random [-0.1. 0.1]	uniform
Momentum	0.5	-
Training Time	2-3 days	600-700 seconds per epoch

Boton Baseline - **0.6953**

- Boton was graduated from Speech Processing Lab. at the year of 2015.
- Author of toolkit **libdnn**.
  - Note that you **CANNOT** use this one on your HW1.
- We utilized his scripts to train this task **without** further speech processing techniques.
- Convolutional Neural Network
- Gaussian Normalization

Boton Baseline - **0.6953**

Input	12x16x16 = 3072	previous 10 frames + subsequent 5 frames + center 1 frame = 16 frames fbank-64 features with order 2 delta features
Hidden	30x3x3-30x3x3-30x3x3-2s-2047-2047	3 conv layers + 1 subsampling layer + 2 hidden layers
Output	1943	state label
Batchsize	256	-
Learning Rate	0.01	-
Dropout	0.1	-
Initialization	random [-0.1. 0.1]	uniform
Momentum	0.5	-
Training Time	10 days	6000-7000 seconds per epoch

Kaldi Baseline - **0.7651**

- Kaldi is a well-known speech recognition open source software.
- Your training alignments are provided by Kaldi.
  - However, I do modify the scripts for better format and comprehension.
- Speaker adaptive learning, pre-training or other advance techniques are merged together for better performance.

Kaldi Baseline - **0.7651**

Input	429	previous 5 frames + subsequent 5 frames + center 1 frame = 11 frames MFCC features
Hidden	1024-1024-1024-1024	4 layers
Output	1943	state label
Initialization	RBM	-
Learning Rate	0.008	-
Other techniques	fMLLR, stacked RBM pretraining,, bottleneck features...	