មហាវិទ្យាល័យវិស្វកម្ម
FACULTY OF ENGINEERING

Introduction to Deep Learning Applications and Theory

Lecture 6
Testing ANN model

Chhoeum Vantha, Ph.D.
Telecom & Electronic Engineering

Previous Week

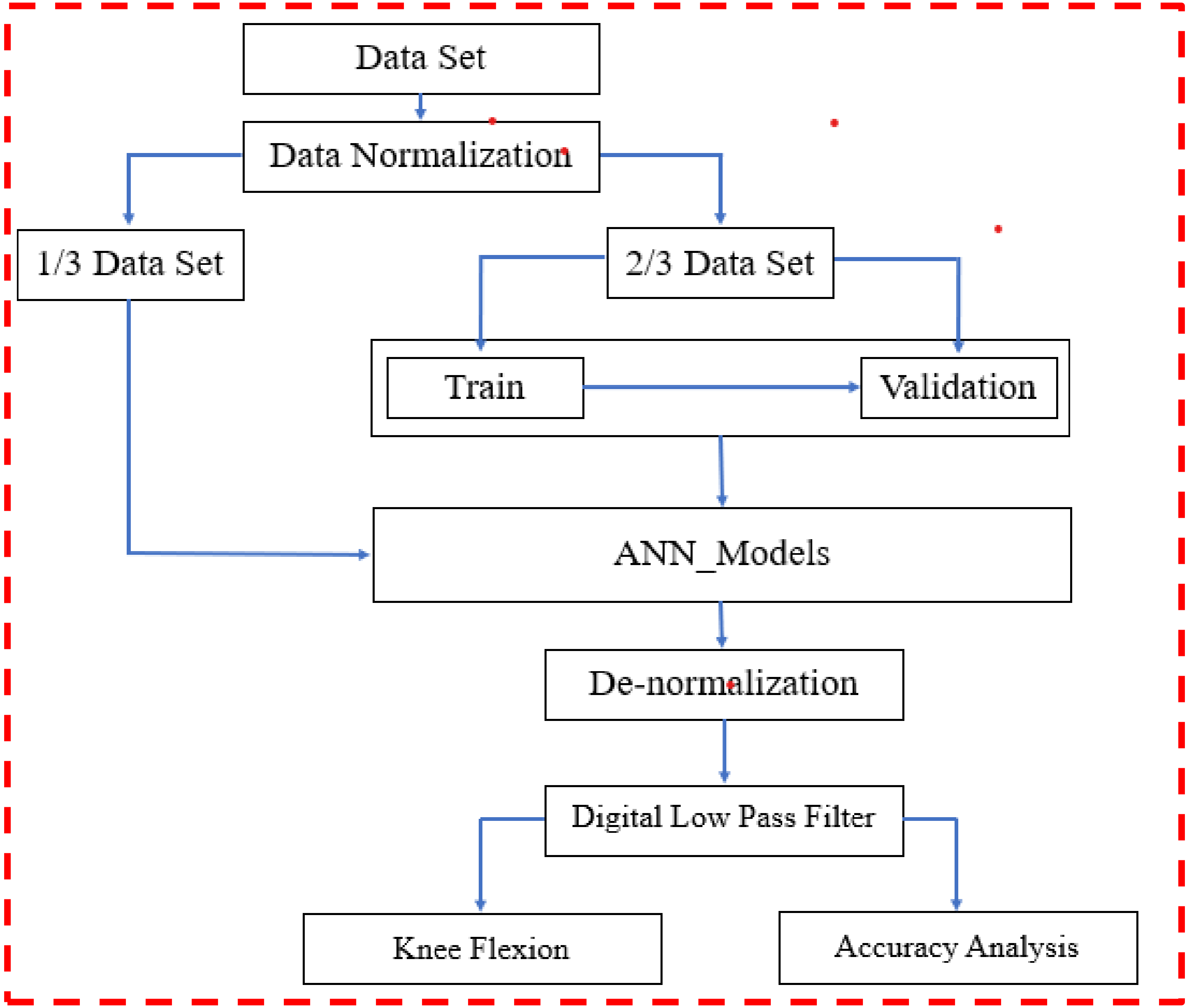
- Layers: Python Operators
- Optimizer
- Summarize Model
- Training Loss
- Validation Loss
- Designing Training Procedures

Contents

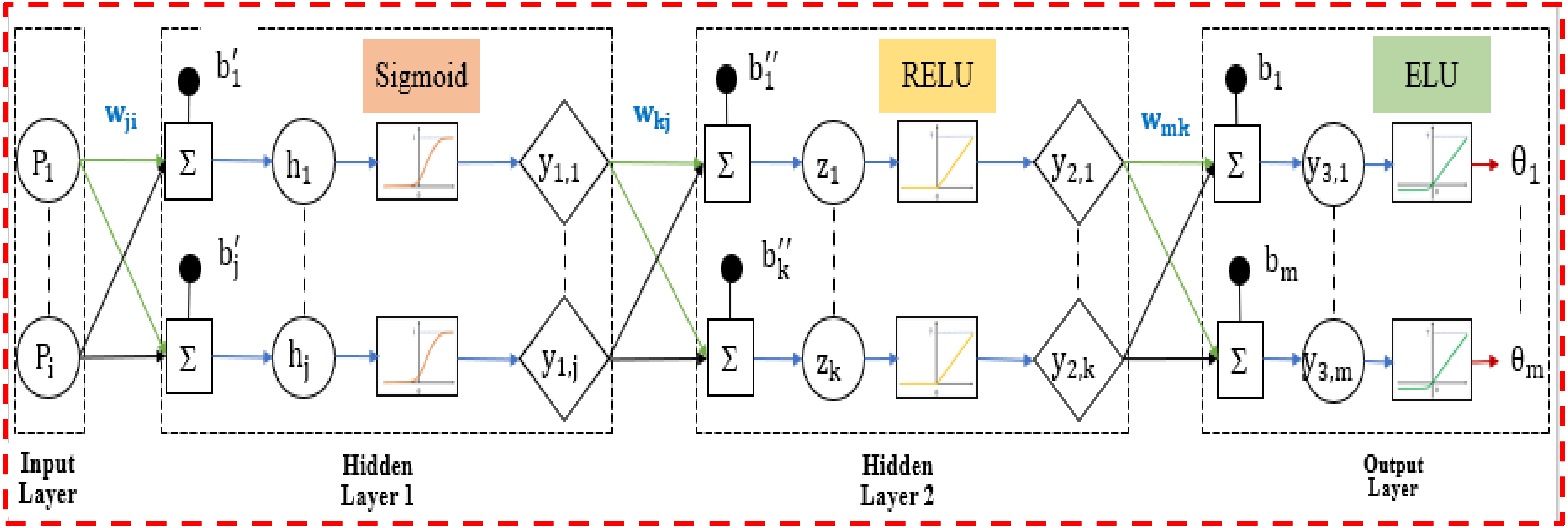
- ANN
  - Dataset
  - Creating ANN model
    - Mathematics behind Deep Learning
    - Check summary of model
    - Visualization (loss and error)
    - Save DL model
  - Testing on Saving model
  - Accuracy Analysis

Data Set

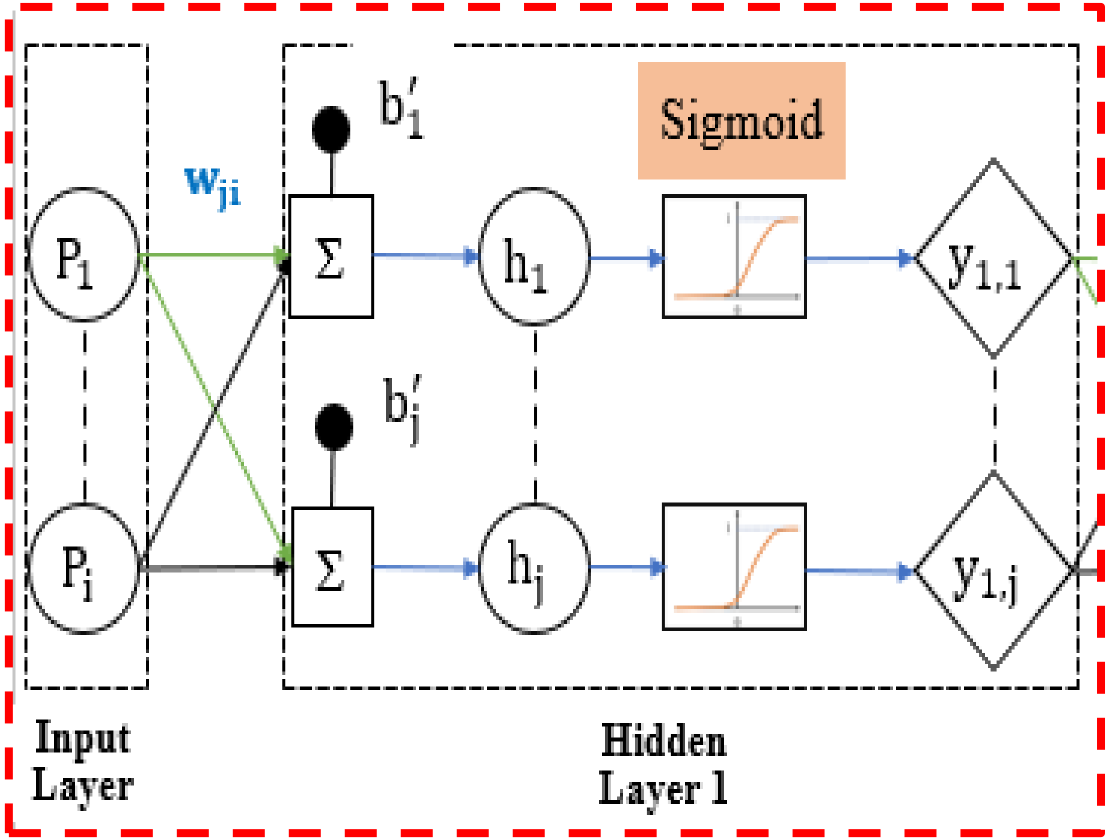
The whole workflow of ANN models



Block diagram of the four layers ANN



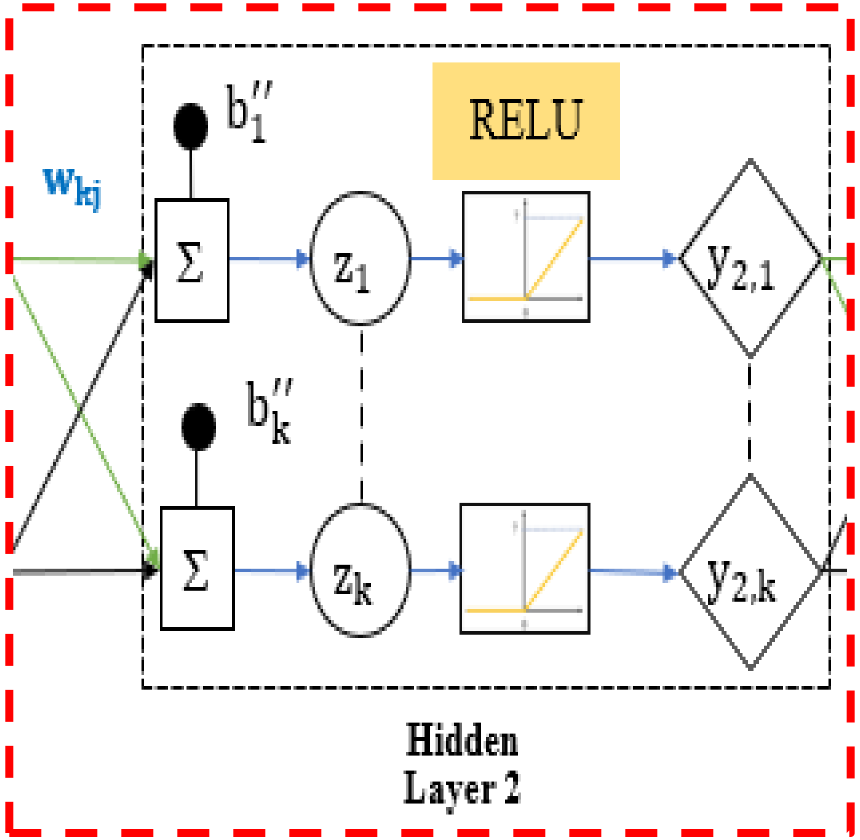
Mathematics behind Deep Learning



$$h_j(n) = \sum_{i=1}^{N^l} P_i(n) w_{ji} + b'_j,$$

$$y_{1,j}(n) = \sigma(h_j(n)) = \frac{1}{1 - e^{h_j(n)}},$$

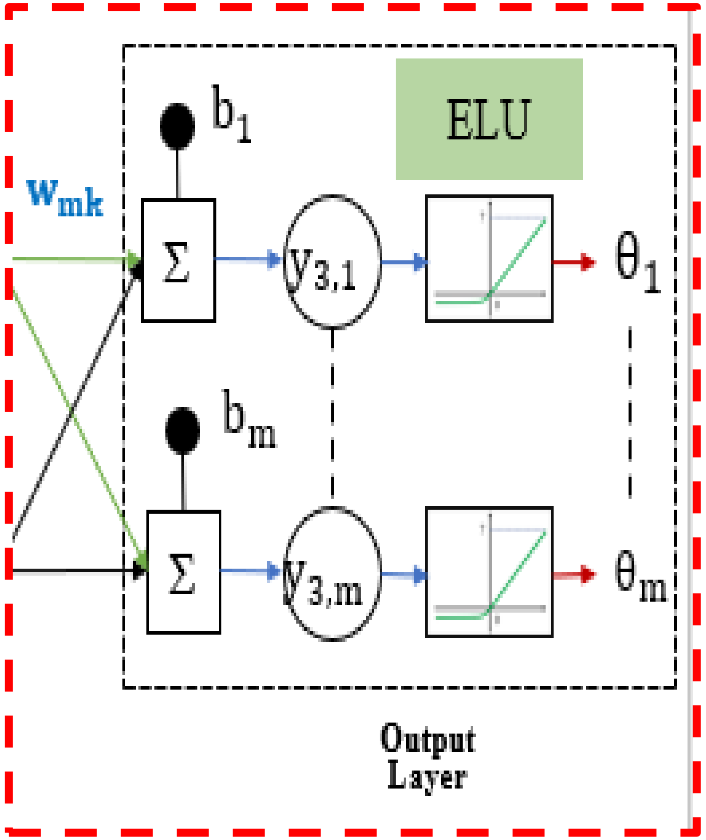
Mathematics behind Deep Learning



$$z_k(n) = \sum_{j=1}^{N^h} y_{1,j}(n) w_{kj} + b_k'',$$

$$y_{2,k}(n) = \mathrm{RELU}(z_k(n)) = \max\,(0, z_k(n)),$$

Mathematics behind Deep Learning



$$y_{3,m}(n) = \sum_{k=1}^{N^z} y_{2,k}(n) w_{mk} + b_m,$$

$$\theta_m(n) = \mathrm{ELU}\big(y_{3,m}(n)\big), \begin{cases} \theta_m(n) = y_{3,m} & y_{3,m} \geq 0 \\ \theta_m(n) = \alpha(e^{y_{3,m}} - 1) & y_{3,m} < 0 \end{cases}$$

Evaluation methods

- **The root mean squared error (RMSE):** either one of two closely related and frequently used measures of the differences between true or predicted values (observed values or an estimator on the other)

Accuracy Analysis

Root mean squared error

In scikit-learn: **root_mean_squared_error**

$$\mathrm{RMSD} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{N}}$$

RMSD = root-mean-square deviation

i = variable i

N = number of non-missing data points

x_i = actual observations time series

$\hat{x}_i$ = estimated time series

```
sklearn.metrics.root_mean_squared_error(y_true, y_pred, *,
sample_weight=None, multioutput='uniform_average')
```

Root mean squared error regression loss.

Accuracy Analysis

- **A correlation coefficient** measures the strength and direction of a linear relationship between two variables:
  - +1: Indicates a perfect positive correlation (as one variable increases, the other increases proportionally)
  - **-1:** Indicates a perfect negative correlation (as one variable increases, the other decreases proportionally)
  - 0: Indicates no linear relationship between the variables

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient

x_i = values of the x-variable in a sample

$\bar{x}$ = mean of the values of the x-variable

y_i = values of the y-variable in a sample

$\bar{y}$ = mean of the values of the y-variable

Accuracy Analysis

Coefficient of determination

In scikit-learn: **r2_score**

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination

RSS = sum of squares of residuals

TSS = total sum of squares

```
sklearn.metrics.r2_score(y_true, y_pred, *, sample_weight=None,
multioutput='uniform_average', force_finite=True)
```

R^2 (coefficient of determination) regression score function.

Accuracy Analysis

- **Mean absolute error:** measures the average absolute difference between the predicted values and the actual target values

Accuracy Analysis

Mean absolute error

In scikit-learn: **root_mean_squared_error**

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE = mean absolute error

y_i = prediction

x_i = true value

n = total number of data points

```
sklearn.metrics.mean_absolute_error(y_true, y_pred, *,
sample_weight=None, multioutput='uniform_average')
```

Mean absolute error regression loss.

Practice 6

Design your own ANN model

a. Draw the final of you ANN model: Train until you get the generalize model

b. Evaluate your model performance:
   - The root mean squared error (RMSE)
   - Mean relative error (MRE)
   - Correlation efficiency (R-value)
   - Mean squared error (MSE)
   - Plot to see the pattern of your model performance

Thanks!