

Introduction to ML & DL

Shan-Hung Wu
shwu@cs.nthu.edu.tw

Department of Computer Science,
National Tsing Hua University, Taiwan

Machine Learning

Outline

- 1 What's Machine Learning?
- 2 What's Deep Learning?
- 3 About this Course...
- 4 FAQ

Outline

① What's Machine Learning?

② What's Deep Learning?

③ About this Course...

④ FAQ

Prior vs. Posterior Knowledge

- To solve a problem, we need an algorithm
 - E.g., sorting

Prior vs. Posterior Knowledge

- To solve a problem, we need an algorithm
 - E.g., sorting
 - *A priori knowledge* is enough

Prior vs. Posterior Knowledge

- To solve a problem, we need an algorithm
 - E.g., sorting
 - *A priori knowledge* is enough
- For some problem, however, we do not have the a priori knowledge
 - E.g., to tell if an email is spam or not
 - The correct answer varies in time and from person to person

Prior vs. Posterior Knowledge

- To solve a problem, we need an algorithm
 - E.g., sorting
 - *A priori knowledge* is enough
- For some problem, however, we do not have the a priori knowledge
 - E.g., to tell if an email is spam or not
 - The correct answer varies in time and from person to person
- Machine learning algorithms use the *a posteriori knowledge* to solve problems

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
 - E.g., sorting
 - *A priori knowledge* is enough
- For some problem, however, we do not have the a priori knowledge
 - E.g., to tell if an email is spam or not
 - The correct answer varies in time and from person to person
- Machine learning algorithms use the *a posteriori knowledge* to solve problems
 - Learnt from *examples* (as extra input)

Example Data \mathbb{X} as Extra Input

- Unsupervised:

$$\mathbb{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N, \text{ where } \mathbf{x}^{(i)} \in \mathbb{R}^D$$

- E.g., $\mathbf{x}^{(i)}$ an email

Example Data \mathbb{X} as Extra Input

- Unsupervised:

$$\mathbb{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N, \text{ where } \mathbf{x}^{(i)} \in \mathbb{R}^D$$

- E.g., $\mathbf{x}^{(i)}$ an email

- Supervised:

$$\mathbb{X} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N, \text{ where } \mathbf{x}^{(i)} \in \mathbb{R}^D \text{ and } \mathbf{y}^{(i)} \in \mathbb{R}^K,$$


- E.g., $y^{(i)} \in \{0, 1\}$ a spam label

General Types of Learning (1/2)

- *Supervised learning*: learn to predict the labels of future data points

$$X \in \mathbb{R}^{N \times D} :$$


$$\mathbf{y} \in \mathbb{R}^{N \times K} : [\mathbf{e}^{(6)}, \mathbf{e}^{(1)}, \mathbf{e}^{(9)}, \mathbf{e}^{(4)}, \mathbf{e}^{(2)}]$$

$$\mathbf{x}' \in \mathbb{R}^D :$$


$$\mathbf{y}' \in \mathbb{R}^K : \textcolor{red}{?}$$

General Types of Learning (1/2)

- **Supervised learning**: learn to predict the labels of future data points

$$X \in \mathbb{R}^{N \times D} :$$



$$y \in \mathbb{R}^{N \times K} : [e^{(6)}, e^{(1)}, e^{(9)}, e^{(4)}, e^{(2)}]$$

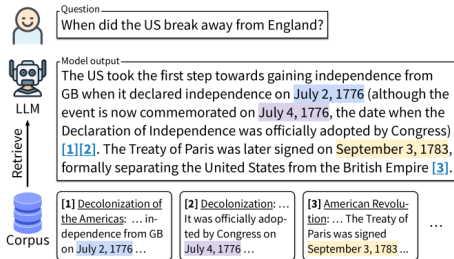
$$x' \in \mathbb{R}^D :$$



$$y' \in \mathbb{R}^K :$$

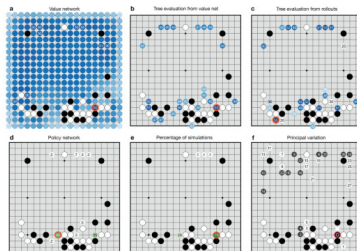
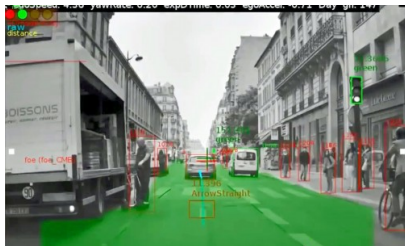
?

- **Unsupervised learning**: learn (latent) patterns in X , and optionally generate new x' s



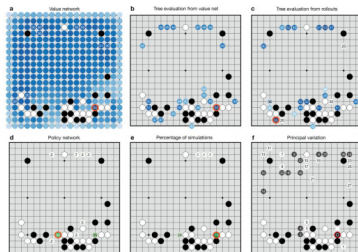
General Types of Learning (2/2)

- **Reinforcement learning:** learn from “good”/“bad” feedback of actions (instead of correct labels) to maximize the goal



General Types of Learning (2/2)

- **Reinforcement learning**: learn from “good”/“bad” feedback of actions (instead of correct labels) to maximize the goal



- AlphaGo [1] is a hybrid of reinforcement learning and supervised learning
 - Supervised learning from the game records
 - Then, reinforcement learning from self-play

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets
- ② Model development
 - ① Assume a **model** $\{f(\cdot; \mathbf{w})\}$ that is a collection of candidate functions f 's (representing posteriori knowledge) we want to discover
 - ① f is assumed to be parametrized by \mathbf{w}

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets
- ② Model development
 - ① Assume a **model** $\{f(\cdot; \mathbf{w})\}$ that is a collection of candidate functions f 's (representing posteriori knowledge) we want to discover
 - ① f is assumed to be parametrized by \mathbf{w}
 - ② Define a **cost function** $C(\mathbf{w}; \mathbb{X})$ (or functional $C[f; \mathbb{X}]$) that measures “how good a particular f can explain the training data”

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets
- ② Model development
 - ① Assume a **model** $\{f(\cdot; \mathbf{w})\}$ that is a collection of candidate functions f 's (representing posteriori knowledge) we want to discover
 - ① f is assumed to be parametrized by \mathbf{w}
 - ② Define a **cost function** $C(\mathbf{w}; \mathbb{X})$ (or functional $C[f; \mathbb{X}]$) that measures “how good a particular f can explain the training data”
- ③ **Training**: employ an algorithm that finds the best (or good enough) function $f^*(\cdot; \mathbf{w}^*)$ in the model that minimizes the cost function

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} C(\mathbf{w}; \mathbb{X})$$

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets
- ② Model development
 - ① Assume a **model** $\{f(\cdot; \mathbf{w})\}$ that is a collection of candidate functions f 's (representing posteriori knowledge) we want to discover
 - ① f is assumed to be parametrized by \mathbf{w}
 - ② Define a **cost function** $C(\mathbf{w}; \mathbb{X})$ (or functional $C[f; \mathbb{X}]$) that measures “how good a particular f can explain the training data”
- ③ **Training**: employ an algorithm that finds the best (or good enough) function $f^*(\cdot; \mathbf{w}^*)$ in the model that minimizes the cost function

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} C(\mathbf{w}; \mathbb{X})$$

- ④ **Testing**: evaluate the performance of the learned f^* using the testing dataset

General Machine Learning Steps

- ① Data collection, preprocessing (e.g., integration, cleaning, etc.), and exploration
 - ① Split a dataset into the training and testing datasets
- ② Model development
 - ① Assume a **model** $\{f(\cdot; \mathbf{w})\}$ that is a collection of candidate functions f 's (representing posteriori knowledge) we want to discover
 - ① f is assumed to be parametrized by \mathbf{w}
 - ② Define a **cost function** $C(\mathbf{w}; \mathbb{X})$ (or functional $C[f; \mathbb{X}]$) that measures “how good a particular f can explain the training data”
- ③ **Training**: employ an algorithm that finds the best (or good enough) function $f^*(\cdot; \mathbf{w}^*)$ in the model that minimizes the cost function

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} C(\mathbf{w}; \mathbb{X})$$

- ④ **Testing**: evaluate the performance of the learned f^* using the testing dataset
- ⑤ Apply the model in the real world

Example for Spam Detection

① Random split of your past emails and labels

① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$

② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$

Example for Spam Detection

① Random split of your past emails and labels

① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$

② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$

② Model development

① **Model:** $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$

Example for Spam Detection

- ① Random split of your past emails and labels
 - ① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$
 - ② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$
- ② Model development
 - ① **Model:** $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$
 - ② **Cost function:** $C(\mathbf{w}; \mathbb{X}) = \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$

Example for Spam Detection

- ① Random split of your past emails and labels
 - ① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$
 - ② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$
- ② Model development
 - ① **Model**: $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$
 - ② **Cost function**: $C(\mathbf{w}; \mathbb{X}) = \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ③ **Training**: to solve $\mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$

Example for Spam Detection

- ① Random split of your past emails and labels
 - ① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$
 - ② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$
- ② Model development
 - ① **Model:** $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$
 - ② **Cost function:** $C(\mathbf{w}; \mathbb{X}) = \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ③ **Training:** to solve $\mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ④ **Testing:** accuracy $\frac{1}{|\mathbb{X}'|} \sum_i 1(\mathbf{x}'^{(i)}, y'^{(i)}; f(\mathbf{x}'^{(i)}; \mathbf{w}^*) = y'^{(i)})$

Example for Spam Detection

- ① Random split of your past emails and labels
 - ① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$
 - ② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$
- ② Model development
 - ① **Model:** $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$
 - ② **Cost function:** $C(\mathbf{w}; \mathbb{X}) = \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ③ **Training:** to solve $\mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ④ **Testing:** accuracy $\frac{1}{|\mathbb{X}'|} \sum_i 1(\mathbf{x}'^{(i)}, y'^{(i)}; f(\mathbf{x}'^{(i)}; \mathbf{w}^*) = y'^{(i)})$
- ⑤ Use f^* to predict the labels of your future emails

Example for Spam Detection

- ① Random split of your past emails and labels
 - ① Training dataset: $\mathbb{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_i$
 - ② Testing dataset: $\mathbb{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_i$
- ② Model development
 - ① **Model:** $\{f : f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}\}$
 - ② **Cost function:** $C(\mathbf{w}; \mathbb{X}) = \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ③ **Training:** to solve $\mathbf{w}^* = \arg \min_{\mathbf{w}} \sum_i 1(\mathbf{w}; f(\mathbf{x}^{(i)}; \mathbf{w}) \neq y^{(i)})$
- ④ **Testing:** accuracy $\frac{1}{|\mathbb{X}'|} \sum_i 1(\mathbf{x}'^{(i)}, y'^{(i)}; f(\mathbf{x}'^{(i)}; \mathbf{w}^*) = y'^{(i)})$
- ⑤ Use f^* to predict the labels of your future emails
 - See Notation

Outline

① What's Machine Learning?

② What's Deep Learning?

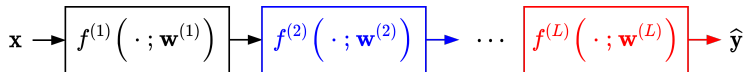
③ About this Course...

④ FAQ

Deep Learning

- ML where an $f(\cdot; \mathbf{w})$ has many (deep) layers

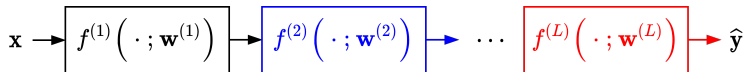
$$\hat{\mathbf{y}} = f^{(L)}(\dots f^{(2)}(f^{(1)}(\mathbf{x}; \mathbf{w}^{(1)}); \mathbf{w}^{(2)}) \dots; \mathbf{w}^{(L)})$$



Deep Learning

- ML where an $f(\cdot; \mathbf{w})$ has many (deep) layers

$$\hat{\mathbf{y}} = f^{(L)}(\dots f^{(2)}(f^{(1)}(\mathbf{x}; \mathbf{w}^{(1)}); \mathbf{w}^{(2)}) \dots; \mathbf{w}^{(L)})$$

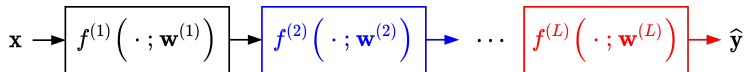


- Pros:
 - Learns to pre-process data automatically
 - Learns a complex function (e.g., visual objects to labels)

Deep Learning

- ML where an $f(\cdot; \mathbf{w})$ has many (deep) layers

$$\hat{\mathbf{y}} = f^{(L)}(\dots f^{(2)}(f^{(1)}(\mathbf{x}; \mathbf{w}^{(1)}); \mathbf{w}^{(2)}) \dots; \mathbf{w}^{(L)})$$



- Pros:
 - Learns to pre-process data automatically
 - Learns a complex function (e.g., visual objects to labels)
- Cons:
 - Usually needs large data to train a model well
 - Higher computation costs (for both training and testing)

Outline

- 1 What's Machine Learning?
- 2 What's Deep Learning?
- 3 About this Course...**
- 4 FAQ

Target Audience

- Graduate (and senior undergraduate) **CS** students
 - Easy-to-moderate level of theory
 - Coding and engineering (in Python)
 - Clean datasets (small & large)

Target Audience

- Graduate (and senior undergraduate) **CS** students
 - Easy-to-moderate level of theory
 - Coding and engineering (in Python)
 - Clean datasets (small & large)
- No prior knowledge about ML is needed

Topics Covered

- Supervised, unsupervised learning, and reinforcement learning

Topics Covered

- Supervised, unsupervised learning, and reinforcement learning
- with *structural* output:



A man holding a tennis racquet on a tennis court.



Two pizzas sitting on top of a stove top oven



A group of young people playing a game of Frisbee



A man flying through the air while riding a snowboard

Syllabus (Tentative)

- Part 1: math review (2 weeks)
 - Linear algebra
 - Probability & information theory
 - Numerical optimization

Syllabus (Tentative)

- Part 1: math review (2 weeks)
 - Linear algebra
 - Probability & information theory
 - Numerical optimization
- Part 2: machine learning basics (3 weeks)
 - Learning theory
 - Parametric/non-parametric models
 - Experiment design

Syllabus (Tentative)

- Part 1: math review (2 weeks)
 - Linear algebra
 - Probability & information theory
 - Numerical optimization
- Part 2: machine learning basics (3 weeks)
 - Learning theory
 - Parametric/non-parametric models
 - Experiment design
- Part 3: deep supervised learning (6 weeks)
 - Neural Networks (NNs), CNNs, RNNs, Transformers

Syllabus (Tentative)

- Part 1: math review (2 weeks)
 - Linear algebra
 - Probability & information theory
 - Numerical optimization
- Part 2: machine learning basics (3 weeks)
 - Learning theory
 - Parametric/non-parametric models
 - Experiment design
- Part 3: deep supervised learning (6 weeks)
 - Neural Networks (NNs), CNNs, RNNs, Transformers
- Part 4: unsupervised learning & generative AI (2 weeks)
 - Autoencoders, manifold learning, generative models

Syllabus (Tentative)

- Part 1: math review (2 weeks)
 - Linear algebra
 - Probability & information theory
 - Numerical optimization
- Part 2: machine learning basics (3 weeks)
 - Learning theory
 - Parametric/non-parametric models
 - Experiment design
- Part 3: deep supervised learning (6 weeks)
 - Neural Networks (NNs), CNNs, RNNs, Transformers
- Part 4: unsupervised learning & generative AI (2 weeks)
 - Autoencoders, manifold learning, generative models
- Part 5: reinforcement learning (2 weeks)
 - Value/gradients policies, action/critics, reinforce RNNs

Grading (Tentative)

- Prerequisite quiz: **10%**
 - ***On next Thu (9/11)***
 - ***You can take this course if you are within top-90***
- Contests (x 3): **45%**
 - At the end of each part
- Assignments: **20%**
 - Come with the labs
- Final exam: **25%**
- Bonus: up to **10%**
 - Math labs (x 4)
 - Optional ML topics (x 2)
 - Course participations

Classes Info

- Lectures on Tue (2 hours)
 - Concepts & theories
 - ~~with companion videos~~
- Labs on Thu (1 hour)
 - Implementation (in Python) & engineering topics
- TA time & more info: [course website](#)

Outline

① What's Machine Learning?

② What's Deep Learning?

③ About this Course...

④ **FAQ**

FAQ (1/2)

Q: Should we team up for the contests?

A: Yes, 2~4 students per team

FAQ (1/2)

Q: Should we team up for the contests?

A: Yes, 2~4 students per team

Q: Which GPU card should I buy?

A: Nvidia GTX 1060 or above; 1050 Ti (4G RAM) minimal

FAQ (1/2)

Q: Should we team up for the contests?

A: Yes, 2~4 students per team

Q: Which GPU card should I buy?

A: Nvidia GTX 1060 or above; 1050 Ti (4G RAM) minimal

Q: Do we need to attend the classes?

A: No, as long as you can pass. But we offer attendance bonus...

FAQ (1/2)

Q: Should we team up for the contests?

A: Yes, 2~4 students per team

Q: Which GPU card should I buy?

A: Nvidia GTX 1060 or above; 1050 Ti (4G RAM) minimal

Q: Do we need to attend the classes?

A: No, as long as you can pass. But we offer attendance bonus...

Q: Is this a light-loading course or heavy-loading one?

*A: Should be **very heavy** to most students. Please **reserve your time***

FAQ (2/2)

Q: What's the textbook?

A: No formal textbook. But if you need one, read the [Deep Learning](#) book

FAQ (2/2)

Q: What's the textbook?

A: No formal textbook. But if you need one, read the [Deep Learning](#) book

Q: Why some sections are marked with "" or "**" in the slides?*

A: The mark "*" means "can be skipped for the first time reader," and
" **" means "materials for reference only"

TODO

- Assigned reading:
 - Calculus
 - Get your feet wet with Python

Reference I

- [1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al.
Mastering the game of go with deep neural networks and tree search.
Nature, 529(7587):484–489, 2016.