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. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
  - *A priori knowledge* is enough
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
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
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 $\mathbf{X}$ as Extra Input

- Unsupervised:
  \[ \mathbf{X} = \{x^{(i)}\}_{i=1}^{N}, \text{ where } x^{(i)} \in \mathbb{R}^{D} \]
  
  - E.g., $x^{(i)}$ an email

- Supervised:
  \[ \mathbf{X} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}, \text{ where } x^{(i)} \in \mathbb{R}^{D} \text{ and } y^{(i)} \in \mathbb{R}^{K} \]
  
  - E.g., $y^{(i)} \in \{0, 1\}$ a spam label
Example Data $\mathbf{X}$ as Extra Input

- Unsupervised:
  $$\mathbf{X} = \{x^{(i)}\}_{i=1}^{N}, \text{ where } x^{(i)} \in \mathbb{R}^D$$
  - E.g., $x^{(i)}$ an email

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

\[
X \in \mathbb{R}^{N \times D} : \begin{bmatrix} 6 & 1 & 9 & 4 & 2 \end{bmatrix} \quad \quad \quad x' \in \mathbb{R}^N : \begin{bmatrix} \mathcal{S} \end{bmatrix}
\]

\[
y \in \mathbb{R}^{N \times K} : \begin{bmatrix} e^{(6)} & e^{(1)} & e^{(9)} & e^{(4)} & e^{(2)} \end{bmatrix} \quad \quad \quad y' \in \mathbb{R}^K : ?
\]
General Types of Learning (1/2)

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

  \[
  X \in \mathbb{R}^{N \times D} : \begin{bmatrix} 6 & 1 & 9 & 4 & 2 \end{bmatrix} \quad x' \in \mathbb{R}^N : \begin{bmatrix} 5 \end{bmatrix}
  \]

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

- **Unsupervised learning**: learn patterns or latent factors in \( X \)
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. The latter is used to tell how good a “move” performed by an agent.
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
    - The latter is used to tell how good a “move” performed by an agent
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
   - Split a dataset into the training and testing datasets

Shan-Hung Wu (CS, NTHU)
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 \( \{f\} \) that is a collection of candidate functions \( f \)'s (representing posteriori knowledge) we want to discover
      1. \( f \) may be parametrized by \( w \)

Shan-Hung Wu (CS, NTHU)
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\} that is a collection of candidate functions \(f\)'s (representing posteriori knowledge) we want to discover
      1. \(f\) may be parametrized by \(w\)
   2. Define an cost function \(C(w)\) (or functional \(C[f]\)) 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
   - Split a dataset into the training and testing datasets

2. Model development
   - Assume a *model* \( \{f\} \) that is a collection of candidate functions \( f \)’s (representing postiori knowledge) we want to discover
     - \( f \) may be parametrized by \( w \)
   - Define an *cost function* \( C(w) \) (or functional \( C[f] \)) 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 \( f^* \) in the model that minimizes the cost function over the training dataset
General Machine Learning Steps

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

2. Model development
   - Assume a model \( \{f\} \) that is a collection of candidate functions \( f \)'s (representing posteriori knowledge) we want to discover
     - \( f \) may be parametrized by \( w \)
   - Define an cost function \( C(w) \) (or functional \( C[f] \)) 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 \( f^* \) in the model that minimizes the cost function over the training dataset

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\} that is a collection of candidate functions f’s (representing posteriori knowledge) we want to discover
      1. f may be parametrized by w
   2. Define an *cost function* \(C(w)\) (or functional \(C[f]\)) 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 \(f^*\) in the model that minimizes the cost function over the training dataset
4. *Testing*: evaluate the performance of the learned \(f^*\) using the testing dataset
5. Apply the model to the real world
Example for Spam Detection

1. Random split of your past emails and labels
   1. Training dataset: $X = \{(x^{(i)}, y^{(i)})\}_i$
   2. Testing dataset: $X' = \{(x'^{(i)}, y'^{(i)})\}_i$
Example for Spam Detection

1. Random split of your past emails and labels
   - Training dataset: \( X = \{ (x^{(i)}, y^{(i)}) \} \)
   - Testing dataset: \( X' = \{ (x'^{(i)}, y'^{(i)}) \} \)

2. Model development
   - Model: \( \{ f : f(x; w) = w^\top x \} \)
Example for Spam Detection

1. Random split of your past emails and labels
   1. Training dataset: \( \mathbf{X} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i} \)
   2. Testing dataset: \( \mathbf{X}' = \{(\mathbf{x}'^{(i)}, y'^{(i)})\}_{i} \)

2. Model development
   1. **Model**: \( \{f : f(x; w) = w^\top x\} \)
   2. **Cost function**: \( C(w) = \sum_i 1(\mathbf{x}^{(i)}; f(\mathbf{x}^{(i)}; w) \neq y^{(i)}) \)
Example for Spam Detection

1. Random split of your past emails and labels
   1. Training dataset: $X = \{(x^{(i)}, y^{(i)})\}_i$
   2. Testing dataset: $X' = \{(x'^{(i)}, y'^{(i)})\}_i$

2. Model development
   1. **Model**: $f : f(x; w) = w^\top x$
   2. **Cost function**: $C(w) = \sum_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)})$

3. **Training**: to solve $w^* = \arg\min_w \sum_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)})$
Example for Spam Detection

1. Random split of your past emails and labels
   1. Training dataset: \( \mathbf{X} = \{(x^{(i)}, y^{(i)})\}_i \)
   2. Testing dataset: \( \mathbf{X}' = \{(x^{(i)}, y^{(i)})\}_i \)

2. Model development
   1. **Model**: \( \{f : f(x; \mathbf{w}) = \mathbf{w}^\top x\} \)
   2. **Cost function**: \( C(\mathbf{w}) = \sum_i 1(x^{(i)}; f(x^{(i)}; \mathbf{w}) \neq y^{(i)}) \)

3. **Training**: to solve \( \mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_i 1(x^{(i)}; f(x^{(i)}; \mathbf{w}) \neq y^{(i)}) \)

4. **Testing**: accuracy \( \frac{1}{|\mathbf{X'}|} \sum_i 1(x'^{(i)}; f(x'^{(i)}) = y'^{(i)}) \)
Example for Spam Detection

1. Random split of your past emails and labels
   1. Training dataset: \( X = \{ (x^{(i)}, y^{(i)}) \}_i \)
   2. Testing dataset: \( X' = \{ (x'^{(i)}, y'^{(i)}) \}_i \)

2. Model development
   1. Model: \( \{ f : f(x; w) = w^\top x \} \)
   2. Cost function: \( C(w) = \Sigma_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)}) \)

3. Training: to solve \( w^* = \arg\min_w \Sigma_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)}) \)

4. Testing: accuracy \( \frac{1}{|X'|} \Sigma_i 1(x'^{(i)}; f(x'^{(i)}) = y'^{(i)}) \)

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

2. Model development
   1. **Model**: \( \{f: f(x; w) = w^\top x\} \)
   2. **Cost function**: \( C(w) = \sum_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)}) \)

3. **Training**: to solve \( w^* = \arg\min_w \sum_i 1(x^{(i)}; f(x^{(i)}; w) \neq y^{(i)}) \)

4. **Testing**: accuracy \( \frac{1}{|X'|} \sum_i 1(x'^{(i)}; f(x'^{(i)}) = y'^{(i)}) \)

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 using models that have many layers (deep)
Deep Learning

- ML using models that have many layers (deep)
- **Pros:**
  - Ene to end—leans how to present data and simplifies data preprocessing
  - Leans complex functions $f$ (e.g., visual objects)
- **Cons:**
Deep Learning

- ML using models that have many layers (deep)

Pros:
- Ene to end—leans how to present data and simplifies data preprocessing
- Leans complex functions $f$ (e.g., visual objects)

Cons:
- Usually need large data to be trained well
Outline

1. What’s Machine Learning?
2. What’s Deep Learning?
3. About this Course...
4. FAQ
Target Audience

- Senior undergraduate and graduate students
  - Easy-to-moderate level of theory
  - Coding and engineering (in Python)
  - Clean datasets (small & large)
Target Audience

- Senior undergraduate and graduate students
  - Easy-to-moderate level of theory
  - Coding and engineering (in Python)
  - Clean datasets (small & large)
- 12 TAs, one for each topic
Topics Covered

- Supervised, unsupervised learning, and reinforcement learning
Topics Covered

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

  - [Image of a man holding a tennis racquet on a tennis court.]
  - [Image of two pizzas sitting on top of a stove top oven.]
  - [Image of a group of young people playing a game of Frisbee.]
  - [Image of a man flying through the air while riding a snowboard.]
Syllabus (Tentative)

- **Part 1: math review (3 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 (4 weeks)**
  - Neural Networks (NNs), CNNs, RNNs

- **Part 4: unsupervised learning (2 weeks)**
  - Autoencoders, manifold learning, GANs

- **Part 5: reinforcement learning (3 weeks)**
  - Value/gradient policies, action/critics, reinforce RNNs
Syllabus (Tentative)

- Part 1: math review (3 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 (4 weeks)
  - Neural Networks (NNs), CNNs, RNNs

- Part 4: unsupervised learning (2 weeks)
  - Autoencoders, manifold learning, GANs

- Part 5: reinforcement learning (3 weeks)
  - Value/gradient policies, action/critics, reinforce
Syllabus (Tentative)

- Part 1: math review (3 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 (4 weeks)
  - Neural Networks (NNs), CNNs, RNNs
- Part 4: unsupervised learning (2 weeks)
  - Autoencoders, manifold learning, GANs
- Part 5: reinforcement learning (3 weeks)
  - Value/gradient policies, action/critics, reinforce RNNs
Syllabus (Tentative)

- Part 1: math review (3 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 (4 weeks)
  - Neural Networks (NNs), CNNs, RNNs
- Part 4: unsupervised learning (2 weeks)
  - Autoencoders, manifold learning, GANs
Syllabus (Tentative)

- Part 1: math review (3 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 (4 weeks)
  - Neural Networks (NNs), CNNs, RNNs
- Part 4: unsupervised learning (2 weeks)
  - Autoencoders, manifold learning, GANs
- Part 5: reinforcement learning (3 weeks)
  - Value/gradient polocies, action/critics, reinforce RNNs
Grading (Tentative)

- Contests (x 5): 75%
  - At the end of each part
- Assignments: 25%
  - Come with the labs
Classes Info

- Lectures
  - Concepts & theories
- Labs
  - Implementation (in Python) & engineering topics

*They are mixed*

- More info can be found in the 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
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 you have 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 you have 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
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”
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”

Q: Can I be enrolled?
A: Variety and juniors take priority
TODO

Assigned reading:
- Calculus
- Get your feet wet with Python
Mastering the game of go with deep neural networks and tree search. 