Discrete Mathematics and Its Applications 2 (CS147)

Lecture 1: Introduction to the module

Fanghui Liu

Department of Computer Science, University of Warwick, UK

Module Organizers

- Fanghui Liu (Week 1 to Week 5): fanghui.liu@warwick.ac.uk
- Ramanujan Sridharan (Week 6 to Week 10): R.Maadapuzhi-Sridharan@warwick.ac.uk

Details can be found in module webpage:
https://warwick.ac.uk/fac/sci/dcs/teaching/modules/cs147/

- google search: Warwick CS -> https://warwick.ac.uk/fac/sci/dcs/
- Teaching -> Modules Taught -> CS147

What the module is about

- The word “application" in the name refers to “applications in theoretical computer science (TCS)" and “applications in machine learning theory"
- This is a (completely) mathematical module.

What the module is about

- The word “application" in the name refers to “applications in theoretical computer science (TCS)" and “applications in machine learning theory"
- This is a (completely) mathematical module.

◦ theoretical computer science: theory of computation, **algorithms** analysis
◦ machine learning theory:

- machine learning: learn rules from data
- theory: TCS, statistical principles

Reasons to analyse algorithms

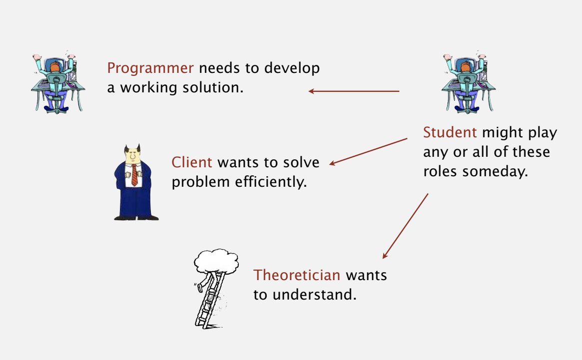

figure credit: Princeton COS226

Reasons to analyse algorithms

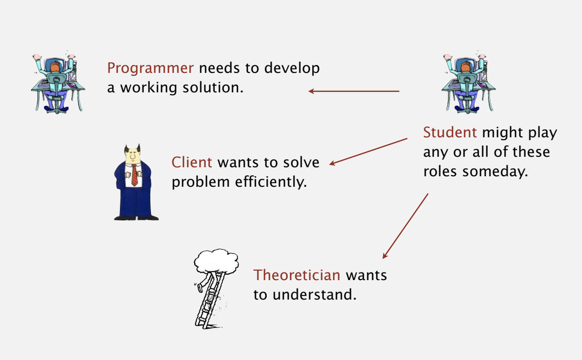

figure credit: Princeton COS226

- predict performance
- compare algorithms
- provide guarantees

Example: image recognition

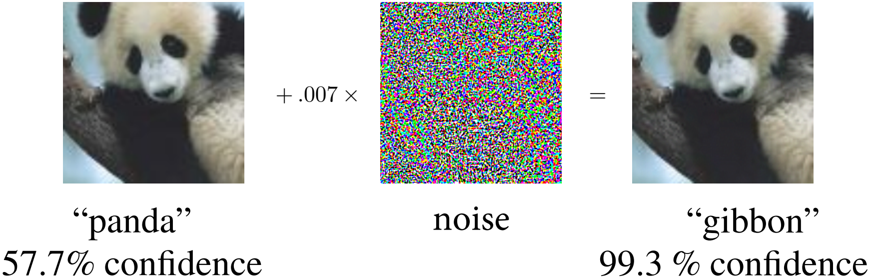

Figure: sensitive output by algorithms. source from [GSS15]

Contents of the module

- CS146: proofs, sequences, sets, relations...
- CS147: algorithm analysis...

Contents of the module

- CS146: proofs, sequences, sets, relations...
- CS147: algorithm analysis...
  - How to analyse runtimes of algorithms
  - discrete probability
  - graph theory and combinatonics

$\Rightarrow$ will prepare you for TCS modules in years 2,3 as well as machine learning theory

*Examples in TCS, ML theory

Corollary 1.3. *Let $\mathcal{D}$ be the distribution over pairs $(x, y) \in \mathbb{R}^d \times \mathbb{R}$ where $x \sim \mathcal{N}(0, \mathrm{Id})$ and $y = F(x)$ for a size-S ReLU network F for which the product of the spectral norms of its weight matrices is a constant.*

Then there is an algorithm that draws $N = d \log(1/\delta) \exp(O(k^3/\varepsilon^2 + kS))$ samples, runs in time $\widetilde{O}(d^2 \log(1/\delta)) \exp(O(k^3 S^2/\varepsilon^2 + kS^3))$, and outputs a ReLU network $\widetilde{F}$ such that $\mathbb{E}[(y - \widetilde{F}(x))^2] \le \varepsilon$ with probability at least $1 - \delta$.

Figure: sample complexity and time complexity [CKM22].

*Examples in TCS, ML theory

Corollary 1.3. *Let $\mathcal{D}$ be the distribution over pairs $(x, y) \in \mathbb{R}^d \times \mathbb{R}$ where $x \sim \mathcal{N}(0, \mathrm{Id})$ and $y = F(x)$ for a size-S ReLU network F for which the product of the spectral norms of its weight matrices is a constant.*

Then there is an algorithm that draws $N = d\log(1/\delta)\exp(O(k^3/\varepsilon^2 + kS))$ samples, runs in time $\widetilde{O}(d^2\log(1/\delta))\exp(O(k^3S^2/\varepsilon^2 + kS^3))$, and outputs a ReLU network $\widetilde{F}$ such that $\mathbb{E}[(y - \widetilde{F}(x))^2] \le \varepsilon$ with probability at least $1 - \delta$.

Figure: sample complexity and time complexity [CKM22].

It answers the following questions:

- what is the performance of this algorithm?
- guarantees for algorithms (under what conditions will they succeed, how much data and computation time is needed)

Student Cohort

- Core module for Discrete Maths students
- Optional module for Maths students

Clarifying two misconceptions

- Only prerequisite: Mathematical maturity
- I am well aware of possible overlaps with other modules and some prior background you may already have

Lectures

Lectures (Week 1 to 10)

- Thursday 9:00 - 10:00 (PLT)
- Friday 15:00 - 16:00 (R0.21)
- Friday 16:00 - 17:00 (Woods-Scawen room)

Lectures

Lectures (Week 1 to 10)

- Thursday 9:00 - 10:00 (PLT)
- Friday 15:00 - 16:00 (R0.21)
- Friday 16:00 - 17:00 (Woods-Scawen room)

After each lecture, I will provide links to

- recording of the lecture
- relevant lecture notes/online book chapters in the module webpage

Be proactive yourself

Seminars

9 seminars from Week 2

- Week 2 seminar questions will be posted in the module webpage before Monday morning of Week 2, and so on.
- Try to solve the questions yourself, before attending your seminar sessions.

Seminars

9 seminars from Week 2

- Week 2 seminar questions will be posted in the module webpage before Monday morning of Week 2, and so on.
- Try to solve the questions yourself, before attending your seminar sessions.

Seminar Groups:

- Group 1: Thursdays 10 am - 11 am in S0.17.
- Group 2: Thursdays 11 am - 12 am at OC0.05.
- Group 3: Thursdays 12 pm - 1 pm at H0.03.
- Group 4: Thursdays 1 pm - 2 pm at S0.18.
- Group 5: Fridays 2 pm - 3 pm at S0.10.

Some important information (I)

Grading

- Coursework 1: 10%
- Coursework 2: 10%
- In-person Examination: 80%

About coursework 1: will post on the module webpage.

◦ Every relevant information will be posted in the module webpage.

◦ Contact me via emails. I will respond within 2 days (if I'm not on travel).

Some important information (II)

- you are having problems with tabula ⇒ email DCS.UG.Support@warwick.ac.uk
- Only exception: You wish to change your seminar group (for valid reason). Then write an email to DCS.UG.Support@warwick.ac.uk and **cc me**. In the email, explain your reason, and specify the groups you can join (see module webpage for group numbers).

◦ valid reason: time conflicts with other lectures/seminars

Recall: Induction and recursion

- induction: **proving** some universal statements from a smaller objects
- recursion: applied to **definition** in terms of smaller objects

Example

Let $n \in \mathbb{N}$, if $n \geq 4$, we have $2^n \geq n^2$.

Recall: Induction and recursion

- induction: **proving** some universal statements from a smaller objects
- recursion: applied to **definition** in terms of smaller objects

Example

Let $n \in \mathbb{N}$, if $n \geq 4$, we have $2^n \geq n^2$.

Proof by induction.

Base case, let $n = 4$, we have $2^n = n^2 = 16$.
For the induction step, assume $2^n \geq n^2$, we need to show $2^{n+1} \geq (n+1)^2 = n^2 + 2n + 1$.

$$2^{n+1} = 2 \times 2^n \geq 2n^2 \geq n^2 + 4n \geq n^2 + 2n + 1\,.$$

□

Recursion

Definition (Recursion)

A problem solving technique in which problems are solved by reducing them to **smaller problems** of the same form.

Example (factorial)

$$1! = 1 \tag{1}$$

$$n! = n \cdot (n-1)! \tag{2}$$

$$6! = 6 \cdot 5! = 6 \cdot 5 \cdot 4! = 6 \cdot 5 \cdot 4 \cdot 3! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$$

Example: Frog jumping problem

Problem

How many ways can a frog hop up a twelve-step staircase if the frog can hop either one or two steps on each hop?

Example: Frog jumping problem

Problem

How many ways can a frog hop up a twelve-step staircase if the frog can hop either one or two steps on each hop?

Solution

Denote $f(n)$ *as the number of ways that a frog hops up to the* n*-th stair. Clearly,* $f(1) = 1$ *and* $f(2) = 1 + 1 = 2$*.*
for the n*-th stair, there is only two ways to reach*

- *hop from the* $(n-1)$*-th stair*
- *hop from the* $(n-2)$*-th stair*

$$\Rightarrow f(n) = f(n-1) + f(n-2)\,. \quad \text{with } f(1) = 1, f(2) = 2.$$

This is a Fibonacci sequence.

References I

[0] Sitan Chen, Adam R Klivans, and Raghu Meka, *Learning deep relu networks is fixed-parameter tractable*, 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 2022, pp. 696–707.
(Cited on pages 10 and 11.)

[0] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, International Conference on Learning Representations, 2015.
(Cited on page 7.)