Intro / overview

CPSC 532D: Modern Statistical Learning Theory
6 September 2022
cs.ubc.ca/~dsuth/532D/22w1/
Admin: me (hi!)

- Danica Sutherland - djsutherland.ml - ICICS X563 - she/her
  - “Danica” (North Am. English pronunciation, not authentic Slavic one)
    / “Professor Sutherland” / “Dr. Sutherland” are all fine

- Recent-ish at UBC (January 2021)
  - 6+2 grad students
  - 2019-20: TTI-Chicago (baby faculty / super-postdoc)
  - 2016-19: University College London (postdoc)
  - 2011-16: Carnegie Mellon (grad school)
Admin: me (hi!)

• My research so far:
  • kernels: especially “deep kernels” and (recently) neural tangents (~80% of work)
  • learning and testing on probability distributions (~60%)
  • various other representation learning stuff (~20%)
  • generative models and evaluation (~20%)
  • statistical learning theory:
    • theorems about kernel / probability distribution stuff (~40% of work)
    • limits of uniform convergence: 4 papers
    • limits of invariant risk minimization: 1 paper

• Taught this course once before (then called 532S, last term)
Admin: course cap

• 30 person cap (but the room might only have 25 seats?)

• As of Wednesday night, at ~37 registered/waitlisted/otherwise interested

• This should be fine! But if you’re not officially registered, have a backup plan

• If you want to audit, email or private Piazza post

• Will probably teach this same course again next year
  • Almost certainly within 2 years – *might* do something different next year
Admin: course format

• Classes are lecture-based
  • Going to do mix of slides and on the board
  • First time teaching on the board, fyi

• Grading:
  • 70% assignments (~6 through the term, lowest dropped)
    • One “special” assignment to read + poke at a paper; cannot be dropped
  • 30% final exam
    • In person during finals period, handwritten
Admin: assignments

• First assignment will be up by tomorrow night on the course site
• “Math problems” + some very light coding to poke at the math
• Hand-in on Gradescope, will be described on Piazza
• **Due Monday the 19th (12 days) at noon**
• Do it in LaTeX; template available to fill in if that helps, or go from scratch

• Most you should be able to do already
• Rest will be covered by lecture 2 (Monday)

• You can do with a partner
• If you’re not yet registered but want in, do the assignment
• If you’re auditing/sitting in: encourage you to do it but *don’t submit*, please
Admin: places to look

- Course website: cs.ubc.ca/~dsuth/532D/
  - Slides, schedule, homeworks, etc

- Piazza: linked from Canvas and course site
  - Announcements will go here, so be sure to sign up
  - Also usual discussion, etc
  - Prefer you post anything course-related here, but email is okay if it’s easier for whatever reason

- Canvas: canvas.ubc.ca/courses/101911
  - Not much will go here, but links to Piazza and (soon) Gradescope
Admin: books

- Going to try for fully self-contained lecture notes…this may not happen
- Largely based on material from three (free!) books – chapter refs as we go
(pause)
“If you’re analyzing data and proving theorems about it in [ESB], that’s statistics.
If you do it in [ICICS], that’s machine learning.”

– Larry Wasserman
(who said it with Baker and Gates, CMU’s equivalents)
Machine learning

• “A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$.”

• “A checkers learning problem:
  • Task $T$: playing checkers
  • Performance measure $P$: percent of games won against opponents
  • Training experience $E$: playing practice games against itself”

• “A handwriting recognition learning algorithm:
  • Task $T$: recognizing and classifying handwritten words within images
  • Performance measure $P$: percent of words correctly classified
  • Training experience $E$: a database of handwritten words with given classifications”

• “a database system that allows users to update data entries would fit our definition of a learning system: it improves its performance at answering database queries, based on the experience gained from database updates”
Rats learn to associate food types ↔ toxin but don’t learn food ↔ shock
lights ↔ shock

“‘Superstition’ in the pigeon” - Skinner

https://www.youtube.com/watch?v=Qv4H81gEGDQ
...why?

• Apparently, different *hypothesis classes*

• Rats maybe have built-in that food ↔ gastric, light ↔ shock, not others
  • Helps when it’s right
  • Makes it impossible to learn that a light is a “poison detector”

• Pigeons, maybe, don’t have these built-ins
  • Presumably could learn that flapping wings → food
  • But can cause *overfitting* in other situations
• One main goal of statistical learning theory: be able to understand these kinds of questions
  • What determines when we can learn?
  • What resources (data in different forms, computation) do we need to do it?

• We’ll strive to do it formally and quantitatively:
  • What kinds of assumptions do we need on the data, the learner, …?
  • Aim for finite-sample, high-probability guarantees
  • How are different analysis techniques related? What limitations are there?
Well-studied foundations…

**Theorem 6.7 (The Fundamental Theorem of Statistical Learning)** \(\text{Let } \mathcal{H} \text{ be a hypothesis class of functions from a domain } \mathcal{X} \text{ to } \{0, 1\} \text{ and let the loss function be the } 0 - 1 \text{ loss. Then, the following are equivalent:}

1. \( \mathcal{H} \) has the uniform convergence property.
2. Any ERM rule is a successful agnostic PAC learner for \( \mathcal{H} \).
3. \( \mathcal{H} \) is agnostic PAC learnable.
4. \( \mathcal{H} \) is PAC learnable.
5. Any ERM rule is a successful PAC learner for \( \mathcal{H} \).
6. \( \mathcal{H} \) has a finite VC-dimension.

which we’re going to learn first!
…but they don’t explain modern ML

Training error consistently decreases with model complexity, typically dropping to zero if we increase the model complexity enough. However, a model with zero training error is overfit to the training data and will typically generalize poorly.

Table 1: The training and test accuracy (in percentage) of various models on the CIFAR10 dataset.

<table>
<thead>
<tr>
<th>model</th>
<th># params</th>
<th>random crop</th>
<th>weight decay</th>
<th>train accuracy</th>
<th>test accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>1,649,402</td>
<td>yes</td>
<td>yes</td>
<td>100.0</td>
<td>89.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>yes</td>
<td>no</td>
<td>100.0</td>
<td>89.31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>no</td>
<td>yes</td>
<td>100.0</td>
<td>86.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td>no</td>
<td>no</td>
<td>100.0</td>
<td>85.75</td>
</tr>
</tbody>
</table>

To put this in concrete terms, on MNIST, having even 72 hidden units in a fully connected first layer yields vacuous PAC bounds.
Uniform convergence may be unable to explain generalization in deep learning

Benign Overfitting in Linear Regression

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

On the robustness of minimum norm interpolators and regularized empirical risk minimizers

Failures of model-dependent generalization bounds for least-norm interpolation

Optimistic Rates: A Unifying Theory for Interpolation Learning and Regularization in Linear Regression
Other important questions

• Do we get “implicit regularization” from optimization algorithms?
• When does (S)GD find a good minimum for neural networks?
  • Analysis via neural tangent kernels
• What can deep networks learn that kernels can’t?
• When do GPs learn the right posterior distribution?
• When can we learn online? When can we learn privately?
  • …and is it foreshadowing that these are on the same bullet?
• Does actively selecting points to be labeled help?
• When does self-supervised learning work?
• Does everything break if training and test aren’t exactly the same distribution?
• Have vision architectures/algorithms overfit to the CIFAR / ImageNet test set?