CSCI699: Machine Learning Theory Fall 2023

Lecture 5: Complexity Review

Instructor: Vatsal Sharan

1 Computational Complexity Review

In the next few lectures, we will explore why some learning problems are likely impossible to solve computationally efficiently. We will use tools from the rich field of computational complexity to argue this. The first definition we need is for the complexity class NP.

Definition 1 (NP)**.** *A decision problem C is in NP if there exists a polynomial-time algorithm A such that for every instance x of C,*

- *If x evaluates to "yes", then $\exists y$, $|y| \leq \text{poly}(|x|), A(x, y) = 1$*
- *If x evaluates to "no", then $\forall y$, $|y| \leq \text{poly}(|x|), A(x, y) = 0$*

The intuition for the definition is that A is some verifier who can verify solutions to the decision problem. y is a certificate or witness which A can check to verify the solution to the problem.

As an example, consider 3SAT on d variables

$$\begin{aligned} 3\text{SAT}_\text{d} =& (x_1 \vee \bar{x}_2 \vee x_5) \\ & \wedge (x_5 \vee x_6 \vee \bar{x}_7) \\ & \vdots \\ & \wedge (x_{d-2} \vee \bar{x}_{d-1} \vee x_d). \end{aligned}$$

Here, the certificate y is just a satisfying assignment. The verifier just checks if the certificate/assignment is valid, note that it is easy to check this (it can be done in running time which is linear in the number of variables). To complete the proof that 3SAT_d is in NP, note that there always exists a certificate if the instance is satisfiable, and there does not exist any certificate if the problem is not satisfiable.

Our next complexity class is a class of randomized polynomial time algorithms.

Definition 2 (RP)**.** *A decision problem C is in RP if there exists a randomized polynomial-time algorithm A such that for every instance x of C,*

- *If x evaluates to "yes", A outputs "yes" w.p. $\geq \frac{2}{3}$*
- *If x evaluates to "no", A outputs "no" w.p. 1.*

The intuition for this definition is that C is in RP if there exists a polynomial-time algorithm with *one-sided error*. In particular, for the case of 3SAT, if the 3SAT instance is satisfiable, then the algorithm will output TRUE w.p. $\geq \frac{2}{3}$. If the 3SAT instance is not satisfiable, then the algorithm will always output FALSE.

It is widely believed that RP $\neq$ NP. This is a slightly stronger version of the famous P $\neq$ NP conjecture. The P $\neq$ NP conjecture states that polynomial time algorithms cannot solve all problems in NP. RP $\neq$ NP claims that even randomized polynomial time algorithms cannot solve all problems in NP. We believe that RP $\neq$ NP, because there is strong evidence that RP = P.

Our final definition is of NP-Completeness.

Definition 3 (NP-Completeness)**.** *A decision problem C is in NP-Complete if,*

- *C is in NP*
- *Every decision problem C' in NP can be reduced to C in polynomial time.*

Intuitively, if a problem is NP-Complete that solving that problem in polynomial time is sufficient for solving all problems in NP in polynomial time.

2 Further reading

Lots of resources are available online to understand this material. For a quick overview, you can watch this lecture by Erik Demaine: https://www.youtube.com/watch?v=eHZifpgyH_4. The first 30 minutes cover most of the ground. For the remaining time he does polynomial time reductions to show various problems are NP hard. If you don't remember how those kind of reduction arguments go, try to watch more of the lecture. You'll also learn why Super Mario Bros. is NP hard.

For a slightly more detailed discussion, I highly recommend reading Avi Wigderson's beautifully written book [1]. Chapter 3 has most of the complexity basics we need. Randomized algorithms appear in Chapter 7.

References

[1] Avi Wigderson. *Mathematics and computation: A theory revolutionizing technology and science.* Princeton University Press, 2019.