# 10-418 / 10-618 Machine Learning for Structured Data

Machine Learning Department
School of Computer Science
Carnegie Mellon University

# Bayesian Networks

Matt Gormley
Lecture 6
Sep. 16, 2019

# Reminders

- **Homework 1: DAgger for seq2seq**
  - Out: Thu, Sep. 12
  - Due: Thu, Sep. 26 at 11:59pm

# APPLICATIONS OF SEQ2SEQ

# seq2seq for MT

## Basic Architecture:

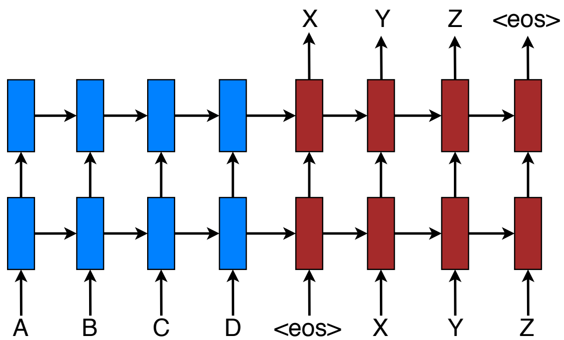



Figure 1: **Neural machine translation** – a stacking recurrent architecture for translating a source sequence `A B C D` into a target sequence `X Y Z`. Here, `<eos>` marks the end of a sentence.

## Results from Sutskever et al. (2014)

| Method | test BLEU score (ntst14) |
|---|---|
| Bahdanau et al. [2] | 28.45 |
| Baseline System [29] | 33.30 |
| Single forward LSTM, beam size 12 | 26.17 |
| Single reversed LSTM, beam size 12 | 30.59 |
| Ensemble of 5 reversed LSTMs, beam size 1 | 33.00 |
| Ensemble of 2 reversed LSTMs, beam size 12 | 33.27 |
| Ensemble of 5 reversed LSTMs, beam size 2 | 34.50 |
| Ensemble of 5 reversed LSTMs, beam size 12 | **34.81** |

Table: performance on WMT'14 English to French test set

## Visualization from Sutskever et al. (2014)

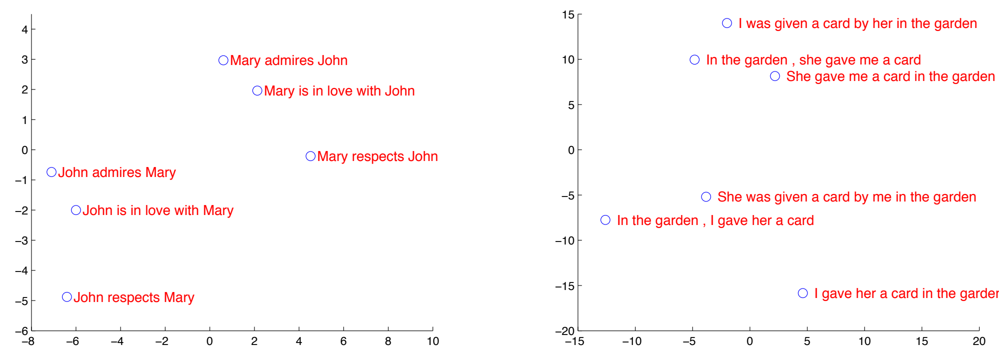



Figure 2: The figure shows a 2-dimensional PCA projection of the LSTM hidden states that are obtained after processing the phrases in the figures. The phrases are clustered by meaning, which in these examples is primarily a function of word order, which would be difficult to capture with a bag-of-words model. Notice that both clusters have similar internal structure.

Figure from Luong et al. (2015)

# seq2seq for ASR

**Listen Attend and Spell**

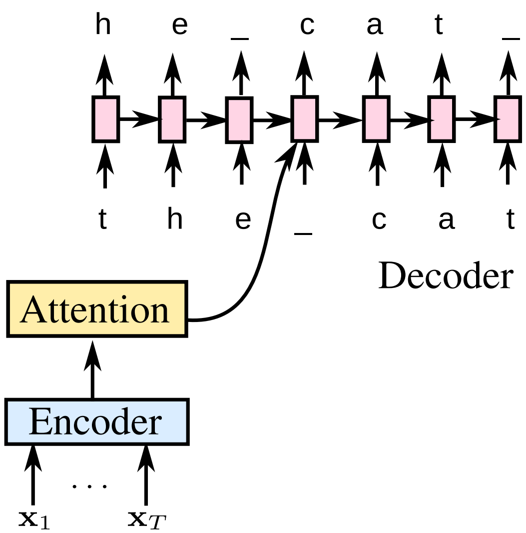



Figure 1: *LAS model.*

$$
\begin{aligned}
\mathbf{h} &= \mathrm{Listen}(\mathbf{x}) \\
P(y_i|\mathbf{x}, y_{<i}) &= \mathrm{AttendAndSpell}(y_{<i}, \mathbf{h})
\end{aligned}
$$



Figure from Irie et al. (2019)

# seq2seq for ASR

## Listen Attend and Spell

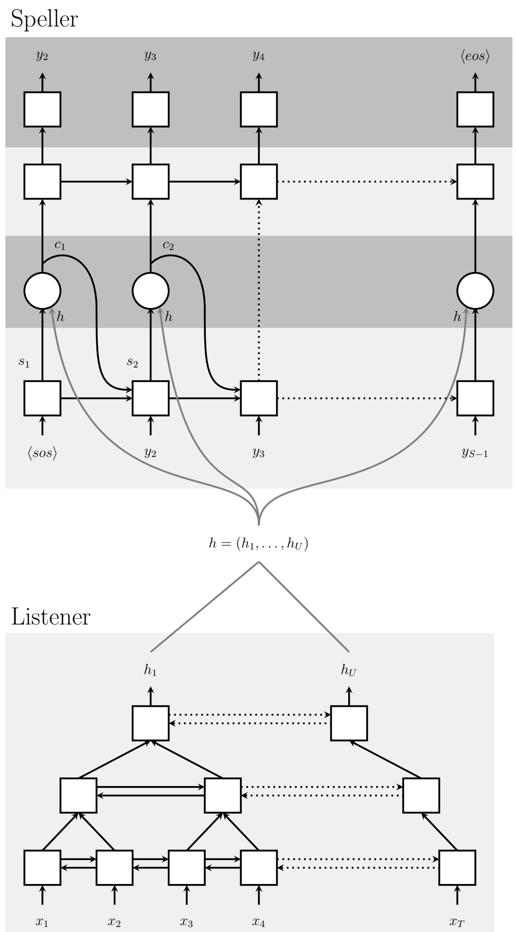



**Fig. 1**: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence $\mathbf{x}$ into high level features $\mathbf{h}$, the speller is an attention-based decoder generating the $\mathbf{y}$ characters from $\mathbf{h}$.

## Results from Park et al. (2019)

Table 3: *LibriSpeech 960h WERs (%).*

| **Method** | **No LM** | | **With LM** | |
|---|---|---|---|---|
| | **clean** | **other** | **clean** | **other** |
| **HMM** | | | | |
| Panayotov et al., (2015) [19] | | | 5.51 | 13.97 |
| Povey et al., (2016) [29] | | | 4.28 | |
| Han et al., (2017) [30] | | | 3.51 | 8.58 |
| Yang et al. (2018) [31] | | | 2.97 | 7.50 |
| **CTC/ASG** | | | | |
| Collobert et al., (2016) [32] | 7.2 | | | |
| Liptchinsky et al., (2017) [33] | 6.7 | 20.8 | 4.8 | 14.5 |
| Zhou et al., (2018) [34] | | | 5.42 | 14.70 |
| Zeghidour et al., (2018) [35] | | | 3.44 | 11.24 |
| Li et al., (2019) [36] | 3.86 | 11.95 | 2.95 | 8.79 |
| **LAS** | | | | |
| Zeyer et al., (2018) [23] | 4.87 | 15.39 | 3.82 | 12.76 |
| Zeyer et al., (2018) [37] | 4.70 | 15.20 | | |
| Irie et al., (2019) [24] | 4.7 | 13.4 | 3.6 | 10.3 |
| Sabour et al., (2019) [38] | 4.5 | 13.3 | | |

Figure from Chan et al. (2016)

# seq2seq for ASR

## Listen Attend and Spell

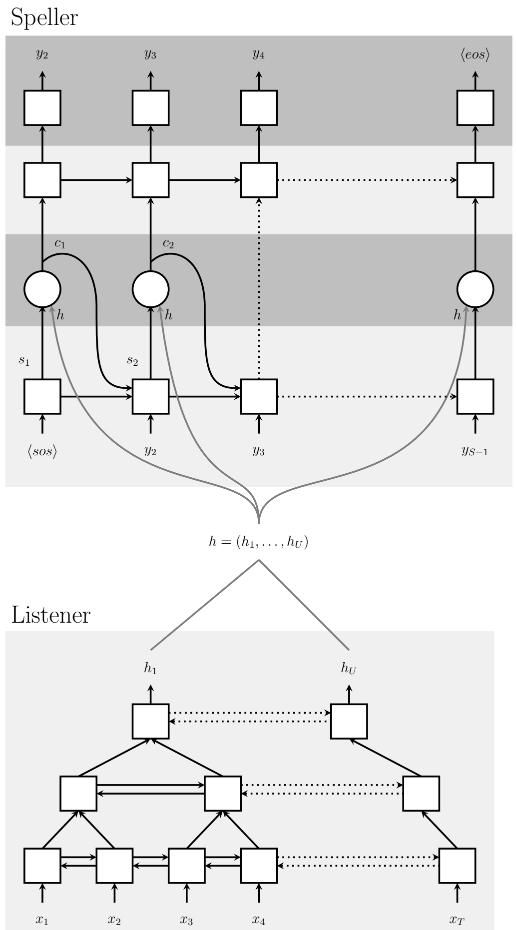



**Fig. 1**: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence $\mathbf{x}$ into high level features $\mathbf{h}$, the speller is an attention-based decoder generating the $\mathbf{y}$ characters from $\mathbf{h}$.

## Results from Park et al. (2019)

Table 3: *LibriSpeech 960h WERs (%).*

| Method | No LM | | With LM | |
|---|---|---|---|---|
| | **clean** | **other** | **clean** | **other** |
| **HMM** | | | | |
| Panayotov et al., (2015) [19] | | | 5.51 | 13.97 |
| Povey et al., (2016) [29] | | | 4.28 | |
| Han et al., (2017) [30] | | | 3.51 | 8.58 |
| Yang et al. (2018) [31] | | | 2.97 | 7.50 |
| **CTC/ASG** | | | | |
| Collobert et al., (2016) [32] | 7.2 | | | |
| Liptchinsky et al., (2017) [33] | 6.7 | 20.8 | 4.8 | 14.5 |
| Zhou et al., (2018) [34] | | | 5.42 | 14.70 |
| Zeghidour et al., (2018) [35] | | | 3.44 | 11.24 |
| Li et al., (2019) [36] | 3.86 | 11.95 | 2.95 | 8.79 |
| **LAS** | | | | |
| Zeyer et al., (2018) [23] | 4.87 | 15.39 | 3.82 | 12.76 |
| Zeyer et al., (2018) [37] | 4.70 | 15.20 | | |
| Irie et al., (2019) [24] | 4.7 | 13.4 | 3.6 | 10.3 |
| Sabour et al., (2019) [38] | 4.5 | 13.3 | | |
| **Our Work** | | | | |
| LAS | 4.1 | 12.5 | 3.2 | 9.8 |
| LAS + SpecAugment | **2.8** | **6.8** | **2.5** | **5.8** |

Park et al. (2019) used the **LAS model** from prior work, and introduced a **data augmentation** method that gave state-of-the-art performance on LibriSpeech 960h and Swichboard 300h tasks



Figure from Chan et al. (2016)

# Image Captioning

p(English | French)

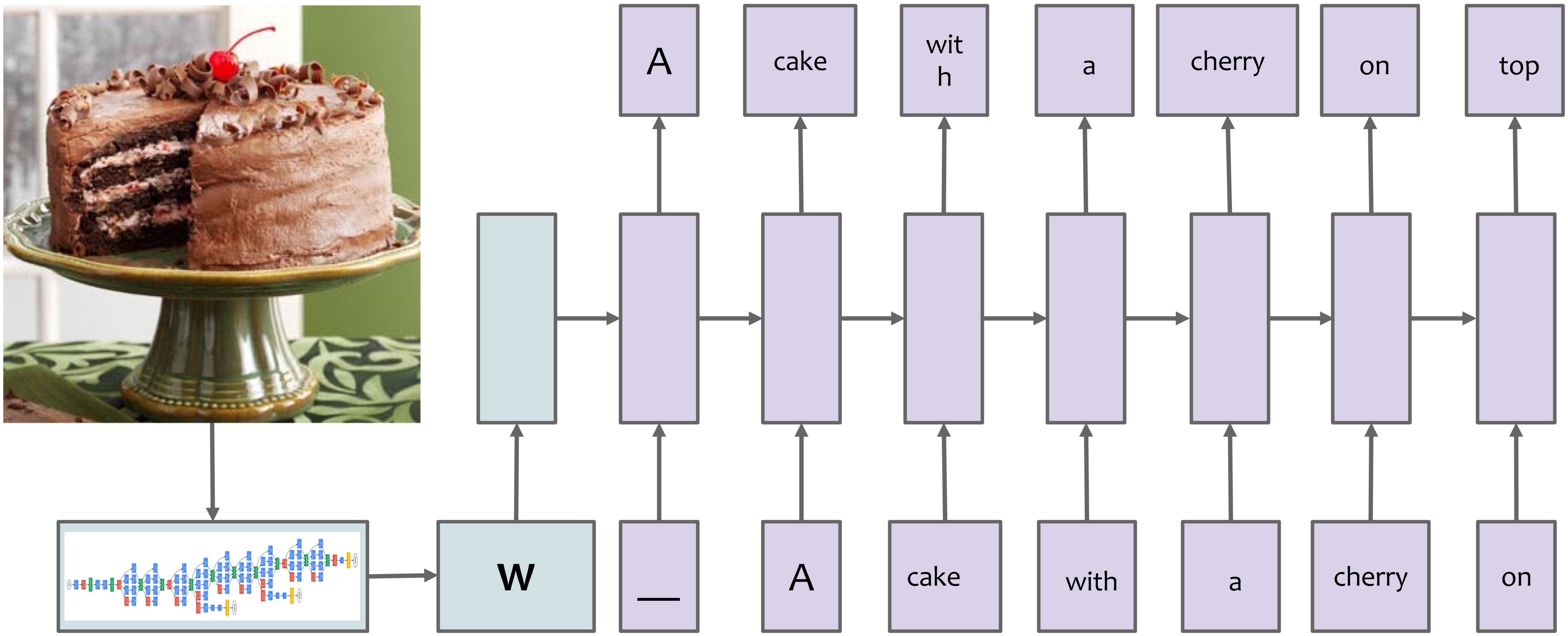
$\downarrow$

p(English | Image)


1. Vinyals, O., et al. "Show and Tell: A Neural Image Caption Generator." *CVPR* (2015).
2. Mao, J., et al. "Deep captioning with multimodal recurrent neural networks (m-rnn)." *ICLR (2015).*
3. Karpathy, A., Li, F., "Deep visual-semantic alignments for generating image descriptions." *CVPR (2015)*.


Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning




$$\theta^\star = \arg\max_\theta p(S|I)$$

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning


*Human: A close up of two bananas with bottles in the background.*

*BestModel: A bunch of bananas and a bottle of wine.*

*InitialModel: A close up of a plate of food on a table.*

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning

*Human: A woman holding up a yellow banana to her face.*

*BestModel: A woman holding a banana up to her face.*

*InitialModel: A close up of a person eating a hot dog.*

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning

*Human: A man outside cooking with a sub in his hand.*

*BestModel: A man is holding a sandwich in his hand.*

*InitialModel: A man cutting a cake with a knife.*

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning


*Human: Someone is using a small grill to melt his sandwich.*

*BestModel: A person is cooking some food on a grill.*

*InitialModel: A pizza sitting on top of a white plate.*

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Image Captioning

*Human: A blue , yellow and red train travels across the tracks near a depot.*

*BestModel: A blue and yellow train traveling down train tracks.*

*InitialModel: A train that is sitting on the tracks.*

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)

# Learning Objectives

## Sequence to Sequence Models

*You should be able to…*

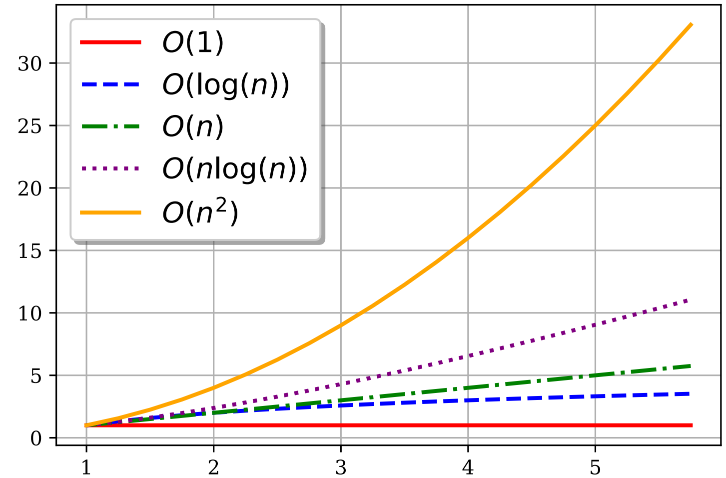
1. Explain the difference between RNNs, RNNLMs, encode-decoder models, and seq2seq models
2. Implement a basic seq2seq model
3. Employ learning to search algorithms to train RNNs (and related models)

# COMPUTATIONAL COMPLEXITY

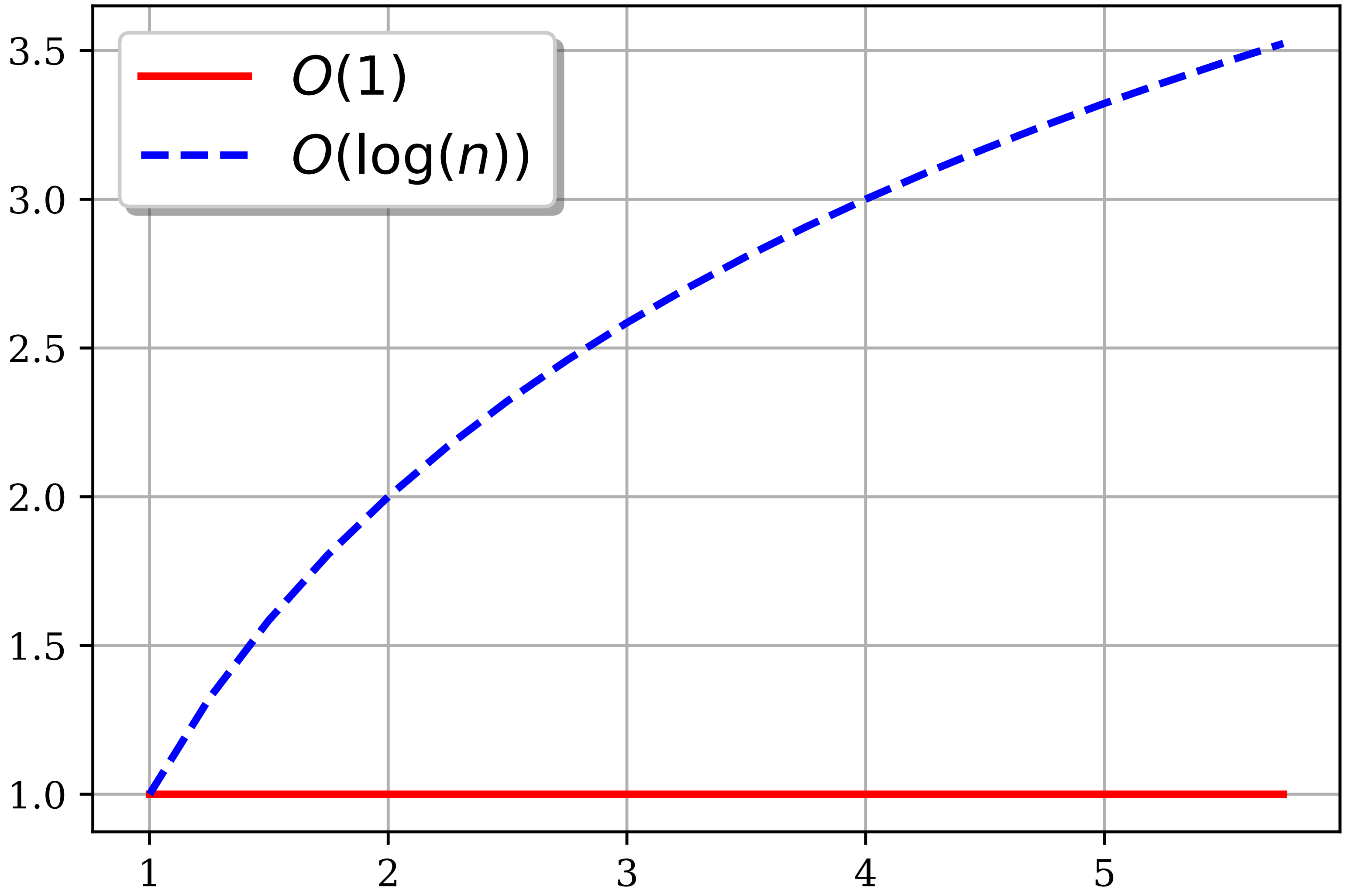
# Analysis of Algorithms

Key Questions:

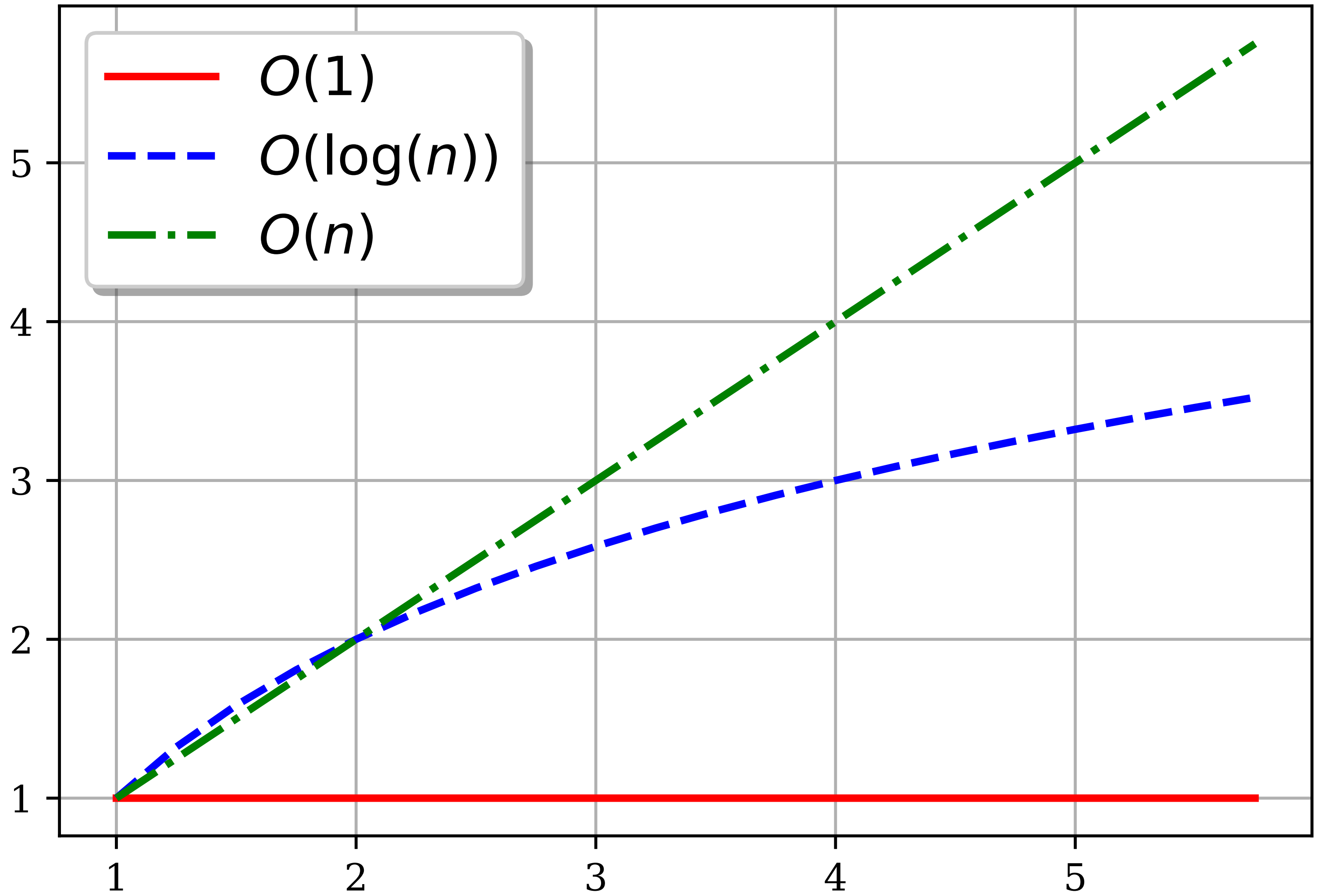
1. Given a single algorithm, will it complete on a given input in a reasonable amount of time/space?
2. Given two algorithms which one is better?



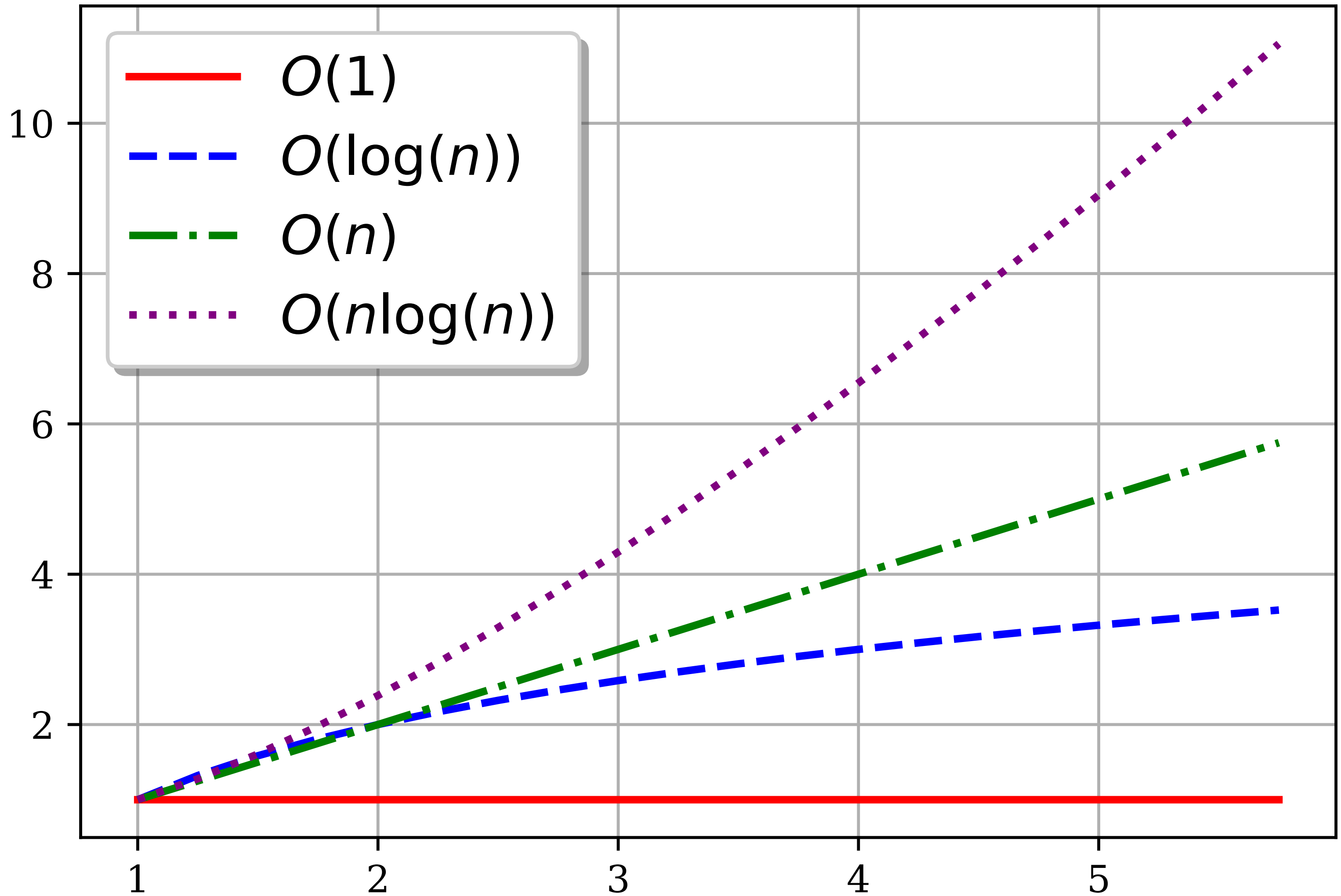
# Comparing Algorithm Runtimes



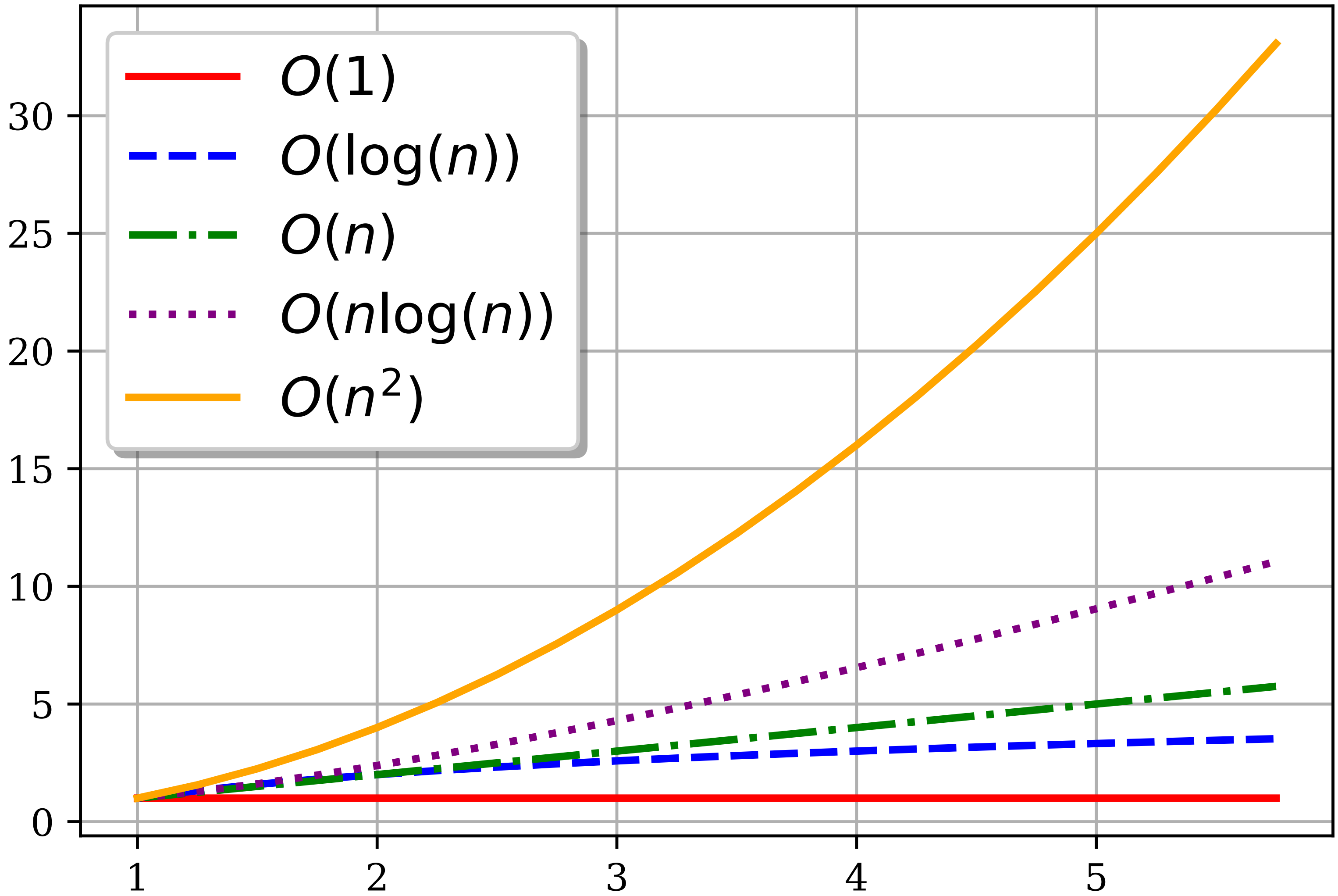
# Comparing Algorithm Runtimes



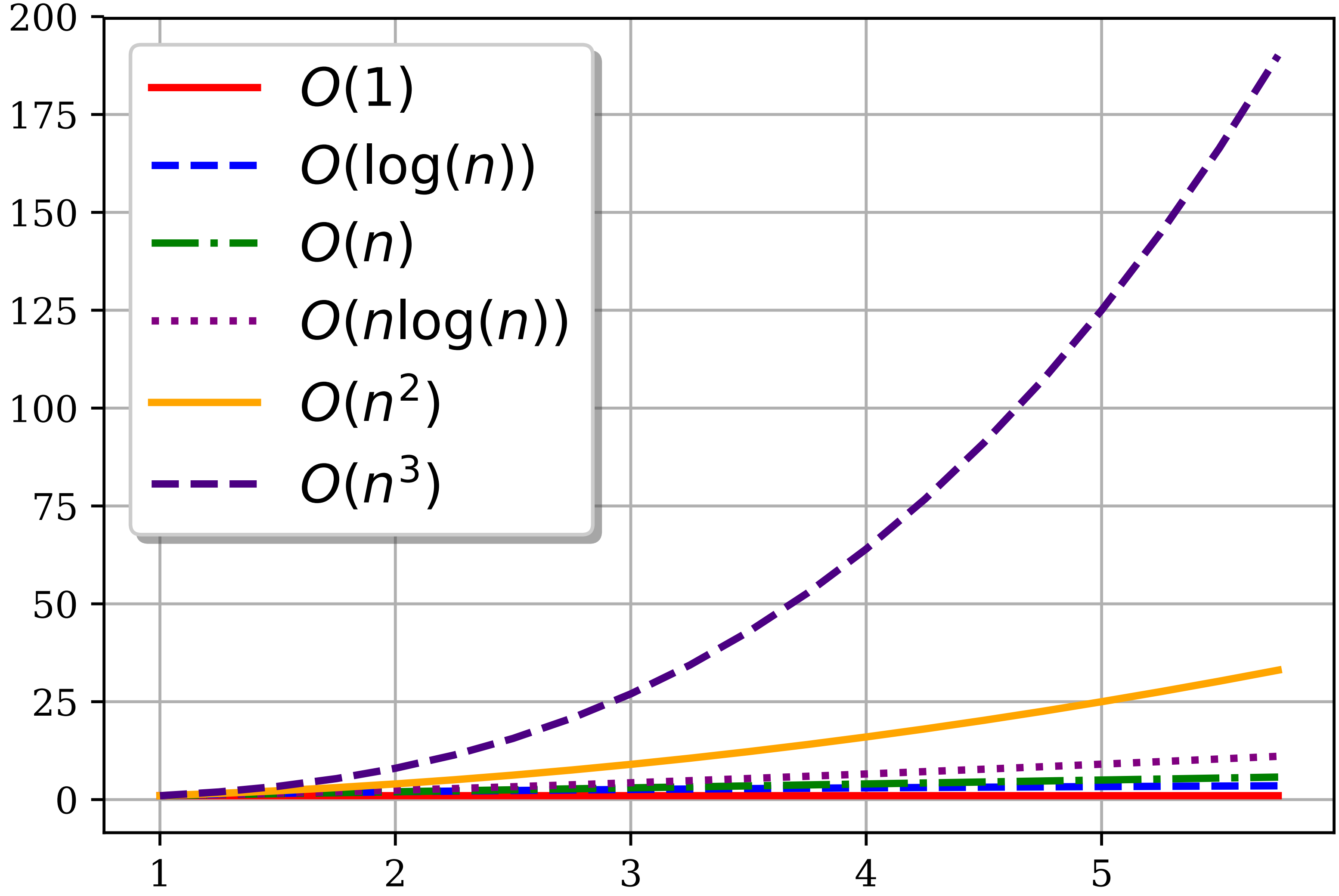
# Comparing Algorithm Runtimes



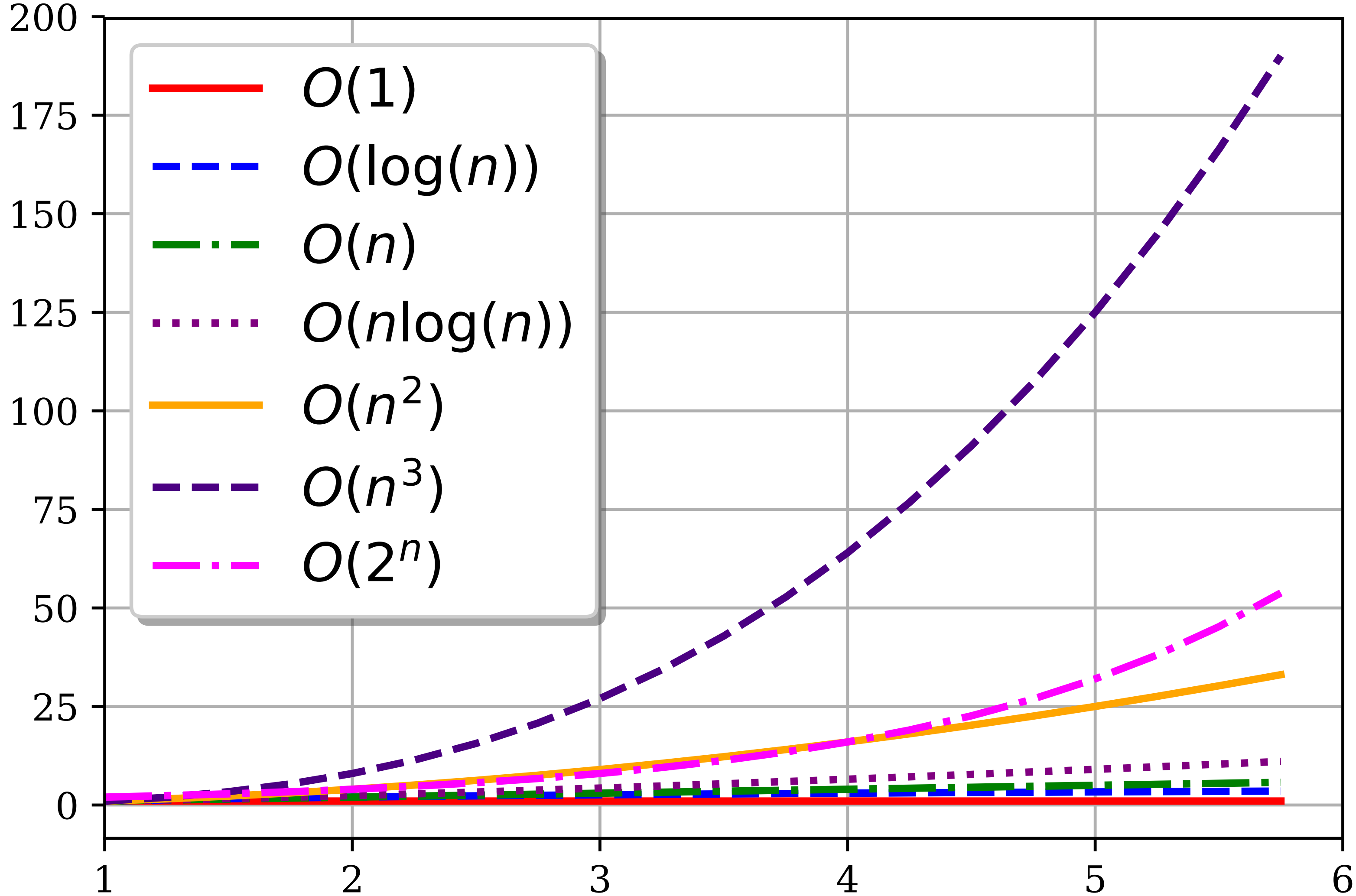
# Comparing Algorithm Runtimes



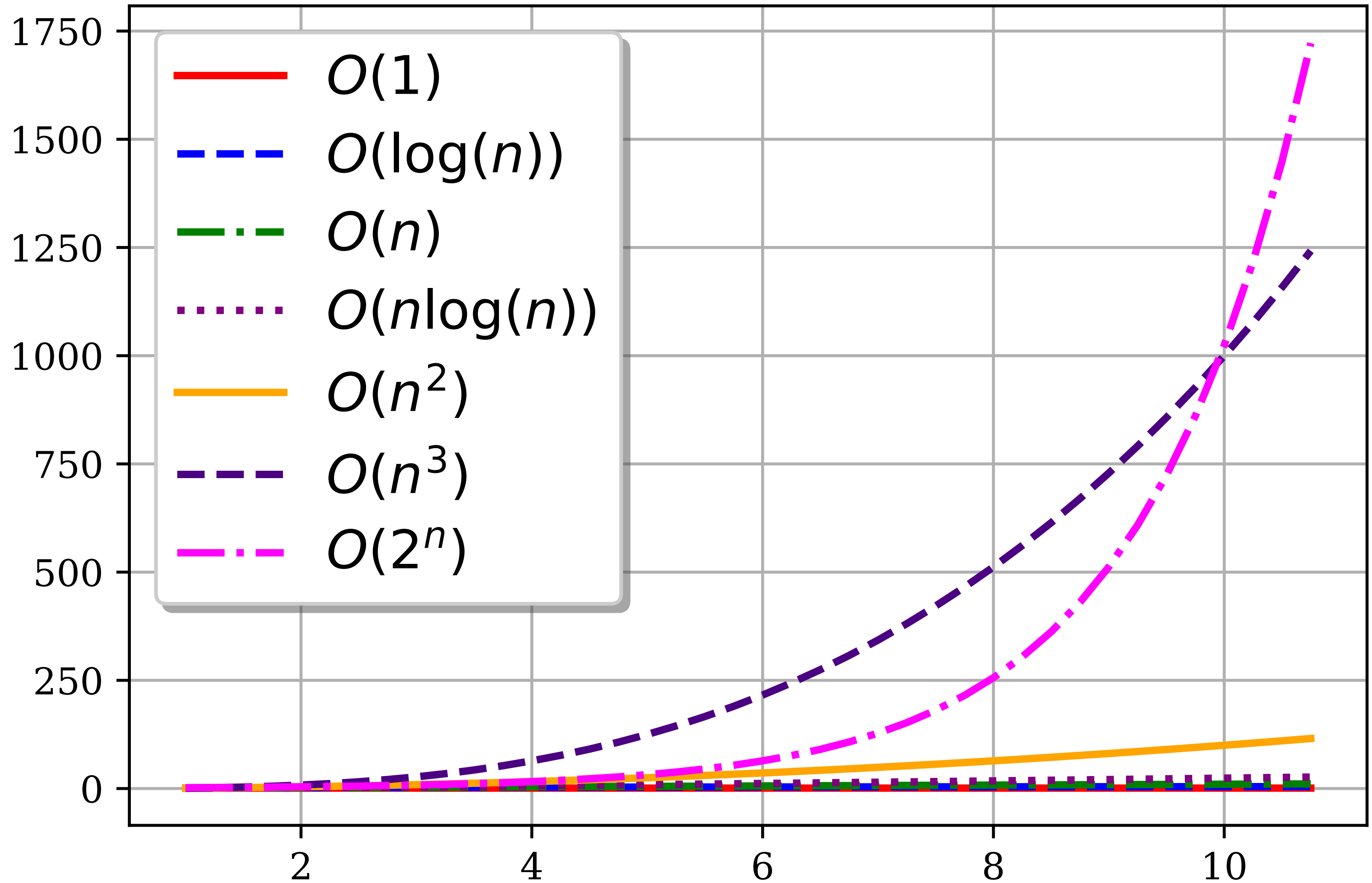
# Comparing Algorithm Runtimes



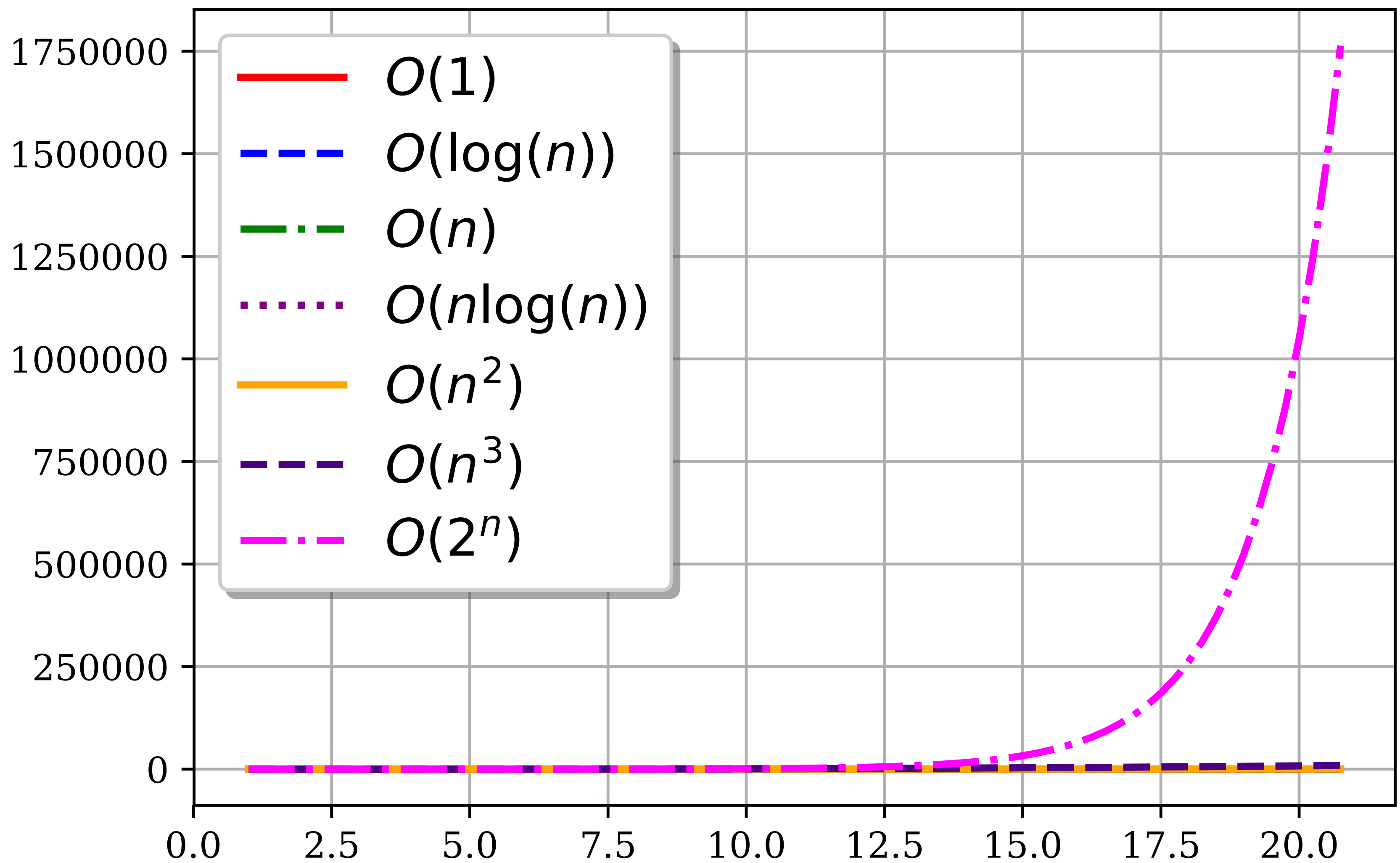
# Comparing Algorithm Runtimes



# Comparing Algorithm Runtimes



# Comparing Algorithm Runtimes



# Comparing Algorithm Runtimes

| Computational Complexity | Name |
|---|---|
| $O(1)$ | constant |
| $O(\log(n))$ | logarithmic |
| $O(n)$ | linear |
| $O(n \log(n))$ | “n log n” |
| $O(n^2)$ | quadratic |
| $O(n^3)$ | cubic |
| $O(2^n)$ | exponential |
| $O(n!)$ | factorial |
| $O(n^n)$ | superexponential |

# Complexity Classes

- An algorithm runs in **polynomial time** if its runtime is a polynomial function of the input size (e.g. $O(n^k)$ for some fixed constant k)
- The **class P** consists of all problems that can be solved in polynomial time

- A problem for which the answer is binary (e.g. yes/no) is called a **decision problem**
- The **class NP** contains all decision problems where 'yes' answers can be verified (proved) in polynomial time
- A problem is **NP-Hard** if given an O(1) oracle to solve it, every problem in NP can be solved in polynomial time (e.g. by reduction)
- A problem is **NP-Complete** if it belongs to both the classes NP and NP-Hard

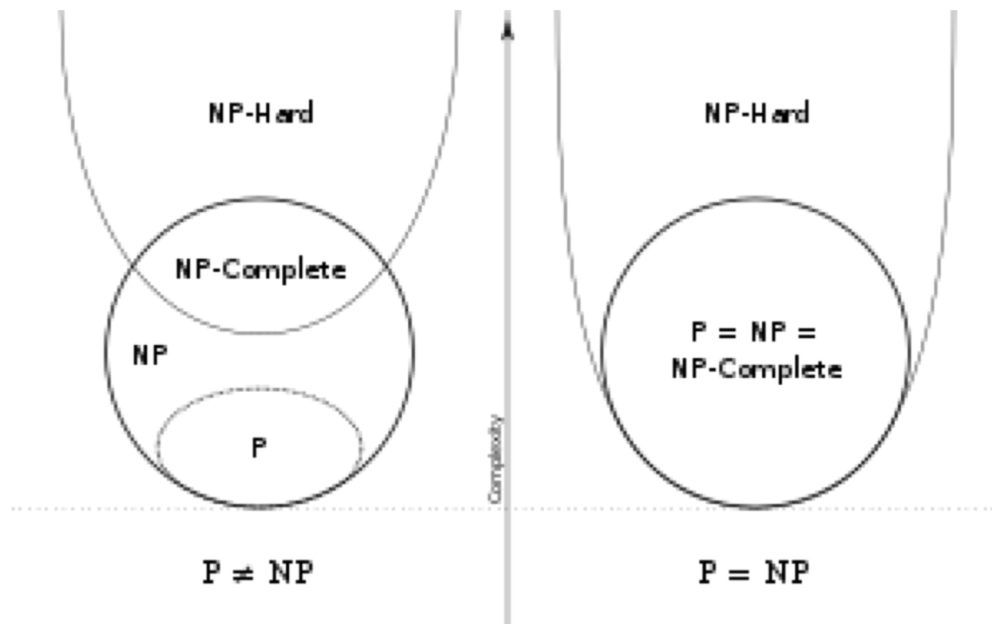



Figure from https://en.wikipedia.org/wiki/NP-completeness

# INFERENCE PROBLEMS

# Inference Problems

*Whiteboard*

- Running example: exponential search space for sequence tagging
- Assumptions leading to a probability distribution
- Intractable problems for arbitrary search spaces:
  - **Problem 1:** Computing the total probability of the hidden states given the observations (Evaluation)
  - **Problem 2:** Computing the marginal probability for a specific observation / timestep (Marginals)
  - **Problem 3:** Finding the most probable assignment to the hidden states (Viterbi decoding)

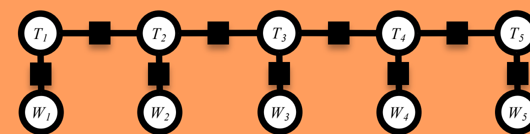
# Exact Inference

## 1. Data

$$\mathcal{D} = \{\boldsymbol{x}^{(n)}\}_{n=1}^{N}$$


## 2. Model

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$




## 3. Objective

$$\ell(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}^{(n)} \mid \boldsymbol{\theta})$$

## 5. Inference

**1. Marginal Inference**

$$p(\boldsymbol{x}_C) = \sum_{\boldsymbol{x}' : \boldsymbol{x}'_C = \boldsymbol{x}_C} p(\boldsymbol{x}' \mid \boldsymbol{\theta})$$

**2. Partition Function**

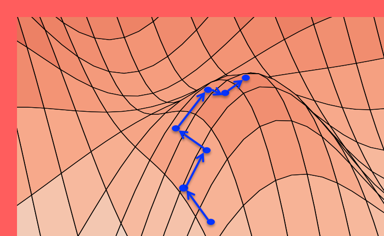
$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$

**3. MAP Inference**

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmax}}\ p(\boldsymbol{x} \mid \boldsymbol{\theta})$$

## 4. Learning

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{D})$$



# 5. Inference

Three Tasks:

**1. Marginal Inference (#P-Hard)**
Compute marginals of variables and cliques

$$p(x_i) = \sum_{\boldsymbol{x}' : x'_i = x_i} p(\boldsymbol{x}' \mid \boldsymbol{\theta}) \qquad \Big| \qquad p(\boldsymbol{x}_C) = \sum_{\boldsymbol{x}' : \boldsymbol{x}'_C = \boldsymbol{x}_C} p(\boldsymbol{x}' \mid \boldsymbol{\theta})$$

**2. Partition Function (#P-Hard)**
Compute the normalization constant

$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$

**3. MAP Inference (NP-Hard)**
Compute variable assignment with highest probability

$$\hat{\boldsymbol{x}} = \operatorname*{argmax}_{\boldsymbol{x}} \; p(\boldsymbol{x} \mid \boldsymbol{\theta})$$

Bayesian Networks

# DIRECTED GRAPHICAL MODELS

# Example: Tornado Alarms

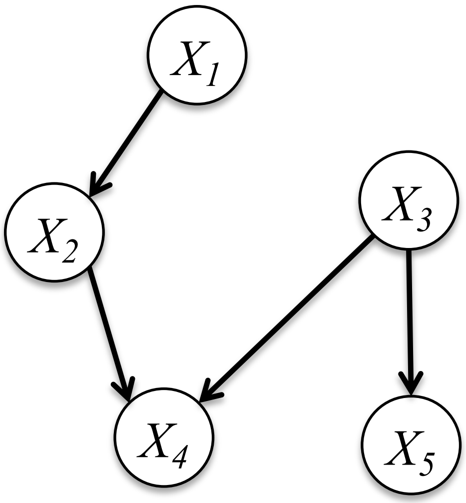
1. Imagine that you work at the 911 call center in Dallas
2. You receive six calls informing you that the Emergency Weather Sirens are going off
3. What do you conclude?

# Directed Graphical Models (Bayes Nets)

*Whiteboard*

- Example: Tornado Alarms
- Writing Joint Distributions
  - Idea #1: Giant Table
  - Idea #2: Rewrite using chain rule
  - Idea #3: Assume full independence
  - Idea #4: Drop variables from RHS of conditionals
- Definition: Bayesian Network

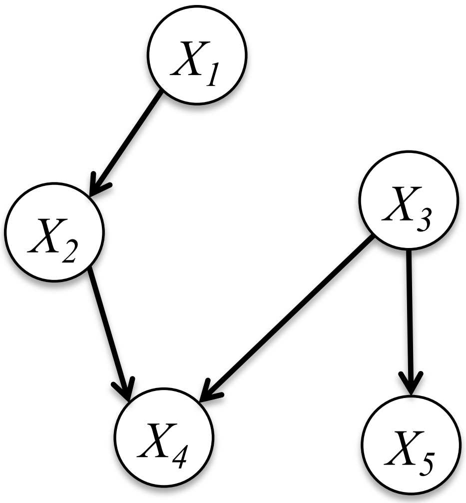
# Bayesian Network




$$p(X_1, X_2, X_3, X_4, X_5) =$$
$$p(X_5|X_3)p(X_4|X_2, X_3)$$
$$p(X_3)p(X_2|X_1)p(X_1)$$

# Bayesian Network

**Definition:**




$$P(X_1 \ldots X_n) = \prod_{i=1}^{n} P(X_i \mid parents(X_i))$$

- A Bayesian Network is a **directed graphical model**
- It consists of a graph **G** and the conditional probabilities **P**
- These two parts full specify the distribution:
  - Qualitative Specification: **G**
  - Quantitative Specification: **P**

# Qualitative Specification

- Where does the qualitative specification come from?
  - Prior knowledge of causal relationships
  - Prior knowledge of modular relationships
  - Assessment from experts
  - Learning from data (i.e. structure learning)
  - We simply prefer a certain architecture (e.g. a layered graph)
  - …

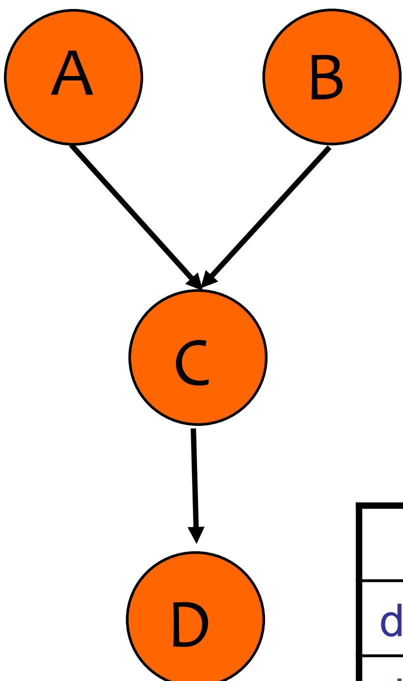
# Quantitative Specification

**Example: Conditional probability tables (CPTs) for discrete random variables**

$$P(a,b,c.d) = P(a)P(b)P(c|a,b)P(d|c)$$

| | |
|---|---|
| $a^0$ | 0.75 |
| $a^1$ | 0.25 |

| | |
|---|---|
| $b^0$ | 0.33 |
| $b^1$ | 0.67 |




| | $a^0b^0$ | $a^0b^1$ | $a^1b^0$ | $a^1b^1$ |
|---|---|---|---|---|
| $c^0$ | 0.45 | 1 | 0.9 | 0.7 |
| $c^1$ | 0.55 | 0 | 0.1 | 0.3 |

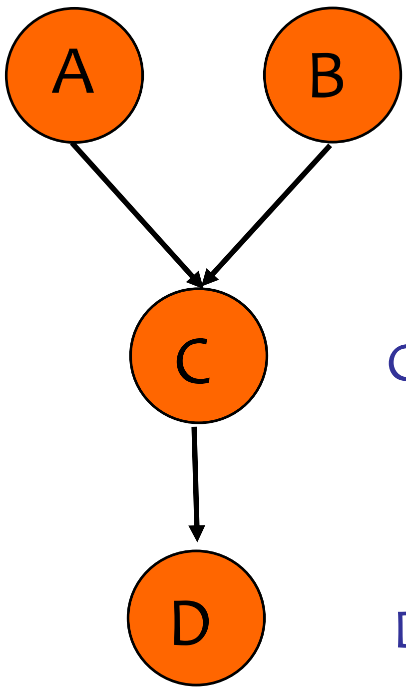
| | $c^0$ | $c^1$ |
|---|---|---|
| $d^0$ | 0.3 | 0.5 |
| $d^1$ | 07 | 0.5 |

# Quantitative Specification

**Example: Conditional probability density functions (CPDs) for continuous random variables**

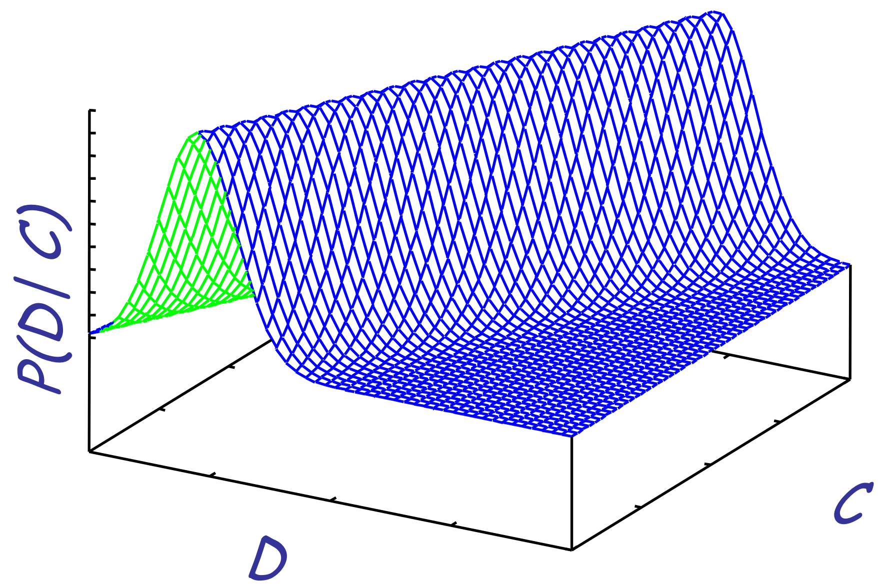
$A \sim N(\mu_a, \Sigma_a)$ $B \sim N(\mu_b, \Sigma_b)$

$$\boxed{\begin{array}{c} P(a,b,c.d) = \\ P(a)P(b)P(c|a,b)P(d|c) \end{array}}$$




$C \sim N(A+B, \Sigma_c)$

$D \sim N(\mu_d + C, \Sigma_d)$



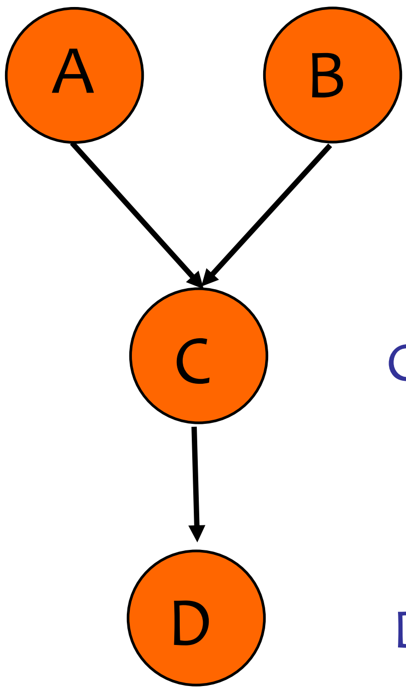
# Quantitative Specification

**Example: Combination of CPTs and CPDs**
**for a mix of discrete and continuous variables**

| | |
|---|---|
| $a^0$ | 0.75 |
| $a^1$ | 0.25 |

| | |
|---|---|
| $b^0$ | 0.33 |
| $b^1$ | 0.67 |

$$P(a,b,c.d) = P(a)P(b)P(c|a,b)P(d|c)$$




$C \sim N(A+B, \Sigma_c)$

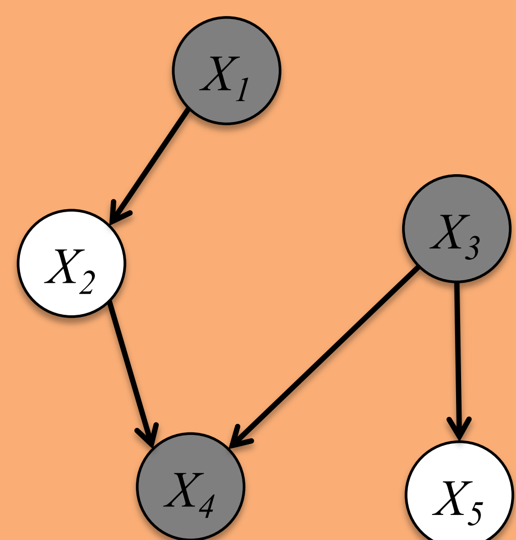
$D \sim N(\mu_d + C, \Sigma_d)$

# Observed Variables

- In a graphical model, **shaded nodes** are "**observed**", i.e. their values are given

**Example:**

$$P(X_2, X_5 \mid X_1 = 0, X_3 = 1, X_4 = 1)$$



# Familiar Models as Bayesian Networks

**Question:**

Match the model name to the corresponding Bayesian Network

1. Logistic Regression
2. Linear Regression
3. Bernoulli Naïve Bayes
4. Gaussian Naïve Bayes
5. 1D Gaussian

**Answer:**