Course Logistics and Introduction to Probabilistic Machine Learning

CS772A: Probabilistic Machine Learning

Piyush Rai

Course Logistics

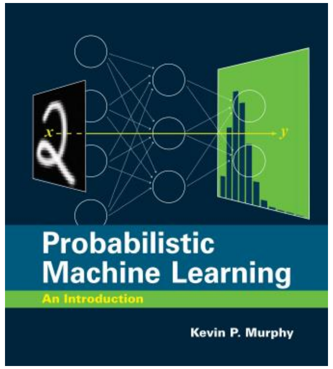
- Course Name: Probabilistic Machine Learning – **CS772A**
- 2 classes each week
  - Mon/Wed 18:00-19:15
  - Venue: RM-101
- Attendance policy: Minimum 60% (biometric attendance)
- All material (readings etc) will be posted on course webpage
- URL: https://www.cse.iitk.ac.in/users/piyush/courses/pml_spring26/pml.html
- Q/A and announcements on Piazza. Please sign up
  - URL: https://piazza.com/iitk.ac.in/secondsemester2026/cs772
  - If need to contact me by email (piyush@cse.iitk.ac.in), prefix subject line with "CS772"
- Unofficial auditors are welcome

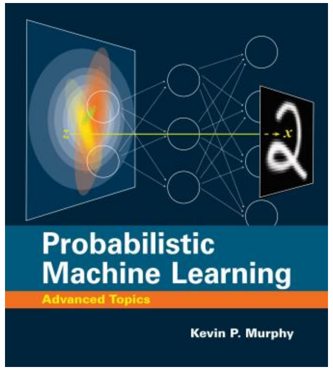
Workload and Grading Policy

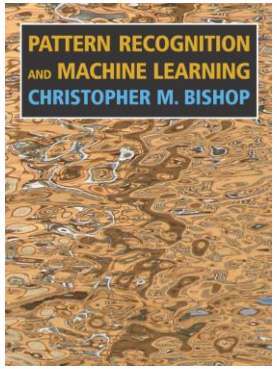
- Evaluation entirely class-room based
- 2 quizzes: 20%
  - In class, closed-book (no cheat-sheet), 45 minutes duration
- 2 homework assignments: 30%
- Mid-sem exam: 20% (date as per DOAA). Closed-book, a cheat-sheet allowed
- End-sem exam: 30% (date as per DOAA). Closed-book, a cheat-sheet allowed
- Practice problems and sample codes will be provided regularly
- Proration: If you miss any quiz/mid-sem, we can prorate it using end-sem marks
  - Proration only allowed on limited grounds (e.g., health related)

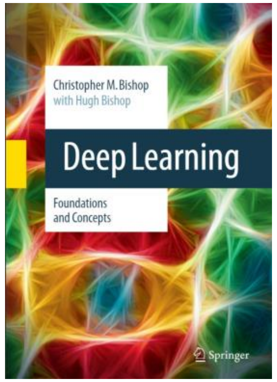
Textbooks and Readings

- Some books that you may use as reference (freely available online)
  - Kevin P. Murphy, Probabilistic Machine Learning: An Introduction (PML-1), The MIT Press, 2022.
  - Kevin P. Murphy, Probabilistic Machine Learning: Advanced Topics (PML-2), The MIT Press, 2022.
  - Chris Bishop, Pattern Recognition and Machine Learning (PRML), Springer, 2007.
  - Chris Bishop and Hugh Bishop, Deep Learning: Foundations and Concepts (DLFC), Springer, 2023.









- Follow the suggested readings for each lecture (may also include some portions from these books), rather than trying to read these books in a linear fashion

Probabilistic Machine Learning

- Machine Learning primarily deals with
  - Predicting output $\boldsymbol{y}_*$ for new (test) inputs $\boldsymbol{x}_*$ given training data $(\boldsymbol{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$
  - Generating new (synthetic) data given some training data $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$
- Probabilistic ML gives a natural way to solve both these tasks (with some advantages)
- Prediction: Learning the predictive distribution

$$p(y_* | x_*, \boldsymbol{X}, \boldsymbol{y})$$

Using this, we can not only get the mean but also the variance (uncertainty) of the predicted output $\boldsymbol{y}_*$

PML is about estimating these distributions accurately and efficiently

Estimating them exactly is hard in general but we can use approximations

- Generation: Learning a generative model of data

$$p(\boldsymbol{x}_* | \boldsymbol{X})$$

Can “sample” (simulate) from this distribution to generate new data

Both are conditional distributions

- At its core, both problems require estimating the underlying distribution of data

Probabilistic Machine Learning

- With a probabilistic approach to ML, we can also easily incorporate “domain knowledge”

- Can specify our assumptions about data using suitable probability distributions over inputs/outputs, usually in the forms

Probability distribution of the output as a function of input

$$p(y_n|x_n,\theta)$$

Unknown parameters of this distribution

$$p(x_n|y_n,\theta)$$

Distribution of the input conditioned on its “label/output”

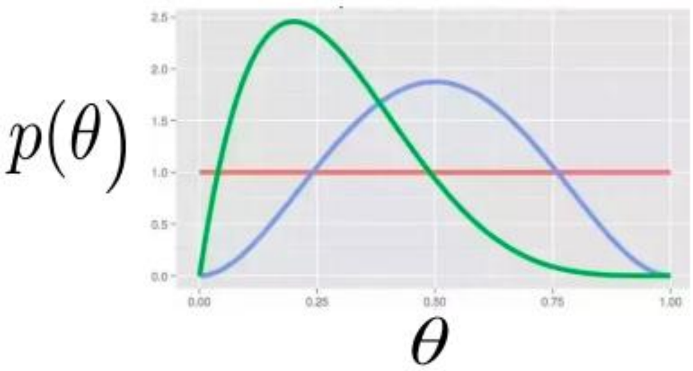
$$p(x_n|\theta)$$

Distribution of the inputs

- Can specify our assumptions about the unknowns $\boldsymbol{\theta}$ using a “prior distribution”

Represents our belief about the unknown parameters before we see the data

$$p(\theta)$$



The “Bayesian Learning” approach

- After seeing some data $\mathcal{D}$, can update the prior into a posterior distribution $p(\theta|\mathcal{D})$

The Core of PML: Two Basic Rules of Probability

- Sum Rule (marginalization): Distribution of $\boldsymbol{a}$ considering for all possibilities of $\boldsymbol{b}$

If $\boldsymbol{b}$ is a discrete r.v.

$$p(a) = \sum_{b} p(a,b) \quad \underline{\textbf{or}} \quad p(a) = \int p(a,b)db$$

If $\boldsymbol{b}$ is a continuous r.v.

- Product Rule (special case of the more general "chain rule" of probability)

$$p(a,b) = p(a)p(b|a) = p(b)p(a|b)$$

- These two rules are the core of most of probabilistic/Bayesian ML
  - Bayes rule easily derived from the sum and product rules

$$p(b|a) = \frac{p(b)p(a|b)}{p(a)} = \frac{p(b)p(a|b)}{\int p(a,b)db}$$

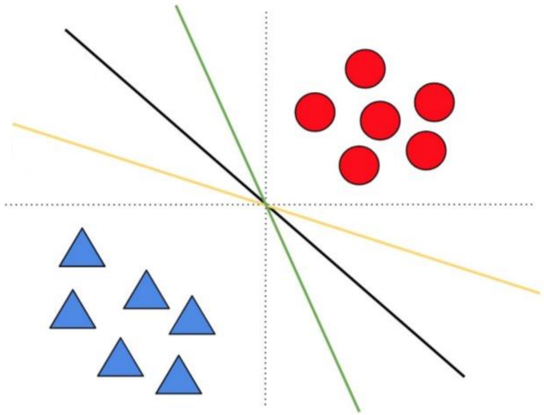
Assuming $\boldsymbol{b}$ is a continuous r.v.

ML and Uncertainty
(and how PML handles uncertainty)

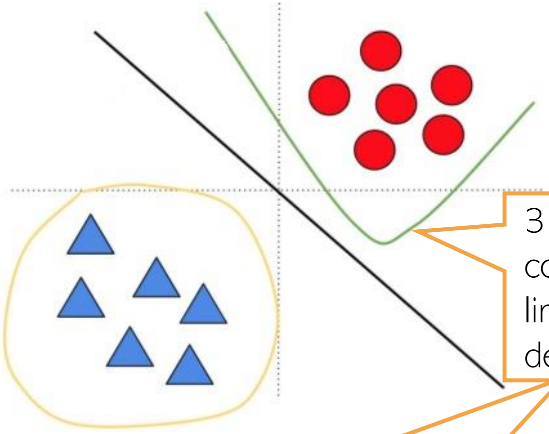
Uncertainty due to Limited Training Data

- Model/parameter uncertainty is due to not having enough training data

Same model class (linear models) but uncertainty about the weights



Uncertainty not just about the weights but also the model class



3 different model classes considered here (with linear, polynomial, circular decision boundaries)

Each model class itself will have uncertainty (like left fig) since there isn't enough training data

- Also called **epistemic uncertainty**. Usually reducible
  - Vanishes with "sufficient" training data

Image credit: Balaji L, Dustin T, Jasper N. (NeurIPS 2020 tutorial)

Uncertainty due to Inherent Noise in Training Data

- Data uncertainty can be due to various reasons, e.g.,
  - Intrinsic hardness of labeling, class overlap
  - Labeling errors/disagreements (for difficult training inputs)
  - Noisy or missing features

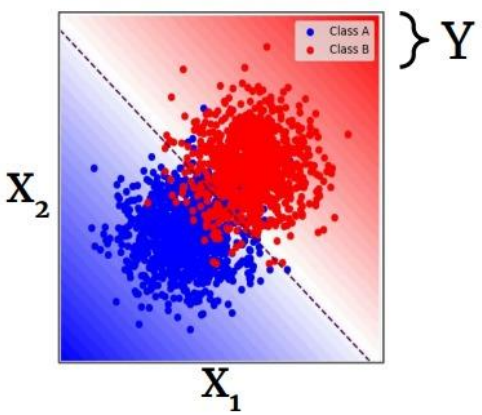


Image credit: Eric Nalisnick

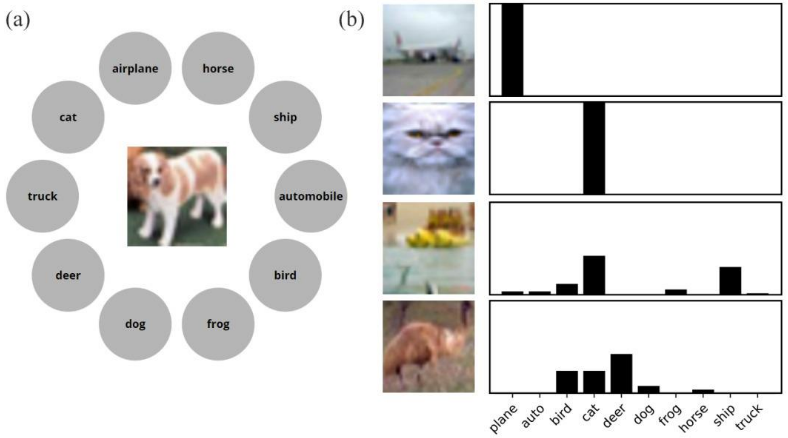


Image source: “Improving machine classification using human uncertainty measurements” (Battleday et al, 2021)

- Also called aleatoric uncertainty. Usually irreducible
  - Won’t vanish even with infinite training data
  - Note: Can sometimes vanish by adding more features (figure on the right) or switching to a more complex model

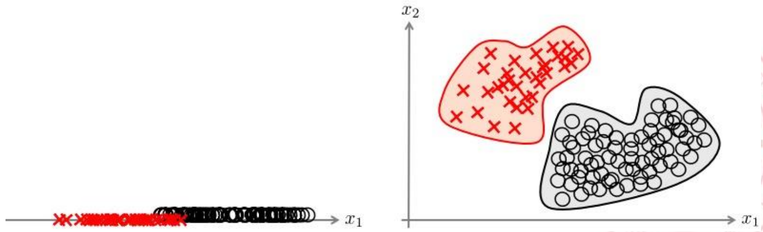


Image source: “Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods” (H&W 2021)

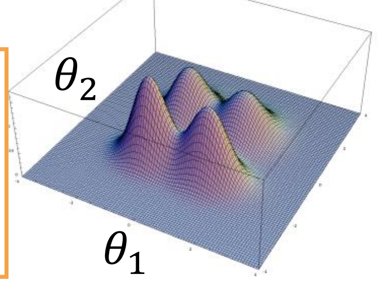
How to Estimate Uncertainty?

In this course, we will mostly focus on the Bayesian approach but other two approaches are also popular and will also be discussed

- Uncertainty in parameters: This can be estimated/quantified via mainly three ways:

$p(\theta|\mathcal{D})$

A case of 2-dim θ

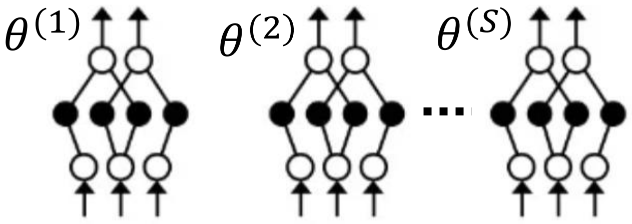


Bayesian way: Treat params as random variables and estimate their distribution conditioned on the given training data (a.k.a. posterior distribution)

Sampling multiple training sets and estimating the parameters from each training set

$$\{\hat{\theta}(\mathcal{D}') : \mathcal{D}' \sim p^*\}$$

Frequentist way: Treat params as fixed unknowns and estimate them using multiple datasets. This yields a set/distribution over the params (not a "posterior" but a distribution nevertheless!)



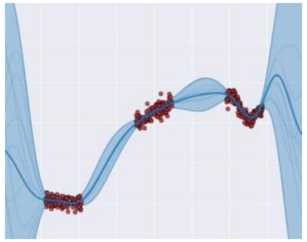
Ensemble: Train the same model with S different initializations or different subsets of the training data. Each run will give a different estimate, so we get a set of param estimates

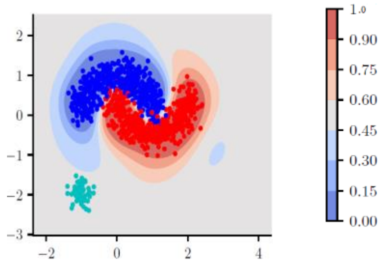
- Uncertainty in predictions: Usually estimated by computing and reporting the mean and variance of predictions made using many possible values of $\boldsymbol{\theta}$. Commonly reported as:

Predictive Distribution

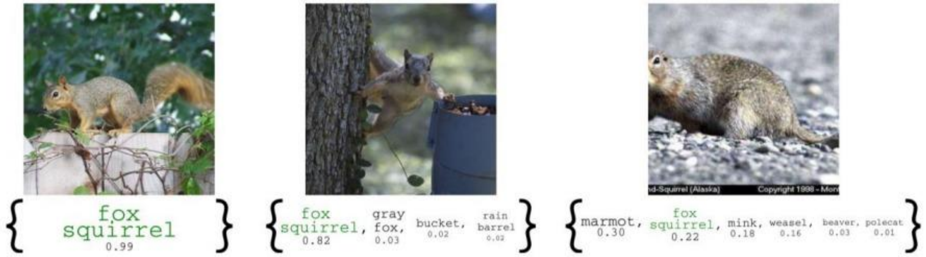
$p(y_*|x_*, \mathcal{D})$

Can get both mean and variance/quantiles of the prediction





Sets/intervals of possible predictions


{ fox squirrel 0.99 }
{ fox squirrel, 0.82 gray fox, 0.03 bucket, 0.02 rain barrel 0.02 }
{ marmot, 0.30 fox squirrel, 0.22 mink, 0.18 weasel, 0.16 beaver, 0.03 polecat 0.01 }
d-Squirrel (Alaska)
Copyright 1998 - Mon

Predictive Uncertainty is Useful

- Predictive uncertainty gives an idea about how much to trust a prediction
- It can also "guide" us in sequential decision-making:

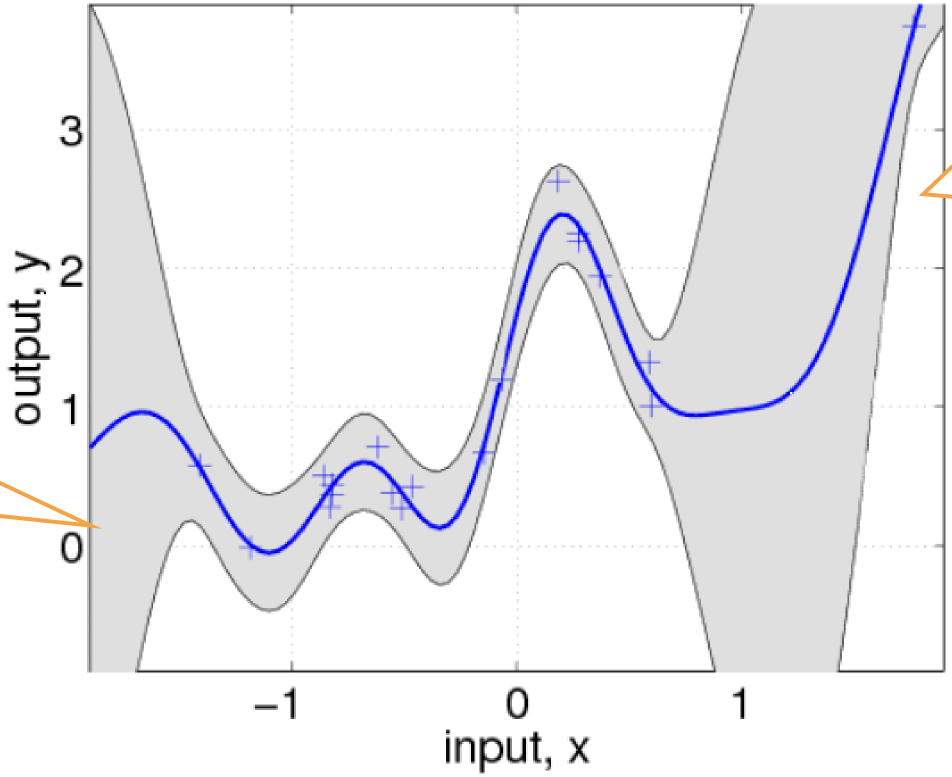
Test output

Test input

$$p(y_*|x_*, D) = \mathcal{N}(y_*|\mu_*, \sigma_*^2)$$

Training data

Blue curve is the mean of the function (learned so far using the available data), shaded region denotes the current predictive uncertainty



Given our current estimate of the regression function, which training input(s) should we add next to improve its estimate the most?

Uncertainty can help here: Acquire training inputs from regions where the function is most uncertain about its current predictions

- Applications in active learning, reinforcement learning, Bayesian optimization, etc

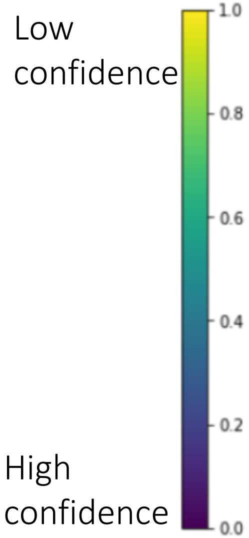
Predictive Uncertainty is Useful

For classification, "confidence" refers to the probability of the class predicted to be the most likely

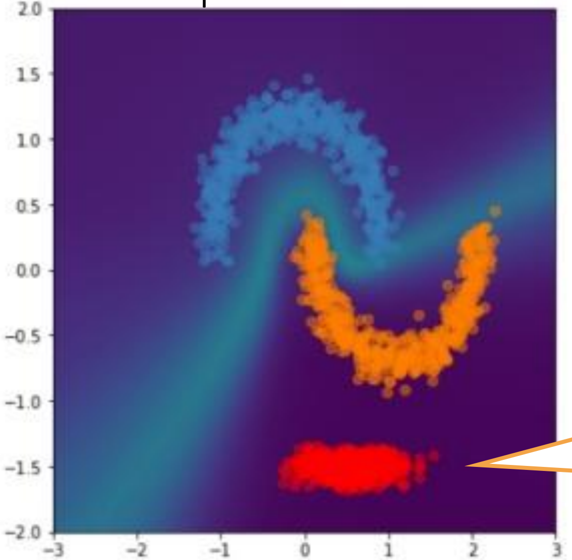
- Many modern deep neural networks (DNN) tend to be overconfident

One of the reasons is that they don't incorporate uncertainty

- Especially true if test data is "out-of-distribution (OOD)"



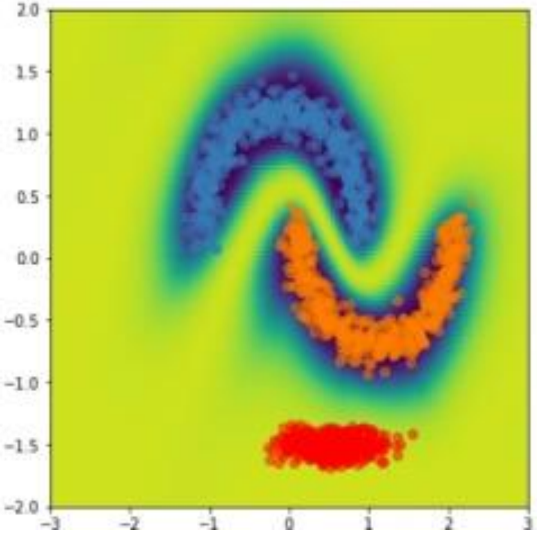
Confidence map of a non-probabilistic DNN



Overconfident model

Model has high confidence for predictions on even inputs that are far away from training data

Confidence map of a probabilistic DNN properly incorporating uncertainty

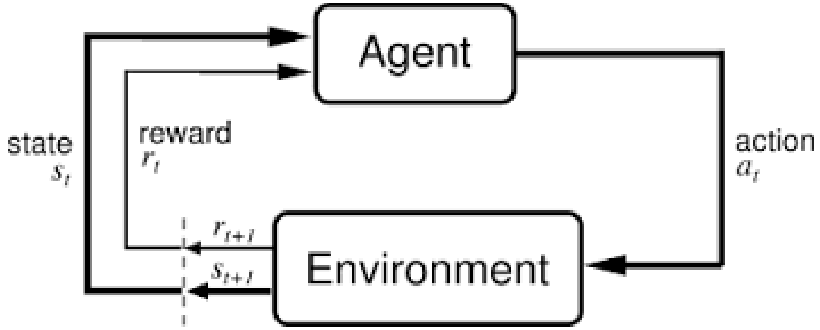


- Prob. deep models often provide better uncertainty estimates to flag OOD data

Image source: "Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness" (Liu et al, 2020)

Reinforcement Learning as Probabilistic Modeling

- Interaction between an agent and an environment
- Interaction trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$
- Can define a probabilistic model for this trajectory

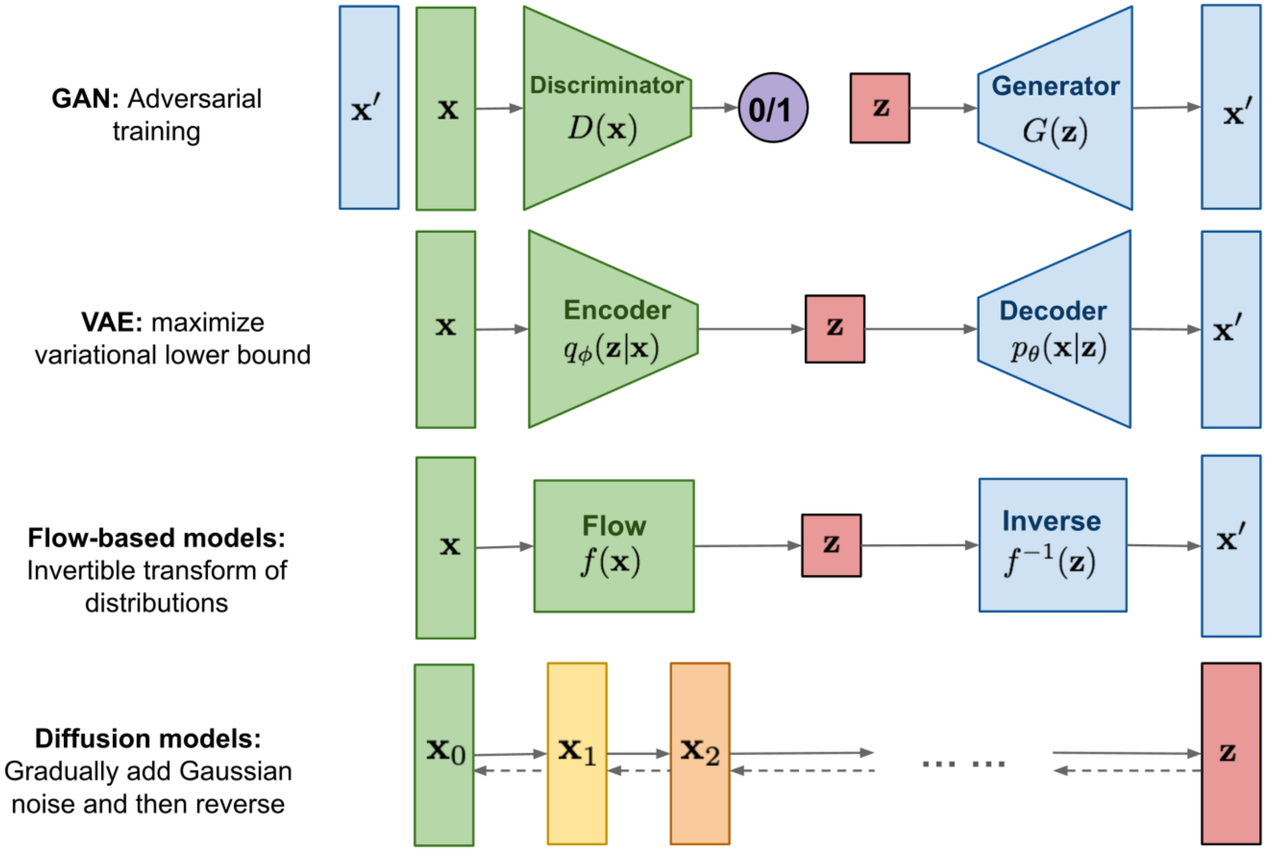


$$p(\tau) = p(s_0) \prod_{t=0}^{T-1} p(a_t|s_t) p(s_{t+1}|s_t, a_t) p(r_t|s_t, a_t)$$

- Dynamics (state transitions, actions taken) may be stochastic
- Rewards may be noisy observations
- States may be observed/latent

(Deep) Generative Models as Probabilistic Modeling

- Generative models of data can also be defined as probabilistic models


GAN: Adversarial training
$\mathbf{x}'$
$\mathbf{x}$
Discriminator $D(\mathbf{x})$
0/1
$\mathbf{z}$
Generator $G(\mathbf{z})$
$\mathbf{x}'$
VAE: maximize variational lower bound
$\mathbf{x}$
Encoder $q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{z}$
Decoder $p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
Flow-based models: Invertible transform of distributions
$\mathbf{x}$
Flow $f(\mathbf{x})$
$\mathbf{z}$
Inverse $f^{-1}(\mathbf{z})$
$\mathbf{x}'$
Diffusion models: Gradually add Gaussian noise and then reverse
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_2$
… …
$\mathbf{z}$

- Learning such models will also be a topic of study in this course

Figure credit: Lilian Weng

(Large) Language Models as Probabilistic Modeling

- An LLM defines a probability distribution over sequences of tokens

$$\boldsymbol{x} = \{x_1, x_2, \dots, x_N\}$$

- Autoregressive modeling is a popular way to define this distribution

$$p(\boldsymbol{x}) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\dots \quad = \prod_{i=1}^{N} p(x_i|\boldsymbol{x}_{<i})$$

- Params $\boldsymbol{\theta}$ of each conditional $\boldsymbol{p(x_i|x_{<i})}$ defined using neural nets (e.g., transformer)

$$p_\theta(x_i|\boldsymbol{x}_{<i}) = \text{softmax}(\underbrace{f_\theta}_{\text{A neural net}}(\boldsymbol{x}_{<i}))$$

- One parameters are estimated, the model can be used to generate data

Tentative List of Topics

- Basics of probabilistic modeling and inference
  - Common probability distributions
  - Parameter estimation
  - Making predictions in probabilistic models
- Probabilistic models for regression, classification, clustering, dimensionality reduction
- Latent Variable Models (for i.i.d., sequential, and relational data)
- Sampling from probability distributions
- Computing intractable posteriors and intractable expectations
  - Approximate Bayesian inference (EM, variational inference, MCMC sampling, etc)
- Deep Generative Models (VAEs, Diffusion Models, Large Language Models)
- Sequential decision-making, Reinforcement Learning