STAT 453: Introduction to Deep Learning and Generative Models

Lecture 23: Review & Open Directions

Ben Lengerich

December 8, 2026

Logistics

Project final report

Tonight at 11:59 PM

Submit on Canvas.

Thanks for last week's presentations.

Final exam

Saturday, December 12

5:05–7:05 PM

Bardeen 140

Ben Lengerich © University of Wisconsin-Madison 2026

The final exam

120 minutes / 150 points / Cumulative

Question formats

36 true/false (1 pt each)

23 multiple choice (3 pts each)

3 short answer (15 pts each)

Several multiple-choice questions ask you to diagnose training curves.

Materials and answers

Open notes Printed or handwritten.

No electronics or calculator Leave answers as fractions, powers, or logs (2^{-10}, $\ln 2$).

Short answers Show your work inside each box. Only work inside the box is graded.

Ben Lengerich © University of Wisconsin-Madison 2026

How to prepare

- **Work the study bank:** same formats as the exam, with an answer key
- **Principles over notation:** be able to say why each method works and what goes wrong
- **Small computations by hand:** softmax with ln-valued scores, KL of Gaussians, chain rule, parameter counts
- **Curves:** for each plot, ask what each line measures and what it must imply
- **Notes:** organize by idea, not by lecture; one page of formulas you can find fast

Ben Lengerich © University of Wisconsin-Madison 2026

The course in one slide

Neural networks → **Generative models** → **Sequence models** → **Large language models**

- perceptron → MLP
- backpropagation
- SGD, momentum, Adam
- regularization, norms, init
- CNNs: weight sharing

- generative vs. discriminative
- autoencoders
- VAEs: ELBO
- GANs: a two-player game
- diffusion: learn to denoise

- RNNs, BPTT
- vanishing / exploding grads
- LSTMs: additive memory
- attention
- Transformers

- GPT: next-token MLE
- scaling, decoding
- SFT, RLHF, DPO
- reasoning, RLVR
- PEFT, RAG (not on final)

Main ideas Backpropagation throughout. LLMs are generative models built on attention.

The Course in Five Ideas

1. Most losses are negative log-likelihoods.

Model	Loss	Likelihood behind it
Regression	Squared error	Gaussian noise, fixed σ
Classifier	Cross-entropy	Categorical (softmax)
Language model	Next-token cross-entropy	Autoregressive: product of $p(x_t \mid x_{<t})$
VAE	−ELBO	A lower bound on log p(x)
Diffusion	Noise-prediction MSE	A (reweighted) ELBO
Reward model, DPO	−log σ(score difference)	Bradley–Terry on preference pairs

MLE minimizes $KL(p_{data} \| p_\theta)$: it must cover every mode of the data

2. Gradients are the chain rule.

Backprop reuses values from the forward pass.

Multiply derivatives along each path. Add contributions to shared weights.

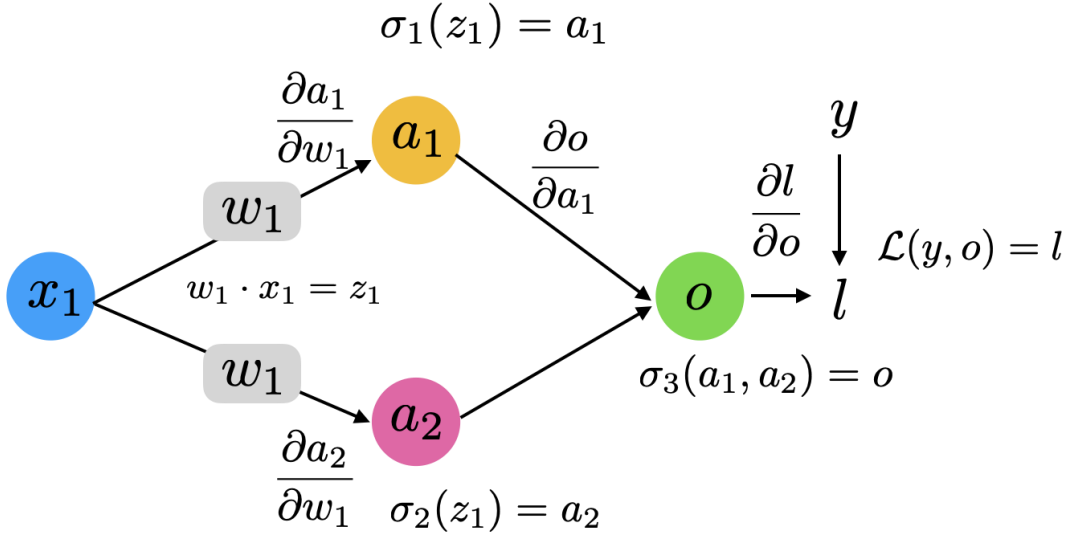
Long chains can make gradients vanish or explode.

Gradients through sampling

Continuous z: reparameterize, z = μ + σ ⊙ ε (VAE).

Discrete tokens: policy gradient, E[R ∇ log π] (RLHF, GRPO).

$$\mathcal{L}\big(y, \sigma_3\big[\sigma_1(w_1 \cdot x_1), \sigma_2(w_1 \cdot x_1)\big]\big)$$



Upper path

$$\frac{\partial l}{\partial w_1} = \frac{\partial l}{\partial o} \cdot \frac{\partial o}{\partial a_1} \cdot \frac{\partial a_1}{\partial w_1} + \frac{\partial l}{\partial o} \cdot \frac{\partial o}{\partial a_2} \cdot \frac{\partial a_2}{\partial w_1}$$ (multivariable chain rule)

Lower path

Ben Lengerich © University of Wisconsin-Madison 2026

3. Additive paths keep gradients alive.

- Products of many derivatives can shrink or explode. **An identity skip connection gives gradients a direct path.**
- **LSTM cell state:** $c_t = f \odot c_{(t-1)}$ + new content
- **Residual connections:** h + F(h) in ResNets and every Transformer block
- Exploding instead? **Clip the gradient norm**

For y = x + F(x), dy/dx = 1 + F'(x). The skip path contributes the 1.

Ben Lengerich © University of Wisconsin-Madison 2026

4. Structure is a strong prior.

- **CNN:** locality + one filter at every position (a sparse, tied W)
- **RNN:** the same update at every time step
- **Attention:** content-based mixing; permutation-equivariant without positions
  - Positional encodings and causal masks put order back in
- Right structure → fewer parameters, less data, better generalization

Ben Lengerich © University of Wisconsin-Madison 2026

5. We optimize flawed proxies (Goodhart's law).

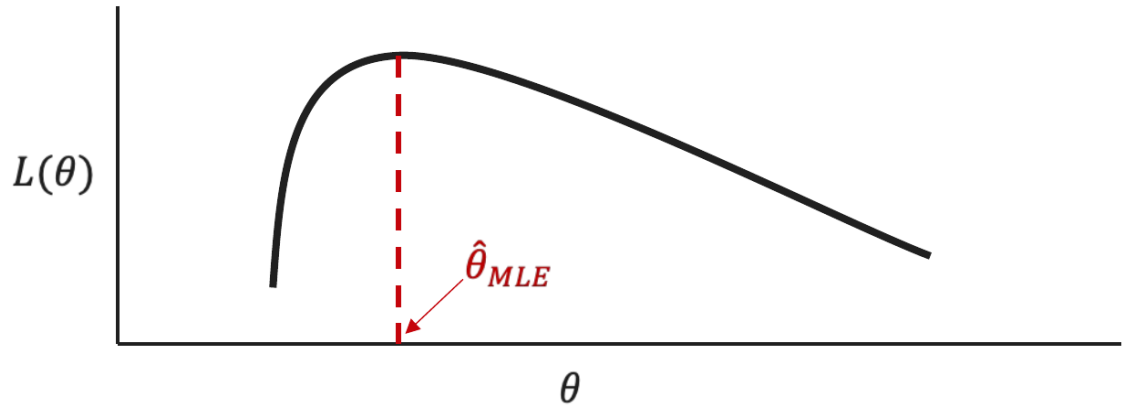
- **Training loss** is a proxy for test loss → overfitting; watch validation loss
- **The discriminator** is a moving proxy for p_{data} → unstable GAN training
- **A reward model** is a proxy for human judgment → reward hacking; KL leash, early stopping
- **A checker** is a proxy for correctness → special-cased unit tests
- **Benchmarks** are proxies for usefulness
- Same fixes everywhere: **hold-out evaluation, regularization toward a reference, stop early**
- Takeaway: Statistical thinking still applies.

Ben Lengerich © University of Wisconsin-Madison 2026

Module by Module

Basics of Estimation from Observations

The likelihood curve



The peak gives the parameter value that makes the observed data most likely.

Lecture 3, Maximum Likelihood Estimation (schematic)

Estimating a probability

Three successes in four independent trials.

Likelihood of the sequence = $p^3(1 - p)$

Maximum likelihood gives $\hat{p} = ¾$.

Estimating network weights

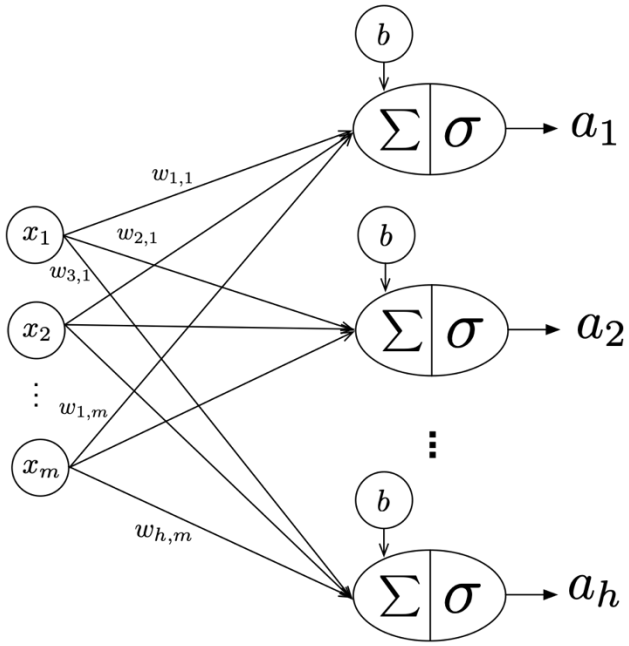
The network predicts p(y | x) for each input.

Minimize mean negative log-likelihood of the observed labels.

Fit on training data. Use held-out data to see whether the fit generalizes.

Ben Lengerich © University of Wisconsin-Madison 2026

What a dense layer computes



note that $w_{i,j}$ refers to the weight connecting the *j*-th input to the *i*-th output.

$$\text{where } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,m} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{h,1} & w_{h,2} & \cdots & w_{h,m} \end{bmatrix}$$

One unit

$z = w^T x + b$

A perceptron thresholds z. ReLU returns max(0, z).

Weights set the boundary's angle and bias sets its offset.

A layer of units

$z = Wx + b$, then $a = \phi(z)$.

Each row of W holds one unit's weights.

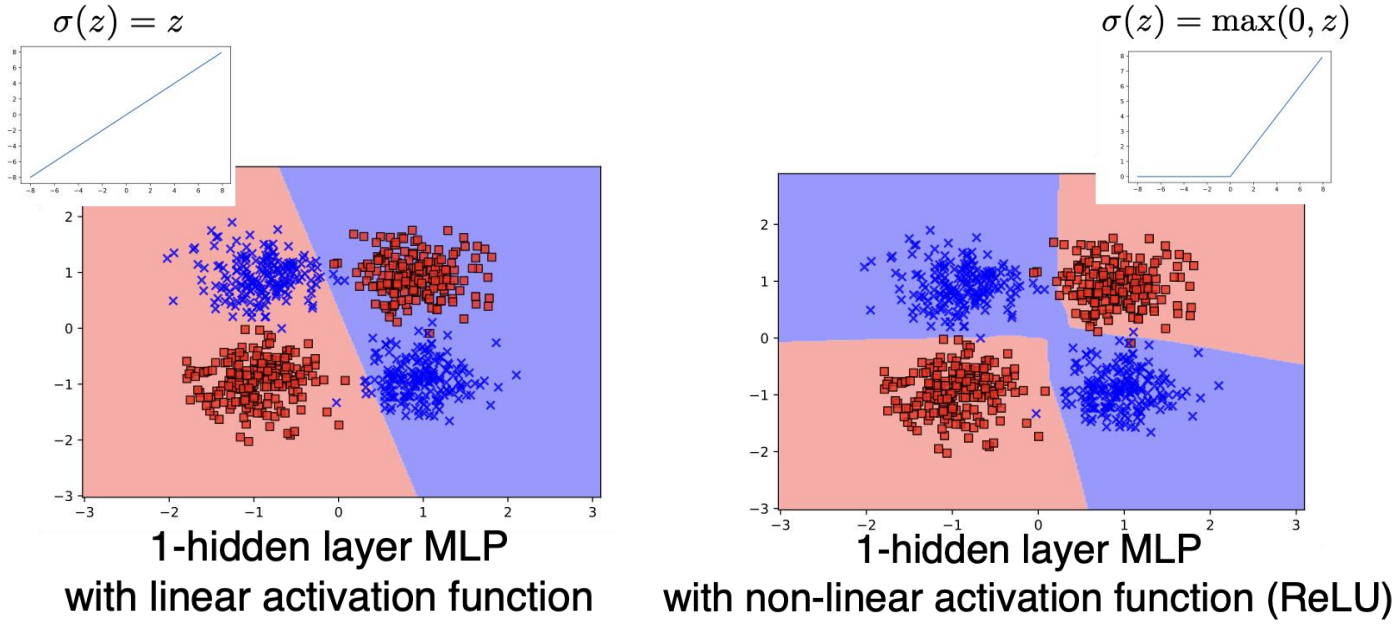
Four inputs and three units give 3 × 4 weights and 3 biases.

A layer with 4 inputs and 3 units has 15 parameters at any batch size.

Ben Lengerich © University of Wisconsin-Madison 2026

Why XOR needs a nonlinear layer

XOR is 1 at (1, 0) and (0, 1), and 0 at (0, 0) and (1, 1). One line cannot separate the classes.



Two ReLU units

$s = x_1 + x_2$

$h_1 = \text{ReLU}(s)$

$h_2 = \text{ReLU}(s - 1)$

For s = 0, 1, 2:

$h_1 - 2h_2$ gives 0, 1, 0.

Lecture 7, Sebastian Raschka's XOR notebook

Stacking affine layers still gives an affine map. ReLU makes piecewise-linear boundaries possible.

Ben Lengerich © University of Wisconsin-Madison 2026

One minibatch training step

Forward pass

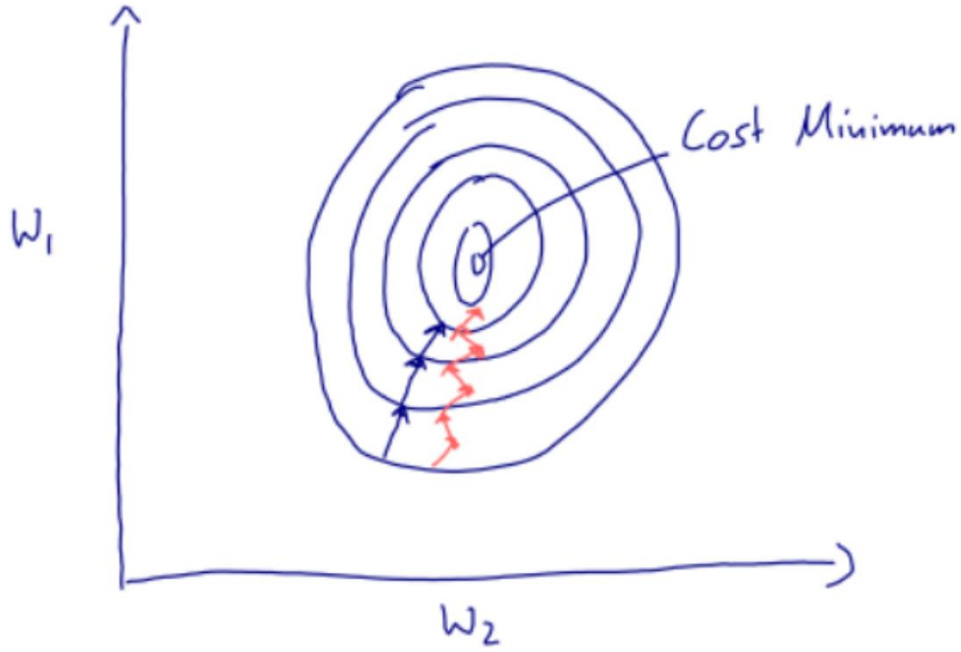
Apply softmax to the logits. For true class y, cross-entropy is −log p(y).

Backward pass

Clear old gradients. Backpropagate the mean batch loss through saved activations.

Parameter update

SGD subtracts η times the gradient. Momentum smooths gradients. Adam also rescales them per parameter.



Different batches give different gradient estimates, so the path can wobble.

Lecture 11, Minibatch Training Recap

At evaluation, use held-out examples and leave the weights fixed.

Ben Lengerich © University of Wisconsin-Madison 2026

Neural networks

Representation and gradients

Perceptron The learning rule corrects misclassified training examples.

Expressivity A wide enough nonlinear network can approximate a target function. Whether training finds it is a separate question.

Backprop Use the chain rule. ReLU passes gradients on positive inputs.

CNNs Know output shapes, parameter counts, and how filters share weights.

Training and generalization

Optimization Momentum smooths gradients. Adam also scales them per parameter.

Learning rate Warm up at the start. Smaller steps late in training help limit the effect of minibatch noise near a minimum.

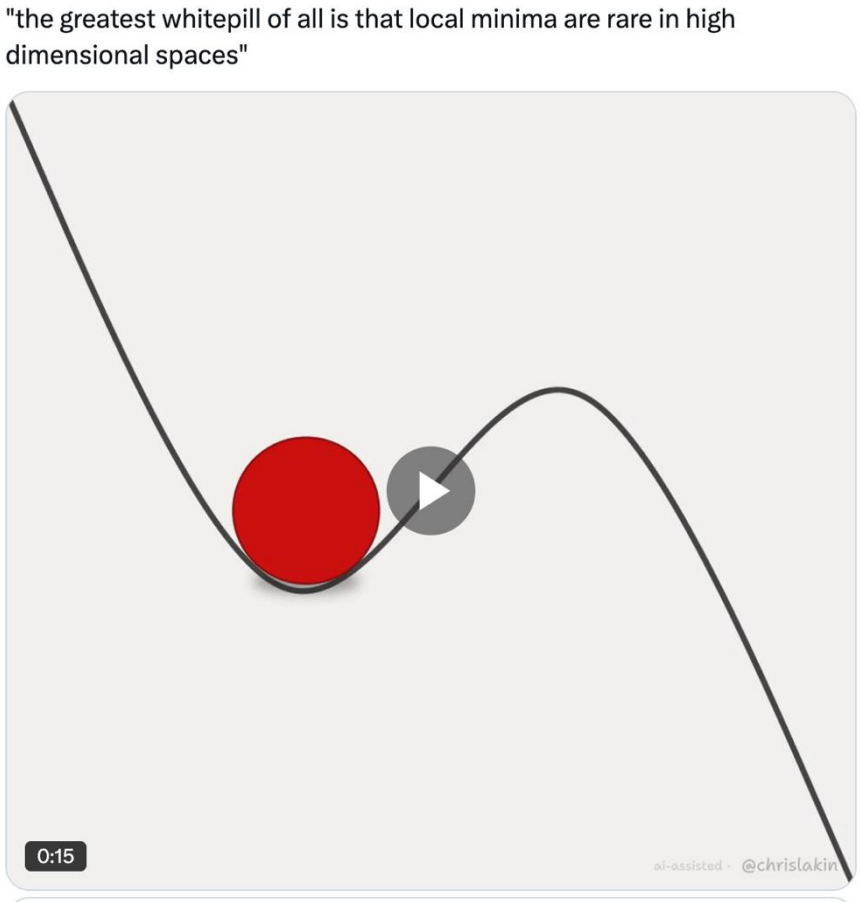
Regularization Weight decay, dropout, augmentation, and early stopping.

Normalization BatchNorm uses batch statistics in training and running estimates at test time. LayerNorm works per example.

Initialization Break symmetry. Xavier and He help control activation variance.

Ben Lengerich © University of Wisconsin-Madison 2026

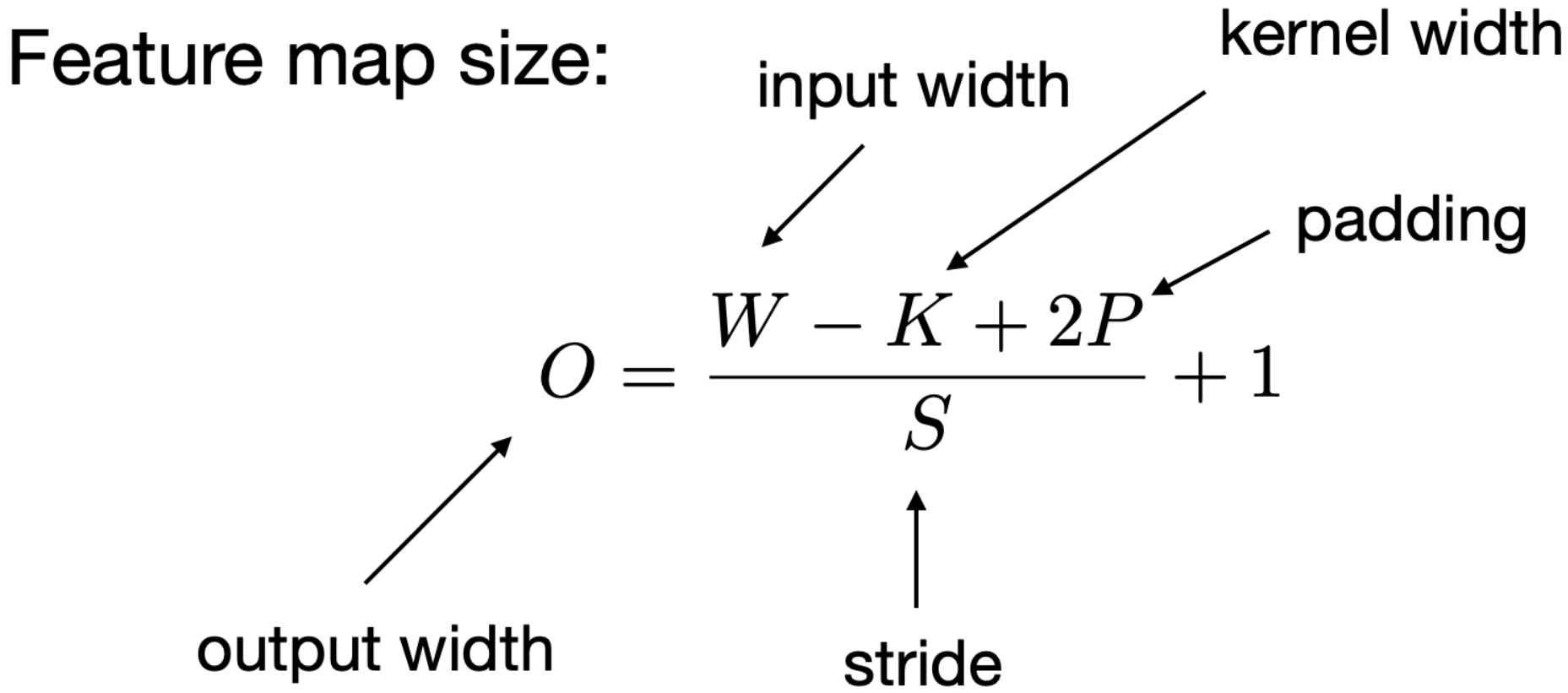
Who’s afraid of non-convex losses? (Lecture 7)



- **In 1-D, a dip traps the ball.** In d dimensions, a critical point is a local minimum only if the loss curves upward in every one of the d directions
- **Most critical points are saddles:** some direction still goes down, so gradient descent can keep moving (Dauphin et al., 2014)
- **Minibatch noise helps** push the parameters off a saddle
- **Caveat:** “rare” is a heuristic, not a theorem for every network. Bad minima can exist; in practice, the minima SGD finds tend to have similar loss

Impact of convolutions on size

Feature map size:



Round the fraction down before adding 1. Stride and padding change the output size.

Ben Lengerich © University of Wisconsin-Madison 2026

Worked examples: neural networks

Backprop through two ReLU units

$z = ux + b$, $h = \mathrm{ReLU}(z)$, $\hat{y} = v^T h$

$L = \frac{1}{2}(\hat{y} - y)^2$, with b held fixed.

$x = 2$, $y = 1$, $u = (1, -1)$, $b = (-1, 1)$, $v = (2, 3)$.

$z = (1, -1)$, $h = (1, 0)$, $\hat{y} = 2$, $L = \frac{1}{2}$

Let $e = \hat{y} - y = 1$.

$\nabla_v L = eh = (1, 0)$

$\partial L / \partial u_i = e v_i \cdot 1[z_i > 0] \cdot x$

$\nabla_u L = (4, 0)$. Unit 2 is inactive.

With $\eta = \frac{1}{4}$, $u \rightarrow (0, -1)$, $v \rightarrow (7/4, 3)$.

Convolution and momentum

64×64×3 input, sixteen 3×3 filters, stride 2, padding 1

$\lfloor (64 - 3 + 2)/2 \rfloor + 1 = 32$

Output shape = 32×32×16

Parameters = $3 \cdot 3 \cdot 3 \cdot 16 + 16 = 448$ at any image size

$v(t) = \beta v(t - 1) + g$, with $g = 1$ and $\beta = 0.8$

Momentum limit = $1/(1 - \beta) = 5$

Ben Lengerich © University of Wisconsin-Madison 2026

Classifier gradients and weight decay

Softmax with a correct prediction

Logits are $(0, \ln 4)$. The true class is class 2.

Softmax gives $p = (\frac{1}{5}, \frac{4}{5})$.

The one-hot target is $y = (0, 1)$.

For cross-entropy, $\partial L/\partial z = p - y$.

Gradient = $(\frac{1}{5}, -\frac{1}{5})$.

The step raises the true-class logit and lowers the other.

L2 regularization with a data gradient

Add $(\lambda/2)\|w\|^2$ to the loss. Use $\lambda = \frac{1}{2}$ and learning rate $\eta = \frac{1}{5}$.

$w = (2, -1)$, data gradient $g = (1, 0)$

Total gradient = $g + \lambda w = (2, -\frac{1}{2})$.

Subtract $\frac{1}{5}(2, -\frac{1}{2})$ from $(2, -1)$.

New weights = $(8/5, -9/10)$.

The second weight shrinks even though its data gradient is zero.

Generative and discriminative classifiers

Logistic regression

Fit $p(y \mid x)$ directly from labeled examples.

For a new x, read the class probabilities from the model.

With more data, it can do better when naive Bayes's independence assumptions fail.

Naive Bayes

Fit $p(y)$ from label frequencies and fit the feature distributions given each class.

Use $p(x, y) = p(y) \prod_j p(x_j \mid y)$.

Sum $p(x, c)$ over classes c to get $p(x)$.

$p(y \mid x) = p(x, y) / p(x)$

For class y, draw new inputs from $p(x \mid y)$.

Conditional independence can help when labeled data are scarce.

Ben Lengerich © University of Wisconsin-Madison 2026

Generative models: what to know

GAN generator losses

For a generated sample, let D = σ(a).

Saturating log(1 − D) has logit gradient −D.

Non-saturating −log D has gradient D − 1.

Near D = 0, −log D has a strong gradient.

Mode collapse leaves some modes uncovered.

Variational autoencoders

ELBO = reconstruction term − KL(q ‖ p)

Any encoder gives a lower bound.

Reparameterize and check for collapse.

Diffusion

The fixed forward process adds noise. Learned denoising takes many steps.

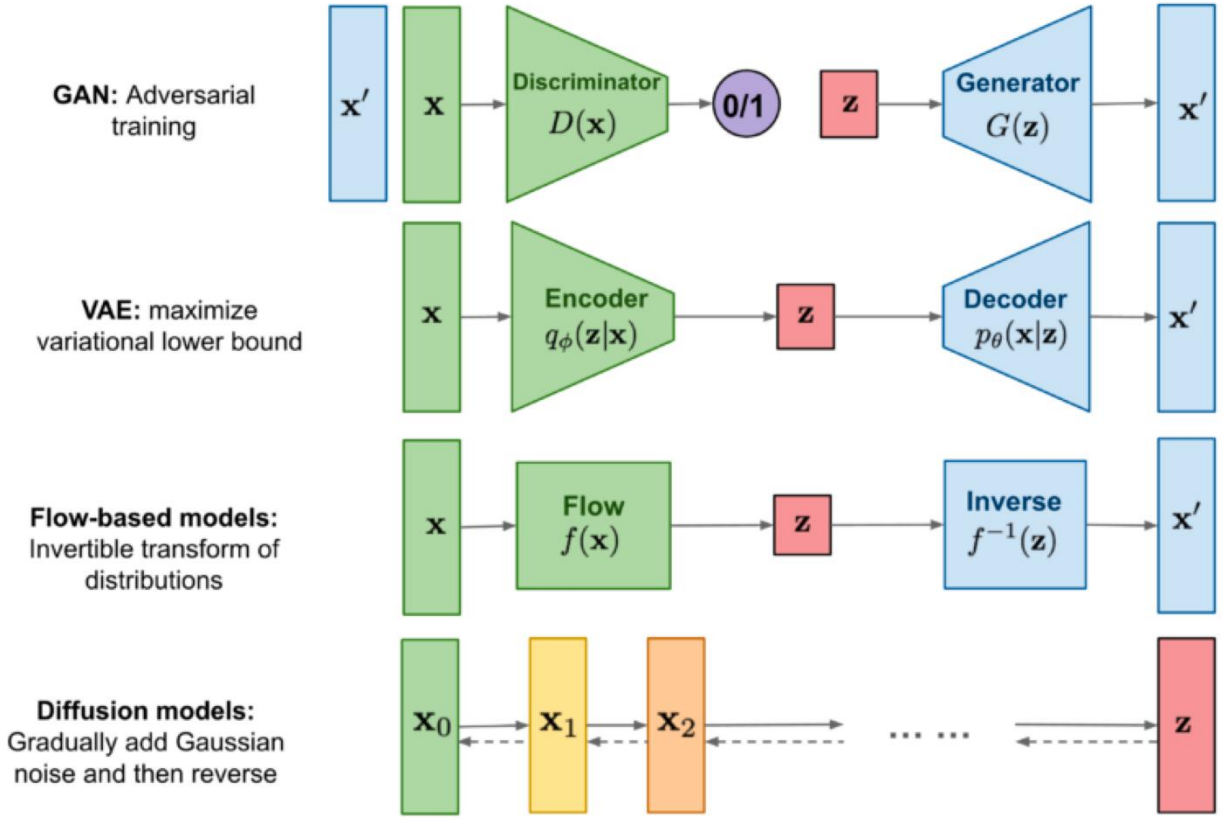


Diagram source: Lilian Weng

Worked examples: generative models

VAE KL term

$KL(N(\mu, \sigma^2) \| N(0, 1))$

$= \frac{1}{2}(\mu^2 + \sigma^2 - 1 - \ln \sigma^2)$

$\mu = 1, \sigma = 1$

$KL = \frac{1}{2}(1 + 1 - 1 - 0) = \frac{1}{2}$

$\mu = 0, \sigma = 1$

KL = 0 (q equals the prior)

Optimal discriminator

$p_data(x) = 0.6, p_g(x) = 0.2$

$D^*(x) = 0.6 / 0.8 = \frac{3}{4}$

$D^* = 1$ where the generator misses a data mode.

$D^* = \frac{1}{2}$ where $p_g = p_data$ and both are positive.

Diffusion forward step

$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{(1 - \bar{\alpha}_t)}\, \varepsilon$

$x_0 = 1, \bar{\alpha}_t = 0.36, \varepsilon = -0.5$

Mean 0.6, variance 0.64

$x_t = 0.6 - 0.4 = 0.2$

Ben Lengerich © University of Wisconsin-Madison 2026

GAN optimization and diffusion denoising

Optimal discriminator for one bucket

One logit a is shared by 2 real and 6 generated examples. Let $D = \sigma(a)$.

Each example contributes $D - y$ to $\partial L/\partial a$.

$2(D - 1) + 6D = 8D - 2$.

The derivative increases with a, so its zero minimizes the loss.

The optimum is $D = \frac{1}{4}$.

Weight each real example 3× and the totals balance. The optimum becomes ½.

Recovering a clean two-dimensional input

Suppose $\bar{\alpha}_t = 9/25$. Then the signal and noise coefficients are ⅗ and ⅘.

The noisy input is $x_t = (1, -1)$.

The denoiser predicts noise $\hat{\varepsilon} = (\frac{1}{2}, -2)$.

Rearrange $x_t = \frac{3}{5}x_0 + \frac{4}{5}\varepsilon$:

$\hat{x}_0 = [(1, -1) - \frac{4}{5}(\frac{1}{2}, -2)] / (\frac{3}{5})$

$\hat{x}_0 = (\frac{3}{5}, \frac{3}{5}) / (\frac{3}{5}) = (1, 1)$.

Ben Lengerich © University of Wisconsin-Madison 2026

Sequence models: what to know

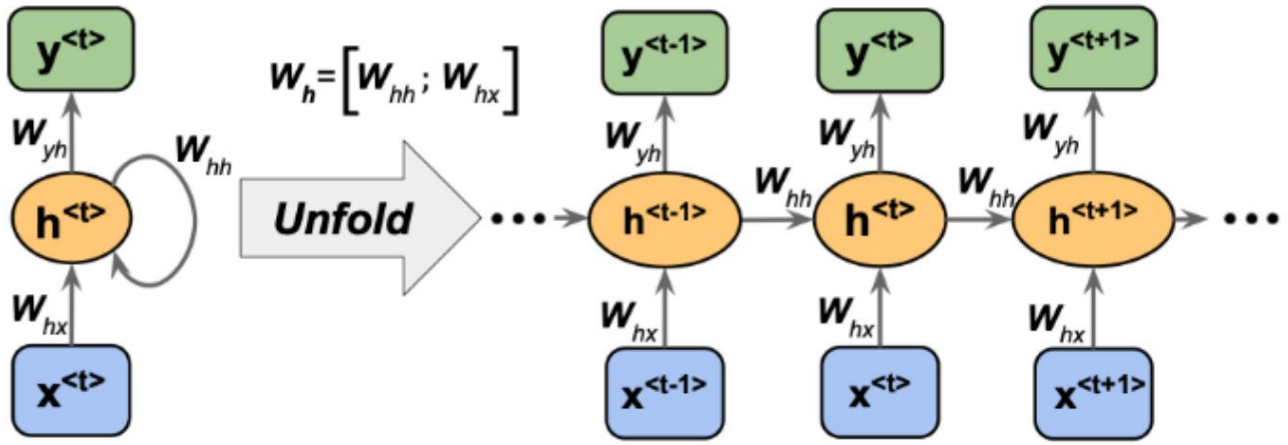
- **RNNs:** one shared update over time; BPTT; hidden states computed sequentially
- **Gradients:** a product of Jacobians over time → vanish or explode; clipping fixes explosion only; LSTMs add an additive path
- **Seq2seq bottleneck:** without attention, the whole input passes through one vector
- **Attention:** $\text{softmax}(QK^T/\sqrt{d_k})V$. If the independent score terms have variance 1, the dot product has variance d_k. Dividing by $\sqrt{d_k}$ restores variance 1 and helps avoid softmax saturation.
- **Transformer block:** attention mixes positions, the feed-forward net works per position, residuals + LayerNorm
- **Positions:** attention alone is permutation-equivariant; learned vs. sinusoidal encodings

Ben Lengerich © University of Wisconsin-Madison 2026

Recurrence and self-attention

RNNs pass state forward step by step. GPT uses causal self-attention over earlier tokens.

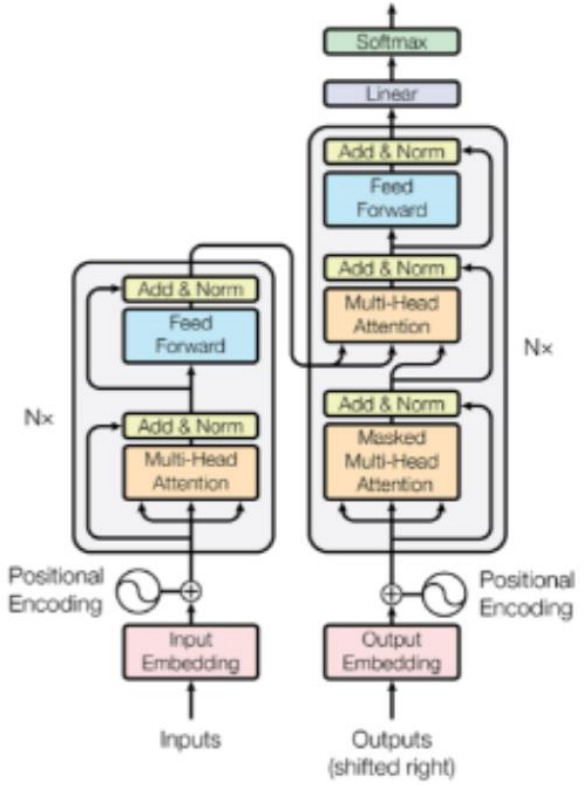
Recurrent neural network



The original Transformer has an encoder and a decoder. GPT uses a decoder-only architecture.

Raschka & Mirjalili, Python Machine Learning, 3rd ed. (2019)

Original Transformer



Vaswani et al., Attention Is All You Need (2017)

Sequence architecture choices

Context and positions

Bidirectional RNN The forward and backward states bring left and right context to each position.

BERT Predicts masked tokens from context on both sides.

Learned absolute positions A model learns one vector per position. A longer sequence needs vectors for positions beyond the trained table.

Heads and experts

Attention heads At width 12, three heads get 4 features each and six get 2 each. Both use the same total width for each Q, K, and V projection.

Top-2 expert routing With 8 experts or 32, each token runs through two expert networks. More experts increase stored parameters and router work.

Ben Lengerich © University of Wisconsin-Madison 2026

Worked examples: sequence models

Attention by hand

Scores = (0, ln 3)

Exponentials = (1, 3)

Softmax weights (¼, ¾)

$v_1 = (2, 0)$, $v_2 = (0, 4)$

$\frac{1}{4}v_1 + \frac{3}{4}v_2 = (0.5, 3)$

Gradient clipping

$g = (6, 8)$, $\|g\| = 10$

Scale g by 2/10 to cap the norm at 2.

Clipped g = (1.2, 1.6)

Direction stays the same.

Vanishing and exploding

$h(t) = Wh(t - 1)$

$W = \mathrm{diag}(-3, \frac{1}{4})$

$\partial h_5 / \partial h_0 = W^5$

$W^5 = \mathrm{diag}(-243, 1/1024)$

The first coordinate grows in magnitude and flips sign each step. The second shrinks.

Ben Lengerich © University of Wisconsin-Madison 2026

Causal attention in training and generation

A query at position 2

Token values are (2, 6, 10, 14). The query's exponentiated scores are (2, 1, 3, 1).

With a causal mask, positions 3 and 4 have zero weight before normalization.

The remaining weights are (⅔, ⅓).

Masked output = (2·2 + 1·6)/3 = 10/3.

With all positions visible, the same query gives (4 + 6 + 30 + 14)/7 = 54/7.

The sequence A B C D

Training The whole sequence is supplied as input.

Position 2 sees A B and predicts C. Position 3 sees A B C and predicts D.

The model computes both predictions in the same forward pass.

Generation With prompt A B, C is still unknown.

Append sampled C before predicting D.

Ben Lengerich © University of Wisconsin-Madison 2026

Large language models: what to know

- **Pretraining:** next-token MLE (= forward KL); scaling laws; decoding (greedy, temperature, top-k, top-p)
- **SFT:** imitate demonstrations, loss on response tokens; pick the checkpoint on validation loss
- **Reward model:** Bradley–Terry; only differences matter. **RLHF:** reward − $\beta \cdot \mathrm{KL}$ to π_{ref}; policy gradient on the model's own samples
- **DPO:** the same objective as a loss on pairs; implicit reward $\beta \log \pi_\theta / \pi_{\mathrm{ref}}$
- **RLVR:** a checker instead of a reward model; GRPO's group baseline; test-time compute
- **Prompting: in-context learning changes the input, not the weights. (PEFT and RAG: not on the final)**

Ben Lengerich © University of Wisconsin-Madison 2026

Pretraining, fine-tuning, and prompting

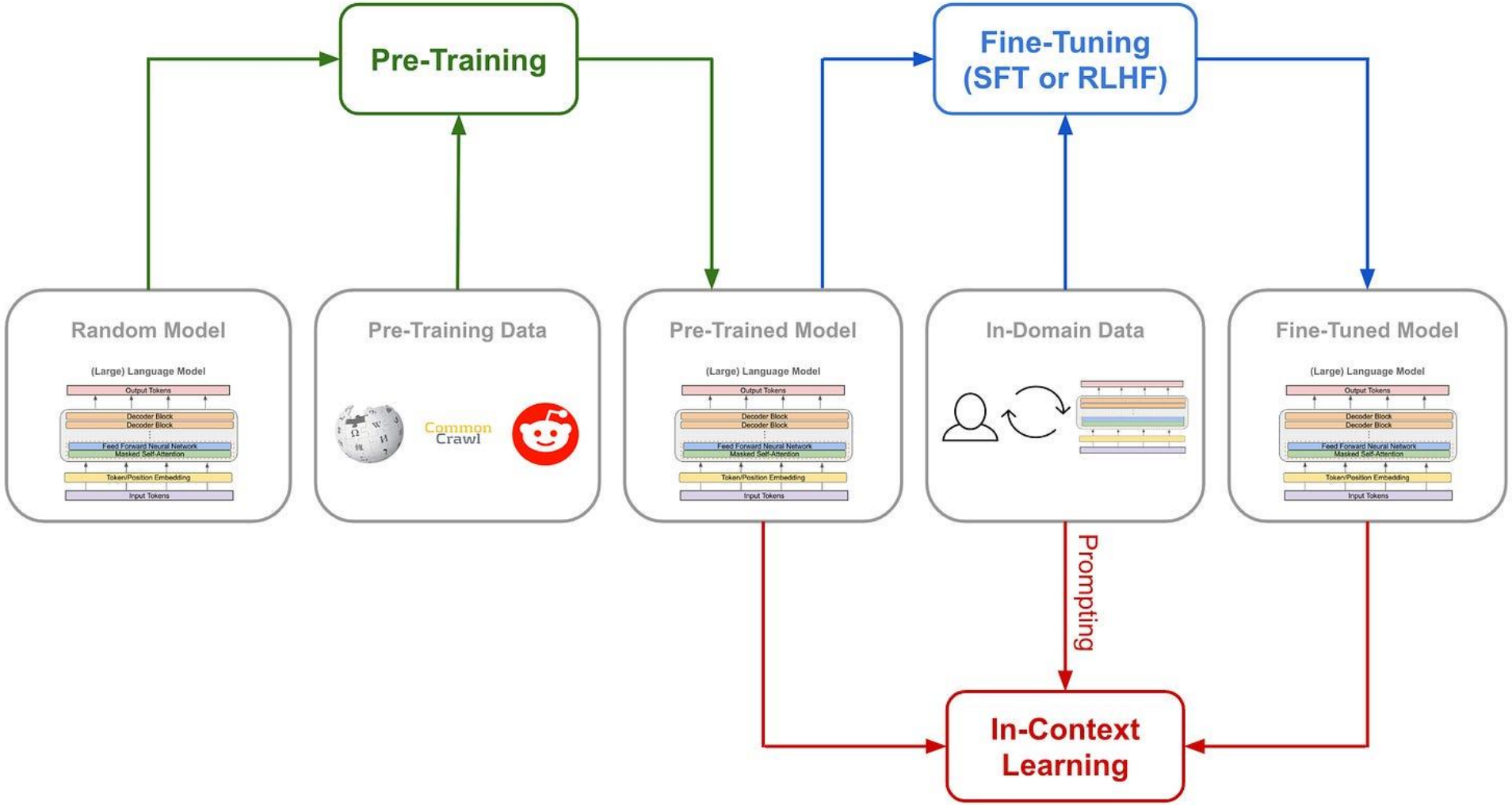


Diagram source: Cameron R. Wolfe

Worked examples: LLMs

Likelihood and preferences

p(a) = ¼ and p(b | a) = ¼.

NLL = 4 bits, or 2 bits/token.

Perplexity = 4

Bradley–Terry Adding c to every reward leaves pairwise preferences unchanged.

DPO At initialization, $\pi\theta = \pi\text{ref}$ and the loss is log 2.

Reinforcement learning and decoding

GRPO Rewards (1, 1, 0, 0) give advantages (+1, +1, −1, −1).

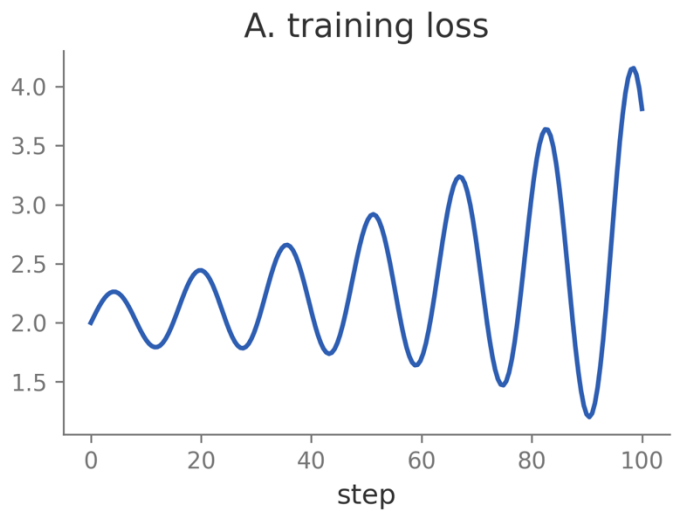
Rewards (1, 1, 1, 1) give zero advantage for every sample, so there is no learning signal.

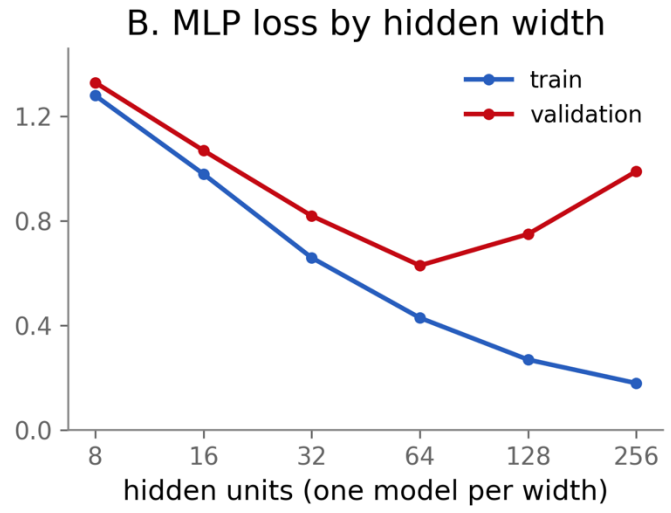
Temperature Logits (ln 9, 0) with $\tau = 2$ give probabilities (¾, ¼).

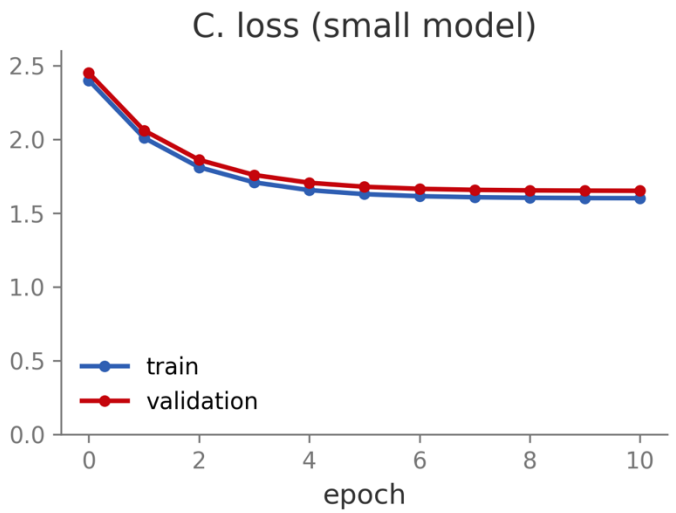
Ben Lengerich © University of Wisconsin-Madison 2026

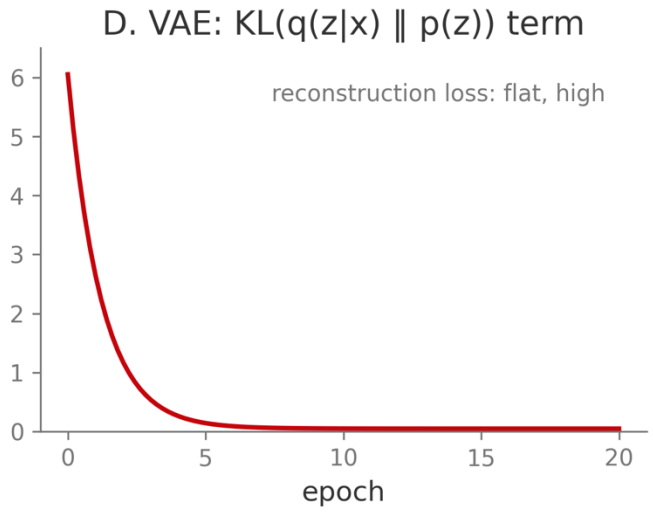
Diagnose the Curves

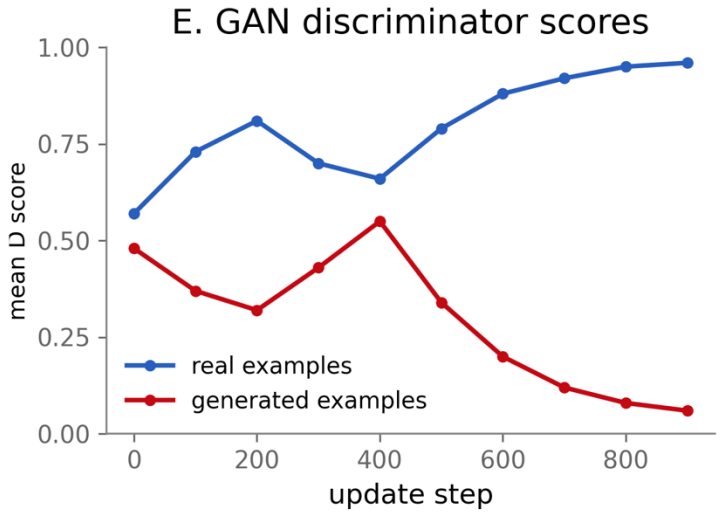
Diagnose each run: what would you change?

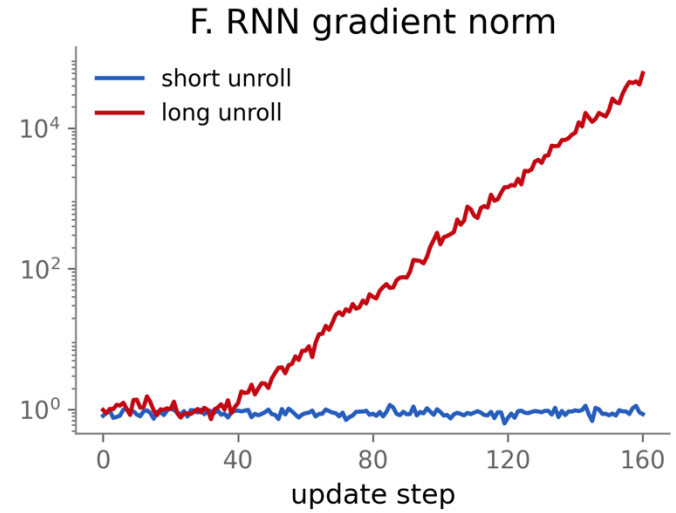












Ben Lengerich © University of Wisconsin-Madison 2026

Diagnose the curves: answers

	What the curves say	What to do
A	Oscillations grow: steps overshoot the minimum	Lower the learning rate
B	Train loss falls as width grows. Validation loss rises past 64 units: overfitting.	Choose 64 units, where validation loss is lowest.
C	Train ≈ validation, both high: underfitting (no gap)	Bigger model or better features, not more regularization
D	KL → 0: $q(z\|x) \approx p(z)$, z carries no information	Posterior collapse: lower β / anneal the KL, weaker decoder
E	D scores approach 1 on real data and 0 on generated data.	Discriminator dominates. Check the balance of D and G updates.
F	With a long unroll, the gradient norm grows by orders of magnitude.	Exploding gradients. Clip the gradient norm.

What does each line measure, and what must be true for it to look like this?

Ben Lengerich © University of Wisconsin-Madison 2026

AI and Society

Every ethical question has a technical side

- **Who is harmed, and how?** Be specific: which people, which decision, which harm
- **Which part of the pipeline causes it?** Data, objective, architecture, training procedure, evaluation, deployment
- **What technical tool reveals or reduces it?** An evaluation, a loss term, a data change, an architecture choice
- Technical fixes are **necessary but not sufficient**: consent, policy, and deployment decisions matter too

Ben Lengerich © University of Wisconsin-Madison 2026

Bias: the average hides the group

Evidence and mechanism

- Training minimizes the **average** loss; a group with 10% of the data gets 10% of the say
- Group A: 90% of data, 5% error. Group B: 10% of data, 40% error
  - Overall error: 0.9·5% + 0.1·40% = **8.5%** (looks fine)
- Gender Shades (2018): commercial face classifiers erred up to **34.7%** on darker-skinned women vs. ≤ **0.8%** on lighter-skinned men

Technical responses

- **Reveal:** report held-out error **per group**, not just overall
- **Reduce:** reweight or rebalance the training data; collect more data for the small group
- **Check the benchmark:** the test set can be skewed too

Buolamwini & Gebru, FAT* 2018

Confident and wrong: hallucination and sycophancy

Harm and mechanism

- **Who is harmed:** people who rely on answers for health, legal, or financial decisions
- **Pretraining:** next-token MLE rewards what is likely, not what is true
- **RLHF:** optimizes what raters prefer; raters (and reward models) sometimes prefer agreeable answers over correct ones (**sycophancy**)
- Goodhart again: the reward model is a proxy

Technical responses

- **RAG:** ground answers in retrieved, citable sources
- **RLVR:** reward answers that can be checked
- **Reward hedging:** penalize confident wrong answers more than "I don't know"
- **Evaluate** truthfulness separately from preference

Sharma et al., ICLR 2024

Privacy: models can memorize their training data

Evidence and mechanism

- **Who is harmed:** people whose personal data was in the training set
- Researchers extracted **hundreds of verbatim training sequences** from GPT-2, including names, phone numbers, and email addresses
- **Mechanism:** MLE must give every training sequence probability; rare or duplicated text can be memorized (overfitting at the level of single examples)
- Fine-tuning on sensitive data (e.g. patient notes) carries the same risk

Technical responses

- **Data:** deduplicate; filter personal information
- **Training:** differential privacy (clip and add noise to per-example gradients)
- **Evaluate:** test whether training examples can be extracted
- **Design:** keep private facts in retrieval with access control, not in the weights

Carlini et al., USENIX Security 2021

Synthetic media: generative models are the point

- **Who is harmed:** people depicted in fake images, audio, or video; anyone deciding what to believe
- GANs and diffusion models are trained to make samples **indistinguishable from real data**
- A deepfake detector is a **discriminator**, and generators are trained to beat discriminators: an arms race
- **Responses:** provenance metadata and watermarks at generation time (both can be stripped); limits on generating real people; detection as one signal, not a guarantee

Ben Lengerich © University of Wisconsin-Madison 2026

More challenges that you(?) will solve

Challenge	Who is harmed	Course connection
Consent and copyright of training data	Writers, artists, data subjects	Pretraining data; distillation from other models' outputs
Energy and compute cost	Communities near data centers; the climate	$C \approx 6ND$ for training; test-time compute multiplies inference cost
Hidden labor	Annotators who label and rank outputs	SFT demonstrations and RLHF comparisons are human work
Removing safety training	Targets of misuse	Cheap fine-tuning (e.g. LoRA) of open weights
Opaque decisions	People affected by automated decisions	Unfaithful chains of thought; interpretability

Ben Lengerich © University of Wisconsin-Madison 2026

Open Problems

Outer and inner alignment

How do probabilistic objectives connect to value-based objectives?

Outer alignment

Does the loss we train on reflect what people want the model to do?

Inner alignment

Does the trained model pursue that goal even outside its training distribution?

Ben Lengerich © University of Wisconsin-Madison 2026

More open problems

Training and adaptation

Scaling RL with delayed rewards

Scaling verifiable rewards

Continual learning

Reasoning and structure

Combining LLMs with symbolic reasoning

Combining LLMs with graphical models

Alignment & interpretability

Formal theory of alignment

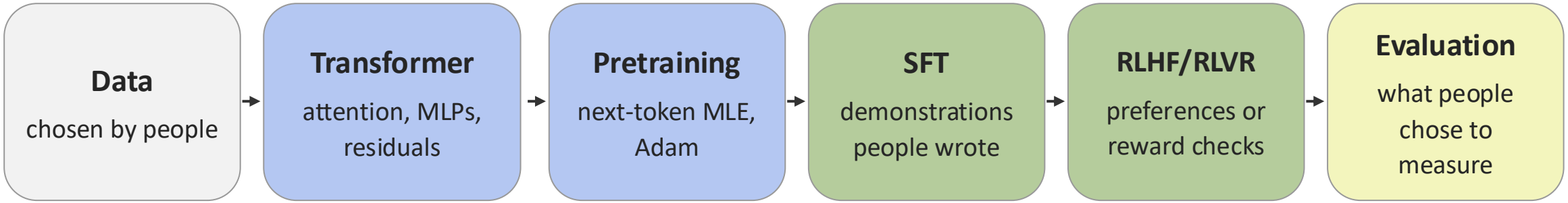
Post-hoc interpretability

Large models that are interpretable by design

Aligning the human–model system

Ben Lengerich © University of Wisconsin-Madison 2026

There's no wizard behind the curtain



You have derived or built every piece of a frontier model in this course.

The next decisions are about scale, data, objectives, whose preferences count, what to measure, and when to ship.

Those decisions are where the ethical questions live.

You are now the people who make them.

Ben Lengerich © University of Wisconsin-Madison 2026

Thank you!

You now know deep generative models.

You have the tools and vocabulary to read, build, and question current AI research.

Final exam

Saturday, December 12, 5:05–7:05 PM, Bardeen 140

Ben Lengerich © University of Wisconsin-Madison 2026

Questions?

