



Introduction to Data Science

Training Feed-Forward Neural Networks

08 Oct 2026

Training Neural Networks

Roadmap

1. Choose the data used for an update

Compare full-batch, stochastic, and minibatch gradient descent.

2. Measure generalization

Separate training, validation, and test data; recognize overfitting.

3. Control overfitting and interpret learning

Use regularization, dropout, and empirical-risk minimization.

Low training error is useful only when the model also works on new data.

Training Neural Networks

Two Different Questions About Learning

CAN WE FIT THE TRAINING DATA?

Did the updates reduce the loss on the examples used for learning?

This is the optimization question.

CAN WE PREDICT NEW DATA?

Does the trained model work on examples not used to fit its weights?

This is the generalization question.

Optimization asks about the training loss; generalization asks about new examples.

Gradient-Descent Variants

Stochastic and Minibatch Gradient Descent

Method	Examples per update	What is computed?
Full batch	All training examples	The complete training gradient.
Stochastic (SGD)	One example	One example's gradient.
Minibatch	A subset of examples	An aggregate of the subset's gradients.

All three methods still update the weights using a gradient-based rule.

The variants differ in how many examples contribute to one update.

Gradient-Descent Variants

A Nearly Flat Region Can Slow the Updates

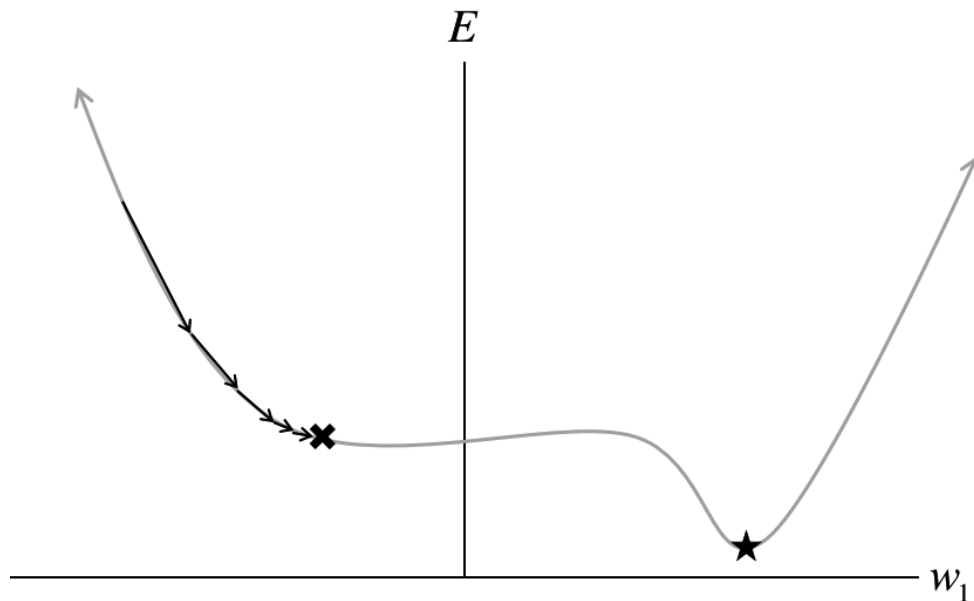


Figure 4-6

The chapter now considers a more complicated loss shape.

Near the flat part, the derivative can be small.

Small derivatives can make the update progress slowly.

A nearly flat region can make gradient-descent updates slow.

Gradient-Descent Variants

Different Examples Suggest Different Local Directions

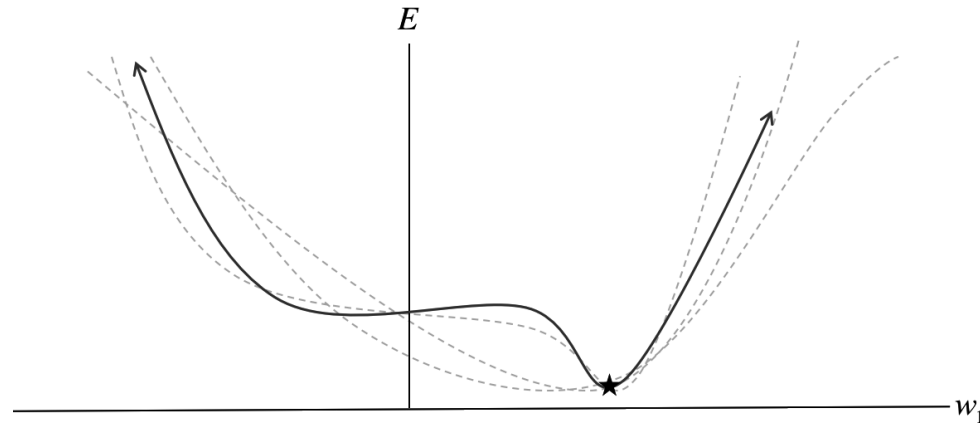


Figure 4-7

A single example need not produce the same loss curve as the full dataset.

Using different examples changes the update direction.

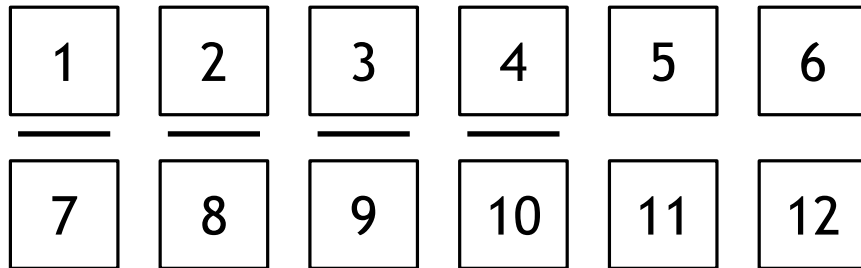
This variation can help movement through some flat regions, but also makes steps noisy.

Stochastic gradients fluctuate because the examples differ.

Gradient-Descent Variants

One Dataset, Different Update Sizes

Twelve training examples



Full batch: use all 12 examples, then update once.

SGD: use one example, then update.

Minibatch of 4: use four examples together, then update.

The underlined cards illustrate one minibatch of four.

A minibatch is a selected subset, not a new kind of training example.

Gradient-Descent Variants

An Epoch Is One Pass Through the Training Set



$$12 \text{ examples} \div 4 \text{ examples per batch} = 3 \text{ updates per epoch}$$

The next epoch makes another pass, commonly in a new shuffled order.

The batch size changes how often weights are updated during that pass.

An update and an epoch are not the same unit of work.

Gradient-Descent Variants

Sum or Mean: Be Consistent

CHAPTER UPDATE FORMULAS

Add the example gradients.

A larger batch may increase the numerical scale of that sum.

EMPIRICAL-RISK FORMULA

Average the example gradients.

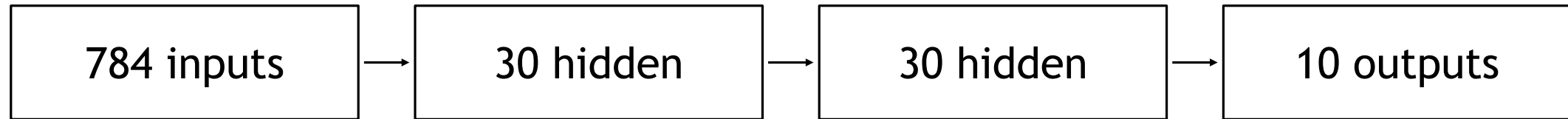
The factor $1/n$ is explicit in the later theory section.

Keep the loss definition and the learning-rate scale consistent.

Changing a gradient sum to a mean changes its scale.

Overfitting

Many Weights Mean Many Adjustable Values



$$784 \times 30 + 30 \times 30 + 30 \times 10 = 24,720 \text{ weights}$$

The 784 inputs come from a 28×28 image.

Including 70 biases gives 24,790 adjustable parameters.

A model can have many parameters even when its layers look small.

Overfitting

Test Sets, Validation Sets, and Overfitting

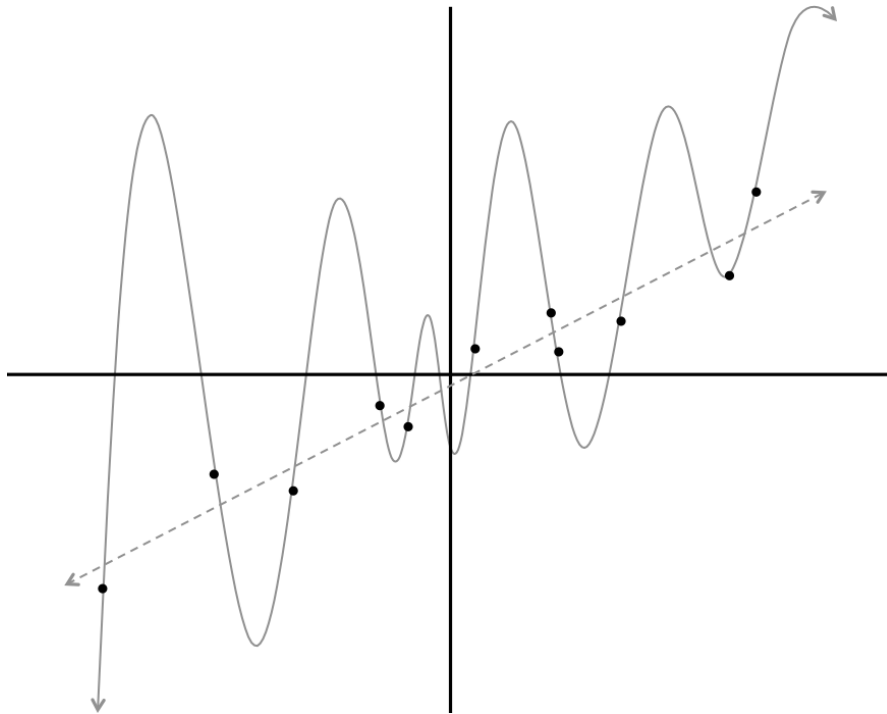


Figure 4-8

The chapter compares a line with a degree-12 polynomial.

The complicated curve fits the training points very closely.

Training fit alone does not tell us which curve will predict new points better.

A curve can fit the observed points closely without predicting new points well.

Overfitting

Add New Points Before Choosing a Model

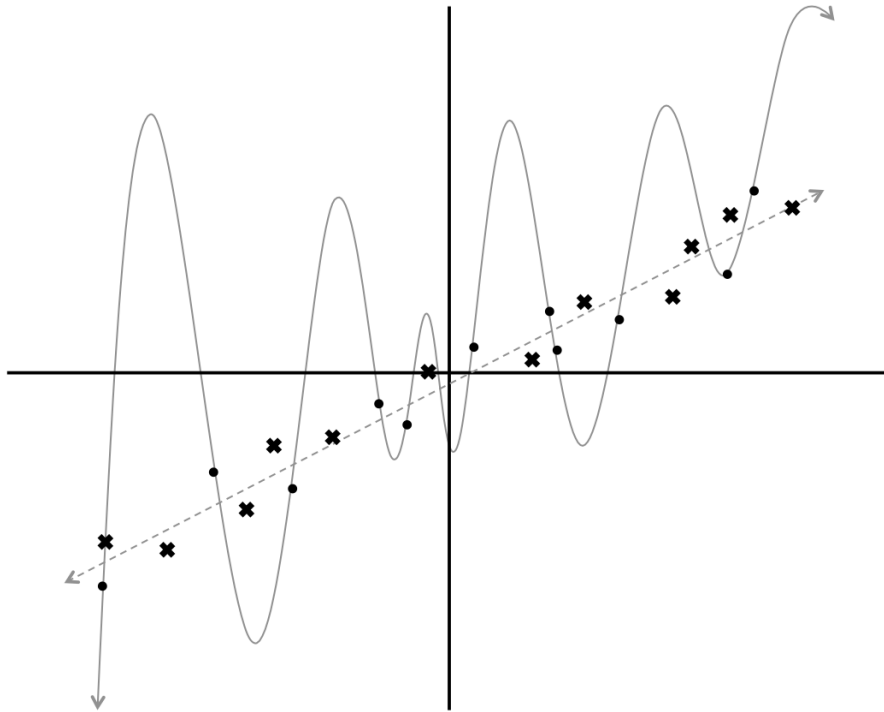


Figure 4-9

The added points provide new evidence.

In this example, the line predicts the new points more accurately.

The high-degree curve has fitted details that do not carry over to new data.

Generalization is tested on examples beyond the points used for fitting.

Overfitting

What Exactly Is Overfitting?

TRAINING DATA

The model adapts to the examples used to update its parameters.

A flexible model can fit even accidental details.

NEW DATA

Those accidental details may not repeat.

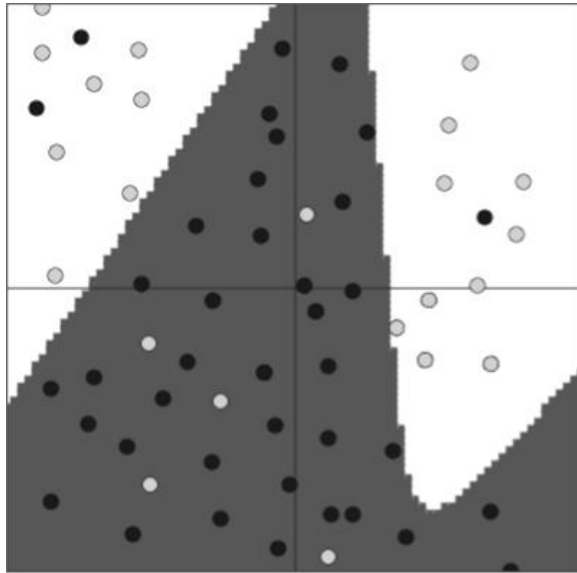
The same model can then perform poorly despite a very small training loss.

Overfitting is a gap between fitting the training examples and generalizing well.

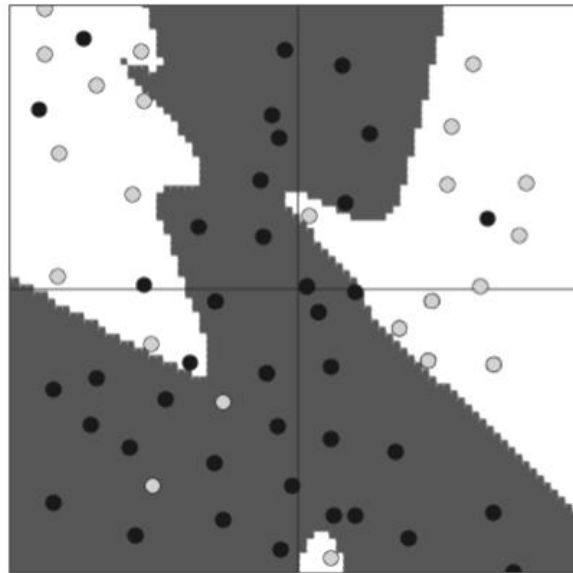
Overfitting

More Hidden Neurons

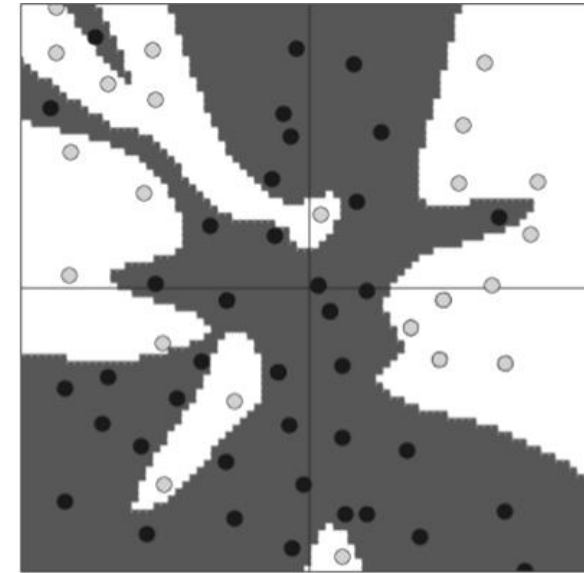
3 hidden neurons



6 hidden neurons



20 hidden neurons



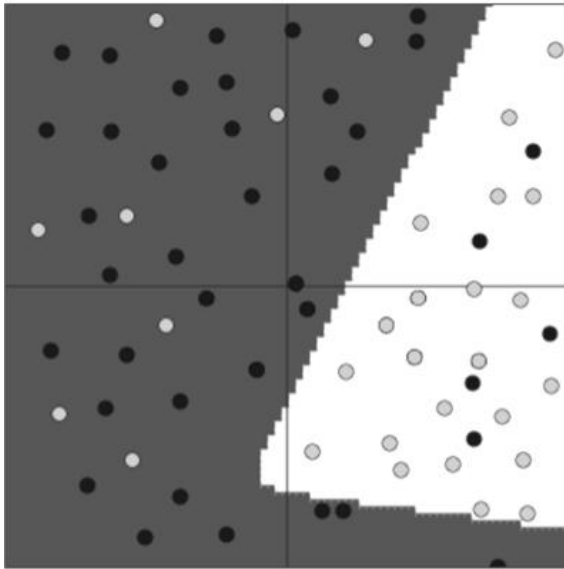
The right-hand boundary can follow smaller local patterns in these examples.

These figures illustrate how model capacity can change a learned boundary.

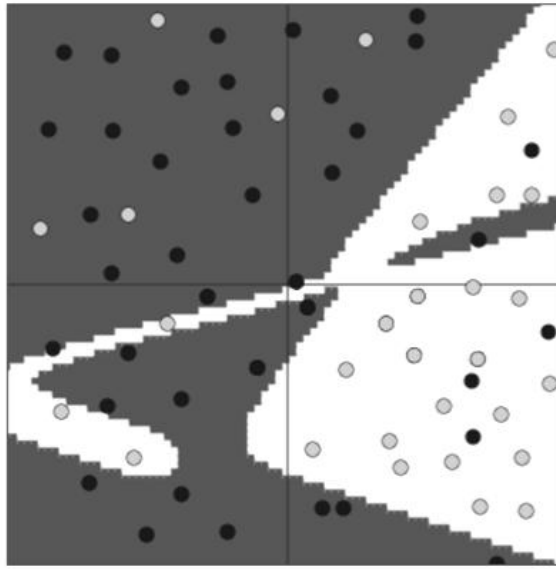
Overfitting

More Hidden Layers: Another Capacity Change

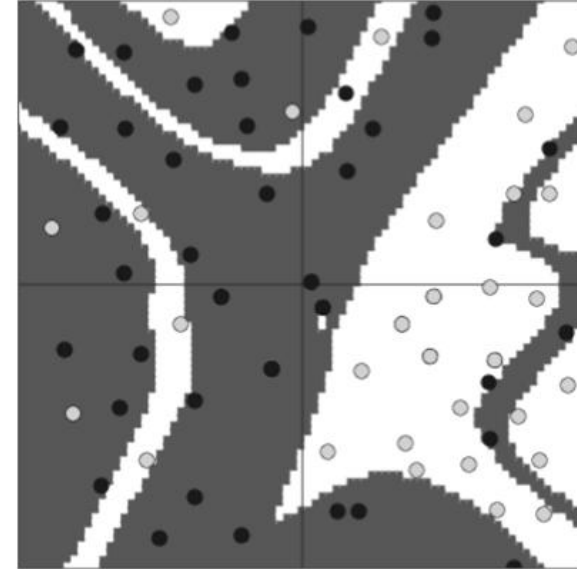
1 hidden layer



2 hidden layers



4 hidden layers



All three examples use three neurons per hidden layer.

Depth is another way to change the functions that a model can represent.

Overfitting

Too Simple, Useful, or Too Flexible for the Data?

TOO SIMPLE

May miss relationships needed for the task.

USEFUL CAPACITY

Can capture patterns that also appear in new data.

HIGH FLEXIBILITY

Can fit accidental details when data are limited.

A larger model is not automatically worse. Evaluate its generalization.

Choose model capacity together with data and measures against overfitting.

Overfitting

Do Not Grade a Model Only on Its Practice Questions



Improving on the examples used for learning is expected.

We also need a separate set to ask whether the learned rule works beyond them.

This is the reason for keeping data outside the training process.

The training set is used to change the model, so its score is not a fresh test.

Data Splits

Separate Training and Test Data

Training	Test
----------	------

Use these examples
to learn weights.

Use these examples
for final evaluation.

The same example should not appear on both sides of this split.

Nonoverlapping datasets give training and evaluation different roles.

Data Splits

Why Add a Validation Set?

Training	Validation	Test
----------	------------	------

Learn the weights.

Choose settings
and stopping time.

Evaluate the final
selected model.

Use validation data to guide choices made during model development.

Data Splits

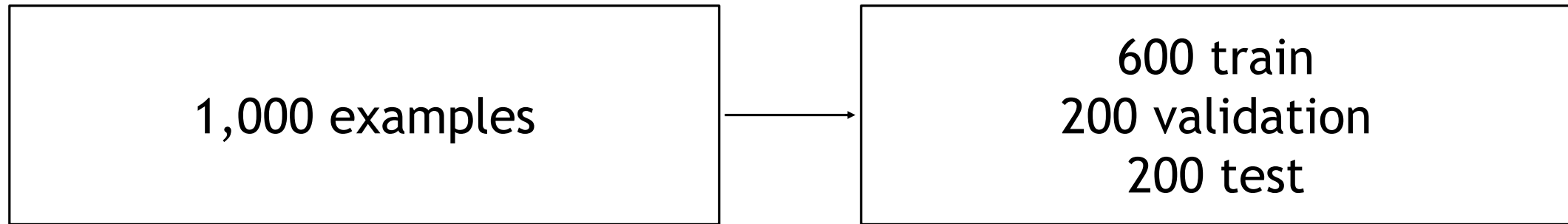
Three Sets, Three Jobs

Set	Used for	Not used for
Training	Gradient updates.	A fresh final evaluation.
Validation	Selecting settings and stopping.	Direct weight-gradient updates.
Test	Final evaluation after selection.	Choosing the winning settings.

Validation selects the model; test evaluates the selected result.

Data Splits

A Concrete Split Example



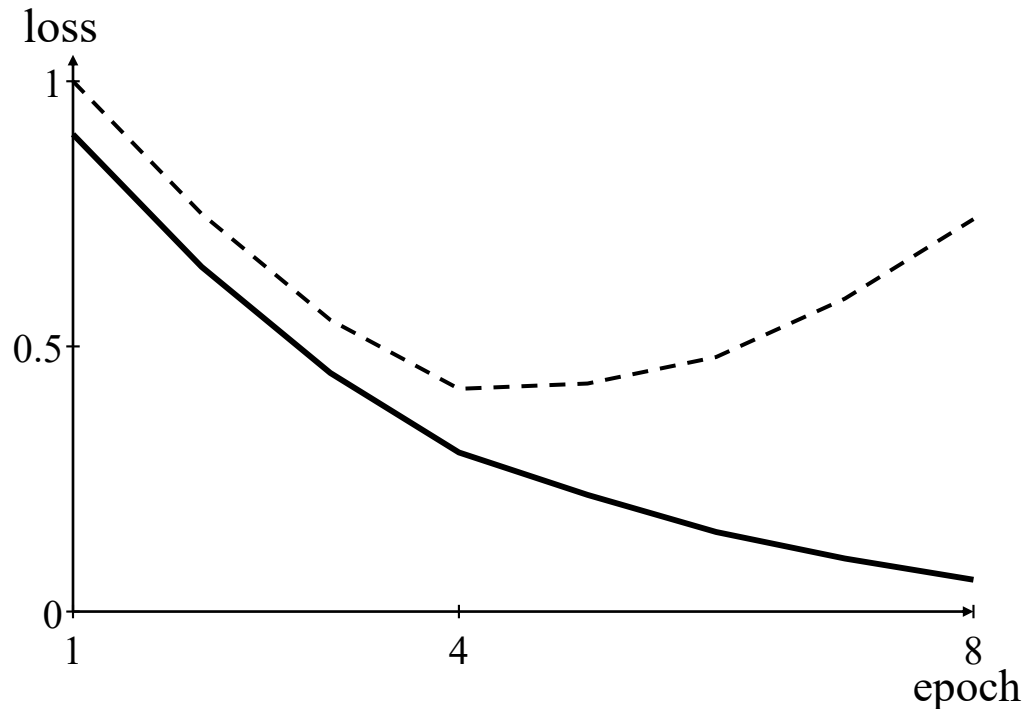
Only the 600 training examples contribute to weight updates.

Validation results help choose the model. Test results are reported after that choice.

The exact split is a design choice; the roles of the three sets must stay clear.

Validation

Read Training and Validation Loss Together



Solid: training loss keeps falling.

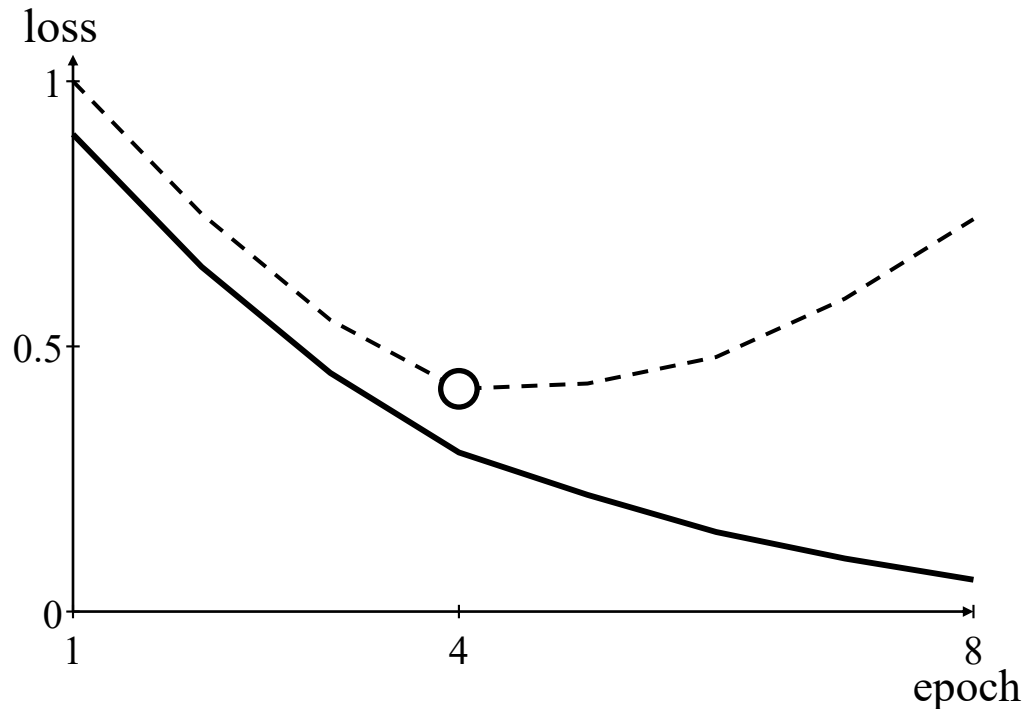
Dashed: validation loss improves at first, then rises.

Later updates fit the training set better but do not improve this validation set.

A decreasing training loss can coexist with a worsening validation loss.

Validation

Early Stopping Uses Validation, Not the Test Set



In this example, epoch 4 has the lowest validation loss.

The model at epoch 8 has a lower training loss but a higher validation loss.

Keep the validation-selected checkpoint for the final evaluation.

Stop selecting later checkpoints when validation no longer improves.

Validation

Different Curves Suggest Different Questions

TRAINING LOSS STAYS HIGH

Can this model fit the task?

Check the representation,
architecture, and optimization setup.

VALIDATION WORSENS

Is the model fitting details that do not
generalize?

Consider the data and measures
against overfitting.

Learning curves guide diagnosis; they do not identify a cause with certainty.

Validation

Hyperparameters Are Chosen Using Validation

Examples of hyperparameters

The learning rate ε , minibatch size, and regularization strength λ .

A validation comparison

Train candidates on the training set and compare them on validation data.

A final evaluation

After choosing the settings, evaluate the selected model on the test set.

Weights are learned; settings such as learning rate and batch size are selected.

Validation

Grid Search: Try Every Listed Combination

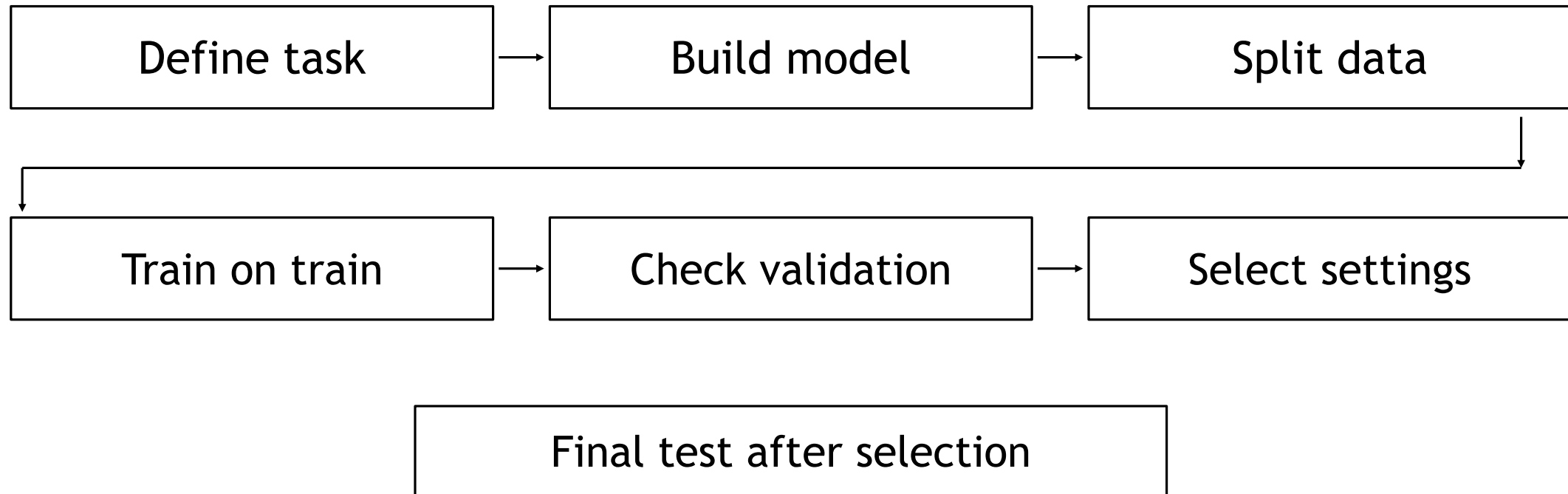
Learning rate	Batch 16	Batch 64	Batch 128
0.001	Train candidate	Train candidate	Train candidate
0.01	Train candidate	Train candidate	Train candidate
0.1	Train candidate	Train candidate	Train candidate

3 learning rates $\times$ 3 batch sizes = 9 candidate settings.

A grid search combines choices from each hyperparameter's candidate list.

Training Workflow

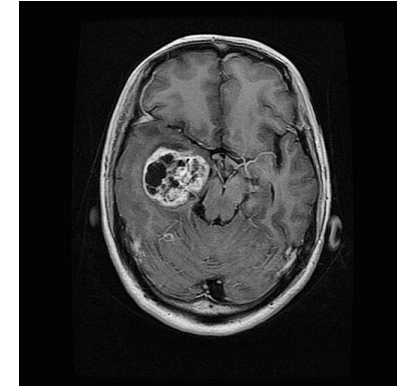
From a Defined Task to a Final Evaluation



Keep the final test outside the model-selection loop.

Training Workflow

Define Inputs and Outputs Before Training



The chapter illustrates an image task with labels: normal, benign tumor, and malignant tumor.

For this example, the architecture needs three output probabilities.

This is an architecture example. We use the class labels only, not medical definitions.

The task determines the representation and the size of the output layer.

Regularization

Preventing Overfitting in Deep Neural Networks

$$\text{New objective} = \text{prediction error} + \lambda f(\theta)$$

The error term rewards accurate predictions on training examples.

The added term penalizes certain parameter settings.

λ controls the strength of that penalty.

Regularization changes what the training objective rewards.

Regularization

The Strength λ Controls the Trade-Off

$\lambda = 0$

No contribution from this penalty term.

MODERATE λ

Fit the data while discouraging large weights.

VERY LARGE λ

The penalty can dominate the prediction error.

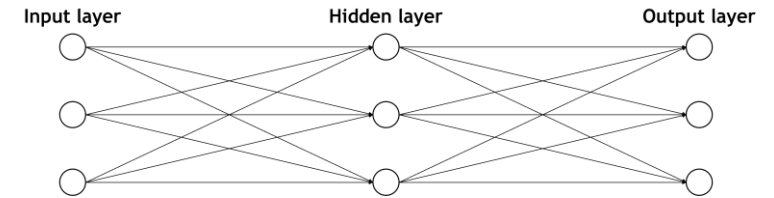
Use validation performance to help choose the strength.

Too much penalty can prevent a model from fitting useful patterns.

Regularization

L2 Regularization Penalizes Squared Weights

$$\text{L2 penalty} = \frac{1}{2}\lambda \sum_k w_k^2$$



Square each weight, add the squares, and multiply by $\frac{1}{2}\lambda$.

A negative weight is penalized just like a positive weight of the same magnitude.

The penalty encourages smaller, more distributed weight values.

Large individual weights contribute strongly to a squared-weight penalty.

Regularization

Compare Two Weight Vectors

Weights	Sum of squares	L2 penalty, $\lambda = 0.1$
(4, 0)	$16 + 0 = 16$	0.8
(2, 2)	$4 + 4 = 8$	0.4

Assume both candidates have the same data error of 0.5.

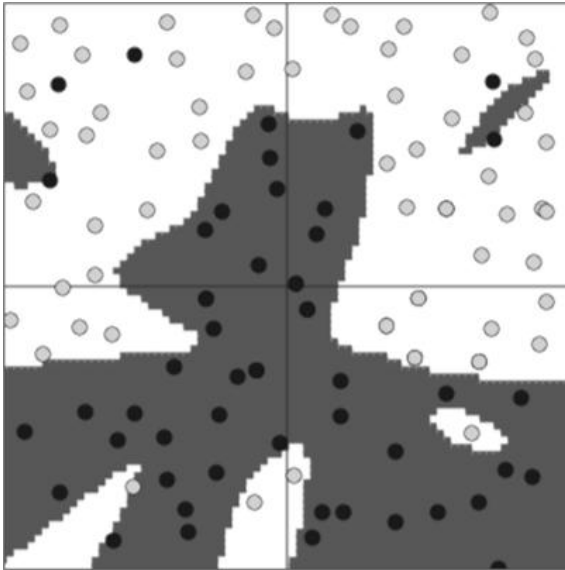
Their total objectives are 1.3 and 0.9, respectively.

For the same prediction error, L2 favors the smaller sum of squared weights.

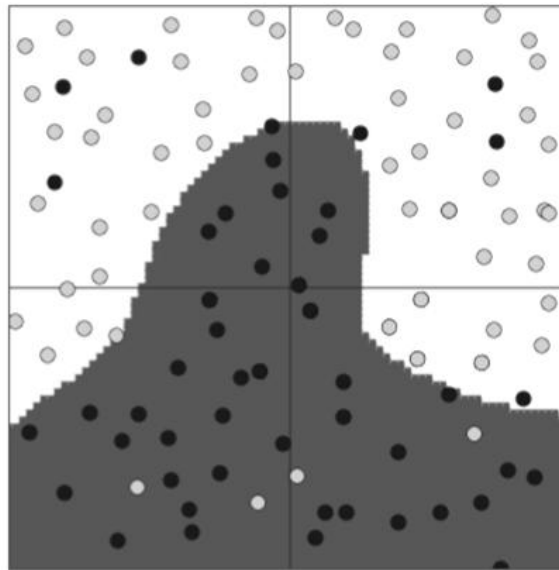
Regularization

Read the Regularization-Strength Examples

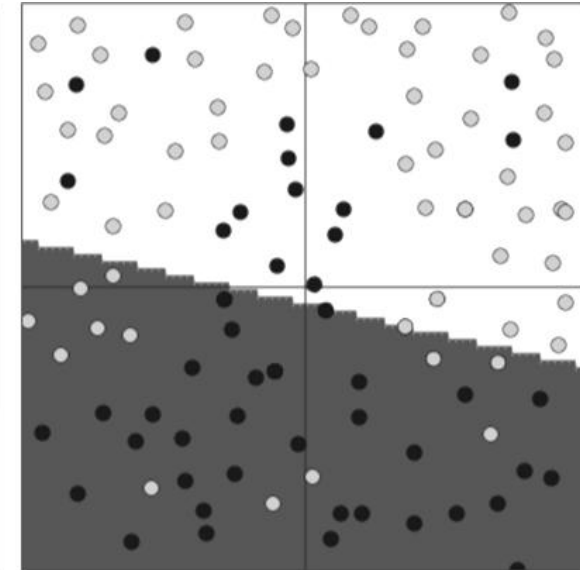
$\lambda = 0.01$



$\lambda = 0.1$



$\lambda = 1$



Here, stronger regularization changes how closely the boundary follows local details.

The chapter varies the penalty while keeping the illustrated network setup fixed.

Regularization

L1 Regularization Uses Absolute Values

$$\text{L1 penalty} = \lambda \sum_k \text{abs}(w_k)$$

$\text{abs}(w)$ means the magnitude of the scalar weight, ignoring its sign.

For weights $(-3, 0, 2)$, the absolute values sum to $3 + 0 + 2 = 5$.

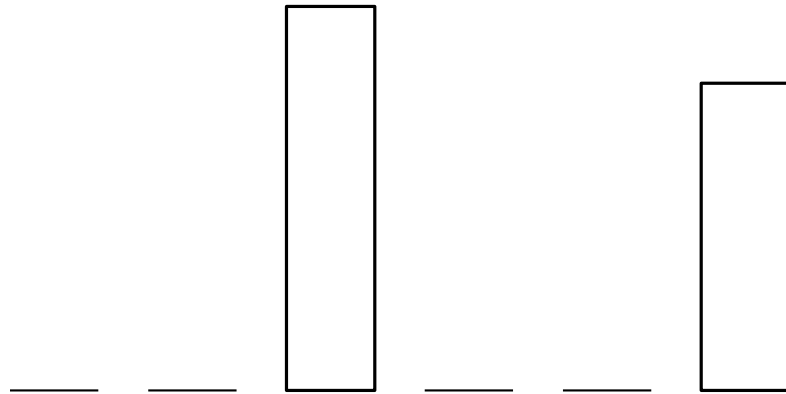
Compared with L2, L1 tends to concentrate the model on fewer inputs.

L1 encourages sparse weight vectors with many values at or near zero.

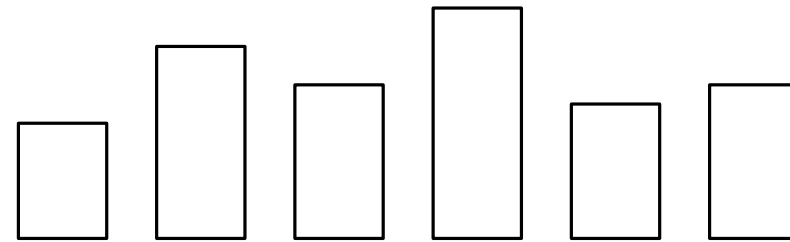
Regularization

L1 and L2 Encourage Different Patterns

SPARSE: L1-LIKE



Dense: L2-LIKE



Sparse and diffuse describe the pattern of weight magnitudes.

Regularization

Max-Norm Constraints Put a Bound on a Weight Vector

$$\text{length}(w) \leq c$$

1. Take a gradient-descent step

The tentative new incoming-weight vector may lie outside the allowed ball.

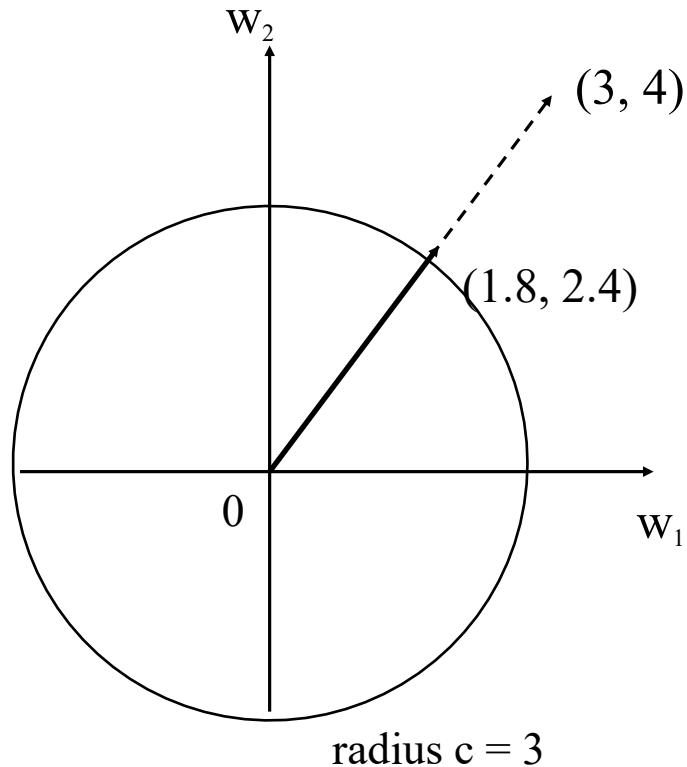
2. Project back if needed

Keep its direction but reduce its length to the maximum c .

A constraint restricts the allowed vector directly rather than only adding a penalty.

Regularization

Max Norm Example



The vector $(3, 4)$ has length 5.

Set the maximum length c to 3.

Multiply by $3/5$ to obtain $(1.8, 2.4)$.

Its length is now exactly 3.

Projection brings an oversized weight vector back to the allowed boundary.

Dropout

Dropout Temporarily Silences Some Activities

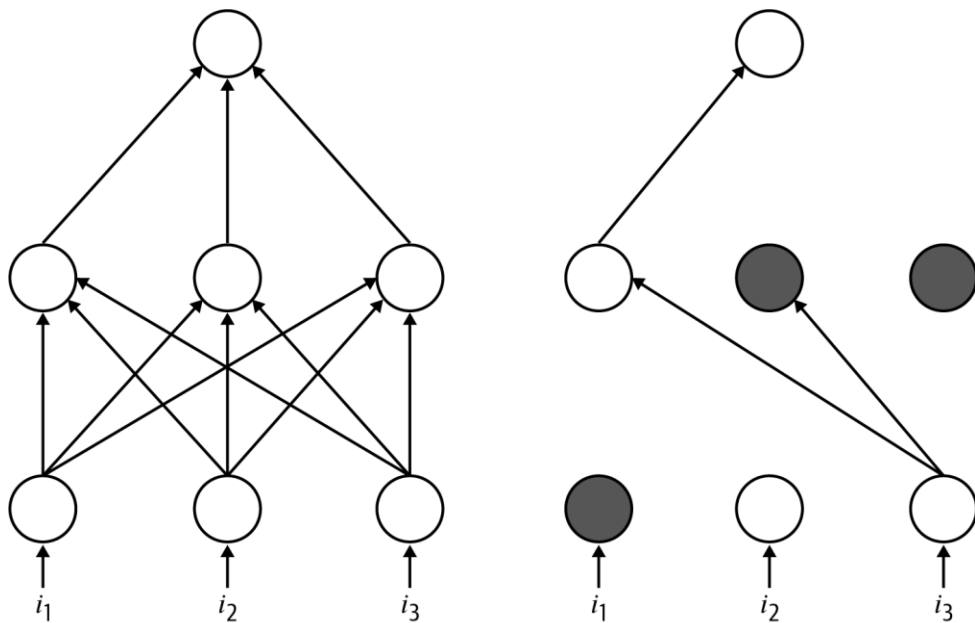


Figure 4-16

During training, a random mask sets some activities to zero.

The surviving activities still pass information onward.

The mask changes across training uses; neurons are not permanently deleted.

Dropout reduces reliance on a fixed set of activities.

Dropout

Be Careful: p Means the Keep Probability Here

KEPT WITH PROBABILITY p

The activity remains available.

For $p = 0.5$, it is kept half the time on average.

DROPPED WITH PROBABILITY $1 - p$

The activity becomes zero.

A particular mask need not keep exactly half the neurons.

Dropout examples used a masking probability.

Dropout

One Random Mask Applied to Four Activities

ACTIVITIES		MASK		MASKED OUTPUT
2		1		2
4		0		0
6	×	1	=	6
8		0		0

A mask value of 0 silences that activity for this use of the network.

Dropout

The Next Mask Can Be Different

Same activities	Mask	Masked output
(2, 4, 6, 8)	(1, 0, 1, 0)	(2, 0, 6, 0)
(2, 4, 6, 8)	(0, 1, 1, 1)	(0, 4, 6, 8)

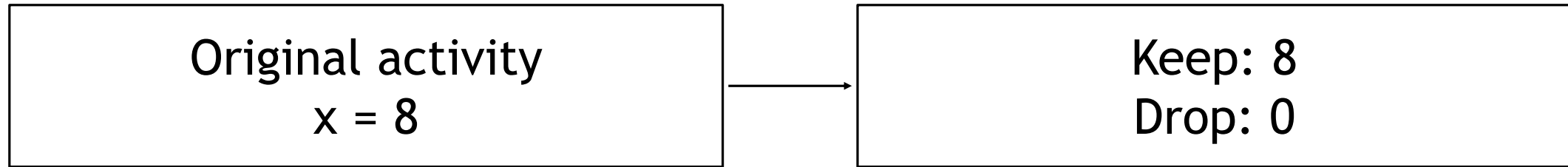
Each use can remove a different set of activities.

The model is encouraged not to depend too strongly on one activity or a small group.

The weights are shared even when different activities are silenced.

Dropout

Without Rescaling, the Average Activity Shrinks

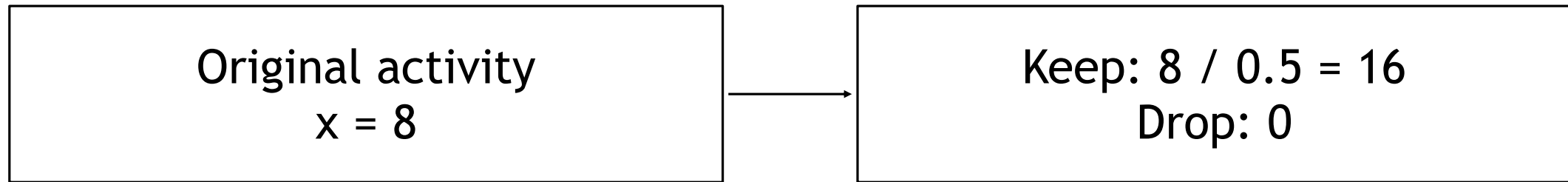


$$p = 0.5: \text{ expected output} = 0.5 \times 8 + 0.5 \times 0 = 4$$

If an activity x survives with probability p , the mean masked output is px .

Dropout

Inverted Dropout Rescales During Training



$$\text{Expected output} = 0.5 \times 16 + 0.5 \times 0 = 8$$

One training output is either 16 or 0. The average is 8.

Divide a surviving activity by p so its expected value stays unchanged.

Dropout

One Vector with Inverted Dropout

Original activities and keep probability

$h = (2, 4, 6, 8)$, with $p = 0.5$.

Apply the mask

Mask $(1, 0, 1, 0)$ gives $(2, 0, 6, 0)$.

Rescale the kept values

Divide by 0.5 to obtain $(4, 0, 12, 0)$.

Apply the mask, then divide the surviving activities by the keep probability.

Dropout

What Happens at Test Time?

TRAINING

Use a random mask.

Divide surviving activities by p .

TEST / INFERENCE

Do not apply the dropout mask.

Do not apply an additional dropout scaling factor.

The expectation calculation concerns each masked activity, not a guarantee of identical predictions.

With inverted dropout, use the ordinary activities at test time.

Learning Theory

Neural Net Learning Theory

The ideal goal

Make predictions with low expected loss on future data.

The available substitute

Use the finite training dataset to approximate that goal.

The practical update

Use sampled examples to estimate the full training gradient.

Minibatches estimate an average gradient using fewer examples.

Learning Theory

Read the Dataset Notation

$X: n \times d$ labels: $n \times 1$ prediction: $f(x; \theta)$

n is the number of training examples. d is the number of input features.

θ collects the model's adjustable parameters.

$L(\text{prediction}, \text{target})$ measures the loss for one example.

The data matrix stores examples in rows and features in columns.

Learning Theory

Population Risk: The Goal Beyond Our Dataset

$$\text{Population risk} = E_{p(x,y)} [L(f(x; \theta), y)]$$

Imagine all possible input-label pairs, weighted by how often they occur.

The expectation averages the model's loss over that distribution.

We do not know the true distribution, so we cannot directly calculate this full average.

Population risk is expected loss under the true data-generating distribution.

Learning Theory

Empirical Risk: Average the Examples We Have

$$\text{Empirical risk} = (1/n) \sum_i L(f(x^{(i)}; \theta), y^{(i)})$$

Four illustrative example losses

0.4, 0.8, 0.2, and 0.6.

Their empirical average

$$(0.4 + 0.8 + 0.2 + 0.6) / 4 = 0.5.$$

Empirical risk replaces the unknown population average with a finite-data average.

Learning Theory

The Full Gradient Is an Average Too

$$\text{Full gradient} = (1/n) \sum_i \nabla_{\theta} L_i$$

Calculate the gradient contribution from each training example.

Average the contributions to obtain the empirical-risk gradient.

With many examples and weights, calculating every contribution at every update can be expensive.

Differentiating a finite average gives the average of its derivatives.

Learning Theory

Why Can One Sample Be a Useful Estimate?

Randomly selected example	Gradient coordinate	Selection probability
1	2	$\frac{1}{4}$
2	-1	$\frac{1}{4}$
3	3	$\frac{1}{4}$
4	0	$\frac{1}{4}$

$$\text{Expected sampled gradient} = (2 - 1 + 3 + 0) / 4 = 1$$

Uniform sampling can give a correct average even when individual estimates differ.

Learning Theory

A Minibatch Averages Several Sampled Contributions

Full training mean

For gradient values $(2, -1, 3, 0)$, the full mean is 1.

One possible minibatch

Choose the first two examples: $(2 - 1) / 2 = 0.5$.

Another possible minibatch

Choose the last two: $(3 + 0) / 2 = 1.5$.

Different minibatches can give different estimates of the same gradient.

Learning Theory

What “Unbiased” Does and Does Not Mean

IT MEANS

With fixed weights and uniform sampling, the expected sampled gradient equals the full training mean.

IT DOES NOT MEAN

Every sampled gradient is exactly correct.

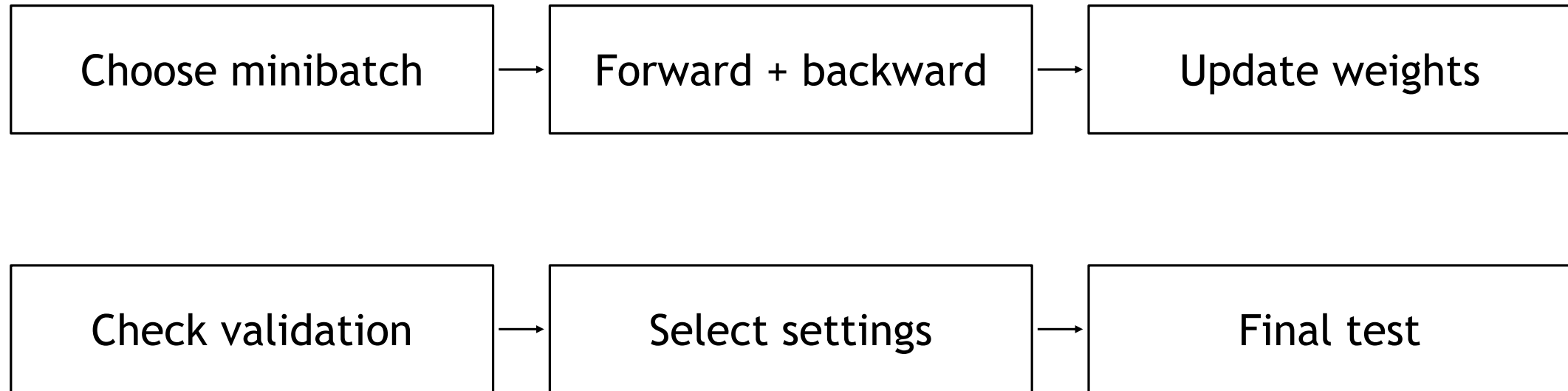
Every update lowers the full loss.

Training performance guarantees future-data performance.

Unbiased describes an expectation, not the accuracy of every individual update.

Summary

Put the Training and Evaluation Pieces Together



Train on one set, select with another, and evaluate the final choice separately.

Summary

Quick Check

1. Batch and epoch

With 12 training examples and a batch size of 4, how many updates form one epoch?

2. Data roles

Which set chooses the learning rate? Which set evaluates the final chosen model?

3. Dropout

For $x = 8$ and keep probability 0.5, what are the inverted-dropout outputs and their mean?

Explain why each training choice is made, not only its name.

Summary

Check: Explanations

1. Three updates

Each update uses four examples; three such batches cover all twelve.

2. Validation, then test

Validation selects settings. Test evaluates the selected model after development.

3. Either 16 or zero, with mean 8

A surviving 8 is divided by 0.5; it is kept with probability 0.5.

The definitions become clearer when attached to a small numerical example.

Summary

What Did We Learn?

Compute practical updates

SGD and minibatches estimate the training gradient from fewer examples.

Check performance on new data

Training, validation, and test sets answer different questions.

Manage overfitting

Regularization, constraints, dropout, and validation-based stopping control the process.

Train to improve predictions, then evaluate generalization.

Summary

Evaluation

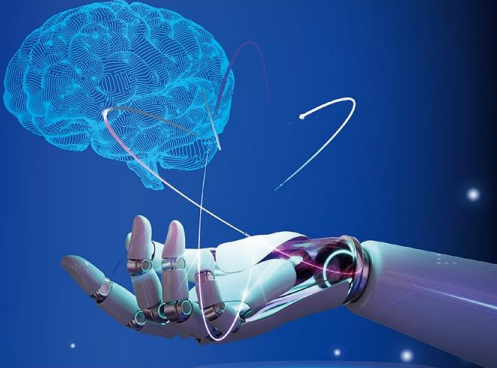
Physical AI Tech Week

- Time: 13 Oct 2026 ~ 14 Oct 2026
- Location: Coex, Seoul
- Participate only a single program
- Bring the name tag or print the picture at Coex
(After entering conference / hall)
- Given 5 points for “Class Participation and Conduct”
- URL: https://event-us.kr/physicalaitech/event/136716?utm_source=homepage&utm_medium=banner&utm_campaign=gtccaylmsk



PHYSICAL AI TECH WEEK

Intelligence into Action - 행동하는 두뇌



일시 : 2026.10.13.(화)~10.14.(수) 장소 : 서울 코엑스 (스튜디오 159, 컨퍼런스룸 E)

피지컬 AI 얼라이언스 데이	피지컬 AI 테크 데이
◆ 개회 및 특별강연 (10:00~11:10) - 크레프톤 이강욱 CAIO - NVIDIA Korea 정소영 대표 - Intel John Healy VP	◆ 개회 및 기조강연 (10:00~12:05) - FriendliAI 전병곤 대표 - MIT 김상배 교수 (Meta, Robotics Architect) - GIST 이규빈 교수
◆ 범부처 사업설명회 (11:10~12:30) - 과기정통부, 중기부, 농식품부, 해수부 등	◆ 기술발표 (13:30~17:30) - LG AI연구원, NC AI, (美) AWS, (佛) AMI Labs, (獨) IDTA 등
◆ 분과 네트워킹 및 간담회 (14:00~16:30)	◆ 피지컬 AI 토크콘서트 (14:00~15:40)

상시 전시 : 피지컬 AI 주요기술 · 기업전시(컨퍼런스룸 E)

과기정통부 | nipa | 정보통신산업진흥원 | PIA | 피지컬AI얼라이언스