Video 27

Sunday, November 22, 2020 5:41 PM

Data augmentation

More data improves generalization

Augment the training set by transforming the examples.

$\Rightarrow$ data augmentation

Transformations depend on the domain

Examples (visual recognition : e.g., object recognition)

- translation
- horizontal / vertical flip $\longrightarrow$ would work for object recognition but not for handwritten digit recognition
- rotation
- noise (e.g., flip random pixels)

⚠ apply these transformations to the training set only (not the test set)

Reduce the number of parameters.

Architectural changes.

Fully connected layers $\longrightarrow$ convolutional layers

decreases the # of parameters.

Weight decay

Regularize a network / model by penalizing large weight values

$$\mathcal{L} = \mathcal{L}_{cross-entropy} + \lambda R$$

- $\mathcal{L}_{cross-entropy}$: measure the error of the model
- λ: hyperparameter
- R: regularization penalty

$$R = \frac{1}{2} \sum_j w_j^2 = \frac{1}{2} \|\vec{w}\|^2$$

Interpreted as weight decay

$$\vec{w} \leftarrow \vec{w} - \alpha \frac{\partial \mathcal{L}}{\partial \vec{w}} = \vec{w} - \alpha \left(\frac{\partial \mathcal{L}_{cross-entropy}}{\partial \vec{w}} + \lambda \frac{\partial R}{\partial \vec{w}} \right)$$

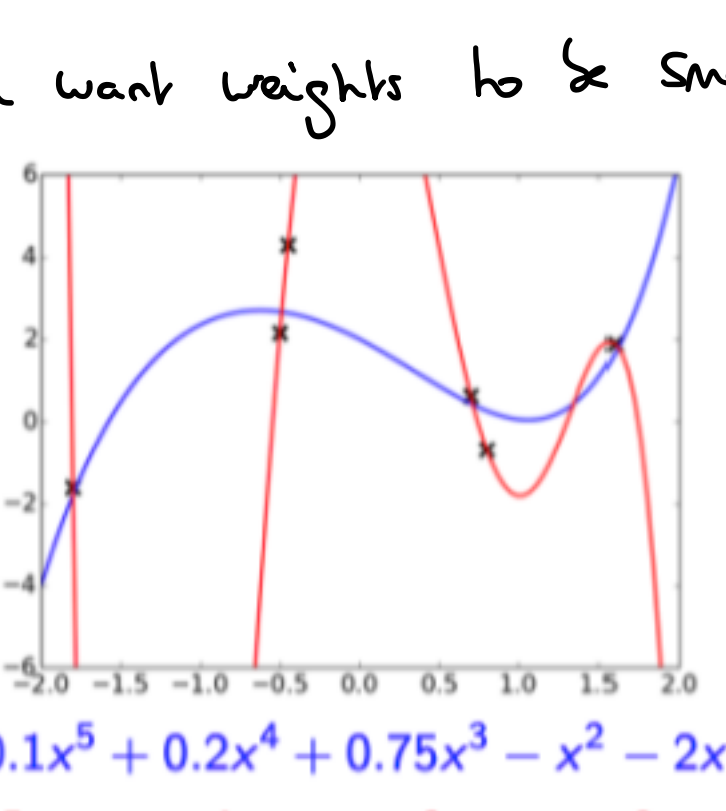
($\frac{\partial R}{\partial \vec{w}} = \vec{w}$)

$$= \vec{w} - \alpha \left(\frac{\partial \mathcal{L}_{cross-entropy}}{\partial \vec{w}} + \lambda \vec{w} \right)$$

$$= (1 - \lambda\alpha)\vec{w} - \alpha \frac{\partial \mathcal{L}_{cross-entropy}}{\partial \vec{w}}$$

$(1 - \lambda\alpha)\vec{w}$: decaying the weight $\vec{w}$

Why do we want weights to be small?



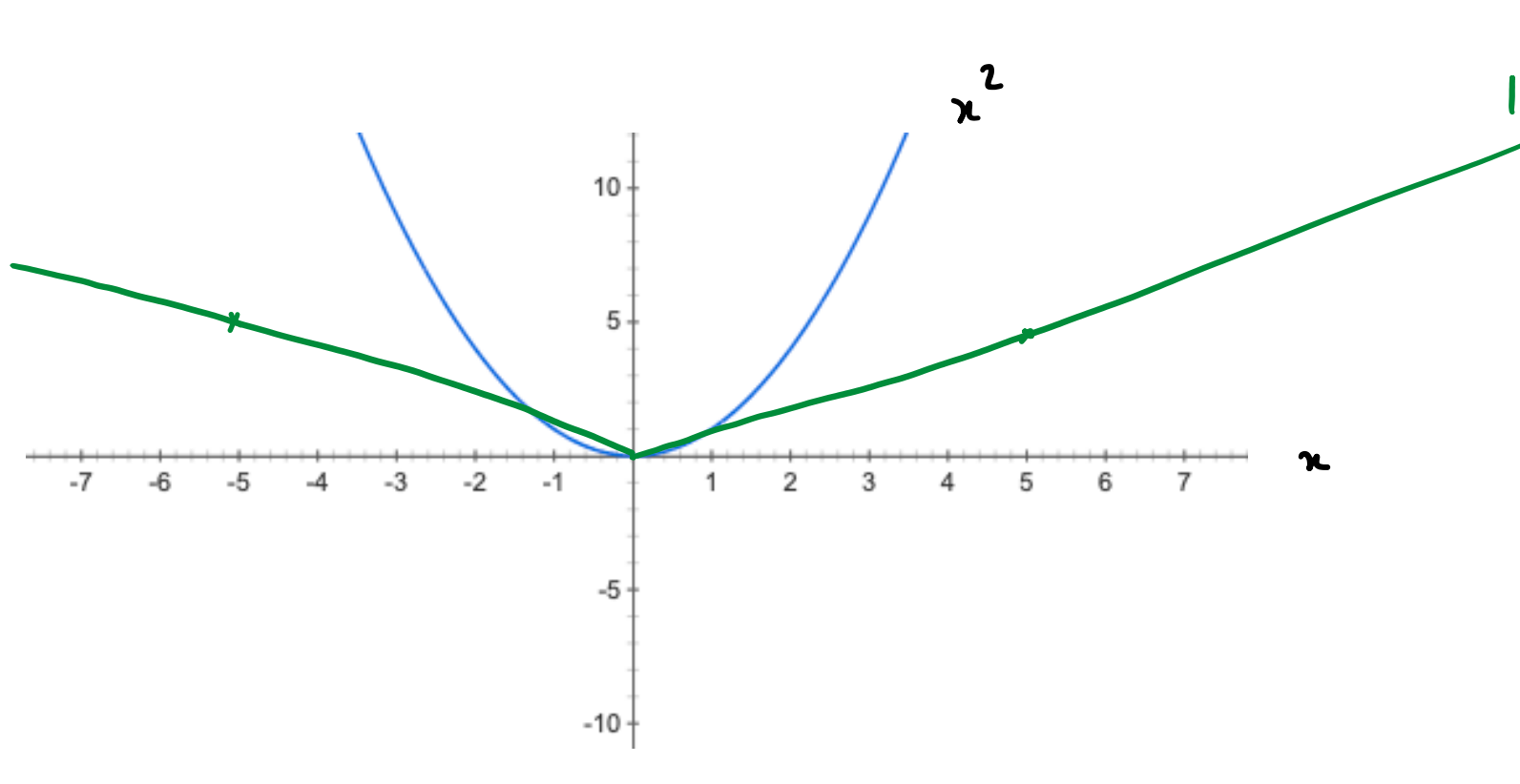
$y = 0.1x^5 + 0.2x^4 + 0.75x^3 - x^2 - 2x + 2$

$y = -7.2x^5 + 10.4x^4 + 24.5x^3 - 37.9x^2 - 3.6x + 12$ } overfits & has large coefficients / weights

l_1 vs l_2 regularization.

$$R = \|\vec{w}\|_2^2 = w_1^2 + w_2^2 + \dots + w_n^2$$

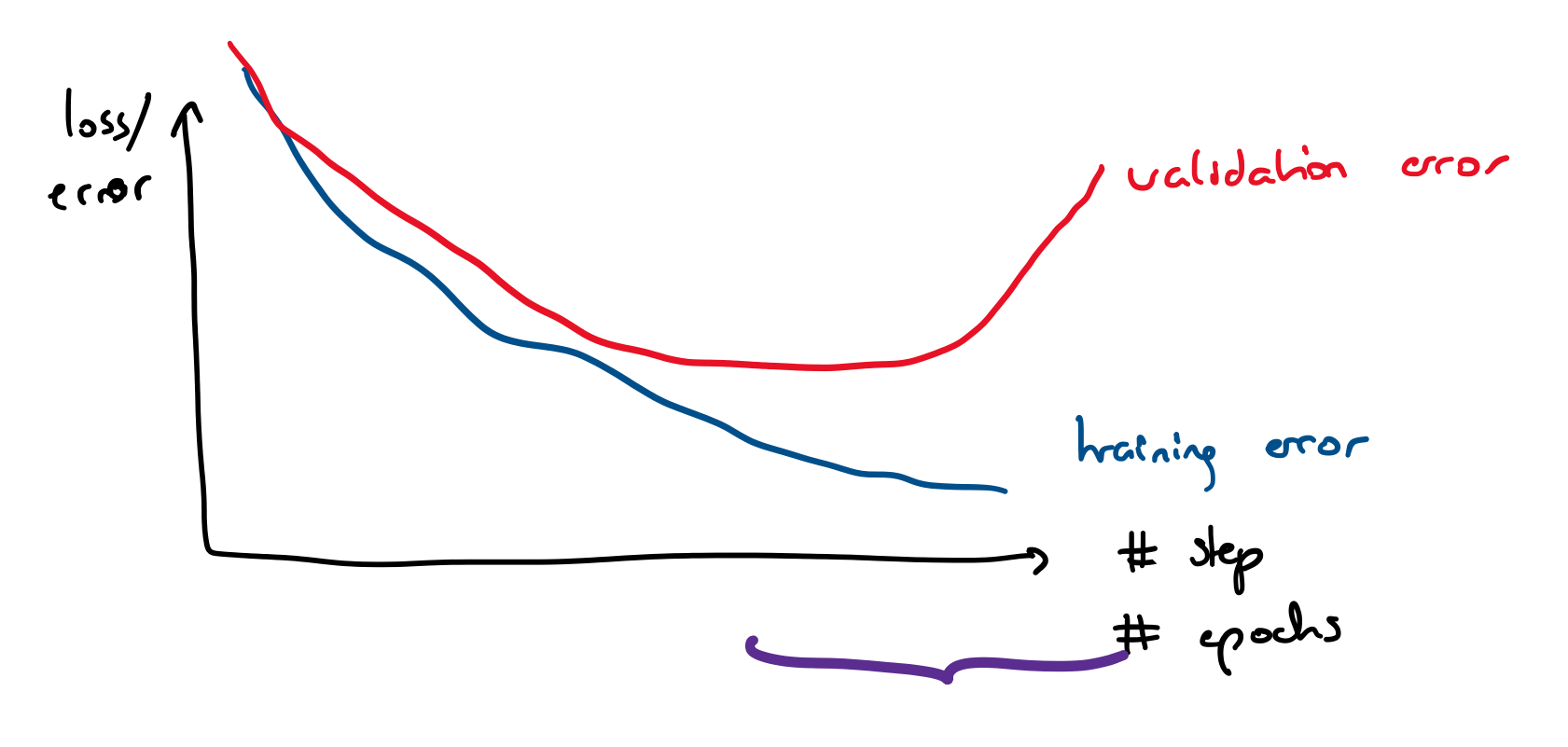
$$R = |\vec{w}| = |w_1| + |w_2| + \dots + |w_n|$$



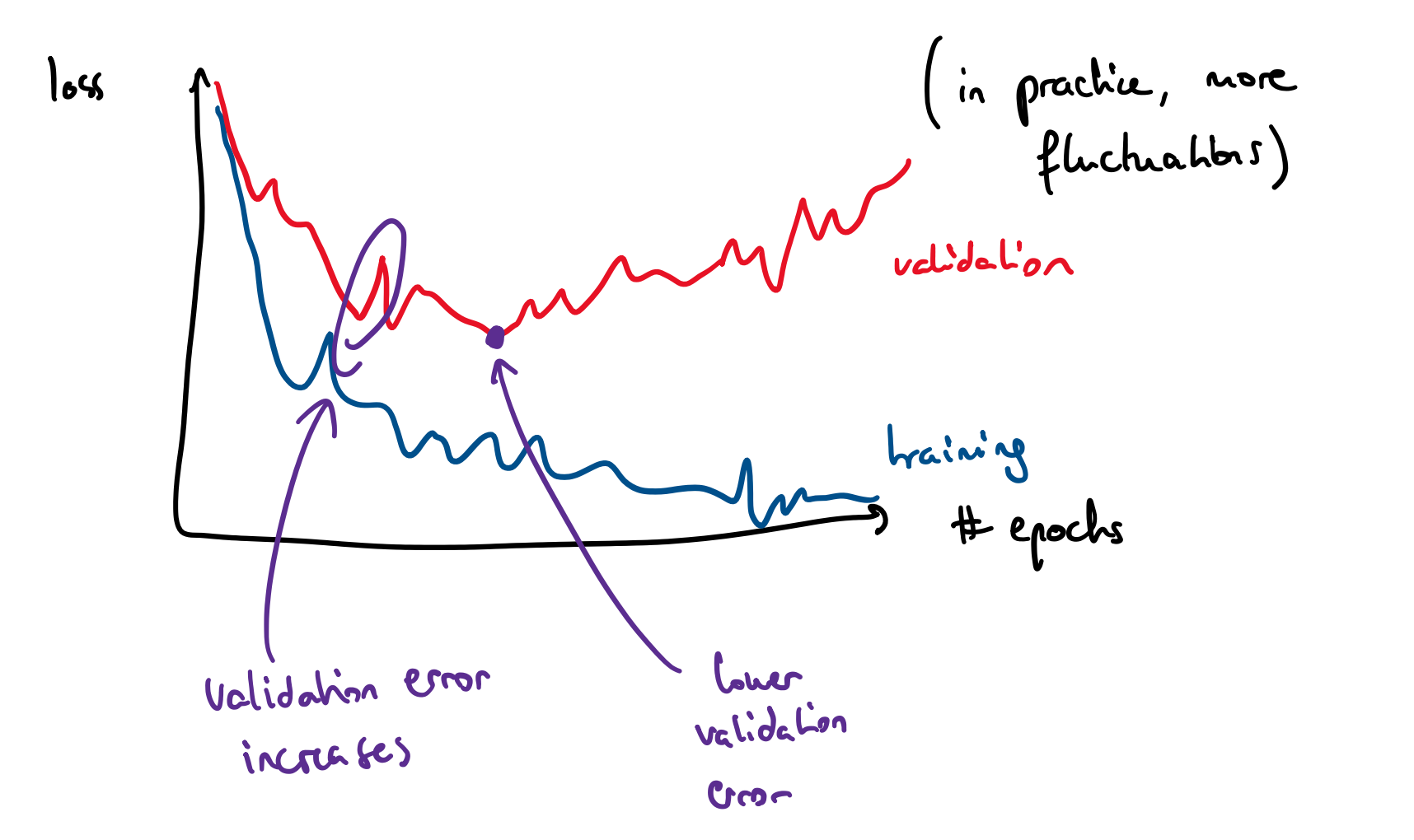
l_2 penalizes large values $\longrightarrow$ closed form solution

l_1 penalizes smaller values $\longrightarrow$ encourage sparsity (have as many weight values set to 0).

Early stopping



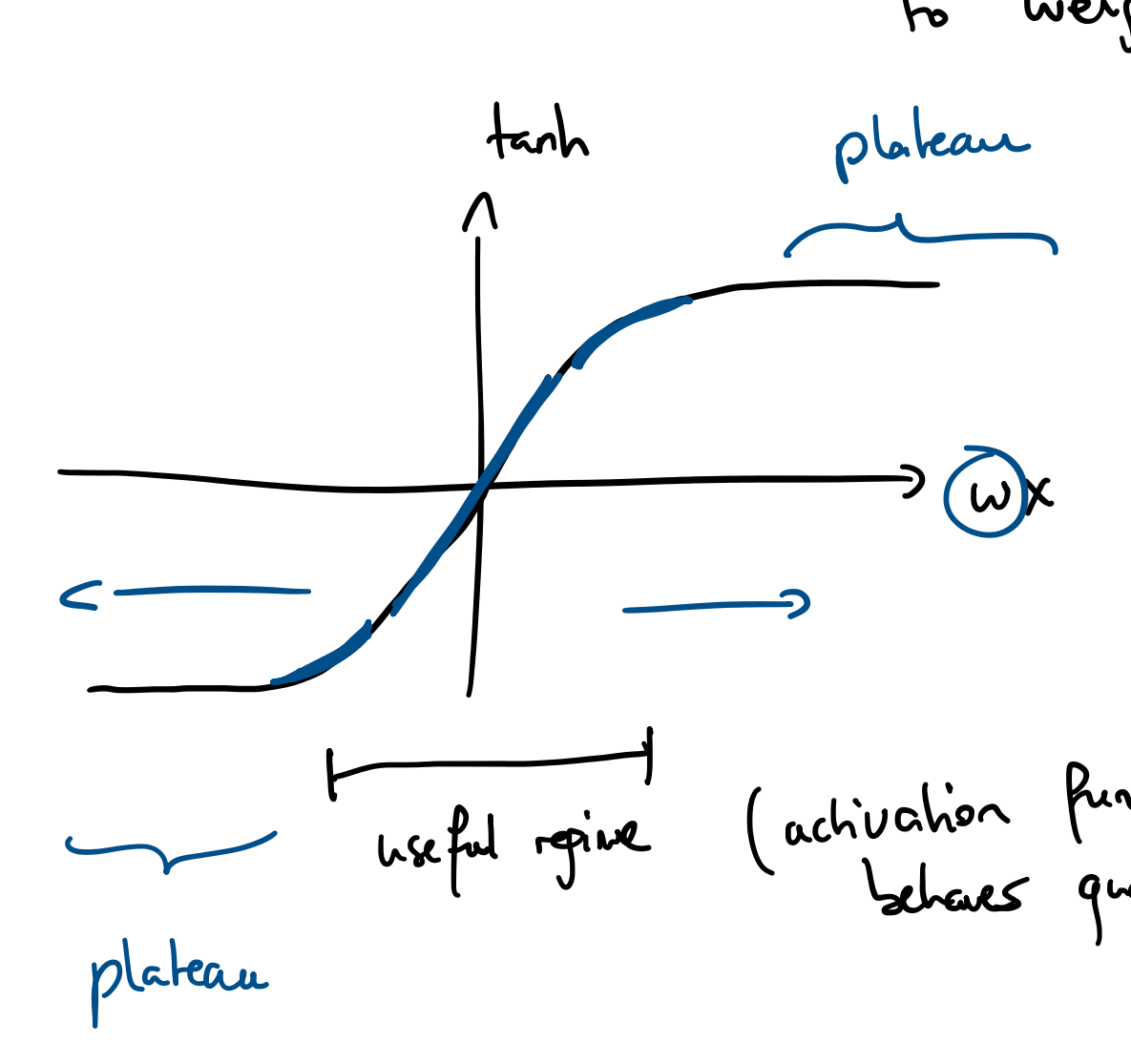
Monitoring validation error & stopping training when it starts to increase.



Patience: use a sliding window of a few epochs to decide when to stop.

Why does early stopping work:

- weights start w/ small values. as we train, they take larger & larger values
  - ↪ early stopping has a similar effect to weight decay.



Ensembles

- If the loss is convex wrt predictions of the model.
  - ↪ squared error, cross-entropy

  true for neural networks which are non-convex wrt. weights.
- If we have a few models & don't know which one is best, we should average their predictions.

$$\mathcal{L}(\lambda_1 y_1 + \dots + \lambda_N y_N, t) \leq \lambda_1 \mathcal{L}(y_1, t) + \dots + \lambda_N \mathcal{L}(y_N, t)$$

for $\lambda_i \geq 0$, $\sum_i \lambda_i = 1$

- N models making predictions y_N
- t: target
- left side: loss of the ensemble
- right side: average loss of each individual model.

Some examples of ensembles:

- training multiple models w/ ≠ random initializations.
- training ——— on ≠ subsets of the training data (bagging)
- training ——— w/ ≠ architectures / hyperparameters / algorithms

Can improve generalization quite a bit.

Downside: increased computational cost.

Dropout

Inject noise in the model's computation: stochastic regularization

$$h_i = \begin{cases} \phi(z_i) & \text{w/ probability } 1-\rho \\ 0 & \text{——— } \rho \end{cases}$$

- ϕ: activation function
- z_i: layer computation e.g., linear wx

ρ is a hyperparameter.

Equivalent to $h_i = m_i \cdot \phi(z_i)$

m_i: mask (Bernoulli random variable) independent for each unit / neuron

$\Rightarrow$ Dropout is like training an ensemble of 2^D different architectures (where D is # of units)

At test time:

- in practice, most disable dropout. multiply all weights by $1-\rho$ obtain the expectation
- run the neural network multiple times independently w/ ≠ dropout masks
  - ↳ average the predictions.
  - ↳ variance estimates the uncertainty of the prediction.