Final Review

CS 4277: Deep Learning

8 Measuring Performance

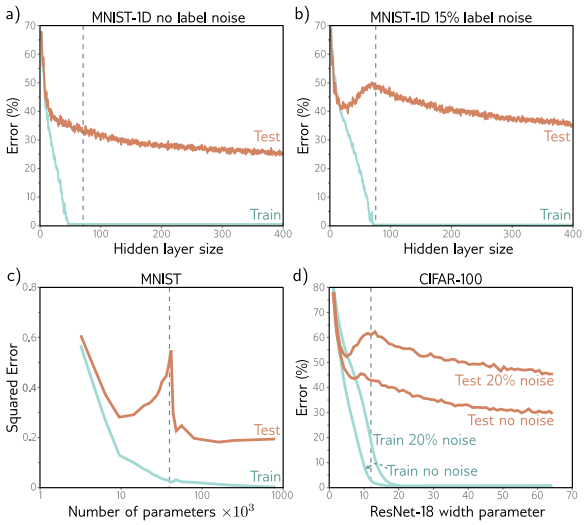
1. * Describe the three principle sources of errors that lead to poor generalization in machine learning and how they can be reduced. (8.2-8.3)

> **Solution:**
>
> 1. Noise may arise because there is a genuine stochastic element to the data generation process, because some of the data are mislabeled, or because there are further explanatory variables that were not observed. Noise is usually a fundamental limitation that cannot be mitigated.
> 2. Bias occurs when the model is not flexible enough to fit the true function perfectly, e.g., a single line cannot represent a sinusoidal function well. We can reduce the bias by making the model more flexible, i.e., increasing its capacity.
> 3. Variance occurs in the particular sample of the data we have in our training set. Another training set drawn from the same underlying function may be different. Variance may also arise from stochastic training algorithms that do not converge to the same model each time they are trained on the same data. We can reduce variance by increasing the quantity of training data.

2. * Describe the bias-variance tradeoff. (8.3.3)

> **Solution:** For a fixed-size training data set, as the model capacity increases the the variance increases and the test error does not decrease, or even increases. With more flexibility the model is able to fit the noise in the training data, leading to overfitting – lower training error with plateauing or increasing test error.

3. * Describe the double-descent phenomenon in deep neural networks. (8.4)



Solution: Panels b, c and d show double descent. For a fixed training/test data set and training procedure, as the model capacity increases the test error reaches nearly zero. For some data, the test error also continues to decrease, but more slowly as the model capacity exceeds the training data. But for may data sets, such as those with label noise, the test error increases as model capacity approaches the point where the capacity equals the number of training examples – as the bias-variance tradeoff predicts. But with increasing model capacity the test error begins to decrease again to below the original pre-overfitting test error.

4. What is the typical approach to choosing hyperparameters? (8.5)

Solution: Empirically, by training with different hyperparameters and testing on a validation set, which is separate from the training and test data.

9 Regularization

5. * What is the goal of regularization?

Solution: To reduce the generalization gap between training and test performance.

6. * What is the standard approach to explicit regularization? (9.1)

Solution: Introducing terms to the loss function that favor certain parameter choices by penalizing other parameter choices.

7. Describe L2 regularization. (9.1.2)

Solution: The most common regularization term, the *L2 norm*, penalizes the sum of the squares of the parameter values:

$$\hat{\phi} = \underset{\phi}{\operatorname{argmin}} \Big(\sum_{i=1}^{I} \ell_i(\boldsymbol{x}_i, \boldsymbol{y}_i) + \lambda \sum_{j} \phi_j^2 \Big) \qquad \text{(Equation 9.5)}$$

This is also referred to as *Tikhonov regularization* or *ridge regression*, or (when applied to matrices) *Frobenius norm regularization.*

8. How is implicit regularization accomplished by SGD? (9.2.2)

Solution: SGD adds noise to the gradient descent because the gradient will be different for different batches. This has the effect of smoothing out the learned function.

9. List 3 heuristic methods of implicit regularization. (9.3)

Solution:

1. Early stopping: stopping training before reaching convergence.
2. Ensembling: train several models and average their predictions.
3. Dropout: drop a random subset of units to 0 at each iteration of SGD, which encourages smaller weights and reduces "kinks" in the learned function.
4. Adding noise to the input data, which smooths out the learned function. Extreme variant: *adversarial training*, which uses an optimization algorithm to find small perturbations in the input data that cause large changes to the output.

10. * Consider a model where the prior distribution over the parameters is a normal distribution with mean zero and variance σ_ϕ^2 so that

$$Pr(\phi) = \prod_{i=1}^{J} \text{Norm}_{\phi_j}(0, \sigma_\phi^2)$$

where j indexes the model parameters. When we apply a prior, we maximize $\prod_{i=1}^{I} Pr(\boldsymbol{y}_i|\boldsymbol{x}_i, \phi)Pr(\phi)$. The associated loss function of this model is equivalent to which regularization technique?

Solution: L2 norm: $\hat{\phi} = \text{argmin}_\phi \left(\sum_{i=1}^{I} \ell_i(\boldsymbol{x}_i, \boldsymbol{y}_i) + \lambda \sum_j \phi_j^2 \right)$

10 Convolutional Networks

11. What is invariance? (10.1)

> **Solution:** $f(T(x)) = f(x)$, i.e., the output of the function $f(x)$ is the same regardless of the application of tranformaction $t(x)$ to the input. For example, a CNN should classifiy a picture as containing a dog even if we translate the position of the dog within the image.

12. What is equivariance? (10.1)

> **Solution:** $f(T(x_i)) = T(f(x_i))$, i.e., the output of the tranformation of the function output is the same as applying the function to transformed input. For example, per-pixel image segmentation should be equivariant to translation.

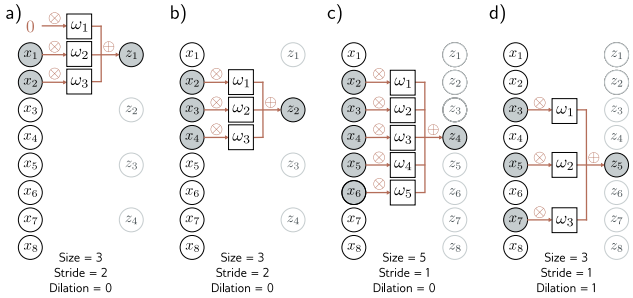
13. * What properties of images make convolutional neural networks well-suited to them?

> **Solution:**
>
> - Images are high-dimensional, leading to a need to reduce the number of parameters compared to a fully-connected network.
> - Nearby pixels are statistically related, so weights can be shared.
> - The interpretation of an image is stable under geometric transformation, e.g., a picture of a dog is a picture of a dog no matter where in the image the dog is located.

14. * What is the motivation for convolutional layers in a neural network?

> **Solution:** They use fewer parameters than fully connected layers, exploit the spatial relationships between nearby pixels, and don't have to re-learn the interpretation of the pixels at every position.

15. * Write out the equation for the 1D dilated convolution with a kernel size of three and a dilation rate of two, as pictured in Figure 10.3d (reproduced below).



Solution:

$$z_i = \omega_1 x_{i-2} + \omega_2 x_i + \omega_3 x_{i+2}$$

16. * T/F The convolution operation is equivariant to translation.

Solution: True

17. T/F The convolution operation is invariant to translation.

Solution: False

18. * Consider a 1D convolutional layer computed using a kernel size of three and has four channels. How many weights and biases are needed for this convolutional layer?

Solution: $3 \times 4 \times 3 = 36$ weights and 4 biases

19. Describe three methods of downsampling. (10.4.1)

Solution:

1. Applying a stride of two effectively downsamples by a factor of 2.
2. Max pooling retains the maximum of $d \times d$ input values.
3. Mean or average pooling averages the inputs.

20. Describe four methods of upsampling. (10.4.2)

Solution:

1. Duplicate channels at each spatial position. For example, duplicate each channel 4 times to double the size.
2. Max unpooling.
3. Bilinear interpolation.
4. Transposed convolution using the transpose of the weight matrix for the downsampling method.

11 Residual Networks

21. * Describe the shattered gradients problem in deep networks. (11.1.1)

Solution: Adding more layers beyond a point (¿ 20ish layers) decreases performance because small changes in the input lead to completely different gradients. In shallow network nearby gradients are correlated, but the correlatoin of nearby gradients quickly drops to zero for deep networks.

22. * What is a residual, a.k.a., skip, connection? (11.2)

Solution: A branch in the computational path whereby the input to each layer is added back to the output.

23. What is the typical order of operations in a residual block? (11.2 - 11.2.1)

Solution: Typically the activation function is applied before the linear transformation, and the residual connection is from before the activation to after the linear transformation. In practice residual blocks contain several such activation-transofrmation layers with a single residual connection from start to end.

24. * Describe the problem of exploding gradients in residual networks. (11.3)

Solution: Recombining the input with the output in a residual connection doubles the variance, growing exponentially with the number of residual blocks. With enough residual blocks, floating point precision can be exceeeded in the forward and backward passes of the backpropagation algorithm.

25. * What is batch normalization and why is it used? (11.4)

Solution: *Batch normalization* or *BatchNorm* shifts and rescales each activation h so that its mean and variance across the batch $\mathcal{B}$ become values that are learned during training. Batch normalization is used to stabilize the forward and backward passes of backpropagation in residual networks – it counteracts the exploding gradients problem.

26. What is the chief drawback of batch normalization and what are its advantages? (11.4.1)

Solution: Disadvantages: Batch normalizatoin adds two extra parameters, γ and σ, at each hidden unit and redundancy in the weights and biases, which decreases efficiency. Advantages:

- Stabilizes forward propagation.
- Enables higher learning rates due to smoother error surfaces.
- Implicitly regularizes by injecting noise into the training process.

12 Transformers

27. * What are the two primary design goals acheived by dot-product self-attention in a language model? (12.2)

Solution:

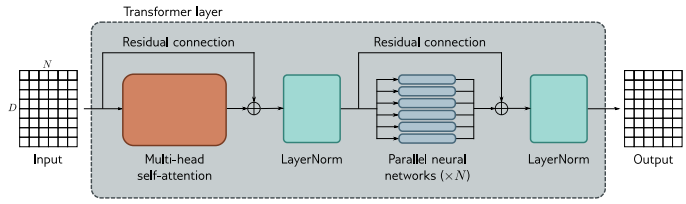
1. Uses parameter sharing to cope with long input passages of differeing lengths.
2. Contains connections between word representations that depend on the words themselves.

28. * Why is positional encoding used in language models? (12.3.1)

Solution: Self-attention by itself is equivariant to permuting word order, but word order is important in language.

29. What are the typical internal sub-layers of a transformer layer? (12.4)

Solution:



30. * What is Tokenization? (12.5.1)

Solution: Turning a sequence of input letters into a sequence of tokens from a vocabulary of possible tokens. The vocabulary and the tokenization is learned, and tokens are typically *sub-word*, such as byte pairs.

31. * Embeddings (12.5.2)

Solution: Every token in the vocabulary is mapped to a D-dimensional vector, typically 1024. The mapping is learned.

32. * Encoders and decoders. (12.6)

Solution: An encoder transforms the text embeddings into a representation that can support a variety of tasks. A decoder predicts the next token in a sequence. Encoder-decoders are used in sequence-to-sequence tasks, where one text string is converted into another (e.g., machine translation).

33. * Pre-training (12.6.1)

Solution: In the pre-training stage, the network is trained using self-supervision. This allows the use of enormous amounts of data without the need for manual labels. Typically the self-supervision task consists of predicting missing words from sentences from a large internet corpus.

34. * Fine-tuning (12.6.2)

Solution: In the fine-tuning stage, the model parameters are adjusted to specialize the network to a particular task. An extra layer is appended onto the transformer network to convert the output vectors to the desired output format. Tasks include text classification, word classification, text span prediction. Fine-tuning is also used to refer to *transfer learning*, in which we train a model on a large general-purpose corpus, and then fine-tune it by continuing training with a specialized corpus or a specialized task setting like chat.

35. * Auto-regressive language modeling (12.7.1)

Solution: The decoder uses its own output as it generates longer output sequences. Formally, the model indirectly computes the joint probability of all tokens by predicting the conditional distributions $Pr(t_n|t_i, \ldots, t_{n-1})$.

36. Few-shot learning (12.7.4)

Solution: Few-shot learning is a type of supervised learning for small training sets with a very small example-to-class ratio. Rather than a traditional training set, few-shot learning algorithms use a *support set*, with very few examples per class. Some people argue that LLMs are capable of few-shot learning by providing a support set in a prompt. For example:

> Poor English input: I eated the purple berries.
> Good English output: I ate the purple berries.
> Poor English input: Thank you for picking me as your designer. I'd appreciate it.
> Good English output: Thank you for choosing me as your designer. I appreciate it.
> Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
> Good English output: The requested changes have been made. or I made the alteration that
> you requested. or I changed things you wanted and made the modifications.
> Poor English input: I'd be more than happy to work with you in another project.
> Good English output: *I'd be more than happy to work with you on another project.*

All text up to the italicized text in the last line is provided in the prompt, and the model generates the italicized text in repsponse after having "learned" from the few examples in the prompt.

13 Graph Neural Networks

37. Graph-level tasks (13.3.1)

Solution: Predict the temperature at which a molecule becomes liquid (a regression task) or whether a molecule is poisonous to human beings or not (a classification task).

38. Node-level tasks (13.3.1)

Solution: The network assigns a label (classification) or one or more values (regression) to each node of the graph, using both the graph structure and node embeddings.

39. Edge-prediction tasks (13.3.1)

Solution: The network predicts whether or not there should be an edge between nodes n and m. For example, in a social network, the network might predict the probability that two people should be friends.

40. * What is the defining feature of graph convolutional neural networks? (13.4)

Solution: Graph convolutional neural networks update each node by aggregating information from nearby nodes.

41. * What is meant by *relational inductive bias* in graph convolutional networks? (13.4)

Solution: They prioritize information from neighbors.

42. How is parameter sharing accomplished in graph convolutional networks? (13.4.2)

Solution: By aggregating information from neighboring nodes by summing their node embeddings.

19 Deep Reinforcement Learning

43. What is meant by *temporal credit assignment*?

> **Solution:** In a sequence of actions reward is often received only at the end. Temporal credit assignment is the problem of assigning value (credit) to intermediate steps that led to the final reward.

44. What is the *Markov property* with respect to states $s_1, s_2, \ldots, s_T$ where $t \in T$ are time steps?

> **Solution:** Transintion probabilities between states are modeled by $Pr(s_{t+1}|s)$. In other words, the next state depends only on the current state. More generally, the next state is dependent on a bounded history of states.

45. * What is the primary advantage of deep reinforcement learning over tabular reinforcement learning?

> **Solution:** Compact representation of the action value function. Tabular RL algorithms are only practical if the state-action space is relatively small.