Other Regularisation Techniques

Tirtharaj Dash

Dept. of CS & IS and APPCAIR
BITS Pilani, Goa Campus

September 16, 2021

Where we are:

Summary of the previous lecture:

- Bias-variance trade-off
- Understanding the relation between generalisation and the bias-variance trade-off
- Regularisation: Weight decay, Sparsity regularisation

Today, some more practical methodologies used to mitigate overfitting via controlling the model complexity:

- Dropout: Intuition, Concept, Practice
- Early-stopping

Dropout I

- In the last lecture, we focused on the idea that a general model is a simple model.
- A simple model is the one with fewer parameters.
- Then, we saw that the (inverse) norm of the parameters is a good measure of simplicity:
  - L_1 regularisation
  - L_2 regularisation
- Another measure of simplicity is smoothness:
  - The function should not be sensitive to small changes to its inputs.
  - Example: A model that is trained to classify images should also be able to deal correctly with a bit of noisy or blurry images.

Dropout II

Intuition:

- Christopher Bishop formalised the idea that training with input noise is equivalent to Tikhonov regularisation.
- This showed that a function be smooth (and thus simple), and the requirement that it be resilient to perturbations in the input.

Bishop, C. M. (1995). Training with noise is equivalent to Tikhonov regularization. Neural computation, 7(1), 108-116.

Note:

- L_2 regularisation is a special case of Tikhonov regularisation in the sense that each parameter is regularised differently in the latter.
- That is: it takes the form

$$||\mathbf{\Lambda w}||^2$$

  where $\mathbf{\Lambda}$ is called the Tikhonov matrix. If $\mathbf{w} \in \mathbb{R}^{d\times 1}$, $\mathbf{\Lambda} \in \mathbb{R}^{d\times d}$.
- $\mathbf{\Lambda} = \lambda I$, where I is the identity matrix, results in the L_2 regularisation

Dropout III

- Dropout is the result of adopting Bishop's idea to the internal layers of a deep network.
- That is:
  - Inject noise into each layer of the network during forward propagation during training.
  - For deep networks with many layers, injecting noise enforces smoothness on the input-output mapping.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. JMLR, 15(1), 1929–1958.

Dropout IV

Concept:

- At each training iteration, zero out some fraction of the nodes in each layer before calculating the activations for the subsequent layer.
- That is: drop out a bunch of neurons (randomly picked) and retain the rest.
- How is it done: Inject the noise in an unbiased manner so that the expected value of each layer—while fixing the others—equals to the value it would have taken absent noise.
- In Bishop's work, he added Gaussian noise to the inputs to a linear model
  - At each training iteration, the input is added with Gaussian noise:

$$\mathbf{x}' = \mathbf{x} + \epsilon$$

    where $\epsilon \sim \mathcal{N}(0, \sigma^2)$.
  - In expectation, $\mathbb{E}[\mathbf{x}'] = \mathbf{x}$.

Dropout V

- In dropout regularisation, we debias each layer by normalising by the fraction of nodes that were retrained (not dropped out).
- With *dropout probability* p, each intermediate activation h is replaced by a random variable h':

$$h' = \begin{cases} 0 & \text{with probability } p \\ \frac{h}{1-p} & \text{otherwise} \end{cases}$$

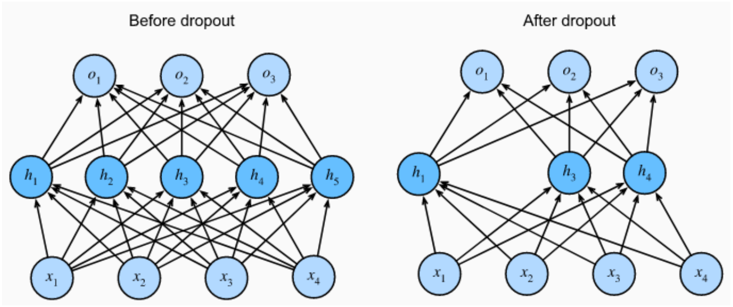
- By design, the expectation remains unchanged:

$$\mathbb{E}[h'] = h$$

Note: $1 - p$ is called keep probability (`keep_prob`). And, this implementation is called the "inverted" dropout.

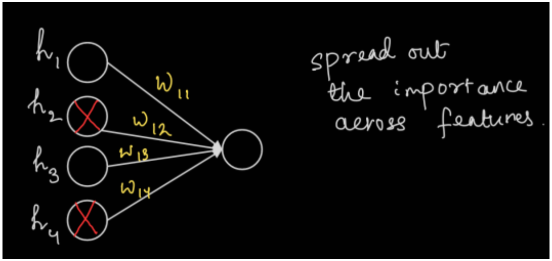
Dropout VI

- A network before and after dropout:



Dropout VII

- What does the model really achieve?
  - A neuron does not focus on a single feature that it receives:

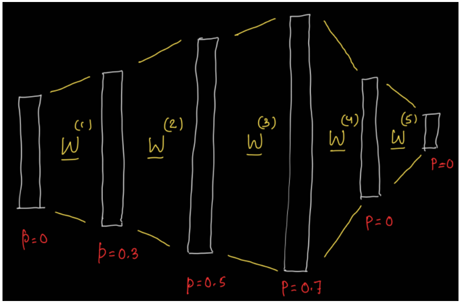


  - It can be shown that dropout achieves the same effects as Tikhonov regularisation: parameters scaled with different λs.

Dropout VIII

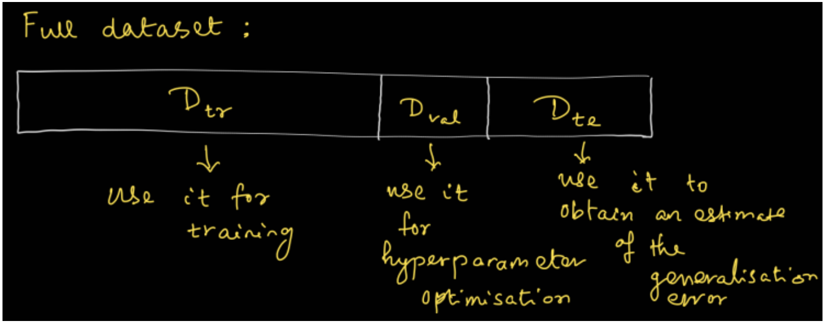
In Practice:

- The inverted dropout technique makes the implementation easy.
- We do not use dropout during testing: We do not want our model to behave in an uncertain way.
- The layers with more parameters should have higher dropout probability (p). For example:



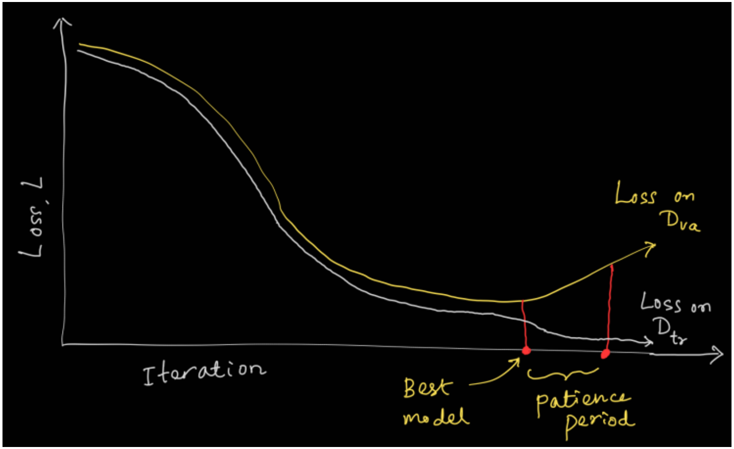
Early Stopping I

- We will be concerned with three different partitions of a dataset:


Full dataset :
D_{tr}
D_{val}
D_{te}
use it for training
use it for hyperparameter optimisation
use it to obtain an estimate of the generalisation error

Early Stopping II

- When training large models with sufficient representational capacity, we often observe that training error decreases steadily over time, but the validation set error begins to rise again.



Early Stopping III

- Early stopping is a simple strategy used during training to deal with overfitting. It works reasonably well in practice.
- Idea: During the training, if we return the model with parameter settings at a point which achieves the lowest validation set error.
    - Hope is that this would lead to a model with better test set error.

Early Stopping IV

- Procedure:
  - Every time the error on $\mathcal{D}_{val}$ improves, we store a copy of the model parameters.
  - When the training algorithm terminates, we return these parameters, rather than the latest parameters.
  - The training algorithm terminates when there is no improvement on $\mathcal{D}_{val}$ for some pre-specified number of iterations (called *patience period*).

Early Stopping V

- Drawback: Training overheads (space and time)
    - For large models, this can be overcome by saving the models on host memory, rather than in GPU or CPU.
    - It will incur a little extra training time while loading the previously saved model.