Deep Learning - Foundations and Concepts

Chapter 1. The Deep Learning Revolution

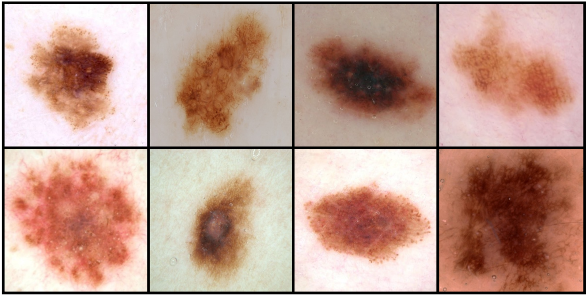
nonlineark@github

February 4, 2025

Outline

Medical diagnosis

Figure: Examples of skin lesions



Medical diagnosis

- Training set: 129K lesion images labelled as either malignant or benign.
- Training:
  - A deep neural network with 25M adjustable parameters.
  - First trained on a much larger data set of 1.28M images of everyday objects, and then *fine-tuned* on the data set of lesion images.
  - This is an example of *transfer learning*.
- This is an example of a *supervised learning* problem.
- This is also an example of a *classification* problem, compare with *regression* problems where the output consists of one or more continuous variables.

Protein structure

Figure: 3D shape of a protein called T1044/6VR4

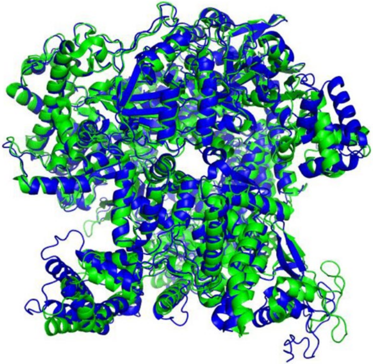


Image synthesis

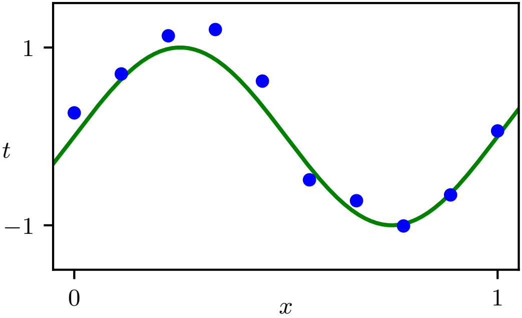
Figure: Synthetic face images

Large language models

- *Autoregressive* language models can generate language as output.
- This is an example of *self-supervised learning*.

Synthetic data

Figure: Plot of a training data set



Synthetic data

Problem

Given input variables $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}$ and target variables $t = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{pmatrix}$, predict the value of $\hat{t}$ for some new value of $\hat{x}$.

Linear models and error function

Problem'

Find $w = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_M \end{pmatrix}$, such that the linear model $y(x; w) = \sum_{j=0}^{M} w_j x^j$ has the smallest error, as defined by $E(w) = \frac{1}{2} \sum_{n=1}^{N} (y(x_n; w) - t_n)^2$.

Linear models and error function

Differentiate $E(w)$, we see that:

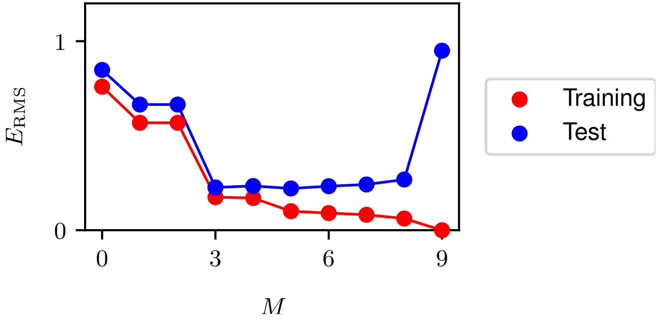
$$
\begin{aligned}
DE(w) &= \sum_{n=1}^{N} (y(x_n; w) - t_n) X_n^T \\
&= \sum_{n=1}^{N} w^T X_n X_n^T - \sum_{n=1}^{N} t_n X_n^T \\
&= w^T X X^T - t^T X^T
\end{aligned}
$$

where $X_n = \begin{pmatrix} 1 \\ x_n \\ \vdots \\ x_n^M \end{pmatrix}$, and $X = \big(X_1, X_2, \ldots, X_N\big)$.

Let $DE(w^*) = 0$, we have $w^* = (XX^T)^{-1} X t$.

Model complexity and regularization

Figure: Root-mean-square error vs. M



Model complexity and regularization

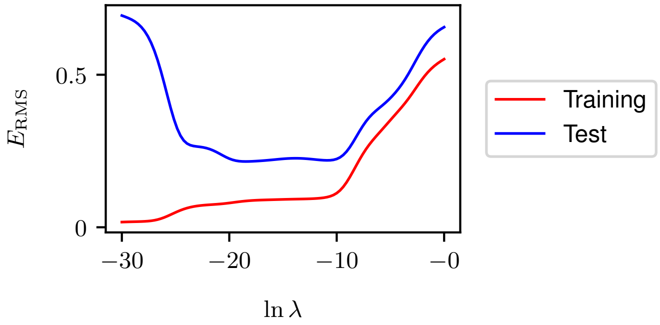
There are several ways to control the *over-fitting* phenomenon:

- Limit the number of parameters in a model according to the size of the available training set.
- *Regularization*: Add a penalty term to the error function to discourage the coefficients from having large magnitudes.

$$\tilde{E}(w) = \frac{1}{2}\sum_{n=1}^{N}(y(x_n; w) - t_n)^2 + \frac{\lambda}{2}||w||^2$$

Model complexity and regularization

Figure: Root-mean-square error vs. $\log \lambda$



Model selection

- *Hyperparameter*: Values are fixed during the minimization of the error function, e.g., M and λ.
- Training set, *validation set* and *test set*:
    - Training set: Determine the coefficients w.
    - Validation set: Select the model having the lowest error.
    - Test set: Sometimes over-fitting to the validation set can occur, keep aside a third test set to evaluate the performance of the selected model.
- *Cross-validation*