Deep Learning: From Zero to Hero

Tal Ben-Haim

June 3, 2023

Machine learning is a rapidly evolving field that has transformed the way we live, work, and interact with technology. It has become an integral part of many industries, including healthcare, finance, transportation, and more. As the world becomes increasingly data-driven, the ability to extract insights from large datasets has become critical for businesses and individuals alike. In this book, we explore the fundamentals of machine learning, from the basics of data analysis and modeling to the latest advancements in artificial intelligence. Through practical examples and real-world applications, we provide a comprehensive overview of this exciting and dynamic field.

Disclaimer: This document will inevitably contain some mistakes— both simple typos and legitimate errors. Keep in mind that these are notes in the process of learning the material, so take what you read with a grain of salt. If you find mistakes and feel like telling me, I will be grateful and happy to hear from you, even for the most trivial of errors. You can reach me by email, in English or Hebrew, at talbenha@gmail.com or simply open issue and I will provide you with the latex source code.

Tal Ben-Haim,
Last Update: June 3, 2023,

Contents

Lecture 1: Signal Processing

1.1 Introduction

Machine learning and signal processing are two closely related fields that have become increasingly important in recent years. Machine learning is the process of teaching computers to learn from data, without being explicitly programmed. Signal processing, on the other hand, is the analysis and manipulation of signals, which can include anything from sound waves to images.

Despite their differences, machine learning and signal processing share a number of fundamental concepts and techniques. In particular, many of the algorithms used in machine learning are based on statistical methods that have their roots in signal processing. For example, principal component analysis (PCA), which is commonly used in machine learning for feature extraction, is based on the spectral analysis of signals (Eigenvalue Decomposition of the covariance matrix).

Conversely, many of the techniques used in signal processing can be seen as a form of learning, in which the goal is to extract meaningful features from data without explicit supervision. For example, convolutional neural networks (CNNs) are a type of deep learning model that rely on convolutional operations to extract features from input signals, making them well-suited for a wide range of signal processing problems. By automating feature extraction and leveraging the scalability of deep learning, CNNs have enabled state-of-the-art performance on tasks including images, speech signals, and texts, demonstrating the powerful connection between machine learning and signal processing.

1.2 Fourier Analysis

Fourier analysis is a mathematical technique used to represent a signal in terms of its frequency components. The basic idea behind Fourier analysis is that any complex signal can be decomposed into a sum of simple sinusoidal functions of different frequencies. This can be useful for a variety of signal processing applications.

1.2.1 Fourier Series

The Fourier series is a mathematical technique that allows us to represent a periodic signal as a sum of sinusoids of different frequencies. The Fourier series of a signal $x(t)$ with period T is given by:

$$x(t) = \sum_{n=-\infty}^{\infty} c_n e^{j2\pi nt/T} \tag{1}$$

where c_n are the Fourier coefficients, given by:

$$c_n = \frac{1}{T}\int_{t_0}^{t_0+T} x(t)e^{-j2\pi nt/T}dt \tag{2}$$

where t_0 is any point in the interval $[0, T]$.

The Fourier series allows us to represent a periodic signal as a sum of sinusoids of different frequencies, where each sinusoid has a frequency that is an integer multiple of the fundamental frequency $f_0 = 1/T$. The Fourier series is particularly useful for analyzing periodic signals, such as those found in electrical engineering and communications.

1.2.2 Fourier Transform (Continuous-Time Signal)

The Fourier series can be extended to non-periodic signals by using the Fourier transform. In this case, the Fourier coefficients are replaced by the Fourier transform that allows us to represent a non-periodic signal as a sum of sinusoids of different frequencies, where each sinusoid has a continuous frequency. The Fourier transform of a signal $x(t)$ is defined as:

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt \tag{3}$$

where ω is the frequency in radians per second, and j is the imaginary unit. The Fourier transform takes a continuous-time signal in the time domain and converts it into a representation in the frequency domain.

The inverse Fourier transform allows us to convert a frequency-domain representation back into the time domain. The inverse Fourier transform of $X(\omega)$ is given by:

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega \tag{4}$$

The Fourier transform is particularly useful for analyzing signals with continuous frequency content, such as analog signals. It allows us to decompose a signal into its constituent frequencies, making it easier to analyze and process. However, the Fourier transform cannot be directly applied to discrete-time signals, such as those sampled from a digital system.

1.2.3 Discrete Fourier Transform (DFT)

For discrete-time signals, we use the discrete Fourier transform (DFT) instead. The DFT is a digital implementation of the Fourier transform that can be used to compute the frequency content of a discrete-time signal. The DFT is closely related to the Fourier series, which allows us to represent a periodic signal as a sum of sinusoids of different frequencies. The DFT is a commonly used technique for analyzing the frequency content of discrete-time signals. The DFT of a sequence $x[n]$ with N samples is given by:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \tag{5}$$

where $X[k]$ is the kth complex DFT coefficient. The inverse DFT (IDFT) can be used to recover the original sequence $x[n]$ from its DFT coefficients:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N} \tag{6}$$

The DFT is commonly used in applications such as audio and image compression, as well as in digital signal processing for filtering and feature extraction.

1.2.4 Short-Time Fourier Transform (STFT)

The short-time Fourier transform (STFT) is a technique used to analyze the frequency content of signals that vary over time. The STFT is computed by applying the Fourier transform to short segments of the signal, known as windows, and then sliding the window along the signal to compute the frequency content at different time points. The STFT of a signal $x[n]$ can be computed as follows:

$$X(m,k) = \sum_{n=0}^{N-1} x[n+mH] w[n] e^{-j2\pi kn/N} \tag{7}$$

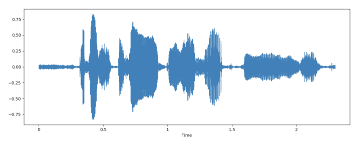
where $X(m,k)$ is the STFT coefficient at time m and frequency k, H is the hop size (i.e., the number of samples between adjacent windows), and $w[n]$ is the window function.

The inverse STFT (ISTFT) can be used to recover the original signal from its STFT coefficients:

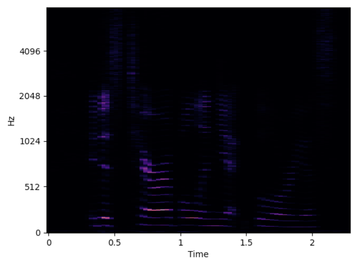
$$x[n] = \frac{1}{N} \sum_{m=0}^{M-1} \sum_{k=0}^{N-1} X(m,k) w[n-mH] e^{j2\pi kn/N} = \frac{1}{N} \sum_{m=0}^{M-1} x_m[n-mH] \tag{8}$$

where $x_m[n]$ is the m-th segment of the signal, and $w[n]$ is the window function. The STFT is commonly used in applications such as speech and music processing, where the frequency content of a signal can change over time.

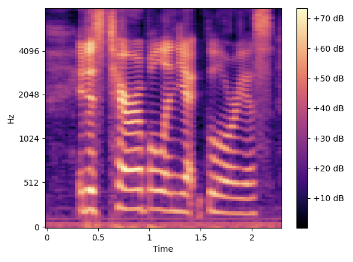
Time-frequency analysis. Time-frequency analysis enables the exploration of dynamic signals, facilitating audio tasks such as sound recognition, speech processing, and music analysis. It is a powerful approach that combines information from both the time and frequency domains to provide a more comprehensive understanding of signals. The STFT divides a signal into small overlapping segments and applies the Fourier transform to each segment, revealing the frequency content at different time intervals. This allows for the examination of how the signal's frequency components change over time. Another important technique in time-frequency analysis is the Mel spectrogram, which is derived from the STFT. The Mel spectrogram applies a non-linear transformation to the STFT magnitude spectrum, emphasizing perceptually important frequency bands while reducing sensitivity to absolute frequency. The Mel spectrogram is commonly used in applications like speech and audio processing, providing a more intuitive representation of the signal's frequency content that aligns with human auditory perception. Figure 1 showcases the raw samples of an audio signal containing the sentence ”It is terrible and yet glorious.”, along with its corresponding Short-Time Fourier Transform (STFT) and Mel spectrogram. The STFT provides insight into the frequency content of the signal at different time intervals, while the Mel spectrogram offers a perceptually meaningful representation of the signal's frequency content. Together, these visualizations illustrate the application of time-frequency analysis techniques to analyze the given audio signal.



(a) Raw Audio Signal



(b) STFT



(c) Mel Spectrogram

Figure 1: (a) The raw samples of an audio signal containing the sentence ”It is terrible and yet glorious,”, along with its STFT (b) and Mel Spectrogram (c).

1.2.5 Fast Fourier Transform (FFT)

The fast Fourier transform (FFT) is an efficient algorithm for computing the DFT of a sequence. The FFT takes advantage of the symmetry properties of the DFT to reduce the number of computations required. Specifically, the FFT computes the DFT of a sequence with length $N = 2^p$ in $O(N \log N)$ operations, compared to the $O(N^2)$ operations required for the standard DFT algorithm.

The FFT is commonly used in applications such as audio and image processing, where large sequences need to be analyzed quickly. The FFT can also be used in conjunction with the STFT to compute the frequency content of a signal over time

1.3 Filtering and Convolution

Filtering is the process of removing unwanted noise or features from a signal while preserving important information. This is often accomplished by convolving the signal with a filter kernel. The filter kernel, also known as the impulse response, specifies how much each input value contributes to the output of the filter. In this section, we will discuss convolution, a fundamental operation in signal processing, and filter design.

1.3.1 Convolution of Continuous Signals

The convolution of two continuous signals $f(t)$ and $g(t)$ is defined as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{9}$$

This can be thought of as a sliding dot product between the two signals, with the dot product taken at each time point t. The resulting output signal is typically of length $2N-1$, where N is the length of the input signals.

1.3.2 Convolution of Discrete Signals

For discrete signals, the convolution can be defined as:

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \tag{10}$$

This is similar to the continuous case, but with the integral replaced by a sum.

1.3.3 Convolution using Matrix Multiplication

Convolution can also be formulated using linear algebra by representing signals as vectors and the convolution operation as a matrix-vector multiplication. Let $\mathbf{f} = [f_1, f_2, \ldots, f_n]^T$ be the input signal and $\mathbf{g} = [g_1, g_2, \ldots, g_m]^T$ be the filter kernel. We can construct a convolution matrix $\mathbf{A}$ such that the output signal $\mathbf{y} = [y_1, y_2, \ldots, y_{n+m-1}]^T$ is given by:

$$\mathbf{y} = \mathbf{A}\mathbf{f} \tag{11}$$

where $\mathbf{A}$ is an $(n+m-1) \times n$ Toeplitz matrix defined as:

$$\mathbf{A} = \begin{bmatrix} g_1 & 0 & 0 & \cdots & 0 \\ g_2 & g_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_m & g_{m-1} & g_{m-2} & \cdots & g_1 \\ 0 & g_m & g_{m-1} & \cdots & g_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & g_m \end{bmatrix} \tag{12}$$

Each row of $\mathbf{A}$ represents a shifted version of the filter kernel $\mathbf{g}$, and the matrix-vector multiplication essentially computes the dot product between the shifted filter kernel and the input signal for each shift. Note that the resulting output signal $\mathbf{y}$ has a length of $n+m-1$ to account for the potential overlap at the boundaries of the convolution.

This formulation can be useful in cases where it is computationally expensive to compute the Fourier transform, such as when dealing with very large signals.

1.3.4 Fourier transform and Convolution

The Fourier transform and convolution are closely related concepts. In fact, the convolution theorem states that the Fourier transform of a convolution of two signals is equal to the pointwise product of their individual Fourier transforms. Mathematically, this can be written as:

$$\mathcal{F}\{f * g\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{g\} \tag{13}$$

where $\mathcal{F}\{\cdot\}$ denotes the Fourier transform operator, and f and g are the two signals being convolved. The pointwise multiplication in the frequency domain is equivalent to the convolution in the time domain.

This property is particularly useful because it allows us to perform convolutions more efficiently by taking advantage of the fast Fourier transform (FFT) algorithm. Instead of directly computing the convolution in the time domain, we can first compute the Fourier transform of the signals, multiply them in the frequency domain, and then inverse transform the result back to the time domain to obtain the convolution. The FFT algorithm is much faster than the direct computation of convolution, especially for large signals, and can significantly speed up many signal processing tasks.

1.3.5 Filter Design

Filter design is the process of designing a filter kernel that can be used to filter a signal. There are many different types of filters, each with its own design criteria and trade-offs. Some common types of filters include low-pass, high-pass, band-pass, and band-stop filters.

A low-pass filter, for example, allows low-frequency signals to pass through while attenuating high-frequency signals. The filter kernel for a low-pass filter can be designed using a windowing method, where a window function is applied to a sinc function. The resulting kernel is then scaled and shifted to obtain the desired frequency response.

1.3.6 Cross-Correlation

Cross-correlation is closely related to convolution, and can be thought of as a measure of similarity between two signals. Given two signals f and g, their cross-correlation is defined as:

$$(f \star g)(t) = \int_{-\infty}^{\infty} f(\tau)g(\tau + t)d\tau \tag{14}$$

where $\star$ denotes the cross-correlation operation. Essentially, this operation slides one of the signals (in this case, g) along the other signal (in this case, f), and computes the overlap between the two signals at each point.

If we consider the case where one of the signals is flipped in time, the cross-correlation operation becomes equivalent to a convolution:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau = (f \star g(-t))(t) \tag{15}$$

where $*$ denotes the convolution operation. In this case, the signal g is flipped in time, which changes the sign of the time variable in the argument of the integral.

Cross-correlation is often used in signal processing to measure the similarity between two signals, and one important application of cross-correlation is in matched filtering.

Matched filtering is a technique used to detect a particular signal (called the "template" signal) in the presence of noise. The idea behind matched filtering is to correlate the incoming signal with a template signal that represents the expected pattern of the desired signal.

The cross-correlation function between the template signal $h(t)$ and the input signal $x(t)$ is given by:

$$y(t) = \int_{-\infty}^{\infty} \bar{h}(\tau)x(t + \tau)d\tau \tag{16}$$

where $\bar{h}(\tau)$ denotes the complex conjugate of $h(\tau)$.

This operation is also called the matched filter operation, because the filter is matched to the template signal.

The matched filter output $y(t)$ can be interpreted as a measure of the similarity between the template signal and the input signal at each point in time. When the input signal contains the desired signal, the matched filter output will be high at the time when the template signal matches the desired signal. By comparing the matched filter output to a threshold value, we can detect the presence of the desired signal in the input signal.

Matched filtering is widely used in various applications, such as radar signal processing, image processing, and communications.

Notes