# Deep Learning - Foundations and Concepts

## Chapter 15. Discrete Latent Variables

nonlineark@github

March 31, 2025

## Outline

# $K$-means clustering

**Problem**

Suppose we have a data set $\{x^1, \ldots, x^N\}$ consisting of $N$ observations of a $D$-dimensional Euclidean variable $x$. Partition the data set into some number $K$ of clusters, where we will suppose for the moment that the value of $K$ is given.

# $K$-means clustering

## Problem'

Find:

- $K$ cluster centers: $\mu_1, \dots, \mu_K \in \mathbb{R}^D$.
- $N$ data point assignment: $r^1, \dots, r^N \in \{\mathrm{e}_1, \dots, \mathrm{e}_K\}$.

such that the error function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_k^n ||x^n - \mu_k||^2$$

which represents the sum of the squares of the distances of each data point to its assigned cluster center, is minimized.

# $K$-means clustering

We can do this through an iterative procedure:

1. Choose some initial values for the $\{\mu_k\}$.
2. E step: Minimize $J$ with respect to the $\{r_k^n\}$, keeping the $\{\mu_k\}$ fixed.
3. M step: Minimize $J$ with respect to the $\{\mu_k\}$, keeping the $\{r_k^n\}$ fixed.
4. Go to step 2 until convergence.

# $K$-means clustering

Consider the E step. It's easy to see that we should assign the $n$th data point to the closest cluster center:

$$r_k^n = \begin{cases} 1, & \text{if } k = \arg\min_j ||x^n - \mu_j||^2 \\ 0, & \text{otherwise} \end{cases}$$

For the M step:

$$\frac{\partial J}{\partial \mu_k} = 2 \sum_{n=1}^{N} r_k^n (x^n - \mu_k)^T$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_k^n x^n}{\sum_{n=1}^{N} r_k^n}$$

so $\mu_k$ is equal to the mean of all the data points $x_n$ assigned to cluster $k$.

# $K$-means clustering

**Algorithm 1:** $K$-means algorithm

$\{r_k^n\} \leftarrow 0$;
**repeat**
    $\{{}^{\text{old}}r_k^n\} \leftarrow \{r_k^n\}$;
    **for** $n \leftarrow 1$ **to** $N$ **do**
        $k \leftarrow \arg\min_j ||x^n - \mu_j||^2$;
        $r_k^n \leftarrow 1$;
        $r_{j \neq k}^n \leftarrow 0$;
    **end**
    **for** $k \leftarrow 1$ **to** $K$ **do**
        $\mu_k \leftarrow \frac{\sum_{n=1}^N r_k^n x^n}{\sum_{n=1}^N r_k^n}$;
    **end**
**until** $\{r_k^n\} = \{{}^{\text{old}}r_k^n\}$;
**return** $\{\mu_k\}, \{r_k^n\}$;

# $K$-means clustering

When updating the prototype vectors, we can also derive a sequential update in which, for each data point $x^n$ in turn, we update the nearest prototype $\mu_k$ using:

$$^{\text{new}}\mu_k = {}^{\text{old}}\mu_k + \frac{1}{N_k}(x^n - {}^{\text{old}}\mu_k)$$

where $N_k$ is the number of data points that have so far been used to update $\mu_k$.

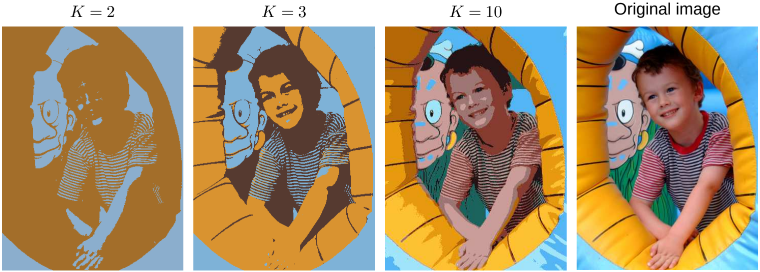
# Image segmentation

Using the $K$-means algorithm to perform (toy) image segmentation:

- Each pixel in an image is a point in a three-dimensional space comprising the intensities of the red, blue and green channels.
- We treat each pixel in the image as a separate data point.
- We can apply the $K$-means algorithm to these pixels, and redraw the image in which we replace each pixel by the center $\mu_k$ to which that pixel has been assigned.

# Image segmentation

Figure: Application of the $K$-means clustering algorithm to image segmentation



# Image segmentation

Using the $K$-means algorithm to perform lossy data compression:

- For each of the $N$ data points, we store only the identity $k$ of the cluster to which it is assigned.
- We also store the values of the $K$ cluster centers $\{\mu_k\}$.

This framework is often called vector quantization, and the vectors $\{\mu_k\}$ are called codebook vectors.