Lab 3 - Image compression with k-means segmentation

October 28, 2020

August Hertz Bugge (s194350) & Kris Back Kruse (s204152) & sarphiv (redacted)

Abstract

Storing images is necessary for many AI tasks, however, data storage at large scale is costly. A widely used technique is to compress images using lossy algorithms like JPEG, but even a small improvement can save money. We compare k-means clustering as a lossy compression algorithm vs. JPEG on 1890 images using qualitatively chosen parameters for our k-means clustering. Our algorithm has a compression ratio of ∼35% better than JPEG. This indicates high compression ratios can be obtained if parameters are optimized for a given use case.

Introduction

Storing large amounts of images requires a lot of storage space which is costly. Many algorithms exists, and is used in different situations, to compressed images. We want to compare the compression of an image, using k-means clustering and the widely used "JPEG" algorithm, in terms of space gained. Our hypothesis is that, using the k-means clustering algorithm, we will get within 20% points of the compression ratio of the "JPEG" algorithm, when compared to the equivalent bitmap file.

Methods

A python script converts images to the CIELAB color space, so a k-means algorithm can be run with euclidean distances between present colors. Amount of clusters is $n_c = 1 + ceil(n_{cf} \cdot log_2(n_{pixels}), \{n_c \in N \mid 0 n_c \leq 255\}$. k-means is run until convergence or n_i iterations. After k-means, similar cluster centers are merged if their colour distance is less than c_{lim}. Image shape, cluster centers, and pixel cluster assignments are saved to a file.
Equivalent JPEG and k-mean images were compared at 1 meter for similarity on a 24" FullHD screen. A qualitative analysis of parameters lead to $n_{cf} = 16$, $n_i = 30$, and $c_{lim} = 1$ being chosen as images became hard to distinguish. 1890 images were compressed using k-means. Their compression ratio to an equivalent BMP file was saved and compared to the JPEG compression ratio.
Each image is only compressed once with k-means giving one file size for each image. Our initialization of cluster centers is non-deterministic leading to variance in file sizes for the same image. The standard deviation of this percentage wise variation for each image is measured and calculated for a subset of 20 images over 20 k-means runs.

Result

	Mean	Conf.	Non-determ. std.
JPEG	19.96%	±0.3202%	N/A
k-means	13.34%	±0.1886%	1.500%

Table 1: Compression ratio relative to an equivalent BMP file

The results show that a JPEG is, on average, ∼20% of an equivalent BMP file, and the k-means is, on average, ∼13% of an equivalent BMP file. k-means compression has a ∼35% better compression ratio than JPEG, and the non-deterministic variation is small enough to not impact our results.

Discussion

The two confidence intervals in our comparison are non-overlapping. Our results are conclusively different leading us to the conclusion: k-means compression beats JPEG in this dataset under this experimental setup. We have therefore outperformed our hypothesis.
This could have been caused by the JPEG images having been compressed lightly to preserve quality, while the k-means compression threw away much more detail. The experiment setup with our qualitative selection of parameters could have unfairly favoured the k-means compression. We can therefore not conclude k-means compression beats JPEG compression in all cases. We have, however, shown it possible to throw away more data without easily perceivable losses, suggesting optimal compression ratios can be obtained if images are compressed dependent upon their usage scenarios.

References

- Image data set: `http://images.cocodataset.org/zips/val2017.zip`
- Source code: `https://github.com/sarphiv/dtu-intro-ai-lab-03`

Learning outcome

Processing many images requires a lot of computational power. In the future, more time should be spent error-checking, optimizing, and preparing as this could ironically lead to a loss of time.