Woody International Publish Limited
An Multidisciplinary Academic Journal Publisher

Journal of Artificial Intelligence and Information, Volume 2, 2025
https://www.woodyinternational.com/

Application of K-Means Algorithm in Image Compression

Hongxia Mao

School of Computer and Software, Jincheng College, Chengdu 611731, Sichuan, China

Abstract: ***In the Internet era, image information is widely used in all walks of life. The amount of data in images is very large, and it is necessary to compress images in order to effectively transmit and store them. The K-Means algorithm is one of the most commonly used clustering algorithms. This article applies the clustering algorithm to image compression applications and uses the K-Means clustering algorithm to compress images through Python programming. From the experimental results, it can be concluded that the K-Means clustering method can indeed compress images.***

Keywords: K-Means; Clustering; Image compression.

Cited as: Mao, H. (2025). Application of K-Means Algorithm in Image Compression. *Journal of Artificial Intelligence and Information*, *2*, 27–31. Retrieved from https://woodyinternational.com/index.php/jaii/article/view/145

1. Introduction

Cluster analysis is an important research field in data mining. It groups data objects into several classes or clusters, making objects in the same cluster more similar, while objects in different clusters differ greatly. Image information is widely used in various industries, and the larger the amount of information contained in an image, the larger the space it occupies. Therefore, when transmitting and saving images, appropriate methods should be used for compression. Image compression refers to the process of reducing the amount of data required to represent an image by eliminating redundant data under certain quality conditions, making it easier to store and transmit the image. Liu et al. (2024) proposed the Promoted Osprey Optimizer as a solution for the Optimal Reactive Power Dispatch (ORPD) problem considering electric vehicle penetration, enhancing operational efficiency in power systems. Chen et al. (2023) introduced Octopus, a framework for in-network content adaptation to mitigate congestion on 5G links, improving network performance. Liang and Chen (2019) developed an SDN-based hierarchical authentication mechanism for IPv6 addresses, aiming to enhance network security. Chen and Bian (2019) designed a streaming media live broadcast system using MSE, focusing on improving broadcast quality. Xie et al. (2024) presented a Conv1D-based approach for multi-class classification of legal citation texts, advancing text classification methodologies. Luo et al. (2024) enhanced e-commerce chatbots using Falcon-7B and 16-bit full quantization, improving chatbot performance and efficiency. Xu et al. (2024) investigated methods to enhance user experience and trust in advanced LLM-based conversational agents. Liu (2024) optimized supply chain efficiency using cross-efficiency analysis and inverse DEA models. Bi et al. (2024) explored the role of AI, specifically ChatGPT, in financial forecasting, highlighting potential and challenges. Chen et al. (2022) introduced a one-stage object referring system with gaze estimation, contributing to advancements in computer vision. Wang et al. (2024) researched autonomous robot navigation based on reinforcement learning, enhancing robotic autonomy. Wu et al. (2024) proposed a lightweight GAN-based image fusion algorithm for visible and infrared images, improving image processing capabilities. Lin et al. (2024) focused on AI and electroencephalogram analysis for optimizing anesthesia depth, providing innovative methods in medical applications. Ren (2024) enhanced YOLOv8 for infrared image object detection with an SPD module and introduced a novel topic segmentation approach for dialogue summarization, advancing object detection and natural language processing techniques, respectively. Fan et al. (2024) researched optimizing real-time data processing in high-frequency trading algorithms using machine learning, contributing to financial technology advancements.

2. K-Means Clustering Algorithm

Clustering is a very important application field in data mining. Clustering refers to dividing samples with high similarity into the same cluster based on the principle of similarity, and dividing samples with high dissimilarity into different clusters. The K-Means algorithm is one of the most commonly used clustering algorithms. The K-Means clustering algorithm, also known as the k-means algorithm, uses distance as a measure of similarity between samples. The smaller the distance between samples, the higher their similarity, and they may be in the same cluster. This algorithm receives parameter k and then divides the sample points into k clusters; The similarity of samples within the same cluster is relatively high; The similarity of samples in different clusters is low. The idea of this algorithm is to cluster k sample points in space and classify the sample points closest to them. Through iterative methods, gradually update each cluster center until the best clustering effect is achieved.

The steps of the algorithm are as follows:

(1) Randomly select k samples from n sample datasets as initial clustering centers;

(2) Calculate the distance between each sample in the dataset and k cluster centers separately, and divide the samples into the class corresponding to the cluster center with the smallest distance;

(3) For each category, recalculate its cluster center;

(4) Repeat steps 2 and 3 until the position of the cluster center no longer changes.

Overall, the clustering idea of K-means algorithm is relatively simple and easy to implement, and the clustering effect is acceptable. It is a simple, efficient and widely used clustering method.

3. Application of K-Means Algorithm in Compressed Images

3.1 Principle of K-Means Algorithm for Compressing Images

When an image is displayed on a computer screen, it occupies the computer's memory space. The calculation formula for the memory occupied by an image is: image height * image width * memory size occupied by one pixel. The number of bytes contained in each pixel directly affects the size of memory occupied by the image. Storing images with different color modes requires different memory sizes, as shown in Table 1:

Table 1: Small comparison of memory usage per pixel for different image types

Image Type	How many bytes per pixel	The number of colors that can be represented
1-bit data graph (Line art)	1/8 byte per pixel	$2^1 = 2$
8-bit Grayscale	1 byte per pixel	2^8
16 bit Grayscale	2 bytes per pixel	2^{16}
24 bit RGB	3 bytes per pixel, which is the most commonly used format in images, such as TIF format.	2^{24}
32-bit printing color mode (CMYK)	4 bytes per pixel	2^{32}
48 bit RGB	6 bytes per pixel	2^{48}

The most basic principle of image compression is to replace some similar colors with one color, reducing the number of color descriptions, so that each corresponding pixel can be described with fewer bytes, and the memory occupied by the image will be reduced. The basic principle of using K-Means clustering algorithm for image compression is to classify each pixel in an image, and replace the values of pixels classified into the same category with the values of the cluster centers of that category, in order to reduce the memory space occupied by the image.

3.2 Experimental Process

When importing images into a program, preprocessing of the image data is required. According to the resolution of the image, the data points in the image are tiled, and each pixel is treated as a 3D sample. The pixel values of an image are converted into n rows and 3 columns of data, where n=height * width

Using KMeans clustering algorithm, set the value of k, cluster all color values in the image, and find the cluster center corresponding to each 3D pixel point.

#The number of colors contained in the compressed image is the number of clusters

```
k=64
Kmeans = KMeans(n_clusters = k, random_state = 0)

#Training model
kmeans. fit(pixel_sample)

#Find the cluster center corresponding to each 3D pixel point
cluster_assignments = kmeans.predict(pixel_sample)
```

Traverse every pixel in the image, find the pixel value of the cluster center corresponding to each pixel value, and replace the value of the pixel with the pixel value of the cluster center.

```
#Traverse each pixel point and find the pixel value corresponding to the cluster center
pixel_count = 0
for i in range(height):
    for j in range(width):
        # 获取像素点的聚类中心的索引
        cluster_idx = cluster_assignments[pixel_count]
        #获取聚类中心索引位置上的像素值
        cluster_valuecluster = centers[cluster_idx]
        # 替换像素点的值
        compressed_img[i][j] = cluster_value
        pixel_count += 1
```

Run the code, Figure 1 shows the image before compression, and Figure 2 shows the compressed image when using the K-Means algorithm with k=64.

Figure 1: Effect before compression

Figure 2: Compressed Effect

3.3 Experimental Results

From the running results, it can be seen that the original image is 92KB (bird. TIF), while the compressed image is only 45K (compressible_bird), achieving the goal of image compression.

4. Conclusion

From the experimental results, it can be seen that the K-Means clustering method can indeed compress images. The smaller the value of k, the greater the compression ratio, but the less color the compressed image has, resulting in a loss of a large amount of pixel color information and a greater difference from the original image. The K-Means algorithm is simple and easy to implement, but it requires users to specify the number of clusters (value of k) in advance, and the clustering results are sensitive to the selection of initial cluster centers, which can easily lead to local optima. In practice, in order to obtain better results, the K-Means algorithm is usually run multiple times with different initial cluster centers. After all sample assignments are completed, when recalculating the centers of k clusters, for continuous data, the cluster centers should take the mean of that cluster.

References

[1] Liu, Z., Jian, X., Sadiq, T., Shaikh, Z. A., Alfarraj, O., Alblehai, F., & Tolba, A. (2024). Promoted Osprey Optimizer: a solution for ORPD problem with electric vehicle penetration. Scientific Reports, 14(1), 28052.
[2] Chen, Y., Tahir, A., Yan, F. Y., & Mittal, R. (2023, December). Octopus: In-Network Content Adaptation to Control Congestion on 5G Links. In 2023 IEEE/ACM Symposium on Edge Computing (SEC) (pp. 199-214). IEEE.
[3] Liang, X., & Chen, H. (2019, July). A SDN-Based Hierarchical Authentication Mechanism for IPv6 Address. In 2019 IEEE International Conference on Intelligence and Security Informatics (ISI) (pp. 225-225). IEEE.
[4] Chen, H., & Bian, J. (2019, February). Streaming media live broadcast system based on MSE. In Journal of Physics: Conference Series (Vol. 1168, No. 3, p. 032071). IOP Publishing.
[5] Xie, Y., Li, Z., Yin, Y., Wei, Z., Xu, G., & Luo, Y. (2024). Advancing Legal Citation Text Classification A Conv1D-Based Approach for Multi-Class Classification. Journal of Theory and Practice of Engineering Science, 4(02), 15–22. https://doi.org/10.53469/jtpes.2024.04(02).03
[6] Luo, Y., Wei, Z., Xu, G., Li, Z., Xie, Y., & Yin, Y. (2024). Enhancing E-commerce Chatbots with Falcon-7B and 16-bit Full Quantization. Journal of Theory and Practice of Engineering Science, 4(02), 52–57. https://doi.org/10.53469/jtpes.2024.04(02).08
[7] Xu, Y., Gao, W., Wang, Y., Shan , X., & Lin, Y.-S. (2024). Enhancing user experience and trust in advanced LLM-based conversational agents. Computing and Artificial Intelligence, 2(2), 1467. https://doi.org/10.59400/cai.v2i2.1467
[8] Liu, M. (2024). Optimizing Supply Chain Efficiency Using Cross-Efficiency Analysis and Inverse DEA Models.
[9] Bi, S., Deng, T., & Xiao, J. (2024). The Role of AI in Financial Forecasting: ChatGPT's Potential and Challenges. arXiv preprint arXiv:2411.13562.
[10] Chen, J., Zhang, X., Wu, Y., Ghosh, S., Natarajan, P., Chang, S. F., & Allebach, J. (2022). One-stage object referring with gaze estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5021-5030).
[11] Wang, Z., Yan, H., Wang, Z., Xu, Z., Wu, Z., & Wang, Y. (2024, July). Research on autonomous robots navigation based on reinforcement learning. In 2024 3rd International Conference on Robotics, Artificial Intelligence and Intelligent Control (RAIIC) (pp. 78-81). IEEE.
[12] Wu, Z., Chen, J., Tan, L., Gong, H., Zhou, Y., & Shi, G. (2024, September). A lightweight GAN-based image fusion algorithm for visible and infrared images. In 2024 4th International Conference on Computer Science and Blockchain (CCSB) (pp. 466-470). IEEE.
[13] Lin, S., Hu, K., Ye, T., Wang, Y., & Shen, Z. (2024). Artificial Intelligence and Electroencephalogram Analysis Innovative Methods for Optimizing Anesthesia Depth. Journal of Theory and Practice in Engineering and Technology, 1(4), 1–10. https://doi.org/10.5281/zenodo.14457933
[14] Ren, Z. (2024). Enhanced YOLOv8 Infrared Image Object Detection Method with SPD Module. Journal of Theory and Practice in Engineering and Technology, 1(2), 1–7. Retrieved from https://woodyinternational.com/index.php/jtpet/article/view/42
[15] Ren, Z. (2024). A Novel Topic Segmentation Approach for Enhanced Dialogue Summarization. World Journal of Innovation and Modern Technology, 7(4), 42-49.

[16] Fan, Y., Hu, Z., Fu, L., Cheng, Y., Wang, L., & Wang, Y. (2024). Research on Optimizing Real-Time Data Processing in High-Frequency Trading Algorithms using Machine Learning. arXiv preprint arXiv:2412.01062.