**System Design Visual Search System**

Video Link: https://www.youtube.com/watch?v=5mkegPovTI8&list=PL\_b\_MRp1BnEj4UuHv1jAGeO1a0VTRXsAk&index=5

1. Objective

A close-up of a logo

Description automatically generated

A screenshot of a video

Description automatically generated

A cup of tea with a spoon

Description automatically generated

1. Framing problem as an ML task

A diagram of a search system

Description automatically generated

A screenshot of a white text

Description automatically generated

Representation learning is a machine learning approach focused on automatically discovering and organizing features or representations from raw data. Instead of manually designing features, the model learns how to represent data in ways that make it easier for algorithms to perform tasks like classification, clustering, or prediction. Deep learning, such as neural networks, is commonly used for this as it can learn hierarchical and complex features directly from the data. This approach is widely used in image processing, NLP, and recommendation systems.

A diagram of a model

Description automatically generated

An embedding is a way of converting items (like words, products, or even people) into numbers, making it easier for computers to understand and work with them. It’s like creating a unique "location" for each item in a large "map" of numbers, where similar items are placed closer together. This helps computers recognize relationships, such as how "apple" is similar to "banana" because both are fruits, even though they’re represented by different numbers. Embeddings are essential in tasks like recommendation systems, search engines, and language translation because they help algorithms find and understand patterns.

Embeddings are numerical representations of items like words, images, or products, mapped into a multi-dimensional "embedding space." In this space, each embedding (a vector of numbers) represents an item, and the distance between these embeddings reflects how similar or different the items are.

For example, suppose we have an image embedding model. When we process images of, say, dogs and cats, each image is transformed into an embedding in this space. If two images show similar things—like two different photos of a golden retriever—their embeddings will be close to each other in the space. Images of cats, on the other hand, will form a separate cluster, closer to other cats.

**Finding Similar Images Using Embedding Space**

To find images similar to a given one, we can calculate the "distance" between the embedding of the query image and the embeddings of other images in the dataset. A common way to do this is by measuring cosine similarity or Euclidean distance.

For example:

1. **Query Image**: You input a photo of a golden retriever.
2. **Embedding Search**: The model transforms this image into an embedding vector.
3. **Compare Distances**: The model finds other embeddings in the dataset that are closest to this vector.
4. **Results**: Images of other dogs, especially golden retrievers, appear as the most similar matches.

This method is used in applications like Google Images, where you can search by image, and it finds visually similar images.

A diagram of a diagram

Description automatically generated

1. Data

A screenshot of a computer

Description automatically generated

A screenshot of a computer game

Description automatically generated

A diagram of a block diagram

Description automatically generated

**Diagram Explanation**

1. **Input Image**: The process starts with an input image, which is a high-dimensional array of pixel values.
2. **Convolutional Layers**:
   * The CNN applies multiple convolutional layers to the image. These layers detect various features like edges, textures, shapes, and complex patterns.
   * Convolutional layers progressively transform the raw pixel data into more abstract feature representations. Each layer builds on the features learned by the previous one.
3. **Fully Connected Layers**:
   * After the convolutional layers, the features extracted from the image are flattened and fed into fully connected (dense) layers.
   * These layers further combine and process the features to capture more complex relationships and representations, ultimately reducing the data to a lower-dimensional vector.
4. **Embedding Vector**:
   * The final output from the fully connected layers is an **embedding vector**, a dense, lower-dimensional representation of the original image.
   * This vector is a series of numbers (as shown, e.g., [0.1, 0.8, -1, 0.7, 0]) that uniquely represents the image in a more compact form, which is useful for various downstream tasks like image retrieval or similarity comparison.

**Bullet Point Explanation**

* **Embedding as Dimensionality Reduction**:
  + Embeddings are a **dimensionality reduction technique**. They transform high-dimensional data (such as raw images) into a lower-dimensional vector representation.
  + This is beneficial because it allows for the storage and processing of complex data in a compressed form without losing essential information.
* **Feature Extraction for Dense Vectors**:
  + The process of creating embeddings involves **extracting distinct features** from the input data, whether it's images or text.
  + For images, these features could include patterns, colors, shapes, or textures that uniquely define the content of the image.
  + The extracted features are then represented as a **dense vector** (embedding) in a multi-dimensional space, where similar images (in terms of content or visual features) are closer together.
* **CNN Architecture**:
  + A **CNN (Convolutional Neural Network)** is used here as the model architecture.
  + CNNs typically consist of two main types of layers:
    - **Convolutional layers** for feature extraction (these detect patterns within the data).
    - **Fully connected layers** (classification layers) to interpret and summarize these features into a lower-dimensional output.

**Applications of Embedding Vectors**

* **Similarity Search**: Embedding vectors allow similar images to be easily found. Since images with similar features have embeddings that are close in the embedding space, we can calculate distances (e.g., using cosine similarity) to find images that are visually similar to a query image.
* **Image Classification and Clustering**: The embedding vector can also be used to classify or cluster images. For example, images of cats and dogs will have distinct clusters in the embedding space, making it easy to identify which category a new image belongs to based on its embedding.