Multimodal AI

August 17, 2024

Chapter 2
Multimodal Joint Representation

Introduction

- Multimodal representations integrate information from different data modalities (e.g., text, images, audio).
- Aim to create a unified representation that captures the underlying information from all modalities.
- Applications include image captioning, audio-visual speech recognition, and cross-modal retrieval.

Joint Representations

- **Definition**: Representations where information from different modalities is combined into a single shared space.
- **Example**:
  - Text and image embeddings combined into a single vector space.
- **Techniques**:
  - Multimodal deep learning models.
  - Use of shared weights and loss functions.

Advantages of Joint Representations

1. **Enhanced Understanding:** By combining information from different modalities, joint representations can capture more context and nuances, leading to better understanding and performance in tasks such as classification, retrieval, and generation.
2. **Cross-modal Applications:** Joint representations enable cross-modal applications such as image captioning, visual question answering, and text-to-image generation, where understanding and generating data across different modalities is crucial.

Example: Text and Image Embeddings

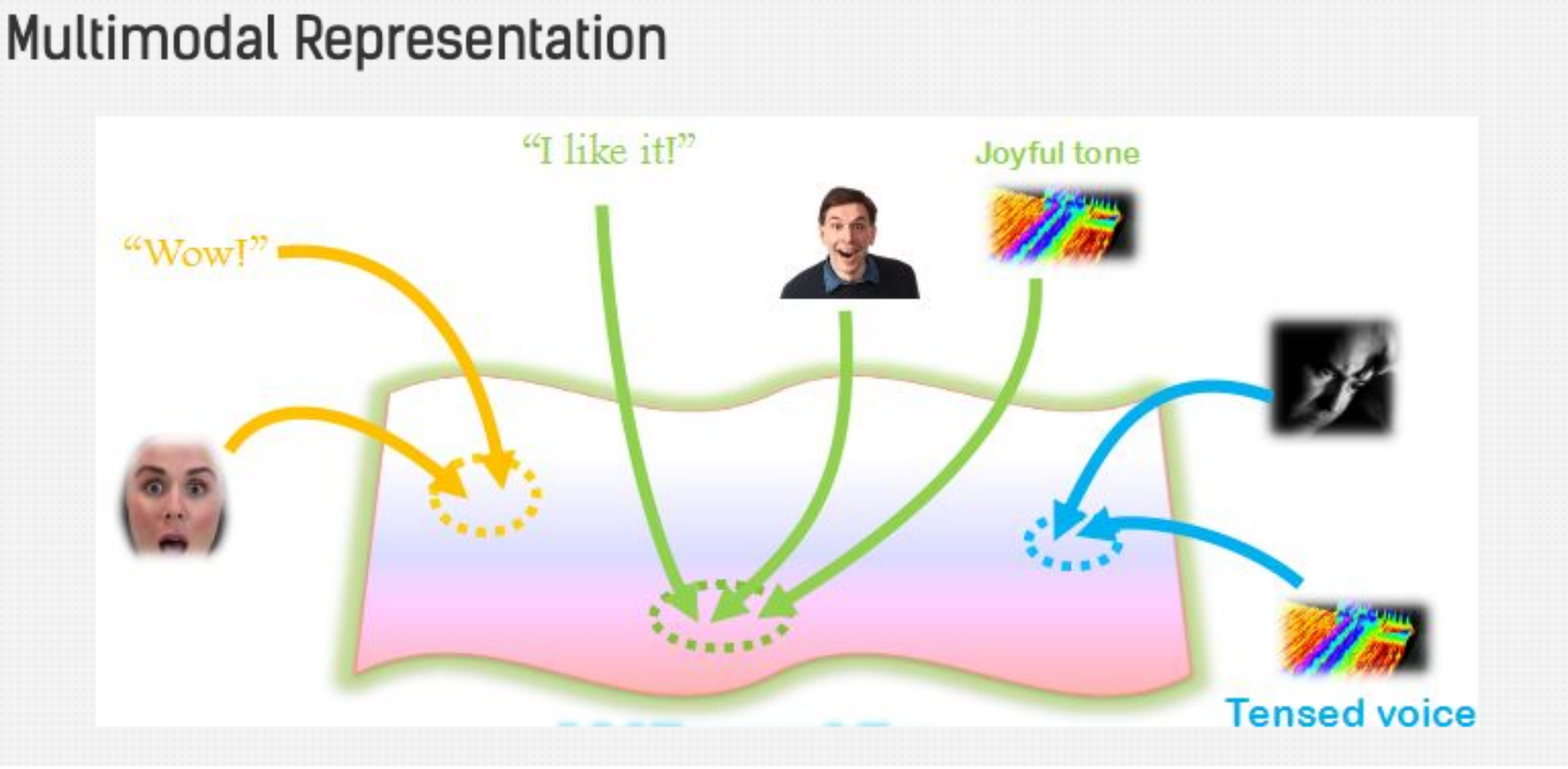
- **Text and Image Embeddings Combined into a Single Vector Space**:
  - Consider an application where we need to link textual descriptions with corresponding images, such as image captioning or visual question answering.
  - **Process**:
    - **Text Embeddings**: Words or sentences are encoded into vector representations using models like Word2Vec, GloVe, BERT, or Transformer-based models.
    - **Image Embeddings**: Images are encoded into vector representations using Convolutional Neural Networks (CNNs) or more advanced models like ResNet or Vision Transformers.
    - **Joint Embedding Space**: These text and image embeddings are mapped into a common vector space where semantically related text and image pairs are close to each other.

Contd...

Example in Practice:

- Using a model like CLIP (Contrastive Language–Image Pretraining), which learns to associate textual descriptions with images by training on a large dataset of text-image pairs.
- Result: A text describing an image (e.g., "a dog playing with a ball") and the image itself will have similar vector representations in the joint embedding space.

Schematic of Joint Representation



Techniques for Creating Joint Representations

- **Multimodal Deep Learning Models**:
  - Use neural networks designed to process and combine information from multiple modalities.
  - Common architectures include:
    - **Multimodal Transformers**: Incorporate self-attention mechanisms to handle multiple types of input data.
    - **Multimodal CNN-RNN Hybrids**: Use CNNs for processing images and RNNs (e.g., LSTMs, GRUs) for processing sequential data like text or audio.

Contd...

Shared Weights and Loss Functions:

- **Shared Weights**: In some models, parameters (weights) are shared across different modalities to learn a coherent joint representation.
- **Shared Loss Functions**: Use loss functions that enforce the learning of similar representations for semantically related data from different modalities.
- **Example**:
  - **Contrastive Loss**: Used in models like CLIP, where the loss function encourages the model to bring the embeddings of related text-image pairs closer and push apart unrelated pairs.
  - **Multimodal Variational Autoencoders (VAEs)**: Use a joint reconstruction loss to ensure the latent space captures information from all modalities.

Visual Semantic Spaces

- **Definition**: Spaces where visual information is embedded in a way that captures semantic relationships.
- **Example**:
  - Image embeddings that align with word embeddings in a shared space (e.g., using models like CLIP).
- **Applications**:
  - Image captioning.
  - Visual question answering.

Multimodal Autoencoder

- **Definition**: Autoencoders designed to learn joint representations by encoding and decoding information from multiple modalities.
- **Structure**:
  - **Encoder**: Maps input data from multiple modalities to a shared latent space.
  - **Decoder**: Reconstructs the original data from the shared latent space.
- **Applications**:
  - Data compression.
  - Cross-modal generation (e.g., generating images from text).

Multimodal Variational Autoencoders (VAEs)

- **Objective:** Capture information from multiple modalities in a shared latent space.
- **Approach:** Use a joint reconstruction loss to ensure the latent space effectively represents all modalities.

Latent Variable Model:

$$p(z, x_1, x_2, \ldots, x_M) = p(z) \prod_{m=1}^{M} p(x_m \mid z) \tag{1}$$

Evidence Lower Bound (ELBO):

$$\begin{aligned} \log p(x_1, x_2, \ldots, x_M) \geq \mathbb{E}_{q(z|x_1,x_2,\ldots,x_M)} \left[\log p(x_1, x_2, \ldots, x_M \mid z)\right] \\ - \mathrm{KL}\left(q(z \mid x_1, x_2, \ldots, x_M) \parallel p(z)\right) \end{aligned} \tag{2}$$

Contd...

Joint Reconstruction Loss:

$$\mathcal{L}_{\text{recon}} = \sum_{m=1}^{M} \mathbb{E}_{q(z|x_1,x_2,\ldots,x_M)} \left[\log p(x_m \mid z)\right] \tag{3}$$

Total Loss:

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \beta \cdot \text{KL}(q(z \mid x_1, x_2, \ldots, x_M) \parallel p(z)) \tag{4}$$

Key Points:

- The joint reconstruction loss ensures the latent space z captures information from all modalities.
- The KL divergence term regularizes the latent space to follow a prior distribution $p(z)$.
- The parameter β controls the trade-off between reconstruction and regularization.

Orthogonal Joint Representations

- **Definition**: Representations where different modalities contribute unique, non-redundant information.
- **Techniques**:
  - Regularization methods to enforce orthogonality.
  - Use of disentangled representations.
- **Applications**:
  - Enhancing interpretability.
  - Reducing redundancy in multimodal systems.

Techniques for Creating Orthogonal Joint Representations

- **Regularization Methods to Enforce Orthogonality**:
  - Introduce regularization terms in the loss function to penalize redundancy between modalities.
  - **Orthogonality Constraints**: Apply mathematical constraints to ensure that the learned representations of different modalities are orthogonal (i.e., their dot product is zero).
- **Use of Disentangled Representations**:
  - Train models to learn disentangled representations where each latent dimension captures independent factors of variation in the data.
  - Techniques include Variational Autoencoders (VAEs) and adversarial learning frameworks.

Regularization Terms in the Loss Function

- **Regularization Term**:
  - Add a term to the loss function that penalizes the dot product of the representations from different modalities.
- **Mathematical Formulation**:
  - Let z_1 and z_2 be the representations from two different modalities.
  - The orthogonality constraint can be expressed as:

$$z_1 \cdot z_2 = 0$$

  - To enforce this during training, we add a regularization term R to the loss function L:

$$R = \lambda \left| z_1 \cdot z_2 \right|$$

    where λ is a regularization coefficient that controls the strength of the penalty.
  - The total loss function becomes:

$$L_{\text{total}} = L + R = L + \lambda \left| z_1 \cdot z_2 \right|$$

Orthogonality Constraints

- **Mathematical Constraints**:
  - Enforce orthogonality by applying constraints directly to the representations.
  - For two representations z_1 and z_2, the orthogonality constraint can be defined as:

$$z_1^\top z_2 = 0$$

- **Implementation**:
  - During training, modify the gradient update step to incorporate the orthogonality constraint.
  - Use a Lagrange multiplier μ to enforce the constraint:

$$L_{\text{constrained}} = L + \mu(z_1^\top z_2)$$

  - Optimize $L_{\text{constrained}}$ using gradient descent or other optimization techniques.

Examples of Regularization Terms

- **Frobenius Norm Regularization**:
  - Use the Frobenius norm to penalize the inner product matrix of the representations.
  - For a batch of representations Z_1 and Z_2:

$$R = \lambda \left\| Z_1^\top Z_2 \right\|_F^2$$

    where $\|\cdot\|_F$ denotes the Frobenius norm.
- **Orthogonality Regularization for Neural Networks**:
  - Apply orthogonality regularization to the weight matrices W_1 and W_2 of neural networks processing different modalities.
  - Ensure the columns of W_1 and W_2 are orthogonal:

$$R = \lambda \left\| W_1^\top W_2 \right\|_F^2$$

Applications of Orthogonal Joint Representations

- **Enhancing Interpretability**:
  - By ensuring that each modality contributes unique information, orthogonal representations make it easier to interpret and understand the role of each modality in the final decision or prediction.
- **Reducing Redundancy in Multimodal Systems**:
  - Orthogonal representations minimize redundant information, leading to more efficient and compact multimodal systems.
  - This reduction in redundancy can improve the performance and robustness of multimodal models.

Regularization Methods

- **Orthogonality Constraints**:
  - Enforce constraints during training to ensure the representations from different modalities are orthogonal.
  - Example: Add a regularization term to the loss function that penalizes the dot product of the representations from different modalities.
- **Loss Function Regularization**:
  - Modify the loss function to include penalties for redundancy.
  - Example: Use mutual information loss to ensure each modality contributes distinct information.

Disentangled Representations

- **Variational Autoencoders (VAEs)**:
  - VAEs can be used to learn disentangled latent representations where each dimension captures a different factor of variation.
  - By encouraging the latent variables to be independent, VAEs help ensure that different modalities contribute unique information.
- **Adversarial Learning**:
  - Use adversarial training techniques to disentangle representations.
  - Example: Adversarial Autoencoders (AAEs) use adversarial loss to enforce disentanglement in the latent space.

Case Studies

- **Orthogonal Projections in Multimodal Fusion**:
  - Example: Using orthogonal projections in multimodal fusion tasks to ensure that image and text embeddings contribute non-overlapping information.
  - Results: Improved accuracy and interpretability in tasks like image captioning and visual question answering.
- **Disentangled Representations for Multimodal Data**:
  - Example: Applying VAEs to multimodal data to achieve disentangled representations that enhance the distinctiveness of each modality.
  - Results: More robust and generalizable models for applications such as cross-modal retrieval and multimodal generation.

Conclusion

- Orthogonal joint representations are crucial for ensuring that each modality in a multimodal system contributes unique and non-redundant information.
- Techniques like regularization and disentangled representations play a key role in achieving orthogonality.
- Applications in interpretability and redundancy reduction highlight the practical benefits of orthogonal joint representations in multimodal AI.

Component Analysis

- **Definition**: Techniques that decompose multimodal data into components that capture the underlying structure.
- **Techniques**:
  - Principal Component Analysis (PCA).
  - Independent Component Analysis (ICA).
- **Applications**:
  - Noise reduction.
  - Feature extraction.

Parallel Multimodal Representations

- **Definition**: Representations where different modalities are processed in parallel and then combined.
- **Techniques**:
  - Late fusion methods.
  - Attention mechanisms to weigh the importance of each modality.
- **Applications**:
  - Multimodal classification.
  - Multimodal retrieval.

Attention Mechanism in Multimodal Learning

- Multimodal learning involves combining information from multiple modalities (e.g., image and text) to improve performance.
- Attention mechanism allows the model to focus on relevant parts of the input data.
- In the context of image and text modalities, attention can be used to align image regions with corresponding textual descriptions.

Mathematical Formulation

Let I be the image feature matrix and T be the text feature matrix. The attention scores between image and text modalities can be calculated as:

$$\alpha_{i,j} = \text{softmax}(I_i^T W_a T_j)$$

where W_a is the attention weight matrix.
The weighted image and text features can then be calculated as:

$$\bar{I} = \sum_{j=1}^{N_t} \alpha_{i,j} I_i$$

$$\bar{T} = \sum_{i=1}^{N_i} \alpha_{i,j} T_j$$

where N_i and N_t are the number of image and text features, respectively.

Benefits of Attention Mechanism

- Attention mechanism allows the model to focus on relevant parts of the input data, improving performance.
- It can help the model to align image regions with corresponding textual descriptions, leading to a better understanding of the multimodal data.
- Attention mechanism can also be interpreted as a form of explanation, showing which parts of the input data are most important for the model's predictions.

Similarity Metrics

- **Definition**: Metrics used to measure the similarity between multimodal representations.
- **Examples**:
  - Cosine similarity.
  - Euclidean distance.
- **Applications**:
  - Cross-modal retrieval.
  - Multimodal clustering.

Canonical Correlation Analysis (CCA)

- **Definition**: A technique that finds linear projections of two modalities such that the projected data is maximally correlated.
- **Process**:
  - Computes canonical variables for each modality.
  - Maximizes the correlation between these variables.
- **Applications**:
  - Multimodal feature learning.
  - Cross-modal retrieval.

Process of CCA

- **Step 1: Compute Canonical Variables**
  - Let X and Y be the data matrices for the two modalities, with n samples each.
  - Find projection vectors w_x and w_y such that:

$$u = Xw_x$$

$$v = Yw_y$$

    where u and v are the canonical variables.
- **Step 2: Maximize Correlation**
  - Maximize the correlation ρ between the canonical variables u and v:

$$\rho = \frac{u^\top v}{\sqrt{u^\top u v^\top v}}$$

  - This leads to the optimization problem:

$$(w_x, w_y) = \arg\max \frac{w_x^\top X^\top Y w_y}{\sqrt{w_x^\top X^\top X w_x}\sqrt{w_y^\top Y^\top Y w_y}}$$

Applications of CCA

- **Multimodal Feature Learning**:
  - CCA can be used to learn shared representations for different modalities.
  - Example: Learning joint embeddings for text and image data.
- **Cross-Modal Retrieval**:
  - Use CCA to retrieve relevant information from one modality using queries from another modality.
  - Example: Retrieving images based on textual descriptions.

Example of CCA

- **Dataset**:
  - Suppose we have a dataset of images X and corresponding textual descriptions Y.
- **Step-by-Step Process**:
  1. Compute the covariance matrices Σ_{XX}, Σ_{YY}, and Σ_{XY}.
  2. Solve the generalized eigenvalue problem:

$$\Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{YX}w_x = \lambda\Sigma_{XX}w_x$$

  3. The eigenvectors w_x and w_y provide the projection directions.
  4. Project the original data onto these directions to obtain the canonical variables.
  5. Use these variables for tasks such as cross-modal retrieval or feature learning.

Mathematical Details

- **Covariance Matrices**:
  - Compute covariance matrices:

$$\Sigma_{XX} = \frac{1}{n-1} X^\top X$$

$$\Sigma_{YY} = \frac{1}{n-1} Y^\top Y$$

$$\Sigma_{XY} = \frac{1}{n-1} X^\top Y$$

- **Generalized Eigenvalue Problem**:
  - Solve the eigenvalue problem to find w_x:

$$\Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX} w_x = \lambda \Sigma_{XX} w_x$$

  - Similarly, solve for w_y:

$$\Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY} w_y = \lambda \Sigma_{YY} w_y$$

Contd...

Canonical Variables:

- Project the data onto the canonical directions:

$$u = Xw_x$$

$$v = Yw_y$$

CCA Summary

- CCA is a powerful technique for finding linear projections that maximize the correlation between two modalities.
- It is widely used in multimodal feature learning and cross-modal retrieval.
- Understanding the mathematical foundation of CCA helps in effectively applying it to various multimodal AI tasks.

Summary

- Multimodal representations are crucial for integrating and leveraging information from diverse data sources.
- Techniques like joint representations, multimodal autoencoders, and canonical correlation analysis enable effective multimodal learning.
- Applications span various domains, enhancing tasks such as image captioning, video analysis, and cross-modal retrieval.

Cross-Modal Retrieval

Introduction to Cross-Modal Retrieval

- **Definition**:
  - Cross-modal retrieval refers to the task of retrieving data from one modality (e.g., images) using a query from another modality (e.g., text).
- **Significance**:
  - Enables effective information retrieval across different types of data.
  - Enhances user experience by allowing flexible querying.
- **Applications**:
  - Image search using textual descriptions.
  - Video retrieval using audio or textual queries.
  - Multimodal recommendation systems.

Core Components of Cross-Modal Retrieval

- **Feature Extraction**:
  - Extract meaningful features from each modality.
  - Techniques include Convolutional Neural Networks (CNNs) for images and Transformers for text.
- **Joint Embedding Space**:
  - Map features from different modalities into a shared space.
  - Ensure semantically similar content from different modalities is close in this space.
- **Similarity Metrics**:
  - Measure similarity between query and data points in the joint space.
  - Common metrics include cosine similarity and Euclidean distance.

Techniques for Cross-Modal Retrieval

- **Canonical Correlation Analysis (CCA)**:
  - Finds linear projections of two modalities that are maximally correlated.
- **Deep Neural Networks**:
  - Use deep learning models to learn complex, non-linear mappings between modalities.
  - Examples include joint embedding models like CLIP.
- **Contrastive Learning**:
  - Train models to minimize the distance between related cross-modal pairs and maximize the distance between unrelated pairs.
  - Example: Contrastive loss used in CLIP.
- **Multimodal Autoencoders**:
  - Learn joint representations by reconstructing input data from multiple modalities.

Case Study: CLIP (Contrastive Language–Image Pretraining)

- **Overview**:
  - Developed by OpenAI to connect text and images through a shared embedding space.
  - Trained on a large dataset of text-image pairs.
- **Methodology**:
  - **Text Encoder**: Uses a Transformer to encode text descriptions into embeddings.
  - **Image Encoder**: Uses a CNN (e.g., ResNet) to encode images into embeddings.
  - **Contrastive Loss**: Encourages embeddings of related text-image pairs to be close and unrelated pairs to be far apart in the shared space.
- **Applications**:
  - Image search: Retrieve images based on textual queries.
  - Zero-shot learning: Perform tasks without direct training on specific datasets.

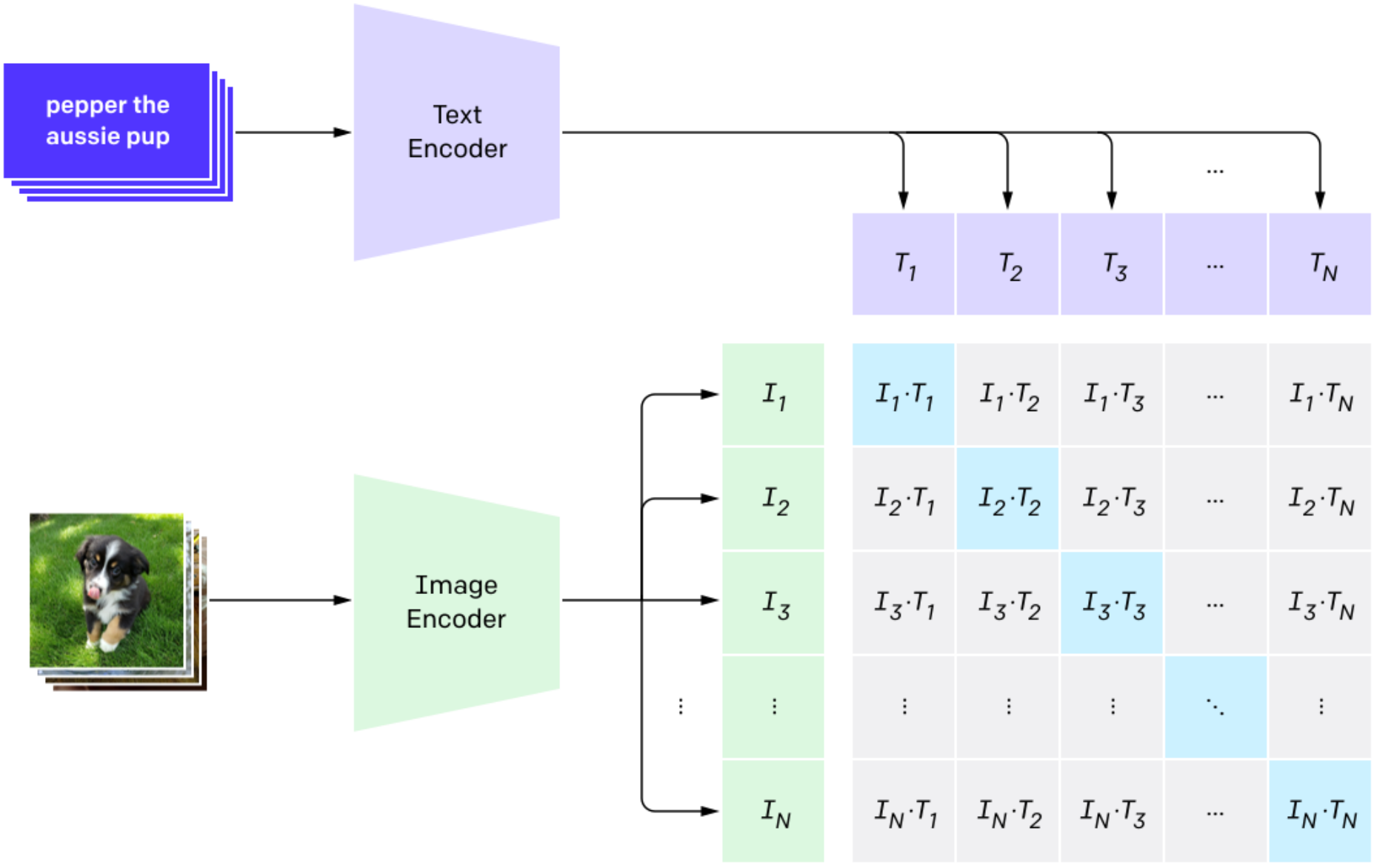
Introduction

- CLIP is a model developed by OpenAI that learns visual concepts from natural language descriptions.
- It can understand images and text in a unified manner.
- CLIP is trained on a large dataset of images paired with text descriptions.

Architecture

- CLIP consists of two main components:
  - **Image Encoder**: Processes images and extracts visual features.
  - **Text Encoder**: Processes text and extracts semantic features.
- Both encoders map inputs to a shared embedding space.

1. Contrastive pre-training



Training Process

- CLIP is trained using a contrastive learning approach.
- Given a batch of images and their corresponding text descriptions:
  - The model maximizes the similarity between the correct image-text pairs.
  - It minimizes the similarity between incorrect pairs.
- The loss function used is the InfoNCE loss.

Overview of the Loss Function

- The loss function in CLIP is designed to maximize the similarity between correct image-text pairs.
- It minimizes the similarity between incorrect pairs.
- The loss function is symmetric and operates on a similarity matrix computed from the embeddings of images and text.

Similarity Matrix

- Given a batch of N image-text pairs, a similarity matrix C is computed:

$$C = \frac{1}{\tau}(E_t \cdot E_a^T)$$

  where:
  - E_t = text embeddings
  - E_a = image embeddings
  - τ = temperature parameter
- C is an $N \times N$ matrix representing the pairwise similarities.

Loss Function Components

- The loss function is defined as:

$$\mathcal{L} = \frac{1}{2}(\ell_{text}(C) + \ell_{image}(C))$$

  where:
  - $\ell_{text}(C)$ computes the loss for text embeddings.
  - $\ell_{audio}(C)$ computes the loss for image embeddings.
- Each component uses the softmax function to derive probabilities from the similarity scores.

Softmax Function

The loss is computed using the softmax function:

- For text loss:

$$\ell_{text}(C) = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp(C_{i,i})}{\sum_{j=1}^{N}\exp(C_{i,j})}\right)$$

- For image loss:

$$\ell_{audio}(C) = -\frac{1}{N}\sum_{j=1}^{N}\log\left(\frac{\exp(C_{j,j})}{\sum_{i=1}^{N}\exp(C_{j,i})}\right)$$

Contrastive Learning Objective

- The objective of the loss function is to:
  - Increase the similarity of the correct image-text pairs (diagonal elements).
  - Decrease the similarity of incorrect pairs (off-diagonal elements).
- This is achieved through the softmax operation, which normalizes the similarity scores.

Summary

- The CLIP loss function effectively learns to associate images with their corresponding text descriptions.
- By maximizing the similarity of correct pairs and minimizing incorrect ones, CLIP achieves robust zero-shot learning capabilities.
- Future work may focus on refining the loss function to improve performance in more complex tasks.

Applications

- **Zero-Shot Learning**: CLIP can classify images without specific training on those classes.
- **Image Search**: Users can search for images using natural language queries.
- **Content Generation**: CLIP can be combined with generative models for creative applications.

Limitations

1. While CLIP usually performs well on recognizing common objects, it struggles on more abstract or systematic tasks such as counting the number of objects in an image, and on more complex tasks such as predicting how close the nearest car is in a photo.
2. CLIP also still has a poor generalization to images not covered in its pre-training dataset. For instance, although CLIP learns a capable OCR system when evaluated on handwritten digits from the MNIST dataset, zero-shot CLIP only achieves 88% accuracy, well below the 99.75% of humans on the dataset.
3. Finally, we've observed that CLIP's zero-shot classifiers can be sensitive to wording or phrasing and sometimes require trial and error "prompt engineering" to perform well.

Conclusion

- CLIP represents a significant advancement in understanding the relationship between text and images.
- Its versatility allows it to be applied in various domains, from image classification to content generation.
- Future work includes improving its robustness and expanding its applications.

Challenges and Future Directions

- **Data Alignment**:
  - Ensuring data from different modalities is accurately aligned.
- **Scalability**:
  - Handling large-scale datasets with diverse modalities.
- **Generalization**:
  - Ensuring models generalize well to unseen data and tasks.
- **Interpretability**:
  - Understanding how models learn and represent cross-modal relationships.
- **Future Directions**:
  - Improved multimodal fusion techniques.
  - Enhanced training strategies for better cross-modal understanding.
  - Development of more sophisticated and robust joint embedding models.

Conclusion

- Cross-modal retrieval is a crucial aspect of multimodal AI, enabling flexible and effective information retrieval across different types of data.
- Techniques such as deep learning, contrastive learning, and multimodal autoencoders play key roles in developing efficient cross-modal retrieval systems.
- Despite challenges, ongoing research and advancements continue to enhance the capabilities and applications of cross-modal retrieval.