Tutorial 10
CLIP: Learning Transferable Visual Models From Natural Language Supervision
CSC412: Probabilistic Machine Learning

Week of March 10, 2025
Jerry Ji

Overview

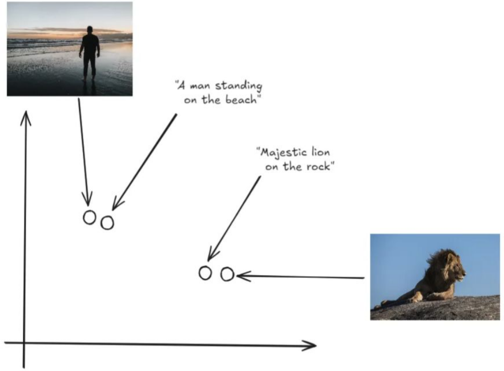
- Motivation and Background
    - Why OpenAI creates CLIP (Contrastive Language-Image Pre-training)?
- Overview of CLIP Approach
- Model Architecture
    - Text and Image Encoder
    - Joint embedding space
- Contrastive Learning
    - Loss and Pseudo Code
- Dataset and Training Details
- Zero-shot Inference
- Summary
    - Key Advances of CLIP
    - Comparison with prior works
    - Future Directions

Motivation & Background

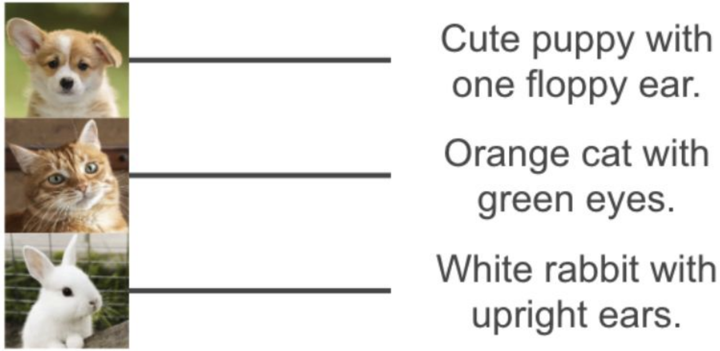
Let’s go back to Mid-2020

- Limitations of traditional supervised vision models
  - fixed label sets
  - dataset-specific training
- Natural language supervision offers a scalable, rich source of training data.
  - (Transformer, BERT, GPT3 were out, nlp were at its prime time)
  - Instead of collecting/annotating labels, obtaining text description associate with each image
- Goal: Learn transferable visual representations without massive manual annotations

Overview of the CLIP Approach



- Learns a shared embedding space for images and text.
  - We seems to make great progress on Text side
  - How can we bring that to imaging world?
- Joint training of an image encoder and a text encoder on 400M (image, text) pairs.
  - Large easier obtainable data pairs instead of tedious human-labeling
  - Joint training with contrastive learning
- Enables zero-shot learning by using natural language to describe tasks.



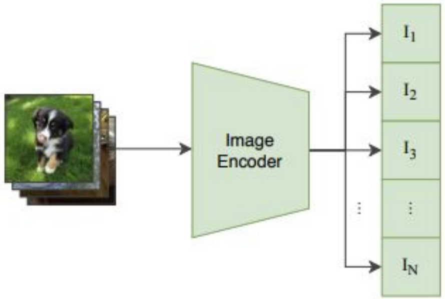
Cute puppy with one floppy ear.

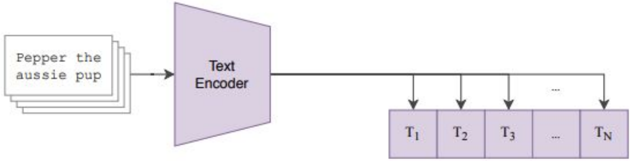
Orange cat with green eyes.

White rabbit with upright ears.

CLIP Architecture – Dual Encoders

- Two main components:
  - Image Encoder: Can be a modified ResNet (e.g., RN50, RN101) or a Vision Transformer (ViT)
  - Text Encoder: A Transformer (with modifications like byte pair encoding (BPE) tokenization)





- Both outputs are linearly projected into a common multimodal embedding space.

Image Encoder Details

- ResNet modifications
    - attention pooling replacing global average pooling, antialiased blur pooling

- Alternative: Vision Transformer with additional layer normalization

- Focus on scalability and computational efficiency.

Text Encoder Details

- Transformer architecture: 12 layers, 63M parameters, 8 attention heads.
  - GPT-3: 96 layers, 175B parameters, 96 attention heads, 12,288 hidden dimension, 2,048 token context window, trained on 570GB of text data.

- Processes lower-cased, BPE-tokenized text (vocab size ≈49,152).

- Uses the [EOS] token representation, which is layer-normalized and linearly projected

Joint Multimodal Embedding Space

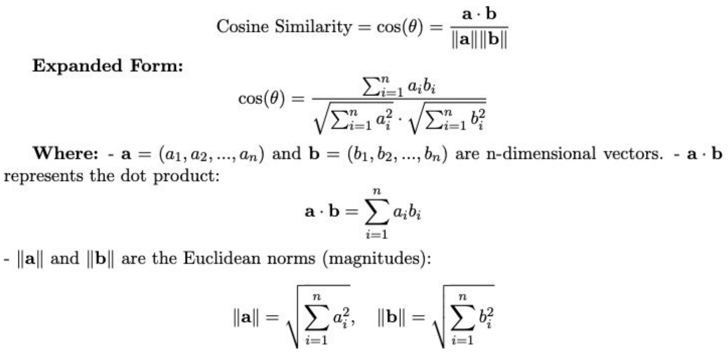
- Both encoders produce L2-normalized feature vectors.

- Linear projection maps these features into a shared embedding space.

- Cosine similarity (scaled by a learned temperature parameter) measures image–text compatibility.

Pre-training Objective – Contrastive Learning

- Contrastive loss: match correct image–text pairs and separate incorrect ones.



$$\text{Cosine Similarity} = \cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}$$

Expanded Form:

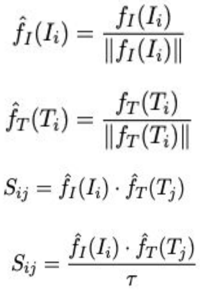
$$\cos(\theta) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \cdot \sqrt{\sum_{i=1}^{n} b_i^2}}$$

Where: - $\mathbf{a} = (a_1, a_2, ..., a_n)$ and $\mathbf{b} = (b_1, b_2, ..., b_n)$ are n-dimensional vectors. - $\mathbf{a} \cdot \mathbf{b}$ represents the dot product:

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{n} a_i b_i$$

- $\|\mathbf{a}\|$ and $\|\mathbf{b}\|$ are the Euclidean norms (magnitudes):

$$\|\mathbf{a}\| = \sqrt{\sum_{i=1}^{n} a_i^2}, \quad \|\mathbf{b}\| = \sqrt{\sum_{i=1}^{n} b_i^2}$$



$$\hat{f}_I(I_i) = \frac{f_I(I_i)}{\|f_I(I_i)\|}$$

$$\hat{f}_T(T_i) = \frac{f_T(T_i)}{\|f_T(T_i)\|}$$

$$S_{ij} = \hat{f}_I(I_i) \cdot \hat{f}_T(T_j)$$

$$S_{ij} = \frac{\hat{f}_I(I_i) \cdot \hat{f}_T(T_j)}{\tau}$$

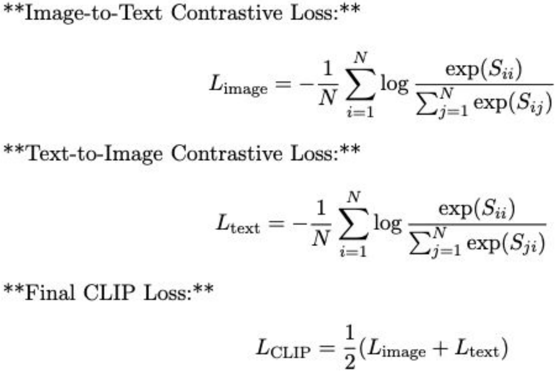


Image-to-Text Contrastive Loss:

$$L_{\text{image}} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(S_{ii})}{\sum_{j=1}^{N} \exp(S_{ij})}$$

Text-to-Image Contrastive Loss:

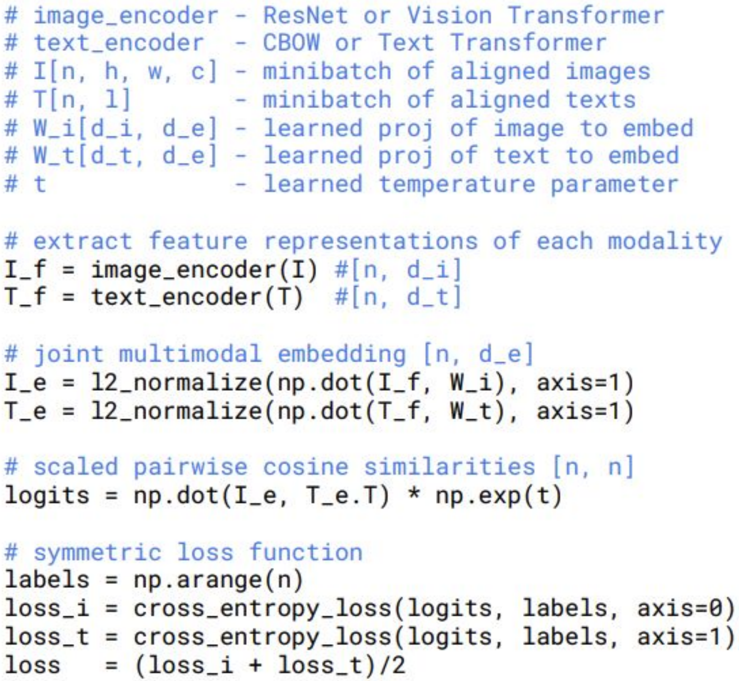
$$L_{\text{text}} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(S_{ii})}{\sum_{j=1}^{N} \exp(S_{ji})}$$

Final CLIP Loss:

$$L_{\text{CLIP}} = \frac{1}{2}(L_{\text{image}} + L_{\text{text}})$$

- Within a batch of N pairs, there are N² potential comparisons.
- Uses symmetric cross-entropy loss for image-to-text and text-to-image predictions.

Pseudocode & Loss Function



```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

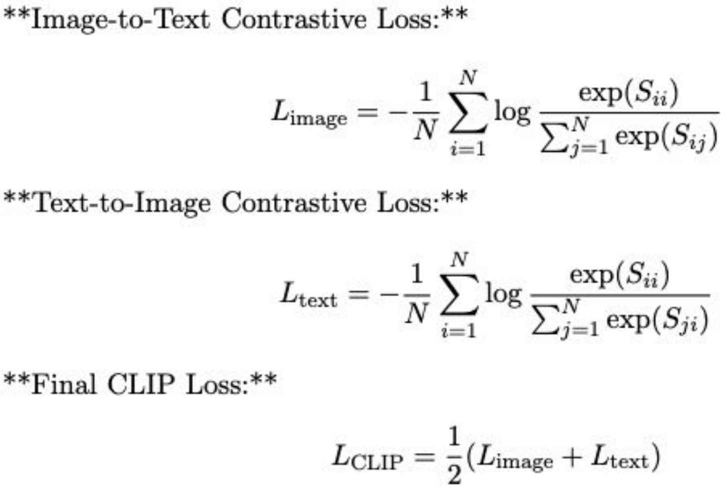


Image-to-Text Contrastive Loss:

$$L_{\text{image}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(S_{ii})}{\sum_{j=1}^{N}\exp(S_{ij})}$$

Text-to-Image Contrastive Loss:

$$L_{\text{text}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(S_{ii})}{\sum_{j=1}^{N}\exp(S_{ji})}$$

Final CLIP Loss:

$$L_{\text{CLIP}} = \frac{1}{2}(L_{\text{image}} + L_{\text{text}})$$

Pre-training Dataset & Data Augmentation

- Utilizes 400 million (image, text) pairs collected from the web.

- Simple image augmentation: random square crops on resized images.

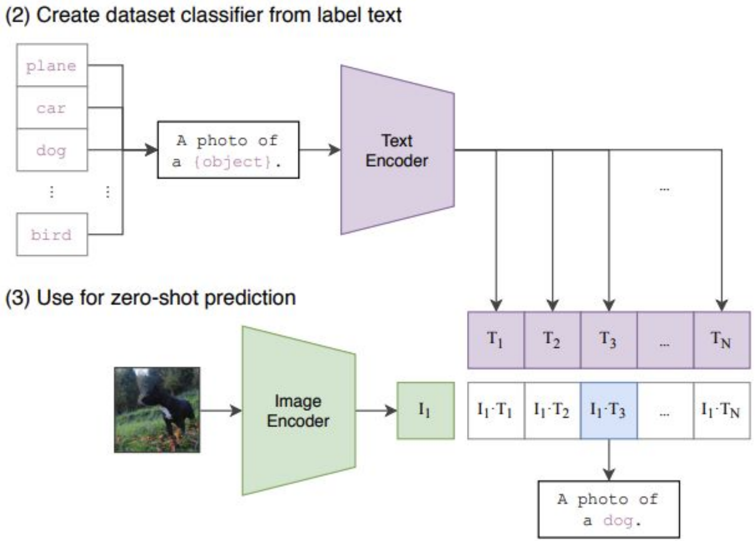
- Diverse natural language captions provide supervision beyond fixed labels.

Training Details & Compute Considerations

- Optimizer: Adam with decoupled weight decay and cosine learning rate schedule.
- Very large minibatch size (e.g., 32,768) and mixed precision training. ***
- Scalable across various models (RN50, RN50x16, ViT variants) for high compute efficiency.

Zero-shot Transfer & Prompt Engineering

- Zero-shot classification: Generate text embeddings for class names or descriptions.
- Use prompt templates like “A photo of a {label}.” to provide context.
- Achieves competitive performance without any dataset-specific training.



Key Advances of CLIP

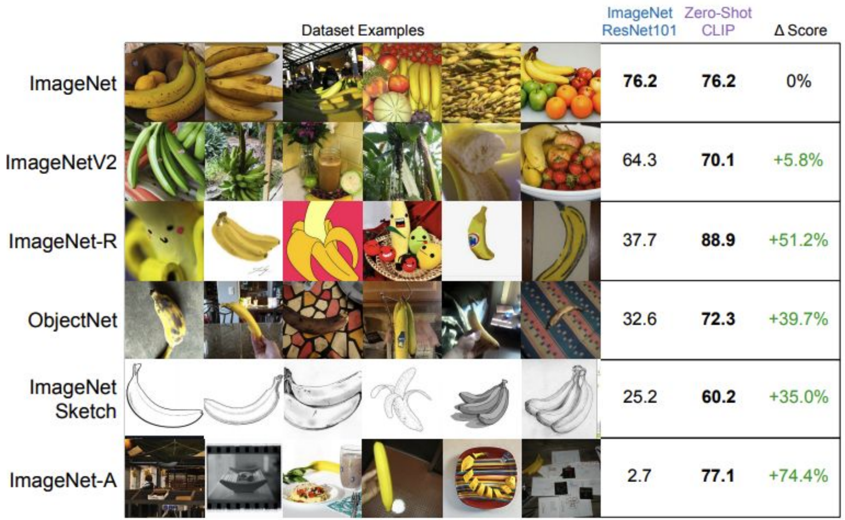
- Scalability: Efficient training on a massive dataset using contrastive learning

- Flexibility: Ability to generalize across tasks with zero-shot transfer

- Robustness: Better handling of distribution shifts compared to traditional supervised models

Comparison to Previous Approaches & Impact

- Contrast with traditional ImageNet supervised learning and prior zero-shot approaches (e.g., Visual N-Grams)
- Significant improvements in zero-shot accuracy (e.g., ImageNet performance)
- Broader impact: Influencing future research in multi-modal learning and zero-shot transfer ***


Dataset Examples

	ImageNet ResNet101	Zero-Shot CLIP	Δ Score
ImageNet	**76.2**	**76.2**	0%
ImageNetV2	64.3	**70.1**	+5.8%
ImageNet-R	37.7	**88.9**	+51.2%
ObjectNet	32.6	**72.3**	+39.7%
ImageNet Sketch	25.2	**60.2**	+35.0%
ImageNet-A	2.7	**77.1**	+74.4%

Limitations

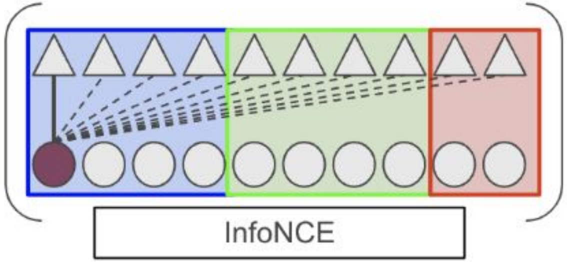
“On most of these datasets, the performance of this baseline is now well below the overall state of the art. Significant work is still needed to improve the task learning and transfer capabilities of CLIP. While scaling has so far steadily improved performance and suggests a route for continued improvement, we estimate around a 1000x increase in compute is required for zero-shot CLIP to reach overall state-of-the-art performance. This is infeasible to train with current hardware. Further research into improving upon the computational and data efficiency of CLIP will be necessary.”

Conclusion & Future Directions

- Recap: Dual encoder architecture with contrastive pre-training enables zero-shot transfer.
    - (draw the learning on the board)

- Highlights: Scalability, flexibility, and robustness achieved through natural language supervision.

- Future Directions: Explore improved few-shot learning, additional modalities, and enhanced prompt engineering.


InfoNCE