0.1.1 CLIP

0.1.1.1 Method

CLIP is a method developed by OpenAI to train a vision-language model using contrastive learning. CLIP stands for (**C**ontrastive **L**anguage-**I**mage **P**retraining). The architecture consists of a separate image encoder $f(\cdot)$ and text encoder $g(\cdot)$, both of which can be any architecture, and a linear projection (linear layer without bias and activation function) on top of the modality-specific encoders.

The forward pass works as follows: For a batch of image-text pairs, the images $\left\{\boldsymbol{H}_{(v,0),i}\right\}_{i=1}^{B}$ (B denotes the batch size) are passed through the image encoder, resulting in an image representation $\boldsymbol{I} \in \mathbb{R}^{B \times D}$. Similarly, the texts $\left\{\boldsymbol{H}_{(w,0),i}\right\}_{i=1}^{B}$ are passed through the text encoder, producing a text representation T. Recall that $\boldsymbol{I}$ and $\boldsymbol{T}$ correspond to the batched representations of the [`I_CLS`] and [`T_CLS`] tokens, as defined in (TODO: cite equ for I) and (TODO: cite equ for T), respectively.

Both the image and text representations produced by the encoders are in separate embedding spaces — one for images and one for text - they are not related to each other initially. However, for contrastive learning to be effective, the embeddings should exist in the same latent space. After all, the embedding for an image and its corresponding text should be the same (or at least very close to each other).

In SHRE, discussed in the previous section, this shared latent space is achieved through a shared encoder on top of the modality-specific encoders, and through a ranking loss [1]. CLIP maps the image and text representations into a shared latent space using linear projections $\boldsymbol{o}_v$ and $\boldsymbol{o}_w$ for image and text, respectively. These linear projections allow the model to map the image and text embeddings in a shared latent space, which is ensured by the contrastive loss. Note that the linear projections $\boldsymbol{o}_v$ and $\boldsymbol{o}_w$ can also be defined as functions, but for consistency with the original paper we use dot product notation, as shown in the following.

The image representation in the shared embedding space is denoted as $\boldsymbol{I}' = \|\boldsymbol{o}_v \boldsymbol{I}^T\|_2$, and the text representation as is given by $\boldsymbol{T}' = \|\boldsymbol{o}_w \boldsymbol{T}^T\|_2$. Since cosine similarity is used as the similarity metric in the contrastive loss, the embeddings are normalized, which is indicated by the l2 norm $\|\cdot\|_2$ around the result of the linear projections. It is important to note that the superscript T denotes the transpose of a matrix, not the batch of text representations.

Then, it is sufficient to perform matrix multiplication of the normalized representations in order to compute the cosine similarity between each pair. The result is given by:

$$\boldsymbol{L} = \exp(t) * \boldsymbol{I}' \boldsymbol{T}'^T, \boldsymbol{L} \in \mathbb{R}^{B \times B} \tag{1}$$

The opteration is quite similar to the batched cosine similarity operation introduced in (TODO: cite vision-lang-contrast). However, it is notable that the cosine similarities $\boldsymbol{L}$ are scaled by $\exp(t)$, where t is a temperature parameter. This parameter is used to control the smoothness of the softmax function, and is a scalar applied element-wise to the cosine similarities, which should be a familiar concept from knowledge distillation (TODO: cite KD section).

In knowledge distillation, the temperature was introduced as a tunable hyperparameter [2], [1]. However, in CLIP it is a learnable parameter that is optimized during training, just like any other parameter in the model, eliminating the need for manual tuning. The temperature t is optimized in log-space, which is why the actual temperature by which logits are scaled, is given by $\exp(t)$ [3].

Although the authors did not provide a specific reason for the optimization in log-space, it is likely that this approach ensures that the temperature is always positive, since $\exp(t)$ always returns a

positive value. Optimizing in log-space may also contribute to greater numerical stability (the logarithm grows at a low rate), resulting in less drastic changes in the temperature during optimization and thereby making training more stable.

In the matrix $\boldsymbol{L}$, the cosine similarity between image i and text j in the batch is denoted by $\boldsymbol{L}_{i,j}$, where the diagonal elements contain the similarity for positive pairs. To maximize the similarity between positive pairs (i, i), and minimize the similarity between negative pairs (i, j), with $i \neq j$, cross-entropy loss is used.

The loss for selecting the correct caption for each image and vice versa is exacly the same as given in (TODO: cite vision-lang-contrast i2t) and (TODO: cite vision-lang-contrast t2i), respectively. The final loss of CLIP is the vision-language contrastive loss, given in (TODO: cite vision-lang-contrast).

$$\mathcal{L}_{\text{CLIP}} = \mathcal{L}_{\text{CL}} = \frac{1}{2} * \left(\mathcal{L}_{\text{CL}}^{\text{i2t}} + \mathcal{L}_{\text{CL}}^{\text{t2i}}\right) \tag{2}$$

CLIP only relies on contrastive learning to train a vision-language model, and therefore requires a high batch size to achieve good results. The authors use a very large batch size of 32,768 [3]. An abstract illustration of the end-to-end training process of CLIP is shown in (TODO: cite figure) in the Appendix.

0.1.1.2 Zero-Shot Image Classification

What makes CLIP special is its method of zero-shot image classification using the trained model. This capability is achieved through prompt engineering on the text encoder. For each class in the dataset, where image classification is desired, the name of the class is injected into a prompt template. The prompt template follows a structure like this: “a photo of a {class name}.”.

CLIP uses 80 different prompts, so for each class in the dataset, 80 distinct prompts are generated (similar to the example shown above). These 80 prompts are passed through the text encoder and text projection, resulting in 80 different text embeddings for one class. These embeddings are then averaged and normalized, yielding a single embedding per class. This embedding captures the semantic meaning of the class name, which the model learned through contrastive pretraining.

To classify an image, the image is passed through the image encoder and image projection, resulting in an image embedding. The cosine similarity between this image embedding and all class embeddings is calculated. The class corresponding to the text embedding with the highest similarity to the image representation is predicted as the class for the image, as demonstrated in Figure 1.

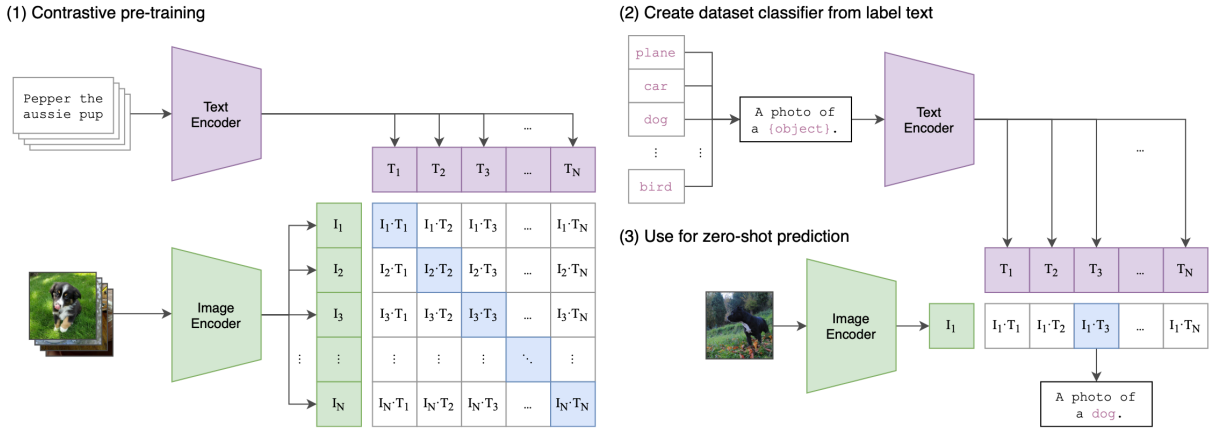

Figure 1: For zero-shot image classification, CLIP uses prompt engineering to create one classifier per image class to predict (2). The class whose classifier has the highest similarity (cosine) with the image representation is the predicted class (3) for the image [3].

The approach reaches a zero-shot accuracy of 76.2% on the validation set of ImageNet-1K [4], with a top-5 accuracy of 95% [3]. This is particularly impressive given that the model has never seen any images from the ImageNet-1K dataset during training, nor has it been trained on any image classification task. It merely achieves this accuracy through its cross-modal understanding between text and image. The model effectively "knows" how the ImageNet-1K classes look visually.

However, it is important to note that these results were based on a vision Transformer following the ViT-L/14@336px architecture for the image encoder. This architecture consists of 24 layers, 16 attention heads, a hidden size of 1024, and processes images at a resolution of 336x336 [3]. For the text encoder, a 12-layer Transformer was used, consisting of 12 attention heads and a hidden size of 768 [3]. According to HuggingFace, the model is 428 million parameters large[1]. Additionally, the model was trained on a custom dataset specifically developed for CLIP, consisting of 400 million image-text pairs [3].

Bibliography

[1] Y. Aytar, C. Vondrick, and A. Torralba, "See, Hear, and Read: Deep Aligned Representations," *arXiv preprint arXiv:1706.00932*, 2017, [Online]. Available: https://arxiv.org/abs/1706.00932

[2] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge Distillation: A Survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021, doi: 10.1007/s11263-021-01453-z.

[3] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, M. Meila and T. Zhang, Eds., in Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 8748–8763.

[4] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

[1] https://huggingface.co/openai/clip-vit-large-patch14