CLOOB: Modern Hopfield Networks with InfoLOOB Outperform CLIP

This blog post talks about CLOOB [1], which is a model for zero-shot classification, i.e., predicting classes the model has never seen before, created by the Institute of Machine Learning at JKU Linz. CLOOB builds and improves upon OpenAI's CLIP [2], but integrates modern Hopfield networks to deal with a problem that is inherent to CLIP.

In the post, we will outline the background, we will talk about the issue with CLIP, how modern Hopfield networks can help with it, and why we need a new objective function when using modern Hopfield networks.

For more on contrastive learning you can check out our blog post about a contrastive learning-based model for prediction of enzyme function.

Motivation and Background

Why

In 2021 OpenAI introduced CLIP (Contrastive Language-Image Pretraining) with great success in the field of zero-shot classification. CLIP was used in the training of DALL-E [3], which is a generative model by OpenAI, that creates images based on text prompts.

The idea and purpose behind CLIP, and thus by extension CLOOB, is that we show the model an image together with a bunch of text prompts and then the model gives us the probability for each of the text prompts to belong to the image respectively, i.e., are a description of its content. This means the model must deal with multimodal data, instead of just images or just text, it works with both images and text simultaneously.

How

To do this, both models employ vision and language features extractors. In the case of CLIP, the feature extractor is either a ResNet or a Vision Transformer for the images and a Continuous Bag of Words Model or a Text Transformer for the prompts. In the case of CLOOB ResNET and a Text Transformer fulfilling the same task. After the features have been extracted the image and text features get embedded into the same embedding space by a linear projection matrix. Other authors have suggested using non-linear embeddings [4][5] but for CLIP and CLOOB their respective authors stick to linear embeddings. Furthermore, all samples have a norm of 1 in the embedding space, which is why we refer to it as the "embedding sphere".

The most exciting and useful part is that the model gets pretrained on a dataset and then applied to a different dataset without any finetuning. This is referred to as zero-shot transfer

learning. To achieve this the model has to generalize well and develop a deeper understanding of the objects depicted and described. Contrastive learning was chosen by the authors of CLIP since it has shown promising results in the previous years.

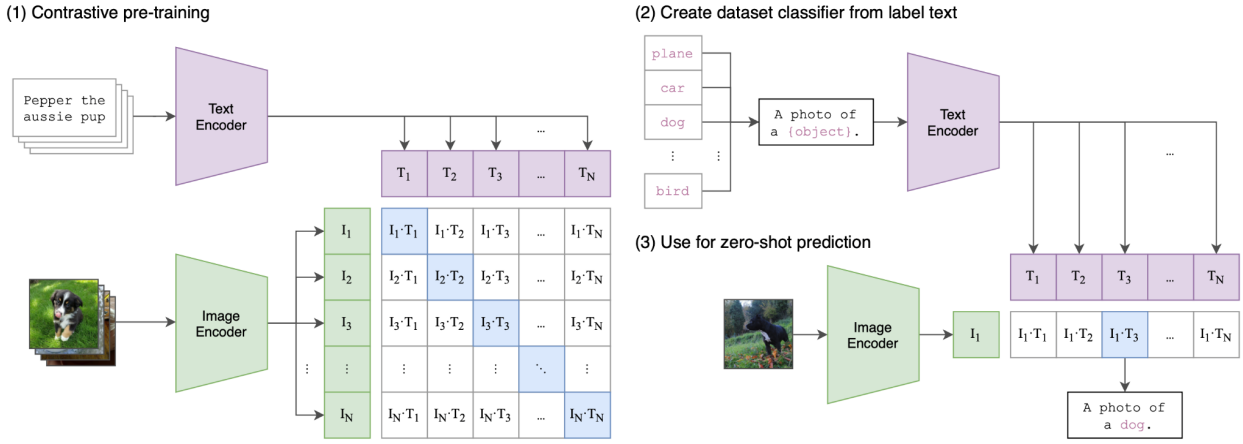

Figure 1: CLIP architecture. CLIP trains jointly an image encoder and a text encoder to predict the right pairing of a batch of image-text pairs. At inference time the model predicts the most likely prompt from a set of prompts corresponding to a given image. Figure adapted from [6], with permission of the authors.

Contrastive Learning

Contrastive learning is a self-supervised learning technique. This means that we do not have training labels and instead rely on the model being able to learn to associate similar samples and distinguish dissimilar ones. A popular, groundbreaking, and fairly easy-to-understand framework for this kind of learning is SimCLR [4], which might be presented in a future blog post.

In contrastive learning, we always have an anchor to which one or more positive samples are compared as well as one or more negative samples. In CLIP (and CLOOB) the anchor is either an image or a prompt, the positive sample is its corresponding text prompt or image, and the negative samples are all other prompts or images. This can be seen in the CLIP architecture, where the diagonal elements correspond to the matched pairs, and off-diagonal elements to the images and prompts that do not belong together.

A Limitation with CLIP

CLIP has been labeled as a “foundation model” [7] and has achieved astonishing results in the field of zero-shot classification and contrastive learning, sometimes even outperforming its supervised peers. But one issue remains the problem of “explaining away”.

Explaining away is known in reasoning as the concept of accepting one cause of an event and dismissing all other causes [8]. This problem is caused by insufficiently extracting co-occurrences and covariance from the multimodal data.

Modern Hopfield Networks

What do they do?

Modern Hopfield networks (MHNs) are a form of associative memory. MHNs are a type of neural network that is able to store multiple patterns. During the inference phase, one can input a partial or noisy pattern, and subsequently retrieve the stored pattern that most closely resembles it.

Why does this help?

What the use of MHNs does is that it amplifies the co-occurrence and covariance structure of the data. The Hopfield network stores reference embeddings which are later retrieved and by the nature of the network embeddings (samples) that are close to each other get averaged, which leads to a richer co-occurrences and covariance structure. Additionally, spurious, i.e., noisy, features get “filtered out” by this averaging operation.

This effect can be observed in the following graphic. For demonstrational purposes, only the network features corresponding to a teacup, a teapot, a piece of cake, and a fork are arranged, such that when activated they form the corresponding objects.

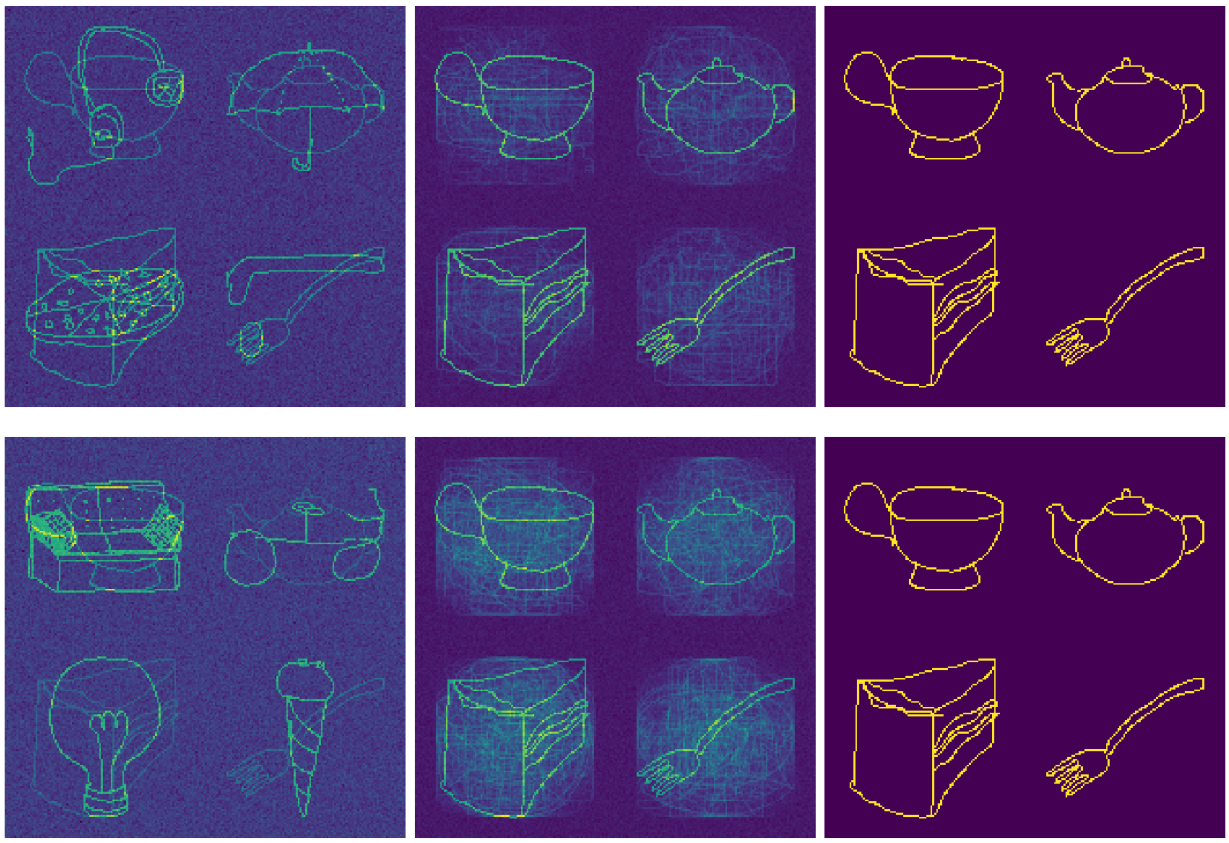

Figure 2: Stylized representation of the input and output to a Hopfield network that has the concept of a teahouse stored in it. Figure adapted from [6], with permission of the authors.

In the rightmost image, we can see the underlying concept of a teahouse, which is the co-occurrence of a teacup, a teapot, a piece of cake, and a fork. This is what the network has stored in its weights.

In the leftmost image, we have the objects observed in two locations, this is the input to the network, this happens in the form of setting the weights to the pattern corresponding to the objects. We can see that all the objects are present with varying levels of strength, but the MHN is able to extract this information and after an update, we get the middle image.

In the middle image, we see that the network converged to something close to our underlying concept of a teahouse. The noise and the spurious features were successfully filtered out.

Where do modern Hopfield networks come in?

The authors of CLOOB propose to use modern Hopfield networks to overcome the explaining away problem [1]. The embedding of a minibatch of image and text pairs is selected and stored in the network. The embeddings are then retrieved from the network and used in the loss function similar to CLIP.

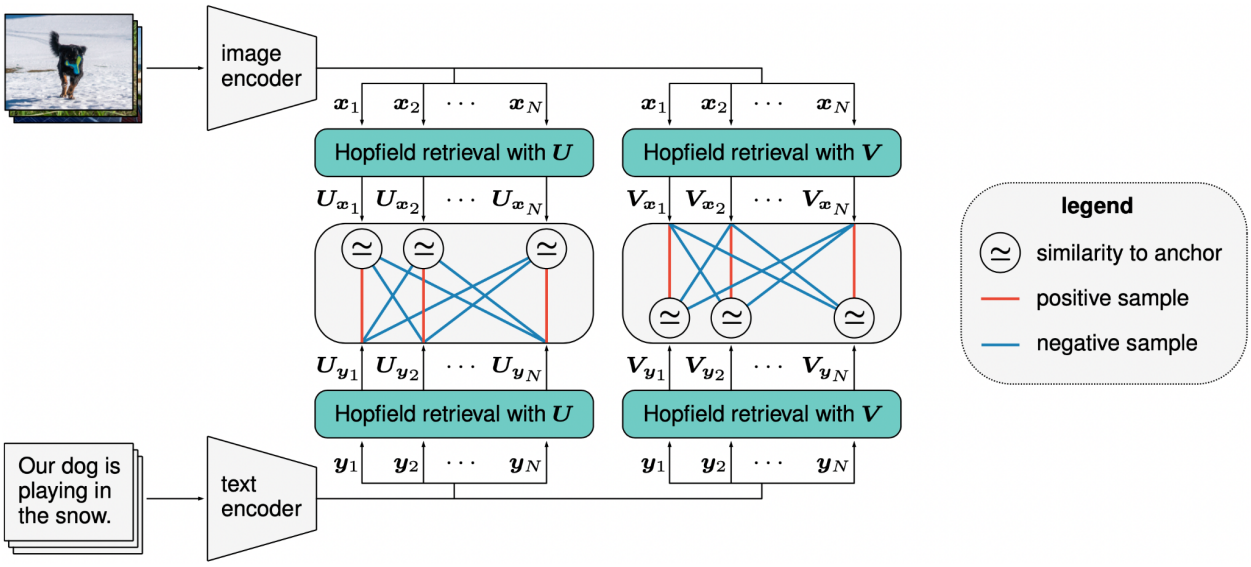

Figure 3: CLOOB's architecture. This is essentially not too different from the CLIP architecture, the main difference being that the embeddings are first stored in an MHN and then retrieved again. Figure adapted from [6], with permission of the authors.

InfoNCE, InfoLOOB, and CLOOB

InfoNCE

InfoNCE (Noice Contrastive Estimation) is the loss function of CLIP [2]. It is a form of cross-entropy loss.

$$\operatorname{L_{InfoNCE}}=-\frac{1}{N}\ln\sum_{i=1}^{N}\frac{\exp(\tau^{-1}\mathbf{x}_i^T \mathbf{y}_i)}{\sum_{j=1}^{N}\exp(\tau^{-1}\mathbf{x}_i^T\mathbf{y}_j)}-\frac{1}{N}\sum_{i=1}^{N }\ln\frac{\exp(\tau^{-1}\mathbf{x}_i^T\mathbf{y}_i)}{\sum_{j=1}^{N}\exp(\tau^{-1}\mathbf{x}_j^T \mathbf{y}_i)}$$

Since $$\Vert \mathbf{x}_i \Vert = \Vert \mathbf{y}_j \Vert = 1$$ this means that the cosine similarity is given by $$\Sim(\mathbf{x}_i, \mathbf{y}_j) = \mathbf{x}_i^T \mathbf{y}_j$$. The first sum of the function is using a fixed image embedding $$\mathbf{x}_i$$ in the denominator and compares this to all text embeddings $$\mathbf{y}_j$$, the second sum is doing to same for a fixed text embedding. In the numerator, both sums have the similarity between a matching image-text pair.

Saturation Problem

The eagle-eyed among you might have realized that in the denominator we still have $$N-1$$ unmatched pairs but also the 1 matching pair. In CLIP this is not a huge problem, as the dissimilarity of the $$N-1$$ unmatching pairs outweighs, but when using MHNs we run into the problem that the matching pair has an extremely high similarity score, and thus it “saturates” the InfoNCE loss function [1].
This can be made clearer if we only view the second sum and the first sample:

$$\operatorname{L_{InfoNCE}}(\mathbf{y}_1)=-\ln\frac{\overbrace{\exp(\tau^{-1}\mathbf{x}_1^T\ \mathbf{y}_1)}^{\textcolor{green}{a}}}{\underbrace{\exp(\tau^{-1}\mathbf{x}_1^T\mathbf{y}_1)}_{\t extcolor{green}{a}}+\underbrace{\textstyle\sum_{j=2}^{N}\exp(\tau^{-1}\mathbf{x}_j^T\mathbf{y}_ 1)}_{\textcolor{red}{b}}}$$

We have terms of the form $$\textcolor{green}{a} / (\textcolor{green}{a} + \textcolor{red}{b})$$. If $$\textcolor{green}{a}$$ is big then any change in $$\textcolor{green}{a}$$ or $$\textcolor{red}{b}$$ will have a diminishing change in the overall loss and stall learning [1]. This is what the authors refer to as the “saturation problem”. How can this be mitigated? It turns out that by simply using a loss function that does not have the positive pair in the denominator we can mitigate this problem.

InfoLOOB

InfoLOOB (Leave One Out Bound) is the loss function that the CLOOB authors propose to use to combat the saturation problem of using MHN with InfoNCE.

$$\operatorname{L_{InfoLOOB}}=-\frac{1}{N}\ln\sum_{i=1}^{N}\frac{\exp(\tau^{-1}\mathbf{x}_i^T\mathbf{y}_i)}{\sum_{j \neq i}^{N} \exp(\tau^{-1} \mathbf{x}_i^T \mathbf{y}_j)}-\frac{1}{N}\sum_{i=1}^{N}\ln\frac{\exp(\tau^{-1}\mathbf{x}_i^T\mathbf{y}_i)}{\sum_{j\neq i}^{N}\exp(\tau^{-1}\mathbf{x}_j^T \mathbf{y}_i)}$$

It is not a new function and has a history extending back to 2019 [9]. What InfoLOOB does differently is that it does not have the positive pair in the denominator. We can see this more clearly in the following representation, where we again only look at the second sum applied to one sample:

$$\operatorname{L_{InfoLOOB}}(\mathbf{y}_1)=-\ln\frac{\overbrace{\exp(\tau^{-1}\mathbf{x}_1^T\mathbf{y}_1)}^{\textcolor{green}{a}}}{\underbrace{\textstyle\sum_{j=2}^{N}\exp(\tau^{-1}\mathbf{x}_j^T \mathbf{y}_1)}_{\textcolor{red}{b}}}$$

With this we only have terms of the form $$\textcolor{green}{a} / \textcolor{red}{b}$$, meaning that any change to $$\textcolor{green}{a}$$ or to $$\textcolor{red}{b}$$ will still have an impact, no matter how large the terms get [1].

CLOOB

With the technicalities out of the way, it is now time to talk about the star of this post: CLOOB. CLOOB is the combination of using MHNs and maximizing the InfoLOOB objective function. To illustrate the operation of CLOOB it is easiest to look at a few lines of PyTorch-like pseudocode:

```python
# image_encoder - ResNet
# text_encoder - Text Transformer

# I[n, h, w, c] - minibatch of images
# T[n, l] - minibatch of texts

# W_i[d_i, d_e] - image projection
# W_t[d_t, d_e] - text projection

# beta - inverse temperature Hopfield retrieval
# tau - temperature InfoLOOB
```

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

# joint multimodal embedding
x = l2_normalize(I_f @ W_i) #[n, d_e]
y = l2_normalize(T_f @ W_t) #[n, d_e]

# Hopfield retrieval H with batch stored
# H(beta, A, B) = B.T @ softmax(beta * A @ B.T)
U_x = H(beta, x, x).T #[n, d_e]
U_y = H(beta, y, x).T #[n, d_e]
V_x = H(beta, x, y).T #[n, d_e]
V_y = H(beta, y, y).T #[n, d_e]

# normalize retrievals
U_x = l2_normalize(U_x) #[n, d_e]
U_y = l2_normalize(U_y) #[n, d_e]
V_x = l2_normalize(V_x) #[n, d_e]
V_y = l2_normalize(V_y) #[n, d_e]

# loss: info_loob(tau, anchors, samples)
# samples contain pos. and neg. embeddings
loss_i = info_loob(tau, U_x, U_y)
loss_t = info_loob(tau, V_y, V_x)
loss = (loss_i + loss_t) * tau
```

As we can see, a minibatch of samples is selected, and the features get extracted by the image and text encoders, these encoded features get normalized to norm 1, and they get fed into the MHN. We now work with the retrieved embeddings, these also get normalized to norm 1, and finally, the loss is computed symmetrically.

Experiments

To compare CLIP and CLOOB the authors of CLOOB ran two experiments, one experiment pretrained on the Conceptual Caption (CC) dataset and one experiment pretrained on the Yahoo Flickr Creative Commons (YFCC100) dataset. Both experiments were tested on the same seven classification datasets. Furthermore, the authors performed an ablation study to

see which of the two components introduced in CLOOB makes the bigger difference, i.e., they were left out to see how performance is impacted.

For the comparison experiments the authors implemented a version of CLIP known as OpenCLIP [10], which is an open-source implementation of CLIP that achieves equivalent results to the numbers reported in CLIP's paper [1].

Conceptual Caption (CC) Pretraining

This is a dataset that is very rich in textual description, it is handcrafted for classification and image captioning tasks. Therefore, it is rather "small", consisting of around 3 million images "only". Since the dataset was smaller it was feasible to train both CLIP and CLOOB multiple times. Therefore what is shown in Table 1 is the average of five runs. Statistically significant results are in bold. The asterisk indicates that the models were trained for 128 epochs, while the ones without were trained for 31 epochs.

Dataset	CLIP RN-50	CLOOB RN-50	CLIP* RN-50	CLOOB* RN-50
Birdsnap	2.26 ± 0.20	$\mathbf{3.06 \pm 0.30}$	2.8 ± 0.16	$\mathbf{3.24 \pm 0.31}$
Country211	0.67 ± 0.11	0.67 ± 0.05	0.7 ± 0.04	0.73 ± 0.05
Flowers102	12.56 ± 0.38	13.45 ± 1.19	13.32 ± 0.43	14.36 ± 1.17
GTSRB	7.66 ± 1.07	6.38 ± 2.11	8.96 ± 1.70	7.03 ± 1.22
UCF101	20.98 ± 1.55	22.26 ± 0.72	21.63 ± 0.65	$\mathbf{23.03 \pm 0.85}$
Stanford Cars	0.91 ± 0.10	$\mathbf{1.23 \pm 0.10}$	0.99 ± 0.16	$\mathbf{1.41 \pm 0.32}$
ImageNet	20.33 ± 0.28	$\mathbf{23.97 \pm 0.15}$	21.3 ± 0.42	$\mathbf{25.67 \pm 0.22}$
ImageNet V2	20.24 ± 0.50	$\mathbf{23.59 \pm 0.15}$	21.24 ± 0.22	$\mathbf{25.49 \pm 0.11}$

Table 1: Zero-shot accuracy comparison of CLIP and CLOOB pretrained on the CC dataset, higher is better. Figure adapted from [6], with permission of the authors.

Yahoo Flickr Creative Commons (YFCC100M) Pretraining

YFCC100M is a dataset containing 100 million multi-media (99.8 million images, and 0.8 million videos) files that were collected by scrubbing the internet, the website Flickr to be precise [11]. YFCC, as referred to by the authors of CLOOB, is a subset of YFCC100M created by filtering out images that have a natural language description and/or titles in English. Therefore the text prompts of this dataset are less rich and often do not carry any useful information. The authors tested multiple image encoders which can be seen in the column names of Table 2. Since this was only done once per model and encoder size, bold numbers are simply the higher values.

Dataset	RN-50		RN-101		RN-50x4	
	CLIP	CLOOB	CLIP	CLOOB	CLIP	CLOOB
Birdsnap	21.8	**28.9**	22.6	**30.3**	20.8	**32.0**
Country211	6.9	**7.9**	7.8	**8.5**	8.1	**9.3**
Flowers102	48.0	**55.1**	48.0	**55.3**	50.1	**54.3**
GTSRB	7.9	**8.1**	7.4	**11.6**	9.4	**11.8**
UCF101	**27.2**	25.3	28.6	**28.8**	31.0	**31.9**
Stanford Cars	3.7	**4.1**	3.8	**5.5**	3.5	**6.1**
ImageNet	34.6	**35.7**	35.3	**37.1**	37.7	**39.0**
ImageNet V2	33.4	**34.6**	34.1	**35.6**	35.9	**37.3**

Table 2: Zero-shot accuracy comparison of CLIP and CLOOB pretrained on the YFCC dataset, higher is better. Figure adapted from [6], with permission of the authors.

Ablation Studies

For the ablation study the author calculated the relative increase in eigenvalues needed to reconstruct 99% of the covariance matrix of the embeddings. At an early stage in training the number of eigenvalues needed to reconstruct the covariance matrix was gathered, and then again at a later stage in training. The relation of these two numbers is the relative increase of number of eigenvalues. This is a relevant measure since it indicates the univormity of the covariance matrix, i.e., the richness of the covariance structure.

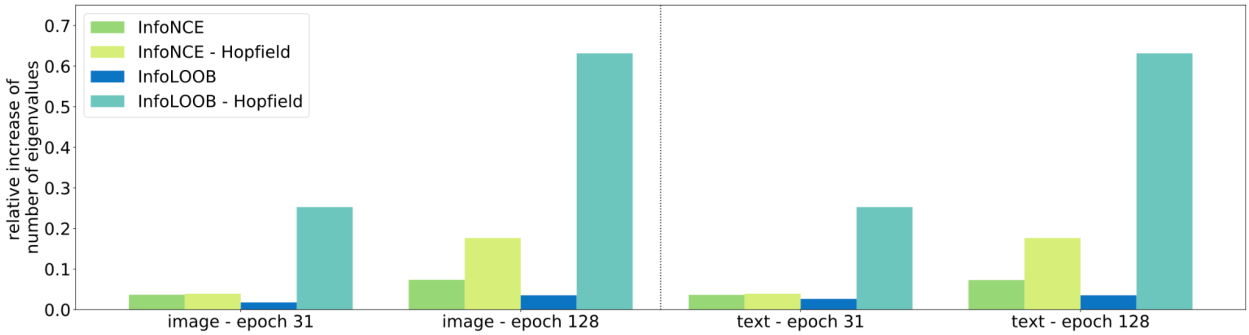

Figure 4: Relative increase of active eigenvalues of the covariance matrix during training. Figure adapted from [6], with permission of the authors.

Conclusions

The introduction of modern Hopfield networks has led to CLOOB being able to extract a far more rich covariance and co-occurrence structure then previous models, allowing to build more relevant features in the embedding space, mitigating the explaining away problem seen in CLIP

[1]. CLOOB can be used for anything that CLIP is used for, their “interface” is interchangeable. With the better performance of CLOOB, this would make sense, according to the authors of CLOOB there is only a 5% penalty in speed, which can be attributed to storing and retrieving embeddings from the MHN. One drawback of MHNs is, that since the data is being stored in a neural network and the network is stored on the GPU it means we are limited by the graphics memory.

References

[1] A. Fürst et al., ‘CLOOB: Modern Hopfield Networks with InfoLOOB Outperform CLIP’, CoRR, vol. abs/2110.11316, 2021.

[2] A. Radford et al., ‘Learning Transferable Visual Models From Natural Language Supervision’, CoRR, vol. abs/2103.00020, 2021.

[3] A. Ramesh et al., ‘Zero-Shot Text-to-Image Generation’, CoRR, vol. abs/2102.12092, 2021.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, ‘A Simple Framework for Contrastive Learning of Visual Representations’, CoRR, vol. abs/2002.05709, 2020.

[5] P. Bachman, R. D. Hjelm, and W. Buchwalter, ‘Learning Representations by Maximizing Mutual Information Across Views’, CoRR, vol. abs/1906.00910, 2019.

[6] E. Rumetshofer and A. Fürst, [author’s blog]. https://ml-jku.github.io/cloob/ (accessed June 19, 2023).

[7] R. Bommasani et al., ‘On the Opportunities and Risks of Foundation Models’, CoRR, vol. abs/2108.07258, 2021.

[8] M. Wellman and M. Henrion, ‘Explaining `explaining away`’, Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 15, pp. 287–292, 04 1993.

[9] B. Poole, S. Ozair, A. van den Oord, A. A. Alemi, and G. Tucker, ‘On Variational Bounds of Mutual Information’, CoRR, vol. abs/1905.06922, 2019.

[10] OpenAI. ‘openai/CLIP: CLIP (Contrastive Language-Image Pre-Training) predict the most relevant text snippet given an image.’ Github. https://github.com/openai/CLIP (accessed June 19, 2023).

[11] B. Thomee et al., “The New Data and New Challenges in Multimedia Research,” CoRR, vol. abs/1503.01817, 2015, [Online]. Available: http://arxiv.org/abs/1503.01817

About me

Pascal Pilz

Pascal Pilz is currently a Bachelor's student studying Artificial Intelligence at Johannes Kepler University in Linz, Austria. Born in the heart of the Salzkammergut, he finished his school education at the Werkschulheim Felbertal near Salzburg, earning an AHS Matura with Physics, Psychology, and English as voluntary exam subjects. Additionally, the school allowed him to complete an apprenticeship in Mechatronics. At JKU Linz, he is part of the Neuron AI team under the Brainery department. His interests focus on the theoretical aspects of machine learning and model architecture design. In his personal life, he enjoys training in Taekwondo as well as other sports like bouldering, cycling, and yoga. To maintain the balance, he also dabbles in video games every once in a while.