CLOOB: Modern Hopfield Networks with InfoLOOB Outperform CLIP

Andreas Fürst ⇤1 Elisabeth Rumetshofer ⇤1 Johannes Lehner1 Viet Tran1
Fei Tang3 Hubert Ramsauer1 David Kreil2 Michael Kopp2
Günter Klambauer1 Angela Bitto-Nemling1 Sepp Hochreiter1 2

1 ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning,
Johannes Kepler University, Linz, Austria
2 Institute of Advanced Research in Artificial Intelligence (IARAI), Vienna, Austria
3 HERE Technologies, Zurich, Switzerland
⇤ Equal contribution

Abstract

CLIP yielded impressive results on zero-shot transfer learning tasks and is considered as a foundation model like BERT or GPT3. CLIP vision models that have a rich representation are pre-trained using the InfoNCE objective and natural language supervision before they are fine-tuned on particular tasks. Though CLIP excels at zero-shot transfer learning, it suffers from an explaining away problem, that is, it focuses on one or few features, while neglecting other relevant features. This problem is caused by insufficiently extracting the covariance structure in the original multi-modal data. We suggest to use modern Hopfield networks to tackle the problem of explaining away. Their retrieved embeddings have an enriched covariance structure derived from co-occurrences of features in the stored embeddings. However, modern Hopfield networks increase the saturation effect of the InfoNCE objective which hampers learning. We propose to use the InfoLOOB objective to mitigate this saturation effect. We introduce the novel “Contrastive Leave One Out Boost” (CLOOB), which uses modern Hopfield networks for covariance enrichment together with the InfoLOOB objective. In experiments we compare CLOOB to CLIP after pre-training on the Conceptual Captions and the YFCC dataset with respect to their zero-shot transfer learning performance on other datasets. CLOOB consistently outperforms CLIP at zero-shot transfer learning across all considered architectures and datasets.

1 Introduction

Contrastive Language-Image Pre-training (CLIP) showed spectacular performance at zero-shot transfer learning (Radford et al., 2021). CLIP learns expressive image embeddings directly from raw text, thereby leverages a much richer source of supervision than just labels. The CLIP model is considered as an important foundation model (Bommasani et al., 2021), therefore a plethora of follow-up work has been published (see Appendix Section A.4). CLIP as a contrastive learning method has two simultaneous goals, namely (i) increasing the similarity of matched language-image pairs and (ii) decreasing the similarity of unmatched language-image pairs. Though CLIP yielded

Code is available at: https://github.com/ml-jku/cloob

We introduce “Contrastive Leave One Out Boost” (CLOOB) which combines modern Hopfield networks with stored pattern, while large et al. image-text pairs. The training set consists of be used in any method which builds upon CLIP. Figure 2 CLOOB: Modern Hopfield Networks with InfoLOOB networks with the “InfoLOOB” objective. Our contributions are:

(a) we propose CLOOB, a new contrastive learning method,
(b) we propose modern Hopfield networks to reinforce covariance structures,
(c) we propose InfoLOOB as an objective to avoid saturation as observed with InfoNCE, and provide theoretical justifications for optimizing InfoLOOB.

2 CLOOB: Modern Hopfield Networks with InfoLOOB

Our novel contrastive learning method CLOOB can be seen as a replacement of CLIP and therefore be used in any method which builds upon CLIP. Figure 1 sketches the CLOOB architecture for image-text pairs. The training set consists of \( N \) pairs of embeddings \( \{x_1, y_1\}, \ldots, (x_N, y_N) \) with \( X = (x_1, \ldots, x_N) \) and \( Y = (y_1, \ldots, y_N) \), \( M \) stored embeddings \( U = (u_1, \ldots, u_M) \), and \( K \) stored embeddings \( V = (v_1, \ldots, v_K) \). The state or query embeddings \( x_i \) and \( y_i \) retrieve \( U_{x_i} \) and \( U_{y_i} \), respectively, from \( U \) — analog for retrievals from \( V \). The samples are normalized:

\[
\|x_i\| = \|y_i\| = \|u_i\| = \|v_i\| = 1.
\]

\( U_{x_i} \) denotes an image-retrieved image embedding, \( U_{y_i} \) a text-retrieved image embedding, \( V_{x_i} \) an image-retrieved text embedding and \( V_{y_i} \) a text-retrieved text embedding. These retrievals from modern Hopfield networks are computed as follows (Ramsauer et al., 2021):

\[
U_{x_i} = U \text{softmax}(\beta U^T x_i), \quad (1) \quad V_{x_i} = V \text{softmax}(\beta V^T x_i), \quad (3)
\]

\[
U_{y_i} = U \text{softmax}(\beta U^T y_i), \quad (2) \quad V_{y_i} = V \text{softmax}(\beta V^T y_i). \quad (4)
\]

The hyperparameter \( \beta \) corresponds to the inverse temperature: \( \beta = 0 \) retrieves the average of the stored pattern, while large \( \beta \) retrieves the stored pattern that is most similar to the state pattern (query).
We use modern Hopfield networks to amplify co-occurrences and the covariance structure. Replacing the original embeddings by retrieved embeddings reinforces features that frequently occur together in stored embeddings. Additionally, spurious co-occurrences that are peculiar to a sample are averaged out. By this means, the covariance structure is reinforced by the retrieved embeddings. By storing the embeddings of the mini-batch examples in the Hopfield memory, we do not require to compute the embeddings of additional samples via text and image encoders.

The next two sections analyze CLOOB’s major components.

In the InfoLOOB loss Eq. (8), CLOOB substitutes the embedded samples \( x_i \) and \( y_i \) by the normalized retrieved embedded samples. In the first term, \( x_i \) and \( y_i \) are substituted by \( U_{x_i} \) and \( U_{y_i} \), respectively, while in the second term they are substituted by \( V_{x_i} \) and \( V_{y_i} \). After retrieval, the samples are re-normalized to ensure \( \|U_{x_i}\| = \|U_{y_i}\| = \|V_{x_i}\| = \|V_{y_i}\| = 1 \).

We obtain the CLOOB loss function:

\[
I_{\text{InfoLOOB}} = - \frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\langle x_i, y_i \rangle)}{\sum_{j \neq i} \exp(\langle x_i, y_j \rangle)} - \frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\langle x_i, y_i \rangle)}{\sum_{j \neq i} \exp(\langle x_i, y_j \rangle)} .
\]

By default, we store the minibatch in the modern Hopfield networks, that is, \( U = X \) and \( V = Y \). Thus, in Eq. (1) \( x_i \) can retrieve itself from \( U = X \), but in Eq. (3) it can not retrieve itself from \( V = Y \). Analogously, in Eq. (4) \( y_i \) can retrieve itself from \( V = Y \), but in Eq. (2) it can not retrieve itself from \( U = X \). By storing the embeddings of the mini-batch examples in the Hopfield memory, we do not require to compute the embeddings of additional samples via text and image encoders. However, the modern Hopfield networks can also store prototypes, templates, or proprietary samples to amplify particular embedding features via the stored samples. Either the original embeddings \( x \) and \( y \) or the retrieved embeddings \( U_x, U_y, V_x, \) and \( V_y \) may serve for the downstream tasks, e.g. for zero-shot transfer learning.

Pseudocode 1 shows CLOOB in a PyTorch-like style. CLOOB has two major components: (i) modern Hopfield networks that alleviate CLIP’s problem of insufficiently exploiting the covariance structure in the data and (ii) the InfoLOOB objective that does not suffer from InfoNCE’s saturation problem.

The next two sections analyze CLOOB’s major components.

### 3 Modern Hopfield Networks for Enriching the Covariance Structure

We use modern Hopfield networks to amplify co-occurrences and the covariance structure. Replacing the original embeddings by retrieved embeddings reinforces features that frequently occur together in stored embeddings. Additionally, spurious co-occurrences that are peculiar to a sample are averaged out. By this means, the covariance structure is reinforced by the retrieved embeddings \( U_x^T U_y \) and \( V_x^T V_y \). The Jacobian \( J \) of the softmax \( p = \text{softmax}(\beta a) \) is \( J(\beta a) = \beta (\text{diag}(p) - pp^T) \).

We define the weighted covariance \( \text{Cov}(U) \), where sample \( u_i \) is drawn with probability \( p_i \), as

\[
[Cov(U)]_{kl} = \langle UJ(\beta a)U^T \rangle_{kl} = \beta (\sum_{i=1}^{M} p_i u_i k u_i d - \sum_{i=1}^{M} p_i u_i k \sum_{i=1}^{M} p_i u_i d) .
\]

The formula of the...
weighted covariance differs from the standard empirical covariance, since the factor $1/M$ is replaced by $p_i$. Thus, $u_i$ is sampled with probability $p_i$ instead with probability $1/M$ (uniformly).

We apply the mean value theorem to the softmax function with mean Jacobian matrix $J^m(\beta a) = \int_0^1 J(\lambda a) \, d\lambda$. The mean Jacobian $J^m(\beta a)$ is a symmetric, diagonally dominant, positive semi-definite matrix with one eigenvalue of zero for eigenvector 1 and spectral norm bounded by $\|J^m\|_2 \leq 0.5\beta$ (see Appendix Lemma A1). According to Appendix Theorem A3, we can express $U^T_{x_i}U_{y_i}$, as:

$$
(\bar{u} + \text{Cov}(U, x_i) x_i)^T (\bar{u} + \text{Cov}(U, y_i) y_i)
$$

(6)

where the mean is $\bar{u} = 1/MU1$ and the weighted covariances are $\text{Cov}(U, x_i) = U^T_m(\beta U^T x_i)U^T$ and $\text{Cov}(U, y_i) = U^T_m(\beta U^T y_i)U^T$. The weighted covariance $\text{Cov}(U, .)$ is the covariance if the stored pattern $u_i$ is drawn according to an averaged $p_i$ given by $J^m(.)$. Maximizing the dot product $U^T_{x_i}U_{y_i}$ forces the normalized vectors $x_i$ and $y_i$ to agree on drawing the patterns $u_i$ with the same probability $p_i$ in order to generate similar weighted covariance matrices $\text{Cov}(U, .)$. If subsets of $U$ have a strong covariance structure, then it can be exploited to produce large weighted covariances and, in turn, large dot products of $U^T_{x_i}U_{y_i}$. Furthermore, for a large dot product $U^T_{x_i}V_{y_i}$, as $x_i$ and $y_i$ have to be similar to each other to extract the same direction from the covariance matrices. The above considerations for $U^T_{x_i}U_{y_i}$, analogously apply to $V^T_{x_i}V_{y_i}$.

We did not use a loss function that contains dot products like $U^T_{x_i}V_{y_i}$, because they have higher variance than the ones we have used. The dot product $U^T_{x_i}V_{y_i}$ has higher variance, since it uses $M + K$ stored patterns, whereas $U^T_{x_i}U_{y_i}$ and $V^T_{x_i}V_{y_i}$ use $M$ and $K$ stored patterns, respectively.

**Modern Hopfield networks enable to extract more covariance structure.** To demonstrate the effect of modern Hopfield networks, we computed the eigenvalues of the covariance matrix of the image and text embeddings. We counted the number of effective eigenvalues, that is, the number of eigenvalues needed to obtain 99% of the total sum of eigenvalues. Figure 2 shows the relative change of the number of effective eigenvalues compared to the respective reference epoch (the epoch before the first learning rate restart). Modern Hopfield networks consistently increase the number of effective eigenvalues during learning. Consequently, modern Hopfield networks enable to extract more covariance structure during learning, i.e. enrich the embeddings by covariances that are already in the raw multi-modal data. This enrichment of embeddings mitigates explaining away. Further details can be found in Appendix Section A2.7.

### 4 InfoLOOB for Contrastive Learning

Modern Hopfield networks lead to a higher similarity of retrieved samples. The increased similarity exacerbates the saturation of the InfoNCE objective. To avoid the saturation of InfoNCE, CLOOB uses the “InfoLOOB” objective. The “InfoLOOB” objective is called “Leave one out upper bound” in Poole et al. (2019) and “L1Out” in Cheng et al. (2020). InfoLOOB is not established as a contrastive
We abbreviate where with modern Hopfield networks steadily extract more covariance structure during learning.

The gradients of Eq. (10) with respect to \( y \) are

\[
\frac{\partial L_{\text{InfoLOOB}}}{\partial y} = -\tau^{-1} x^T y + \tau^{-1} \text{lse}(\tau^{-1}, X^T y),
\]

where lse is the log-sum-exp function (see Eq. (A73) in the Appendix).

The gradients of Eq. (9) and Eq. (10) with respect to \( y \) are

\[
\frac{\partial L_{\text{InfoNCE}}}{\partial y} = -\tau^{-1} y^T x_1 + \tau^{-1} \text{lse}(\tau^{-1}, X^T y),
\]

\[
\frac{\partial L_{\text{InfoLOOB}}}{\partial y} = -\tau^{-1} x^T y + \tau^{-1} \text{lse}(\tau^{-1}, X^T y).
\]

In analogy to Wang & Isola (2020), \( a \) is called the “alignment score” that measures the similarity of matched pairs and \( b \) the “uniformity penalty” that measures the similarity of unmatched pairs.

**Gradients of InfoNCE and InfoLOOB loss functions.** Eq. (9) and Eq. (10) are equal to

\[
-\tau^{-1} y^T x_1 + \tau^{-1} \text{lse}(\tau^{-1}, X^T y),
\]

\[
-\tau^{-1} x^T y + \tau^{-1} \text{lse}(\tau^{-1}, X^T y).
\]

Figure 2: Relative change in the number of the effective eigenvalues of the embedding covariance matrices, which were obtained from image and text encoders at two different training points. Models with modern Hopfield networks steadily extract more covariance structure during learning.

**InfoNCE and InfoLOOB loss functions.** \( N \) samples are drawn iid from \( p(x, y) \) giving the training set \( \{(x_1, y_1), \ldots, (x_N, y_N)\} \). For the sample \( y_1 \), InfoNCE uses for the matrix of negative samples \( X = (x_1, \ldots, x_N) \), while InfoLOOB uses \( \tilde{X} = (x_2, \ldots, x_N) \). The matrices differ by the positive sample \( x_1 \). For the score function \( f(x, y) \), we use \( f(x, y) = \exp(\tau^{-1} \text{sim}(x, y)) \) with the similarity \( \text{sim}(x, y) = y^T x \) and \( \tau \) as the temperature. We have the InfoNCE and InfoLOOB loss functions:

\[
L_{\text{InfoNCE}} = -\frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\tau^{-1} x_i^T y_1)}{\sum_{j=1}^{N} \exp(\tau^{-1} x_j^T y_1)} - \frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\tau^{-1} x_i^T y_1)}{\sum_{j=1}^{N} \exp(\tau^{-1} x_j^T y_1)},
\]

\[
L_{\text{InfoLOOB}} = -\frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\tau^{-1} x_i^T y_1)}{\sum_{j \neq i}^{N} \exp(\tau^{-1} x_j^T y_1)} - \frac{1}{N} \sum_{i=1}^{N} \ln \frac{\exp(\tau^{-1} x_i^T y_1)}{\sum_{j \neq i}^{N} \exp(\tau^{-1} x_j^T y_1)}.
\]

We abbreviate \( y = y_1 \) leading to the pair \((x_1, y)\) and the negatives \( \tilde{X} = (x_2, \ldots, x_N) \). In the second sum of the losses in Eq. 7 and Eq. 8, we consider only the first term, respectively:

\[
L_{\text{InfoNCE}}(y) = -\ln \frac{\exp(\tau^{-1} x_1^T y)}{\sum_{j=2}^{N} \exp(\tau^{-1} x_j^T y)},
\]

\[
L_{\text{InfoLOOB}}(y) = -\ln \frac{\exp(\tau^{-1} x_1^T y)}{\sum_{j=2}^{N} \exp(\tau^{-1} x_j^T y)}.
\]
Figure 3: Ajne uniformity test statistics for image and text embeddings for two different epochs during training. A high test statistic indicates low uniformity of an embedding. Models trained with the InfoLOOB objective develop more uniform image and text embeddings on the hypersphere.

By and large, the gradient of InfoNCE is scaled by \((1 - p_1)\) compared to the gradient of InfoLOOB, where \(p_1\) is the softmax similarity between the anchor \(y\) and the positive sample \(x_1\). Consequently, InfoNCE is saturating with increasing similarity between the anchor and the positive sample. For more details we refer to Appendix Section A.1.4.

This saturation of InfoNCE motivated the use of the InfoLOOB objective in order to decrease the uniformity penalty even for large alignment scores. The uniformity penalty decreases since learning does not stall and the most prominent features become down-scaled which makes negative examples less similar to the anchor sample. The InfoNCE objective Eq. 9 has the form \(a/(a + b)\), while the InfoLOOB objective Eq. 10 has the form \(a/b\). InfoLOOB does not saturate and keeps decreasing the uniformity penalty \(b\). Figure 3 shows how InfoLOOB leads to an increase in the uniformity of image and text embeddings on the sphere, which is described by the statistics of the uniformity test of Ajne that was extended by Prentice (Ajne, 1968; Prentice, 1978). Higher uniformity on the sphere correlates with a lower uniformity penalty \(b\). For more details we refer to Appendix Section A.2.7.

**Theoretical justification for optimizing InfoLOOB.** The InfoNCE information is a lower bound on the mutual information, which was proven by Poole et al. (2019). In the Appendix Section A.1, we elaborate more on theoretical properties of the bounds and properties of the objective functions. Specifically, we show that InfoLOOB with neural networks is not an upper bound on the mutual information. Thus, unlike hitherto approaches to contrastive learning we use InfoLOOB as an objective, since it does not suffer from saturation effects as InfoNCE.

5 Experiments

CLOOB is compared to CLIP with respect to zero-shot transfer learning performance on two pre-training datasets. The first dataset, Conceptual Captions (CC) (Sharma et al., 2018), has a very rich textual description of images but only three million image-text pairs. The second dataset, a subset of YFCC100M (Thomee et al., 2016), has 15 million image-text pairs but the textual description is less rich than for CC and often vacuous. For both pre-training datasets, the downstream zero-shot transfer learning performance is tested on seven image classification datasets.

5.1 Conceptual Captions Pre-training

**Pre-training dataset.** The Conceptual Captions (CC) (Sharma et al., 2018) dataset contains 2.9 million images with high-quality captions. Images and their captions have been gathered from the web via an automated process and have a wide variety of content. Raw descriptions of images are from the *alt-*HTML attribute.

**Methods.** The CLOOB implementation is based on OpenCLIP (Ilharco et al., 2021), which achieves results equivalent to CLIP on the YFCC dataset (see Section 5.2). OpenCLIP also reports results on the CC dataset. As CLIP does not train models on CC, we report results from this reimplementation as baseline. Analogously to Radford et al. (2021, Section 2.4), we used the modified ResNet (He et al., 2016) and BERT (Devlin et al., 2018, 2019) architectures to encode image and text input. We used the ResNet encoder ResNet-50 for experiments on CC.
Table 1: Zero-shot results for models trained on CC with ResNet-50 vision encoders for two different checkpoints. Results are given as mean accuracy over 5 runs. Statistically significant results are shown in bold. CLIP and CLOOB were trained for 31 epochs while CLIP* and CLOOB* were trained for 128 epochs. In the majority of tasks CLOOB significantly outperforms CLIP.

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

Hyperparameter selection and learning schedule. The hyperparameter values of OpenCLIP were used as default, concretely, a learning rate of $1 \times 10^{-3}$ and a weight decay of 0.1 for the Adam optimizer (Kingma et al., 2014) with decoupled weight decay regularization (Loshchilov & Hutter, 2019). Deviating from OpenCLIP, we used a batch size of 512 due to computational restraints, which did not change the performance. The learning rate scheduler for all experiments was cosine annealing with warmup and hard restarts (Loshchilov & Hutter, 2017). We report the hyperparameter $\tau$ (default 0.07) from CLIP as $\tau^{-1}$ of 14.3 to be in the same regime as the hyperparameter $\beta$ for the modern Hopfield networks. The main hyperparameter search for CLOOB (also for YFCC pre-training in the next section) was done with ResNet-50 as the vision encoder. Learnable $\tau^{-1}$ in combination with the InfoLoOB loss results in undesired learning behavior (see Appendix Section A.1.4). Therefore, we set $\tau^{-1}$ to a fixed value of 30, which was determined via a hyperparameter search (see Appendix Section A.2.2). For modern Hopfield networks, the hyperparameter $\beta$ was set to 8. Further we scaled the loss $L_{\text{InfoLoOB}}$ with $\tau$ to remove the factor $\tau^{-1}$ from the gradients (see Appendix Section A.1.4) resulting in the loss function $\tau L_{\text{InfoLoOB}}$.

Evaluation metrics: Zero-shot transfer learning. We evaluated and compared both CLIP and CLOOB on their zero-shot transfer learning capabilities on the following downstream image classification tasks. Birdsnap (Berg et al., 2014) contains images of 500 different North American bird species. The Country211 (Radford et al., 2021) dataset consists of photos across 211 countries and is designed to test the geolocalization capability of visual representations. Flowers102 (Nilsback & Zisserman, 2008) is a dataset containing images of 102 flower species. GTSRB (Stallkamp et al., 2011) contains images for classification of German traffic signs. UCF101 (Soomro et al., 2012) is a video dataset with short clips for action recognition. For UCF101 we followed the procedure reported in CLIP and extract the middle frame of every video to assemble the dataset. Stanford Cars (Krause et al., 2013) contains images of 196 types of cars. ImageNet (Deng et al., 2009) is a large scale image classification dataset with images across 1,000 classes. ImageNetv2 (Recht et al., 2019) consists of three new test sets with 10,000 images each for the ImageNet benchmark. For further details see Appendix Section A.2.3.

Results. We employed the same evaluation strategy and used the prompts as published in CLIP (see Appendix Section A.2.3). We report zero-shot results from two checkpoints in Table 1. CLIP and CLOOB were trained for a comparable number of epochs used in CLIP (see Appendix Section A.2.2) while CLIP* and CLOOB* were trained until evaluation performance plateaued (epoch 128). In both cases CLOOB significantly outperforms CLIP on the majority of tasks or matches its performance. Statistical significance of these results was assessed by an unpaired Wilcoxon test on a 5% level.

5.2 YFCC Pre-training

Pre-training dataset. To be comparable to the CLIP results, we used the same subset of 15 million samples from the YFCC100M dataset (Thomee et al., 2016) as in Radford et al. (2021), which we refer to as YFCC. YFCC was created by filtering YFCC100M for images which contain natural language descriptions and/or titles in English. It was not filtered by quality of the captions, therefore the textual descriptions are less rich and contain superfluous information. The dataset with 400
million samples used to train the CLIP models in Radford et al. (2021) has not been released and, thus, is not available for comparison. Due to limited computational resources we were unable to compare CLOOB to CLIP on other datasets of this size.

**Methods.** Besides experiments with a ResNet-50 image encoder, we additionally conducted experiments with the larger ResNet variants ResNet-101 and ResNet-50x4. In addition to the comparison of CLOOB and CLIP based on the OpenCLIP reimplementation (Ilharco et al., 2021), we include the original CLIP results (Radford et al., 2021, Table 12).

**Hyperparameter selection.** Hyperparameters were the same as for the Conceptual Captions dataset, except learning rate, batch size, and $\beta$. For modern Hopfield networks, the hyperparameter $\beta$ was set to 14.3, which is default for $\tau^{-1}$ in Radford et al. (2021). Furthermore, the learning rate was set to $5 \times 10^{-4}$ and the batch size to 1024 as used in OpenCLIP of Ilharco et al. (2021). All models were trained for 28 epochs. For further details see Appendix Section A.2.2.

**Evaluation metrics.** As in the previous experiment, methods were again evaluated at their zero-shot transfer learning capabilities on downstream tasks.

Table 2: Results of CLIP and CLOOB trained on YFCC with ResNet-50 encoder. Except for one linear probing dataset, CLOOB consistently outperforms CLIP at all tasks.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Linear Probing</th>
<th>Zero-Shot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CLIP (OpenAI)</td>
<td>CLOOB (ours)</td>
</tr>
<tr>
<td>Birdsnap</td>
<td>47.4</td>
<td>56.2</td>
</tr>
<tr>
<td>Country211</td>
<td>23.1</td>
<td>20.6</td>
</tr>
<tr>
<td>Flowers102</td>
<td>94.4</td>
<td>96.1</td>
</tr>
<tr>
<td>GTSRB</td>
<td>66.8</td>
<td>78.9</td>
</tr>
<tr>
<td>UCF101</td>
<td>69.2</td>
<td>72.3</td>
</tr>
<tr>
<td>Stanford Cars</td>
<td>31.4</td>
<td>37.7</td>
</tr>
<tr>
<td>ImageNet</td>
<td>62.0</td>
<td>65.7</td>
</tr>
<tr>
<td>ImageNet V2</td>
<td>-</td>
<td>58.7</td>
</tr>
</tbody>
</table>

Table 3: Zero-shot results for the CLIP reimplementation and CLOOB using different ResNet architectures trained on YFCC. CLOOB outperforms CLIP in 7 out of 8 tasks using ResNet-50 encoders. With larger ResNet encoders CLOOB outperforms CLIP on all tasks. The performance of CLOOB scales with increased encoder size.

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

**Results.** Table 2 provides results of the original CLIP and CLOOB trained on YFCC. Results on zero-shot downstream tasks show that CLOOB outperforms CLIP on all 7 tasks (ImageNet V2 results have not been reported in Radford et al. (2021)). Similarly, CLOOB outperforms CLIP on 6 out of 7 tasks for linear probing. Results of CLOOB and the CLIP reimplementation of OpenCLIP are given in Table 3. CLOOB exceeds the CLIP reimplementation in 7 out of 8 tasks for zero-shot classification using ResNet-50 encoders. With larger ResNet encoders, CLOOB outperforms CLIP on all tasks. Furthermore, the experiments with larger vision encoder networks show that CLOOB performance increases with network size. Results of CLOOB zero-shot classification on all datasets are shown in Appendix Section A.2.4.
5.3 Ablation studies

CLOOB has two new major components compared to CLIP: (1) modern Hopfield networks and (2) the InfoLOOB objective instead of the InfoNCE objective. To assess effects of the new major components of CLOOB, we performed ablation studies.

**Modern Hopfield networks.** Modern Hopfield networks amplify the covariance structure in the data via the retrievals. Ablation studies confirm this amplification as modern Hopfield networks consistently increase the number of effective eigenvalues of the embedding covariance matrices during learning. Figure 2 shows the relative change of the number of effective eigenvalues compared to the respective reference epoch, which is the epoch before the first learning rate restart. These results indicate that modern Hopfield networks steadily extract more covariance structure during learning. Modern Hopfield networks induce higher similarity of retrieved samples, which in turn leads to stronger saturation of the InfoNCE objective. As a result, we observe low uniformity (see Figure 3) and a small number of effective eigenvalues (see Appendix Figure A1).

**Modern Hopfield networks with InfoLOOB.** CLOOB counters the saturation of InfoNCE by using the InfoLOOB objective. The effectiveness of InfoLOOB manifests in an increased uniformity measure of image and text embeddings on the sphere, as shown in Figure 3. The ablation study verifies that modern Hopfield networks together with InfoLOOB have a strong synergistic effect.

**InfoLOOB.** However, using solely InfoLOOB results in overfitting of the alignment score. This overfitting occasionally leads to high similarities of unmatched pairs (see Figure 4), which may decreases the zero-shot downstream performance. The reason for this is that the top-1 evaluation metric is very sensitive to occasionally high similarities of prompts of the incorrect class. Yeh et al. (2021) and Zhang et al. (2022) reported similar observations of overfitting.

![Figure 4: Distribution of the cosine similarity of matched pairs and the cosine similarity of the 10 unmatched pairs that have the highest similarity score with the anchor. Modern Hopfield networks lead to higher values of both matched and unmatched pairs. InfoLOOB without Hopfield has high similarity scores of the matched pairs which correlate with high similarity scores of the top-10 unmatched pairs. In contrast, InfoLOOB with Hopfield does not suffer from this overfitting problem.](image)

CLOOB balances the overfitting of InfoLOOB with the underfitting of modern Hopfield networks and remains in effective learning regimes. For more details and further ablation studies see Appendix Section A.2.1.

6 Conclusion

We have introduced “Contrastive Leave One Out Boost” (CLOOB), which combines modern Hopfield networks with the InfoLOOB objective. Modern Hopfield networks enable CLOOB to extract additional covariance structure in the data. This allows for building more relevant features in the embedding space, mitigating the explaining away problem. We show that InfoLOOB avoids the saturation problem of InfoNCE. Additionally, we theoretically justify the use of the InfoLOOB objective for contrastive learning and suggest it as an alternative to InfoNCE. At seven zero-shot transfer learning tasks, the novel CLOOB was compared to CLIP after pre-training on the Conceptual Captions and the YFCC dataset. CLOOB consistently outperforms CLIP at zero-shot transfer learning across all considered architectures and datasets.
Acknowledgments

The ELLIS Unit Linz, the LIT AI Lab, the Institute for Machine Learning, are supported by the Federal State Upper Austria. IARAI is supported by Here Technologies. We thank the projects AI-MOTION (LIT-2018-6-YOU-212), AI-SNN (LIT-2018-6-YOU-214), DeepFlood (LIT-2019-8-YOU-213), Medical Cognitive Computing Center (MC3), INCONTROL-RL (FFG-881064), PRIMAL (FFG-873979), S3AI (FFG-872172), DL for GranularFlow (FFG-871302), AIRI FG 9-N (FWF-36284, FWF-36235), ELISE (H2020-ICT-2019-3 ID: 951847). We thank Audi.JKU Deep Learning Center, TGW LOGISTICS GROUP GMBH, Silicon Austria Labs (SAL), FILL Gesellschaft mbH, Anyline GmbH, Google, ZF Friedrichshafen AG, Robert Bosch GmbH, UCB Biopharma SRL, Merck Healthcare KGaA, Verbund AG, Software Competence Center Hagenberg GmbH, TÜV Austria, Frauscher Sensonic and the NVIDIA Corporation.
References


Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Our method is currently limited to natural images and short text prompts as inputs, and, thus its use for other types of images, such as medical or biological images, is unexplored. While we hypothesize that our approach could also be useful for similar data in other domains, this has not been assessed.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] One potential danger arises from users that overly rely on systems built on our method. For example in the domain of self-driving cars, users might start paying less attention to the traffic because of the AI-based driving system. Finally, our method might also be used to automate various simple tasks, which might lead to reduced need for particular jobs in production systems. As for almost all machine learning methods, our proposed method relies on human-annotated training data and thereby human decisions, which are usually strongly biased. Therefore, the responsible use of our method requires the careful selection of the training data and awareness of potential biases within those.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide the URL to a GitHub repository that contains the code.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5.1, Section 5.2 and Appendix Section A.2.2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We added error bars for all experiments for which this was computationally feasible (see Table 1).
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We used several different servers equipped with GPUs of different types, such as V100 and A100. The total amount of compute is roughly 11,000 GPU hours (with A100).

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] For the model implementations we used PyTorch (Paszke et al., 2017, BSD license) and for monitoring the runs we used Weights & Biases (Biewald, 2020, MIT license).
   (b) Did you mention the license of the assets? [Yes] See above.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We provide the code as supplementary material.
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [Yes] We only use public datasets that can be used for research purposes, such as the YFCC dataset which was published under the Creative Commons licence.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] As almost all computer vision and natural language datasets, the data suffers from many biases including social biases. We refer to Yang et al. (2020) for a detailed analysis of biases in such datasets.

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]