Cluster and Predict Latents Patches for Improved Masked Image Modeling

Anonymous authors
Paper under double-blind review

Abstract

Masked Image Modeling (MIM) offers a promising approach to self-supervised representation learning, however existing MIM models still lag behind the state-of-the-art. In this paper, we systematically analyze target representations, loss functions, and architectures, to introduce CAPI – a novel pure-MIM framework that relies on the prediction of latent clusterings. Our approach leverages a clustering-based loss, which is stable to train, and exhibits promising scaling properties. Our ViT-L backbone, CAPI, achieves 83.8% accuracy on ImageNet and 32.1% mIoU on ADE20K with simple linear probes, substantially outperforming previous MIM methods and approaching the performance of the current state-of-the-art, DINOv2.

1 Introduction

Recent advances in large-scale visual representation learning have established foundation models as a cornerstone of modern computer vision. Self-supervised representations have proven particularly effective in domains with limited annotations, such as satellite imagery (Tolan et al., 2024) and medical imaging (Xu et al., 2024; Vorontsov et al., 2024; Chen et al., 2024; Moutakanni et al., 2024; Dermyer et al., 2025), while enabling breakthroughs in more fundamental vision tasks like monocular depth estimation (Yang et al., 2024a; Bochkovskii et al., 2024; Yang et al., 2024b), keypoint matching (Edstedt et al., 2024), and tracking (Tumanyan et al., 2025). This shift from small supervised models to large self-supervised generalist models mirrors the evolution in natural language processing since the publication of BERT (Devlin et al., 2018) and GPT (Radford et al., 2018), where large-scale models pretrained on web-scale unlabeled data have become ubiquitous foundation models.

However, the impressive scalability of language models remains unmatched in vision: the best self-supervised visual encoders contain around one billion parameters (Oquab et al., 2024), hundreds of times smaller than current language models (Liu et al., 2024). A reason for this gap may be found in the discrepancy between the pretraining tasks used in vision and in language. The success of language models stems from the generality of the language modeling task: modeling the distribution of data, conditioned on context. Naturally, researchers have attempted to adapt this approach to computer vision, which resulted in masked image modeling (MIM): the prediction of missing image content given surrounding context.

Yet, existing MIM approaches have not matched the representation quality of alternative self-supervised methods. Pixel-level reconstruction objectives, *e.g.* MAE (He et al., 2022), provide good initialization for fine-tuning, but yield poor frozen representations, possibly because the target is too low-level to capture the task's inherent uncertainty (LeCun, 2022; Assran et al., 2023). Instead of reconstructing pixels, other works propose reconstructing in a pretrained encoder's latent space (Zhang et al., 2022; Fang et al., 2023; 2024). However, this approach requires an existing encoder and cannot learn representations from scratch.

The most promising direction uses the latent representation of the *online* model – or an exponential moving average (EMA) of it – as a learning target for the model, bootstrapping an informative latent space from scratch. This approach is used in the current state-of-the-art method for self-supervised learning of representations, DINOv2 (Oquab et al., 2024). However, these methods often suffer from poor stability (Zhou et al., 2022) and sensitivity to hyperparameters (Assran et al., 2023), requiring additional objectives like contrastive learning to produce competitive representations (Zhou et al., 2022; Oquab et al., 2024; Alkin et al., 2024).

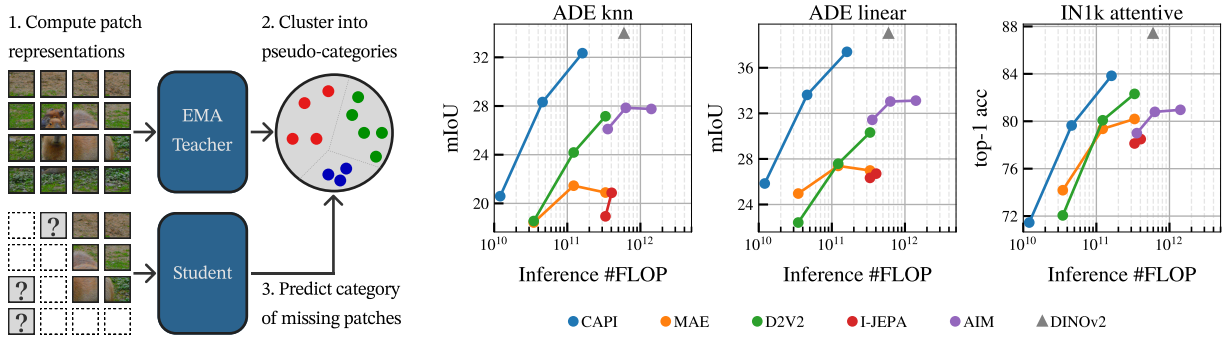


Figure 1: **CAPI Method overview:** image patches embedded by a teacher are grouped into clusters. Their assignments are then used as the training signal for the student. The teacher and the student are jointly learned via self-distillation. The loss is purely about predicting the content of missing patches and does not rely on augmentations or a contrastive loss. **Evaluation scores:** we evaluate frozen representations on ADE20K segmentation with a k-nn and linear probe and on ImageNet-1k classification with an attentive probe. We compare to MAE, data2vec 2.0, I-JEPA, and AIM. Compared to other masked image models, CAPI achieves higher performance with fewer FLOP, scaling well with model size, and approaches the scores of DINOv2+reg.

In this work, we focus on online latent masked image modeling. We isolate it from other stabilizing objectives, and systematically study the design choices it involves. We center the discussion around three aspects of the masked image modeling principle: the target representation, the formulation of the loss, and the architecture used to perform predictions. We show that with the right implementation, a simple masked image modeling objective can lead to features competitive with the current state of the art in SSL. In brief, our method relies on a pair of teacher-student vision transformers trained with self-distillation (fig. 1 left). The training signal for the student comes from the patch representations of the teacher, which is updated through an exponential moving average (EMA). The patch embeddings of the teacher are converted to soft assignments on pseudo-categories using a learned online clustering, while the student receives a partially masked image as input and is trained to predict the clustering assignments of the missing patches. Using this method, we train CAPI, a 300-million parameter visual encoder whose representations approach DINOv2's performance while significantly outperforming previous masked image models (fig. 1 right).

2 Related Work

Self-supervised representation learning. Self-supervised learning (SSL) is a pre-training paradigm in which a model is optimized to solve a pretext task on unlabeled data, often collected at scale. As the model is not trained to match specific human annotations, the benefit of this type of training is to produce a generalist model, that can be adapted to solve many different downstream tasks. Depending on the application, these tasks can be solved either by fully fine-tuning the mode, or by using the representations extracted with the frozen model. In the SSL literature, some works have focused mainly on full fine-tuning (He et al., 2022; Huang et al., 2023), while others have shown that frozen representations can reach excellent performance on a wide range of tasks, avoiding costly fine-tuning (Oquab et al., 2024) and generalizing to annotation-scarce domains where fine-tuning is not possible (Tolan et al., 2024; Xu et al., 2024). Historically, early work on self-supervised learning focused on hand-crafted pretext tasks such as predicting the rotation of an image (Gidaris et al., 2018), the relative position of patches (Doersch et al., 2015) or the color of a grayscale image (Zhang et al., 2016). Subsequent works pushed the field forward with methods based on clustering (Caron et al., 2018; 2020; 2021) and contrastive learning (Chen et al., 2020b; He et al., 2020). Nowadays, the best self-supervised encoder inherits from these families (Oquab et al., 2024) and complements them with a masked image modeling objective (Zhou et al., 2022). However, training with both global

and MIM objectives can prove difficult, as multiple components can interact negatively[1]. In this work, we study the masked image modeling component in isolation, suggesting improvements to properly stabilize the optimization objective in the absence of a global term.

Pixel reconstruction. Learning by predicting a missing part of an image was first proposed in Context encoders (Pathak et al., 2016). This was envisioned as conceptually similar to denoising autoencoders (Vincent et al., 2008), in the case where the noise is a masking process. More recently, the success of masked language modeling (Devlin et al., 2018) and autoregressive pretraining in natural language processing (Radford et al., 2018) brought a new wave of interest for transferring these ideas to vision. iGPT (Chen et al., 2020a) was the first effort to train a transformer (Vaswani et al., 2017) by generating pixels. Chen et al. (2020a) proposed rasterizing images to very low resolution, then training for autoregressive next-token prediction. Then, the advent of the ViT (Dosovitskiy et al., 2021) architecture sparked further interest in the field. Following the initial exploration of Dosovitskiy et al. (2021), BeiT (Bao et al., 2021) tried using the quantized latents of a dVAE as targets for a masked image pretraining, using the tokenizer from DALL·E (Ramesh et al., 2021). BeiT has proven useful as an initialization for further fine-tuning, but severely underperformed baselines in representation learning. To simplify BeiT, SimMIM, and MAE (Xie et al., 2021; He et al., 2022) concurrently proposed using raw pixels as targets. Thanks to a clever encoder/decoder architecture, MAE proved very stable and reached interesting representations, despite its simplicity. However, it still fell short of previous SSL methods in terms of representation quality: MAE models need to be scaled to a ViT-H size to match the linear probing performance of a 25× smaller DINOv1 ViT-S/16(Caron et al., 2021).

Latent reconstruction. Concurrently to MAE, Zhou et al. (2022) proposed iBOT. To obtain a more semantic tokenization, iBOT used the online output of the model being trained as the reconstruction target for masked image modeling. This led to good representations, and the method was reused to obtain the current state-of-the-art (Oquab et al., 2024). However, the iBOT objective was very unstable and required an additional DINO (Caron et al., 2021) loss to stabilize the training. The idea of using online representations as targets was then reused in data2vec (Baevski et al., 2022) and I-JEPA (Assran et al., 2023; Bar et al., 2024). I-JEPA in particular proposed reusing the encoder/decoder architecture of MAE and removing the projection head of iBOT, to obtain a more stable objective in the latent space. The improvements in I-JEPA established a new tradeoff between stability and performance, but it was still both sensitive to hyperparameters (-12 points on IN-1k when changing the "target scale" from $[0.15, 0.2]$ to $[0.125, 0.2]$ (Assran et al., 2023))) and weaker than DINOv2 (81.1 on ImageNet-1k, 5.4 below DINOv2 while using a model twice bigger).

Clustering in self-supervised learning. Our approach is also related to methods that use clustering for self-supervised learning. First of this line of work, DeepCluster (Caron et al., 2018) proposed using a simple k-means to obtain pseudo-labels. Subsequently, SwAV (Caron et al., 2020) introduced an online clustering to replace the offline k-means. DINO (Caron et al., 2021) built on SWaV, and made the clustering be implicitly learned by the MLP projection head of the student. Finally, iBOT (Zhou et al., 2022) proposed to reuse the DINO projection head for masked image modeling, implicitly using a clustering as target. Our approach reuses this idea of using a clustering to create the targets of a masked image modeling objective, but backtracks to the origin of this line of work: instead of using an MLP head that performs an implicit clustering, we use an explicit clustering. By separating the clustering process from the rest of the training, we isolate the two components, making the training more stable and transparent.

3 Approach

At a high level, masked image modeling involves masking a part of the input, feeding the visible region to a prediction model, and optimizing it to predict the content of the missing parts. Despite the simple formulation, the effectiveness of reconstruction-based methods and the properties of the trained models are dramatically influenced by a number of design choices. In this section, we discuss the three aspects of

[1] Zhou et al. (2022) showed that using a shared head for the DINO and iBOT losses gave better results at small scales. However, at large scale, Oquab et al. (2024) observed that this caused the iBOT loss to explode late into the training, significantly degrading performance. Untying the two heads prevented the two losses from competing directly, stabilizing the training.

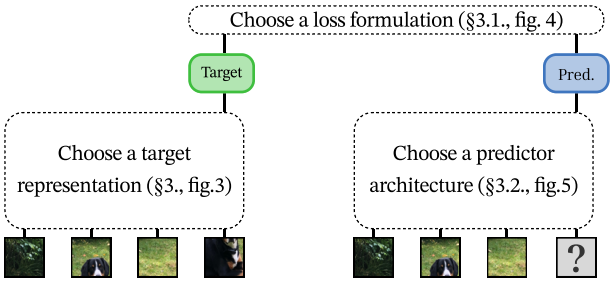


Figure 2: Overview of the components of a reconstruction-based model. We identify three main choices involved in designing a masked image model: the choice of targets (fig. 3), the loss function (Section 3.1, fig. 4) and the architecture of the predictor (Section 3.2, fig. 5).

masked image modeling depicted in Figure 2: the type of patch representation used as target (fig. 3), the loss formulation (Section 3.1, fig. 4), and the prediction architecture (Section 3.2, fig. 5). Based on our findings, we introduce CAPI, an SSL model that enjoys stable learning and strong representation capabilities.

Overview of training. In short, our main design choices are: first, we reconstruct images in latent space with a teacher-student framework, following iBOT. Then, we formulate our loss using a clustering component, inspired by DeepCluster, SwAV, and DINO, and draw inspiration from the regularization methods they introduce, in particular the Sinkhorn-Knopp (Sinkhorn & Knopp, 1967). Finally, we employ a cross-attention predictor model, separate from the encoder, to perform reconstructions, following crossMAE (Fu et al., 2024).

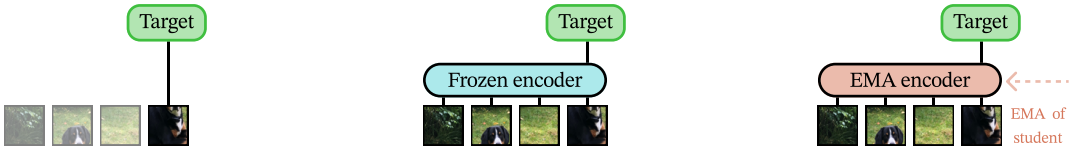
Our encoder and the predictor are transformers (Dosovitskiy et al., 2021) and, during pre-training, they operate in tandem as the student (fig. 1 left). The teacher is an EMA of the encoder. The typical values for one training iteration using square images of side 224 pixels are:

- We pass the full image to the teacher, collect $n = 14 \times 14 = 196$ patch tokens, and apply an online clustering to obtain soft assignments that will be used as learning targets.
- The encoder receives a partial view of the input image: we apply a patch embedding layer to obtain n patch tokens, we drop $p_{\text{drop}} \times n$ of these patches and pass to the encoder the remaining $n_{\text{keep}} = (1 - p_{\text{drop}}) \times n = 69$ patches ($p_{\text{drop}} = 65\%$).
- The encoder takes these $n_{\text{keep}} = 69$ tokens along with $n_{reg} = 16$ learnable register tokens (Darcet et al., 2024), and processes them to obtain $n_{\text{encoded}} = n_{\text{keep}} + n_{\text{reg}} = 85$ encoded tokens.
- We sample $n_{\text{pred}} = 7$ coordinates among the dropped set, and for each we forward a `[MSK]` token through the predictor, which predicts their assignment by cross-attending to the encoded view.

A detailed visual diagram of the method can be found in fig. 8.

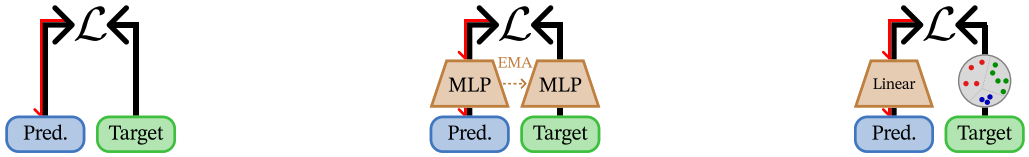
3.1 Clustering-based loss formulation

Latent clustering methods such as SwAV and DINO employ a cross-entropy loss between the student and teacher output distributions. These distributions, produced by a linear or MLP head, can be seen as soft



(a) Pixel targets (iGPT, MAE, AIM)
(b) Frozen teacher (BeiT,PeCo,EVA)
(c) EMA teacher (iBOT, Data2Vec, I-JEPA, Ours)

Figure 3: The target representations commonly used in MIM. We focus on the EMA representations.



(a) Direct loss (MAE, I-JEPA) (b) DINO loss (iBOT, DINOv2) (c) Clustering (proposed)

Figure 4: The different loss formulations considered here. We depict in red the flow of the gradient.

cluster memberships, where the centroids correspond to the prototypes. Replicating the DINO objective, iBOT proposes to pass the student predictions through an MLP head and to pass the teacher embeddings through the EMA of this head. But this ignores a specificity of masked image modeling: while in DINO both targets and predictions are `[CLS]` tokens, in iBOT the targets are patch tokens while the predictions are special `[MSK]` tokens. This causes a *distribution mismatch*: the MLP head of the student is trained with `[MSK]` inputs but is instead applied to regular patch tokens in the teacher. This mismatch would be even stronger in an asymmetric architecture as in MAE or I-JEPA, where the targets and predictions come from two different networks (see fig. 5). Without the stabilizing effect of the DINO loss, the iBOT formulation is unable to bootstrap itself and results in trivial representations (see table 1c). Our proposition is simple: we decouple the training of the teacher projection from the student's head by directly learning an online clustering of the teacher patch tokens. This way, the training remains stable even in the absence of a stabilizing loss.

Online clustering. Inspired by the minibatch k-means algorithm and the SwaV loss, we define our online clustering process as follows. Let $X \in \mathbb{R}^{n \times d}$ be the output of the teacher, with row vectors x_i for $i \in \{1, \dots, n\}$. We apply an L2 normalization and a linear projection to obtain the assignment logits $l_i \in \mathbb{R}^p$:

$$l_i = C \cdot \frac{x_i}{||x_i||}, \tag{1}$$

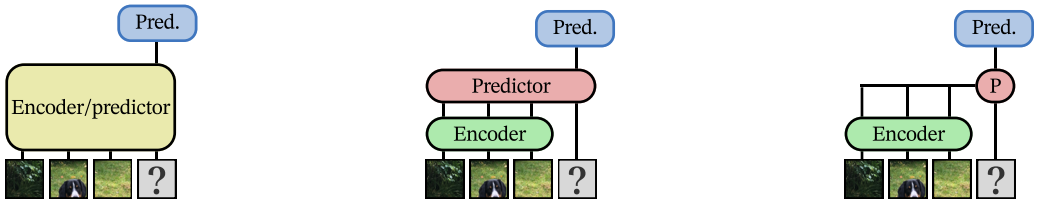
where C is a matrix in $\mathbb{R}^{p \times d}$, whose rows are the p "centroids" in dimension d. We then apply a softmax with temperature τ to obtain soft assignments:

$$a_i = \frac{\exp(l_i/\tau)}{\sum_{k=1}^{p} \exp(l_k/\tau)}. \tag{2}$$

We wish to estimate C by solving the following problem:

$$\min_{C} \quad -\sum_{i=1}^{n} \sum_{k=1}^{p} a_i^{(k)} \log a_i^{(k)}, \tag{3}$$

where $a_i^{(k)}$ is the k-th coordinate in a_i. By minimizing the entropy, we force the assignments to be as close to one-hot as possible, which pushes the centroids towards their "assigned" samples. This can be seen as a form



(a) Fused predictor (BeiT, iBOT) (b) Self-att. predictor (MAE, I-JEPA) (c) Cross-att. predictor (CrossMAE, ours)

Figure 5: The different predictor architectures discussed in the paper. Here, the boxes each represent a transformer. The black lines represent the residual stream for a token.

of clustering with a logistic loss. However, simply solving this problem can result in empty clusters. This is a common problem in mini-batch k-means (Sculley, 2010), and is usually solved by adding a reassignment phase: the centroids of the empty clusters are discarded and moved next to another non-empty centroid. In our case, we do not wish the centroids to move that abruptly, as it could disturb the training of the student. Instead, we propose to use the Sinkhorn-Knopp (SK) algorithm (Sinkhorn & Knopp, 1967), used in SwAV (Caron et al., 2020) and DINOv2 (Oquab et al., 2024). Using the SK algorithm, we obtain a', an assignment where the distribution of tokens over the clusters is near-uniform:

$$\{a'_1, \dots, a'_n\} \leftarrow \mathrm{SK}(\{l_1, \dots, l_n\}, \tau'), \tag{4}$$

with τ' another temperature parameter. Note that we do not backpropagate through the SK algorithm. We adapt the loss in Problem (3), and learn the centroids C by minimizing the cross-entropy between a and a':

$$-\sum_{i=1}^{n}\sum_{k=1}^{p} {a'}_i^{(k)} \log a_i^{(k)}. \tag{5}$$

We minimize this loss with AdamW (Loshchilov, 2017) alongside the training of the main model.

Positional collapse. A crucial point of self-supervised learning is avoiding trivial solutions, *i.e.* situations where the model minimizes the loss without learning useful features. These failure modes, also known as representation collapse, are usually addressed by adding regularization mechanisms. For example, color jittering (Chen et al., 2020b) helps prevent the model from focusing uniquely on color. While training CAPI, we observed a specific type of representation collapse as the positional encoding started outweighing the content of the patch embeddings. In the extreme case, the model learns to predict the position of the masked tokens instead of their content, resulting in a zero loss with a trivial model. In most observed cases, both content and position information are *entangled* in the target representation, while in the ideal scenario, the target would consist purely of semantic features.

We propose a simple solution to alleviate the problem. By running the SK separately at each position in Eq. (4), we force the joint distribution of tokens over positions and clusters to be near-uniform. Uniformity of the joint distribution directly implies zero mutual information between the clustering and the targets. This way, the modified SK ensures that our targets contain no positional information.

3.2 Predictor architecture

Model architecture is another important component in reconstruction-based SSL. Two broad categories are widely used in previous work. BeiT, iBOT, and SimMIM use a *fused* architecture: a single vision transformer that takes as inputs patches and mask tokens (Fig. 5a). Fused architectures are difficult to train and yield poor results, as reported in previous work (Zhou et al., 2022) and confirmed by our experiments. MAE uses a *split* architecture, with an encoder that only forwards the patches, saving memory and compute, and a predictor that forwards both patches and mask tokens (Fig. 5b).

In this work, we use an even lighter architecture, which was initially explored in crossMAE (Fu et al., 2024) for pixel-based reconstruction. In this case, the predictor forwards only the mask tokens (Fig. 5c), which can access the context of the encoded patches via cross attention. Using a cross-attention predictor has two main advantages. First, it allows further efficiency gains, as we only forward a reduced set of tokens in the predictor, and we can even subsample this set of tokens. Second, mask tokens do not interact with each other in the cross-attention mechanism, *i.e.* each prediction is independent of other positions. This alleviates the need for multiple predictor forward passes with different prediction sets, as used in I-JEPA.

4 Experiments

In this section, we report empirical evaluations of our model. We describe experimental details and present some ablation studies. Then we discuss whole-image understanding results and dense prediction results.

4.1 Experimental setup

Pretraining dataset. Most methods from the self-supervised learning literature choose to pretrain on ImageNet-1k. This dataset is usually chosen because of its relatively small size, allowing for easy experimentation, and the ability to compare to existing methods pretrained on it. However, this has led to an overspecialization of SSL methods to the type of object-centric images found in ImageNet-1k. Recent foundation models obtain state-of-the-art results by exploiting much larger datasets, such as ImageNet-22k (Zhou et al., 2022) and LVD-142M (Oquab et al., 2024). If we are to design a method that can produce new foundation models, we believe it is crucial to design it to be able to handle such large datasets.

To this end, we carry out all ablation experiments on ImageNet-22k. It is composed of 14M images from 22k categories taken from the WordNet ontology. Although it is close to ImageNet-1k in nature, its much larger size and diversity make it suitable to train excellent foundation models, as reported by Oquab et al. (2024).

For our longer experiments, we train on multiple datasets: ImageNet-1k, for comparability with previous works, ImageNet-22k, to test scaling, Places205, to test training on more diverse and less object-centric data, and finally a large-scale automatically curated dataset. Following Oquab et al. (2024), we filter web-crawled images to obtain a dataset of roughly 140M images, which we call Loca.

Model architecture. We do all our experiments with a Vision Transformer (Dosovitskiy et al., 2021) of 300M parameters (ViT-L). This architecture is widely used in various computer vision tasks, and most baselines provide a model of comparable size. We equip the vision transformer with registers (Darcet et al., 2024). These additional tokens were recently proposed as a way to add an information buffer, which enabled the model to produce smoother feature maps. For the decoder, we use 12 transformer blocks that cross-attend to the output of the encoder. This is similar to a standard transformer decoder (Vaswani et al., 2017), with the difference that we do not include self-attention layers. In this decoder, every token is forwarded independently and separately attends to the encoder output. When using a different encoder size, we align the embedding dimension, MLP ratio, and number of attention heads of the decoder to those of the encoder, and use a decoder depth equal to half that of the encoder.

Implementation Details. The learning rate follows a linear warmup followed by a cosine annealing. We truncate out the last 20% of the cosine, as proposed in I-JEPA (Assran et al., 2023). To simplify the choices of parameters and schedules, we set the teacher EMA momentum to $\mu = 1 - lr$, and we set the learning rate for the clustering to half of the backbone learning rate. The impact of the most important hyperparameters will be discussed in section 4.2. All our pretraining hyperparameters are summarized in table 6.

Evaluation protocol. All the evaluations reported in this paper fall into two categories: image classification and semantic segmentation. For all classification tasks, we use an *attentive probe* (Assran et al., 2023; El-Nouby et al., 2024; Bardes et al., 2023). We use this evaluation protocol because our model does not learn a single global image representation, preventing the use of a linear probe. In this evaluation, we train an attentive pooling to extract a global vector and use this vector as input to a linear layer. The parameters of this probe are trained in a supervised fashion, and we report accuracy on the validation set. For segmentation tasks, we use lightweight classifiers on top of frozen local features. Previous works used a linear head trained with gradient descent on features of images augmented with various augmentations (Oquab et al., 2024). To obtain a more lightweight evaluation, we simply extract the features from all images in the dataset without augmentations, then train a linear logistic regression on these features with L-BFGS (Byrd et al., 1995) using an off-the-shelf library (Raschka et al., 2020). Although this results in lower mIoU numbers, the simplicity of the evaluation allows us to grid over different hyperparameters, producing very robust results. For an even more lightweight classifier, we also consider a non-parametric k-NN segmentation evaluation. For each test patch, we retrieve k most similar patches in the training data and pool the segmentation label for that patch. We chose the optimal regularization parameters by doing a grid search using 10% of the training set. For all segmentation tasks, we measure performance using mIoU.

	ADE	IN1k
Fused	23.8	73.1
Split, self-attn	27.9	77.7
Split, cross-attn	**29.1**	**81.4**

(a) Predictor architecture

	ADE	IN1k
random	23.6	76.4
block	25.6	79.9
inv. block	27.2	80.7
inv. block `+roll`	**29.1**	**81.4**

(b) Masking strategy

head	loss	ADE	IN1k
∅	I-JEPA	23.7	79.3
MLP	iBOT	1.7	11.1
MLP	CAPI	26.4	80.8
Linear	CAPI	**29.1**	**81.4**

(c) Loss formulation

	ADE	IN1k
[0.2, 1.0]	27.9	81.4
[0.6, 1.0]	**29.1**	**81.4**
[1.0, 1.0]	28.9	80.9

(d) Crop range

	ADE	IN1k
55%	28.0	81.1
65%	**29.1**	**81.4**
75%	28.1	81.2

(e) Masking ratio

depth	width	ADE	IN1k
5	1536	**30.9**	**81.5**
12	1024	29.1	81.4
21	768	28.3	81.3

(f) Predictor shape

	ADE	IN1k
0	25.9	79.3
16	**29.1**	**81.4**

(g) Number of registers

	ADE	IN1k
learnable	**30.0**	**81.6**
RoPE	29.1	81.4

(h) Positional encoding

	ADE	IN1k
Standard	28.5	81.3
Proposed	**29.1**	**81.4**

(i) Sinkhorn-Knopp algorithm

Table 1: Ablation study of the main parameters and design choices in our algorithm. We report both image segmentation and classification. We highlight the default setting in gray, and bold the best-performing solution. An in-depth analysis of these results is provided in Sec. 4.2.

Baselines. We compare to the performance of previous models trained using masked image modeling: BeiT (Bao et al., 2021), MAE (He et al., 2022), data2vec 2.0 (Baevski et al., 2023), I-JEPA (Assran et al., 2023), and AIM (El-Nouby et al., 2024). To provide additional points of comparison, we report in grey the performance of iBOT (Zhou et al., 2022) and DINOv2+reg (Oquab et al., 2024; Darcet et al., 2024), who use a DINO loss to stabilize a MIM objective.

4.2 Ablation Studies

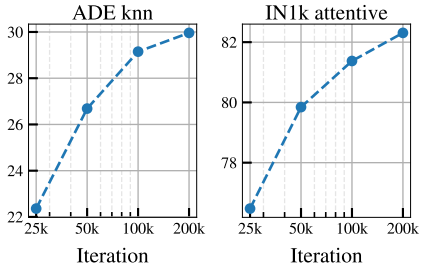
We conduct extensive ablation studies to study the effect of design choices on performance. To make the ablation study more tractable, we train on the ImageNet-22k dataset for 100k iterations with a patch size of 16. To provide slightly more robust results, the default setting was run twice with different seeds, and the results of the two runs were averaged. All results are presented in Table 1.

Predictor architecture. We evaluate the different predictor architectures. The split predictor produces better representations while training 32% faster than the fused predictor (table 1a). Using pure cross-attention in the predictor obtains better representations and allows an additional 18% speedup by avoiding a forward on patch tokens. It also removes the dependency between different predictions, alleviating the need for repeated forwards of the predictor as in I-JEPA.

Masking strategy. The most common masking strategies in the literature are random masking, block masking (inpainting), or inverse block masking (outpainting). Inverse block masking induces a bias on the position of the masked patches: most often, the model will see the center of the image, and predict the edges. To prevent this, we propose applying a random circular shift to the mask before using it (`+roll`). This ensures that all positions in the image are equally likely to be masked. We ablate the type of masking in table 1b. The masking ratio is 65% for all strategies except random masking, where it is set to 90%, which increases performance. Random masking is much less effective than the other strategies, and inverse block masking works best, with a clear improvement when using `+roll`.

Loss formulation. We evaluate the different strategies for computing a loss function discussed in Sect. 3.1. We compare the performance of direct Huber loss, an iBOT loss with an MLP head, as well as our clustering-

#prototypes	ADE	IN1k
1024	14.9	73.8
2048	19.8	77.4
4096	27.5	80.7
8192	28.5	81.3
16384	**29.1**	81.4
32768	**29.1**	**81.7**



dataset	ADE20K	IN1k
IN1k	28.0	**81.7**
IN22k	29.1	81.4
Loca	**30.6**	81.2

Figure 6: Additional ablation experiments. **(Left)** Influence of the number of prototypes. **(center)** Influence of the training length. Each point here is an independent training. **(right)** Influence of the training dataset.

based loss with a linear or MLP head. With no head and direct loss, the model starts well but quickly regresses to worse representations. The iBOT head does not work in our setup, probably because of the split predictor design. Finally, the proposed clustering head alleviates all these issues and allows for stable training and good representations.

Crop range. To prevent overfitting, we use random cropping and flipping augmentations. We tweak the bounds of the cropping scale to $[c, 1.0]$ and try various c. We observe that the method can train well without any augmentation, resizing the training images to a fixed 224x224 resolution (80.9 on ImageNet with $c = 1.0$). We get better scores with minor cropping augmentation with a range of $[0.6, 1.0]$.

Masking ratio. In our model, we need to set the ratio m of image patches that are masked, and reconstructed. We train our model for various m and check the final performance. The optimal masking ratio seems to be around 0.65, but the algorithm seems quite robust to the choice of this parameter.

Predictor shape. We study the impact of the predictor depth on performance. We train the model with predictors of various depths and adapt the width to match the total number of parameters. Shallow networks will have a larger width. We see that a more shallow predictor leads to better performance, but in our informal experiments, we have observed that such architectures are less stable in long schedules. For this reason, we stick to the predictor with 12 layers and a width of 1024.

Registers. In our model, the local feature maps serve as a supervisory signal to train the model, so high-quality feature maps are crucial. To this end, we use register tokens, which were proposed to improve the quality of feature maps. We see that using registers has a large effect on performance, with an improvement of 3.2 points on ADE and 2.1 points on ImageNet when using 16 registers (table 1g).

Positional encoding. In our experiments, we consider using different versions of positional encoding. We try the classic learnable position embeddings as well as relative ones such as RoPE (Su et al., 2024). Because of the ease of use and transferability to higher resolutions, we settle on using RoPE.

Sinkhorn-Knopp algorithm. We describe the problem of positional collapse in section 3.1. Changing the set of points considered in the Sinkhorn solves it, improving stability and granting a small performance increase (table 1i). We did not observe any positional collapse when using it.

Number of prototypes. We evaluate the effect of varying the number of prototypes p in our clustering-based loss. The performance generally increases with the number of prototypes, at the cost of a higher memory footprint. We use $K = 16384$, which strikes a good balance between memory and performance.

Scaling. We study the effect of three scaling axes on the performance of our model: number of parameters, training length, and dataset size. To study the scaling potential of our algorithm, we train additional ViT-S and ViT-B models. We report the performance of the family of models in Fig. 1. We see that performance

Model	Arch.	Dataset	ImageNet					iNat	Places205	SUN397
			val	v2	ReaL	A	ObjectNet			
iBOT	ViT-L/16	IN1k	80.9	86.5	70.3	41.9	28.9	70.5	62.0	64.6
I-JEPA	ViT-H/14	IN1k	79.5	85.3	68.7	37.0	22.6	64.2	59.9	61.9
MAE	ViT-L/16	IN1k	79.4	85.3	68.9	38.3	19.3	69.3	60.6	61.8
Data2Vec 2.0	ViT-L/16	IN1k	80.1	85.7	69.4	42.1	24.6	65.4	61.9	64.4
CAPI	ViT-L/14	IN1k	**82.9**	**87.6**	**72.9**	**47.5**	**43.7**	**76.8**	**65.4**	**70.9**
DINOv2	ViT-g/14	LVD-142M	87.4	90.3	80.1	67.0	81.7	88.3	68.8	79.3
BeiT	ViT-L/16	IN22k	40.8	46.1	30.7	8.7	2.1	26.5	36.8	29.9
I-JEPA	ViT-H/14	IN22k	78.1	84.3	67.4	38.6	25.1	67.7	60.2	65.5
AIM	ViT-600M/14	DFN-2B+	79.0	84.8	67.9	41.0	20.4	73.5	62.5	66.1
CAPI	ViT-L/14	IN22k	83.6	88.1	74.3	55.2	52.4	**82.0**	66.3	74.5
CAPI	ViT-L/14	Places205	79.2	84.7	68.4	39.1	33.0	73.4	**68.6**	**77.5**
CAPI	ViT-L/14	Loca	**83.8**	**88.2**	**74.8**	**55.3**	**56.8**	81.2	67.1	75.6

Table 2: Image classification results. For each baseline method, we report the model size closest to ViT-L/14.

consistently improves with model size, and for all models, our algorithm outperforms the state-of-the-art. In Fig. 6 (center), we report the effect of the number of training iterations and pretraining dataset on performance. We train models using the ablation configs, on ImageNet-22k and with patch size 16. In Fig. 6 (right), we show the influence of the training data on model performance. Using a larger dataset has a positive effect on segmentation, while only slightly deteriorating the ImageNet-1k accuracy.

4.3 Results

Image classification. We evaluate our model and compare it to state-of-the-art reconstruction-based SSL models. We run the evaluation on four datasets including object recognition, fine-grained classification, and scene recognition. We use ImageNet-1k (Russakovsky et al., 2015), iNaturalist 2021 (Van Horn et al., 2021), Places205 (Zhou et al., 2017), and SUN397 (Xiao et al., 2010). For ImageNet, we also report OOD robustness by running inference on additional test sets: ImageNet-V2 (Recht et al., 2019), ImageNet-ReaL (Beyer et al., 2020), ImageNet-A (Hendrycks et al., 2021), and ObjectNet (Barbu et al., 2019). For each model, we resize the image to 224×224 and collect the patch tokens output by the model. We feed those to an attentive probe implemented as a single layer of multi-head cross-attention with a single query (head size $d//64$, no residual). For c classes and an embedding size d, the probe contains $2d^2 + (3 + c)d$ parameters, which are trained with AdamW for 10 epochs, selecting the best learning rate for each model/task on a held-out split of the training set. More details on the protocol are available in appendix F. All the results are summarized in Table 2.

We see that our model outperforms all previous state-of-the-art models, by a large margin. When training on ImageNet-1k, we observe very good results, outperforming all other reconstruction-based models of comparable size. CAPI excels particularly on out-of-distribution generalization, outperforming all baselines by more than 19 points on ObjectNet. Additionally, while the gap with other methods is somewhat limited on ImageNet, the difference in scene classification (SUN397) is much larger. Interestingly, we observe that CAPI works particularly well on larger and more diverse datasets. Our three CAPI models outperform all baselines on all datasets, except the CAPI-Places205 which is slightly below AIM on ImageNet-A. It should be noted, however, that AIM was trained on more than 2 billion images, with a sampling distribution tailored towards ImageNet. CAPI significantly reduces the gap between reconstruction-based methods and our topline DINOv2+reg: the gap on ImageNet goes from 8.4 points to 3.6, and on SUN397 goes from 13.2 to 1.8.

Dense image understanding. As we have seen above, our model allows training high-quality local features that can be successfully pooled to solve image-level tasks. We also want to evaluate our model on dense prediction problems such as image segmentation. To this end, we run k-NN and linear segmentation following the protocol described in the experimental details. We run this for all the baselines reported above on

			ADE-20k		Pascal-VOC		Cityscapes	
Model	Arch.	Dataset	k-NN	linear	k-NN	linear	k-NN	linear
iBOT	ViT-L/16	IN1k	26.0	30.7	60.2	68.8	35.7	39.8
I-JEPA	ViT-H/14	IN1k	20.8	25.7	56.7	63.6	26.4	34.5
MAE	ViT-L/16	IN1k	21.5	27.4	53.7	61.5	32.8	38.5
Data2Vec 2.0	ViT-L/16	IN1k	24.2	27.6	57.5	58.1	32.8	38.2
CAPI	ViT-L/14	IN1k	**29.2**	**34.4**	**60.7**	**69.7**	**35.6**	**41.7**
DINOv2	ViT-g/14	LVD-142M	34.0	39.0	63.0	72.8	42.0	46.8
BeiT	ViT-L/16	IN22k	3.5	8.3	6.9	19.1	15.6	24.0
I-JEPA	ViT-H/14	IN22k	18.9	26.3	55.0	64.2	23.2	34.2
AIM	ViT-600M/14	DFN-2B+	26.1	31.4	60.2	67.0	32.1	38.2
CAPI	ViT-L/14	IN22k	29.7	35.2	61.1	70.4	35.2	41.0
CAPI	ViT-L/14	Places205	**35.2**	**39.1**	61.7	69.4	**39.5**	**44.6**
CAPI	ViT-L/14	Loca	32.1	37.2	**63.8**	**72.7**	38.9	44.3

Table 3: Comparison with the state of the art on image segmentation using frozen features. We report both k-NN and linear segmentation performance. For reference, we also report the performance of some other non-MIM SSL models. This shows that CAPI narrows the gap using only a MIM approach.

three datasets. We use ADE-20k (Zhou et al., 2017), Pascal VOC 2012 (Everingham et al., 2010), and Cityscapes (Cordts et al., 2016). We report mIoU for all configurations in Table 3.

As in the classification evals, CAPI trained on ImageNet-1k outperforms all reconstruction-based baselines by quite a wide margin on all evaluation setups. When training on larger datasets, the conclusion is similar: CAPI outperforms all baselines by a wide margin and even beats DINOv2+reg in some setups. When trained on Places205, CAPI achieves mIoU 1.2 points higher than DINOv2+reg. To our knowledge, this is the first time DINOv2+reg is bested on segmentation with frozen features on ADE20K. DINOv2+reg, however, remains the most versatile model, with good results across the board and the best results on Cityscapes.

4.4 Additional explorations

As a final set of experiments, we investigate some additional properties of our model. We investigate its robustness to change of input resolution and try to obtain global representations using the predictor.

High-resolution image understanding. Our model was trained on 224×224 images. To compare with the I-JEPA model trained natively at high resolution, we try to evaluate our model on 448×448 images. In table 4, we see that our model does not require high-resolution training or evaluation to achieve the best performance. Our model trained at 224 and evaluated at 224 outperforms the large I-JEPA model trained at 448 and evaluated at 448. Moreover, using RoPE embeddings, CAPI is more robust to resolution changes: it loses only 0.5% (versus 1.0% for I-JEPA) when increasing the resolution.

Qualitative feature analysis. We propose a qualitative assessment of the dense features in Fig. 7. The dense features computed with CAPI are amongst the most discriminative and smooth. We see the emergence of distinct objects, without much noise in uniform regions. We observe that the CAPI features are less noisy

Method	train res.	eval@224	eval@448
I-JEPA-H	224	79.4	78.4
I-JEPA-H	448	79.6	82.5
CAPI-L	224	**83.8**	**83.5**

Table 4: ImageNet-1k attentive probing accuracy of I-JEPA and CAPI at different input resolutions.

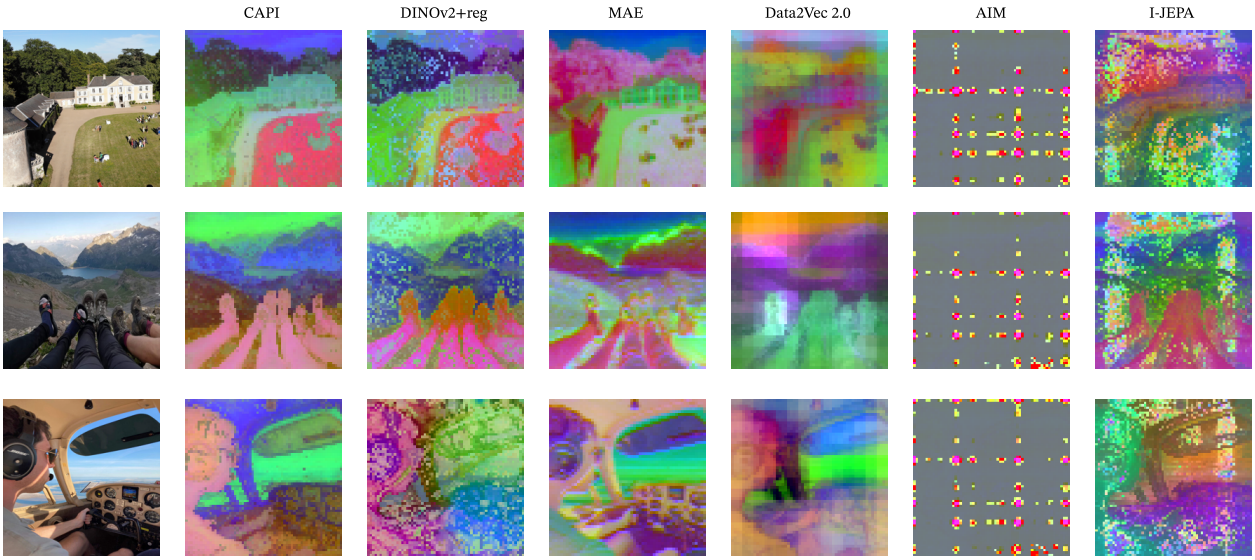


Figure 7: Visualization of the features of CAPI and baseline models. We apply a PCA to the features and map the three first components to RGB. The features produced by CAPI are discriminative and smooth.

than the ones from DINOv2+reg, while being more focused on semantics and less on colors than the MAE features. For example, the shaded building in the first image has CAPI features similar to the other buildings, while in MAE the features are closer to other dark areas of the image.

Obtaining global representations. In all the previous evaluations reported in the experimental section, we trained a head on top of local features. Our model does not provide an aggregate representation like the `[CLS]` token in DINOv2. In this experiment, we try to exploit the predictor to obtain global image representations. We forward the whole image through the encoder, and then pass the same amount of mask tokens through the first attention layer of the predictor, cross-attending to the patch tokens. We obtain a global representation by average-pooling the output of this predictor attention layer. We train a linear model on this representation on several classification datasets and compare it with the average pooling of the patch embeddings in table 5. We see that using the attention pooling learned by the predictor provides better representations than averaging local features from the encoder.

5 Discussion and Concluding Remarks

In this paper, we have proposed a novel reconstruction-based self-supervised learning algorithm. Our algorithm is based on an online clustering of dense features computed with a teacher network. The latent assignments are used as targets to train the student. We propose to implement the student as an encoder followed by a predictor: a cross-attention decoder. The teacher is updated as an EMA of the encoder. The proposed algorithm is simple and allows the training of a state-of-the-art model. Our ViT-L outperforms all available reconstruction-based models, including much larger architectures. We have shown promising scaling trends until the 300M model sizes of the ViT family, opening up a potential for further scaling in future work.

	IN1k		iNat21	SUN397
Representation	k-NN	Linear	Linear	Linear
avg. pooling	57.1	77.1	49.1	73.3
predictor pooling	**73.8**	**81.1**	**69.6**	**77.4**

Table 5: Classification using predictor representations, compared to the average pooling of the patch tokens.

References

Benedikt Alkin, Lukas Miklautz, Sepp Hochreiter, and Johannes Brandstetter. Mim-refiner: A contrastive learning boost from intermediate pre-trained representations, 2024. 1

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, 2023. 1, 3, 7, 8, 22

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *ICML*, 2022. 3, 22

Alexei Baevski, Arun Babu, Wei-Ning Hsu, and Michael Auli. Efficient self-supervised learning with contextualized target representations for vision, speech and language. In *ICML*, 2023. 8

Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2021. 3, 8, 17, 22

Amir Bar, Florian Bordes, Assaf Shocher, Mido Assran, Pascal Vincent, Nicolas Ballas, Trevor Darrell, Amir Globerson, and Yann LeCun. Stochastic positional embeddings improve masked image modeling. In *ICML*, 2024. 3

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *NeurIPS*, 2019. 10

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. V-jepa: Latent video prediction for visual representation learning, 2023. 7

Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *CoRR*, abs/2006.07159, 2020. URL https://arxiv.org/abs/2006.07159. 10

Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second, 2024. 1

Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM*, 1995. 7, 20

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 2, 3

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020. 2, 3, 6

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2, 3, 17

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020a. 3

Richard J Chen, Tong Ding, Ming Y Lu, Drew FK Williamson, Guillaume Jaume, Andrew H Song, Bowen Chen, Andrew Zhang, Daniel Shao, Muhammad Shaban, et al. Towards a general-purpose foundation model for computational pathology. *Nature Medicine*, 2024. 1

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020b. 2, 6

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 11

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024. 4, 7, 8, 22

Patrick Dermyer, Angad Kalra, and Matt Schwartz. Endodino: A foundation model for gi endoscopy, 2025. 1

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, 2018. 1, 3

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 2

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3, 4, 7

Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19790–19800, 2024. 1

Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Angel Bautista, Alexander Toshev, Vaishaal Shankar, Joshua M Susskind, and Armand Joulin. Scalable pre-training of large autoregressive image models. In *ICML*, 2024. 7, 8, 21, 22

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 11

Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023. 1

Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 2024. 1

Letian Fu, Long Lian, Renhao Wang, Baifeng Shi, Xudong Wang, Adam Yala, Trevor Darrell, Alexei A Efros, and Ken Goldberg. Rethinking patch dependence for masked autoencoders, 2024. 4, 6

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 2

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 2

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 1, 2, 3, 8, 17

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, 2021. 10

Zhicheng Huang, Xiaojie Jin, Chengze Lu, Qibin Hou, Ming-Ming Cheng, Dongmei Fu, Xiaohui Shen, and Jiashi Feng. Contrastive masked autoencoders are stronger vision learners. *TPAMI*, 2023. 2

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 2007. 23

Diederik P Kingma. Adam: A method for stochastic optimization. In *ICLR*, 2014. 20

Yann LeCun. A path towards autonomous machine intelligence, 2022. 1

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report, 2024. 1

I Loshchilov. Decoupled weight decay regularization, 2017. 6, 20

maintainers and TorchVision contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016. 20

Théo Moutakanni, Piotr Bojanowski, Guillaume Chassagnon, Céline Hudelot, Armand Joulin, Yann LeCun, Matthew Muckley, Maxime Oquab, Marie-Pierre Revel, and Maria Vakalopoulou. Advancing human-centric ai for robust x-ray analysis through holistic self-supervised learning, 2024. 1

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *TMLR*, 2024. 1, 2, 3, 6, 7, 8

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 3

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. 1, 3

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 3

Sebastian Raschka, Joshua Patterson, and Corey Nolet. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence, 2020. 7, 20

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019. 10

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 10

David Sculley. Web-scale k-means clustering. In *WWW*, 2010. 6

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 1967. 4, 6

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024. 9

Jamie Tolan, Hung-I Yang, Benjamin Nosarzewski, Guillaume Couairon, Huy V Vo, John Brandt, Justine Spore, Sayantan Majumdar, Daniel Haziza, Janaki Vamaraju, et al. Very high resolution canopy height maps from rgb imagery using self-supervised vision transformer and convolutional decoder trained on aerial lidar. *Remote Sensing of Environment*, 2024. 1, 2

Narek Tumanyan, Assaf Singer, Shai Bagon, and Tali Dekel. Dino-tracker: Taming dino for self-supervised point tracking in a single video. In *ECCV*, 2025. 1

Grant Van Horn, Elijah Cole, Sara Beery, Kimberly Wilber, Serge Belongie, and Oisin Mac Aodha. Benchmarking representation learning for natural world image collections. In *CVPR*, 2021. 10

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3, 7

P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 3

Eugene Vorontsov, Alican Bozkurt, Adam Casson, George Shaikovski, Michal Zelechowski, Kristen Severson, Eric Zimmermann, James Hall, Neil Tenenholtz, Nicolo Fusi, et al. A foundation model for clinical-grade computational pathology and rare cancers detection. *Nature medicine*, 2024. 1

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 10

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv*, 2021. 3

Hanwen Xu, Naoto Usuyama, Jaspreet Bagga, Sheng Zhang, Rajesh Rao, Tristan Naumann, Cliff Wong, Zelalem Gero, Javier González, Yu Gu, et al. A whole-slide foundation model for digital pathology from real-world data. *Nature*, 2024. 1, 2

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024a. 1

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2, 2024b. 1

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016. 2

Xinyu Zhang, Jiahui Chen, Junkun Yuan, Qiang Chen, Jian Wang, Xiaodi Wang, Shumin Han, Xiaokang Chen, Jimin Pi, Kun Yao, et al. Cae v2: Context autoencoder with clip target, 2022. 1

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 10, 11

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022. 1, 2, 3, 6, 7, 8, 17

A Detailed overview

In Figure 8, we provide a detailed overview of the complete method, with tensor sizes annotated for a reference CAPI ViT-L/14 model.

B Loss curve

We report in Figure 9 the loss curve of our CAPI ViT-L model. After an initial adjustment period, the loss trends smoothly downwards, with no sign of instability or plateauing. Compared to other latent masked image modeling methods such as I-JEPA or iBOT, this trend is reassuring, and might indicate good potential for further scaling.

C Blockwise masking strategy

The so-called "block masking" strategy used in many masked image modeling methods is by no means standardizes and can actually refer to several different implementations. The most common block masking implementation was proposed in BeiT (Bao et al., 2021), then reused in iBOT (Zhou et al., 2022) and MAE (He et al., 2022). It involves sampling many rectangular regions and doing multiple attempts to mask out approximately the right number of patches. Another implementation was proposed in I-JEPA, adding multiple constraints on the masks, to obtain a similar multi-block mask. Additionally, some methods postprocess the proposed mask to obtain a constant number of masked patches, in order to keep the same sequence length in all batch elements.

In CAPI, we propose a simpler heuristic: we sample a single rectangular mask, and truncate out the excess patches at the lower right end. Conversely, our implementation of inverse block masking is to sample a block mask, then simply invert it.

D Self-distillation interpretation

It was observed in DINO (Caron et al., 2021) that the downstream scores of the EMA model were consistently higher than the ones of the online model during training. This led to the interpretation of DINO as a self-distillation method, where the EMA model, the "teacher" distilled its slightly better representations into the online model, the "student". We observe that this interpretation still seems to hold in CAPI, albeit to a lesser extent, as evidenced by the comparison of teacher and student performance in Figure 10.

E Modified Sinkhorn-Knopp

We provide the pseudo-code for the standard Sinkhorn-Knopp and for our modified version in Figure 11. Both the original code and the proposed change are very simple. The actual code additionally contains an initial additive shift to prevent numerical instabilities in the exponential, as well as a collective `all_reduce` for distributed training.

F Detailed evaluation protocol

In all cases, our evaluations are performed with a frozen model, and use only the patch tokens outputted by the vision transformer. The input images are always at resolution 224×224.

F.1 Classification

The backbone model is kept frozen, and we extract only the patch tokens from its output. On top of these features, we train an attentive pooling classifier, consisting of a learned query, two learned k anv v projections, and a final projection to the number of classes. The attention is multi-head, with the head dimension being fixed at 64 and the number of heads being $\frac{d_{model}}{64}$. This head is optimized with a cross-entropy loss and the

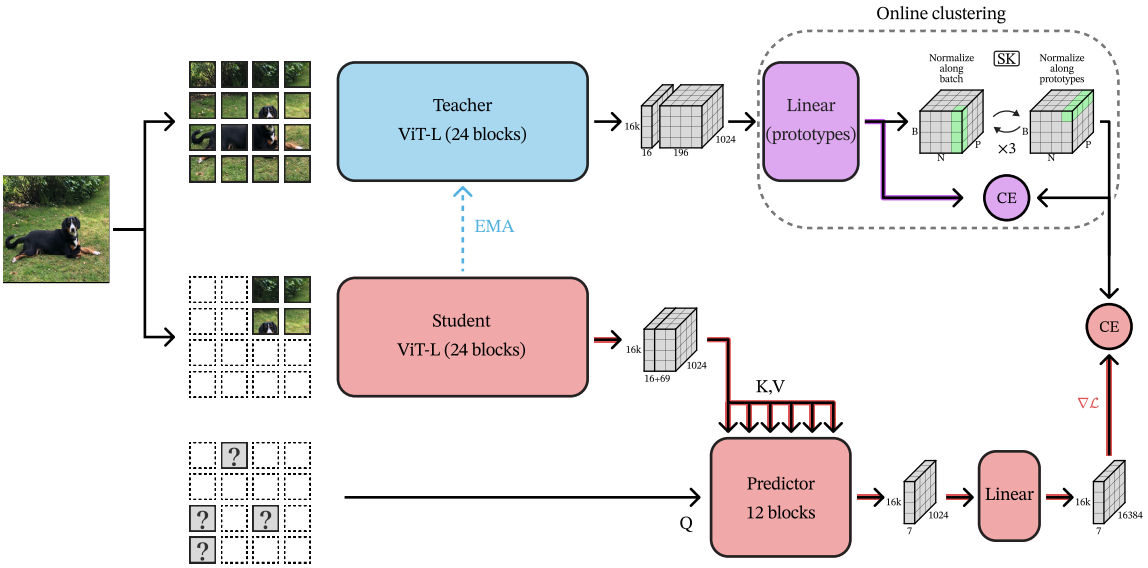


Figure 8: Detailed overview of our method with reference tensor sizes for a CAPI ViT-L/14 model. We denote in red the parts that are trained by the main loss, in purple the parts that are trained with the clustering loss, and in blue the parts that are updated by the EMA.

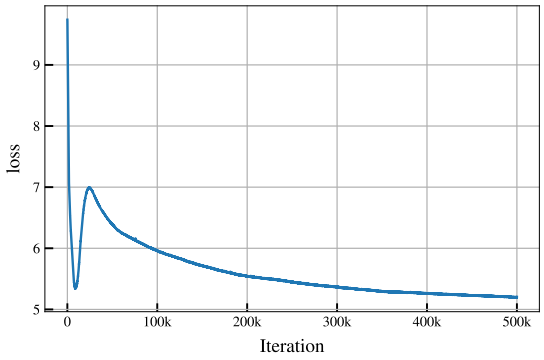


Figure 9: The loss curve of our CAPI ViT-L during training.

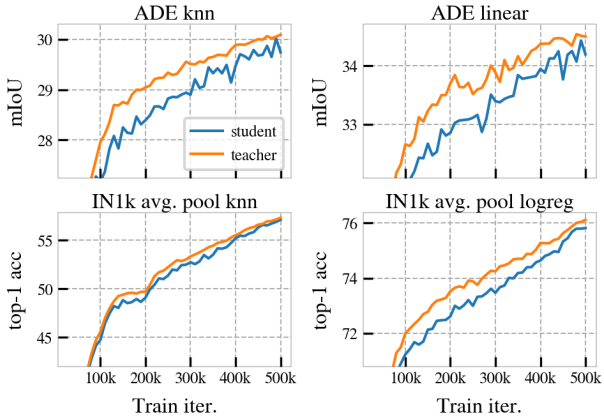


Figure 10: Comparative downstream scores of the teacher model and the student model throughout training.

```
3  M: Tensor  # b n d
4  M = M.exp()
5  for _ in range(3):
6-     M /= M.sum(dim=(0, 1))
7      M /= M.sum(dim=2)
```

(a) Standard SK

```
3  M: Tensor  # b n d
4  M = M.exp()
5  for _ in range(3):
6+     M /= M.sum(dim=0)
7      M /= M.sum(dim=2)
```

(b) Modified algorithm

Figure 11: PyTorch pseudo-code for the proposed modified Sinkhorn-Knopp algorithm. We normalize by the sum of the tokens for every given position, instead of normalizing across all positions.

Hyperparameter	Value
Batch size	16384
Optimizer	AdamW
Learning rate	1e-3
Teacher momentum	$1 - lr$
Clustering lr	$\frac{1}{2}lr$
lr schedule	linear warmup + trunc. cosine
Warmup length	10%
cosine truncation	20%
Weight decay	0.1
AdamW β	(0.9, 0.95)
Number of prototypes	16384
Student temperature	0.12
Teacher temperature	0.06
Num SK iter	3
Stochastic depth	0.2
Weight init	xavier_uniform
Norm layer	RMSnorm
Norm ε	1e-5
Patch embed lr	$0.2 \cdot$ lr
Norm layer wd	$0.1 \cdot$ wd
Image size	224
Augmentations	RRCrop, HFlip
Training dtype	bf16
Parallelism	FSDP
Pred. / im	7
Layerscale	No
Biases	No
Rope frequencies	logspace(7e-4, 7), axial
Masking type	inverse block`+roll`
Masking ratio	65%

Table 6: CAPI pretraining recipe

		knn			logreg		
model	standardization	ADE	Cityscapes	VOC2012	ADE	Cityscapes	VOC2012
CAPI	False	32.5	39.2	64.9	37.9	44.7	73.2
CAPI	True	33.0	39.2	65.2	37.7	44.3	73.3
aim 600M	False	14.3	27.8	38.5	7.1	28.3	61.3
aim 600M	True	nan	32.1	60.2	31.5	38.2	67.0
dinov2 vitg14+reg	False	33.8	42.0	63.1	38.9	46.8	72.9
dinov2 vitg14+reg	True	34.0	42.0	63.0	39.0	46.8	72.8
ijepa vith14 in1k	False	20.1	25.9	57.4	25.2	33.5	63.0
ijepa vith14 in1k	True	20.8	26.4	56.7	25.7	34.5	63.6
ijepa vith14 in22k	False	17.9	22.9	56.2	24.1	32.8	63.6
ijepa vith14 in22k	True	18.9	23.2	55.0	26.4	34.2	64.2
mae vitl16	False	7.8	25.3	17.3	27.4	38.4	61.5
mae vitl16	True	21.5	32.8	53.7	27.4	38.5	61.5

Table 7: Comparison of segmentation results with and without standardization

AdamW optimizer (Kingma, 2014; Loshchilov, 2017) for 12500 iterations at batch size 1024 (10 ImageNet epochs). The learning rate is warmed up linearly for 1250 iterations then annealed with a cosine schedule. We grid the weight decay over (5e-4, 1e-3, 5e-2) and the peak learning rate over (1e-5, 2e-5, 5e-5, 1e-4, 2e-4, 5e-4, 1e-3, 2e-3, 5e-3, 1e-2), training one classifier head for each pair of hyperparameters (30 in total). We choose the optimal hyperparameters using the accuracy on a 10% held-out part of the training set, then finally report the accuracy of this classifier on the test set. The training dataset is lightly augmented using a `torchvision RandomResizedCrop` (maintainers & contributors, 2016) with default hyperparameters and a random horizontal flip. During evaluation, the images are resized to 256 then center cropped to 224×224 pixels.

F.2 Segmentation

We compute the features for the train and test set considered at resolution 224, and hold out 10% of the train set as a validation set. Using these frozen features, the segmentation problem is reduced to a simple classification problem, on which we can use simple k-NN and linear classifiers. The linear classifier is trained for logistic regression with L-BFGS (Byrd et al., 1995) regularized with L2 penalty, as implemented in the `cuml` library (Raschka et al., 2020). In both cases, a grid of hyperparameters is tested, and the ones performing best on the validation set are retained. For the k-NN classifier, we grid the number of neighbors over (1, 3, 10, 30), and the distance used over (L2, cosine). For the linear classifier, we grid the regularisation parameter `C`, testing 8 values along a log-space between 10^{-6} and 10^{5}.

F.3 Feature standardization

Some of the baselines suffer from poor conditioning of their features, which can cause very bad results when fitting a logistic regression over these features. To reduce this issue, in the segmentation evaluation we standardize features by substracting their mean and dividing them by their standard deviation. These statistics are computed using the features from the training set only. This significantly improves the scores of pixel reconstruction-based methods, while the other methods are mostly unaffected. We report a comparison of segmentation results with and without standardization in table 7. In the rest of the paper, all segmentation results are obtained with standardization.

F.4 Baselines

All baselines are vision transformers, allowing us to use the same evaluation protocol. We feed the 224×224 image to the model after imagenet normalization of the pixel values, and extract the patch tokens after the

Table 8: Summary of all models mentioned in the paper. We associate to each a unique uid, and detail the hyperparameters which are not constant across all runs.

uid	dataset	#iter	patch size	enc depth	pred depth	enc dim	pred dim	lr	mom.	clust. lr	masking	ratio	roll	teacher head	student head	loss	pos. enc.	SK	#reg.	crop scale
Meb2b	Loca	500k	14	24	12	1024	1024	1e-03	0.999	5e-04	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Mc2dd	Loca	500k	14	12	6	768	768	1e-03	0.999	5e-04	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
M8c4d	Loca	500k	14	12	6	384	384	2e-03	0.998	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Adcab	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Ae3f9	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
A0dd4	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	random	90%	False	clustering	Linear	CE	rope	modified	16	[60%,100%]
A9b4a	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	block	65%	False	clustering	Linear	CE	rope	modified	16	[60%,100%]
A7cc0	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	False	clustering	Linear	CE	rope	modified	16	[60%,100%]
A3bb3	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	identity	identity	Huber	rope	standard	16	[60%,100%]
A2fcb	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	EMA	MLP	CE	rope	modified	16	[60%,100%]
Aeb48	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	MLP	CE	rope	modified	16	[60%,100%]
A74f9	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[100%,100%]
Ae7b3	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[20%,100%]
A41b8	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	55%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Ac8bc	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	75%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Af989	IN22k	100k	16	24	5	1024	1536	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
A2da0	IN22k	100k	16	24	21	1024	768	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
A9ce8	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	0	[60%,100%]
A1177	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	learn.	modified	16	[60%,100%]
A72fb	IN22k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	standard	16	[60%,100%]
M5e2e	IN22k	500k	14	24	12	1024	1024	1e-03	0.999	5e-04	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
M2d34	IN1k	500k	14	24	12	1024	1024	1e-03	0.999	5e-04	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
M8319	P205	500k	14	24	12	1024	1024	1e-03	0.999	5e-04	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Abe05	IN22k	25k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Acd44	IN22k	50k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
A2ca8	IN22k	200k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Aab94	IN1k	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]
Aa428	Loca	100k	16	24	12	1024	1024	2e-03	0.996	1e-03	inv. block	65%	True	clustering	Linear	CE	rope	modified	16	[60%,100%]

last transformer block. For the specific case of AIM (El-Nouby et al., 2024), we follow the advice from the original paper and extract the patch tokens after before the end of the ViT, specifically after layer 18.

G Compute cost and environmental footprint

We measure the training of a CAPI ViT-L model to take 180h on 32 A100 GPUs, amounting to 5763 A100 hours. This consumed around 2651 kWh of electricity, which we estimate to amount to approximately 928 kgCO2eq. The entire project used 3.75M A100 hours, which we similarly estimate to have emitted 604 tCO2eq for the electricity consumption. Note that the carbon footprint estimations here are purely scope 2 estimations, *i.e.* limited to electricity consumption, and are further limited to the electricity consumption of the GPUs. A full carbon accounting should additionally include many other harder to estimate emissions, such as the electricity consumption of the other server components and the rest of the datacenter appliances, and scope 3 emissions from the component manufacturing, datacenter construction, and their respective end-of-life.

H List of models used

We provide in Table 8 the list of all models presented in this paper, along with a unique hash identifier and the relevant hyperparameters. Non-listed hyperparameters are detailed in Table 6. To disambiguate any possible unclarities in the presented results, Table 9 provides the mapping from tables and figures to model identifiers.

I Visualisations

In Figures Figure 12 and 13, we provide visualisations of the feature maps of CAPI compared to other state-of-the-art self-supervised vision models.

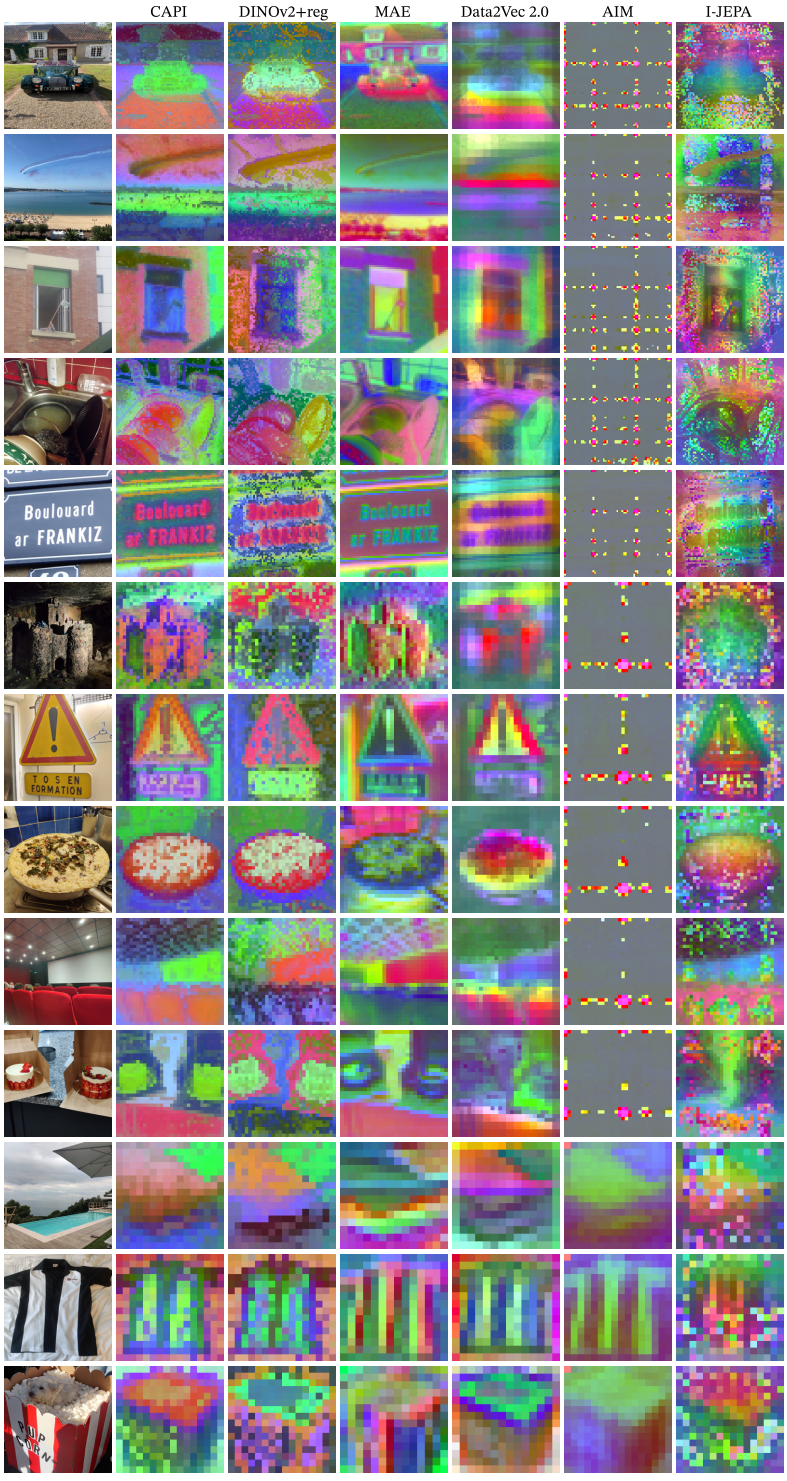


Figure 12: Visualization of the features produced by CAPI and other vision models at various resolutions: CAPI ViT-L/14, DINOv2+reg ViT-g/14 (Darcet et al., 2024), BEiT ViT-L/16 (Bao et al., 2021), AIM ViT-3B/14 (El-Nouby et al., 2024), MAE ViT-H/14 (El-Nouby et al., 2024), I-JEPA ViT-H/14 (Assran et al., 2023), and data2vec2 ViT-L/16 (Baevski et al., 2022). We apply a PCA decomposition to the dense outputs produced by each model for each image individually, and rescale the three first components to the RGB range for visualization.

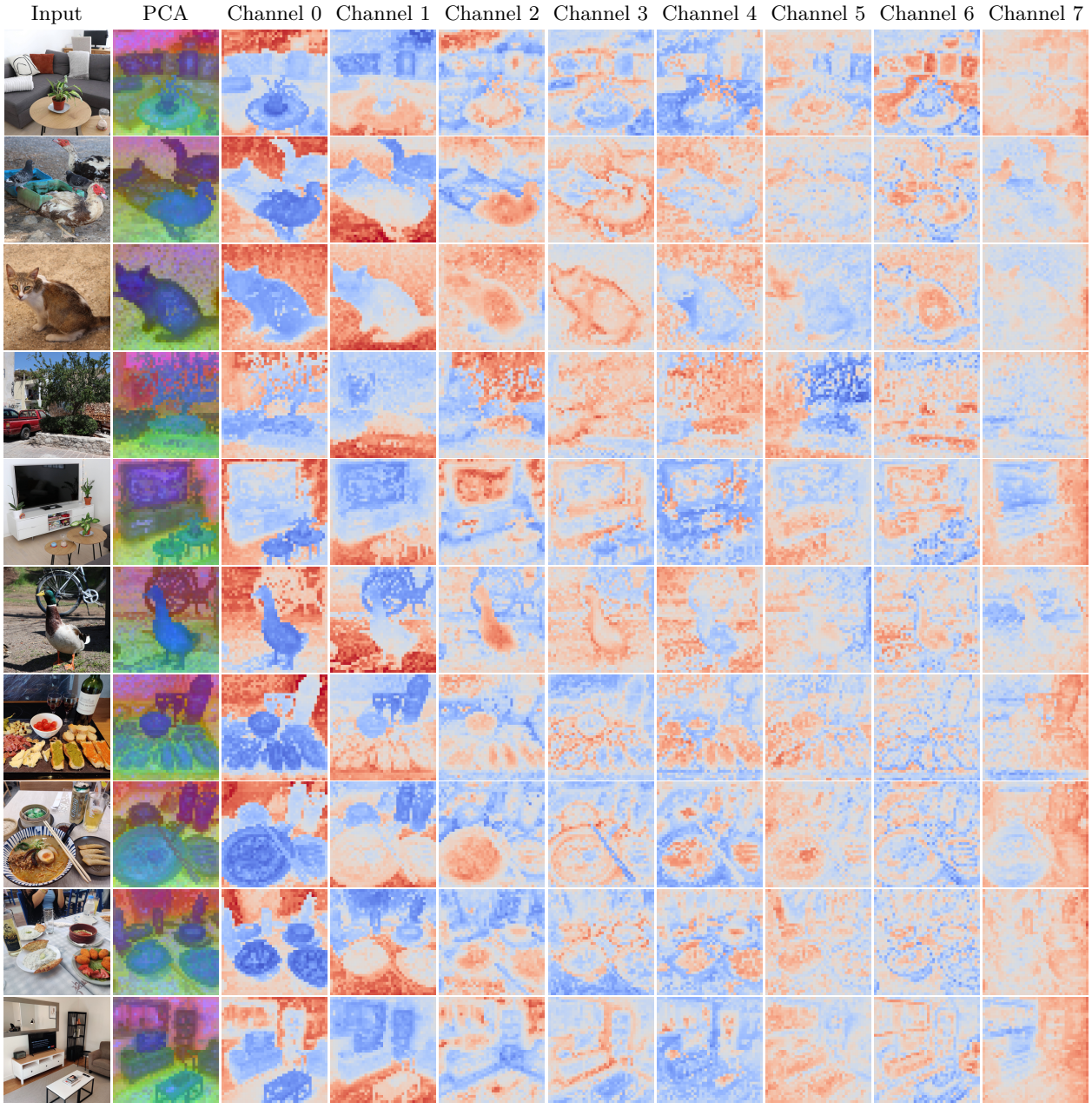


Figure 13: Visualization of the features produced by CAPI ViT-L/14 applied to images at 560 pixel resolution. We apply a PCA decomposition to the dense outputs produced by the model across all images. The first column shows the first 3 components as RGB. The next eight columns show the first eight channels individually using a `coolwarm` colormap from Matplotlib (Hunter, 2007).

Table 9: Reference of models used in different figures and tables.

Fig	Models used
fig. 1	Meb2b, Mc2dd, M8c4d
table 1a	Adcab, Ae3f9, Aa5a3, A7d26
table 1b	Adcab, Ae3f9, A0dd4, A9b4a, A7cc0
table 1c	Adcab, Ae3f9, A3bb3, A2fcb, Aeb48
table 1d	Adcab, Ae3f9, A74f9, Ae7b3
table 1e	Adcab, Ae3f9, A41b8, Ac8bc
table 1f	Adcab, Ae3f9, Af989, A2da0
table 1g	Adcab, Ae3f9, A9ce8
table 1h	Adcab, Ae3f9, A1177
table 1i	Adcab, Ae3f9, A72fb
table 2	Meb2b, M5e2e, M2d34, M8319
table 3	Meb2b, M5e2e, M2d34, M8319
table 4	Meb2b
fig. 7	Meb2b
table 5	Meb2b
fig. 6	Adcab, Ae3f9, Abe05, Acd44, A2ca8, Adcab, Ae3f9, Aab94, Aa428
fig. 10	Meb2b
fig. 13	Meb2b
fig. 12	Meb2b