An Empirical Study on Unifying JEPA and Language Supervision for Visual Representation Learning

Shixuan Liu*
University of Illinois Urbana-Champaign
Urbana, IL
shixuanl@illinois.edu

Daniel A Li*
Massachusetts Institute of Technology
Cambridge, MA
dali@mit.edu

Yiwei Lyu
University of Michigan
Ann Arbor, MI
yiweilyu@umich.edu

Akhil Kondepudi
University of Michigan
Ann Arbor, MI
akhilk@umich.edu

Honglak Lee
University of Michigan
LG AI Research
Ann Arbor, MI
honglak@eecs.umich.edu

Todd C Hollon
University of Michigan
Ann Arbor, MI
tocho@med.umich.edu

Abstract

Unified visual representations from language supervision and self-supervision offer the potential to advance general-purpose vision models. In this work, we present an empirical study on unifying joint-embedding predictive architecture (I-JEPA) with language supervision from CLIP for visual representation learning. I-JEPA is unique among self-supervised learning methods in that it is predictive rather than contrastive or generative, enabling faster convergence with less compute while still producing strong representations. Existing works have shown that joint training with language supervision and other visual self-supervision methods yield improved model performance, but combining language supervision with I-JEPA remains unexplored. We introduce CLIPred, a framework that jointly optimizes the two objectives, and systematically evaluate it across zero-shot classification, retrieval, and probing tasks. CLIPred outperforms CLIP-only, I-JEPA-only, and sequentially applying the two, and offers better zero-shot transfer than DINOv2+CLIP with lower training cost, though with trade-offs in probing performance. Our experiments further examine the effects of loss weighting, amount of data used by each objective, and batch size on our framework, We conduct further analysis on design choices of the architecture and the semantics of the patch embeddings generated by CLIPred. This work provides the first comprehensive assessment of combining I-JEPA and CLIP, highlighting both the benefits and limitations of the framework as well as recommendations on when and how to apply the framework.

1 Introduction

Learning effective visual representations is a cornerstone of modern computer vision. High-quality visual representation models are crucial backbones for a wide variety of important applications, such as classification, object detection, segmentation, and multimodal language generation by vision-language models (VLMs). Recent advances in visual representation learning have enabled models to

Preprint. Under review.

generalize across domains and tasks with minimal or no supervision [1, 2], and these advancements are achieved mostly through approaches from one of two paradigms: language supervision and self-supervised learning (SSL).

Language supervision aligns images with natural language descriptions using large-scale image-text pairs, usually via contrastive objectives, such as CLIP [1] or SigLIP [3]. Visual representation models trained through language supervision have demonstrated exceptional zero-shot capabilities and few-shot transferability [1]. Alternatively, self-supervised learning (SSL) does not require image-text pairs and, instead, uses visual pretext tasks defined on images only to learn visual features.

SSL pre-training on images generally falls into two categories according to [4]: invariance-based pre-training and generative pre-training. Invariance-based pre-training trains the encoder to generate similar embeddings for different views of the same image. Common invariance-based SSL methods include contrastive learning (e.g. SimCLR [5], MoCo [2]) and self-distillation (e.g. DINO [6], IBOT [7]). Generative pre-training trains the encoder to generate parts of an image given other parts of an image, e.g. masked autoencoders [8]. By capturing structures without relying on text supervision, these methods have closed the gap with, and in some cases surpassed, language-supervised models on a variety of downstream vision benchmarks [9].

Given the successes of both paradigms, a natural question arises: *does joint training with language supervision and SSL result in better visual representations?* Existing works have explored combining CLIP with invariance-based SSL like SimCLR [10] or DINOv2 [11], and achieved improved performance over single objectives. However, these approaches are often computationally expensive.

Recently, a new approach called joint-embedding predictive architecture [4] (JEPA) has emerged as a powerful SSL strategy that combines the strengths of both invariance-based and generative SSL. I-JEPA [4] applies the JEPA approach to images, where two random disjoint regions of the same image, the context and the target, are encoded by a context encoder and a target encoder. A predictor model is then required to predict the encoded target given the encoded context. The context encoder and the predictor are directly trained through the predictive objective, while the target encoder is updated as an exponential moving average of the context encoder, similar to the teacher-student framework from self-distillation. I-JEPA achieves strong performance with significantly less compute budget compared to other SSL approaches. However, combining CLIP with I-JEPA can be challenging due to stark differences in training objective and architecture, and thus remains unexplored.

In this paper, we conduct an empirical study systematically studying the potential strengths and weaknesses of combining I-JEPA and CLIP objectives in visual representation learning, as a step towards ***unifying representations*** **learned from language supervision and self-supervision**. We are especially interested in examining how the approach works with smaller training datasets like MSCOCO [12], because incorporating more training signals from multiple objectives is particularly important when training data is limited. Our main contribution is as follows:

- We design a novel visual representation learning framework, CLIPred, that allows jointly training with CLIP and I-JEPA objectives.
- We conduct comprehensive experiments on CLIPred, evaluating zero-shot and linear probing performances on several key benchmarks.
- We further conduct detailed analysis on evaluation and architecture design choices, required training resources, as well as the properties of patch-level embeddings from CLIPred.
- Through this empirical study, we are able to summarize the pros and cons for combining I-JEPA with CLIP through CLIPred, which can provide valuable insight for future researchers and developers in selecting visual representation learning methods. We also provide some recommendations on how to train with CLIPred based on our experiment results.

2 Related Work

Contrastive language-image pre-training (CLIP): Contrastive language-image pre-training aligns an image encoder with a text encoder using large-scale web image-text pairs. CLIP [1] popularized this dual-encoder setup and demonstrated strong zero-shot transfer across various recognition tasks by treating text prompts as classifiers after pre-training. Variants have focused on data scale/quality and objective design. On the data-oriented side, researchers explored using hard-negatives in each batch to improve model compositionality [13, 14, 15]. On the objective side, ALIGN [16] and SigLIP

[3] replace the softmax InfoNCE loss from CLIP with a pairwise sigmoid loss that is more tolerant of batch size. Some more recent recipes, such as SigLIP 2 [17], explicitly incorporate auxiliary objectives (captioning and self-supervised losses) and online data curation to further enhance transferability and robustness, indicating a trend toward multi-objective vision-language pre-training.

Visual self-supervised learning (SSL): Common SSL methods for images include contrastive methods, self-distillation, and generative/masked modeling. Contrastive methods (e.g., SimCLR [5], MoCo [2]) learn invariances by pulling together augmented views of the same image and pushing apart others. Design choices such as strong data augmentation, projection heads, large batches, queues, and momentum encoders are key to stability and performance. Generative/masked modeling methods reconstruct masked content using Masked Autoencoders (MAE) [8], showing that heavy masking and lightweight decoders yield scalable pre-training that transfers well after fine-tuning.

Self-distillation avoids explicit negatives by predicting targets produced by a slowly updated teacher model. BYOL [18] introduced online/target networks updated by an exponential moving average (EMA). SimSiam [19] demonstrated that a stop-gradient mechanism can prevent collapse even without a momentum teacher. DINO [6] applied self-distillation to ViTs with centering/sharpening and multi-crop, revealing strong emergent semantics. DINOv2 [20] scaled this recipe on curated data to produce robust all-purpose features competitive with language-supervised models on many downstream tasks.

I-JEPA [4] predicts target patch embeddings of masked regions from a context block, with targets computed by an EMA target encoder and the loss applied purely in representation space. This removes reliance on hand-crafted view augmentations while retaining semantic prediction targets, yielding strong linear-probe/transfer and much faster convergence (thus less training compute) than pixel-space reconstruction.

Combining CLIP and SSL: Several works explore ways to fuse language supervision with image-only SSL signals. While some works attempted applying the objectives sequentially (e.g. LiT [21] and DeCLIP [22]), most attempts were made through jointly training with all the objectives: SLIP [10] (CLIP [1] + SimCLR [5]) adds an SSL loss on the image encoder alongside the contrastive image-text loss and shows gains in low/medium-scale regimes (e.g., YFCC15M [23]), while also noting increased training cost and that benefits do not obviously scale on very large uncurated corpora. Follow-ups include iCLIP [24], which includes multi-task classification and CLIP heads, and MaskCLIP [25], which injects masked self-distillation into CLIP [1] to improve local/patch-level features without sacrificing zero-shot ability. Closer to our focus, DINOv2 Meets Text [11] builds directly on strong image-only features (DINOv2 [20]) and unifies image-level and patch-level vision-language alignment objectives, showing that coupling SSL backbones with text alignment can benefit both global and dense tasks. This line demonstrates a complementary path to CLIP-only recipes by starting from curated SSL features and adding language alignment losses.

Summary and positioning: Prior works show (i) CLIP-style pre-training offers excellent zero-/few-shot transfer via language supervision, (ii) self-distillation and masked prediction deliver strong, efficient image-only representations, and (iii) combining these signals via SLIP-style joint losses, LiT-style staging, or SSL-backbone-plus-text alignment, as in DINOv2 Meets Text [11], can improve robustness and data efficiency but may complicate optimization and add compute. Our work differs by jointly optimizing CLIP [1] and I-JEPA [4], a **predictive** SSL objective distinct from SimCLR [5] and DINOv2 [20], and analyzing patch-level embeddings to clarify complementarities and conflicts between predictive SSL and language supervision.

3 Method

We propose a novel training framework, CLIPred, that integrates I-JEPA objective with CLIP objective (contrastive learning with language supervision). This framework leverages information from both visual and textual modalities, and contains modules from both objectives. CLIPred builds upon the I-JEPA framework (Section 3.1) and extends it with additional modules supporting the CLIP objective (Section 3.2). The entire framework is illustrated in Figure 1.

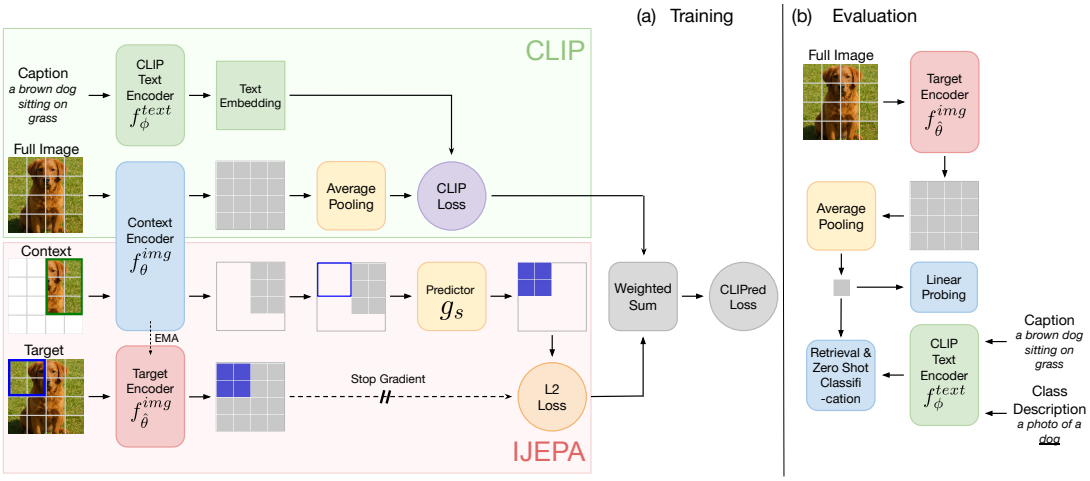

Figure 1: Illustration of CLIPred framework during training and evaluation. During training, the frameworks combines I-JEPA and CLIP objectives to joinly train the encoders; during inference, the target encoder is used to generate the image embeddings for zero-shot and linear probing.

3.1 I-JEPA base framework

CLIPred is built on top of the code base of I-JEPA [4], which models semantic relationships between visible and masked regions without relying on invariance to augmentations. During training, each training image x is split into a context region $x_s \in \mathbb{R}^{H_s \times W_s \times C}$ and k target regions $\{x_t^{(i)}\}_{i=1}^k$ where each $x_t^{(i)} \in \mathbb{R}^{H_t^{(i)} \times W_t^{(i)} \times C}$, with $x_s \cap x_t^{(i)} = \emptyset$, i.e. there is no overlap between the context and target regions. The context region is encoded by the context encoder f_θ^{img}, while the target region is encoded by the momentum-updated target encoder $f_{\hat{\theta}}^{\text{img}}$.

The context encoder produces a patch-level output $z_s = f_\theta^{\text{img}}(x_s) \in \mathbb{R}^{n \times d}$, where n is the number of patches in the context region. Then, for each target region, a mask $m_t^{(i)}$ is generated and is passed through a gradient-updated predictor network $g_s(\cdot)$ together with z_s, producing a predicted embedding $\hat{z}_t^{(i)} = g_s(z_s, m_t^{(i)}) \in \mathbb{R}^{n^{(i)} \times d}$, where $n^{(i)}$ is the number of patches in the target region $x_t^{(i)}$. Meanwhile, the target embeddings for all patches in the entire image is generated by $z_t = f_{\hat{\theta}}^{\text{img}}(x)$, and then the target embedding for each target region $z_t^{(i)} \in \mathbb{R}^{n^{(i)} \times d}$ is obtained by taking its corresponding $n^{(i)}$ patch embeddings from z_t.

The I-JEPA objective is to minimize the discrepancy between the predicted and actual target embeddings for each target region via an L2 loss, $\mathcal{L}_{\text{IJEPA}} = \frac{1}{m} \sum_{i=1}^{m} \left\| \hat{z}_t^{(i)} - z_t^{(i)} \right\|_2^2$. This loss encourages the context encoder to learn high-level semantics by predicting meaningful content from the context without access to low-level pixel information of the target. Note that $\mathcal{L}_{\text{IJEPA}}$ is only used to update the context encoder f_θ^{img} and the predictor g_s, not the target encoder $f_{\hat{\theta}}^{\text{img}}$. Instead, the target encoder $f_{\hat{\theta}}^{\text{img}}$ is updated via an exponential moving average (EMA) of the context encoder weights $\hat{\theta} \leftarrow \alpha\hat{\theta} + (1-\alpha)\theta$ where α is the smoothing factor. All 3 modules (the context encoder f_θ^{img}, the target encoder $f_{\hat{\theta}}^{\text{img}}$, and the predictor g_s) are implemented with ViT architecture without a CLS token.

3.2 Integration of CLIP

The goal of CLIP objective [1] is to learn a shared embedding space between images and text. Typical CLIP frameworks consists of a text encoder and an image encoder. To integrate CLIP objective into the I-JEPA framework, we have to use the context encoder f_θ^{img} as CLIP image encoder because the target encoder $f_{\hat{\theta}}^{\text{img}}$ is an EMA of the context encoder and thus should not be updated through CLIP gradient. We include an additional text encoder module, $f_\phi^{\text{text}}(t_i)$, to encode the text. During

training, let $(x_i, t_i)_{i=1}^N$ be a batch of matched image–text pairs. The image encoder produces an image embedding for x_i using the entire image as input (not just the context), and the text encoder produces a text embedding for t_i. Since f_θ^{img} does not have a CLS token, average pooling is applied to all patch embeddings to obtain the image embedding, following the original I-JEPA paper [4]. In practice, we apply learnable linear projection layers π_{img} and π_{text} on top of the encoders to ensure the image and text embeddings have the same dimension and are comparable. We denote $u_i = \pi_{\text{img}}(AvgPool(f_\theta^{\text{img}}(x_i)))$, $v_i = \pi_{\text{text}}(f_\phi^{\text{text}}(t_i))$ as the projected image and text feature vectors. We further normalize these projections to obtain unit-length embeddings $z_i^{\text{img}} = \frac{u_i}{|u_i|}, z_i^{\text{text}} = \frac{v_i}{|v_i|}$ so that similarity can be measured directly by dot product. For any image i and text j in the batch, let $s_{i,j} = z_i^{\text{img}} \cdot z_j^{\text{text}}$ be the cosine similarity between the image embedding of pair i and the text embedding of pair j. The CLIP objective loss is defined as

$$\mathcal{L}_{\text{img}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s_{i,i}/\tau)}{\sum_{j=1}^{N}\exp(s_{i,j}/\tau)}, \quad \mathcal{L}_{\text{text}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(s_{i,i}/\tau)}{\sum_{j=1}^{N}\exp(s_{j,i}/\tau)},$$
$$\mathcal{L}_{CLIP} = \frac{1}{2}(\mathcal{L}_{\text{img}} + \mathcal{L}_{\text{text}})$$

where $\tau > 0$ is a temperature hyperparameter. By minimizing $\mathcal{L}_{CLIP}$, the model learns a joint embedding space where corresponding images and texts are aligned and non-matching pairs are repelled.

CLIPred framework trains with both objectives jointly. The overall training loss is a weighted sum of the two components, $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{I-JEPA}} + \lambda\mathcal{L}_{\text{CLIP}}$, where λ is the weighting factor. All modules except the target encoder is updated with gradient from $\mathcal{L}_{\text{total}}$, while the target encoder is updated through EMA from the context encoder.

4 Experiments

4.1 Research questions

The first and foremost research question we want to address is **RQ1: Does jointly training with IJEPA and CLIP objective improve visual representation quality?** We systematically evaluate and compare the performance of CLIPred with single-objective and multi-objective baselines. In the following research questions, we aim to gain a deeper understanding of the factors that influence the training of CLIPred, including the proportion of data used for each objective (**RQ2**), the weighting of objectives (**RQ3**), and the batch size (**RQ4**).

4.2 Implementation and experiment setup details

Implementation and training details: We implement CLIPred using standard PyTorch with the Distributed Data Parallel (DDP) protocol. The image encoder is a standard ViT-base without the CLS token, while the text encoder is based on the Hugging Face language transformer[1]. The majority of experiments is conducted using MSCOCO [12] as training data, which contains approximately 118k training image and caption pairs. We randomly sample one out of the five available captions in MSCOCO for each training image. The hyperparameters are available in Table 4 in Appendix.

Baselines: We compare CLIPred to both single-objective approaches (I-JEPA only, CLIP only), sequential objectives (I-JEPA $\rightarrow$ CLIP) as well as combining CLIP with other SSL methods, such as DINOv2. The I-JEPA only baseline strictly follows the original I-JEPA paper [4], and for the CLIP-only baseline, we use exactly the same ViT model as the image encoder, and the average patch tokens as the image embedding. For sequential objectives, we take the I-JEPA-only pre-trained model and fine-tune it with CLIP objective alone. Our implementation of DINOv2+CLIP baseline follows [11], using the concatenation of the CLS token and average patch tokens as the image

[1]We adopt the text encoder architecture from `openai/clip-vit-base-patch32` with randomly initialized weights

Zero-shot	**CIFAR-10**	**CIFAR-100**		**ImageNet-1k**			**ImageNet-V2**		
classification	Top-1	Top-1	Top-5	Top-1	Top-5	Top-50	Top-1	Top-5	Top-50
CLIP	28.57	1.80	5.99	2.33	6.75	23.91	2.22	6.27	22.45
DINOv2+CLIP	13.33	1.17	5.73	1.68	3.80	10.97	1.41	3.13	10.66
IJEPA → CLIP	19.49	1.54	6.63	1.50	3.32	10.79	1.47	3.47	10.45
CLIPred	**31.94**	**2.10**	**7.60**	**4.10**	**10.30**	**29.29**	**3.50**	**9.79**	**27.45**

Zero-shot	**MSCOCO Test**				**Flickr Test**			
retrieval	T2I@1	T2I@5	I2T@1	I2T@5	T2I@1	T2I@5	I2T@1	I2T@5
CLIP	7.59	22.92	9.38	37.08	0.77	2.80	0.97	3.23
DINOv2+CLIP	6.97	20.64	9.08	24.54	0.37	1.43	0.63	2.25
IJEPA → CLIP	6.02	19.11	6.56	21.90	0.36	1.32	0.50	1.86
CLIPred	**13.68**	**36.60**	**18.30**	**44.22**	**1.78**	**5.81**	**2.33**	**7.44**

k% shot probing	**ImageNet-1k**			**ImageNet-V2**		
% probing data	1%			1%		
	Top-1	Top-5	Top-50	Top-1	Top-5	Top-50
CLIP	11.86	27.12	61.51	8.89	21.68	54.50
IJEPA	7.45	18.84	50.05	5.54	14.08	42.99
DINOv2+CLIP	**20.38**	**40.75**	72.91	**18.37**	**36.09**	**67.27**
IJEPA → CLIP	17.24	36.88	71.42	13.17	29.94	64.30
CLIPred	18.28	38.31	**73.63**	14.04	31.37	66.59

k% shot probing	**CIFAR-10**			**CIFAR-100**					
% probing data	1%	10%	100%	1%		10%		100%	
	Top-1	Top-1	Top-1	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
CLIP	53.56	68.83	75.73	15.91	38.05	38.41	69.66	53.31	81.78
IJEPA	48.51	65.81	73.93	12.33	32.12	33.35	63.32	50.57	80.21
DINOv2+CLIP	**64.37**	**75.91**	**83.16**	**23.73**	**49.82**	**47.91**	**77.46**	**61.32**	**87.58**
IJEPA → CLIP	55.77	70.91	74.61	15.40	39.62	34.12	64.91	51.28	79.87
CLIPred	54.43	67.56	77.87	16.76	38.82	36.89	67.00	53.90	82.51

Table 1: Zero-shot and Linear probing results, comparing IJEPA+CLIP (i.e. CLIPred) to baselines.

embedding. All baselines are also trained from random initialization and trained on the exact same dataset. The hyperparameters of all baselines are in Table 4 in Appendix.

Evaluation details: We evaluate CLIPred and baselines on zero-shot classification, zero-shot retrieval, and linear probing. Zero-shot classification and linear probing are evaluated on Imagenet-1k [26], Imagenet-V2 [27], CIFAR-10 [28] and CIFAR-100 [28] datasets, while zero-shot retrieval is evaluated on MSCOCO [12] test split and Flickr30k [29] test split. For zero-shot classification[2] and zero-shot retrieval, we obtain image embeddings from the linear-projected average-pooled target encoder output ($u_i = \pi_{img} AvgPool(f^{\text{img}}_{\hat{\theta}}(x))$) and text embeddings through the CLIP text encoder, and calculate cosine similarity between the two. Since zero-shot tasks require textual embeddings, they are only applicable to methods trained with CLIP objectives. For few-shot linear probing, we take the average-pooled target embedding $AvgPool(f^{\text{img}}_{\hat{\theta}}(x))$, and employ the scikit-learn [30] protocol to fit a linear layer, using 1%, 10%, or 100% of the training data from the evaluated dataset. 1%-shot probing is conducted on ImageNet due to the scale of this dataset.

4.3 Results

RQ1: Does jointly training with IJEPA and CLIP objective through CLIPred improve visual representations? Answer: Yes. We compare the performance of CLIPred to baselines on all evaluation tasks in Table 1. We make the following observations: (1) CLIPred consistently outperforms CLIP-only, I-JEPA-only, and sequential objectives across all metrics in zero-shot classification and retrieval, as well as in the majority of probing metrics. This indicates that joint training I-JEPA and CLIP through CLIPred framework is preferred over apply either objective alone or applying them sequentially. (2) When compared to DINOv2+CLIP, CLIPred is significantly better at zero-shot tasks (while DINOv2+CLIP performed worse than CLIP alone), indicating that CLIPred preserves the zero-shot capabilities from CLIP training, while combining DINOv2 with CLIP may not be able to do the same. (3) In linear probing, CLIPred falls short from DINOv2+CLIP's performance (especially in CIFAR tasks), thus showing a potential weakness in CLIPred in linear probing performance and generalizability to non-224x224 images when compared to DINOv2+CLIP.

[2]For zero-shot classification, we use the fixed template "A photo of a {}" across all datasets.

ImageNet-1k

Data for CLIP \ Data for I-JEPA	0%	10%	25%	100%
0%	0.10	3.06	4.64	7.45
10%	2.22	6.74	8.93	9.20
25%	3.04	8.71	11.22	10.39
100%	11.86	12.51	15.89	18.33

ImageNet-V2

Data for CLIP \ Data for I-JEPA	0%	10%	25%	100%
0%	0.10	2.12	3.74	5.52
10%	0.28	4.92	6.54	6.65
25%	2.38	6.55	7.94	7.59
100%	8.89	9.08	11.84	14.09

CIFAR10

Data for CLIP \ Data for I-JEPA	0%	10%	25%	100%
0%	10.00	52.60	63.61	73.93
10%	43.97	65.15	69.89	73.16
25%	51.61	68.51	72.85	73.60
100%	75.73	73.81	78.20	77.87

CIFAR100

Data for CLIP \ Data for I-JEPA	0%	10%	25%	100%
0%	1.00	25.96	36.93	50.57
10%	20.51	40.54	45.54	50.28
25%	26.51	44.02	49.75	51.33
100%	53.31	49.54	56.49	53.90

Table 2: Performance under varying levels of text and image supervision. Each row indicates the strength (%data used) of text supervision and each column shows the strength of image supervision. Each cell shows the top-1 linear probing accuracies on CIFAR10, CIFAR100, and ImageNet datasets respectively (100% training data for CIFAR and 1% for ImageNet).

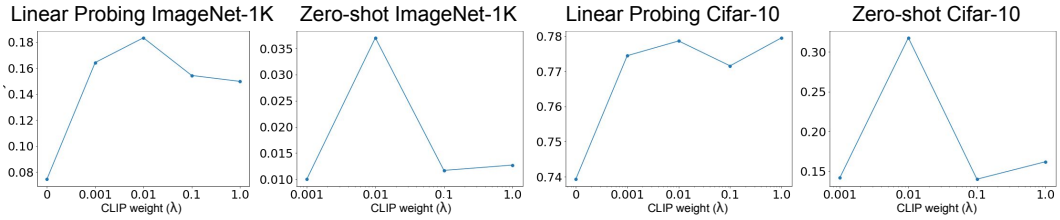

Figure 2: CLIPred performance with different relative weighting of CLIP objective (compared to I-JEPA with 1.0 weight). The optimal λ is 0.01 when CLIPred is applied to MSCOCO.

RQ2: Does CLIPred scale with data used for each objective? Answer: Yes. To understand how CLIPred scales with data being used for each objective, we perform a study where we only allow a certain percentage of data (from MSCOCO) to be used for each objective. The results are shown in Table 2, and we found that CLIPred almost always do better when we feed more data to any of the two objectives. This indicates that, when training an image encoder with CLIPred, we should always use as much data as possible for both objectives, even if the amount of data fed to the two objectives is imbalanced. For example, if we want to apply CLIPred to a certain domain where we have a lot of images but only a small subset of them have captions, we should still apply the I-JEPA objective to all images and the subset with caption to CLIP objective, and we will likely obtain improved performance over only using the subset with captions for both modalities.

RQ3: How do relative weights of two objectives affect CLIPred training? Answer: They matter, where 1.0I-JEPA+0.01CLIP works best on MSCOCO. Since the nature of I-JEPA objective and CLIP objectives are fundamentally different, how would different weighting of the two losses affect encoder training? To gain a better understanding of this, we compared performance of CLIPred with different relative weighting of the two objectives (i.e. different λ), and the results are shown in Figure 2. We found that the performance of CLIPred is actually quite sensitive to the relative weighting of the two objectives, with optimal λ being 0.01 when applying CLIPred to MSCOCO. This suggests that tuning the objective weight hyperparameter λ is essential when applying CLIPred.

RQ4: How does batch size affect CLIPred training? Answer: The bigger, the better. We compare performance of CLIPred trained with different batch sizes (100,200,400) in Figure 3. We found that larger batch sizes can improve CLIPred performance over the majority of metrics.

5 Additional Analysis

In this section, we conduct additional experiments to analyze the evaluation protocol, patch representation aggregation strategy, training resources, and visualizations of CLIPred.

5.1 Evaluation protocol analysis

In CLIPred, both the context encoder and the target encoder can be used to generate representation of an image. Therefore, we compare performance between representations from each encode for

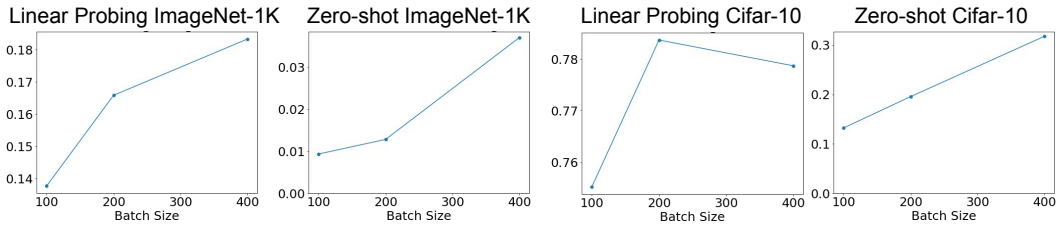

Figure 3: CLIPred performance with different batch sizes.

Zero-shot	**CIFAR-10**	**CIFAR-100**		**ImageNet-1k**			**ImageNet-V2**		
classification	Top-1	Top-1	Top-5	Top-1	Top-5	Top-50	Top-1	Top-5	Top-50
CLIPred	**31.94**	**2.10**	**7.60**	**4.10**	**10.30**	**29.29**	**3.50**	**9.79**	**27.45**
Eval with Context Encoder	31.79	1.93	6.99	3.70	9.47	28.07	3.10	8.92	26.09
CLIPred with AttnPool	18.12	1.20	5.51	1.21	2.81	9.62	1.10	2.76	9.10

Linear probing	**ImageNet-1k**			**ImageNet-V2**		
% probing data	1%			1%		
	Top-1	Top-5	Top-50	Top-1	Top-5	Top-50
CLIPred	**18.33**	**38.32**	**73.67**	**14.04**	**31.37**	**66.59**
Eval with Context Encoder	16.97	36.12	71.61	13.21	29.40	64.28
CLIPred with AttnPool	15.98	34.27	68.58	12.27	27.45	61.06

Linear probing	**CIFAR-10**			**CIFAR-100**					
% probing data	1%	10%	100%	1%		10%		100%	
	Top-1	Top-1	Top-1	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
CLIPred	**54.43**	**67.56**	**77.87**	**16.76**	**38.82**	**36.89**	**67.00**	**53.90**	**82.51**
Eval with Context Encoder	53.46	66.52	75.99	15.89	37.70	35.11	65.49	52.26	80.45
CLIPred with AttnPool	50.92	64.53	75.97	15.14	36.56	33.95	63.74	52.04	80.70

Table 3: Zero-shot and Linear probing results on CIFAR-10, CIFAR-100, and ImageNet. Using the target encoder as representation model consistently outperforms the context encoder when evaluating CLIPred (section 5.1), and replacing average pooling with attention pooling did not yield better performance (section 5.2).

zero shot classification and probing, and the results are in Table 3. We found that the target encoder consistently outperforms the context encoder across all tasks. The target encoder being better at probing was expected, since the original I-JEPA paper [4] reported that the target encoder's representations do better on probing tasks; however, the target encoder doing better at zero-shot classification is really surprising, since the context encoder was the one directly being optimized for CLIP objective.

5.2 Patch embedding aggregation strategy

In the experiments above, we have followed the original I-JEPA paper [4] and used average pooling as the patch embedding aggregation strategy for both CLIP training objective and evaluation. However, since CLIPred now includes CLIP supervision, we would like to explore whether it is better to replace the average pooling layer with attention pooling. The results are shown in Table 3, comparing CLIPred with "CLIPred with AttnPool". We found that replacing average pooling with attention pooling resulted in worse performance across all tasks. Therefore, we recommend using average pooling when applying CLIPred.

5.3 Training resources

All of our experiments were ran on 4 Nvidia A40 GPUs. We measure the training time of each method on MSCOCO until the best-performing epoch (i.e. the checkpoints used for evaluation) in Figure 4. We found that the time it takes to train CLIPred is roughly equal to the sum of the times taken to train CLIP only and IJEPA only, and that training CLIPred is 3X faster than training DINOv2+CLIP.

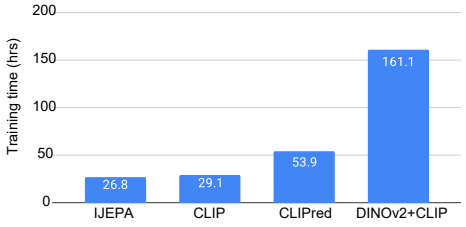

Figure 4: Training time of each method on MSCOCO on 4 Nvidia A40 GPUs.

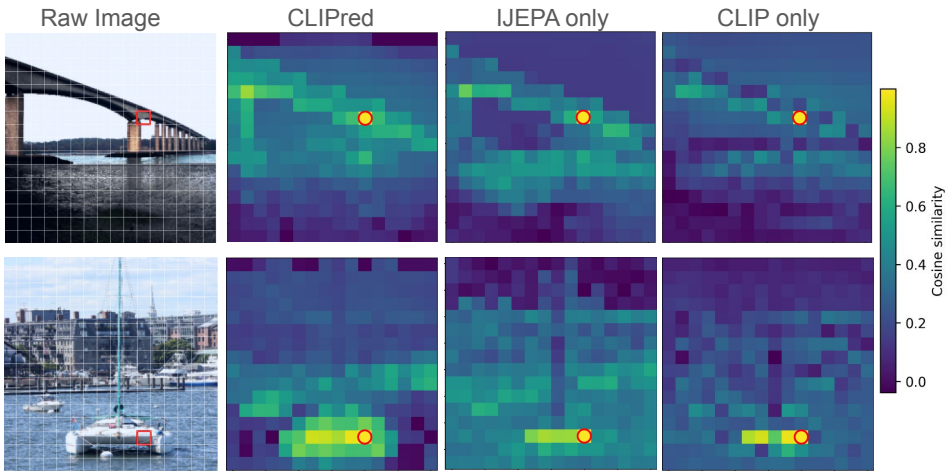

Figure 5: Patch embedding cosine similarity analysis. Red square/circle indicates the selected patch that all other patches are being compared to. We can see that the patches of high similarity to the selected patch from CLIPred best resembles the same type of object (bridge/boat) compared to IJEPA-only and CLIP-only.

5.4 Patch embedding cosine similarity analysis

Since I-JEPA generates patch-level embeddings, We can further analyze the properties of the patch-level embeddings by visualizing cosine similarities between embeddings of one particular patch and all other patches, following [31]. As shown in Figure 5, CLIPred's patch embeddings with high similarity to the highlighted patch best resemble the objects of interest: on the top image, the selected patch is a part of the bridge, and the patches with high cosine similarity to the selected patch resembles the entire bridge, while the IJEPA-only visualization has more highlight on the body of water and the CLIP-only visualization did not cover the entire bridge; on the bottom image, the selected patch is a part of a boat, and CLIPred visualization highlights all 3 boats present in the image, while CLIP-only only highlighted 1 boat and IJEPA-only highlighted the buildings/water as well. Therefore, we found that CLIPred's visualizations best highlight the object from the selected patches, and thus show that CLIPred learns more semantically meaningful patch embeddings compared to IJEPA-only and CLIP-only.

6 Conclusion

In this paper, we proposed a novel framework, CLIPred, that allows joint-training with I-JEPA and CLIP objectives, and conducted an empirical study to explore and analyze the potential strengths and weaknesses of combining the two objectives, as well as which factors matter the most when applying CLIPred. We summarize our findings as follows:

Why/when you should use CLIPred: We found that jointly training I-JEPA and CLIP objectives through CLIPred can significantly improve performance over many evaluation tasks (zero-shot classification & retrieval, linear probing) when compared to only applying one of the two objectives or sequentially applying the objectives. Moreover, CLIPred inherited the computation efficiency advantage of I-JEPA and trains significantly faster compared to DINOv2+CLIP. CLIPred also consistently have strong zero-shot performance, and it produces semantically meaningful patch-level embeddings. Therefore, CLIPred could be a good option when you want to learn a strong image representation model with both image-level and patch-level embeddings and have limited training data, limited computation resources, or want strong zero-shot performance.

Why/when you should not use CLIPred: Our experiment shows that, when compared to DINOv2+CLIP, CLIPred is much better in zero-shot performance but less well suited for linear probing. So if you have enough computating resources (DINOv2+CLIP needs 3X as much compute compared to CLIPred) and plan to fine-tune a classification head on the pre-trained model for some downstream task, you should consider using DINOv2+CLIP instead.

What needs attention when applying CLIPred: CLIPred performance is very sensitive to the relative weight between I-JEPA objective and CLIP objective (i.e. the λ hyperparameter). We found that 0.01 is best when applying CLIPred on MSCOCO, and we recommend carefully tuning this hyperparameter when applying CLIPred to other datasets. Bigger batch sizes help, and we recommend feeding as much data as possible/applicable to both objectives when training. We also recommend always using the target encoder for all downstream evaluations.

What we are still uncertain about CLIPred (limitations): This empirical study is limited to jointly applying SSL methods and CLIP on relatively small datasets like MSCOCO [12]. However, the performance gains from combining SSL with CLIP may not always scale to larger datasets, as observed by some previous works [32]. In the future, we plan to extensively explore how CLIPred scales with larger and more diverse training datasets.

References

[1] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.

[2] Kaiming He et al. "Momentum contrast for unsupervised visual representation learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9729–9738.

[3] Xiaohua Zhai et al. "Sigmoid loss for language image pre-training". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 11975–11986.

[4] Mahmoud Assran et al. "Self-supervised learning from images with a joint-embedding predictive architecture". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 15619–15629.

[5] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 1597–1607. URL: `https://proceedings.mlr.press/v119/chen20j.html`.

[6] Mathilde Caron et al. "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 9650–9660.

[7] Jinghao Zhou et al. "ibot: Image bert pre-training with online tokenizer". In: *arXiv preprint arXiv:2111.07832* (2021).

[8] Kaiming He et al. "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16000–16009.

[9] David Fan et al. "Scaling language-free visual representation learning". In: *arXiv preprint arXiv:2504.01017* (2025).

[10] Norman Mu et al. "SLIP: Self-supervision meets Language-Image Pre-training". In: *arXiv preprint arXiv:2112.12750* (2021).

[11] Cijo Jose et al. *DINOv2 Meets Text: A Unified Framework for Image- and Pixel-Level Vision-Language Alignment*. 2024. arXiv: `2412.16334 [cs.CV]`. URL: `https://arxiv.org/abs/2412.16334`.

[12] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: `1405.0312 [cs.CV]`. URL: `https://arxiv.org/abs/1405.0312`.

[13] Mert Yuksekgonul et al. "When and why Vision-Language Models behave like Bags-of-Words, and what to do about it?" In: *International Conference on Learning Representations*. 2023. URL: `https://openreview.net/forum?id=KRLUvxh8uaX`.

[14] Philipp J Rösch et al. "Enhancing conceptual understanding in multimodal contrastive learning through hard negative samples". In: *arXiv preprint arXiv:2403.02875* (2024).

[15] Shixuan Liu et al. "An empirical study of CLIP fine-tuning with similarity clusters". In: *NeurIPS 2024 Workshop on Fine-Tuning in Modern Machine Learning: Principles and Scalability*.

[16] Chao Jia et al. "Scaling up visual and vision-language representation learning with noisy text supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 4904–4916.

[17] Michael Tschannen et al. *SigLIP 2: Multilingual Vision-Language Encoders with Improved Semantic Understanding, Localization, and Dense Features*. arXiv preprint arXiv:2502.14786. 2025.

[18] Jean-Bastien Grill et al. "Bootstrap your own latent: A new approach to self-supervised learning". In: *Advances in Neural Information Processing Systems*. 2020.

[19] Xinlei Chen and Kaiming He. "Exploring simple siamese representation learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 15750–15758.

[20] Maxime Oquab et al. "Dinov2: Learning robust visual features without supervision". In: *arXiv preprint arXiv:2304.07193* (2023).

[21] Xiaohua Zhai et al. “LiT: Zero-Shot Transfer with Locked-Image Text Tuning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18123–18133.

[22] Yangguang Li et al. *Supervision Exists Everywhere: A Data Efficient Contrastive Language-Image Pre-training Paradigm*. arXiv preprint arXiv:2110.05208. 2021.

[23] Bart Thomee et al. “YFCC100M: The new data in multimedia research”. In: *Communications of the ACM* 59.2 (2016), pp. 64–73.

[24] Yixuan Wei et al. “iCLIP: Bridging Image Classification and Contrastive Language-Image Pre-Training for Visual Recognition”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 2776–2786.

[25] Xiaoyi Dong et al. “MaskCLIP: Masked Self-Distillation Advances Contrastive Language-Image Pre-training”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 10995–11005.

[26] Jia Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[27] Benjamin Recht et al. “Do ImageNet Classifiers Generalize to ImageNet?” In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. 2019, pp. 5389–5400.

[28] Alex Krizhevsky and Geoffrey Hinton. *Learning Multiple Layers of Features from Tiny Images*. Tech. rep. 0. Toronto, Ontario: University of Toronto, 2009.

[29] Bryan A. Plummer et al. “Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 2641–2649.

[30] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[31] Oriane Siméoni et al. “DINOv3”. In: *arXiv preprint arXiv:2508.10104* (2025).

[32] Floris Weers et al. “Masked autoencoding does not help natural language supervision at scale”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 23432–23444.

[33] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG]. URL: https://arxiv.org/abs/1711.05101.

A Model hyperparameters

The hyperparameters for CLIPred and baselines are listed in Table 4. The hyperparameters for sequential objectives (I-JEPA $\rightarrow$ CLIP) is the same as I-JEPA only and CLIP-only during each objective's training.

Table 4: Hyperparameters of CLIPred and baselines.

Hyperparameter	**CLIP**	**IJEPA**	**CLIPred**	**DINOv2+CLIP**
batch size (total)	400	400	400	400
epochs	300	1000	1000	1000
learning rate schedule	Cosine Decay	Cosine Decay	Cosine Decay	Cosine Decay
peak learning rate	0.001	0.001	0.001	2.0e-4
start learning rate	0.0002	0.0002	0.0002	0.0
final learning rate	1.0×10^{-6}	1.0×10^{-6}	1.0×10^{-6}	1.0×10^{-6}
warmup (in epochs)	10	40	40	80
weight decay	0.04	0.04	0.04	0.04
final weight decay	0.4	0.4	0.4	0.2
optimizer	AdamW [33]	AdamW	AdamW	AdamW
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	$\beta_1, \beta_2 = 0.9, 0.999$	$\beta_1, \beta_2 = 0.9, 0.999$	$\beta_1, \beta_2 = 0.9, 0.999$
eps	1.0×10^{-8}	1.0×10^{-8}	1.0×10^{-8}	1.0×10^{-8}
start EMA momentum	/	0.996	0.996	0.994
final EMA momentum	/	1.0	1.0	1.0
ema schedule	/	Linear	Linear	Cosine
start teacher temp	/	/	/	0.04
final teacher temp	/	/	/	0.07
λ (CLIP loss weight)	/	/	0.01	0.1