Open Vocabulary Semantic Segmentation with Patch Aligned Contrastive Learning

Jishnu Mukhoti¹  Tsung-Yu Lin²  Omid Poursaeed²  Rui Wang²  Ashish Shah²  Philip Torr¹  Ser-Nam Lim²
¹University of Oxford  ²Meta AI

Abstract

We introduce Patch Aligned Contrastive Learning (PACL), a modified compatibility function for CLIP’s contrastive loss, intending to train an alignment between the patch tokens of the vision encoder and the CLS token of the text encoder. With such an alignment, a model can identify regions of an image corresponding to a given text input, and therefore transfer seamlessly to the task of open vocabulary semantic segmentation without requiring any segmentation annotations during training. Using pre-trained CLIP encoders with PACL, we are able to set the state-of-the-art on the task of open vocabulary zero-shot segmentation on 4 different segmentation benchmarks: Pascal VOC, Pascal Context, COCO Stuff and ADE20K. Furthermore, we show that PACL is also applicable to image-level predictions and when used with a CLIP backbone, provides a general improvement in zero-shot classification accuracy compared to CLIP, across a suite of 12 image classification datasets.

1. Introduction

Understanding the semantic content in visual scenes has been one of the most important problems studied in computer vision at various levels of granularity. Work on this problem has led to significant improvements along several threads including image level predictions like image classification [13, 54, 58], object level predictions within an image like object detection [33, 51, 53, 59–61], as well as pixel level predictions in an image like semantic segmentation [10, 29, 51, 53]. Although in image classification we require only a single label per image for prediction, for scene understanding at a higher level of granularity like segmentation, training supervised models requires annotations at a pixel level. Such annotations require significant human effort and are often very expensive to obtain. This impedes training such supervised models on a large scale with millions of images.

![High level overview of our model](image)

Figure 1. High level overview of our model. We train an alignment between the patch level embeddings from the image encoder and the CLS embedding from the text encoder. This alignment can then be used to perform open-vocabulary semantic segmentation in a zero-shot manner.

One way to tackle this problem could be to train models in an unsupervised manner without requiring any segmentation annotations. The best methods [11, 19] in this category exploit the similarity between internal representations of self-supervised image encoders [5]. This similarity is then used to identify and cluster similar regions of the image as segmentations. These models however are significantly outperformed by their fully supervised counterparts on most segmentation benchmarks.

Recent improvements in multi-modal foundation models has led to the possibility of training on very large scale datasets scraped off the internet [41]. These datasets mostly contain pairs of images and their corresponding natural language text descriptions. Models like CLIP [41], ALIGN [23], Florence [59] and CoCa [58] trained on such large internet scale datasets have been shown to transfer very well to several downstream tasks. Furthermore, having been trained on natural language textual descriptions, these models are often expected to recognize a wide variety of real-world visual concepts which can be expressed in natural language, a setting better known as open vocabulary prediction.

The natural question then is whether these multi-modal
models can be used for pixel level predictions, i.e., semantic segmentation in the open vocabulary setting. Prior works on this topic [17, 28, 32, 56, 57] show that this is indeed possible. However, 3 of these works use either fully supervised segmentation annotations [32], class-agnostic segmentation masks [17] or a region proposal model trained using segmentation annotations [57], thereby being limited by the availability of expensive segmentation annotations/masks. To the best of our knowledge, only two models: ViL-Seg [32] and GroupViT [56] perform the task of open-vocabulary semantic segmentation while being trained solely on image-text data. Among these two, the better performer, GroupViT, defines a modified vision transformer (ViT) [15] architecture to naturally find semantic clusters within an image. Due to a different architecture, their model has to be trained end-to-end from scratch using image-text datasets and cannot leverage pre-trained vision encoders.

In this work, we tackle the problem of open-vocabulary semantic segmentation without using any segmentation annotations or masks, with a model purely trained on image-text data. We start with the observation in [19] that self-supervised ViT models like DINO [5], have similar patch representations for semantically similar regions of an image. We find this observation to be true for CLIP’s ViT based vision encoders as well. However, we also find that CLIP does not exhibit a patch level alignment between its vision and text encoders, primarily owing to the fact that its contrastive loss only aligns the CLS image and text tokens.

Inspired from previous work on contrastive learning for weakly supervised phrase grounding [18], we define a new compatibility function for contrastive loss to train an alignment between the patch tokens of the vision encoder and the CLS token of the text encoder. In particular, we take the cosine similarity between the text CLS token and the vision patch tokens and use these similarities as weights to compute a weighted sum over vision tokens. The final compatibility function is then simply the cosine similarity between the weighted sum of the vision patch tokens thus obtained and the CLS text token. We find that models trained on our Patch Aligned Contrastive Learning (PACL) loss indeed exhibit the desired patch level fine-grained alignment. Thus, at inference time, the compatibility function can be used to make image level predictions and the patch level alignment can be used for zero-shot transfer to semantic segmentation. A high level overview of our model is shown in Figure 1.

Note that unlike GroupViT, our PACL method is more flexible and general and can be used with any pre-trained ViT based encoders as well. We evaluate PACL with a pre-trained CLIP encoder on the task of zero-shot semantic segmentation using 4 different datasets: Pascal VOC [16], Pascal Context [36], COCO Stuff [4] and ADE20K [63]. On all 4 datasets, PACL consistently beats previous base-lines [17, 28, 32, 56], even the ones which use segmentation annotations or segmentation masks for training. In addition, we find that PACL trained on top of a CLIP backbone leads to a general improvement in zero-shot classification performance across a suite of 12 different image classification datasets.

Thus, in a nutshell, our contributions are as follows. Firstly, we propose Patch Aligned Contrastive Learning (PACL), a modified compatibility function for contrastive loss in order to train an alignment between the patch representations of a ViT based vision encoder and the CLS text representation of a text encoder. We show that this alignment can be used to find regions within an image corresponding to a given text input and hence, can be used for zero-shot transfer to open-vocabulary semantic segmentation. Secondly, we show that PACL with a pre-trained CLIP encoder obtains state-of-the-art scores on zero-shot semantic segmentation across 4 different segmentation benchmarks: Pascal VOC, Pascal Context, COCO Stuff and ADE20K. Finally, PACL with a CLIP backbone also shows a general improvement in performance on zero-shot classification tasks across 12 different image classification datasets.

2. Related Work

In this section, we discuss some of the relevant works motivating our method.

Supervised semantic segmentation: Given an image, the task of semantic segmentation [34] involves classifying every pixel in the image to one of a fixed set of classes. Naturally, supervised datasets for semantic segmentation like Pascal VOC [16], ADE20K [63] and Cityscapes [12] contain images with class annotations for every pixel. A significant amount of work [8, 43, 49, 62] has been done to leverage these datasets and generate strong models for semantic segmentation. However, since annotating images at a pixel level is laborious and expensive, these datasets remain limited to a relatively small number of classes.

Unsupervised semantic segmentation: Identifying that the requirement of dense annotations is the problem, some works [11, 19, 22, 35, 47, 52] have tried to leverage self-supervised learning techniques to train features which can be used for segmentation without requiring dense annotations. Notable among these works is STEGO [19] which uses the localized feature correspondences in self-supervised models like DINO [5] for the task of unsupervised segmentation. In our work, we study the existence of a similar feature correspondence in vision encoders of multi-modal models like CLIP [41] and use it to train a patch level alignment between image and text modalities. Note however, that it is still difficult for such unsupervised segmentation approaches to scale up to a large number of visual concepts.
Natural language supervision: Recently, the availability of datasets with millions of image-text pairs scraped from the internet has made it possible to train large-scale multi-modal fusion models on such datasets. Such models [23, 25, 41, 45, 58, 59] are able to transfer well to several downstream tasks including vision-language pre-training (VLP) [7] tasks like image-text retrieval [50] and visual question answering [1], as well as vision specific tasks like zero-shot image classification [23, 41, 58] and object detection [24, 61]. Given the large-scale training of such multi-modal fusion models, it is then natural to ask if these models can be leveraged to scale up the task of semantic segmentation and recognise a large number of visual concepts at a fine-grained level.

Natural language supervision for zero-shot segmentation: Some work has been done in this direction of using large-scale multi-modal models, like CLIP [41], for the task of semantic segmentation. For instance, LSeg [28] trains a segmentation model as its vision encoder and uses the frozen text encoder from CLIP to align pixel level embeddings with text. The resulting model is able to recognise conceptually similar labels which are not present within the training set. However, it trains the vision encoder in a fully supervised manner using segmentation annotations. OpenSeg [17] on the other hand is based on the ALIGN [23] model and trains using image-text data and class-agnostic segmentation annotations. ViL-Seg [32] trains using only image-text data with a vision based contrasting and a cross-modal contrasting objective along with an online clustering head to segment visual embeddings. Finally, GroupViT [56] proposes a modified ViT architecture which allows grouping semantically similar tokens into clusters useful for open vocabulary segmentation. To the best of our knowledge, ViL-Seg and Group-ViT are the only existing methods which solely use image-text data for training an open vocabulary semantic segmentation model. In our work, we propose a simple modification to the CLIP compatibility function for contrastive loss, which enables training an alignment between the patch tokens of a ViT based vision encoder and the CLS token of a text encoder. This alignment can then be seamlessly utilized for the task of semantic segmentation without using any segmentation annotations or class-agnostic segmentation masks during training.

### 3. Patch Level Alignment in CLIP

The contrastive training of CLIP ensures that the CLS tokens obtained from CLIP’s transformer based vision and text encoders are aligned for similar image-text pairs. However, such an alignment between image and text at a patch level does not necessarily exist. To empirically study this, we use a semantic segmentation dataset, Pascal VOC [16], and classify each patch in the dataset to one of a fixed set of classes. The patch level vision tokens are classified using the same zero-shot classification [41] method normally used on the CLS vision token. The classification accuracy, thus obtained, provides a measure of patch level alignment between the vision and text representations in the model, where a high classification accuracy indicates a high alignment and vice-versa.

More formally, let $\mathcal{D}_{seg} = \{(x, y)\}_{i=1}^N$ be the semantic segmentation dataset where $x \in \mathbb{R}^{H,W}$ and $y \in \mathbb{R}^{H,W}$. We represent CLIP’s vision and text encoders as $f_v : \mathbb{R}^{C,H,W} \rightarrow \mathbb{R}^{D_v}$ and $f_t : \mathbb{R}^l \rightarrow \mathbb{R}^{D_t}$ respectively. Similarly, let $e_v : \mathbb{R}^{D_v} \rightarrow \mathbb{R}^D$ and $e_t : \mathbb{R}^{D_t} \rightarrow \mathbb{R}^D$ be the linear embedders to project the vision and text encodings to the joint $D$ dimensional space. Normally, for zero-shot classification, we measure the cosine similarity between the vision and text embeddings: $s(x, c) = \frac{e_v(f_v(x)) \cdot e_t(f_t(y))}{\|e_v(f_v(x))\| \cdot \|e_t(f_t(y))\|}$ for each class name $c$ and compute the predictive probability as: $p(c|x) = \frac{e_v(f_v(x)) \cdot e_t(f_t(y))}{\sum_{c'} e_v(f_v(x)) \cdot e_t(f_t(y))}$. A simple modification to the vision encoder: $f_v : \mathbb{R}^{C,H,W} \rightarrow \mathbb{R}^{T,D_v}$, where $T$ is the number of tokens or patches, allows us to perform the same classification method on every patch.

In Table 1, second column (Pre-Alignment), we show the patch classification accuracy thus obtained for two CLIP models: ViT-B/16 and ViT-L/14. In Figure 2, first and

<table>
<thead>
<tr>
<th>CLIP Vision Encoder</th>
<th>Patch Classification Accuracy</th>
<th>Pre-Alignment</th>
<th>Post-Alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViT-B/16</td>
<td>52.49</td>
<td>96.51</td>
<td></td>
</tr>
<tr>
<td>ViT-L/14</td>
<td>27.91</td>
<td>95.33</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Accuracy for patch level classification on Pascal VOC. For a pre-trained CLIP model, the accuracy is low indicating low patch level alignment between image and text. On applying our PACL alignment method, the accuracy significantly increases for both CLIP encoders indicating higher image text patch level alignment.

![Image 313x582 to 372x627](Image 313x582 to 372x627)

Figure 2. Patch level alignment between the word “cat” and images of cats. In the first row, we show the original images, in the second row, we show the patch level alignment in CLIP ViT-B/16 and in the third row, we show the alignment for our method.

![Image 375x582 to 434x627](Image 375x582 to 434x627)

Figure 3. ROC curve indicating semantic coherence of CLIP and DINO vision encoders. CLIP encoders outperform DINO.
second rows, we show qualitative samples of alignment on CLIP ViT-B/16, for 4 images of cats from Pascal VOC. With a patch classification accuracy of 52.49% for ViT-B/16 and 27.91% for ViT-L/14, it is clear that the alignment we seek is very poor at the patch level. Surprisingly, note that for ViT-L/14, a model known to provide better image level prediction performance than ViT-B/16, the patch level alignment is significantly worse. Hence, pre-trained CLIP models cannot be used for open vocabulary segmentation as the CLIP contrastive learning objective does not ensure patch level alignment between image and text modalities.

4. Semantic Coherence in Vision Encoders

Due to the poor patch level alignment between pre-trained CLIP image and text encoders, our next question is whether we can train such an alignment in CLIP. This would however require the pre-trained vision encoder to be semantically coherent. In other words, semantically similar regions in an image should produce similar patch representations in the vision encoder. This property has been studied before in image self-supervised models like DINO [19]. We use a similar test to quantify semantic coherence of CLIP’s vision encoders.

In particular, we collect all patch representations from the vision encoder for each image in Pascal VOC and store the corresponding target classes using the segmentation labels. Let $f_v(x_{1,1},j) \in \mathbb{R}^{D_v}$ and $f_v(x_{2,p,q}) \in \mathbb{R}^{D_v}$ be the patch representations obtained at index $(i,j)$ of image $x_1$ and index $(p,q)$ of image $x_2$ respectively. We compute the cosine similarity $(f_v(x_{1,1},j) \cdot f_v(x_{2,p,q}))$ between the patch representations and use this as a binary classifier to predict if the two patches have the same target label. Let the segmentation labels for the two patches be $l(x_{1,1},j)$ and $l(x_{2,p,q})$ respectively. Since we have labels for each pixel, we decide the label for each patch by majority-voting. The target value for binary classification is 1 if $l(x_{1,1},j) = l(x_{2,p,q})$, else 0. Note that performance on this binary classification task is indicative of semantic coherence, as a good classifier would require patch representations corresponding to same labels to have high cosine similarity and vice-versa.

We present the ROC curve and the AUROC scores for CLIP and DINO in Figure 3. Surprisingly, we find that CLIP’s vision encoders outperform DINO on semantic coherence. This is encouraging as it indicates that we can indeed train a mapping between similar vision tokens and their corresponding text representations. We also present qualitative results in Figure 4 where we plot the patch level cosine similarity between a chosen patch (marked in yellow X in Figure 4a) and the remaining patches in the same image as well as a different image having the same class (dog). We do this for CLIP ViT-B/16 in Figure 4b and Figure 4c and for DINO ViT-B/16 in Figure 4d and Figure 4e. In both cases, CLIP’s encoder seems to perform at par or better than DINO. Motivated by these observations, in the next section, we discuss a method to train a patch level alignment between the vision tokens and the CLS text token in CLIP using purely image-text data.

5. Patch Aligned Contrastive Learning (PACL)

In the previous section, we showed that although CLIP lacks a patch level alignment between image and text representations, such an alignment can indeed be trained. However, note that this is a difficult problem as there is no ground-truth text data annotating each patch in an image-text dataset. Hence, training such an alignment can only be done in a weakly supervised fashion. Inspired from previous work on weakly supervised phrase grounding [18], in this section, we propose a modification on CLIP’s contrastive loss, to learn an alignment between the vision patch tokens and the CLS text token.

A modified compatibility function for contrastive loss: Our method is simple in the sense that the only change we make to CLIP’s training is in the compatibility function of its contrastive loss. Normally, for an image-text pair $(x,y)$, CLIP computes the CLS vision and text embeddings as $e_v(f_v(x))$ and $e_t(f_t(y))$ respectively, where $f_v : \mathbb{R}^{C,H,W} \rightarrow \mathbb{R}^{D_v}$, $f_t : \mathbb{R}^{L} \rightarrow \mathbb{R}^{D_t}$ are the vision and text encoders and $e_v : \mathbb{R}^{D_v} \rightarrow \mathbb{R}^D$, $e_t : \mathbb{R}^{D_t} \rightarrow \mathbb{R}^D$ are the vision and text embedders to project the representations into the same dimensional space. The compatibility function $\phi(x,y)$ is the cosine similarity between the vision and text CLS embeddings: $\phi(x,y) = \frac{e_v(f_v(x)) \cdot e_t(f_t(y))}{|e_v(f_v(x))| \cdot |e_t(f_t(y))|}$. Given this compatibility function, CLIP uses the InfoNCE [38] contrastive loss to learn vision and text representations.
which are aligned for similar image-text pairs:

\[
\mathcal{L}_x = \frac{1}{k} \sum_{i=1}^k \left( \frac{e^{\theta(x_i, y_i)}}{\sum_{j=1}^k e^{\theta(x_i, y_j)}} \right)
\]

\[
\mathcal{L}_y = \frac{1}{k} \sum_{i=1}^k \left( \frac{e^{\theta(x_j, y_i)}}{\sum_{j=1}^k e^{\theta(x_j, y_j)}} \right)
\]

(1)

with the contrastive loss being \( \mathcal{L}_{\text{InfoNCE}} = 1/2(\mathcal{L}_x + \mathcal{L}_y) \).

Note that the above loss function produces an alignment between the CLS image and text tokens but as we observed in Section 3, it does not produce the desired alignment at patch level between vision and text encoders. In order to then train this alignment, we make the following changes to CLIP’s loss. First, we use the patch tokens instead of the CLS token from the vision encoder, \( f_v : \mathbb{R}^{C,H,W} \rightarrow \mathbb{R}^{T,D_v} \), where \( T \) is the number of tokens or patches. Next, we use a modified vision embedder \( \hat{e}_v : \mathbb{R}^{T,D_v} \rightarrow \mathbb{R}^{T,D} \) to generate embeddings in the shared \( D \)-dimensional space for all patch tokens. We compute the patch level similarity

\[
s(x, y) = \hat{e}_v(f_v(x)) e_t(f_t(y))
\]

(2)

between all vision patch embeddings and the CLS text embedding, where \( s(x, y) \in \mathbb{R}^{T} \). We normalize the patch level similarity to the range \([0,1]\) by applying a softmax function across tokens, \( \alpha(x, y) = \text{softmax}(s(x, y)) \). Finally, we take a weighted sum across all vision patch embeddings where the weights of the tokens are obtained from the patch level similarities \( \alpha(x, y) \) as:

\[
\hat{\theta} = \hat{e}_v(f_v(x))^T \alpha(x, y)
\]

(3)

where \( \hat{\theta} \in \mathbb{R}^{D} \). The updated compatibility function \( \hat{\phi}(x, y) \) is then computed as the following dot product:

\[
\hat{\phi}(x, y) = \left( \frac{\hat{\theta}}{||\hat{\theta}||}, \frac{e_t(f_t(y))}{||e_t(f_t(y))||} \right).
\]

(4)

We use this modified compatibility function with InfoNCE contrastive loss for training and we call this method Patch Aligned Contrastive Learning. Figure 5, shows a diagrammatic representation of the steps involved in computing the compatibility function for an image-text pair.

**Grounded in Mutual Information:** To understand how our compatibility function \( \hat{\phi}(x, y) \) works, we go back to the relation of the InfoNCE [38] loss with mutual information (MI). Let \( x \in X \) and \( y \in Y \) be two multivariate random variables with a joint probability function \( p(x,y) \). MI between \( x \) and \( y \), computed as \( \mathbb{I}[x,y] = \mathbb{E}_{p(x,y)} \log \frac{p(x,y)}{p(x)p(y)} \), captures the amount of information shared between \( x \) and \( y \). However, MI is computationally intractable and hence requires approximations in order to be estimated. The InfoNCE loss \( \mathcal{L}_{\text{InfoNCE}}(\theta) \) defined using a compatibility function \( \phi(x, y) \) with model parameters \( \theta \) provides such an estimate and is a lower bound on MI as:

\[
\mathbb{I}[x,y] \geq \log(k) - \mathcal{L}_{\text{InfoNCE}}(\theta),
\]

where \( k \) is the batch size in InfoNCE loss with one positive sample and \( k - 1 \) negative samples per batch. Hence, minimizing \( \mathcal{L}_{\text{InfoNCE}} \) maximises the lower bound estimate of MI.

In vanilla CLIP training, the random variables \( x \) and \( y \) are images \( x \) and texts \( y \) respectively, the compatibility function \( \phi(x, y) \) is \( \phi(x, y) \), i.e., the cosine similarity between CLS vision and text token embeddings, and the model parameters are \( \theta = \{f_v, e_v, f_t, e_t\} \). Since, we modify the compatibility function \( \hat{\phi}(x, y) \) using a weighted sum over vision tokens, to maximise MI \( \mathbb{I}[x,y] \) between image and text, \( \mathcal{L}_{\text{InfoNCE}} \) will have to attend to regions of the image which correspond to the task and assign such regions a higher value in \( s(x, y) \). This indicates that \( s(x, y) \) intuitively captures patch level alignment between image and text modalities. To empirically verify this, we conduct the same patch level classification task described in Section 3 where for each patch, we compute the similarity \( s(x, y) \) for all classes and predict the class with the highest similarity. Results are in Table 1, third column (Post-Alignment) with qualitative results in Figure 2, third row. In both cases, we indeed observe a stark improvement in patch level alignment compared to vanilla CLIP using our modified compatibility function.

It is worth noting here that a similar contrastive learning approach has been used for the problem of weakly supervised phrase grounding in [18]. Their approach learns a mapping between ROI features from an object detector and word representations from a language model using an attention based weakly supervised contrastive learning. Although similar to our approach, they require the use of an object detector to provide ROI features, whereas we use CLIP’s vision encoder patch tokens as region features, having shown (see Section 4) that such features indeed are semantically coherent. Furthermore, they also use a contextualised language model to generate negative samples for contrastive loss, whereas our method fits in seamlessly with the contrastive setting in CLIP. Finally, whereas they target weakly supervised phrase grounding, we aim to learn a multi-modal model which is zero-shot transferable to the task of open vocabulary semantic segmentation.

**Inference:** At inference time, we can compute both image level as well as dense predictions. For image level predictions, similar to CLIP, we simply use our compatibility function \( \hat{\phi}(x, y) \) to compute similarity between an image and text. For semantic segmentation, given an image \( x \) and a set of classnames \( Y = \{y_1,...,y_C\} \), we compute \( s(x, y_c) \) \( \forall c \in \{1,...,C\} \) as a mask for each class and then use a softmax across classes. In the next section, we provide a detailed set of experiments to show the performance of our approach at both zero-shot semantic segmentation as well as image classification tasks.
Figure 5. Compatibility function $\phi(x, y)$ for Patch Aligned Contrastive Learning (PACL). The image encoder $f_v$ and embedder $e_v$ produce patch level representations for each image whereas the text encoder $f_t$ and embedder $e_t$ produce the CLS representation for a given text. We compute the cosine similarity between the CLS text embedding and the vision patch embeddings and use them as weights to take a weighted sum over vision patch tokens. We use the cosine similarity between the weighted sum and the CLS text token as our compatibility $\phi(x, y)$.

6. Experiments & Discussion

6.1. Zero-shot Semantic Segmentation

In the previous section, we described PACL, a multi-modal contrastive objective to train an alignment between vision patch embeddings and CLS text embeddings in CLIP. In this section, we evaluate the quality of this alignment through zero-shot transfer to semantic segmentation. We present implementation and training details for PACL, evaluation settings for zero-shot segmentation, and finally, results and a discussion on the same.

Training a small vision embedder: In Section 4, we have shown that CLIP’s pre-trained vision encoders $f_v$ have a relatively strong semantic coherence. In order to leverage this coherence and the large scale pre-training of CLIP, we keep the image encoder $f_v$, the text encoder $f_t$ and the text embedder $e_t$ frozen from a pre-trained CLIP model. We only train the vision embedder, i.e., $\theta = \{e_v\}$. Note that the modification of the vision encoder from $f_v$ (outputs the CLS vision token) to $\hat{f}_v$ (outputs the patch tokens) does not require any re-training. For $\hat{f}_v$, we use a residual block with two linear layers in the main branch and a single linear layer in the residual connection. There is a ReLU non-linearity between the two linear layers (see Appendix A.1). We find this simple architecture to work well for our applications.

Image-text datasets for training: We train our model purely on publicly available image-text datasets. In particular, we use Google Conceptual Captions (GCC) 3M [44], Google Conceptual Captions (GCC) 12M [6] and YFCC-15M, a subset of YFCC-100M [46] provided by CLIP [41], with a total number of approximately 30M training samples. Similar to GroupViT [56], in addition to the text descriptions in the datasets, we extract nouns from these descriptions, and randomly select one of 7 CLIP prompts (like “itap of a (.”), see Appendix A.2.2, to form sentences with these nouns. We add these sentences to the text descriptions as well. More details on the datasets can be found in Appendix A.3. Note that we do not use any segmentation annotations or class-agnostic segmentation masks during training. Further training details are in Appendix A.2.

Stride trick at inference: Since CLIP ViT-B/16 and ViT-L/14 use either $16 \times 16$ or $14 \times 14$ patches, the number of tokens generated is much smaller than the number of pixels, which is a problem for fine-grained predictions in segmentation. One workaround is to upscale the image at inference time to a larger size. We however find instead that a change to the stride of the convolutional layer to extract image patches in ViT can provide better fine-grained patches at inference time. In particular, we use a stride of $4 \times 4$ and upscale the resulting segmentations to image size using bi-linear interpolation.

Segmentation datasets for evaluation: Similar to recent works [17, 56] on zero-shot semantic segmentation, we use the following datasets for evaluation: a) Pascal VOC [16] (PV-20): 20 foreground classes with 1449 validation images, b) Pascal Context [36] (PC-59): 59 classes with 5k validation images, c) COCO Stuff [4] (CS-171): 171 “thing” or “stuff” classes and 5k validation images, d) ADE20K [63] (A-150): 150 classes with 2k validation images. Further details on these datasets can be found in Appendix A.3. For all datasets, we report the mean intersection over union (mIoU) [16], the most popular evaluation metric for semantic segmentation.

Comparative Baselines: We compare PACL with some of the most recently published methods on zero-shot semantic segmentation. In particular, we use LSeg [28], ViLSeg [32], GroupViT [56] and OpenSeg [17] as comparative baselines. In addition, we also compare with two relatively older approaches in zero-shot segmentation: SPNet [55] and ZS3Net [3]. Note that some of these methods work under
relatively relaxed constraints. In particular, SPNet, ZS3Net and LSeg use full segmentation annotations during training and OpenSeg uses class-agnostic segmentation masks. Furthermore, unlike us, ViL-Seg, SPNet and ZS3Net evaluate on a small subset of “unseen” classes from Pascal VOC, Pascal Context and COCO Stuff. To our knowledge, GroupViT and ViL-Seg are the only two methods which solely use image-text data for training. We also add a baseline using vanilla CLIP by taking the alignment between the vision patch embeddings and the text CLS embedding from CLIP’s pre-trained model.

Results & discussion: In Table 2, we report the mIoU for each baseline on the 4 segmentation datasets mentioned above. Note that the numbers shown for SPNet, ZS3Net and ViL-Seg are obtained from the ViL-Seg paper [32] and the numbers for all other baselines are obtained from their respective papers (cited in the table). In Figure 6, we show qualitative results of our method (i.e., PACL + CLIP) on Pascal VOC and ADE20K images (more in Appendix B.2). With mIoU scores of 72.3, 50.1, 38.8 and 31.4 on Pascal VOC, Pascal Context, COCO Stuff and ADE20K respectively, it is clear that PACL outperforms all other baselines consistently even though it works under a stricter set of assumptions, i.e., it does not use any segmentation annotations and is evaluated on all classes of the segmentation datasets. This is further corroborated from our qualitative results in Figure 6. It is interesting to note from Figure 6 that vanilla CLIP mostly seems to identify the correct classes in its predictions, just not the locations of those classes within the image. This relates to the problem of a lack of alignment between the CLS text token and the vision patch tokens which we had seen earlier (see Figure 2) and this problem is solved through the introduction of the PACL contrastive objective. Since PACL, as an approach, is not tied to any particular encoder, we next test its performance using different pre-trained encoders as well as different datasets on the zero-shot segmentation task.

Ablations on datasets and encoders: We perform an ablation by training PACL on a combination of different image-text training sets and different pre-trained vision encoders. For vision encoders we use CLIP ViT-B/16, CLIP ViT-L/14 and also use DINO’s [5] ViT-B/16 models. For training sets, we use GCC12M, (GCC12M + YFCC15M) and (GCC3M + GCC12M + YFCC15M). We report the mIoU obtained on Pascal VOC from each of the (model, dataset) combinations in Table 3.

These results provide two surprising observations. Firstly, PACL seems to generate an alignment even between DINO’s pre-trained vision encoder and CLIP’s text encoder, although these encoders have been trained separately and independently of each other. With an mIoU of 55.4, even the worst performing DINO baseline outperforms all competitive zero-shot segmentation baselines in Table 2 except OpenSeg. Secondly, PACL trained using CLIP’s ViT-B/16 consistently outperforms ViT-L/14 even though ViT-L/14 is known to be a clear winner in terms of image level zero-shot tasks. In fact, there is a trend in performance where CLIP ViT-B/16 outperforms CLIP ViT-L/14 which outperforms DINo ViT-B/16. This is also noticeable in Figure 7 where CLIP encoders generate relatively better segmentation masks than DINO. This observation is strongly reminiscent of the one in Section 4 and Figure 3, where we note that semantic coherence\(^2\) is strongest in CLIP ViT-B/16 followed by CLIP ViT-L/14 and finally by DINO ViT-B/16. These empirical observations suggest that PACL is a general contrastive learning method which can be used to train a patch level alignment and works independent of vision and text encoders as long as the vision encoders exhibit the property of semantic coherence. Indeed, semantic coherence seems to be the most important factor behind the success of PACL.

\(^2\)Semantic coherence is the property which enables a vision encoder to generate similar patch/token level representations for semantically similar regions of an image.
Table 2. Results on zero-shot semantic segmentation on Pascal VOC (PV-20), Pascal Context (PC-59) and COCO Stuff (CS-171) and ADE20K (A-150) datasets. We provide the encoder architecture, external training dataset (if any) as well as if those methods use segmentation annotations or class-agnostic segmentation masks. Our method (CLIP + PACL) consistently outperforms all previous approaches.

Table 3. Ablation on zero-shot segmentation across encoder architectures and datasets on Pascal VOC (PV-20). In the Text Encoder column, B/16 (L/14) indicates the pre-trained text encoder trained for CLIP ViT-B/16 (L/14).

6.2. Image Classification & Future Work

In Section 5, we mention that the modified compatibility function of PACL can be used to make image level predictions, similar to CLIP. In this section, we test our PACL models on zero-shot image classification. We then end with a discussion of possible future avenues from our work.

Zero-shot image classification results: We apply PACL trained using CLIP ViT-B/16 and ViT-L/14 encoders on (GCC3M + GCC12M + YFCC15M) to zero-shot image classification on 12 different datasets including ImageNet [14], 4 datasets considered to be standard distribution shifts on ImageNet: ImageNet-A [21], ImageNet-R [20], ImageNet-Sketch [48] and ImageNet-V2 [42], as well as 7 other standard classification datasets, detailed in Appendix A.3. We report the difference in classification accuracy between PACL + CLIP and vanilla CLIP for all the datasets in Figure 8 (all classification accuracies in Appendix B.3). PACL + CLIP outperforms vanilla CLIP on 10 and 7 out of the 12 classification datasets for ViT-B/16 and ViT-L/14 encoders respectively. Also note that except on ImageNet-R for ViT-L/14, PACL consistently outperforms vanilla CLIP on ImageNet and its distribution shifts. This observation is encouraging as it provides evidence in favour of our approach being applicable for image level applications in addition to segmentation. In the remainder of this section, we discuss possible avenues for future research from our work.

Exploring PACL for image-level applications: As seen above, since PACL is a general compatibility function for contrastive loss, it can be applied to all image level tasks. We show this through zero-shot image classification. However, it would be interesting to further explore PACL as an independent contrastive learning method. In particular, training models from scratch on PACL instead of the standard CLIP loss might provide additional benefits in the context of general VLP tasks like image-text retrieval [50]. Since our work is focused around zero-shot semantic segmentation, we keep this exploration out of the scope of this work and as potential avenue for future research.

Exploring other ways to generate patch level alignment: All the current methods on zero-shot open vocabulary segmentation, including ours, use CLIP like models, i.e., models with individual vision and text encoders with a fusion of modalities at the end of the encoders. However, there could be other ways of fusing modalities which could also lead to a generation of patch level alignment between image and text. In particular, one of the seemingly likely candidates of multi-modal fusion for generating patch level alignment could be cross-attention between image and text tokens, often seen in architectures used in VLP training [25, 31, 45] etc. Studying the patch level alignment in these models to see if they can be transferred to dense prediction tasks is also an interesting area of future exploration.

7. Conclusion

In this work, we explored Patch Aligned Contrastive Learning (PACL), a modified compatibility function for image-text contrastive loss which learns an alignment between patch tokens obtained from a ViT based vision encoder and the CLS token from a text encoder. We show that such an alignment allows a model to identify regions of an image corresponding to a given text input, thereby enabling a seamless zero-shot transfer to the task of semantic segmentation, without requiring any segmentation annota-
tions or masks during training. On 4 different segmentation datasets, we beat previous approaches on zero-shot open vocabulary segmentation, including the ones which use expensive segmentation annotations or masks for training. Finally, we show that PACL can also be used to make image level predictions and, when used with a pre-trained CLIP encoder, provides a general improvement in classification accuracy across 12 different image classification datasets.

References


[31] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Fei-Fei Li, MarcoAndreeto, Marc’AurelioRanzato, and RoozbehMottaghi. XianjieChen, XiaobaiLiu, Nam-GyuCho, Seong-WhanLee, SanjaFidler, RaquelUrtasun, and AlanYuille. The role of context for object detection and semantic segmentation in the wild. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014. 2, 6, 8, 13


[40] Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, RaduSoriciut, and VittorioFerrari. Connecting vision and language with localized narratives. In ECCV, 2020. 8


[46] BartThomee, DavidA Shamma, GeraldFriedland, BenjaminElizalde, KarlNi, DouglasPoland, DamianBorth, and Li-JiaLia. Yfcc100m: The new data in multimedia research. In Proceedings of ACL, 2015. 6, 8


A. Additional Implementation Details

In this section, we describe the implementation details for the proposed PACL method. Particularly, in Appendix A.1, we describe the architecture of the vision embedder used for training PACL, in Appendix A.2, we describe specifics of training including hyperparameters and prompt engineering details. Finally, in Appendix A.3, we describe details of image-text datasets used for training as well as segmentation and image classification datasets used for evaluation.

A.1. Vision Embedder Architecture

In Section 6.1, we have discussed that the proposed PACL approach is flexible in the sense that PACL can be applied using pre-trained frozen encoders. Particularly, since CLIP’s pre-trained vision encoders have desirable properties (see Semantic Coherence in Section 4), we use these pre-trained encoders from CLIP to train PACL to transfer to the task of zero-shot semantic segmentation. This simplifies the training to just a small vision embedder on top of the vision encoder. In this section and in Figure 9, we present the architecture of the Vision embedder. In particular, we use a single residual block with two linear layers in the main branch and a single linear layer in the residual branch. There is a ReLU non-linearity between the two linear layers in the main branch. The resulting model requires training a mere 1.1M parameters whereas the architecture has a total of 150M parameters for CLIP ViT-B/16. This helps us in scaling up and training on a larger batch size for our experiments as there is no gradient propagation through the frozen image and text encoders.

A.2. Training details for Vision Embedder

In Appendix A.2.1, we describe the architecture of pre-trained encoders as well as the hyperparameters used for training the PACL models. In Appendix A.2.2, we provide some details on CLIP’s prompt engineering used to derive best results from the text encoder of a pre-trained CLIP model.

A.2.1 Architecture and Hyperparameters

As mentioned above, we only train PACL using a Vision embedder on top of a pre-trained CLIP vision encoder. This allows us the flexibility not only of using multiple pre-trained vision encoders but also combinations of different vision and text encoders. In Section 6.1, we show an ablation with combinations of different pre-trained vision and text encoders. In particular, we use: a) CLIP ViT-B/16 vision and text encoders, b) CLIP ViT-L/14 vision and text encoders and b) DINO ViT-B/16 vision encoder with CLIP ViT-B/16 text encoder. For each of these combinations, we train a vision embedder as discussed in Appendix A.1 and report zero-shot semantic segmentation results in Table 3 of the main paper.

All our models are trained on a single node with 4 NVIDIA A100 GPUs with a GPU memory of 40GB in each GPU. We use AdamW as the optimizer with beta values 0.9 and 0.98, an eps value of 1e−6 and a weight decay of 0.2. We use an initial learning rate of 5e−4 and reduce the learning rate using a Cosine Annealing schedule where the maximum number of iterations is set as the total number of iterations during training (i.e., number of epochs × number of iterations per epoch). We use a batch size of 4096 (1024 per GPU) and train the model for a total of 10 epochs on image-text data. We do not use any segmentation annotations or class-agnostic segmentation masks during training. We provide details on these image-text datasets in Appendix A.3. When the training dataset is a combination of GCC-3M, GCC-12M and YFCC-15M, the model takes 10 days to train on 4 NVIDIA A100 GPUs.

A.2.2 Prompt Engineering

Since we use CLIP’s [41] pre-trained text encoders, we follow the prompt engineering guidelines following CLIP’s OpenAI repository during inference time. In particular, during inference, we compute the average embedding from the text encoder using a set of 7 prompts: itap of a (), a bad photo of the (), a origami (), a photo of the large (), a () in a video game, art of the (), a photo of the small (), where we put the name of the class within the parenthesis (). We use the mean of the embeddings from the the prompts for each class in order to compute cosine similarity with the patch representations from the vision encoder. This is similar to the way CLIP performs zero-shot image classification, however CLIP uses only the CLS token from the vision encoder to compute cosine similarity.

A.3. Training and Evaluation Datasets

Image-text datasets for training: We use primarily 3 different image-text datasets for training all our models. Firstly, we use Google Conceptual Captions (GCC)
3M, which contains approximately 3 million images, each annotated with a caption. The images are scraped from the web and the corresponding captions are obtained from the AL-text HTML data associated with each image from the web. Secondlly, we use Google Conceptual Captions (GCC) 12M, which is similar to GCC-3M but containing a much larger corpus of image-text pairs with approximately 12 million samples. The primary purpose of GCC-12M is for pre-training whereas GCC-3M is a relatively less noisy dataset meant for fine-tuning pre-trained models. Thirdly, we use YFCC-15M, a subset of 15 million samples from the popular YFCC-100M [46] dataset, which is one of the largest publicly available datasets containing image-text information obtained from Flickr. The subset of approximately 15 million images is defined by CLIP [41] by filtering images from YFCC-100M with natural language titles and/or descriptions in English.

**Semantic segmentation datasets for zero-shot segmentation:** We use the following semantic segmentation datasets for zero-shot evaluation on the task of semantic segmentation: a) Pascal VOC [16]: it has 20 foreground classes and 1 background class with 1449 validation images. We measure performance only on foreground classes and mask predictions with entropy above 1.5 as background, b) Pascal Context [36]: it has 59 classes with 5k validation images of indoor and outdoor scenes, c) COCO Stuff [4]: it has 172 classes categorised into either “thing” classes or “stuff” classes and has 5k validation images, d) ADE20K [63]: the version we evaluate on is widely used and has 150 classes with 2k validation images. For all datasets, we report the mean intersection over union (mIoU), the most popular evaluation metric for semantic segmentation.

**Image classification datasets for zero-shot classification:** We evaluate PACL on a suite of 12 image classification datasets which include ImageNet [15], 4 well-known distribution shifts on ImageNet as well as 7 other popular image classification datasets. ImageNet is a very popular image classification dataset with 1000 classes relating to concepts contained in the WordNet hierarchy. We use 50000 validation samples in ImageNet for evaluation. The 4 datasets considered to be popular distribution shifts on ImageNet are: a) ImageNet-A [21] which contains natural real-world images from 200 classes in ImageNet but which are mostly mis-classified by well-known ResNet classifiers, b) ImageNet-R [20] which contains cartoons, graphics and other art renditions of images from 200 classes in ImageNet, c) ImageNet-Sketch [48] which contains 500000 validation images, 50 from each of the 1000 ImageNet classes constructed by making the Google search, “sketch of ()” where () is the ImageNet class concerned and d) ImageNet-V2 [42] which has 10000 validation images obtained by following the same collection procedure as ImageNet original images, in order to make the distribution of ImageNet-V2 as similar as possible to ImageNet. The other 7 image classification datasets include: a) CIFAR-10 [27] having 10000 test images from 10 classes including different types of automobiles and animals, b) CIFAR-100 [27] having 10000 test images from 100 classes instead of 10 obtained in a similar fashion as CIFAR-10, c) Stanford Cars [26] with 8041 test images containing cars of different makes and models, d) Caltech-101 [30] having 101 categories of images with 40-800 images per class, e) Food-101 [2] containing 101 classes of food items organized by the type of food, with approximately 25000 test images, f) Oxford-IIIT Pets [39], a dataset with 37 categories of pets with approximately 200 images per class, and g) Flower dataset [37] having 102 different categories of flowers with between 40 and 258 images for each class.

**B. Additional Results**

**B.1. Semantic Coherence in CLIP**

Semantic coherence is a property of ViT based vision encoders where semantically similar regions of the image have similar patch/token level representations in the feature space of the vision encoder. In Section 4 and Figure 4, we have shown both quantitative and qualitative results comparing the semantic coherence of a CLIP and a DINO ViT-B/16 vision encoders. Particularly, we had seen that CLIP’s ViT-B/16 vision encoder performs better than DINO. In Figure 10, we present additional qualitative examples to further corroborate our observations in Section 4. We show qualitative examples from the Bird, Plane and Sheep classes in Pascal VOC and plot the patch level similarity between a selected patch from the original image (marked using a yellow cross) and all patches from the same image as well as a different image. The similarity is shown using a heatmap where yellow and red shades indicate high similarity and blue shades indicate low similarity. Our observations are similar to the ones in Section 4 and we see that CLIP performs competitively or better than DINO. While DINO seems to cover semantically meaningful regions in the images, it doesn’t cover the entirety of the relevant object. CLIP seems to be doing a better job at covering all the patches for the object as highly similar to the marked patch, thereby indicating better semantic coherence.

**B.2. Qualitative Segmentation Results**

In Figure 6, we showed qualitative results for the task of zero-shot semantic segmentation on both Pascal VOC and ADE20K datasets. In this section, we present more qualitative results on the same. Particularly, in Figure 11, we show additional qualitative results on 8 images from Pascal VOC covering different concepts including bus, cat, dog, bird, potted plant, bottle and plane. Similarly, in Figure 12.

Figure 10. Additional qualitative results on semantic coherence between CLIP and DINO ViT-B/16. a): we show the original image of a class (bird in top row, aeroplane in middle row and sheep in bottom row) with the patch marker (yellow X near the centre). b, c): we show CLIP vision encoder cosine similarity across all patches for the same and a different image of the same class. d, e): we show the same for DINO.

Figure 11. Additional qualitative results on zero-shot semantic segmentation on Pascal VOC. The first row shows the original images, the second row shows the corresponding ground-truth labels and the third row shows the predictions from our best performing model, i.e., PACL trained using a pre-trained CLIP ViT-B/16 encoder on GCC-3M + GCC-12M + YFCC-15M.

we provide qualitative segmentation results from various indoor and outdoor scenes from ADE20K. Similar to our observations in the main paper, we find that the zero-shot segmentation results are decent and our models can recognise a large variety of concepts without ever having been trained on segmentation annotations or masks for any of them. This shows the potential of using the scale of large image-text datasets for zero-shot transfer to semantic segmentation.

B.3. Zero-shot Image Classification

In Figure 8 of the main paper, we showed the difference in zero-shot classification accuracies for PACL models trained with CLIP backbones as compared to vanilla CLIP models on a suite of 12 image classification tasks includ-