Supplemental Material

A. Extended implementation details

Here we provide additional implementation details to reproduce the results reported in this work. In particular, batch size was set to 5 and learning rate was fixed at 2.5e-4. Stochastic Gradient Descent was used to optimize the model with a momentum parameter 0.9, and a weight decay of 5e-4. We use a multi-scale testing scheme with the different scales set as 0.5, 0.75, 1.0, 1.25 and 1.5, and the outputs are aggregated using max-pool operations, following recent works [69] [66]. The choice of the default CRF hyperparameters chosen were guided by [7].

B. Additional details on the prompt learning experiments in Table 1

Table 1 in the main manuscript showcases the impact of various prompt learning techniques, when implemented for the fine-tuning of our models. In this section we summarize how these approaches were used in our work:

CoOp [77] uses a trainable context vector of a fixed length ($[\mathrm{V}]_1[\mathrm{V}]_2 \dots [\mathrm{V}]_N[\mathrm{CLS}]$.), which we set as 3 keeping in mind the length of the default context string: "A photo of". The vector was initialized by tokenizing the same string and encoding it using the pre-trained CLIP [47] encoder. The new trainable weights are trained using the same training objectives as in Equation 5. This technique essentially makes the context of the prompt learnable which can be analysed as a possible modification on [66].

DeFo [58] uses trainable context vectors as well as a trainable class vector ($[\mathrm{V}]_1[\mathrm{V}]_2 \dots [\mathrm{V}]_N[V_{CLS}$.). We set default context by tokenizing and encoding the context string "A photo of", as in our implementation of the CoOp run. The default class vectors are initialized in a same way using the classnames from the dataset. The training objective for the new weights were the same as the rest of the pipeline, as in the case of our implementation of the CoOp prompt learning strategy.

Target Optimization is the prompt learning alternative to the proposed POLE pipeline, because essentially it just learns the class vector, keeping the rest of the prompt fixed ('*A photo of* $[\mathrm{V}_{CLS}]$."). As in the other methods, the training objectives remains the same as in Equation 5. The class vectors are initialized using the same initialization scheme as DeFo and CoOp experiments.

Please note that as our subsequent downstream task is weakly supervised semantic segmentation, we could not employ the exact model specifications used by [58] and [76] in their entirety. We only used their prompt learning scheme for analyzing what impact these techniques may have on Weakly Supervised Semantic Segmentation. We also found 2.5e-7 experimentally to be a good learning rate for training the prompt weights, which is 0.001 times the learning rate for the rest of the pipeline. Using learning rates larger impacts the performance drastically. The process of handling the text prompts for each of these techniques is described schematically in Figure S1. The text prompt input it tokenized into a vector of length 77 (Diagram shows only 11 for simplicity). Each token in the string is then tokenized into an unique number that composes the tokenized vector. Typically, there is a number denoting the start of the prompt, called the *prefix*, followed by the *context* tokens, the *class* token, a *punctuator* token (fullstop, in this case) and a blank *suffix* (an array of zeroes). This vector is then embedded as floating point tensors where each token is turned into a vector of fixed length. We selectively convert these vectors as trainable parameters or frozen, depending upon what prompt learning scheme we replicate. Eventually, the CLIP text encoder encodes the embedding and generates a tensor of the same dimensions as that of the masked image encoding.

C. Additional details on the corpus and synonyms

In our method, we employ the synonyms obtained from the ChatGPT training corpus. As this corpus is not formally accessible, we use the chatGPT web application to provide synonyms to the ground truth categories. We used the input query prompt: *"Give me 5 semantically similar words for [CLS] and also print the cosine similarity scores based on CLIP model"*; where [CLS] is a classname. From the list of synonyms obtained from GPT, top m synonyms for each class were taken. The list of synonyms collected from the ChatGPT web application are listed in Table S1.

Table S1. List of synonyms for each class (associated PascalVOC ground truth) obtained from ChatGPT.

Class	Synonym 1	Synonym 2	Synonym 3
aeroplane	aircraft	airplane	plane
bicycle	bike	cycle	pedal bike
bird	avian	fowl	feathered friend
boat	ship	vessel	watercraft
bottle	flask	container	jar
bus	coach	transit	omnibus
car	automobile	vehicle	sedan
cat	feline	kitty	tomcat
chair	seat	armchair	recliner
cow	bovine	heifer	bull
dining table	kitchen table	dinner table	breakfast table
dog	canine	puppy	hound
horse	equine	mare	stallion
motorbike	motorcycle	bike	scooter
player	person	individual	human
potted plant	houseplant	flowerpot	planter
sheep	lamb	ewe	ram
sofa	couch	loveseat	settee
train	railway	locomotive	subway
tv monitor	television	display screen	flat screen

Furthermore, to evaluate the performance of using synonyms based on the popular ChatGPT, we trained our model based on four different corpus: English Wikipedia,

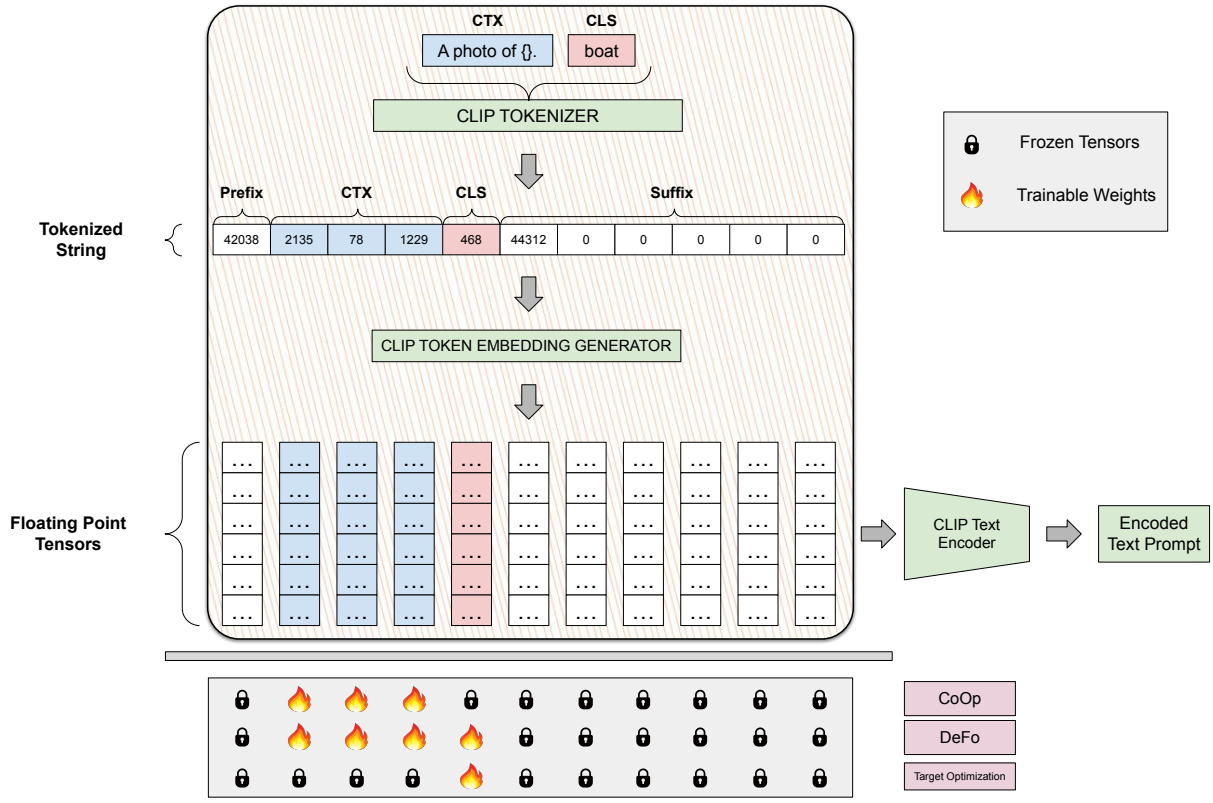

Figure S1. Schematic to show how the text prompts are processed and how each prompt learning technique modifies the process.

Google News, British National Corpus and English Gigaword. For each class, we search for the 10 closest words to the ground truth class name on the webvectors service (http://vectors.nlpl.eu/explore/embeddings/en/associates/). We then selected the top m words from the list for every class in each corpus. The similarity scores are based on the word2vec models employed by the web application, trained on the corpuses mentioned. For each corpus, we use the ground truth category name jointly with the top-m synonyms for every class, in order to construct a pool of words from which CLIP selects the closest word to a given masked image.

In Figure S2 we further investigate, for several images, which is the best synonym selected across the different corpus. We can observe, for example, that for most corpus the associated ground truth category to a given image is rarely selected. While these correlations may come from the subjectivity when describing an object (e.g., *'aeroplane" vs. 'plane"*, or *'tv monitor" vs.'television"*), we believe that in other cases the problem is magnified due to suboptimal category descriptions. Particularly, the class *'person"* is employed systematically in PASCAL VOC2012, which it is replaced in CLIMS [66] by 'player". Nevertheless, we observe that even having both class names in the list of potential synonyms, none of them present the highest correlation for a given image (i.e., three out of four corpus select other category names: *'someone"*, *'someone"* and *'human"*).

This analysis is further supported by the radar plots in Figure S3, which depict the frequency at which each ground truth category is selected as the [CLS] token. These plots reveal interesting observations, which suggest that only three classes (i.e., *'bus"*, *'dog"* and *'TV monitor"*) are indeed the most correlated categories in more than 80% of the images of the whole dataset. On the other hand, a vast majority of categories, the ground truth labels are not selected as optimal synonym in at least 50% of the images.

Figure S2. Which is the best synonym across corpus? This figure illustrates several examples of the best synonym selected (*green bar*) for the different corpus, compared to the associated image ground truth (*blue font*). We use red circles to identify the target class.

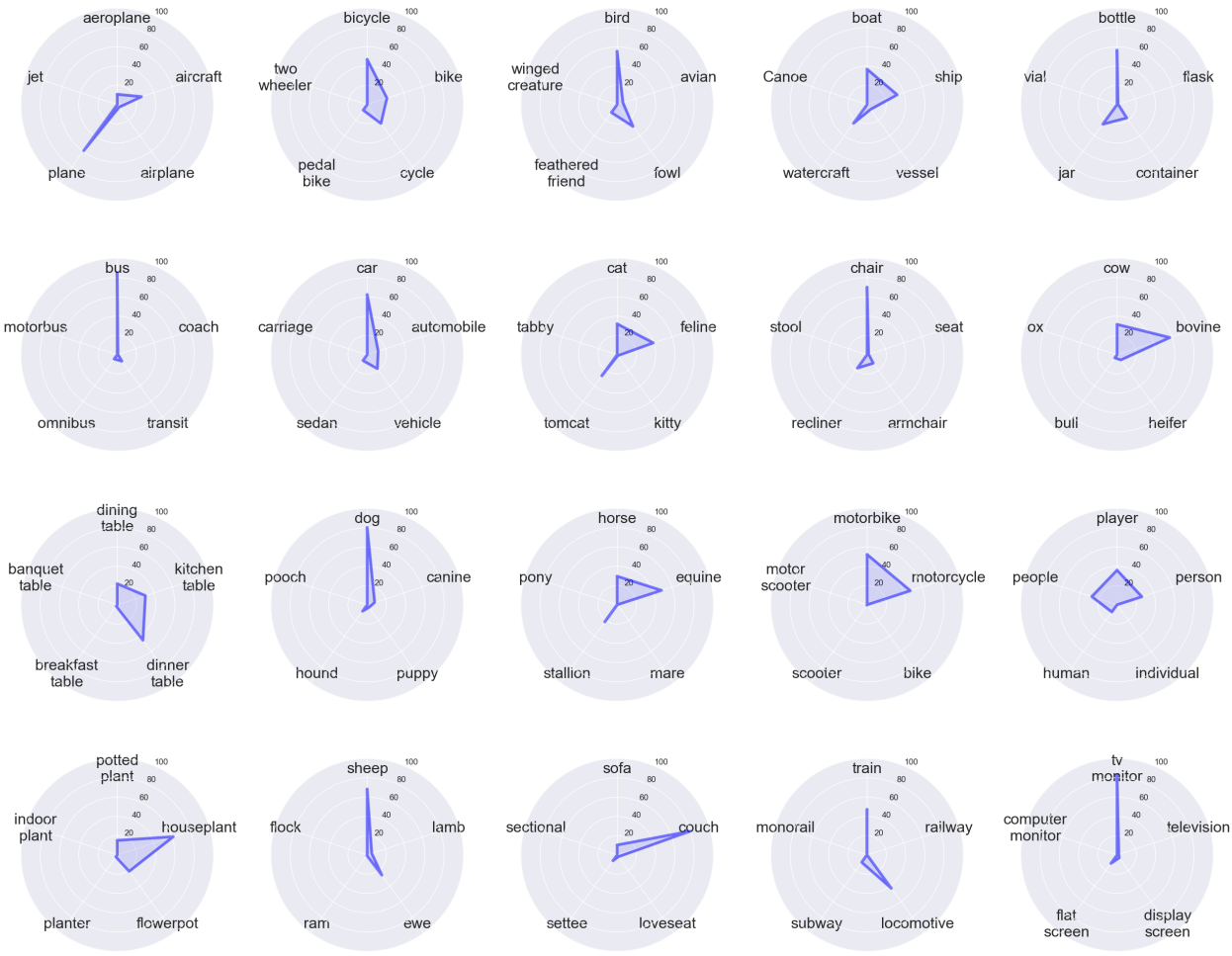

Figure S3. Classwise radial plots for respective fractions of the total number of instances, where a certain word was chosen for an instance of the class. An inward point on the plots indicates that the number of instances where the primary classname itself was chosen in the best prompt, is quite low.