AutoCLIP: Auto-tuning Zero-Shot Classifiers for Vision-Language Models

Anonymous authors
Paper under double-blind review

Abstract

Classifiers built upon vision-language models such as CLIP have shown remarkable zero-shot performance across a broad range of image classification tasks. Prior work has studied different ways of automatically creating descriptor sets for every class based on prompt templates, ranging from manually engineered templates over templates obtained from a large language model to templates built from random words and characters. Up until now, deriving zero-shot classifiers from the respective encoded class descriptors has remained nearly unchanged, i.e., classify to the class that maximizes cosine similarity between its averaged encoded class descriptors and the image encoding. However, weighing all class descriptors equally can be suboptimal when certain descriptors match visual clues on a given image better than others. In this work, we propose AutoCLIP, a method for *auto-tuning zero-shot classifiers*. AutoCLIP tunes per-image weights to each prompt template at inference time, based on statistics of class descriptor-image similarities. AutoCLIP is fully unsupervised, has only a minor additional computation overhead, and can be easily implemented in few lines of code. We show that AutoCLIP outperforms baselines across a broad range of vision-language models, datasets, and prompt templates consistently and by up to 3 percent point accuracy.

1 Introduction

Classifiers built upon vision-language models (VLMs) such as CLIP (Radford et al., 2021) and CoCa (Yu et al., 2022) have shown strong zero-shot transfer capabilities across various tasks. Such zero-shot transfer is appealing since it allows for obtaining high-performing classifiers on novel domains without the overhead of data acquisition and labelling. However, it has been observed that prompt engineering plays a crucial role for obtaining strong zero-shot classifiers, that is: zero-shot classifiers derived from VLMs need to be constructed based on a set of prompt templates (parameterized by the class name) that cover potential variation of the domain. These prompt templates can be hand-designed (Radford et al., 2021), generated by a large-language model (Menon & Vondrick, 2022), or randomly generated (Roth et al., 2023).

Prompts can also be learned via test-time prompt tuning (TPT) (Shu et al., 2022; Zhao et al., 2023). This approach makes the zero-shot classifier adaptable to the datum of interest, which is possible by effectively leveraging the knowledge of the general-purpose VLM. Shu et al. (2022) tune prompts so that the predictive entropy for a single image is minimized, while Zhao et al. (2023) maximizes a CLIP reward. These prior TPT methods require the VLM's image encoder to process several augmentations for each image. Moreover, gradients with respect to the prompts require backpropagation through the VLM's text encoder, thereby substantially increasing the overall inference cost.

We propose to not tune the prompts but instead use a large set of predefined and fixed prompt templates and to adapt the weights of those prompt templates for each image at test-time. This approach has the major advantage that adaptation takes place entirely in the embedding space without requiring additional forward or backward passes through the VLM's encoders, which significantly lowers the test-time computation and memory overhead compared to prior TPT methods. Our work is similar to Allingham et al. (2023), but comes

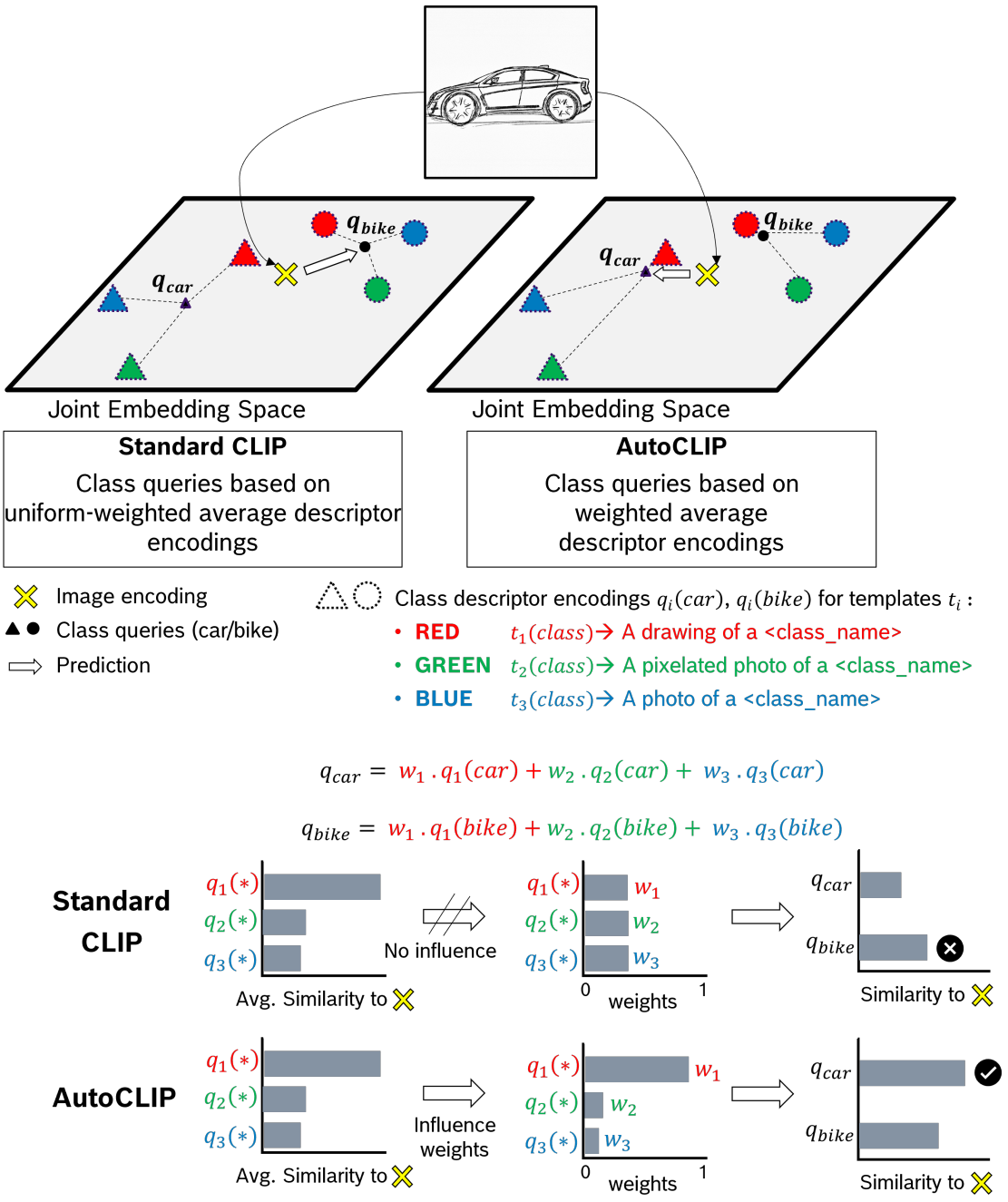

Figure 1: **Conceptual Illustration of AutoCLIP.** CLIP's zero-shot classifiers are based on a set of prompt templates t_i ("A photo of a <class_name>", "A drawing of a <class_name>", ...). Inserting class names c into these templates gives a set of class descriptors that are encoded into a joint embedding space together with the respective image. Standard CLIP averages encoded class descriptors $q_i(c)$ into class queries q_c, and classifies to the class that has maximal cosine similarity with the encoded image. However, this ignores that some prompt templates describe the image of interest better than others (their embeddings have higher average similarity): for instance, when the image is a drawing, the template "A drawing of a <class_name>" results in stronger class descriptors than other templates and should thus be weighted higher when computing class queries. AutoCLIP determines such weights directly from class descriptor-image similarities in the embedding space. Here, the car image is taken from Atkinson (2015).

with the major advantages that our approach can adapt weights for single samples and does not require access to the pre-training feature distribution.

We briefly summarize the standard way of constructing zero-shot classifiers from VLMs (see Figure 1 left). At first, a collection of prompt templates is instantiated for each class to form a set of class descriptors (e.g., "A photo of a *car*", and "A drawing of a *car*" are sample class descriptors of class *car*). These descriptors are processed by the text encoder and the resulting encoded descriptors are averaged to obtain the image-independent class queries (e.g. q_{car}). Besides, the image encoder processes the input image to be classified to get the image encoding, which lies in the same embedding space as class queries. The cosine similarity of the encoded image to every (averaged) class query is computed, and the output prediction is assigned to the class with maximum similarity.

This work follows a similar zero-shot classification setup, except that we change how class queries are computed. Instead of a simple average of the encoded class descriptors, we propose to take a weighted average, wherein weights of the encoded class descriptors are automatically tuned for each image separately. The weights are determined in a manner that prompt templates whose resulting class descriptors are closer to the respective image embedding get higher weightage than those being less similar (see Figure 1 right). Our approach is motivated by the intuition that prompt templates with high similarity describe relevant properties of the image better than ones with lower similarity (see Figure 6 for evidence supporting this intuition). We denote our method that automatically adapts the weights of the encoded class descriptors for each image as AutoCLIP.

We empirically show that AutoCLIP improves the performance of zero-shot classifiers across many datasets, VLMs, and prompt strategies with little inference-time overhead. Note that AutoCLIP is fully zero-shot as it does not require any supervision from the target task. Furthermore, AutoCLIP makes no assumptions on the underlying VLM and can thus be broadly applied, potentially also to multi-modal models beyond VLMs such as ImageBind (Girdhar et al., 2023b).

Overall, our main contributions are as follows: we introduce AutoCLIP (Section 3.2), a novel procedure for constructing zero-shot classifiers from vision-language models. AutoCLIP leverages statistics of class descriptor-image similarities to automatically determine weights of the prompt templates. We further discuss a method for automatically tuning AutoCLIP's step size such that the entropy of the prompt template's weights is controlled (Section 3.4). We propose a default entropy reduction factor, which is shared across all the experiments. By this, AutoCLIP comes essentially without free hyperparameters, which is important as hyperparameters cannot be tuned in zero-shot settings. We evaluate AutoCLIP on a large number of datasets, vision-language models, and prompt templates (Section 4) as well as in a controlled setting (Section 5). We find that it improves performance on the vast majority (85%) of settings, by 0.45 percent point accuracy on average, and by up to 3 percent point in some settings. These gains come essentially for free with the only cost being a very small inference time overhead (see Section A.1 in the appendix), as our approach operates entirely in the embedding space. Considering these benefits, we believe that the proposed AutoCLIP can serve as a default zero-shot inference strategy for VLMs.

2 Related Work

Vision-Language Pretraining. Deep learning with vision-language pretraining has enabled zero-shot transfer capabilities, i.e., the resulting vision-language models (VLMs) are able to perform zero-shot classification on vastly diverse unseen target datasets given only text prompts of individual target classes. CLIP is one of the state-of-the-art VLMs pretrained on the well-curated WebImageText dataset containing 400 million image-text pairs using a contrastive loss (Radford et al., 2021). In terms of datasets used, ALIGN requires less dataset preprocessing enabling training on a dataset of over a billion image-text pairs (Jia et al., 2021). Florence (Yuan et al., 2021) expands models to other common modalities (e.g., videos). In terms of the training loss, CoCa (Yu et al., 2022) leverages an additional captioning loss allowing models to be used in generative applications. In our work, we study how to optimally use text prompts of the target classes with these VLMs.

Prompt Construction. Conventionally, one or several manually designed text prompts per target class are employed for zero-shot classification (Radford et al., 2021; Jia et al., 2021). Recent research demonstrates that introducing additional prompts can improve overall performance. DCLIP (Menon & Vondrick, 2022) generates additional prompts based on querying the large-language model GPT-3 (Brown et al., 2020). WaffleCLIP (Roth et al., 2023) has shown that classification performance can be further boosted by appending random words or characters to predefined prompt templates. To derive a zero-shot classifier, these works weight all text prompts uniformly. In contrast, we propose an approach to adjust weights of individual prompts per input sample dynamically at test time.

Test-Time Adaptation. Our work can be considered as a test-time adaption approach for VLMs. TENT (Wang et al., 2020) demonstrates that adapting models to minimize prediction entropy can improve model performance at test time. In the context of VLMs, TPT (Shu et al., 2022) optimizes prompts of target classes based on the entropy minimization objective. RLCF (Zhao et al., 2023) demonstrates that minimizing the entropy objective can lead to overfitting under distribution shift and proposes adaptation based on average CLIP scores. In contrast to these previous works, we do not perform any adaptation of prompts or model parameters, but refine weights of individual (encoded) prompts, which is considerably cheaper in terms of computation and memory consumption. Most similar to our work is Zero-shot Prompt Ensembling (ZPE) (Allingham et al., 2023), which also determines prompt weights in embedding space. However, ZPE requires an entire batch of target domain samples and the availability of image features representing the feature distribution in pre-training ("source domain"). In contrast, our work operates on single images in a source-free setting.

3 AutoCLIP

We outline the common approach for building zero-shot classifiers for VLMs like CLIP in Section 3.1. Thereupon, we detail our proposed AutoCLIP as an auto-tuned alternative in Section 3.2, followed by describing how the required gradient can be calculated in closed-form in Section 3.3, and finally explain how AutoCLIP's step size can be automatically determined in Section 3.4.

3.1 Background: Zero-Shot Classifiers for Vision-Language Models

Let us consider a classification task $\mathcal{X} \mapsto \mathcal{C}$, where $\mathcal{X}$ corresponds to the input domain and $\mathcal{C} = \{c_1, \dots, c_C\}$ is a set of C classes. We assume that there exists a pretrained VLM such as CLIP that provides a joint embedding space $\mathcal{E}$ and corresponding embedding functions $E_X : \mathcal{X} \mapsto \mathcal{E}$ that maps input data $x \in \mathcal{X}$ into embedding space $\mathcal{E}$ and $E_T : \mathcal{T} \mapsto \mathcal{E}$ that maps text into the same embedding space $\mathcal{E}$. Let there be K prompt templates $t_1, \dots t_K : \mathcal{C} \mapsto \mathcal{D}$ that map class name $c \in \mathcal{C}$ to (textual) class descriptors $d \in \mathcal{T}$. These prompt templates can be either manually designed (Radford et al., 2021), generated by a large language model (Menon & Vondrick, 2022), or randomly generated (Roth et al., 2023). Algorithm 1 summarizes the standard zero-shot classifier for VLMs: average the class descriptor encodings $e^{(d)}$ into class queries q_j, then compute cosine similarities s_j between class query and encoded image $e^{(x)}$, and classify to the class that maximizes similarity.

3.2 Auto-Tuning Zero-Shot Classfiers

AutoCLIP modifies Line 7 in Algorithm 1. Instead of computing class queries as simple average of class descriptor encodings $q_j = 1/K \sum_{i=1}^{K} e_{ij}^{(d)}$, AutoCLIP uses a weighted average: $q_j = \sum_{i=1}^{K} w_i e_{ij}^{(d)}$ with learnable w satisfying $w_i \geq 0$, $\sum_{i=1}^{K} w_i = 1$, which we enforce by reparameterizing $w = \mathrm{softmax}(\rho)$ and $\rho \in \mathbb{R}^K$. AutoCLIP's guiding intuition (see Figure 1) is to assign higher weights w_i to prompt templates t_i that result in class descriptor encodings $e_{ij}^{(d)}$ that are more similar to the encoded image $e^{(x)}$, that is: t_i with large $e_{ij}^{(xd)} = e_{ij}^{(d)} \cdot e^{(x)}$ $(j = 1, \dots, C)$. This is inspired by the observation that class descriptors having higher similarity in the embedding space describe the image better (according to contrastive pretraining objectives in typical VLMs).

Algorithm 1 Zero-Shot Classifier for a single sample x

1: ▷ Generate $K \times C$ class descriptors
2: $d \leftarrow \{t_i(c_j) \,|\, i \in \{1, \ldots, K\}, j \in \{1, \ldots, C\}\}$
3: ▷ Encode image of interest x with VLM
4: $e^{(x)} \leftarrow E_X(x)/||E_X(x)||_2$
5: ▷ Encode all class descriptors with VLM
6: $e_{ij}^{(d)} \leftarrow E_T(d_{ij})/||E_T(d_{ij})||_2$
7: $w_i \leftarrow 1/K$ ▷ Uniform prompt template weights
8: **for** $j \in 1, \ldots, C$ **do**
9: ▷ Class queries as average class descriptor encodings
10: $q_j \leftarrow \sum_{i=1}^{K} w_i e_{ij}^{(d)}$
11: ▷ Cosine similarity between $e^{(x)}$ and class query q_j
12: $s_j \leftarrow e^{(x)} \cdot q_j$
13: **end for**
14: ▷ Assign x to class $c_{j^\star}$ with maximum similarity
15: $j^\star \leftarrow \arg\max_j s_j$

When determining the template's weights w, we have C descriptor-image similarities $e_{ij}^{(xd)}$ for each template t_i. AutoCLIP needs to aggregate those C similarities across classes when assigning larger weights to more relevant prompt templates. Intuitively, simply averaging all C similarities ("mean" aggregation) ignores that, in the classification objective, we ultimately only care about classes that result in the descriptors closest to $e^{(x)}$; however, taking only the class with highest similarity per template into account ("max" aggregation) ignores inherent ambiguity in the image and was found to be suboptimal (Roth et al., 2023). We propose a middle ground of aggregating via a smooth approximation to the maximum function via $\text{logsumexp}_j(e_{ij}^{(xd)}) = \log \sum_{j=1}^{C} \exp e_{ij}^{(xd)}$. This logsumexp aggregation takes all classes into account but assigns higher importance to more relevant classes (ones resulting in higher similarities to the image x). AutoCLIP then determines weights w_i such that $\text{logsumexp}_j(s_j) = \text{logsumexp}_j(\sum_{i=1}^{K} w_i e_{ij}^{(xd)}) = \text{logsumexp}_j(\text{softmax}(\rho) \cdot e_{:j}^{(xd)})$ gets increased by one step of gradient ascent in the direction of $\nabla_\rho \,\text{logsumexp}_j(\text{softmax}(\rho) \cdot e_{:j}^{(xd)})$. We note that $-\,\text{logsumexp}$ has been interpreted as the energy function of a data point (for appropriately trained classifiers) (Grathwohl et al., 2020); in this view, AutoCLIP can be interpreted as minimizing the energy and maximizing the probability density $p(x)$ of x under the zero-shot classifier (see Section A.3 for more details).

We summarize AutoCLIP in Algorithm 2. We initialize $\rho = \mathbf{0}$, which corresponds to an unweighted average of the class descriptor encodings (Line 8). Similar to Algorithm 1, we compute the pairwise cosine similarities s_j between encoded image $e^{(x)}$ and class queries q_j (Line 9-14). Instead of directly classifying to the class with maximum similarity to the image, AutoCLIP updates the class descriptor weights first. For this, the gradient $g = \nabla_\rho \,\text{logsumexp}_j(s_j)$ is computed (Line 16), an appropriate step size α is selected (Line 18, see Section 3.4), and $\rho = \alpha \cdot g$ and $w = \text{softmax}(\rho)$ are updated (Line 20). Based on the new w, AutoCLIP computes updated class queries q_j and class-image similarities (Line 21-26) and finally selects the class with maximum similarity for the image (Line 28). It is worth emphasizing that AutoCLIP is permutation-invariant in the prompt templates t_i.

We note that Line 9-20 could be repeated for several iterations with smaller step sizes; however preliminary experiments indicate no advantage of doing more than one iteration. We call AutoCLIP "auto-tuned" because its weights w are automatically adapted for every input independently. Moreover, we note that in practice, models like CLIP scale $e^{(xd)}$ by a learned temperature (exponential logit scale) τ to obtain well calibrated classifiers; we use the same temperature for scaling $e^{(xd)}$ in the logsumexp aggregation (as there is no labelled data in a zero-shot setting on which a temperature could be tuned).

Algorithm 2 AutoCLIP: Auto-Tuned Zero-Shot Classifier for a single sample x

▷ Generate $K \times C$ class descriptors
$d \leftarrow \{t_i(c_j) \,|\, i \in \{1, \ldots, K\}, j \in \{1, \ldots, C\}\}$
▷ Encode image of interest x with VLM
$e^{(x)} \leftarrow E_X(x)/||E_X(x)||_2$
▷ Encode all class descriptors with VLM
$e_{ij}^{(d)} \leftarrow E_T(d_{ij})/||E_T(d_{ij})||_2$
▷ Uniform weights $w_i = 1/K$
$\rho \leftarrow \mathbf{0}; \quad w_i \leftarrow \mathrm{softmax}(\rho)$
for $j \in 1, \ldots, C$ **do**
    ▷ Class queries as average class descriptor encodings
    $q_j \leftarrow \sum_{i=1}^{K} w_i e_{ij}^{(d)}$
    ▷ Cosine similarity between $e^{(x)}$ and class query q_j
    $s_j \leftarrow e^{(x)} \cdot q_j$
end for
▷ Compute gradient (Section 3.3)
$g \leftarrow \nabla_\rho \log \sum_{j=1}^{C} \exp(s_j)$
▷ Determine stepsize (Section 3.4)
$\alpha \leftarrow \mathrm{BISECT}(\mathrm{sm\_entropy}(\alpha \cdot g) - \beta \log_2 K, 0, 10^{10})$
▷ Update ρ with one gradient ascent step and step size α
$\rho \leftarrow \alpha \cdot g; \quad w_i \leftarrow \mathrm{softmax}(\rho)$
for $j \in 1, \ldots, C$ **do**
    ▷ Class queries as average class descriptor encodings
    $q_j \leftarrow \sum_{i=1}^{K} w_i e_{ij}^{(d)}$
    ▷ Cosine similarity between $e^{(x)}$ and class query q_j
    $s_j \leftarrow e^{(x)} \cdot q_j$
end for
▷ Assign x to class $c_{j^\star}$ with maximum similarity
$j^\star \leftarrow \arg\max_j s_j$

3.3 Closed-form Computation of Gradient

While $\nabla_\rho \,\mathrm{logsumexp}(s)$ can be easily computed using automatic differentiation, we note that there can be runtime environments for inference such as on edge devices where running automatic differentiation is undesirable. For such cases, the gradient $\nabla_\rho \,\mathrm{logsumexp}_j(s_j)$ can also be computed in closed-form: $\left(\nabla_\rho \,\mathrm{logsumexp}_j(s_j)\right)_i = \sum_{k=1}^{K}(\sum_{j=1}^{C} \mathrm{softmax}(s)_j \cdot e_{ij}^{(xd)}) \cdot w_i(\delta_{ik} - w_k)$, with δ_{ij} being the Kronecker delta function with $\delta_{ii} = 1$ and $\delta_{ij} = 0$ for $i \neq j$.

3.4 Auto-Tuning the Step Size

The only free hyperparameter of AutoCLIP is the step size α. We note that in a zero-shot setting, there is by definition no labeled data on which such free hyperparameters can be tuned. Because of this, free hyperparameters need to be selected globally in a dataset-independent manner. However, a global choice for the step size α is problematic since the scale of the gradient $g = \nabla_\rho \,\mathrm{logsumexp}(s)$ depends on the dataset, and the step size would have to be adapted accordingly. We address this by proposing a different parameterization in which the free hyperparameter is easily interpreted and the step size α is a derived quantity. Specifically, we control the entropy of the query weights w, $\mathrm{entropy}(w) = -\sum_{i=1}^{K} w_i \log_2 w_i$. The standard, uniform weights have maximum entropy $\log_2 K$ and we set the target entropy to $\beta \cdot \log_2 K$, where the entropy reduction factor $\beta \in [0, 1]$ is the new free hyperparameter that we set globally to $\beta = 0.85$. Intuitively, $\beta \to 1$ corresponds to more equally weighted prompt templates while $\beta \to 0$ to selecting the prompt template with maximum similarity. We present an ablation of the effect of β's choice on AutoCLIP in Figure 4.

	CLIP RN50	CLIP ViT-B-32	CLIP ViT-B-16	CLIP ViT-L-14	DataComp ViT-L-14	CoCa ViT-L-14
CUB200	47.75 (+0.5)	52.84 (+0.7)	57.12 (+1.3)	64.43 (+0.7)	84.79 (+0.8)	73.90 (+0.6)
EuroSAT	34.95 (-1.2)	46.16 (-0.7)	55.93 (+1.4)	55.09 (+0.6)	65.09 (+1.8)	54.77 (-0.4)
Food101	80.26 (+1.4)	84.13 (+1.3)	88.85 (+0.9)	93.71 (+0.4)	94.52 (+0.3)	90.46 (+0.4)
Oxford Pets	83.09 (+2.6)	85.63 (+2.9)	85.89 (+1.9)	91.64 (+0.9)	92.82 (+0.9)	92.03 (+1.2)
ImageNet	60.42 (+0.6)	63.80 (+0.6)	68.70 (+0.5)	75.89 (+0.3)	79.07 (+0.0)	75.63 (+0.2)
ImageNetV2	53.44 (+0.4)	56.49 (+0.8)	62.54 (+0.6)	70.17 (+0.4)	72.21 (+0.2)	68.08 (+0.1)
ImageNetR	29.32 (+0.9)	51.04 (+1.0)	59.13 (+1.0)	73.98 (+0.4)	78.85 (+0.6)	75.59 (+0.8)

Table 1: Accuracy of AutoCLIP (and Δ Accuracy to baseline zero-shot classifier in parenthesis) for $K = 100$ WaffleCLIP prompt templates across models and datasets, averaged over 7 runs.

With sm_entropy$(\alpha \cdot g)$ denoting the entropy of the weights $w = \text{softmax}(\alpha \cdot g)$, selecting the step size α is now equivalent to solving for $f(\alpha) = 0$ for $f(\alpha) = \text{sm\_entropy}(\alpha \cdot g) - \beta \cdot \log_2 K$. As sm_entropy$(\alpha \cdot g)$ monotonically decreases with α, we use bisection on $\alpha \in [0, 10^{10}]$ for finding α with $f(\alpha) \approx 0$. We note that sm_entropy$(0 \cdot g) = \log_2 K$ and thus $f(0) > 0$ for all $\beta < 1$; similarly, sm_entropy$(\alpha \cdot g) \approx 0$ for $\alpha = 10^{10}$ in all settings we considered and thus $f(10^{10}) < 0$ for all $\beta > 0$, which together satisfies the prerequisites for running bisection. The additional bisection has little overhead compared to the cost of encoding the image x with E_x (see Section A.1 in the appendix for details).

4 Experiments

Experimental Setting In this section, we compare AutoCLIP to standard zero-shot classifiers on a wide range of zero-shot image classification benchmarks and a variety of settings. We conduct experiments on the datasets CUB200 (Welinder et al., 2010), EuroSAT (Helber et al., 2019), Food101 (Bossard et al., 2014), Oxford Pets (Parkhi et al., 2012), ImageNet (Russakovsky et al., 2015), ImageNetV2 (Kornblith et al., 2019), ImageNet-R (Hendrycks et al., 2021), and ImageNet-C (Hendrycks & Dietterich, 2019). We study six different vision-language models: from CLIP (Radford et al., 2021), we use ResNet-50 (RN50) (He et al., 2015) and vision transformer (ViT-B/32, ViT-B/16, and ViT-L/14) model variants (Dosovitskiy et al., 2021). Moreover, we use the ViT-L/14 model variant from DataComp (Gadre et al., 2023) and the one trained with CoCa (Yu et al., 2022).

Additionally, we study three ways of generating prompt templates: 1) using the 80 manually designed templates from Radford et al. (2021) (CLIP), 2) templates based on querying a large-language model (DCLIP) (Menon & Vondrick, 2022), and 3) templates that append random words or characters to predefined prompt templates (WaffleCLIP) (Roth et al., 2023). We vary the number of templates from $K = 4$ to $K = 500$; if there is a fixed number of templates available such as in CLIP/DCLIP, templates are sampled with replacement. To account for randomness in the template construction/sampling, we report results averaged over 7 runs. We base our implementation on `https://github.com/ExplainableML/WaffleCLIP` from Roth et al. (2023) and highly appreciate their code release under a permissible license. We report the difference of accuracy of AutoCLIP compared to the baseline zero-shot classifier with uniform prompt template weights ("Δ Accuracy"). Absolute performance across different datasets and VLMs is shown in Table 1 (and in Table 2 and Table 3 in the appendix).

Results We present the main results in Figure 2. Overall, the figure contains 990 different combinations comparing AutoCLIP with the baseline; AutoCLIP is better in 840 cases ($\approx 85\%$) and on average it is better by 0.45 percent point accuracy. We also observe a trend that for larger number of prompt templates K, the advantage of AutoCLIP (Δ Accuracy averaged across datasets, models and CLIP/DCLIP/WaffleCLIP) increases: from $\Delta = 0.06$ for $K = 4$ over $\Delta = 0.33$ for $K = 10$ and $\Delta = 0.49$ for $K = 50$ to $\Delta = 0.57$ for $K = 200$. When aggregating over models, datasets and number of prompt templates, AutoCLIP achieves the largest average improvement for WaffleCLIP ($\Delta = 0.61$), but still improves for CLIP ($\Delta = 0.40$) and

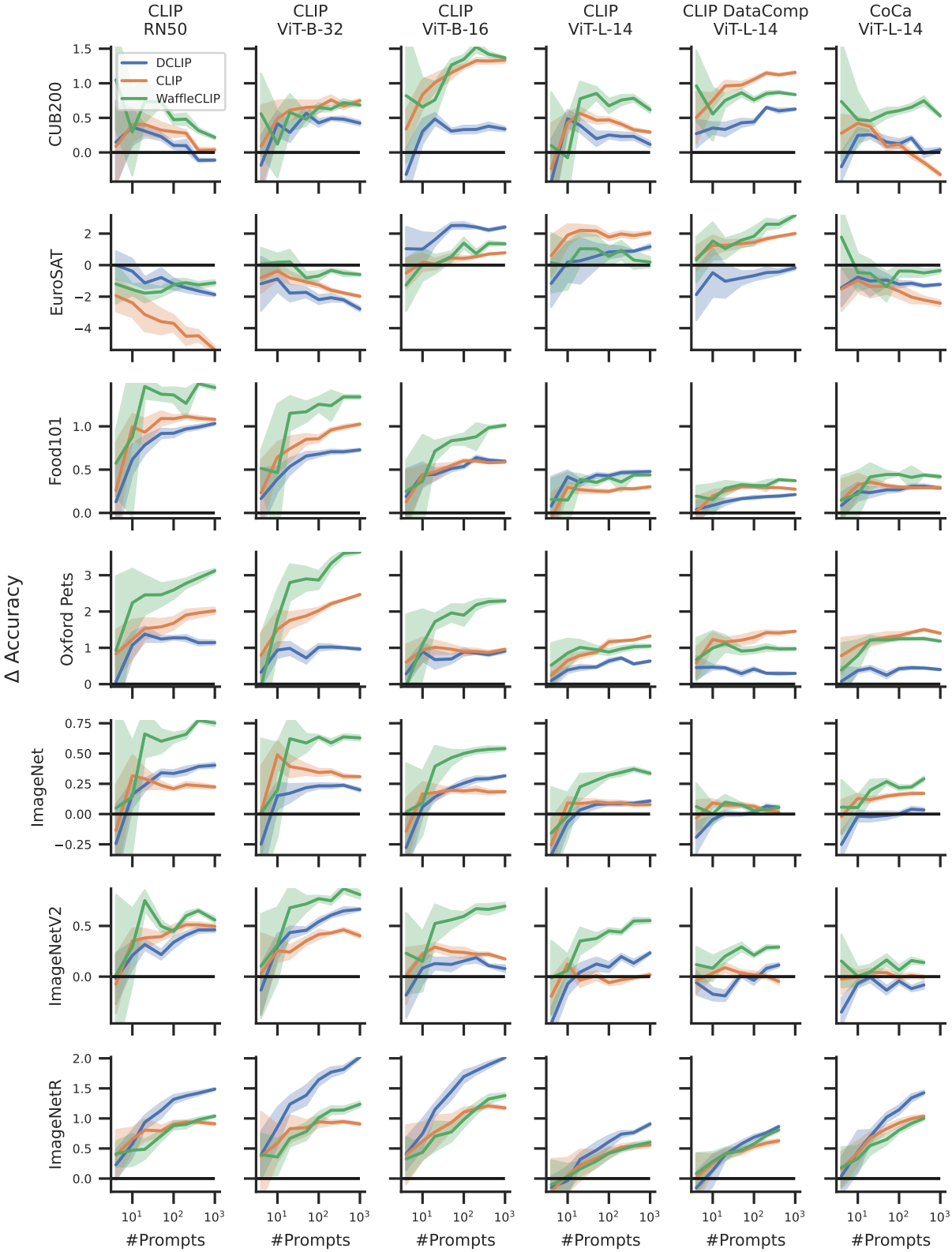

Figure 2: Accuracy improvement (Δ Accuracy) of AutoCLIP over baseline zero-shot classifier across models, datasets, and prompt ensembles. Shown are mean and standard error over 7 runs.

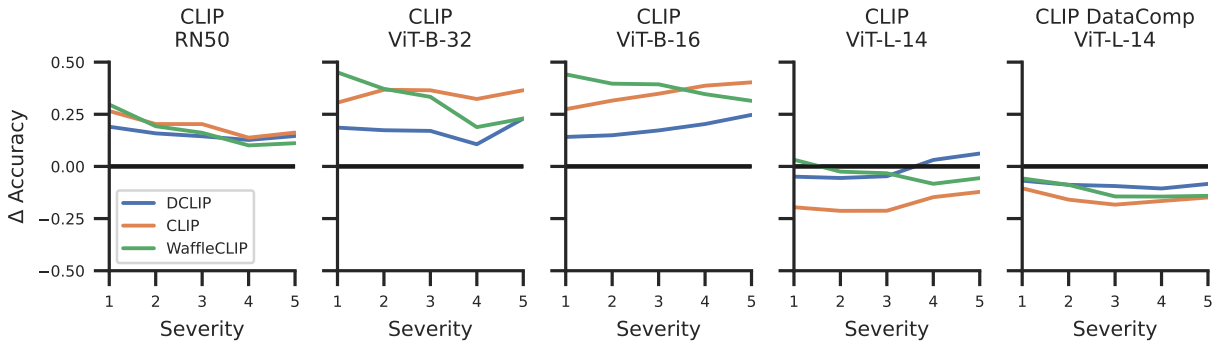

Figure 3: ImageNet-C accuracy improvement (Δ Accuracy) of AutoCLIP over baseline zero-shot classifier for $K = 100$ across models, corruption severity and prompt ensembles, averaged over corruptions and 7 runs.

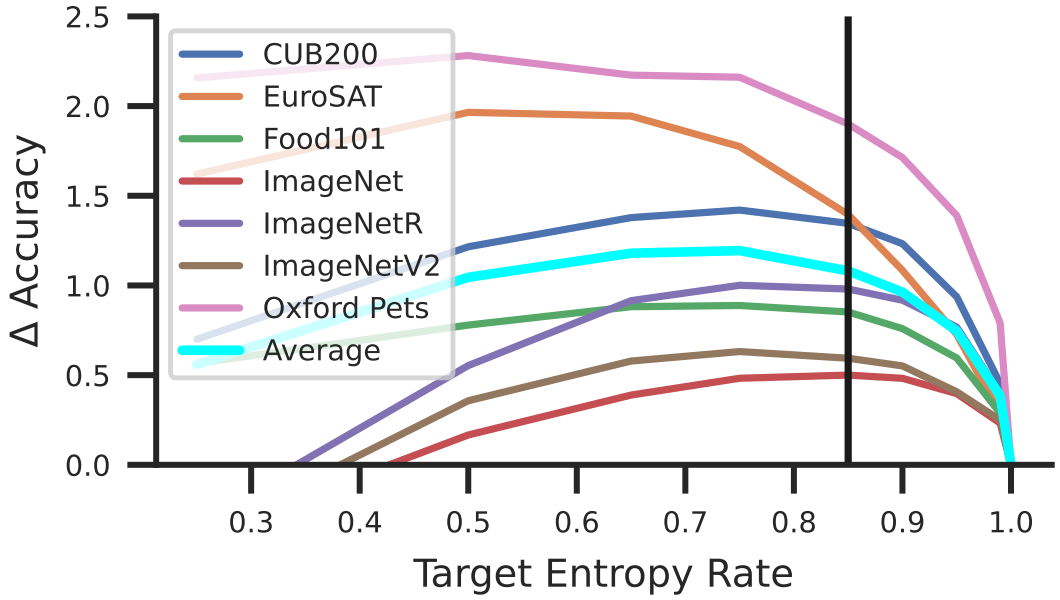

Figure 4: Ablation on target entropy rate β. Shown is the accuracy improvement (Δ Accuracy) of AutoCLIP over baseline zero-shot classifier for a CLIP ViT-B-16, and 100 WaffleCLIP prompt templates, averaged over 7 runs.

DCLIP ($\Delta = 0.29$). Taken together, the findings indicate that AutoCLIP benefits from larger (increased K) and more diverse (WaffleCLIP) sets of prompt templates.

When comparing different vision-language models, AutoCLIP brings the biggest benefit for CLIP ViT-B-16 ($\Delta = 0.68$) and the smallest one for CoCa ViT-L-14 ($\Delta = 0.19$), with all other models having average Δ between 0.36 and 0.52. Comparing different datasets, AutoCLIP performs strongest on Oxford Pets ($\Delta = 1.15$) and worst on EuroSAT ($\Delta = -0.24$); we hypothesize that this is because EuroSAT is in general a challenging dataset for CLIP on which the image encoder produces embeddings that are not very informative about image properties, which deteriorates the prompt weight selection as it becomes harder to decide which prompt describes an image of interest well. We note that EuroSAT is the only setting on which AutoCLIP hurts performance on average; on all other datasets, AutoCLIP improves performance: Δ(CUB200) = 0.5, Δ(Food101) = 0.52, Δ(ImageNet) = 0.17, Δ(ImageNetV2) = 0.2, and Δ(ImageNetR) = 0.71. While these

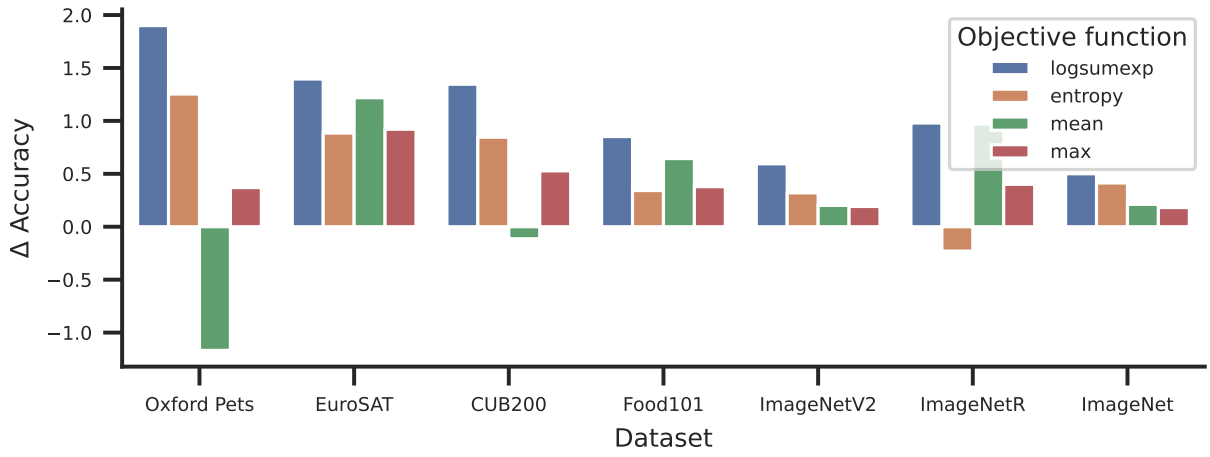

Figure 5: Comparison of different objective functions for auto-tuning. Shown is the accuracy improvement (Δ Accuracy) of AutoCLIP over baseline zero-shot classifier for a ViT-B-16, and 100 WaffleCLIP prompt templates, averaged over 7 runs.

improvements are of modest magnitude, they essentially come for free by a mere change of the inference procedure.

In Figure 3, we present results on ImageNet-C for WaffleCLIP with $K = 100$ for different severities and averaged across corruptions. AutoCLIP consistently improves performance for the smaller vision-language models (RN50, ViT-B-32, ViT-B-16) and sees a minor drop of performance for the two ViT-L-14 variants. Averaged across all models, corruptions, and severities, AutoCLIP improves performance by $\Delta = 0.11$. We provide plots for each corruption separately for WaffleCLIP prompt templates in the appendix in Figure 9. The biggest average benefit of AutoCLIP is obtained for the low-frequency corruptions "saturate" ($\Delta = 0.22$), "brightness" ($\Delta = 0.22$), and "contrast" ($\Delta = 0.23$); the smallest average benefit for "shot-noise" ($\Delta = 0.05$) and "snow" ($\Delta = 0.06$).

Ablations We ablate AutoCLIP's choice of the target entropy rate β (which defaults to 0.85) and the objective function (defaults to logsumexp). In Figure 4, we observe that AutoCLIP's performance for most datasets does not depend strongly on the specific choice of the target entropy rate β as Δ Accuracy stays relatively constant in the range $\beta \in [0.7, 0.9]$. This is a desirable property as in a zero-shot setting without labeled data, tuning β per dataset would be infeasible. For two datasets (Oxfort Pets and EuroSAT), our default value of $\beta = 0.85$ was suboptimal and a considerably smaller choice of $\beta = 0.7$ would have obtained considerably better results. Also on average, $\beta = 0.7$ performs favorably and we recommend this choice for future work on other datasets and tasks. We provide results for a similar experiment in which we directly control the step size α in Section A.4. Directly controlling α reduces computation overhead further, but optimal choices of step size α vary more strongly across datasets than choices for the target entropy rate β.

We motivated the choice of logsumexp as AutoCLIP's aggregation/objective function in Section 3.2 as striking a good compromise between max and mean aggregation. In Figure 5, we empirically confirm that the logsumexp aggregation performs favorably compared to max/mean aggregation on all datasets. Moreover, it also outperforms entropy aggregation, which is a popular choice for test-time adaptation (Wang et al., 2020; Shu et al., 2022).

In Figure 6, we show the prompt template weights ($K = 30$) obtained by AutoCLIP on 500 Food101 samples. Samples are structured in 10 blocks of 50 samples each, where each block corresponds to one class. Prompt template weights are relatively similar for instances belonging to the same (unknown) class but vary across classes. Some templates like the ones starting with "A tattoo of..." or "A drawing of..." get consistently low weights as the images of the Food101 dataset do not look like tattoos or origami, while templates starting with

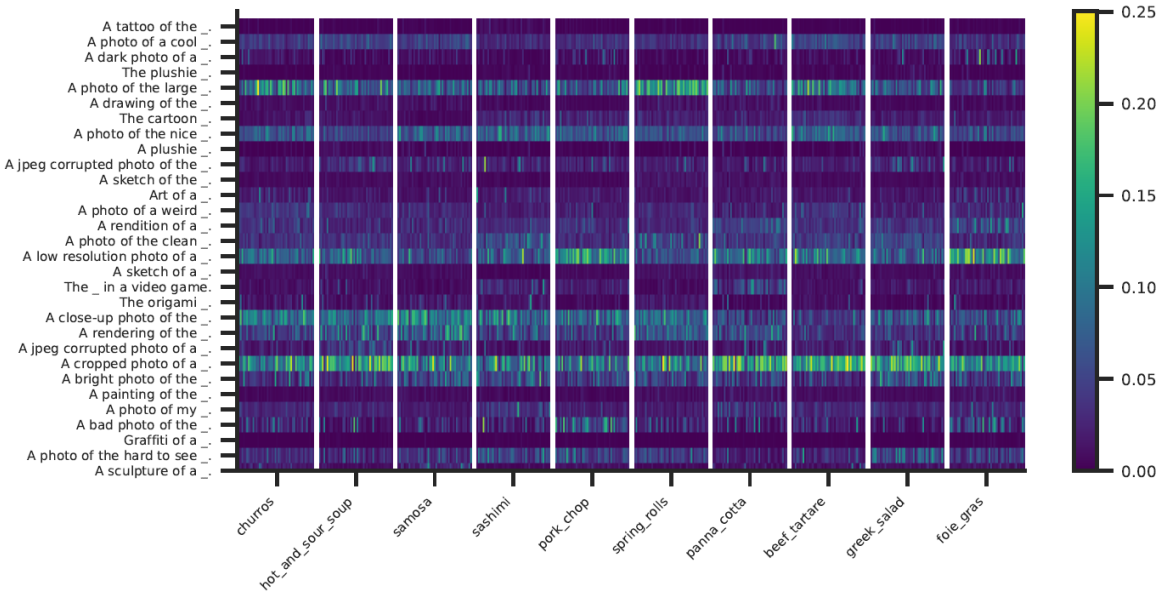

Figure 6: Illustration of prompt template weights w on 500 samples from the Food101 dataset, with blocks of 50 samples belonging to the same (unknown) class. CLIP backbone is a ViT-B-16 and 30 CLIP prompt templates are used.

"A photo of..." tend to get higher weights, as Food101 contains mostly actual photos. Note that the weight distribution looks different on other datasets like ImageNet-R, with higher weights for "artistic" prompts (see Figure 8 in the appendix). Overall, this confirms that AutoCLIP can adapt the zero-shot classifier on the fly to properties of the respective image. Moreover, AutoCLIP provides accuracy improvements with only minor additional inference overhead as discussed in Section A.1 in the appendix.

5 Analysis in a Controlled Setting

We study AutoCLIP in a controlled setting, in which we directly sample embedding vectors in an embedding space corresponding to encoded image and class descriptors, without actually encoding images or text prompts. By this, we can control key properties of the embeddings and study how they influence AutoCLIP's performance. While the setting is strongly simplified, we will nevertheless gain some insights that provide possible explanations for some of the key findings from Section 4.

We set the number of classes to $C = 5$, the number of embedding dimensions to $d = 128$, the number of prompt templates to $K = 10$, and the number of instance to 200. We sample class descriptor embeddings $E_T(d_{ij})$ as follows: let $c_j \sim \mathcal{N}(0, 1, d) \in \mathbb{R}^d$ be C d-dimensional standard normal-distributed class means, let $p_i \sim \mathcal{N}(0, 1, d) \in \mathbb{R}^d$ be K d-dimensional standard normal-distributed prompt embedding means, and $\Psi_{ij} \sim \mathcal{N}(0, 1, d) \in \mathbb{R}^d$ be KC d-dimensional standard normal-distributed prompt-class coupling terms. We then set $E_T(d_{ij}) = (1 - \rho)(c_j + p_i) + \rho\Psi_{ij}$, where $\rho \in [0, 1]$ controls the "entanglement" between class and prompt template embeddings. Intuitively, $\rho = 0$ simulates a setting in which the VLM's text encoder perfectly separates class and prompt template related information such that their combination is additive. Increasing ρ results in a stronger entanglement such that class and prompt template-related information interact more strongly and are no longer additive. We then set the image embeddings corresponding to a $E_T(d_{ij})$ to $E_X(x) = E_T(d_{ij}) + \xi$ with $\xi \sim \mathcal{N}(0, \varepsilon, d)$. Here ε controls the "instance noise", that is: how much the image embeddings are spread around the corresponding class descriptor embedding.

Figure 7 compares the accuracy of mean-, max- and AutoCLIP-aggregation ($\beta = 0.85$) for different values of entanglement ρ and instance noise ε. In general, lower entanglement favors mean aggregation (the standard in CLIP), while higher entanglement favors max-aggregation. Usually, VLM text encoders are

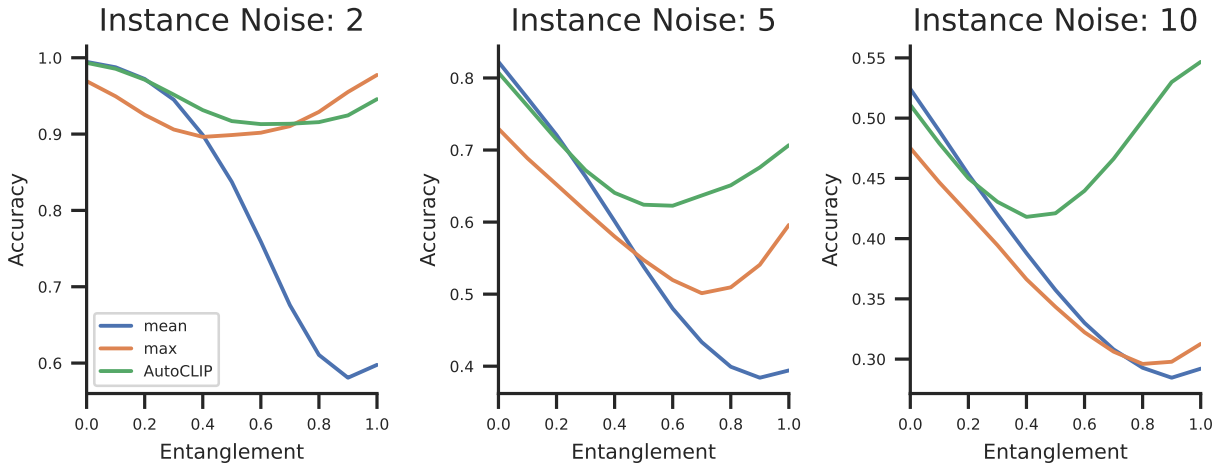

Figure 7: Comparison of AutoCLIP to "mean" aggregation (Line 10 in Algorithm 1) and "max" aggregation ($s_j \leftarrow \max_i e_{ij}^{(d)} \cdot q_j$) in a controlled and simplified setting. "Instance Noise" controls how strongly instance encodings vary from their respective mean. "Entanglement" controls how strongly the text encoding of prompt template and class name are entangled. Shown is mean over 100 random seeds.

good in terms of disentangling concepts, which explains why prior work (Roth et al., 2023) has found max-aggregation to perform inferior compared to mean-aggregation. On the other hand, we observe that AutoCLIP nearly always outperforms max-aggregation (except for the small instance noise and strong entanglement setting), also outperforms mean-aggregation for moderate entanglement ($\rho > 0.4$), and performs very close to mean-aggregation for smaller entanglement.

This provides a possible explanation for the findings from Figure 2: for smaller (and weaker) VLMs, the text embeddings are more entangled and thus AutoCLIP provides stronger benefits compared to standard mean-aggregation. For larger VLMs like ViT-L-14 based ones, the entanglement decreases and thus the benefit of AutoCLIP is smaller. Moreover, for small entanglement and large instance noise, AutoCLIP can also be slightly worse than mean-aggregation in Figure 7. This settings likely corresponds to the ViT-L-14 (low class-prompt entanglement) on ImageNet-C (large instance noise) in Figure 3, where AutoCLIP also performs slightly worse than mean-aggregation. However, in general Figure 7 shows that AutoCLIP provides a favorable trade-off between mean- and max-aggregation for many practical settings.

6 Conclusion

We have proposed AutoCLIP, a method for improving zero-shot classifiers on vision-language models. It automatically tunes per-image weights of prompt templates before aggregating them into class queries. AutoCLIP improves performance over standard zero-shot classifiers on the vast majority of settings, with only minimal inference-time overhead. We believe that due to its simplicity and low cost, AutoCLIP has the potential to be broadly applied in conjunction with vision-language models. For future work, it is exciting to explore if AutoCLIP also benefits other zero-shot tasks built on top of multi-modal modals such as object detection with OWL-ViT (Minderer et al., 2022) or multi-modal prompting with ImageBind (Girdhar et al., 2023a).

References

James Urquhart Allingham, Jie Ren, Michael W. Dusenberry, Xiuye Gu, Yin Cui, Dustin Tran, Jeremiah Zhe Liu, and Balaji Lakshminarayanan. A simple zero-shot prompt weighting technique to improve prompt ensembling in text-image models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023.

Jeremy Atkinson. Car beurre eta. `https://openverse.org/image/5d960316-3209-4ea6-bf4c-458449f9a588?q=Car%20drawing`, 2015. Accessed: 2023-09-27.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pp. 446–461. Springer, 2014.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, Eyal Orgad, Rahim Entezari, Giannis Daras, Sarah Pratt, Vivek Ramanujan, Yonatan Bitton, Kalyani Marathe, Stephen Mussmann, Richard Vencu, Mehdi Cherti, Ranjay Krishna, Pang Wei Koh, Olga Saukh, Alexander Ratner, Shuran Song, Hannaneh Hajishirzi, Ali Farhadi, Romain Beaumont, Sewoong Oh, Alex Dimakis, Jenia Jitsev, Yair Carmon, Vaishaal Shankar, and Ludwig Schmidt. Datacomp: In search of the next generation of multimodal datasets, 2023.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15180–15190, June 2023a.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190, 2023b.

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HkxzxONtDB`.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015. URL `https://api.semanticscholar.org/CorpusID:206594692`.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2661–2671, 2019.

Sachit Menon and Carl Vondrick. Visual classification via description from large language models. *arXiv preprint arXiv:2210.07183*, 2022.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection with vision transformers. *arXiv preprint arXiv:2205.06230*, 2022.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Karsten Roth, Jae Myung Kim, A Koepke, Oriol Vinyals, Cordelia Schmid, and Zeynep Akata. Waffling around for performance: Visual classification with random words and broad concepts. *arXiv preprint arXiv:2306.07282*, 2023.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289, 2022.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yezhou Yang. Test-time adaptation with clip reward for zero-shot generalization in vision-language models. *ArXiv*, abs/2305.18010, 2023. URL `https://api.semanticscholar.org/CorpusID:258959424`.

A Appendix

A.1 Inference time overhead of AutoCLIP

In this paragraph, we provide some measurements on inference time overhead by AutoCLIP. We provide numbers for the case of a ViT-L-14 on the Oxford Pets dataset. Here, encoding an image takes 12.64ms on a V100 (minimum over 100 images). The baseline "averaging" zero-shot classifiers takes additional 0.08ms (average over 640 samples) on top to classify a sample. AutoCLIP takes additional 1.54ms (average over 640 samples) for classification when running bisection for autotuning the step size. For a fixed step size, the overhead of AutoCLIP is 0.45ms. Thus, AutoCLIP with autotuning raises inference time from 12.64ms to 14.18ms. In contrast, TPT (Shu et al., 2022) and RLCF (Zhao et al., 2023), which did not report compute or memory requirements, require encoding multiple image augmentations. TPT states "We augment a single test image 63 times using random resized crops and construct a batch of 64 images, including the original one.", which means that the image encoding time (for instance the 12.64ms from above) is increased by a factor of 64x, plus additional overhead for backpropagating through the text encoder, which likely brings the inference time per sample close to 1s (or more if multiple test-time adaptation steps are conducted). We note that for bisection, we use an independent call to `scipy.optimize.bisect` (Virtanen et al., 2020) (maxiter=100, xtol=1e-2, rtol=1e-2). A batched variant of bisection could speed-up many workloads.

A.2 Additional Experimental results

We present additional experimental results. Table 2 and Table 3 show the absolute performance of AutoCLIP on different datasets and VLMs for DCLIP and CLIP prompt templates, respectively, similar to Table 1 in the main paper for WaffleCLIP templates. Figure 8 illustrates prompt weights on the ImageNetR dataset. Figure 9 contains results of AutoCLIP in terms of Δ Accuracy on ImageNetC for every corruption seperately.

In Figure 10, we show an additional comparison of AutoCLIP to a stronger baseline which is based on TopR aggregation. In this TopR aggregation, for each image R prompt templates are selected whose resulting encoded class descriptors have maximum average cosine similarity to the encoded image. We note that choosing R is non-trivial in a zero-shot setting due to the lack of labelled validation data. In the figure, we compare AutoCLIP against this TopR-CLIP for $K = 100$ DCLIP prompt template, across the same VLMs and datasets as in Figure 2. We provide results for different choices of R: overall, for the best choice of $R = 20$, AutoCLIP is better on 86% of the cases and by 0.40 percent point accuracy on average.

A.3 Motivation of logsumexp in AutoCLIP

As discussed in Section 3.2, AutoCLIP determines weights w_i such that $\text{logsumexp}_j(s_j) = \text{logsumexp}_j(\sum_{i=1}^{K} w_i e_{ij}^{(xd)}) = \text{logsumexp}_j(\text{softmax}(\rho) \cdot e_{:j}^{(xd)})$ gets increased by one step of gradient ascent in the direction of $\nabla_\rho \text{logsumexp}_j(\text{softmax}(\rho) \cdot e_{:j}^{(xd)})$. Empirically, we find in Figure 5 that this logsumexp objective function outperforms alternative loss functions such as entropy, mean, or max. On the one hand,

	CLIP RN50	CLIP ViT-B-32	CLIP ViT-B-16	CLIP ViT-L-14	DataComp ViT-L-14	CoCa ViT-L-14
CUB200	47.75 (+0.1)	53.00 (+0.4)	57.82 (+0.3)	64.57 (+0.3)	85.38 (+0.4)	73.69 (+0.1)
EuroSAT	36.39 (-1.2)	45.88 (-2.2)	59.22 (+2.5)	57.89 (+0.8)	60.08 (-0.7)	57.15 (-1.2)
Food101	79.12 (+0.9)	83.43 (+0.7)	88.53 (+0.5)	93.14 (+0.4)	93.89 (+0.2)	89.77 (+0.3)
Oxford Pets	85.92 (+1.3)	87.11 (+1.0)	88.53 (+0.9)	94.08 (+0.6)	94.00 (+0.4)	93.54 (+0.4)
ImageNet	60.62 (+0.3)	63.89 (+0.2)	69.10 (+0.3)	75.92 (+0.1)	79.02 (+0.0)	75.41 (+0.0)
ImageNetV2	53.60 (+0.3)	56.73 (+0.5)	62.22 (+0.2)	70.01 (+0.1)	71.95 (-0.0)	67.91 (-0.0)
ImageNetR	28.14 (+1.3)	49.51 (+1.6)	58.37 (+1.7)	73.12 (+0.6)	78.06 (+0.7)	73.73 (+1.1)

Table 2: Accuracy of AutoCLIP (and Δ Accuracy to baseline zero-shot classifier in parenthesis) for $K = 100$ DCLIP prompt templates across models and datasets, averaged over 7 runs.

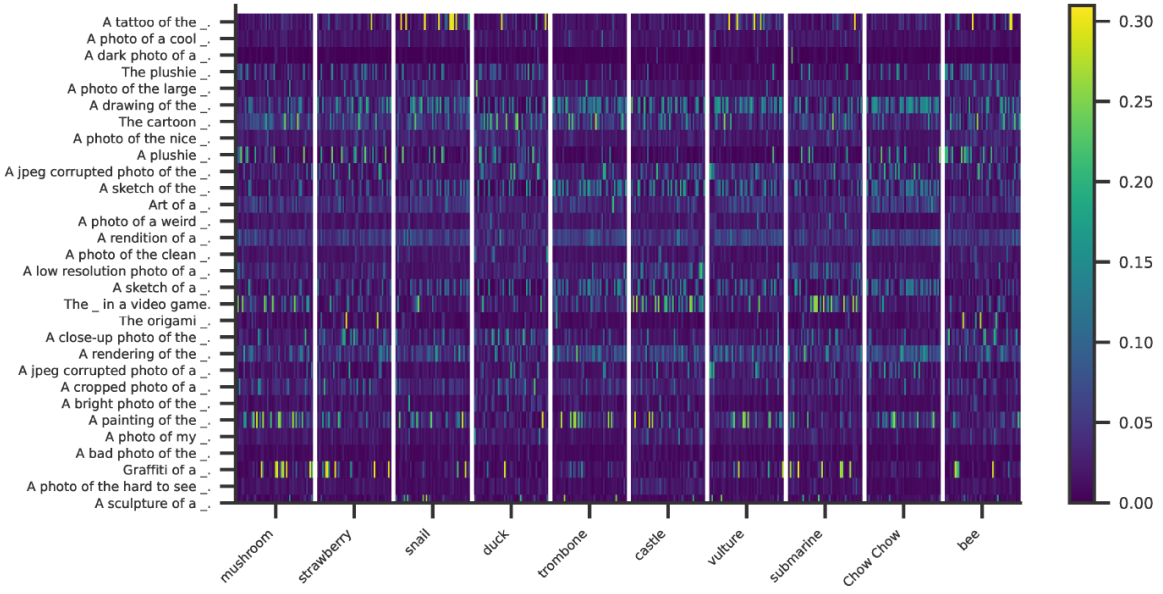

Figure 8: Illustration of prompt template weights w on 500 samples from the ImageNetR dataset, with blocks of 50 samples belonging to the same (unknown) class. CLIP backbone is a ViT-B-16 and 30 CLIP prompt templates are used.

we attribute this to logsumexp being "in-between" mean and max as as a "smooth maximum". On the other hand, we hypothesize that logsumexp is a reasonable choice because its relationship to the energy function of a data point — in that view AutoCLIP can be interpreted as minimizing the energy and maximizing the probability density $p(x)$ of x under the zero-shot classifier. We elaborate this relationship in more detail in the following paragraph.

According to (Grathwohl et al., 2020), for a softmax-based classifier that assigns logits $f_\theta(\mathbf{x})$ to class-probablities via $p(y|\mathbf{x}) = \text{softmax}(f(\mathbf{x}))$, one can slightly re-interpret the logits obtained from f_θ to define $p(\mathbf{x}, y)$ and $p(\mathbf{x})$. For this, one re-uses the logits to define an energy-based model of the joint distribution of data point $\mathbf{x}$ and labels y via $p_\theta(\mathbf{x}, y) = \frac{\exp(f_\theta(\mathbf{x})_{[y]})}{Z(\theta)}$, where $Z(\theta)$ is the unknown normalizing constant and $E_\theta(\mathbf{x}, y) = -f_\theta(\mathbf{x})_{[y]}$ is the energy. Moreover, $p_\theta(\mathbf{x}) = \sum_y p_\theta(\mathbf{x}, y) = \frac{\sum_y \exp(f_\theta(\mathbf{x})_{[y]})}{Z(\theta)}$, and accordingly the energy corresponds to $E_\theta(\mathbf{x}) = -\log \sum_y \exp(f_\theta(\mathbf{x})_{[y]})$.

	CLIP RN50	CLIP ViT-B-32	CLIP ViT-B-16	CLIP ViT-L-14	DataComp ViT-L-14	CoCa ViT-L-14
CUB200	47.00 (+0.3)	52.36 (+0.7)	56.99 (+1.2)	63.94 (+0.5)	85.52 (+1.1)	73.99 (+0.1)
EuroSAT	32.28 (-3.7)	44.78 (-1.3)	56.76 (+0.4)	52.96 (+1.8)	61.94 (+1.4)	51.58 (-1.7)
Food101	79.69 (+1.1)	83.64 (+0.9)	88.83 (+0.6)	93.33 (+0.2)	94.55 (+0.3)	90.36 (+0.3)
Oxford Pets	84.30 (+1.7)	85.20 (+2.0)	88.42 (+0.9)	93.24 (+1.2)	93.79 (+1.3)	92.67 (+1.3)
ImageNet	59.90 (+0.2)	63.31 (+0.3)	68.43 (+0.2)	75.38 (+0.1)	79.29 (+0.1)	75.79 (+0.2)
ImageNetV2	52.98 (+0.5)	56.00 (+0.4)	62.12 (+0.2)	69.56 (-0.1)	72.09 (+0.0)	67.90 (-0.0)
ImageNetR	27.11 (+0.9)	47.74 (+0.9)	56.28 (+1.1)	71.30 (+0.4)	78.26 (+0.5)	74.51 (+0.9)

Table 3: Accuracy of AutoCLIP (and Δ Accuracy to baseline zero-shot classifier in parenthesis) for $K = 100$ CLIP prompt templates across models and datasets, averaged over 7 runs.

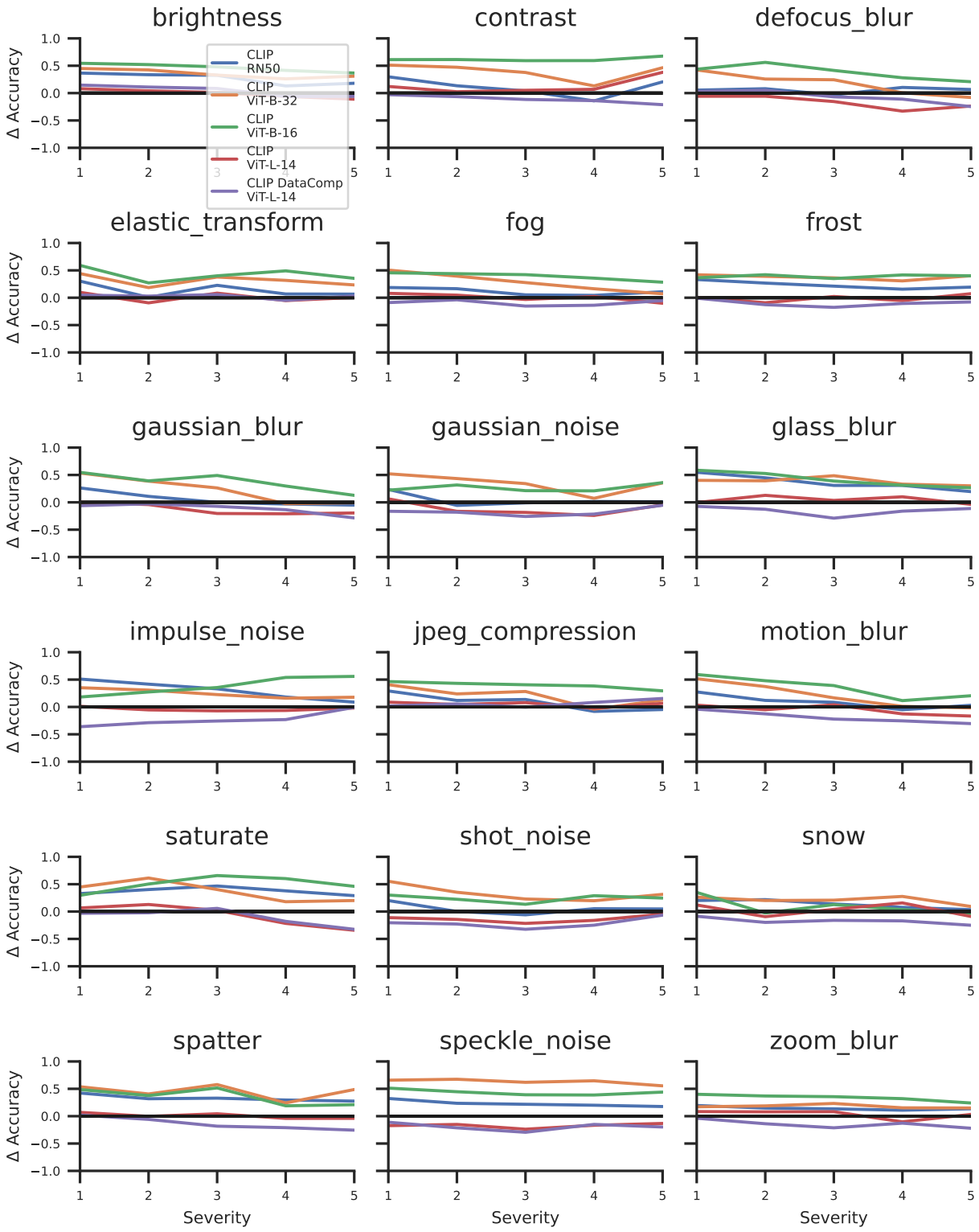

Figure 9: ImageNetC Accuracy improvement (Δ Accuracy) of AutoCLIP over baseline zero-shot classifier for WaffleCLIP across models, corruptions, averaged over 7 runs.

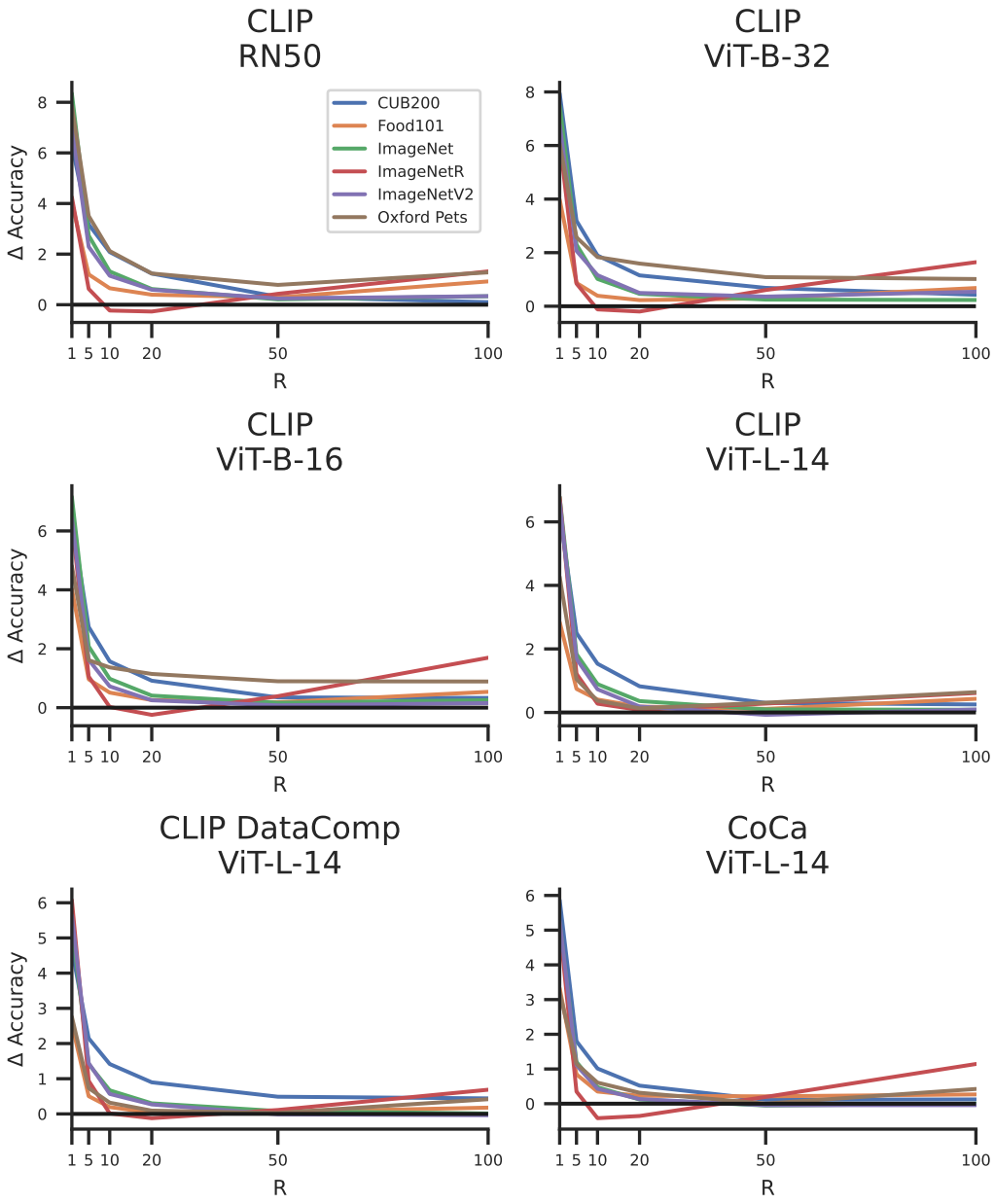

Figure 10: Accuracy improvement (Δ Accuracy) of AutoCLIP with $K = 100$ DCLIP prompt templates over TopR zero-shot classifier with different values of R across models, averaged over datasets and 7 runs.

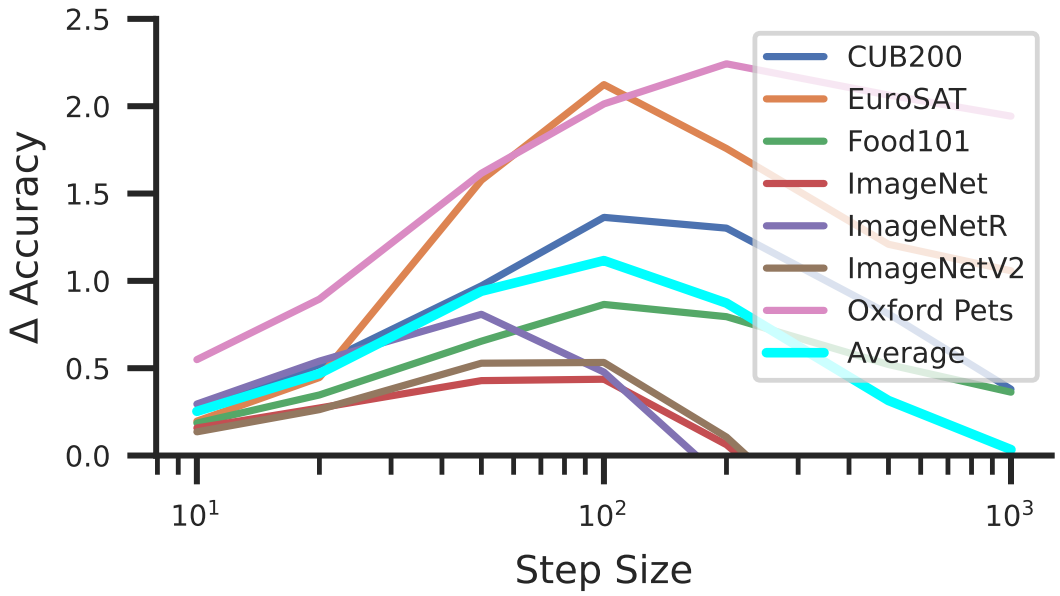

Figure 11: Control experiment when directly choosing a fixed step size α (and not a target entropy rate β). Shown is the accuracy improvement (Δ Accuracy) of AutoCLIP over a baseline zero-shot classifier for a CLIP ViT-B-16, and 100 WaffleCLIP prompt templates, averaged over 7 runs.

In the case of AutoCLIP, the logits $f_\theta(\mathbf{x})$ correspond to the cosine similarities s_j (with optional temperature scaling). Accordingly, $-\operatorname{logsumexp}_j(s_j)$ can be interpreted as the energy, and AutoCLIP, which maximizes $\operatorname{logsumexp}_j(s_j)$, can be interpreted as a method for minimizing the energy of the zero-shot classifier.

We note that this interpretation works best for classifiers that were trained with an energy-based loss term that encourages low energy on the data manifold and high energy elsewhere as discussed by (Grathwohl et al., 2020). However, empirically we find it to also work well on CLIP-based zero-shot classifiers.

A.4 Step Size Selection in AutoCLIP

AutoCLIP controls the target entropy rate β as free hyperparameter as an alternative parameterization to directly controlling the step size α. The reason for this is that optimal choices of step size α vary more strongly across datasets than choices for the target entropy rate β. We show this in Figure 11 (compared to Figure 4): while the reparametrization from α to β does not change the peak performance of AutoCLIP on a dataset much, it affects the alignments of curves across datasets. We observe that the curves are more aligned when varying β than when varying α. Quantitatively, the mean pairwise Pearson product-moment correlation coefficient between the curves for β in Figure 4 is 0.66, while it is only 0.45 for the curves for α in Figure 11. Thus, in a zero-shot setting in which hyperparameters cannot be tuned on a per-dataset basis, the more aligned behavior across datasets for target entropy rate β is preferable to the the step size α.