SMOOTHOPERATOR: ENHANCING REPRESENTATIONS FOR FINE-GRAINED OPEN-SET RECOGNITION VIA MODULATED LABEL SMOOTHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Open Set Recognition (OSR) aims to enable models to accurately classify known classes while rejecting samples from unseen classes. A key challenge in OSR lies in the inability to model the unbounded distribution of unknown classes during training, often leading to the misclassification of samples from these classes. Rather than modeling unknowns, recent work shapes the feature space so that known classes are compact and well separated, and spherical representation learning methods have achieved strong results this way. Label smoothing has been identified as one of the key drivers of this success, yet it applies the same coefficient to every training sample, regardless of how well each sample is already embedded. We show that the spherical representation learning objectives used in OSR share a single alignment–uniformity structure in which labels enter only through the alignment term. Label smoothing therefore acts as an alignment dial, and a fixed coefficient sets this dial to the same value for every sample. We propose a plug-in, SmoothOperator (SmoothOP), which sets the smoothing coefficient of each sample from its **prominence**, an embedding-space signal measuring how clearly the sample's own class stands out against its strongest competing class. Our method integrates into four existing spherical representation learning methods at minimal training overhead. SmoothOP assigns strong smoothing to samples with high prominence, which reduces their alignment and relaxes their pull. Conversely, samples with low prominence receive reduced smoothing, which translates to increased alignment, pulling them to the class center. Empirically, SmoothOP yields more compact class clusters and, despite modifying only the alignment term, better dispersed ones, which together improve OSR performance. On the Semantic Shift Benchmark, SmoothOP-augmented variants generally outperform their base objectives across datasets, degrees of semantic shift, and OSR post-processors, with gains of up to 4.7% in AUROC, OSCR, and closed-set accuracy.

1 INTRODUCTION

Traditional machine learning classifiers typically operate under a closed-set assumption, where the model assumes all test samples belong to one of the classes encountered during training. In real-world scenarios, however, this assumption often fails as models frequently encounter "unknown" classes not present in the initial training set. For instance, a diagnostic tool trained to recognize five specific skin pathologies might erroneously force a sixth, unseen condition into one of its existing categories. Open Set Recognition (OSR) mitigates this risk by enabling systems to correctly identify and label samples from novel classes as "unknown" while preserving high classification performance on the original known data (Scheirer et al., 2013).

The main challenge in OSR is that, by definition, unknown classes are not available during training; yet, the model must still reject them during inference. To tackle this problem, earlier OSR literature has followed two main approaches. One approach is to explicitly represent or synthesize the unknown (Bendale & Boult, 2016; Zhou et al., 2021; Chen et al., 2020a; 2022; Neal et al., 2018; Kong & Ramanan, 2022). Another approach models the known classes with reconstruction-based or generative models (Oza & Patel, 2019; Sun et al., 2020; Huang et al., 2023) (Sec. 2). However, Vaze et al. (2022) showed that standard closed-set classifiers can match or exceed dedicated OSR methods

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

when their closed-set performance is optimized. These results stem from the training procedure rather than from any OSR-specific component, suggesting that the learned feature representation plays a central role in open-set performance (Dietterich & Guyer, 2022; Sun et al., 2022; Bahavan et al., 2026). In terms of Scheirer et al. (2013)'s open-set risk framework, this translates to feature geometry: known classes should be well separated and occupy a compact volume in the feature space, thereby reserving the remaining regions for unknowns.

Building on this, recent OSR methods explicitly shape the feature geometry through representation learning. A model is first trained with a spherical representation learning objective, either tailored to OSR (Xu et al., 2023; Bahavan et al., 2026) or general-purpose (Khosla et al., 2020), and a classifier is then fine-tuned on top of the learned representation for classification and unknown rejection. Label smoothing has been identified as one of the performance drivers during the representation learning stage (Bahavan et al., 2026; Xu et al., 2023), yet it applies a single fixed coefficient to every training sample. To understand how smoothing acts on these objectives, we decompose them into two terms (Wang & Isola, 2020): an *alignment* term, which pulls each sample toward its own class and makes classes compact, and a *uniformity* term, which spreads samples across the hypersphere and keeps classes apart. We find that these objectives share a common structure: the uniformity term does not depend on the ground-truth labels, and the labels only determine how strongly each sample is pulled toward each class within the alignment term (Prop. 1). Since label smoothing modifies only the labels, it acts purely on alignment. Increasing the smoothing coefficient shifts part of this alignment pull from its own class to the other classes, weakening its alignment while leaving the uniformity term unchanged (Prop. 2). Label smoothing is thus an alignment dial, and a fixed coefficient sets this dial to the same value for every sample.

This motivates SmoothOP, a plug-in that sets the smoothing coefficient for each sample individually. The coefficient is controlled by prominence, an embedding-space signal measuring how clearly a sample's own class stands out against its strongest competing class. Samples with high prominence are already clearly separated from competing classes, so they receive strong smoothing, which reduces their alignment and helps prevent overfitting. Samples with low prominence lie close to a competing class and receive weak smoothing, which strengthens their alignment and pulls them toward their class center. SmoothOP integrates into existing spherical representation learning methods with minimal training cost and yields class clusters that are more compact and better dispersed.

In this work, we make the following contributions:

- We show that the contrastive and spherical objectives used in current OSR methods share a single alignment–uniformity decomposition in which labels enter only through the alignment term, and that label smoothing modifies that term alone.
- Based on those formal arguments, we propose SmoothOP, a plug-in that sets the smoothing coefficient for each sample from from prominence, which measures how clearly the sample's own class stands out against its strongest competing class. SmoothOP integrates into four existing spherical representation learning objectives at minimal training cost.
- We empirically validate that our plug-in yields class clusters that are more compact and better dispersed.
- On the Semantic Shift Benchmark (Vaze et al., 2022), our SmoothOP-augmented variants generally outperform their base objectives across different datasets, degrees of semantic shift, and OSR post-processors, with gains of up to 4.7% in AUROC, OSCR, and accuracy.

2 Related Work

The literature on Open-Set Recognition (OSR) broadly distinguishes between two families of methods: discriminative and generative (Geng et al., 2020). Discriminative methods aim to enhance classifiers to correctly identify known classes while simultaneously flagging inputs from unknown classes; representative examples include OpenMax (Bendale & Boult, 2016) and PROSER (Zhou et al., 2021). A related line of discriminative work uses class prototypes as anchors for decision boundaries (Yang et al., 2022), while others define reciprocal points exclusion zones in feature space that bound known class regions to tighten the open-space boundary (Chen et al., 2020a; 2022). A shared weakness of these approaches is their difficulty characterizing the full extent of open space, compounded by their reliance on Euclidean geometry (Wang et al., 2024) (discussed further in

Sec 3). Generative approaches instead learn the data distribution or manifold structure and detect unknowns either through synthetic sample generation (Neal et al., 2018; Kong & Ramanan, 2022) or via reconstruction error (Oza & Patel, 2019; Huang et al., 2023; Sun et al., 2020). Some methods blend both paradigms (Perera et al., 2020). Despite their effectiveness, generative and hybrid methods carry a high computational cost and can be unreliable when out-of-distribution samples bear close resemblance to known classes (Vaze et al., 2022; Huang et al., 2023). More recently, representation learning has emerged as a promising direction for OSR (Xu et al., 2023). Vaze et al. (Vaze et al., 2022) demonstrated that well-calibrated logit-based classifiers can rival dedicated OSR systems, attributing OSR gains primarily to stronger closed-set architectures and training procedures rather than OSR-specific design choices.

Semantic Shift vs Covariate Shift Covariate shift changes the input distribution while keeping label semantics fixed. Semantic shift, the setting of OSR, changes the label space through novel classes while the input domain stays largely unchanged (formal definitions in Appendix B.3). To isolate the effects of semantic shift from low-level covariate noise, the Semantic Shift Benchmark (SSB) (Vaze et al., 2022) was proposed(Vaze et al., 2022). In these settings, unknown classes share significant visual attributes with known classes, requiring the model to learn highly precise semantic boundaries.

Post-processors In addition to training time-based methods, there are new OSR methods which can be integrated after the model is trained, which mainly affect changes in the score functions(Cruz et al., 2024; Park et al., 2023; Sun et al., 2022). This is elaborated in Appendix D.3. While these methods are suitable for improving the model's ability to separate known from unknown data, they don't improve the model's accuracy or classification performance.

Fine-grained OSR Most OSR benchmarks involve coarse-grained category shifts, leaving the fine-grained regime comparatively understudied (Lang et al., 2024; Vaze et al., 2022). Lang et al. (Lang et al., 2024) address this gap by introducing a hierarchical adversarial training objective for fine-grained OSR; however, their method requires access to structured label hierarchies such as taxonomic annotations, restricting its applicability to domains where such metadata is readily available.

3 Problem Formulation and Evaluation Metrics

To begin, we define a supervised training set for a classification task as $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \subset \mathcal{X} \times \mathcal{C}$, where $\mathcal{X}$ is the image space and $\mathcal{C} = \{1, \ldots, C\}$, with $C := |\mathcal{C}|$, is the set of 'known' classes. Each discrete class label $y_i \in \mathcal{C}$ admits a vector representation via a one-hot encoding function, yielding: $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iC}]^\top \in \{0,1\}^C$, where the element corresponding to the ground-truth class $c = y_i$ is $y_{ic} = 1$ and $y_{ik} = 0$ for $k \neq c$. Both the one-hot encoded label $\mathbf{y}_i$ and any subsequent soft targets lie in the probability simplex $\Delta^{C-1} := \{\mathbf{g} \in \mathbb{R}^C_{\geq 0} : \sum_k g_k = 1\}$.

In the closed-set scenario, the model is evaluated on a test set where the labels are strictly drawn from the training categories: $\mathcal{D}_K = \{(\mathbf{x}_j, y_j)\}_{j=1}^{M_K} \subset \mathcal{X} \times \mathcal{C}$. In this setting, the model returns a distribution over known classes as $p(y \mid \mathbf{x}, y \in \mathcal{C})$. Conversely, in Open-Set Recognition (OSR), test images may also originate from unseen classes $\mathcal{U}$, where $\mathcal{U}$ denotes the set of unknown classes and $\mathcal{C} \cap \mathcal{U} = \emptyset$. The OSR-evaluation test set is a union of known and unknown samples: $\mathcal{D}_{\text{Open}} = \mathcal{D}_K \cup \mathcal{D}_U$. The unknown component is defined as: $\mathcal{D}_U = \{(\mathbf{x}_j, y_j)\}_{j=1}^{M_U} \subset \mathcal{X} \times \mathcal{U}$.

Scoring Function. In the open-set setting, the model provides both a class distribution $p(y \mid \mathbf{x}, y \in \mathcal{C})$ and a scoring function $\mathcal{S}(\mathbf{x})$ to indicate whether or not the test sample belongs to any of the known classes. The final prediction $\hat{y}$ is determined by an operational threshold θ:

$$\hat{y} = \begin{cases} \text{Known class given via, } \operatorname{argmax}_{y \in \mathcal{C}} \; p(y \mid \mathbf{x}, y \in \mathcal{C}) & \text{if } \mathcal{S}(\mathbf{x}) \geq \theta, \\ \text{Unknown class} & \text{if } \mathcal{S}(\mathbf{x}) < \theta. \end{cases} \tag{1}$$

4 Methodology

Scheirer et al. (2013) point out that the key to optimizing OSR performance is to simultaneously reduce the empirical classification risk on known-class data and the open-space risk, the risk of assigning a known class to regions in the representation space that lie far from any regions associated

with known-class training data. A representation space with compact, well-separated class clusters satisfies this property and is therefore the desired geometry. Representation-based OSR methods enforce this geometry through a first-stage objective $\mathcal{L}_F$ that shapes the latent space (Table 1).

Concretely, OSR methods based on representation learning follow a three-stage pipeline. The first two stages are inherited from, and theoretically grounded in, self-supervised learning(Grill et al., 2020; Chen et al., 2020b). In the first stage, an encoder is trained to produce geometrically structured representations, guided by $\mathcal{L}_F$. During the second stage, a lightweight classifier is trained atop the frozen encoder to map representations to a class distribution $p(y \mid \mathbf{x}, y \in \mathcal{C})$, a procedure referred to as linear probing. In the third stage, for the specific objective of OSR, which is different from closed-set classification, the inference phase comprises two steps: (a) computing a scoring function Eq. (1) $S(x)$ to detect unknown samples, typically via post-processors, and (b) classifying samples identified as known with the closed-set posterior. We intervene in the first stage with a plug-in, a form of sample-adaptive label smoothing algorithm, that plugs into the loss function and upgrades existing representation objectives into stronger variants. Section 4.1 will detail the general learning framework, Section 4.2 will give the theoretical motivation behind our method, and Section 4.3 will detail the engineering behind the plug-in.

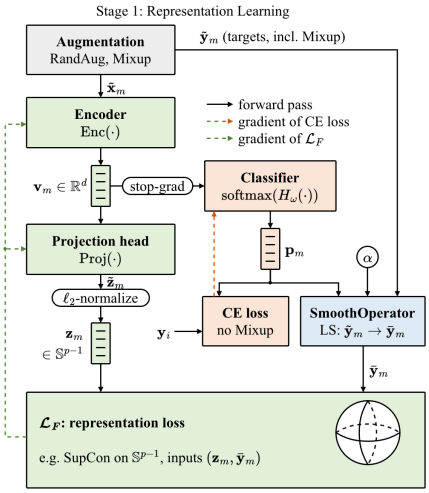


Figure 1: **Overview of the Proposed Framework.** A three-stage process in which an input sample passes through an augmentation pipeline before reaching the feature encoder. An Observer Classifier estimates prominence from gradient-detached features to modulate label smoothing in the representation learning objective.

4.1 General Representation Learning Framework

Across OSR representation learning, the main differences are in how the loss function is constructed. So we first introduce a general framework and elaborate on the loss function design. This section will describe it from the point of view of data flow.

Stage 1: *RandAugment.* Every input batch $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^B$ is pre-processed using the RandAugment protocol (Cubuk et al., 2020) to produce $\{\mathbf{x'}_i, \mathbf{y}_i\}_{i=1}^B$. Two label-altering operations then produce the soft targets central to our framework.

Label Smoothing. Consistent with established baselines (Vaze et al., 2022), we replace the one-hot target $\mathbf{y}_i$ with a smoothed target via the map $\mathrm{LS}_\varepsilon : \Delta^{C-1} \to \Delta^{C-1}$,

$$\mathbf{y}'_i := \mathrm{LS}_\varepsilon(\mathbf{y}_i) = (1-\varepsilon)\,\mathbf{y}_i + \frac{\varepsilon}{C} \cdot \mathbf{1}, \qquad \varepsilon \in [0,1], \tag{2}$$

where ε is the label smoothing coefficient, and $\mathbf{1}$ is an all-ones vector.

Mixup. For objectives that employ it (e.g., ConOSR, SpHOR), Mixup regularization (Zhang et al., 2018) generates virtual samples.

$$\hat{\mathbf{x}}_i = \lambda \mathbf{x}'_i + (1-\lambda)\mathbf{x}'_{\pi(i)}, \quad \hat{\mathbf{y}}_i = \lambda \mathbf{y}'_i + (1-\lambda)\mathbf{y}'_{\pi(i)}, \tag{3}$$

where $\lambda \sim \mathrm{Beta}(2,2)$ is drawn per batch, $\pi(i)$ is a a random permutation of the batch.

Both these operations map the simplex to itself, so the target remains a valid distribution. The final augmented batch, denoted as $\{(\tilde{\mathbf{x}}_m, \tilde{\mathbf{y}}_m)\}_{m=1}^{2B} := \{(\mathbf{x}'_i, \mathbf{y}'_i)\}_{i=1}^B \cup \{(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i)\}_{i=1}^B$, provides a unified set of original and synthetic samples on which the model is trained.

Training, Encoding and Projection. An encoder network $Enc(\cdot)$ is employed to map each sample into a d-dimensional representation $\mathbf{v}_m = \mathrm{Enc}(\tilde{\mathbf{x}}_m)$. The representations are mapped through a projection head $Proj(\cdot)$ to yield p-dimensional projections $\tilde{\mathbf{z}}_m = Proj(\mathbf{v}_m)$. The $\tilde{\mathbf{z}}_m$ are L_2 normalized to produce $\mathbf{z}_m$. For brevity, we denote the composite mapping by f, i.e., $f(\mathbf{x}) \triangleq$

$\mathrm{Proj}(\mathrm{Enc}(\mathbf{x})) \,/\, \|\mathrm{Proj}(\mathrm{Enc}(\mathbf{x}))\|_2$, such that $\mathbf{z}_m = f(\tilde{\mathbf{x}}_m) \in \mathbb{S}^{p-1}$. The encoder and projection head are trained jointly to minimize the representation objective $\mathcal{L}_F$, computed on the normalized projections. This reflects the standard contrastive learning protocol(Khosla et al., 2020). We use four representation objectives: **ConOSR, SpHOR, Supervised Contrastive Learning with Label Smoothing (SupCON) and SpHOR without Mixup (SpHOR-Lite)** (Khosla et al., 2020; Xu et al., 2023; Bahavan et al., 2026). The algorithm details and how SmoothOP is fused with it are detailed in the Appendix E.

Stage 2: Linear Probing. Once Stage 1 is complete, the projection head is discarded and the encoder $\mathrm{Enc}(\cdot)$ is frozen. To enable class-level predictions for the known set $\mathcal{C}$, we append a linear classification head $H_\omega(\cdot)$ to the fixed d-dimensional embeddings $\mathbf{f}$. During this stage, we optimize the weights ω using a standard cross-entropy loss over the known classes: $\mathcal{L}_{\mathrm{CE}}(\omega) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k\in\mathcal{C}} y_{ik}\,\log\left[\mathrm{Softmax}\big(H_\omega(\mathbf{v}_i)\big)\right]_k$. No label smoothing in this stage.

Stage 3: Open-Set Inference. With (Enc, H_ω) frozen, the posterior in Eq. (1) is computed as $p(y \mid \mathbf{x}, y \in \mathcal{C}) = \mathrm{Softmax}(H_\omega(\mathbf{v}))$, $\mathbf{v} = \mathrm{Enc}(\mathbf{x})$, and the score $\mathcal{S}(\mathbf{x})$ as a training-free post-processor on $\mathbf{v}$, $p(y \mid \mathbf{x}, y \in \mathcal{C})$; prediction then follows Eq. (1).

4.2 Representation Objectives: Alignment and Uniformity

Contrastive representation losses share a single structure. In this framework, a reference sample (the *anchor*) is attracted to semantically similar samples (*positives*) and repelled from semantically distinct ones (*negatives*). The objectives differ in their choice of positives and negatives, the magnitude of attraction or repulsion and the inclusion of auxiliary regularization terms. While self-supervised methods focus on instance discrimination, supervised and OSR-based methods extend this machinery to enforce class-level discrimination. The strong empirical performance of contrastive learning approaches is explained by two properties of projections: *alignment* and *uniformity* (Wang & Isola, 2020).

Alignment ensures that for any reference sample acting as an *anchor*, its semantic *positives* are mapped to close proximity in the projection space, making the projections robust to noise.

Definition 1 (Alignment)**.** Formally, given a distribution of positive pairs p_{pos}, alignment is defined via expected distance:

$$\mathcal{L}_{\text{align}}(f;\gamma) \triangleq \mathbb{E}_{(x,x^+)\sim p_{\text{pos}}}\left[\|f(x)-f(x^+)\|_2^{\gamma}\right] = 2 + 2\mathbb{E}_{(x,x^+)\sim p_{\text{pos}}}\left[-f(x)^\top f(x^+)\right], \quad \gamma = 2. \tag{4}$$

Uniformity, on the other hand, dictates that the projections of samples drawn from the data distribution should be distributed roughly uniformly across the unit hypersphere $\mathbb{S}^{p-1}$, preventing representation collapse and retaining maximal information.

Definition 2 (Uniformity)**.** For samples drawn i.i.d. from the data distribution p_{data},

$$\mathcal{L}_{\text{uniform}}(f;t) \triangleq \log \mathbb{E}_{x,x' \overset{\text{i.i.d.}}{\sim} p_{\text{data}}}\left[e^{-t\|f(x)-f(x')\|_2^2}\right] = -2t + \log \mathbb{E}_{x,x' \overset{\text{i.i.d.}}{\sim} p_{\text{data}}}\left[\exp\left(2t\, f(x)^\top f(x')\right)\right] \tag{5}$$

We first decompose all four objectives into alignment and uniformity, as shown in Table 1. This yields the following proposition.

Proposition 1 (Shared structure)**.** *Let $\mathcal{L}_i$ be the target-dependent loss of any objective in Table 1, with attraction weights $S_{i\cdot}$ satisfying $\sum_a S_{ia} = 1$ over the method's contrast set. Then $\mathcal{L}_i$ decomposes exactly as $\mathcal{L}_i = \mathcal{L}^{(i)}_{\text{align}} + \mathcal{L}^{(i)}_{\text{unif}}$, where $\mathcal{L}^{(i)}_{\text{unif}}$ is independent of the targets $\tilde{\mathbf{y}}$, and $\mathcal{L}^{(i)}_{\text{align}}$ depends on the targets only through $S_{i\cdot}$.*

We leverage this decoupling to our advantage. Because label smoothing acts solely on the targets $\tilde{\mathbf{y}}_i$, it exclusively modulates alignment. The table and the proposition are detailed in Appendix F.

Proposition 2 (Label smoothing as an alignment dial)**.** *Under the conditions of Prop. 1, applying SmoothOP to the anchor targets leaves $\mathcal{L}^{(i)}_{\text{unif}}$ unchanged and modifies $\mathcal{L}^{(i)}_{\text{align}}$ only through $S_{i\cdot}$. For the SpHOR-Lite/SpHOR, increasing ε_i transfers attraction mass $\varepsilon_i(1-1/C)$ from the ground-truth label embedding uniformly onto the others; for the SupCON/ConOSR with one-hot contrast labels, each positive's weight decreases and each negative's weight increases strictly in ε_i.*

Method	Target $S_{i\cdot}$	Alignment $\mathcal{L}_{\text{align}}^{(i)}$	Uniformity $\mathcal{L}_{\text{unif}}^{(i)}$
SupCON	$S_{ij} = \frac{s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j)}{\sum_{n\neq i} s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_n)}$	$\frac{1}{2\tau}\sum_{j\neq i} S_{ij}\,\lVert \mathbf{z}_i - \mathbf{z}_j\rVert_2^2$	$\log \sum_{n\neq i} \exp\left(-\frac{\lVert \mathbf{z}_i - \mathbf{z}_n\rVert_2^2}{2\tau}\right)$
ConOSR	$S_{ij} = \frac{s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j)}{\sum_{n\neq i} s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_n)}$	$\frac{1}{2\tau}\sum_{j\neq i} S_{ij}\,\lVert \mathbf{z}_i - \mathbf{z}_j\rVert_2^2$	$\log \sum_{n\neq i} \exp\left(-\frac{\lVert \mathbf{z}_i - \mathbf{z}_n\rVert_2^2}{2\tau}\right)$
SpHOR-Lite	$S_{ik} = \frac{\tilde{y}_{ik}}{\sum_{j=1}^{C} \tilde{y}_{ij}}$	$\frac{1}{2\tau}\sum_{k=1}^{C} S_{ik}\,\lVert \mathbf{z}_i - \boldsymbol{\mu}_k\rVert_2^2$	$\log \sum_{j=1}^{C} \exp\left(-\frac{\lVert \mathbf{z}_i - \boldsymbol{\mu}_j\rVert_2^2}{2\tau}\right)$
SpHOR	$S_{ik} = \frac{\tilde{y}_{ik}}{\sum_{j=1}^{C} \tilde{y}_{ij}}$	$\frac{1}{2\tau}\sum_{k=1}^{C} S_{ik}\,\lVert \mathbf{z}_i - \boldsymbol{\mu}_k\rVert_2^2$	$\log \sum_{j=1}^{C} \exp\left(-\frac{\lVert \mathbf{z}_i - \boldsymbol{\mu}_j\rVert_2^2}{2\tau}\right)$

Table 1: Alignment–uniformity decomposition of the target-dependent loss of each representation learning objective. The target distribution S_{ab}, the sole carrier of the label smoothing degree, appears only in the alignment term, while the uniformity term is independent of the targets. This is detailed in Appendix F. Here $s(\cdot,\cdot)$ is cosine similarity and $\tilde{\mathbf{y}}_i$ is the label-smoothed target

.

Proof in Appendix E (Props. 4, 5).

4.3 Prominence and Adaptive Label Smoothing

Our approach consists of two main components: a per-sample score that quantifies how a sample's own class stands out against its strongest competing class, and a bounded mapping function that translates this score into an individual smoothing coefficient ε_i.

To compose a signal related to the structure of the embedding geometry, we simultaneously train an **Observer classifier** $H_\omega(\cdot)$ (A linear head, which is reused during the Second Stage) using cross-entropy on the gradient-detached embeddings $\mathbf{v_m}$. (so it doesn't interfere with the training dynamics and prevents circularity). This supplies the evolving predicted class probabilities $\mathbf{p}^{(i)} = [p_1^{(i)}, \ldots, p_C^{(i)}]^\top$.

Definition 3 (Prominence)**.** Let $\mathbf{x}_i$ be a sample with ground-truth class c, let $p_c^{(i)}$ be the observer's predicted probability for c, and let $\hat{p}^{(i)} = \max_{k\in\mathcal{C}\setminus\{c\}} p_k^{(i)}$ be the highest probability among the competing classes. We define the probability margin $u_i \in [-1, 1]$ as

$$u_i := \hat{p}^{(i)} - p_c^{(i)}. \tag{6}$$

A sample is *prominent* when its own class clearly stands out against its strongest competitor, i.e., when u_i is low; its prominence decreases as u_i increases. Conversely, a sample with u_i near or above zero is ambiguous: it is similar to its own class but also to at least one competing class, and can easily be misclassified.

Note that this margin has two modes of behavior depending on whether the classifier classifies the sample correctly or not:

- *Correct Prediction:* the predicted class is j, so $\hat{p}^{(i)} \leq p_j^{(i)}$ and u_i are negative. As the model grows more confident, it assigns more weight to the true class, and u_i decreases towards -1. Effectively, the sample has high prominence.
- *Incorrect Prediction:* the maximum probability belongs to a class other than j, so $p_j^{(i)} < \hat{p}^{(i)}$ and u_i is positive. A margin near zero means the model cannot separate the true class from a competing one, which is genuine ambiguity. A large positive margin is a confidently wrong prediction. Effectively, as u_i *increases*, the sample has low prominence.

If compact, well-separated clusters are the desired geometry, samples that could be confused with another class are where it breaks down: they sit at the edge of their cluster, close to a competing one, and are the most likely to be misclassified. These are exactly the samples with low prominence.

Samples with high prominence already fit this geometry, so it is those with low prominence that need a stronger pull toward their class. This motivates our algorithm design, which is presented below.

To make label smoothing adaptive, we convert the smoothing coefficient into a per-sample basis:

$$\mathbf{y_i}' := \mathrm{SmoothOP}(\mathbf{y}_i) = (1-\varepsilon_i)\,\mathbf{y}_i + \frac{\varepsilon_i}{C}\cdot\mathbf{1}, \qquad \varepsilon \in [0,1], \tag{7}$$

We modulate this per-sample smoothing coefficient based on the sample's corresponding prominence as follows:

$$\varepsilon_i \;=\; \varepsilon \cdot 2\,\mathrm{sigmoid}(-|\alpha|u_i) \tag{8}$$

Here, ε is the original default label smoothing coefficient, except that it is modulated by the prominence. α is the modulating strength.

In this setup, the degree of smoothing reduces with prominence. For incorrectly classified, difficult samples that have high u, the degree of smoothing is less. However, for correctly classified, easy samples, the degree of smoothing is high. Thus, the strength of the correction is based on the α value.

Proposition 3 (Properties of the sigmoid modulator)**.** *Let $\varepsilon_i = 2\varepsilon\,\sigma(-|\alpha|u_i)$ with $\sigma(t) = (1+e^{-t})^{-1}$, $\varepsilon \in (0,1)$, $\alpha \geq 0$. Then: (i)* Boundedness: *$\varepsilon_i \in (0, 2\varepsilon)$ for all $u_i \in \mathbb{R}$; in particular $\alpha = 0$ yields $\varepsilon_i \equiv \varepsilon$, recovering fixed smoothing (Eq. 30). (ii)* Monotonicity: *for $\alpha > 0$, ε_i is strictly decreasing in u_i.*

While prominence calculation is straightforward for individual samples, characterizing the prominence of a synthetic sample generated via Mixup (Zhang et al., 2018) presents a unique challenge, because it has two target classes. We hypothesize that the prominence of a composite sample is a function of the constituent samples' prominence, propagated through the mixing manifold. Consistent with the linear interpolation principle of Mixup, we define the composite prominence $U(\hat{\mathbf{x}}_k)$ as a weighted combination of the individual prominence (We retrieve this via the indexes of the pre-mixed samples):

The constituent margins are read from the *unmixed* forward pass and retrieved through the mixing indices.

$$\mathbf{u}^{\lambda}_{i,j} = \lambda\cdot\mathbf{u}_i + (1-\lambda)\cdot\mathbf{u}_j \qquad \varepsilon^{\lambda}_{i,j} \;=\; 2\varepsilon\cdot\mathrm{sigmoid}\big(-|\alpha|\,\mathbf{u}^{\lambda}_{i,j}\big) \in (0,\,2\varepsilon), \tag{9}$$

This composite margin induces a single smoothing coefficient, applied once to the mixed target:

$$\hat{\mathbf{y}}^{'}_k = (1-\varepsilon^{\lambda}_{i,j})\big(\lambda\mathbf{y}_i + (1-\lambda)\mathbf{y}_j\big) + \frac{\varepsilon^{\lambda}_{i,j}}{C}\cdot\mathbf{1}, \tag{10}$$

where $\mathbf{y}_i, \mathbf{y}_j$ are the *one-hot* targets of the constituent samples. This is empirically ablated in Appendix D.3.

5 Experimental Setup

Datasets. We evaluate our approach on the three fine-grained datasets in Semantic Shift Benchmark (Vaze et al., 2022), which comprises three fine-grained classification datasets: CUB (Wah et al., 2011), FGVC-Aircraft (Maji et al., 2013), and Stanford Cars (Krause et al., 2013). Each dataset has an associated unknown class test sets ('Easy' and 'Hard') based on their semantic similarity to the training classes. Further details on the datasets are in Appendix B.3.

Hyperparameter Tuning. Base smoothing λ is 0.3 on CUB and 0.2 on Aircraft and Cars. This was selected directly from Vaze et al. (2022) and we adopt these values without further tuning. We find the optimal α parameter for each dataset–method combination using a three-fold cross-validation on a training split, with images from different fine-grained datasets as validation unknowns. We take the α that maximizes mean OSCR under MSP. We defer the complete sweep grid to Appendix B.1.

Training. We use a ResNet-50 pretrained on the Places-365 dataset (via Supervised Training) for our experiments. We optimize with SGD (momentum 0.9, learning rate 0.05, weight decay 10^{-4}). We train with distributed data parallelism on 4 GPUs (2 nodes $\times$ 2 GPUs), with 128 images per GPU, for 200 epochs. The learning rate follows a cosine schedule with warm restarts: a 10-epoch linear warmup, then three cosine cycles of 66 epochs each (two restarts), annealed to $10^{-3}\times$

Table 2: **Fine-grained OSR performance on the Semantic Shift Benchmark (SCARS, FGVC-Aircraft, and CUB): OSCR and AUROC** Each dataset has an "Easy" and "Hard" unknown class split(Vaze et al., 2022). We designate OSCR as the primary metric. For each representation learning method (ConOSR(Xu et al., 2023), SpHOR(Bahavan et al., 2026), Supervised Contrastive Learning(Khosla et al., 2020)) and its augmented variant (+), we evaluate performance across four SOTA scoring functions/postprocessors. Our augmented variants consistently outperform baselines across various metrics and datasets. All methods utilize a ResNet-50 backbone pre-trained on the Places dataset. Gains are in blue. The results are given as the mean ± sd over three training runs with fixed seeds.

		OSCR						AUROC					
		CUB		Aircraft		Cars		CUB		Aircraft		Cars	
		E	H	E	H	E	H	E	H	E	H	E	H
MSP	CE	58.64 ± 0.71	55.75 ± 0.44	69.63 ± 0.72	67.97 ± 0.45	83.28 ± 0.36	76.48 ± 0.25	73.20 ± 1.32	68.42 ± 0.24	76.94 ± 0.55	75.13 ± 1.05	87.54 ± 0.33	79.53 ± 0.38
	SupCON(w/o LS)	59.54 ± 1.30	54.49 ± 1.17	69.16 ± 1.02	68.46 ± 0.82	82.13 ± 0.51	75.61 ± 0.79	74.85 ± 0.87	66.51 ± 1.18	77.14 ± 1.06	75.53 ± 0.72	86.37 ± 0.80	78.75 ± 0.82
SupCON	+SmoothOP: CUB −0.5, Aircraft −4, Cars −3												
MSP	Baseline	61.42 ± 0.87	56.64 ± 0.84	70.87 ± 1.75	66.93 ± 2.34	83.61 ± 0.74	75.36 ± 1.46	76.65 ± 0.45	68.76 ± 0.29	80.28 ± 0.60	74.82 ± 1.34	88.49 ± 0.13	78.70 ± 1.07
	+SmoothOP	62.54 ± 0.49	57.15 ± 0.85	72.08 ± 1.12	69.26 ± 0.97	83.69 ± 0.29	75.60 ± 0.91	78.18 ± 0.62	69.34 ± 0.98	80.38 ± 0.75	76.63 ± 1.58	88.14 ± 0.59	78.69 ± 0.90
	gain	+1.82%*	+0.91%	+1.70%	+3.49%	+0.09%	+0.32%	+2.00%*	+0.85%	+0.12%	+2.43%	−0.40%	−0.01%
MLS	Baseline	62.55 ± 1.09	56.16 ± 0.71	72.06 ± 1.95	67.09 ± 2.39	84.07 ± 0.77	75.58 ± 1.41	79.23 ± 0.50	68.65 ± 0.09	82.18 ± 0.83	75.40 ± 1.41	89.30 ± 0.27	79.09 ± 0.99
	+SmoothOP	63.43 ± 0.55	56.67 ± 0.83	73.22 ± 1.02	69.21 ± 1.13	84.22 ± 0.15	75.72 ± 0.74	80.20 ± 0.61	69.17 ± 1.00	82.08 ± 0.67	76.88 ± 1.77	89.06 ± 0.39	78.99 ± 0.74
	gain	+1.41%	+0.91%	+1.60%	+3.15%	+0.19%	+0.19%	+1.22%*	+0.76%	−0.13%	+1.97%	−0.27%	−0.13%
NNGuide	Baseline	52.05 ± 1.53	49.22 ± 0.88	68.93 ± 2.52	66.19 ± 3.04	77.99 ± 2.75	73.67 ± 1.83	67.08 ± 1.30	62.80 ± 1.83	77.72 ± 1.78	74.24 ± 1.93	82.80 ± 2.94	78.04 ± 1.42
	+SmoothOP	54.19 ± 0.65	50.22 ± 2.75	70.75 ± 1.97	68.56 ± 1.10	79.51 ± 1.45	74.05 ± 0.47	69.43 ± 0.70	63.48 ± 3.60	78.99 ± 2.13	76.26 ± 1.71	84.29 ± 1.57	78.14 ± 0.36
	gain	+4.12%	+2.03%	+2.65%	+3.58%	+1.96%	+0.52%	+3.51%	+1.07%	+1.63%	+2.72%	+1.81%	+0.13%
PostMax	Baseline	63.18 ± 1.28	55.85 ± 0.78	71.69 ± 2.47	64.47 ± 0.83	84.01 ± 0.76	76.08 ± 0.71	80.04 ± 0.62	68.36 ± 0.20	81.92 ± 1.62	72.63 ± 0.68	89.18 ± 0.35	79.56 ± 0.59
	+SmoothOP	63.75 ± 0.68	56.53 ± 0.36	72.62 ± 1.22	67.51 ± 0.79	84.36 ± 0.49	75.31 ± 0.99	80.62 ± 0.36	69.03 ± 0.73	81.54 ± 1.04	75.08 ± 1.19	89.18 ± 0.57	78.57 ± 1.02
	gain	+0.90%	+1.22%	+1.30%	+4.70%*	+0.42%	−1.00%	+0.72%†	+0.98%	−0.47%	+3.37%	−0.00%	−1.25%†
ConOSR	+SmoothOP : CUB −3, Aircraft −1.5, Cars −4												
MSP	Baseline	65.60 ± 1.43	59.80 ± 1.19	75.87 ± 0.61	71.62 ± 1.01	85.49 ± 0.77	77.91 ± 0.75	79.41 ± 1.18	70.59 ± 0.80	82.83 ± 0.34	77.70 ± 1.22	89.66 ± 0.66	80.89 ± 0.65
	+SmoothOP	65.19 ± 1.12	59.86 ± 1.29	76.94 ± 0.45	72.09 ± 0.65	86.13 ± 0.71	78.17 ± 0.92	79.13 ± 0.32	70.92 ± 0.62	83.58 ± 0.74	77.72 ± 0.98	90.03 ± 0.50	80.93 ± 0.80
	gain	−0.64%	+0.11%	+1.41%*	+0.65%	+0.74%†	+0.34%	−0.35%	+0.46%	+0.90%†	+0.03%	+0.41%	+0.05%
MLS	Baseline	66.29 ± 1.43	58.80 ± 0.96	77.72 ± 0.22	71.01 ± 1.24	85.47 ± 0.50	77.72 ± 0.72	81.03 ± 1.35	69.85 ± 0.59	85.71 ± 0.34	77.56 ± 1.50	89.84 ± 0.32	80.88 ± 0.62
	+SmoothOP	65.94 ± 1.07	58.91 ± 0.94	78.60 ± 0.16	71.48 ± 0.77	86.09 ± 0.64	78.01 ± 1.01	80.92 ± 0.34	70.29 ± 0.30	86.09 ± 0.07	77.52 ± 1.12	90.18 ± 0.56	80.93 ± 0.87
	gain	−0.52%	+0.19%	+1.14%*	+0.66%	+0.73%†	+0.37%	−0.14%	+0.63%	+0.44%	−0.05%	+0.38%	+0.06%
NNGuide	Baseline	58.38 ± 3.15	51.32 ± 2.30	76.64 ± 0.99	70.65 ± 1.50	82.39 ± 0.70	76.27 ± 0.34	73.10 ± 3.08	63.02 ± 1.82	84.80 ± 0.80	77.31 ± 1.77	86.68 ± 1.04	79.68 ± 0.36
	+SmoothOP	57.90 ± 1.82	51.19 ± 1.58	77.70 ± 1.06	71.01 ± 0.60	83.38 ± 0.96	76.84 ± 0.70	72.89 ± 1.50	63.30 ± 0.78	85.42 ± 1.64	77.19 ± 0.92	87.37 ± 1.37	79.96 ± 0.51
	gain	−0.83%	−0.25%	+1.38%*	+0.50%	+1.20%	+0.74%	−0.29%	+0.44%	+0.74%	−0.15%	+0.80%	+0.35%
PostMax	Baseline	66.58 ± 1.43	58.82 ± 0.77	78.66 ± 0.13	69.59 ± 2.01	85.59 ± 0.18	78.07 ± 0.55	81.37 ± 1.37	69.88 ± 0.44	86.94 ± 0.70	75.97 ± 2.23	89.96 ± 0.08	81.22 ± 0.44
	+SmoothOP	66.25 ± 1.14	58.95 ± 0.63	79.31 ± 0.74	70.61 ± 1.40	86.20 ± 0.40	78.46 ± 0.79	81.25 ± 0.46	70.29 ± 0.21	86.99 ± 0.84	76.53 ± 1.72	90.28 ± 0.63	81.37 ± 0.58
	gain	−0.49%	+0.22%	+0.82%	+1.46%	+0.71%†	+0.50%	−0.15%	+0.59%	+0.05%	+0.74%	+0.36%	+0.18%
SpHOR-Lite	+SmoothOP: CUB −8, Aircraft −1.5, Cars −0.5												
MSP	Baseline	63.99 ± 1.35	59.18 ± 0.73	69.10 ± 1.94	67.91 ± 0.81	84.27 ± 0.22	75.85 ± 0.58	77.96 ± 1.08	70.13 ± 0.31	75.50 ± 1.98	74.05 ± 0.61	88.20 ± 0.20	78.46 ± 0.55
	+SmoothOP	65.24 ± 0.98	59.90 ± 0.60	69.36 ± 1.25	67.80 ± 0.89	84.47 ± 0.18	75.86 ± 0.86	79.25 ± 0.57	70.62 ± 0.64	75.76 ± 1.55	73.84 ± 1.20	88.35 ± 0.21	78.41 ± 0.97
	gain	+1.96%*	+1.21%*	+0.38%	−0.16%	+0.23%	+0.01%	+1.65%†	+0.70%	+0.34%	−0.28%	+0.16%	−0.06%
MLS	Baseline	64.70 ± 1.24	58.12 ± 0.67	70.28 ± 1.82	67.63 ± 0.88	84.31 ± 0.24	75.60 ± 0.54	79.88 ± 0.92	69.40 ± 0.55	77.42 ± 1.83	74.03 ± 0.71	88.47 ± 0.14	78.34 ± 0.51
	+SmoothOP	65.72 ± 0.97	59.08 ± 0.74	70.50 ± 1.26	67.64 ± 1.01	84.55 ± 0.25	75.72 ± 0.82	80.49 ± 0.57	70.00 ± 0.83	77.64 ± 1.55	73.95 ± 1.35	88.59 ± 0.21	78.36 ± 0.95
	gain	+1.58%*	+1.65%*	+0.32%	+0.00%	+0.28%	+0.15%	+0.76%	+0.86%†	+0.28%	−0.11%	+0.14%	+0.03%
NNGuide	Baseline	61.16 ± 1.31	55.68 ± 0.83	66.05 ± 2.32	66.52 ± 1.36	83.20 ± 0.23	75.50 ± 0.27	75.83 ± 1.40	67.08 ± 0.62	71.81 ± 2.47	72.46 ± 1.20	87.82 ± 0.30	78.74 ± 0.28
	+SmoothOP	61.91 ± 0.98	56.23 ± 1.46	66.32 ± 2.01	66.76 ± 1.53	83.30 ± 0.18	75.62 ± 0.63	76.03 ± 0.93	67.18 ± 1.44	72.10 ± 2.45	72.66 ± 1.86	87.69 ± 0.42	78.70 ± 0.68
	gain	+1.23%†	+1.00%	+0.41%	+0.36%	+0.12%	+0.16%	+0.26%	+0.15%	+0.40%	+0.29%	−0.15%	−0.06%
PostMax	Baseline	65.02 ± 1.13	57.89 ± 0.73	73.11 ± 1.39	67.13 ± 0.53	84.26 ± 0.47	75.71 ± 0.64	80.58 ± 0.76	69.40 ± 0.31	80.98 ± 1.39	73.64 ± 0.28	88.44 ± 0.15	78.48 ± 0.52
	+SmoothOP	65.91 ± 0.86	58.87 ± 0.42	73.14 ± 1.40	66.95 ± 0.58	84.45 ± 0.49	75.74 ± 0.76	80.88 ± 0.47	69.90 ± 0.30	80.95 ± 1.63	73.36 ± 0.88	88.52 ± 0.27	78.42 ± 0.87
	gain	+1.37%*	+1.69%*	+0.04%	−0.27%*	+0.22%	+0.04%	+0.37%	+0.72%†	−0.04%	−0.39%	+0.09%	−0.08%
SpHOR	+SmoothOP: CUB −8, Aircraft −2, Cars −3												
MSP	Baseline	67.35 ± 1.17	61.70 ± 0.58	77.38 ± 0.57	72.12 ± 0.30	86.29 ± 0.76	78.67 ± 1.26	80.76 ± 0.92	72.24 ± 0.24	84.43 ± 0.50	78.02 ± 0.36	90.25 ± 0.43	81.39 ± 1.03
	+SmoothOP	68.26 ± 0.93	62.20 ± 0.50	77.55 ± 0.32	72.04 ± 0.42	87.13 ± 0.39	79.05 ± 0.53	81.83 ± 0.57	72.62 ± 0.36	84.62 ± 0.40	77.90 ± 0.50	90.89 ± 0.05	81.57 ± 0.36
	gain	+1.35%*	+0.81%*	+0.22%	−0.12%	+0.98%†	+0.49%	+1.32%†	+0.53%*	+0.23%	−0.15%	+0.71%†	+0.23%
MLS	Baseline	67.90 ± 1.12	60.26 ± 0.59	78.34 ± 0.53	70.97 ± 0.22	86.20 ± 0.84	78.28 ± 1.21	82.18 ± 0.88	70.90 ± 0.41	85.87 ± 0.51	76.94 ± 0.19	90.36 ± 0.55	81.17 ± 0.94
	+SmoothOP	68.67 ± 0.92	60.98 ± 0.51	78.54 ± 0.23	71.12 ± 0.75	87.04 ± 0.46	78.66 ± 0.55	82.96 ± 0.59	71.52 ± 0.50	86.08 ± 0.26	77.11 ± 0.85	90.95 ± 0.16	81.31 ± 0.33
	gain	+1.12%†	+1.19%*	+0.26%	+0.22%	+0.97%†	+0.48%	+0.95%	+0.88%*	+0.25%	+0.22%	+0.65%	+0.16%
NNGuide	Baseline	63.82 ± 0.85	55.98 ± 0.95	76.20 ± 0.23	70.37 ± 0.36	84.05 ± 1.42	76.94 ± 0.82	77.80 ± 0.55	66.54 ± 1.01	82.99 ± 0.40	76.00 ± 0.48	88.17 ± 1.28	79.99 ± 0.59
	+SmoothOP	64.50 ± 0.80	57.31 ± 1.10	76.33 ± 0.51	70.72 ± 1.01	85.05 ± 0.65	77.60 ± 0.55	77.99 ± 0.27	67.59 ± 0.96	83.11 ± 0.34	76.34 ± 1.21	88.88 ± 0.47	80.36 ± 0.51
	gain	+1.07%	+2.38%*	+0.17%	+0.49%	+1.19%	+0.85%	+0.24%	+1.57%*	+0.15%	+0.46%	+0.81%	+0.47%
PostMax	Baseline	68.19 ± 1.13	60.41 ± 0.73	78.78 ± 0.46	69.88 ± 0.41	86.15 ± 0.75	78.22 ± 0.70	82.52 ± 0.91	71.03 ± 0.48	86.34 ± 0.43	75.65 ± 0.43	90.32 ± 0.48	81.12 ± 0.43
	+SmoothOP	68.87 ± 1.02	61.22 ± 0.66	79.14 ± 0.14	70.09 ± 0.80	86.90 ± 0.43	78.55 ± 0.31	83.19 ± 0.75	71.73 ± 0.53	86.73 ± 0.31	75.88 ± 0.80	90.82 ± 0.17	81.21 ± 0.20
	gain	+1.01%	+1.33%*	+0.46%	+0.31%	+0.87%†	+0.42%	+0.81%	+0.99%*	+0.46%*	+0.31%	+0.55%	+0.12%

the base rate. The contrastive temperature is 0.1. Feature dimension is 2048. The projector is a $\mathrm{Linear}(2048 \to 2048)$ layer. We augment four baseline representation methods, i.e., SpHOR(Bahavan et al., 2026), SpHOR without Mixup (SpHOR-Lite), ConOSR (Xu et al., 2023), and Supervised Contrastive Learning (SupCON) (Khosla et al., 2020). All methods were trained over three runs using fixed seeds (0, 1, 2), with results reported as the mean across these trials for methodological rigor. The backbone's classifier head is retrained (with the backbone frozen) for 20 epochs in stage 2 before evaluations. Full hyperparameter settings (including Stage 2) are listed in Appendix B.1.

Evaluation and Metrics. We evaluated all other methods using state-of-the-art post-processors/scoring functions, including MLS(Vaze et al., 2022) , nnGuide ($k = 5$) (Park et al., 2023), PostMAX (Cruz et al., 2024), and Maximum Softmax Probability (MSP). OSR performance is evaluated on two axes: how well unknowns can be rejected and how well knowns can be classified. The former is quantified using *AUROC*. The latter is quantified by *Closed-set accuracy*. Since strength on one axis can mask weakness on the other, neither suffices alone: Both axes are jointly evaluated by *Open-Set Classification Rate (OSCR)* (Dhamija et al., 2018). A detailed explanation/justification of the metrics is in Appendix A and post-processors is in Appendix B.4.

6 Results and Discussion

Superiority of Augmented Variants in SSB Benchmark: Table 2 reports OSR results on the SSB Benchmark. Our proposed SmoothOP boosts performance in open-set recognition along both

Table 3: **Fine-grained OSR performance on the Semantic Shift Benchmark (SCARS, FGVC-Aircraft, and CUB): Closed-set Accuracy** Gains are in blue. The results are given as the mean ± sd over three training runs with fixed seeds.

	CUB			Aircraft			Cars		
Method		+SmoothOP	Δ%		+SmoothOP	Δ%		+SmoothOP	Δ%
SupCON	71.74 ± 1.16	72.29 ± 0.62	**+0.77%**	81.21 ± 1.62	82.99 ± 0.52	**+2.19%**	90.87 ± 0.87	91.40 ± 0.32	**+0.58%**
ConOSR	75.89 ± 0.53	75.42 ± 1.23	**−0.62%**	85.73 ± 0.67	86.19 ± 0.73	**+0.54%**	92.67 ± 0.18	93.26 ± 0.53	**+0.64%**
SpHOR-Lite	74.20 ± 0.88	74.58 ± 0.75	**+0.51%**†	83.53 ± 0.76	83.67 ± 0.66	**+0.17%**	91.80 ± 0.62	91.86 ± 0.45	**+0.07%**
SpHOR	77.05 ± 0.84	77.29 ± 0.94	**+0.31%**	85.83 ± 0.46	85.91 ± 0.24	**+0.09%**	92.93 ± 0.69	93.32 ± 0.51	**+0.42%**†

Table 4: **Feature geometry in terms of the gains in Intra-class compactness (Comp) and Inter-class dispersion (Disp).** Each Δ reflects the gain Comp/Disp when the baseline method was augmented with SmoothOP. Colored in blue means the intervention improved the feature geometry in that aspect. Results over 3 seeds.

	SupCON		ConOSR		SpHOR-Lite		SpHOR	
	Comp. ↓	Disp. ↑	Comp. ↓	Disp. ↑	Comp. ↓	Disp. ↑	Comp. ↓	Disp. ↑
CUB	−0.0052	+0.0111	+0.0005	+0.0035	−0.0135	+0.0004	−0.0152	+0.0256
Aircraft	−0.0167	+0.0166	−0.0088	+0.0169	−0.0059	+0.0126	−0.0148	+0.0360
Cars	−0.0124	+0.0155	−0.0068	+0.0250	−0.0052	+0.0058	−0.0201	+0.0457

axis: it improves separability from unknowns as shown via AUROC, and it also improves closed-set performance (As shown in Table 3). In summation, this net result reflects in improved OSCR in Table 2 across all training-testing regimes. While minor fluctuations occur in specific sub-metrics, likely due to the absence of dataset-specific hyperparameter tuning, the overall consistency across three distinct datasets underscores the generalizability of our approach. We further show it is consistent across the Operating Curves in Appendix C.1 and that it comes at an insignificant performance tax quantified via FLOPS in Appendix C.2.

Empirical Validation: While SmoothOP intervenes via the loss in the projection space, the inference happens in the encoder representations. Thus, to probe the geometric properties of the representations induced by SmoothOP and relate them to OSR, we use two metrics defined in (Li et al., 2022).

Definition 4 (Intra-class compactness (Comp), Inter-class dispersion (Disp))**.** Let $\mathcal{F}_i = \{\mathrm{Enc}(\mathbf{x}) : (\mathbf{x}, y) \in \mathcal{D}_K,\ y = i\}$ and $\bar{\mathbf{v}}_i = \sum_{\mathbf{v}\in\mathcal{F}_i} \mathbf{v} \,/\, \|\sum_{\mathbf{v}\in\mathcal{F}_i} \mathbf{v}\|_2$. Then:

$$\mathrm{Comp} \triangleq \frac{1}{C}\sum_{i=1}^{C} \frac{1}{|\mathcal{F}_i|^2} \sum_{\mathbf{v}_j,\mathbf{v}_k\in\mathcal{F}_i} \|\mathbf{v}_j - \mathbf{v}_k\|_2 \quad (\downarrow), \qquad \mathrm{Disp} \triangleq \frac{1}{C(C-1)} \sum_{i\neq j} \|\bar{\mathbf{v}}_i - \bar{\mathbf{v}}_j\|_2 \quad (\uparrow). \tag{11}$$

We observe that, in Table 4, it produces compact, well-separated geometry that Section 4 links to reduced open-space risk. SmoothOP reduces Comp in 11 of 12 settings and increases Disp in all 12, so the tighter classes do not come at the cost of inter-class separation. The geometry is also consistent with open-set performance. The single setting where Comp does not improve, ConOSR on CUB, also regresses in the easy split under every post-processor (Table 2). Conversely, SpHOR, which shows the largest and most consistently significant geometric gains, also shows the most consistent open-set improvements. The feature geometry is analyzed further using the lens of Angular Separability and Norm Separability in Appendix D.4 (Bahavan et al., 2026). Extended Ablations are presented in Appendix D.

7 Limitations and Future Work

SmoothOP redistributes label smoothing around a base coefficient ϵ rather than adding smoothing of its own, so its effect scales with how much smoothing the base objective uses. objectives tuned to little or no smoothing leave it little to redistribute. Thus, the next natural step is to make the base coefficient ϵ adaptive.

AI USE STATEMENT

Generative AI tools were only used in the preparation of this manuscript for purposes of grammar checking and to verify mathematical claims. No AI systems were used for data analysis, model development, or experimental design. All research decisions and scientific contributions remain the responsibility of the authors.

ETHICS STATEMENT

The research conforms with the ICLR Code of Ethics. This work involves standard image classification benchmarks with no personal data, sensitive content, or harmful applications.

REPRODUCIBILITY STATEMENT

We use only the public Semantic Shift Benchmark with its published splits. All hyperparameters, backbone initialisation and hardware are listed in Appendix B, and every number is the mean over three fixed seeds (1, 2, 3). Code and configuration files are also included in the supplementary materiels.

REFERENCES

Thiru Thillai Nadarasar Bahavan, Sachith Seneviratne, and Saman Halgamuge. Sphor: A representation learning perspective on open-set recognition for identifying unknown classes in deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 6901–6910, June 2026.

Abhijit Bendale and Terrance E. Boult. Towards open set deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1563–1572, 2016.

Guangyao Chen, Limeng Qiao, Yemin Shi, Peixi Peng, Jia Li, Tiejun Huang, Shiliang Pu, and Yonghong Tian. Learning open set network with discriminative reciprocal points. In *Proceedings of the European Conference on Computer Vision*, pp. 507–522, 2020a.

Guangyao Chen, Peixi Peng, Xiangqian Wang, and Yonghong Tian. Adversarial reciprocal points learning for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8065–8081, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 2020b.

Steve Cruz, Ryan Rabinowitz, Manuel Günther, and Terrance E. Boult. Operational open-set recognition and postmax refinement. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part VI*, pp. 475–492, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-72657-6. doi: 10.1007/978-3-031-72658-3_27. URL https://doi.org/10.1007/978-3-031-72658-3_27.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 702–703, 2020.

Akshay Raj Dhamija, Manuel Günther, and Terrance E. Boult. Reducing network agnostophobia. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 9175–9186, Red Hook, NY, USA, 2018. Curran Associates Inc.

Thomas G. Dietterich and Alexander Guyer. The familiarity hypothesis: Explaining the behavior of deep open set methods. *Pattern Recognition*, 132:108931, 2022. doi: 10.1016/j.patcog.2022.108931.

Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Hongzhi Huang, Yu Wang, Qinghua Hu, and Ming-Ming Cheng. Class-specific semantic reconstruction for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45 (4):4214–4228, 2023.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Shu Kong and Deva Ramanan. Opengan: Open-set recognition via open data generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–10, 2022. doi: 10.1109/TPAMI.2022.3184052.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 554–561. IEEE, 2013.

Nico Lang, Vésteinn Snæbjarnarson, Elijah Cole, Oisin Mac Aodha, Christian Igel, and Serge Belongie. From coarse to fine-grained open-set recognition. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17804–17814, 2024. doi: 10.1109/CVPR52733.2024.01686.

Tianhong Li, Peng Cao, Yuan Yuan, Lijie Fan, Yuzhe Yang, Rogerio Feris, Piotr Indyk, and Dina Katabi. Targeted supervised contrastive learning for long-tailed recognition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6908–6918, 2022. doi: 10.1109/CVPR52688.2022.00679.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. URL `https://arxiv.org/abs/1306.5151`.

Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 613–628, September 2018.

Poojan Oza and Vishal M. Patel. C2ae: Class conditioned auto-encoder for open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2302–2311, 2019. doi: 10.1109/CVPR.2019.00241.

Jaewoo Park, Yoon Gyo Jung, and Andrew Beng Jin Teoh. Nearest neighbor guidance for out-of-distribution detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1686–1695, 2023.

Pramuditha Perera, Vlad I. Morariu, Rameswar Jain, Varun Manjunatha, Curtis Wigington, Vicente Ordonez, and Vishal M. Patel. Generative-discriminative feature representations for open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11814–11823, 2020.

Walter J. Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E. Boult. Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7): 1757–1772, 2013. doi: 10.1109/TPAMI.2012.256.

Xin Sun, Zhenning Yang, Chi Zhang, Keck-Voon Ling, and Guohao Peng. Conditional gaussian distribution learning for open set recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13477–13486, 2020. doi: 10.1109/CVPR42600.2020.01349.

Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. Out-of-distribution detection with deep nearest neighbors. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pp. 20827–20840. PMLR, 2022. URL https://proceedings.mlr.press/v162/sun22d.html.

Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: A good closed-set classifier is all you need. In *Proceedings of the International Conference on Learning Representations*, pp. 1–14, 2022. URL https://openreview.net/forum?id=5hLP5JY9S2d.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011. URL https://authors.library.caltech.edu/records/cvm3y-5hh21.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Yu Wang, Junxian Mu, Pengfei Zhu, and Qinghua Hu. Exploring diverse representations for open set recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5731–5739, 2024.

Baile Xu, Furao Shen, and Jian Zhao. Contrastive open set recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 10546–10556, 2023. doi: 10.1609/aaai.v37i9.26253.

Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, Qing Yang, and Cheng-Lin Liu. Convolutional prototype network for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2358–2370, 2022. doi: 10.1109/TPAMI.2020.3045079.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Da-Wei Zhou, Han-Jia Ye, and De chuan Zhan. Learning placeholders for open-set recognition. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4399–4408, 2021.

A Evaluation Metrics

To begin, we define a supervised training set for a classification task as $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \subset \mathcal{X} \times \mathcal{C}$, where $\mathcal{X}$ is the image space and $\mathcal{C}$ is the set of 'known' classes.

In the closed-set scenario, the model is evaluated on a test set where the labels are strictly drawn from the training categories: $\mathcal{D}_{\text{M}} := \mathcal{D}_k = \{(\mathbf{x}_j, y_j)\}_{j=1}^{M} \subset \mathcal{X} \times \mathcal{C}$. In this setting, the model returns a distribution over known classes as $p(y \mid \mathbf{x})$.

Conversely, in Open-Set Recognition (OSR), test images may also originate from unseen classes $\mathcal{U}$, where $\mathcal{U}$ denotes the set of unknown classes and $\mathcal{C} \cap \mathcal{U} = \emptyset$. The OSR-evaluation test set is a union of known and unknown samples: $\mathcal{D}_{\text{Open}} = \mathcal{D}_k \cup \mathcal{D}_u$. The unknown component is defined as: $\mathcal{D}_u = \{(\mathbf{x}_j, y_j)\}_{j=1}^{K} \subset \mathcal{X} \times \mathcal{U}$.

In the open-set setting, the model provides both a class distribution $p(y \mid \mathbf{x}, y \in \mathcal{C})$ and a score function $\mathcal{S}(\mathbf{x})$ to indicate whether or not the test sample belongs to any of the known classes. The final prediction $\hat{y}$ is determined by an operational threshold θ:

$$\hat{y} = \begin{cases} \text{argmax}_{y \in \mathcal{C}}\ p(y \mid \mathbf{x}) & \text{if } \mathcal{S}(\mathbf{x}) \geq \theta \\ \text{Unknown} & \text{if } \mathcal{S}(\mathbf{x}) < \theta \end{cases} \tag{12}$$

We evaluate models with three metrics:

Standard **Closed-set Accuracy** is:

$$\text{Acc} = \frac{1}{|\mathcal{D}_k|} \sum_{i=1}^{|\mathcal{D}_k|} \mathbb{I}(\hat{y}_i = y_i) \tag{13}$$

The **Area Under the Receiver Operating Characteristic (AUROC)** is a threshold-independent metric utilized to evaluate the model's capability to discriminate between known classes $\mathcal{C}$ and unknown classes $\mathcal{U}$ based on the scoring function $\mathcal{S}(\mathbf{x})$. However, this metric is flawed as it doesn't account for the classification accuracy. Thus, we need to establish a link between AUROC (unknown rejection) and known-class classification across varying thresholds. For a given operational threshold θ, we define the **Correct Classification Rate (CCR)(Dhamija et al., 2018)** and the **False Positive Rate (FPR)(Dhamija et al., 2018)** as follows:

$$\text{CCR}(\theta) = \frac{1}{|\mathcal{D}_k|} \sum_{\mathbf{x}_i \in \mathcal{D}_k} \mathbb{I}(y_i = \hat{y}_i) \cdot \mathbb{I}(\mathcal{S}(\mathbf{x}_i) > \theta), \quad \text{FPR}(\theta) = \frac{1}{|\mathcal{D}_u|} \sum_{\mathbf{x}_i \in \mathcal{D}_u} \mathbb{I}(\mathcal{S}(\mathbf{x}_i) > \theta) \tag{14}$$

Thus, CCR contextualizes Acc for OSR, taking into account if the known class is mistakenly rejected as unknown or if the unknown class is mistakenly classified as known. However, it requires a threshold θ. Based on CCR and FPR, we define two metrics to measure OSR performance: a threshold-independent metric, the Open-set Classification Rate (OSCR), and a threshold-dependent CCR5 (where $FPR = 5\%$).

The **Open-set Classification Rate (OSCR)(Dhamija et al., 2018)** summarizes the performance of a model across all possible operational thresholds θ. It is defined as the area under the curve formed by plotting the Correct Classification Rate (CCR) against the False Positive Rate (FPR):

$$\text{OSCR} = \int_0^1 \text{CCR}(\text{FPR}) d\text{FPR} \tag{15}$$

For many safety-critical applications, this is a standard threshold. While recent OSR literature has introduced metrics such as Open-Set Accuracy (OOSA), we do not employ it in our evaluation. OOSA requires a surrogate set, often from a different dataset, to estimate the operational threshold θ. We argue that this approach does not accurately measure the *Semantic Shift*, as the threshold estimation is decoupled from the actual semantic boundary between known and unknown classes in the target domain, and it requires a validation dataset.

B EXPERIMENTAL DETAILS

B.1 HYPERPARAMETER AND DATA AUGMENTATION

We experimented with different loss functions and hyperparameters tailored to each dataset. A crucial component of our training was the use of **RandAugment**, a data augmentation technique that automatically finds the optimal combination of augmentation policies. The specific hyperparameters used for each loss function and dataset combination are detailed in Table 5. The RandAugment policy is controlled by two main parameters: N, the number of augmentation transformations to apply, and M, the magnitude of the transformations.

Table 5: Hyperparameters for Fine-Grained Datasets. We take the exact parameters from MLS(Vaze et al., 2022)

Loss	Dataset	Image Size	RandAug N	RandAug M	Label Smoothing	Batch Size
SpHOR/Sphor-lite	CUB	224	2	30	0.3	128
SpHOR/Sphor-lite	FGV-Aircraft	224	2	15	0.2	128
SpHOR/Sphor-lite	SCARS	224	2	15	0.2	128
ConOSR/SupCON	CUB	224	2	30	0.3	128
ConOSR/SupCON	FGV-Aircraft	224	2	15	0.2	128
ConOSR/SupCON	SCARS	224	2	15	0.2	128

B.2 STAGE 02 TRAINING

We freeze the stage-1 backbone and retrain only the linear classifier $\mathrm{Linear}(2048 \to C)$ on its features, starting from the stage-1 head. We pass each training image through the frozen network once, with the same RandAugment setting as stage 1, and cache the 2048-d features. The head then trains on this cache for 20 epochs with batch size 128, using standard cross-entropy and no label smoothing. We optimise with Adam at learning rate 10^{-4}, with no weight decay and no learning-rate schedule. The head is retrained once per stage-1 checkpoint, so each seed gets its own head, and every open-set score is computed from that head.

B.3 DATASET

The fundamental challenge in OSR is distinguishing between Covariate Shift and Semantic Shift. While covariate shift refers to changes in the marginal input distribution $P_{\text{test}}(\mathbf{x}) \neq P_{\text{train}}(\mathbf{x})$ without changing label semantics $P_{\text{test}}(y \mid \mathbf{x}) = P_{\text{train}}(y \mid \mathbf{x})$, OSR induces a semantic shift. Here, the conditional label distribution changes due to novel classes $P_{\text{test}}(y \mid \mathbf{x}) \neq P_{\text{train}}(y \mid \mathbf{x})$, while the input domain ideally remains static $P_{\text{test}}(\mathbf{x}) \approx P_{\text{train}}(\mathbf{x})$.

We conducted experiments on the Semantic Shift Benchmark (SSB) (Vaze et al., 2022), which uses three fine-grained classification datasets: CUB (Wah et al., 2011), FGVC-Aircraft (Krause et al., 2013), and Stanford Cars (Maji et al., 2013). For each dataset, SSB partitions the original categories into *known* classes used for training and *unknown* classes reserved for evaluation.

CUB-200-2011. CUB-200-2011 is a fine-grained bird classification dataset containing 11,788 images from 200 bird species. In the SSB protocol, 100 species are treated as known classes, while the remaining species are divided into multiple groups of unknown classes according to semantic similarity. The dataset is challenging due to subtle inter-class differences and limited training samples per class.

Stanford Cars. The Stanford Cars dataset contains 16,185 images across 196 car model categories. Following the SSB setup, 98 classes are used as known categories, and the remaining classes serve as unknown categories under different semantic shift levels. The dataset includes large viewpoint variations and fine-grained distinctions at the level of production year and model.

FGVC-Aircraft. FGVC-Aircraft consists of 10,000 images over 100 aircraft variants. The dataset provides a semantic hierarchy from manufacturer to model. 50 aircraft categories are used as known classes, while the remaining categories are treated as unknown classes. Semantic partitions are constructed based on hierarchical similarity.

Table 6: Statistics of datasets used in the Semantic Shift Benchmark (SSB). Known classes are used for training, while unknown classes are reserved for open-set evaluation.

Dataset	Total Images	Total Classes	Known Classes	Unknown Classes
CUB	11,788	200	100	100
Stanford Cars	16,185	196	98	98
FGVC-Aircraft	10,000	100	50	50

B.4 Scoring Functions

Let $f(\mathbf{x}) \in \mathbb{R}^C$ be the classifier logits over C classes, and let $\mathbf{z} = \phi(\mathbf{x})$ be the feature vector taken from the layer just before the classifier.

MSP. The largest softmax probability over all classes:

$$S_{\mathrm{MSP}}(\mathbf{x}) = \max_c \frac{\exp f_c(\mathbf{x})}{\sum_j \exp f_j(\mathbf{x})}. \tag{16}$$

MLS. The largest logit over all classes:

$$S_{\mathrm{MLS}}(\mathbf{x}) = \max_c f_c(\mathbf{x}). \tag{17}$$

PostMax. PostMax normalizes the maximum logit by the feature magnitude and maps it through the cumulative distribution of a Generalized Pareto Distribution (GPD) fit on correctly classified training samples:

$$S_{\mathrm{PostMax}}(\mathbf{x}) = \Psi_{\mu,\sigma,\xi}\left(\frac{\max_c f_c(\mathbf{x})}{\|\mathbf{z}\|}\right), \tag{18}$$

where $\Psi_{\tilde{\mu},\tilde{\sigma},\tilde{\xi}}$ is the GPD CDF with location $\tilde{\mu}$, scale $\tilde{\sigma}$, and shape $\tilde{\xi}$ estimated by maximum likelihood on the normalized max-logits of correctly classified training samples. Dividing by $\|\mathbf{z}\|$ counteracts the observation that unknown samples on large-scale models tend to have larger feature norms, and the GPD turns the score into a calibrated probability.

NNGuide. Multiply a base classifier score S_{base} (we use negative energy, $S_{\mathrm{base}}(\mathbf{x}) = \log \sum_c \exp f_c(\mathbf{x})$) by the average of the top-k confidence-weighted similarities:

$$S_{\mathrm{NNGuide}}(\mathbf{x}) = S_{\mathrm{base}}(\mathbf{x}) \cdot \frac{1}{k} \sum_{i=1}^{k} s_{(i)} \, \mathrm{sim}(\mathbf{z}_{(i)}, \mathbf{z}), \tag{19}$$

where $s_i = S_{\mathrm{base}}(\mathbf{x}_i)$ is the base score of bank sample i, and the indices (i) are sorted by descending $s_i \cdot \mathrm{sim}(\mathbf{z}_i, \mathbf{z})$. Weighting neighbors by their confidence pulls the search toward high-confidence ID regions, which lowers scores on far-OOD inputs while still separating near-OOD inputs around class boundaries.

C Extended Results

C.1 Operating Curve Analysis

Robustness Across a Wide Range of Operating Thresholds: The Open-Set Classification Rate curve characterizes the trade-off between accurately classifying known-class samples (CCR) and correctly rejecting unknowns (FPR) across varying operational thresholds θ. As illustrated in Fig. 2, 3, 4, 5, our “Enhanced" variants reliably shift the OSCR curves upward for the hard sample splits. This indicates that by focusing on hard positives during training, the model maintains a higher CCR at lower FPR levels. This consistent performance gain across thresholds confirms that our strategy effectively improves open-set recognition. This further validates our hypothesis.

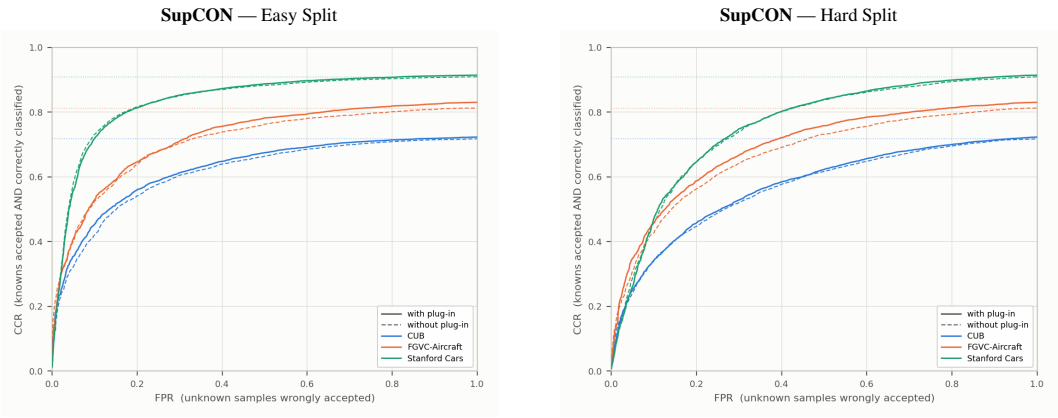


Figure 2: **OSCR operating curves for SupCON against SSB with the MSP post-processor. Solid** indicates with SmoothOP ; **dashed** is the baseline without the plug-in. The curves are vertically averaged over 3 seeds for a fixed FPR. The results are shown for both the Easy and Hard Splits. SmoothOP pushes out baselines almost consistently across all operating points.

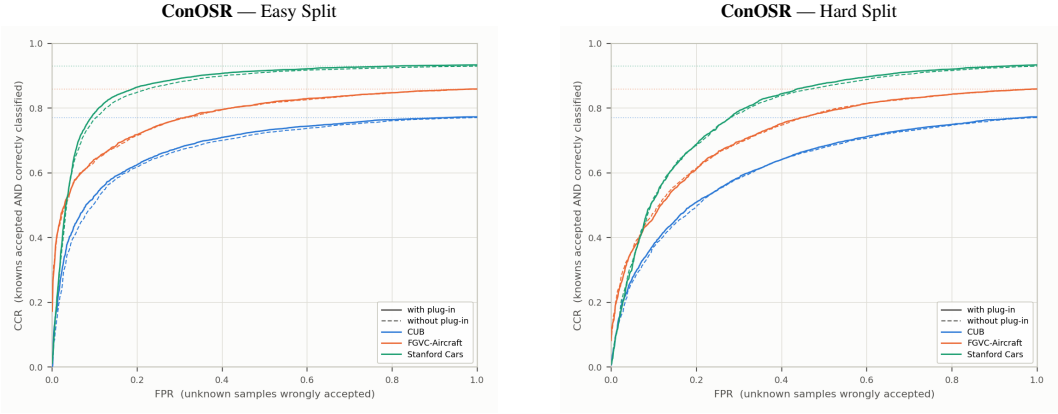


Figure 3: **OSCR operating curves for ConOSR against SSB with the MSP post-processor. Solid** indicates with SmoothOP ; **dashed** is the baseline without the plug-in. The curves are vertically averaged over 3 seeds for a fixed FPR. The results are shown for both the Easy and Hard Splits. SmoothOP pushes out baselines almost consistently across all operating points.

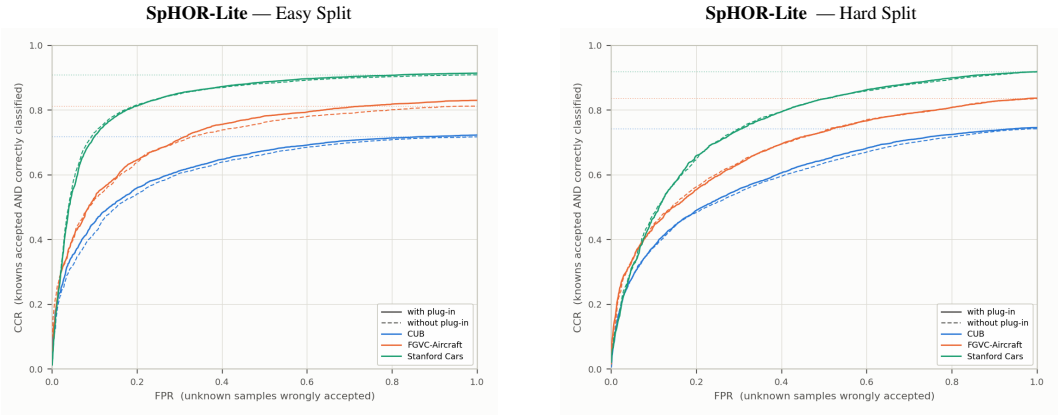


Figure 4: **OSCR operating curves for SpHOR-Lite against SSB with the MSP post-processor. Solid** indicates with SmoothOP ; **dashed** is the baseline without the plug-in. The curves are vertically averaged over 3 seeds for a fixed FPR. The results are shown for both the Easy and Hard Splits. SmoothOP pushes out baselines almost consistently across all operating points.

Table 7: Performance cost of the Observer classifier on the ResNET-50 backbone. For heavier backbones, the cost will become even smaller.

Dataset	C	Head params	% of backbone params	% of train step
CUB	100	204 900	0.80%	0.010%
FGVC-Aircraft	50	102 450	0.40%	0.005%
Stanford Cars	98	200 802	0.79%	0.010%

C.2 FLOPs for Observer classifier

It's important to quantify the performance tax when augmenting the conventional training with SmoothOP. Given that the only difference is that an additional Observer classifier which supplies the u_i is training in parallel with the main-backbone using the cross-entropy loss. Note that this is independent of the any hyper-parameters. Table 7 reports the training cost regardless in terms of FLOPs. As observed, the cost is almost negligible.

D Extended Ablations and Discussion

D.1 Ablation: Inter-batch and Intra-batch variation decomposed.

SmoothOP varies the amount of smoothing along two axes: (1) Intra-batch, across the samples in a batch at a given step; and (2) Inter-batch, across batches over the course of training. The inter-batch component is simply the average smoothing per batch, which acts as a label smoothing schedule. The question for this ablation is whether the intra-batch component matters, or whether the schedule alone accounts for the gains. Recall the per-sample coefficient from Section 4.3:

$$\varepsilon_i \;=\; \varepsilon \cdot 2\,\mathrm{sigmoid}(-|\alpha| u_i) \tag{8}$$

Here ε and α are shared across the batch, so u_i is the only source of per-sample variation in ε_i. The intra-batch effect therefore comes down to which sample receives which coefficient. To test it, we compare the standard method (treatment) against two controls that take the computed ε_i and redistribute them within the batch before the loss is computed.

Treatment This is the standard SmoothOP algorithm. We denote P^{treat} as the test performance (OSCR) of a model trained under this regime.

Batch mean Instead of the per-sample ε_i, every sample in the batch receives the batch mean: $\varepsilon_i^{\text{bm}} = \frac{1}{B}\sum_j \varepsilon_j$. We denote $P^{batch-mean}$ as the test performance (OSCR) of a model trained under this regime. We also define $\Delta^{batch-mean} = P^{treat} - P^{batch-mean}$, which measures how much of the gain is from the intra-batch variation and assignment.

Reversed Assignment Batch mean removes the intra-batch variation entirely, so it cannot tell us whether the variation helps because the right samples get the right coefficients. To test this, we keep each batch's coefficients but reverse their order: $\varepsilon_i^{\text{rev}} = \varepsilon_{(n+1-r_i)}$, where r_i is the rank of ε_i and $\varepsilon_{(k)}$ is the k-th smallest coefficient. This arrangement preserves the batch-mean effect and the intra-batch variation, but inverts the assignment. We denote $P^{reverse}$ as the test performance (OSCR) of a model trained under this regime. We also define $\Delta^{reverse} = P^{treat} - P^{reverse}$, which measures how much the correct assignment matters compared to the inverted one.

In all three regimes, all other hyperparameters, base coefficients, and seeds are kept the same as in standard training.

If the per-sample effect exists and is not a confound of the batch-mean schedule, we should see positive gains Δ^{rev} and Δ^{bm}. Ideally, we expect that $\Delta^{\text{rev}} > \Delta^{\text{bm}} > 0$. Removing the assignment should hurt, but inverting it should hurt more.

As observed in Table 8, for method–dataset combinations, the gains averaged across post-processors are positive, which indicates that the effect exists by and large. We observe that our prediction

		OSCR, mean of 3 seeds				treatment − control	
arm	data	Baseline	+SmoothOP	+Batch-mean	+Rev.As.	$\Delta^{Batch-mean}$	$\Delta^{reverse}$
easy							
SupCON	CUB	59.67	60.83	60.46	60.58	+0.37	+0.25
	Aircraft	70.87	72.12	71.91	70.88	+0.22	+1.25
	Cars	82.40	82.92	82.97	82.69	−0.05	+0.23
ConOSR	CUB	64.05	63.67	63.81	64.20	−0.14	−0.52
	Aircraft	77.17	78.11	76.64	77.57	+1.47	+0.54
	Cars	84.69	85.42	84.96	84.83	+0.45	+0.59
SpHOR-Lite	CUB	63.67	64.64	65.01	64.77	−0.37	−0.13
	Aircraft	69.62	69.81	70.06	69.27	−0.25	+0.55
	Cars	84.00	84.17	84.09	83.96	+0.07	+0.21
SpHOR	CUB	66.76	67.56	67.30	67.79	+0.26	−0.23
	Aircraft	77.67	77.89	78.08	78.08	−0.19	−0.19
	Cars	85.64	86.51	86.16	86.07	+0.36	+0.44
mean over 12 blocks						+0.18	+0.25
hard							
SupCON	CUB	54.37	55.01	54.93	54.38	+0.08	+0.63
	Aircraft	66.16	68.60	67.81	67.68	+0.79	+0.92
	Cars	75.13	75.17	76.05	75.25	−0.88	−0.08
ConOSR	CUB	57.06	57.11	57.05	57.32	+0.06	−0.21
	Aircraft	70.72	71.29	70.35	70.34	+0.94	+0.95
	Cars	77.44	77.84	77.68	77.26	+0.17	+0.59
SpHOR-Lite	CUB	57.62	58.43	58.57	57.95	−0.14	+0.48
	Aircraft	67.28	67.27	67.49	66.93	−0.22	+0.34
	Cars	75.64	75.71	76.17	75.76	−0.46	−0.05
SpHOR	CUB	59.47	60.34	60.19	60.23	+0.15	+0.11
	Aircraft	70.85	71.00	71.05	70.98	−0.04	+0.03
	Cars	77.99	78.44	78.25	78.05	+0.20	+0.40
mean over 12 blocks						+0.05	+0.34

Table 8: OSCR averaged over MSP, MLS, NNGuide-5 and PostMax across 3 seeds. The three regimes are Batch mean (+Batch-mean), Reversed Assignment (+Rev.As.) and Treatment (+SmoothOP). To isolate the effectiveness of the per-sample assignment, we calculate two gains by subtracting the Reversed Assignment/Batch mean regime from the SmoothOP regime, which highlights the gain in terms of the per-sample smoothing. If the gain is positive, it is marked in blue and is supportive of our hypothesis.

$\Delta^{\text{rev}} > \Delta^{\text{bm}} > 0$ holds on average ($+0.25 > +0.18$ on easy, $+0.34 > +0.05$ on hard). Inverting the assignment is more harmful to the method than removing it. However, both Reversed Assignment and Batch mean have the same mean statistics. It must be noted that the negative gains are not uniformly distributed but concentrated in some methods. For example, for SpHOR-Lite (without Mixup), 5 out of 6 blocks for Δ^{bm} are negative and can be attributed purely to the smoothing schedule; however, in practice SpHOR is used over SpHOR-Lite, since the latter is an ablation.

D.2 Temporal Behavior of the Smoothing Method

In this section, we are interested in tracking the behavior of ε_i, and specifically how it behaves in a temporal sense. Note that ε_i is not stationary, as it is a function of the model's own prominence u_i. As the model trains, it becomes more confident, and u drifts toward -1, and ε is boosted. To observe how ε behaves, we plot the two information in a temporal chart: (a) the batch-mean value of ε; (b) inter-decile (p_{10}–p_{90}) range across the batch for ε. We use this instead of a standard deviation because the distribution is skewed towards the upper bound (ε is bounded). This is illustrated for all method-dataset combinations in Fig. 6.

Based on this chart, we can come with the following conclusions. The SmoothOP does not give a constant intervention during the training. It is time-varying and gradually increases. For example, for SpHOR/CUB it goes from $\lambda = 0.3$ to 0.524. This is an expected behavior as the model becomes more confident. From another point of view, this can also be seen as a form of regularization as the

model becomes more confident. In addition to an adaptive per-sample smoothing method, the method also functions as a time-varying smoothing schedule.

D.3 Ablation: Mixing smoothing coefficients vs mixing margins.

In Section 4.3, we use the following scheme, which we refer to as **Margin Blending**, for Mixup. The constituent margins are read from the forward pass and retrieved through the mixing indices:

$$\mathrm{u}_{i,j}^{\lambda} = \lambda \cdot \mathrm{u}_i + (1-\lambda) \cdot \mathrm{u}_j \tag{20}$$

This composite margin induces a single smoothing coefficient, applied once to the mixed target:

$$\varepsilon_{i,j}^{\lambda} \;=\; 2\varepsilon \cdot \mathrm{sigmoid}\big(-|\alpha|\,\mathrm{u}_{i,j}^{\lambda}\big), \qquad \varepsilon_{i,j}^{\lambda} \in (0,\, 2\varepsilon) \tag{21}$$

$$\hat{\mathbf{y}}_k^{'} = (1-\varepsilon_{i,j}^{\lambda})\big(\lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j\big) + \frac{\varepsilon_{i,j}^{\lambda}}{C} \cdot \mathbf{1}, \tag{22}$$

where $\mathbf{y}_i, \mathbf{y}_j$ are the *one-hot* targets of the constituent samples.

However, it can be argued that the coefficients can be blended as well. Thus, we ablate it against an alternative scheme known as **coefficient blending:**

$$\mathrm{u}_{i,j}^{\lambda} = \lambda \cdot \mathrm{u}_i + (1-\lambda) \cdot \mathrm{u}_j \tag{23}$$

This composite margin induces a single smoothing coefficient, applied once to the mixed target:

$$\varepsilon_i \;=\; 2\varepsilon \cdot \mathrm{sigmoid}\big(-|\alpha|\,\mathrm{u}_i\big), \qquad \varepsilon_i \in (0,\, 2\varepsilon) \tag{24}$$

$$\varepsilon_j \;=\; 2\varepsilon \cdot \mathrm{sigmoid}\big(-|\alpha|\,\mathrm{u}_j\big), \qquad \varepsilon_j \in (0,\, 2\varepsilon) \tag{25}$$

$$\varepsilon_{i,j}^{\lambda} \;=\; \lambda \cdot \varepsilon_i + (1-\lambda) \cdot \varepsilon_j, \qquad \varepsilon_{i,j}^{\lambda} \in (0,\, 2\varepsilon) \tag{26}$$

$$\hat{\mathbf{y}}_k^{'} = (1-\varepsilon_{i,j}^{\lambda})\big(\lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j\big) + \frac{\varepsilon_{i,j}^{\lambda}}{C} \cdot \mathbf{1}, \tag{27}$$

where $\mathbf{y}_i, \mathbf{y}_j$ are the *one-hot* targets of the constituent samples.

As observed in Table 9, **Margin blending wins.** It is ahead on 11 of the 12 cells in all dataset-method combinations in OSCR Hard, and by $+0.27$ OSCR points on both splits. The formula we used for blending margins turns out to be better in practice, but the success of our method doesn't depend on it.

		OSCR-Easy			OSCR-Hard		
Objective	Dataset	coeff.	margin	Δ	coeff.	margin	Δ
ConOSR	CUB	65.41	65.19	−0.23	59.72	59.86	+0.14
ConOSR	Aircraft	76.56	76.94	+0.38	71.70	72.09	+0.38
ConOSR	Cars	86.05	86.13	+0.08	78.05	78.17	+0.12
SpHOR	CUB	67.80	68.26	+0.45	61.98	62.20	+0.21
SpHOR	Aircraft	76.85	77.55	+0.70	71.50	72.04	+0.53
SpHOR	Cars	86.92	87.13	+0.21	78.80	79.05	+0.25
Mean over the six arms		76.60	76.86	+0.27	70.29	70.57	+0.27

Table 9: **Margin blending beats coefficient blending on** 11 **of** 12 **cells.** OSCR under MSP at shared hyperparameters, mean over 3 seeds; Δ = margin blending − coeff. blending, red where margin blending is ahead.

D.4 How does SmoothOP optimize representation geometry for separability?

While intra-class alignment and inter-class uniformity assess representations within known classes, they fail to evaluate how effectively feature representations distinguish known classes from unknown ones. To address this, Bahavan et al. (2026) proposed Angular Separability (AS) and Norm Separability (NS):

		AS ↓			NS ↑		
arm	data	w/o	+SmoothOP	gain	w/o	+SmoothOP	gain
easy							
SupCON	CUB	**0.8793**	0.8796	−0.0003	69.22	**72.21**	+2.99
	Aircraft	**0.9130**	0.9159	−0.0030	77.59	**78.07**	+0.48
	Cars	**0.8959**	0.8987	−0.0027	**84.98**	84.14	−0.84
ConOSR	CUB	0.8771	**0.8766**	+0.0005	**68.94**	68.52	−0.42
	Aircraft	**0.8939**	0.8948	−0.0010	68.05	**72.62**	+4.57
	Cars	0.8804	0.8804	0.0000	79.95	**81.98**	+2.03
SpHOR-Lite	CUB	**0.8589**	0.8695	−0.0106	68.24	**72.16**	+3.92
	Aircraft	**0.9002**	0.9061	−0.0060	63.49	**65.38**	+1.89
	Cars	**0.8837**	0.8865	−0.0028	83.25	**83.99**	+0.74
SpHOR	CUB	**0.8356**	0.8425	−0.0069	56.60	**65.35**	+8.75
	Aircraft	**0.8618**	0.8668	−0.0050	60.29	**64.07**	+3.78
	Cars	**0.8543**	0.8551	−0.0008	63.13	**70.54**	+7.41
mean over 12 blocks (+SmoothOP ahead)				−0.0032 (1)			+2.94 (10)
hard							
SupCON	CUB	**0.8968**	0.9005	−0.0038	65.08	**65.72**	+0.64
	Aircraft	**0.9298**	0.9307	−0.0009	73.71	**74.83**	+1.12
	Cars	**0.9212**	0.9221	−0.0009	75.24	**75.90**	+0.66
ConOSR	CUB	0.8952	**0.8947**	+0.0005	65.66	**66.06**	+0.40
	Aircraft	**0.9135**	0.9177	−0.0042	72.14	**72.46**	+0.33
	Cars	**0.9006**	0.9038	−0.0032	72.58	**73.84**	+1.26
SpHOR-Lite	CUB	**0.8736**	0.8838	−0.0102	64.82	**66.74**	+1.92
	Aircraft	**0.9042**	0.9090	−0.0049	69.61	**70.37**	+0.77
	Cars	**0.9031**	0.9063	−0.0032	74.08	**74.54**	+0.46
SpHOR	CUB	**0.8566**	0.8641	−0.0075	58.39	**62.59**	+4.20
	Aircraft	**0.8800**	0.8847	−0.0047	65.80	**68.00**	+2.20
	Cars	**0.8761**	0.8806	−0.0044	61.49	**66.42**	+4.93
mean over 12 blocks (+SmoothOP ahead)				−0.0039 (1)			+1.57 (12)

Table 10: Angular Separability (AS, Eq. 28, lower is better) and Norm Separability (NS, Eq. 29, AUROC in %, higher is better) on the test set, mean of 3 seeds, without and with +SmoothOP. Within each pair the better level is bold. Gain is the improvement in the metric's own direction, w/o − +SmoothOP for AS and +SmoothOP − w/o for NS, paired by seed; a positive gain is red.

$$\mathrm{AS} \;=\; \frac{1}{|D_u|}\sum_{\mathbf{u}\in D_u}\max_{\mathbf{v}\in D_k}\left(\frac{\mathbf{u}\cdot\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}\right) \qquad (\downarrow), \tag{28}$$

$$\mathrm{NS} \;=\; \mathrm{AUROC}\big(\{\|\mathbf{v}\| : \mathbf{v}\in D_k\}, \{\|\mathbf{u}\| : \mathbf{u}\in D_u\}\big) \qquad (\uparrow). \tag{29}$$

Angular Separability (AS) (Eq. 28) measures the average cosine similarity between each unknown sample ($\mathbf{u} \in D_u$) and its nearest known-class counterpart ($\mathbf{v} \in D_k$) in the test set (Bahavan et al., 2026). It quantifies the degree to which unknown samples fall into the "familiarity trap," where lower values indicate better angular separation between known and unknown classes.

Norm Separability (NS) (Eq. 29) evaluates the AUROC obtained by using feature vector norms to distinguish known-class test samples from unknown-class samples (Bahavan et al., 2026). Higher values signify that unknown classes can be easily detected via a simple, class-agnostic threshold on feature magnitudes.

Table 10 shows a clear trade-off. Adding SmoothOP improves norm separability (NS) by +2.94 AUROC points on Easy and +1.57 on Hard on average. This comes at a small cost in angular separability (AS), which drops by less than 0.5% of its value on both splits (0.0032 on Easy and 0.0039 on Hard, against levels of 0.84–0.93) The NS gain is several times larger than the AS loss, and the net effect is better unknown rejection.

D.5 Hyperparameter Analysis

Validation Rule We design a "validation rule" to find the optimal hyperparameter α^* in the absence of a test dataset. We take the training dataset split into 4-fold to get the validation known class set and the training dataset. We train the model across a range of $\alpha^* \in \{8, 4, 3, 2, 1.5, 1, 0.5\}$. Afterwards, we use a different fine-grained dataset's images as the unknown class validation set (CUB$\rightarrow$ Cars $\rightarrow$ Aircraft $\rightarrow$ CUB). After training, for each of the split, using the MSP post-processor, we take the argmax over the OSCR performance with the known and unknown set.

We pick the optimal α based on the best performance on these curves as shown in Figure 7 and Figure 8.

E Methodology: Integration

We first state SmoothOP as a mechanism on the label space, then the four base methods in their published form (Section E.1), and finally the effect of the smoothing factor ε_i on each of them. Every base method is recovered exactly at $\varepsilon_i \equiv 0$.

SmoothOP is a transformation on the label space Δ^{C-1}, applied independently of the loss function. For a sample i with smoothing factor $\varepsilon_i \in [0, 1]$:

$$\mathbf{y}_i' := \mathrm{SmoothOP}(\mathbf{y}_i) = (1 - \varepsilon_i)\,\mathbf{y}_i + \frac{\varepsilon_i}{C} \cdot \mathbf{1}, \qquad \varepsilon_i \in [0, 1], \tag{30}$$

where $\mathbf{1} \in \mathbb{R}^C$ is a vector of ones.

This transformation has two key properties:

Simplex closure: If $\mathbf{y}_i \in \Delta^{C-1}$, then $\mathbf{y}_i' \in \Delta^{C-1}$ always sums to 1:

$$\sum_{k=1}^{C} y_{ik}' = 1 \tag{31}$$

Endpoints: The transformation reduces to the original one-hot vector or the uniform distribution at its limits:

$$\mathbf{y}_i'\big|_{\varepsilon_i=0} = \mathbf{y}_i, \qquad \mathbf{y}_i'\big|_{\varepsilon_i=1} = \frac{1}{C}\mathbf{1} \tag{32}$$

E.1 Base Methods

E.1.1 SpHOR

The SpHOR framework employs the Von Mises-Fisher Alignment Loss ($\mathcal{L}_{\mathrm{vMFAL}}$) to map features onto a hyperspherical embedding space. For a dataset of N samples (x_i, y_i), the loss is defined as:

$$\mathcal{L}_{\mathrm{vMFAL}} := -\frac{1}{N} \sum_{i=1}^{N} \underbrace{\sum_{k=1}^{C} S_{ik} \log P_{ik}}_{=:\mathcal{L}_{\mathrm{vMFAL}}^{(i)}} \tag{33}$$

where the predicted probability P_{ik} for sample i and class k is given by the centered cosine similarity:

$$P_{ik} = \frac{\exp(z_i^\top \boldsymbol{\mu}_k / \tau)}{\sum_{j=1}^{C} \exp(z_i^\top \boldsymbol{\mu}_j / \tau)} \tag{34}$$

In this formulation, $\boldsymbol{\mu}_k$ denotes the label embedding for class k, and τ is a temperature scaling parameter. Both the latent feature vector z_i and the label embeddings $\boldsymbol{\mu}_k$ are L_2-normalized. The term S_{ik} represents the soft label similarity, defined as:

$$S_{ik} = \frac{y_{ik}}{\sum_{j=1}^{C} y_{ij}} = y_{ik} \tag{35}$$

To simulate the open space, SpHOR additionally trains on Mixup-generated virtual samples. For a pair (a, b) and $\lambda \sim \mathrm{Beta}(2, 2)$,

$$\tilde{\mathbf{x}}_j = \lambda \mathbf{x}_a + (1-\lambda)\mathbf{x}_b, \qquad \tilde{\mathbf{y}}_j = \lambda \mathbf{y}_a + (1-\lambda)\mathbf{y}_b \ \in \ \Delta^{C-1}, \tag{36}$$

and $\mathcal{L}_{\mathrm{vMFAL}}^{\mathrm{Mixup}(j)}$ is the per-sample term of (33) evaluated at $(\tilde{z}_j, \tilde{S}_{jk} = \tilde{y}_{jk})$:

$$\mathcal{L}_{SpHOR} := -\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\mathrm{vMFAL}}^{(i)} \ - \ \frac{1}{N}\sum_{j=1}^{N} \mathcal{L}_{\mathrm{vMFAL}}^{\mathrm{Mixup}(j)} \ + \ R(\{\boldsymbol{\mu}_k\}_{k=1}^{C}) \tag{37}$$

R here is a regularization loss based on the label embeddings.

E.1.2 Ablation without Mixup: SpHOR-Lite

To verify that our results do not hinge on Mixup, we also report SpHOR-Lite, which drops the Mixup term from Eq. (37):

$$\mathcal{L}_{SpHOR-Lite} := -\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\mathrm{vMFAL}}^{(i)} + \ R(\{\boldsymbol{\mu}_k\}_{k=1}^{C}) \tag{38}$$

E.1.3 Supervised Contrastive Learning

The SupCON framework adopts Supervised Contrastive Learning with Soft Targets (SupCon-ST) as its representation learning objective. For a batch of N samples, given the L_2-normalized projection features $\{z_i\}_{i=1}^{N}$ obtained after RandAugment, the loss is defined as:

$$\mathcal{L}_{\text{SupCon-ST}} := -\frac{1}{N}\sum_{i=1}^{N} \underbrace{\sum_{j \neq i} S_{ij} \log P_{ij}}_{=:\mathcal{L}_{\text{SupCon-ST}}^{(i)}} \tag{39}$$

where the pairwise prediction P_{ij} is the softmax-normalized cosine similarity over all other samples in the batch:

$$P_{ij} = \frac{\exp(z_i^\top z_j/\tau)}{\sum_{k \neq i} \exp(z_i^\top z_k/\tau)} \tag{40}$$

and S_{ij} denotes the soft pairwise target derived from the label vectors:

$$S_{ij} = \frac{s(\mathbf{y}_i, \mathbf{y}_j)}{\sum_{k \neq i} s(\mathbf{y}_i, \mathbf{y}_k)}, \qquad s(\mathbf{y}_i, \mathbf{y}_j) = \frac{\mathbf{y}_i^\top \mathbf{y}_j}{\|\mathbf{y}_i\|\|\mathbf{y}_j\|} \tag{41}$$

$$\mathcal{L}_{SupCON} := -\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\text{SupCon-ST}}^{(i)} \tag{42}$$

E.1.4 ConOSR

ConOSR additionally enforces contrast against Mixup-generated virtual samples $\{\tilde{z}_j\}_{j=1}^{N}$ that simulate the open space, constructed as in (36). The weights are applied to both the augmented batch and the Mixup batch:

$$\mathcal{L}_{ConOSR} := -\frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\text{SupCon-ST}}^{(i)} - \frac{1}{N}\sum_{j=1}^{N} \mathcal{L}_{\text{SupCon-ST}}^{\mathrm{Mixup}(j)} \tag{43}$$

Note that the two terms are calculated independently: clean anchors are contrasted exclusively with clean samples, while Mixup anchors are contrasted only with Mixup samples. Unlike the original ConOSR implementation, which computes a single loss over the combined union of both batches, we separate them. This separation ensures that contrast labels within each term share a uniform norm, 1 for clean samples and $\sqrt{\lambda^2 + (1-\lambda)^2}$ for Mixup samples. Consequently, Lemma 1 holds for both terms, allowing the target to simplify into the inner-product expression in (44). Concatenating the batches jointly would introduce varying contrast norms across groups, breaking this reduction.

E.1.5 Integration with SupCON and ConOSR

The label smoothing is only applied to the anchor labels, and not to the contrast labels. All the contrast labels $\mathbf{y}_j$ remains one-hot for each anchor label. In this section we will derive the relationship between the smoothing coefficient ϵ_i and the soft pairwise target S_{ij}. We show that the relationship is different depending on if the contrast sample i is a positive (same class label) or negative (different class label)

We first show that this normalization has no effect in Equation 41.

Lemma 1 (Reduction to inner products)**.** *Let $\mathbf{y}'_i \in \Delta^{C-1}$, and let all contrast labels have equal norm, $\|\mathbf{y}_k\| = c > 0$ for all $k \neq i$. If $\sum_{k\neq i}(\mathbf{y}'_i)^\top \mathbf{y}_k > 0$, then*

$$S_{ij} = \frac{(\mathbf{y}'_i)^\top \mathbf{y}_j}{\sum_{k\neq i}(\mathbf{y}'_i)^\top \mathbf{y}_k}. \tag{44}$$

Proof. For every $k \neq i$, $s(\mathbf{y}'_i, \mathbf{y}_k) = (\mathbf{y}'_i)^\top \mathbf{y}_k \,/\, (c\,\|\mathbf{y}'_i\|)$. The factor $1/(c\,\|\mathbf{y}'_i\|)$ is common to the numerator and to every term of the denominator of (41), and cancels. □

Proposition 4 (Smoothed SupCON target)**.** *If $\varepsilon_i \in (0, 1]$ or $n_i^+ \geq 1$, then*

$$S_{ij} = \frac{(1-\varepsilon_i)\,1\{y_j = y_i\} + \varepsilon_i/C}{(1-\varepsilon_i)\,n_i^+ + \varepsilon_i K/C}. \tag{45}$$

Proof. Taking the inner product of (30) with a one-hot contrast label $\mathbf{y}_k$, and using $\mathbf{y}_i^\top \mathbf{y}_k = 1\{y_k = y_i\}$ and $\mathbf{1}^\top \mathbf{y}_k = 1$,

$$(\mathbf{y}'_i)^\top \mathbf{y}_k = (1-\varepsilon_i)\,1\{y_k = y_i\} + \frac{\varepsilon_i}{C}. \tag{46}$$

Summing over $k \neq i$ gives $(1-\varepsilon_i)\,n_i^+ + \varepsilon_i K/C$, which is positive under the stated condition. Lemma 1 with $c = 1$ yields (45). □

At $\varepsilon_i = 0$, (45) recovers the published target $S_{ij} = \mathbb{1}\{y_j = y_i\}/n_i^+$; at $\varepsilon_i = 1$ it is uniform, $S_{ij} = 1/K$; and $\sum_{j\neq i} S_{ij} = 1$ for all ε_i.

Proposition 5 (Monotone mass transfer)**.** *Let $1 \leq n_i^+ < K$. Increasing ε_i moves target mass from positives to negatives: every positive's weight decreases, every negative's weight increases, and*

$$\frac{S_{i,\mathrm{pos}}}{S_{i,\mathrm{neg}}} = 1 + \frac{C\,(1-\varepsilon_i)}{\varepsilon_i} \tag{47}$$

decreases strictly in $\varepsilon_i \in (0, 1]$.

Proof. Both terms in (45) share a denominator, so the ratio (47) is immediate and clearly decreasing. Since each row of S sums to 1, a smaller positive-to-negative ratio means positives lose mass and negatives gain it. □

In summation, for each anchor, increasing smoothing therefore steadily shifts target weight S_{ij} from positives to negatives, and decreasing it sharpens the target back toward the positives.

F Alignment and Uniformity: Formal Statements and Proofs

F.1 Decomposition of SpHOR-Lite and SpHOR

The SpHOR-Lite objective (Bahavan et al., 2026, Eqs. 7, 10) is

$$\mathcal{L}_{\mathrm{SpHOR}} = \mathcal{L}_{\mathrm{vMF}} + \mathcal{R}_{\mathrm{Ortho}}, \tag{48}$$

where

$$\mathcal{L}_{\mathrm{vMF}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{C} S_{ik}\log P_{ik}, \qquad P_{ik} = \frac{\exp(\mathbf{z}_i^\top \boldsymbol{\mu}_k/\tau)}{\sum_{j=1}^{C}\exp(\mathbf{z}_i^\top \boldsymbol{\mu}_j/\tau)}, \qquad S_{ik} = \frac{\tilde{y}_{ik}}{\sum_{j=1}^{C}\tilde{y}_{ij}}, \tag{49}$$

$$\mathcal{R}_{\mathrm{Ortho}} = \log \frac{1}{C(C-1)} \sum_{k \neq l} \exp\Big(\frac{(\boldsymbol{\mu}_k^\top \boldsymbol{\mu}_l)^2}{\tau} \Big). \tag{50}$$

$\mathcal{R}_{\mathrm{Ortho}}$ depends only on the prototypes $\{\boldsymbol{\mu}_k\}_{k=1}^C$; it contains neither the targets $\tilde{\mathbf{y}}_i$ (hence no S_{ik}) nor the projections $\mathbf{z}_i$. The targets therefore enter $\mathcal{L}_{\mathrm{SpHOR}}$ only through $\mathcal{L}_{\mathrm{vMF}}$, which is the target-dependent loss decomposed in Table 1. Expanding the log in $\mathcal{L}_{\mathrm{vMF}}$,

$$\begin{aligned}
\mathcal{L}_{\mathrm{vMF}} &= -\frac{1}{N} \sum_{i=1}^{N} \left(\sum_{k=1}^{C} S_{ik} \, \frac{\mathbf{z}_i^\top \boldsymbol{\mu}_k}{\tau} - \Big(\sum_{k=1}^{C} S_{ik} \Big) \log \sum_{j=1}^{C} \exp\Big(\frac{\mathbf{z}_i^\top \boldsymbol{\mu}_j}{\tau} \Big) \right) \\
&= \underbrace{-\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{C} S_{ik} \, \frac{\mathbf{z}_i^\top \boldsymbol{\mu}_k}{\tau}}_{\text{Alignment}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} \log \sum_{j=1}^{C} \exp\Big(\frac{\mathbf{z}_i^\top \boldsymbol{\mu}_j}{\tau} \Big)}_{\text{Uniformity}},
\end{aligned}$$

using $\sum_{k=1}^{C} S_{ik} = 1$, which holds by construction of S_{ik}; this matches Bahavan et al. (2026, Eq. 9). Since $\mathbf{z}_i, \boldsymbol{\mu}_k \in \mathbb{S}^{p-1}$, $\mathbf{z}_i^\top \boldsymbol{\mu}_k = 1 - \frac{1}{2}\|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2^2$. Substituting, the constants $-1/\tau$ and $+1/\tau$ cancel:

$$\mathcal{L}_{\mathrm{vMF}} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2\tau} \sum_{k=1}^{C} S_{ik} \, \|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2^2 + \log \sum_{j=1}^{C} \exp\Big(- \frac{\|\mathbf{z}_i - \boldsymbol{\mu}_j\|_2^2}{2\tau} \Big) \right), \tag{51}$$

which is the prototype row of Table 1. Hence

$$\mathcal{L}_{\mathrm{SpHOR}} = \frac{1}{N} \sum_{i=1}^{N} \Big(\mathcal{L}_{\mathrm{align}}^{(i)} + \mathcal{L}_{\mathrm{unif}}^{(i)} \Big) + \mathcal{R}_{\mathrm{Ortho}}, \tag{52}$$

where only $\mathcal{L}_{\mathrm{align}}^{(i)}$ depends on the targets. For SpHOR, the batch additionally contains the mixup samples (Bahavan et al., 2026, Eq. 3); the derivation is unchanged.

F.2 Decomposition of SupCON and ConOSR

$$\mathcal{L}_{\mathrm{scst}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq i} S_{ij} \log P_{ij}, \qquad P_{ij} = \frac{\exp(\mathbf{z}_i^\top \mathbf{z}_j / \tau)}{\sum_{n \neq i} \exp(\mathbf{z}_i^\top \mathbf{z}_n / \tau)}, \qquad S_{ij} = \frac{s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j)}{\sum_{n \neq i} s(\tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_n)}. \tag{53}$$

Expanding the log,

$$\begin{aligned}
\mathcal{L}_{\mathrm{scst}} &= -\frac{1}{N} \sum_{i=1}^{N} \left(\sum_{j \neq i} S_{ij} \, \frac{\mathbf{z}_i^\top \mathbf{z}_j}{\tau} - \Big(\sum_{j \neq i} S_{ij} \Big) \log \sum_{n \neq i} \exp\Big(\frac{\mathbf{z}_i^\top \mathbf{z}_n}{\tau} \Big) \right) \\
&= \underbrace{-\frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq i} S_{ij} \, \frac{\mathbf{z}_i^\top \mathbf{z}_j}{\tau}}_{\text{Alignment}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} \log \sum_{n \neq i} \exp\Big(\frac{\mathbf{z}_i^\top \mathbf{z}_n}{\tau} \Big)}_{\text{Uniformity}},
\end{aligned}$$

using $\sum_{j \neq i} S_{ij} = 1$, which holds by construction of S_{ij}; this matches Xu et al. (2023, Eq. 17). Since $\mathbf{z}_i \in \mathbb{S}^{p-1}$, $\mathbf{z}_i^\top \mathbf{z}_j = 1 - \frac{1}{2}\|\mathbf{z}_i - \mathbf{z}_j\|_2^2$. Substituting, the constants $-1/\tau$ and $+1/\tau$ cancel:

$$\mathcal{L}_{\mathrm{scst}} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2\tau} \sum_{j \neq i} S_{ij} \, \|\mathbf{z}_i - \mathbf{z}_j\|_2^2 + \log \sum_{n \neq i} \exp\Big(- \frac{\|\mathbf{z}_i - \mathbf{z}_n\|_2^2}{2\tau} \Big) \right), \tag{54}$$

which is the batch row of Table 1. For ConOSR, the batch additionally contains the mixup samples, whose targets are the mixed labels (Xu et al., 2023, Eq. 3); the derivation is unchanged.

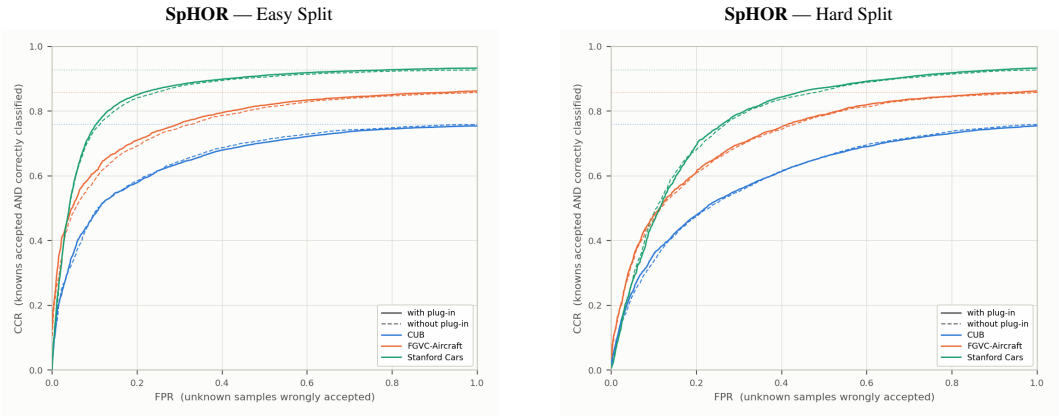


Figure 5: **OSCR operating curves for SpHOR against SSB with the MSP post-processor. Solid** indicates with SmoothOP ; **dashed** is the baseline without the plug-in. The curves are vertically averaged over 3 seeds for a fixed FPR. The results are shown for both the Easy and Hard Splits. SmoothOP pushes out baselines almost consistently across all operating points.

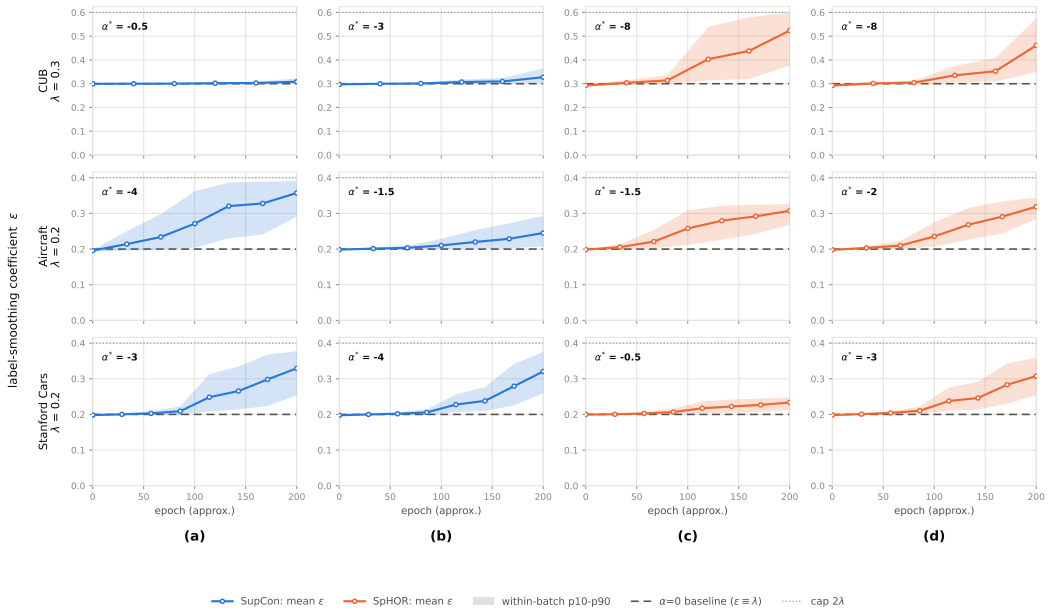


Figure 6: Evolution of the Label Smoothing coefficient ε across the training epochs across all dataset-method combinations, **(a)** SupCON, **(b)** ConOSR, **(c)** SpHOR-Lite, **(d)** SpHOR , and rows are the three datasets, each annotated with its inherited λ. Results averaged over 3 seeds. The line is the batch-mean of ε and the band is **inter-decile** (p_{10}–p_{90}) **range across the batch**. The dashed line is the $\alpha = 0$ baseline. The X-axis is the time mapped to the training epochs

.

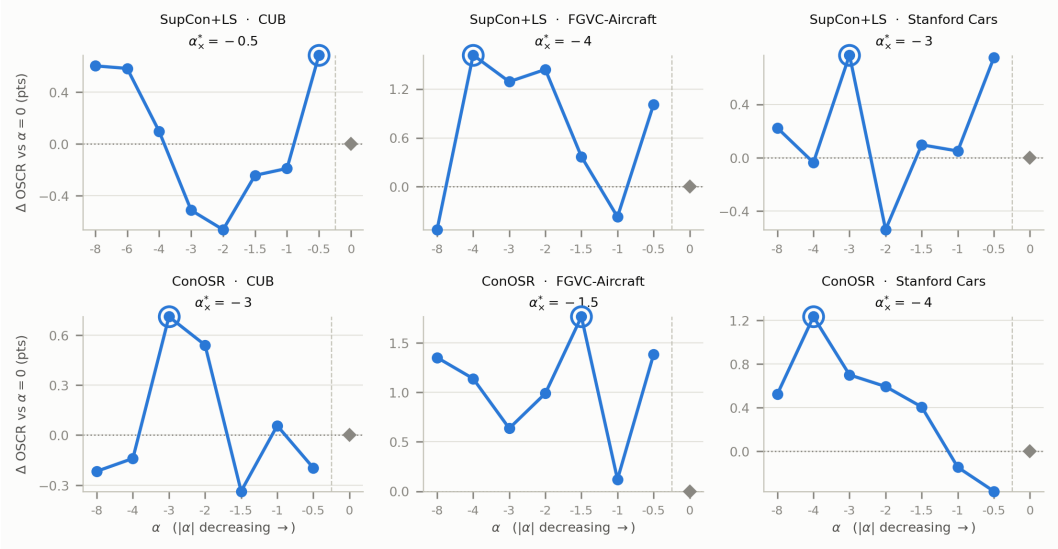


Figure 7: SupCON and ConOSR: Hyperparameter Sweep for α

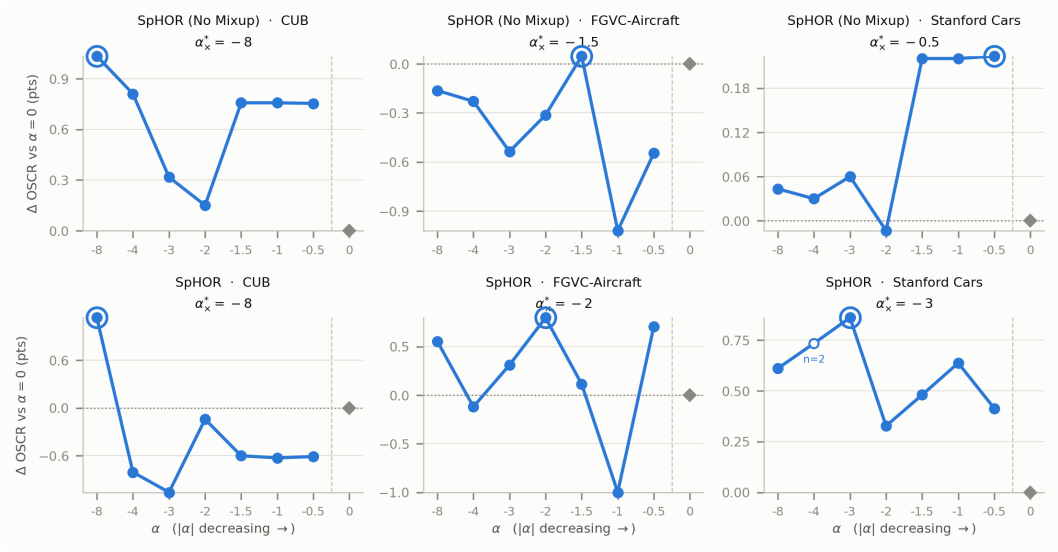


Figure 8: SpHOR-Lite and SpHOR: Hyperparameter Sweep for α