Phantom of Benchmark Dataset: Resolving Label Ambiguity Problem on Image Recognition in the Wild

Hyunhee Chung*
SOCAR AI Research
Seoul, Republic of Korea
esther@socar.kr

Kyung Ho Park*
SOCAR AI Research
Seoul, Republic of Korea
kyungho96@korea.ac.kr

Taewon Seo
SOCAR AI Research
Seoul, Republic of Korea
t1won.seo@gmail.com

Sungwoo Cho
SOCAR AI Research
Seoul, Republic of Korea
peter8526@kaist.ac.kr

Abstract

While deep neural networks achieved supreme accomplishments in image recognition tasks, they conventionally utilize a benchmark dataset that presumes a well-designed label space where each image corresponds to a particular class; we denote these data as obvious samples. However, we claim this assumption is not always justified in the real world as well as widely-utilized ImageNet. We discover that a label ambiguity problem exists, in which several samples are inherently ambiguous and can be annotated as a particular label. In this study, we propose a series of analyses on the label ambiguity and suggest a solution to resolve it along with the following contributions. First, we define label ambiguity types that exist in conventional image recognition and publicize the corresponding datasets retrieved from ImageNet and the real world. We further reveal that this label ambiguity degrades the classification performance; thus, we justify the necessity of careful treatment of the label ambiguous samples. Second, we propose Consistent Sample Selector (CSS), a novel framework that solves this label ambiguity problem. Given obvious and ambiguous samples, the proposed CSS learns representations on each label with obvious samples and selects ambiguous samples that embrace semantics consistent with the obvious ones; thus, it aims to update the training set by concatenating obvious samples and selected ambiguous ones. Lastly, we empirically examine our CSS effectively elevates the classification performance and simultaneously improves the inductive bias, similar to how human vision recognizes.

1. Introduction

Recent advancements in deep neural networks accomplished supreme success in various computer vision applications based on public benchmark sets [10, 36, 9, 6]. One implicit consensus on this public benchmark is that its label space is well-designed; each label's samples share similar characteristics. We denote these samples as *obvious samples*, as one sample clearly belongs to one single label. However, due to the label ambiguity of natural image samples, we urge that this assumption is not always justified, especially in the real world. When the practitioners establish computer vision applications, they start with building a dataset. During this dataset acquisition procedure, several samples frequently exist that are inherently ambiguous to be assigned to a particular class; we denote these samples as *ambiguous samples*. As these ambiguous samples have attributes of multiple classes simultaneously, they can be labeled differently following the annotator's bias. We presume this inconsistent labeling demerits learning representations on each label; thus, we necessitate a solution against it. We define the label ambiguity problem as a deteriorated classification performance due to the careless inclusion of ambiguous samples in the labeled dataset.

While prior works solved this label ambiguity problem by generating multiple weak annotations on these ambiguous samples, they require a particular amount of resource consumption for labeling; the practitioner should employ many labelers. To improve this limit, we tighten this assumption: the practitioners can acquire one label per sample. Following the conventional data annotation procedure, a labeler annotate a given sample as one of the labels if it is an obvious sample. Conversely, a labeler isolates a given sample as an ambiguous one if it cannot be clearly anno-

tated as one label. Suppose the practitioners have a well-labeled dataset with obvious samples and a set of unlabeled ambiguous samples. How can we maximize the classification performance as well as representation quality?

To this end, we shed light on the label ambiguity problem, a cutting-edge problem in image recognition problem by proposing a series of analyses. The key contributions are as follows. First, we propose in-depth elaborations regarding the label ambiguity problem. We firstly define two label ambiguity types and publicize corresponding datasets for the research community. Furthermore, we empirically examine that careless use of these ambiguous samples indeed deteriorates classification performance; thus, there should be a sophisticated treatment towards the label ambiguity. Second, we suggest a Consistent Sample Selector (CSS), a novel solution that solves the label ambiguity problem. We empirically examine that our CSS escalates the classification performance compared to other baselines. Third, we scrutinized the underlying reasons behind our CSS's supremacy in resolving the label ambiguity problem. We discover that a model trained with CSS understands a given image similar to a human's visual understanding. We additionally reveal that a model trained under the CSS has a larger knowledge capacity; thus, it can describe more patterns of a given image.

2. Related Works

Semi-Supervised Learning The semi-supervised learning (SSL) shares a similar objective with our circumstance: both approaches use learned representations from well-labeled samples to understand unlabeled ones [3, 2, 7, 8, 30, 33, 21, 17]. However, the details are clearly different because the unlabeled samples at SSL have a solid ground truth; they can be annotated as one of the given labels. Conversely, the unlabeled samples at label ambiguity problem do not have ground truth, as they can be annotated as various labels. While the SSL differs in detail, we employ the SSL method as a baseline because it shares similar motivation and objective with our label ambiguity problem. Throughout recent works on SSL, we categorize these studies into two streams: consistency regularization and pseudo-labeling. First, consistency regularization approaches aim to establish a function that is invariant to the input perturbations or augmentations [30], and this function would effectively utilize unlabeled samples to understand the given label space [2]. Second, pseudo-labeling approaches provide pseudo labels on unlabeled samples based on the inductive bias built upon the labeled samples [17]. Among these works, we employ the pseudo-labeling paradigm, especially Uncertainty-guided Pseudo-label Selection (UPS) [29] as it is state-of-the-art in public benchmark settings of SSL study.

Uncertainty Estimation Recent studies on deep neural networks started to understand how confident the model predicts a given sample, as well as its accuracy[28, 5, 4, 12]. As reasonably-estimated uncertainty can benefit understanding the model's decisions or make deep neural networks more trustworthy; thus, numerous studies have been proposed [38, 23, 24, 20]. The early approach is a confidence-based method, which extracts the logit vector yielded by the trained model and regards a prediction on a given sample as confident if its logit value goes beyond a particular threshold [14, 15]. As this approach is such a naive one, the bayesian approach has been proposed [16, 35].

3. Preliminaries

3.1. Scenario

This section describes a scenario that illustrates how real world practitioners encounter ambiguous samples. When the practitioners aim to establish a dataset, they firstly accumulate a set of unlabeled samples. Then, they acquire a labeling scheme that defines the characteristics of each label, and the labelers follow this scheme to annotate unlabeled samples. The labeler annotates an unlabeled sample as one label when it has characteristics of a particular label, and we define these annotated ones as obvious samples. Conversely, supposing an unlabeled sample cannot be assigned to one label, labelers isolate these samples. For example, labelers isolate the sample when it has multiple attributes of each label or ambiguous characteristics between multiple labels. We denote these isolated ones as ambiguous samples. Consequentially, when the practitioners establish a dataset, they have two types of sub-datasets: 1) a set of well-labeled obvious samples, 2) a set of unlabeled ambiguous samples. We highlight that our study assumes the existence of the aforementioned two sub-datasets.

3.2. Label Ambiguity and Datasets

We define label ambiguity as a circumstance where an image cannot be annotated as one of the given labels. As a root cause of this phenomenon, we scrutinize that an ambiguous sample has attributes of multiple labels. We presume this characteristic makes the labelers get confused; thus, they shall isolate these samples during the annotation procedure. While previous studies on label ambiguity did not clearly define ambiguous samples and this multidisciplinary, our study firstly breaks down two categories of ambiguity: object ambiguity and texture ambiguity. We elaborate on these ambiguity types in the following sections and publicize the corresponding datasets shown in Figure 1. We also describe the number of samples at the training, validation, and test set in Table 1. We will publicize the whole dataset via an accessible URL shortly. Note that training set have both obvious and ambiguous samples while the vali-

dation and test sets have obvious samples only. We hereby emphasize that our study aims to escalate understanding on given labels under the existence of ambiguous samples.

Texture Ambiguity We define texture ambiguity as when an ambiguous sample's discriminative characteristics are texture-wise attributes, and it is interpreted differently following the labeler's bias. Referring to Figure 1, suppose we solve a binary classification on car images between two labels: *Normal* and *Defect*. Note that *Normal* implies a car without any defects (i.e., scratch, dents) where *Defect* means a car with damaged areas. We denote discriminative characteristics between these labels as a texture-wise attribute, as samples at both labels look similar to each other, but the texture on their surface differs. In this case, the severity of this damaged area matters for labelers. What if ambiguous samples shown in Figure 1 is given to the labeler? It certainly has a damaged area on the surface, but some labelers might annotate it as *Normal* because the damaged area is too small. The others annotate it as *Defect* because there at least bears a damaged area. Unfortunately, we could not retrieve texture-wise ambiguous samples in the public benchmarks. Instead, in collaboration with *Anonymous Institution Name*, which is the largest car-sharing platform in *Anonymous Country Name*, we establish real-world classification tasks regarding car images: car defect classification and car dirt classification. We describe these datasets below.

- **Car-Defect:** This dataset includes two obvious labels of *Normal* and *Defect*. The *Normal* samples have car images without any defects, while *Defect* samples are car images with the damaged surface. We set ambiguous samples as car images with weak damage on their surface.
- **Car-Dirt:** This dataset includes two obvious labels of *Normal* and *Dirt*. The *Normal* samples have car images without any dirty areas, while *Dirt* samples are car images with the dirty surface. We set ambiguous samples as car images with weak dirt on their surface.

Object Ambiguity We define object ambiguity as when an ambiguous sample simultaneously embraces multiple labels' object-wise attribute. Referring to Figure 1, suppose we solve a binary classification between *Frog* and *Tadpole*, and the discriminative characteristics between two labels are the existence of leg object. How would the labelers annotate ambiguous samples shown in Figure 1? When we design a labeling scheme as 'a frog should have legs, and a tadpole does not have legs', these samples would be annotated as 'Tadpole'. But, most conventional *Tadpole* samples do not have any legs; then, we presume the inclusion of these ambiguous samples (tadpoles with a few legs) into the *Tadpole* label would deteriorate the learned representation. Conversely, if we change the labeling scheme to 'a frog should have at least one leg, and tadpole should not have any legs', they would be annotated as *Frog*. To analyze this object ambiguity, we create two classification datasets consisting of two labels with obvious samples and one set of ambiguous samples. We subsample these datasets from the ImageNet [10], and denote them as ImageNet-Frog and ImageNet-Panda. We emphasize that even the ImageNet (a widely-used public benchmark dataset) surprisingly embraces ambiguous samples in the label; we reconfirm the importance of analyzing the label ambiguity problem as it has not been cautiously dealt with in the prior computer vision studies. The detailed descriptions on these datasets are as follows.

- **ImageNet-Frog:** Given samples under the *Frog* and *Tadpole* label in ImageNet, we re-label them. We annotate frog images with four legs as *Frog* label, tadpole images with no legs as *Tadpole*, and let the other samples be ambiguous ones.
- **ImageNet-Panda:** We select *Panda* and *Raccoon* classes as obvious classes. And the *Red Panda* class that has both the shape of a panda and raccoon are assigned to ambiguous samples.

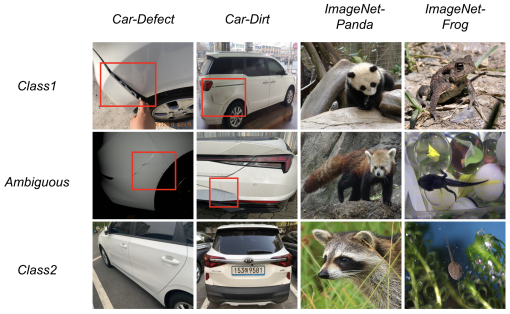


Figure 1. Sample images on the proposed datasets. Note that we draw red bounding box for discriminative cues.

Table 1. Dataset configuration

Dataset	Training Set		Val Set	Test Set
	Obvious	Ambiguous		
Car-Defect	4757	5552	1189	1385
Car-Dirt	1755	491	377	377
ImageNet-Frog	722	102	179	100
ImageNet-Panda	1568	1077	524	522

3.3. Implementation Details

For CNN architectures, we employ widely-utilized ResNet-50 and Progressive Multi-Granularity Networks (PMG). As we presume several classification tasks bear fine-grained discriminative characteristics, we decide to additionally use PMG architecture following its compelling

performances in fine-grained recognition studies [9, 18, 37, 11]. We set the learning objective as minimizing conventional cross-entropy loss, optimize with Adam optimizer, and use dropout under the rate of 0.2. Please refer to the attached code and artifacts for mode detailed implementation details.

4. Do Ambiguous Samples Indeed Degrade Classification Performance?

Objective and Setup As a preliminary analysis, we hereby examine whether the careless use of ambiguous samples indeed degrades the classification performance. Given obvious samples from two classes (class 1 and 2) and ambiguous samples, we postulate four settings of the training set. For the first option, as a baseline, we employ a training set consisting of obvious samples only. As we did not utilize any ambiguous samples in the training stage, we presume this option is the safest but naive one. For the other options, we set three cases of careless use of ambiguous samples: adding ambiguous samples to the obvious samples of class 1 or 2, and randomly splitting ambiguous samples into the half, and add them to each class. We additionally examine how the number of ambiguous samples affects classification performance. We set the ambiguous ratio, which describes the percentage of utilized ambiguous samples. Upon three ambiguous rations of 100%, 50%, and 25%, we measure the accuracy and F1-score on the test set following four classifiers trained upon different settings. The experiment results are shown in Table 2.

Analysis Upon the experiment results, we firstly clarify that the careless use of ambiguous samples indeed degrades the classification performance in every dataset. Compared to the baseline (denoted as *Class 1 v. Class 2*), every careless addition of ambiguous samples yields a decrease in classification performances. Based on these findings, we hereby justify the necessity of a solution against this label ambiguity problem to prevent performance degradation from the careless use of ambiguous samples. To take a further step, we interestingly find that the pattern of the aforementioned negative impact differs following the ambiguity ratio. We once presumed that a model trained under a high ambiguity ratio would experience much severe performance drop. However, we discover that the high ambiguity ratio does not always correlate to the high performance drop. Instead, this pattern varies following the label space (i.e., the dataset consists of obvious samples) and how the practitioners add ambiguous samples. For example, in *Car-Dirt* dataset, *Class 1 + Ambiguous v. Class 2* achieve a particular performance decrease at the ambiguous rate of 100% and 25%, but accomplish a minimal drop at the ambiguous rate of 50%. This pattern becomes different at *Car-Defect* dataset while these two datasets are categorized in the same texture ambiguity. The magnitude of the performance drop is also different. While the careless use of ambiguous samples at *ImageNet-Frog* makes the drop within 10% of accuracy, this magnitude becomes much larger than 30% at *ImageNet-Panda*. Consequentially, as the landscape of performance degradation differs following the data's characteristics, we expect a solution against this label ambiguity problem to be established adaptive to the label space and the obvious sample's characteristics.

5. Our Approach: Consistent Sample Selector

5.1. Motivation

To tackle down the problem caused by careless use of ambiguous samples, we ideate the following question: what if we can select ambiguous samples that are beneficial to recognizing given labels? We hypothesize that an ambiguous sample with consistent cues with the obvious samples will benefit the inductive bias, although they were not annotated as obvious ones. If so, what can be a proxy for filtering this 'beneficial' sample adaptive to the representation of obvious samples? To answer the following questions, we expect the use of uncertainty can be a useful proxy for describing beneficial ambiguous samples. For a reason behind our ideation, previously-proposed studies on semi-supervised learning [29] support that uncertainty yielded by a well-trained classifier is a significant hint of understanding samples that are not seen a priori. We interpret an ambiguous sample with low uncertainty from the initial classifier would embrace consistent cues with the obvious samples; thus, this would contribute to a better understanding of given classes. For an ambiguous sample with low uncertainty, vice versa. Moreover, as we once declare that a proxy for selecting meaningful ambiguous samples should be adaptive to the given label space. We analyze the uncertainty to satisfy the aforementioned requirement as it is yielded by the model trained on a given label space; thus, there's no heuristically-designed human intervention to extract uncertainty on a given sample. To this end, our study employs two representative uncertainty scores: *Confidence score* and *Bayesian score*. The confidence score implies a maximum logit value extracted right after the softmax activation, and the bayesian score is estimated with the promising uncertainty estimation method proposed in [12].

Before we design the solution, we hereby examine how the model trained under obvious samples yields uncertainty scores on unseen obvious and ambiguous samples. We presume the model would provide low Bayesian score and high confidence score on unseen obvious samples as they bear characteristics consistent with the given label space. Conversely, we expect the model would yield comparatively higher Bayesian score and lower confidence score in ambiguous samples. To examine our hypothesis, we trained a classifier based on ImageNet-Frog's obvious training sam-

Table 2. Analysis of the impact of careless use of ambiguous samples toward image recognition.

Ambiguous Ratio	Settings	Texture Ambiguity				Object Ambiguity			
		Car-Defect		Car-Dirt		ImageNet-Frog		ImageNet-Panda	
		Acc	F1-Score	Acc	F1-Score	Acc	F1-Score	Acc	F1-Score
Baseline	Class 1 v. Class 2	0.9096	0.9281	0.9682	0.9682	0.8500	0.8474	0.9483	0.9715
100%	Class 1 + Ambiguous v. Class 2	0.8857	0.7159	0.9337	0.9336	0.8200	0.8125	0.5574	0.6981
	Class 1 v. Class 2 + Ambiguous	0.8279	0.9201	0.9257	0.9255	0.8400	0.8377	0.6513	0.7697
	Class 1 + 50 % Ambiguous v. Class 2 + 50 % Ambiguous	0.8992	0.9081	0.9390	0.9390	0.7600	0.7478	0.6781	0.7968
50%	Class 1 + Ambiguous v. Class 2	0.7386	0.8213	0.9629	0.9629	0.7600	0.7363	0.6284	0.7551
	Class 1 v. Class 2 + Ambiguous	0.8742	0.9288	0.9496	0.9496	0.8400	0.8390	0.6628	0.7687
	Class 1 + 50 % Ambiguous v. Class 2 + 50 % Ambiguous	0.8317	0.8922	0.9469	0.9469	0.8400	0.8284	0.6801	0.7731
25%	Class 1 + Ambiguous v. Class 2	0.7602	0.8393	0.9496	0.9496	0.8300	0.8205	0.6312	0.7598
	Class 1 v. Class 2 + Ambiguous	0.8945	0.9118	0.9496	0.9496	0.8300	0.8205	0.6312	0.7598
	Class 1 + 50 % Ambiguous v. Class 2 + 50 % Ambiguous	0.7444	0.8471	0.9602	0.9602	0.7800	0.7726	0.6130	0.7251

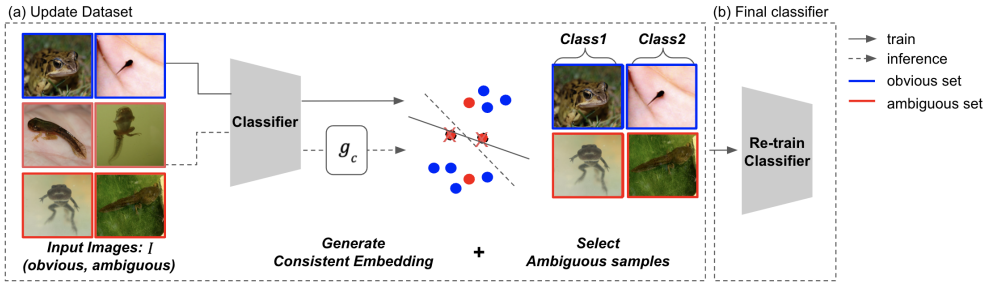


Figure 2. Overall architecture of Consistent Sample Selector

ples and measured both scores at obvious validation samples and training ambiguous samples. We illustrate each option's average scores in Table 3. Referring to this result, we discover that our hypothesis is valid. To take one step further, we conclude that a sample with effective discriminative cues (which belongs in obvious samples) can be represented with low Bayesian score and high confidence score. For the sample without much relevant discriminative cues, vice versa.

To this end, we propose Consistent Sample Selector (CSS), a simple but effective framework that utilizes particular ambiguous samples to elevate the classification performance as well as representation quality. We denote our framework as CSS as it aims to utilize ambiguous samples that include knowledge or cues consistent with the training set with obvious samples. Our CSS filters out ambiguous samples with sufficient discriminative cues for the target task and re-trains the model with these selected samples. We illustrate an overall architecture of CSS in Figure 2, and the detailed procedures are provided in the following sections.

Table 3. Average uncertainty scores yielded by a model trained under obvious samples

Score	Dataset	
	Obvious	Ambiguous
Confidence Score	0.9836	0.9015
Bayesian Score	0.0559	0.1893

5.2. Methodology

Generating an Initial Classifier First and foremost, CSS trains an initial classifier with obvious samples in the training set. This step aims to convey a base inductive bias to the model that illustrates given classes' discriminative cues. We presume a inductive bias based on obvious samples embraces a particular amount of discriminative characteristics on given classes.

Selecting Ambiguous Samples with Consistency Given the trained initial classifier, as a core procedure of CSS, it select ambiguous samples with consistent knowledge of the given classes. With the initial classifier trained with only an obvious set, we aimed to find the ambiguous samples that have consistent knowledge that strengthens the representation generated only with obvious samples. Given the trained initial classifier and ambiguous sample $x^{(i)}$, we can get the predicted logit vector denoted as p_x. With this $p_x^{(i)}$, we can get the Bayesian score of the sample denoted as $B(p_x^{(i)})$ and the confidence score with $C(p_x^{(i)})$ itself. Then, we decide whether to select this ambiguous sample or not by applying thresholding. We set two thresholds for Bayesian and confidence scores, respectively: κ and τ. We select a given ambiguous sample if its Bayesian and confidence score simultaneously goes larger than the aforementioned thresholds, and we formally describe this procedure in Equation 1. Note that we let $g_c^{(i)}$ as a binary indicator illustrating whether we use the given ambiguous sample or not. Consequentially, we acquire a pair of ambiguous sample $x^{(i)}$ and its corresponding indicator $g_c^{(i)}$. For ambiguous

samples with an indicator of 1, we updated them to the obvious samples following the label with maximum confidence ($C(p_x^{(i)})$).

$$g_c^{(i)} = \mathbb{1}[B(p_x^{(i)}) \leq \kappa]\mathbb{1}[C(p_x^{(i)}) \geq \tau]. \quad (1)$$

Re-training the classifier Lastly, CSS establishes an updated training set by concatenating an initially-given training set with the selected ambiguous samples. Then, we re-train the classifier with this updated training set. We highlight that this re-training procedure first adds consistent samples to the initial training set and trains the model, not simply fine-tunes the initially-trained model on consistent samples. We empirically check this fine-tuning excessively drops the classification performance as these small (compared to the training set) ambiguous samples confuse the model to acquire correct representations of the given classes. To this end, the CSS results in a re-trained classifier that better understands given classes' characteristics compared to the one trained only with obvious samples. Throughout the following analyses, we examine the effectiveness of CSS in image recognition tasks and whether it indeed improves the quality of inductive bias.

6. Effectiveness of CSS

Objective and Setup

First and foremost, we aim to validate whether the proposed CSS improves the classification performances in image recognition tasks. For comparative studies, we employ several baseline approaches. We firstly consider a supervised model training only with obvious training samples, denoted as **OO**, and we presume the practitioners can naively use them. Moreover, we additionally apply the model calibration method to the **OO** as calibrated classifiers are known to be robust in various challenging environments [25, 1]. We hypothesize that a robust classifier would acquire a more contextual understanding of given images; thus, we expect it can be a solid baseline. We use a supervised classifier with label smoothing [26, 22] and denote it as **OO-LS**.

Moreover, we employ Uncertainty-guided Pseudo-label Selection (**UPS**) as a baseline, the state-of-the-art method in pseudo-labeling approaches of semi-supervised learning [40, 34]. We highlight that our CSS has several novelties compared to the UPS. While UPS simultaneously utilizes positive and negative learning, our CSS only employs positive learning. Positive learning lets the model predict the correct label while negative learning forces the model to choose the incorrect label. As unlabeled samples in the SSL have a solid ground-truth label, we presume using both positive and negative learning can benefit representation learning. However, unlabeled (ambiguous) samples do not have a single ground-truth label in the label ambiguity problem setting. Instead, these unlabeled samples tend to simultaneously include discriminative cues of multiple labels. We expect the model can sufficiently predict the correct label of the given ambiguous sample while it cannot choose the incorrect label (as both labels can be correct labels).

Lastly, we employ Uncertainty-Guided Pseudo-Labeling (**UGPL**) which deals with the label ambiguity problem [27]. We highlight our work's novelties compared to the UGPL as follows. First, the proposed CSS utilizes both confidence and Bayesian score as a proxy for selecting beneficial samples, while the UGPL only employs a confidence score only. Second, the UGPL is only examined under the synthetically-created samples, but our study validates its effectiveness in both public benchmark and real-world datasets. Moreover, as we expect label smoothing to enhance the inductive bias's quality, we also implement UGPL along with label smoothing (denoted as **UGPL-LS**). We acknowledge that the proposed CSS and baselines commonly require empirically-optimized thresholds; thus, we measure the classification performances at various threshold levels and report the best ones. The comparative experiment results are described in Table 4.

Analysis Following the results shown in Table 4, we propose the following takeaways. First, we discover that the proposed CSS accomplishes the best classification performance in every label ambiguity type. While the careless use of ambiguous samples degraded the classification performance (shown in the earlier section), we prove that the appropriate use of ambiguous samples can sufficiently escalate the classification performance. Second, compared to UPS, eliminating negative learning at CSS was beneficial in understanding ambiguous samples. We confirm our hypothesis on the inferior impact of negative learning is correct; ambiguous samples can be interpreted as the ones with discriminative cues of multiple labels. Therefore, the model should be trained in the manner of predicting correct labels, not predicting incorrect labels. Lastly, compared to UGPL, the use of the Bayesian score contributes to better performance escalation. We interpret that the simultaneous use of confidence and Bayesian score improves the quality of selected ambiguous samples. Still, we acknowledge there exists an improvement avenue that the machine learning practitioners should empirically optimize the best thresholds of confidence and bayesian score. We note that further studies can deal with automated or parameterized ways of selecting these thresholds in future works.

7. Does CSS improve Representation Quality?

Objectives and Setup We then further scrutinize whether the CSS provides qualified inductive bias, not just only achieving good performance. To evaluate this representation quality, we regard a model that recognizes a given image similar to the human vision as a qualified one. As

Table 4. Comparative experiment results. The bold number implies the best performance in a given dataset.

Methods	Object Ambiguity								Texture Ambiguity							
	Car-Defect				Car-Dirt				ImageNet-Frog				ImageNet-Tadpole			
	Acc	Prec	Recall	F1-Score	Acc	Prec	Recall	F1-Score	Acc	Prec	Recall	F1-Score	Acc	Prec	Recall	F1-Score
OO	0.9067	0.9992	0.8404	0.9281	0.9629	0.9628	0.9630	0.9629	0.8500	0.8455	0.8510	0.8474	0.9483	0.9973	0.9483	0.9715
OO-LS	0.9314	0.9319	0.9316	0.9314	0.9655	0.9659	0.9659	0.9655	0.8700	0.8657	0.8715	0.8678	0.9559	0.9984	0.9559	0.9761
UPS	0.8823	0.8892	0.8872	0.8823	0.9348	0.9381	0.9342	0.9348	0.8763	0.8729	0.8728	0.8763	0.9519	0.9943	0.9512	0.9629
UGPL	0.9350	0.9355	0.9352	0.9350	0.9735	0.9735	0.9737	0.9735	0.8700	0.8657	0.8715	0.8678	0.9579	0.9984	0.9579	0.9774
UGPL-LS	0.9401	0.9401	0.9402	0.9401	0.9788	0.9789	0.9787	0.9788	0.8900	0.8882	0.8855	0.8867	0.9655	0.9984	0.9655	0.9813
CSS (OURS)	**0.9437**	**0.9448**	**0.9440**	**0.9437**	**0.9788**	**0.9787**	**0.9789**	**0.9788**	**0.9899**	**0.9999**	**0.9937**	**0.9448**	**0.9693**	**0.9703**	**0.9693**	**0.9693**

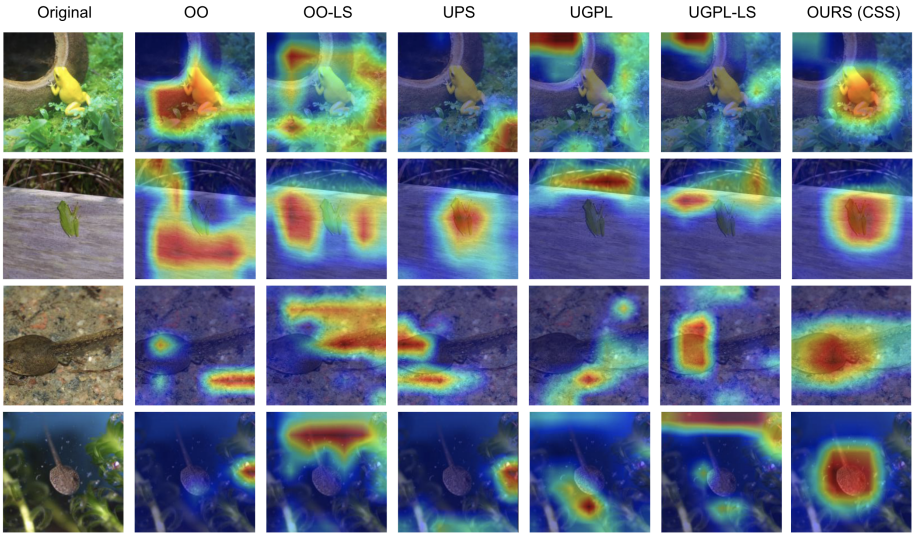


Figure 3. Grad-CAM results yielded by the CSS and baselines. We conclude that the proposed CSS empowers the model to recognize discriminative cues of given image most similar to the human vision.

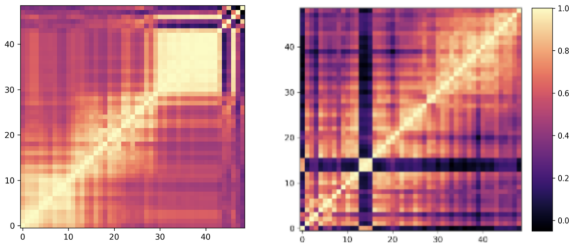
a wide range of prior studies pursues a model that understands similar to humans, we evaluate our definition of this qualified representation is sufficiently justified. Our study utilizes Grad-CAM [32, 31], which is a widely-utilized method to visually interpret where the model primarily focuses on for recognizing a given image. We prepare two classifiers trained only with obvious samples and CSS, and examine whether the CSS yields a inductive bias that better understands the image compared to the other one. The results are shown in Figure 3. Note that we utilize the trained model under CSS as the one with the best classification performance in each dataset.

Analysis Upon the results shown in Figure 3, we analyze that CSS surprisingly conveys improved inductive bias compared to the baseline. We observe that CSS lets the model correctly recognize discriminative cues while the baseline mistakenly focuses on the other area. Throughout the analyses, we discover several takeaways. First, while the trained model only with obvious samples achieves a promising performance, it does not always guarantee that model correctly captures the discriminative cues. Second, we examine that CSS conveys better inductive bias as well as enhanced classification performance. We analyze that adding selected ambiguous samples benefit the model to scrutinize discriminative cues in given classes deeply; thus, it reduces the discrepancy between the trained model and human visual recognition.

8. What drives an Improved Inductive Bias?

Objective and Setup We further scrutinize what drives the aforementioned improved understanding. We presume an answer exists in knowledge capacity. We hypothesize a model trained under CSS would illustrate wider knowledge than the others; thus, it enables the model to acquire more contextual understanding. To understand the knowledge capacity of the trained model, we utilize Centered Kernel Alignment (CKA). Note that we use CKA in the analysis as it is state-of-the-art in its research area. The CKA measures representation similarity between two models and quantifies it in the range of 0 (not similar) to 1 (very similar). Given a trained model, we acquire a layer-wise similarity that measures representation similarity scores among the layers within the same model. Suppose representation similarity scores are high among the layers. It implies that representations that exist in various layers are similar to each other; thus, the model cannot implicit fruitful knowledge unless it has many layers. For the low layer-wise similarity, vice versa. In a nutshell, we analyze whether the proposed

CSS makes the model embrace more knowledge compared to the others. The results are shown in Figure 4.



(a) Obvious Only (b) CSS (OURS)

Figure 4. Visualized representation similarity of CNNs trained under the different settings. Note that both x and y axis implies blocks at ResNet-50.

Analysis Referring to Figure 4, we discover that CSS acquires a less-similar representation landscape compared to the others; thus, it enables the model to acquire more knowledge. The representation landscape trained only with obvious samples shows a clear block structure [19], which implies layers in a block share such similar knowledge. We interpret that training a model only with obvious samples cannot effectively leverage every layer within a model. Conversely, a model trained under the CSS embraces less severe block structure, as well as lower layer-wise similarity scores. We regard this representation landscape with fewer similarities to prove that CSS allows the model to illustrate wider knowledge; thus, it enables the model to capture discriminative cues on given classes correctly.

9. Do iterating CSS Maximize Performance?

Objective and Setup Lastly, supposing a re-trained model acquires superior classification performance and inductive bias, what if we re-train the model multiple times? We question whether iterating the proposed CSS procedures for N times can further improve the classification performance. Given obvious training samples and ambiguous samples, we proceed with a single iteration of CSS. Then, it yields the following artifacts: an updated training set, a re-trained model, and not-selected ambiguous samples (denoted as residuals). For the $N = 2$ iteration, we proceed with the CSS's second and third steps on residuals with a re-trained classifier. At each iteration, we measure the classification performance and the number of selected ambiguous samples at the CSS's second step. Upon these setups, the experiment results are described in Table 5.

Analysis Referring to Table 5, we observe that iterated CSS does not accomplish enhanced performance rather than a single iteration. We hypothesize these results stem from the fixed threshold on uncertainty scores at CSS. When we process the first iteration of CSS, the classifier's representation changes compared to the initial model (trained with obvious training samples only). Given a sample, the initial and re-trained models would interpret it differently; then, the confidence and uncertainty levels would also become different. We hereby expect the second iteration of CSS should take different threshold levels (as a model yielded after the first iteration has different inductive bias), but the proposed CSS utilizes thresholds optimized to the first iteration. Accordingly, these not-optimized thresholds create an inferior selection of ambiguous samples at $N > 2$ iterations; thus, it creates performance degradation. We presume these thresholds shall be adapted following updated inductive bias, but we leave this point as an improvement avenue. Consequentially, we recommend that practitioners use our CSS for a single iteration, and we presume it can be further improved with different thresholds at each iteration.

Table 5. Classification performance and the number of selected ambiguous samples (#) at each CSS iteration

Dataset	Texture Ambiguity				Object Ambiguity			
	Car-Defect		Car-Dirt		ImageNet-Frog		ImageNet-Panda	
Loops	Acc	#	Acc	#	Acc	#	Acc	#
1(Baseline)	**0.9168**	361	**0.9788**	87	**0.9899**	71	**0.9483**	9
2	0.9096	2655	0.9309	45	0.9499	17	0.9039	4
3	0.9038	2792	0.9267	45	0.9699	11	0.8659	3
4	0.9024	3208	0.8514	23	0.9599	10	0.8505	2
5	0.8944	3358	0.8859	20	0.9599	9	0.7662	2

10. Discussions and Conclusion

In this study, we firstly categorize label ambiguity into two types (texture and object ambiguity) and examine the careless use of these ambiguous samples degrade the classification performance. We then propose CSS as a solution to this problem. Throughout experiments, we prove our CSS's effectiveness in classification performance as well as its enhanced representation quality. We additionally analyze that CSS empowers the model to recognize given image similar to humans as well as embraces wider knowledge rather than other settings. Lastly, we validate that iterating the proposed CSS does not contribute to much performance escalation; thus, we recommend the practitioners use the CSS with a single iteration. Still, we acknowledge that our CSS has several improvement avenues. As the CSS requires preset thresholds on Bayesian and confidence scores, future works would look for a parameterized, or automated procedure for selecting them. Furthermore, the CSS shall be validated in various tasks such as multi-class classifications, object detection [39] or semantic segmentation [13].

11. Acknowledgement

This work is supported by the Korea Agency for Infrastructure Technology Advancement grant funded by the Ministry of Land, Infrastructure, and Transport (Grant RS-2022-0014579).

References

[1] Daniel Alvarez-Coello, Daniel Wilms, Adnan Bekan, and Jorge Marx Gómez. Towards a data-centric architecture in the automotive industry. *Procedia Computer Science*, 181:658–663, 2021.

[2] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019.

[3] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

[4] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[6] Fabio Maria Carlucci, Paolo Russo, Tatiana Tommasi, and Barbara Caputo. Hallucinating agnostic images to generalize across domains. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3227–3234. IEEE, 2019.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[8] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *International workshop on artificial intelligence and statistics*, pages 57–64. PMLR, 2005.

[9] Afshin Dehghan, Syed Zain Masood, Guang Shu, Enrique Ortiz, et al. View independent vehicle make, model and color recognition using convolutional neural network. *arXiv preprint arXiv:1702.01721*, 2017.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[11] Ruoyi Du, Dongliang Chang, Ayan Kumar Bhunia, Jiyang Xie, Zhanyu Ma, Yi-Zhe Song, and Jun Guo. Fine-grained visual classification via progressive multi-granularity training of jigsaw patches. In *European Conference on Computer Vision*, pages 153–168. Springer, 2020.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[13] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, 2017.

[14] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[15] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[16] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR, 2015.

[17] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5070–5079, 2019.

[18] Parneet Kaur, Karan Sikka, and Ajay Divakaran. Combining weakly and webly supervised learning for classifying food images. *arXiv preprint arXiv:1712.08730*, 2017.

[19] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019.

[20] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[21] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

[22] Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pages 6448–6458. PMLR, 2020.

[23] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[24] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

[25] Mohammad Motamedi, Nikolay Sakharnykh, and Tim Kaldewey. A data-centric approach for training deep neural networks with less data. *arXiv preprint arXiv:2110.03613*, 2021.

[26] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

[27] Kyung Ho Park and HyunHee Chung. Uncertainty guided pseudo-labeling: Estimating uncertainty on ambiguous data for escalating image recognition performance. In *ICAART (2)*, pages 541–551, 2022.

[28] Mark N Read, Kieran Alden, Jon Timmis, and Paul S Andrews. Strategies for calibrating models of biology. *Briefings in Bioinformatics*, 21(1):24–35, 2020.

[29] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. *arXiv preprint arXiv:2101.06329*, 2021.

[30] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29, 2016.

[31] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[32] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? *arXiv preprint arXiv:1611.07450*, 2016.

[33] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng MaXiaoyu Tao, and Nanning Zheng. Transductive semi-supervised deep learning using min-max features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 299–315, 2018.

[34] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

[35] Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust bayesian neural networks. *Advances in neural information processing systems*, 29, 2016.

[36] Dong Wang and Xiaoyang Tan. Unsupervised feature learning with c-svddnet. *Pattern Recognition*, 60:473–485, 2016.

[37] Yaming Wang, Vlad I Morariu, and Larry S Davis. Learning a discriminative filter bank within a cnn for fine-grained recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4148–4157, 2018.

[38] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[39] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[40] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.