CRAP: SEMI-SUPERVISED LEARNING VIA CONDITIONAL ROTATION ANGLE PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-supervised learning (SlfSL), aiming at learning feature representations through ingeniously designed pretext tasks without human annotation, has achieved compelling progress in the past few years. Very recently, SlfSL has also been identified as a promising solution for semi-supervised learning (SemSL) since it offers a new paradigm to utilize unlabeled data. This work further explores this direction by proposing a new framework to seamlessly couple SlfSL with SemSL. Our insight is that the prediction target in SemSL can be modeled as the latent factor in the predictor for the SlfSL target. Marginalizing over the latent factor naturally derives a new formulation which marries the prediction targets of these two learning processes. By implementing this framework through a simple-but-effective SlfSL approach – rotation angle prediction, we create a new SemSL approach called Conditional Rotation Angle Prediction (CRAP). Specifically, CRAP is featured by adopting a module which predicts the image rotation angle **conditioned on the candidate image class**. Through experimental evaluation, we show that CRAP achieves superior performance over the other existing ways of combining SlfSL and SemSL. Moreover, the proposed SemSL framework is highly extendable. By augmenting CRAP with a simple SemSL technique and a modification of the rotation angle prediction task, our method has already achieved the state-of-the-art SemSL performance.

1 INTRODUCTION

The recent success of deep learning is largely attributed to the availability of a large amount of labeled data. However, acquiring high-quality labels can be very expensive and time-consuming. Thus methods that can leverage easily accessible unlabeled data become extremely attractive. Semi-supervised learning (SemSL) and self-supervised learning (SlfSL) are two learning paradigms that can effectively utilize massive unlabeled data to bring improvement to predictive models.

SemSL assumes that a small portion of training data is provided with annotations and the research question is how to use the unlabeled training data to generate additional supervision signals for building a better predictive model. In the past few years, various SemSL approaches have been developed in the context of deep learning. The current state-of-the-art methods, e.g. MixMatch (Berthelot et al., 2019), unsupervised data augmentation (Li et al., 2018), converge to the strategy of combining multiple SemSL techniques, e.g. Π-Model (Laine & Aila, 2017), Mean Teacher (Tarvainen & Valpola, 2017), mixup (Zhang et al., 2018), which have been proved successful in the past literature.

SlfSL aims for a more ambitious goal of learning representation without any human annotation. The key assumption in SlfSL is that a properly designed pretext predictive task which can be effortlessly derived from data itself can provide sufficient supervision to train a good feature representation. In the standard setting, the feature learning process is unaware of the downstream tasks, and it is expected that the learned feature can benefit various recognition tasks. SlfSL also offers a new possibility for SemSL since it suggests a new paradigm of using unlabeled data, i.e., use them for feature training. Recent work (Zhai et al., 2019) has shown great potential in this direction.

This work further advances this direction by proposing a new framework to seamlessly couple SlfSL with SemSL. The key idea is that the prediction target in SemSL can serve as a latent factor in the course of predicting the pretext target in a SlfSL approach. The connection between the predictive targets of those two learning processes can be established through marginalization over the latent

factor, which also implies a new framework of SemSL. The key component in this framework is a module that predicts the pretext target conditioned on the target of SemSL. In this preliminary work, we implement this module by extending the rotation angle prediction method, a recently proposed SlfSL approach for image recognition. Specifically, we make its prediction conditioned on each candidate image class, and we call our method Conditional Rotation Angle Prediction (CRAP). The proposed framework is also highly extendable. It is compatible with many SemSL and SlfSL approaches. To demonstrate this, we further extend CRAP by using a simple SemSL technique and a modification to the rotation prediction task. Through experimental evaluation, we show that the proposed CRAP achieves significantly better performance than the other SlfSL-based SemSL approaches, and the extended CRAP is on par with the state-of-the-art SemSL methods. In summary, the main contributions of this paper are as follows:

- We propose a new SemSL framework which seamlessly couples SlfSL and SemSL. It points out a principal way of upgrading a SlfSL method to a SemSL approach.
- Implementing this idea with a SlfSL approach, we create a new SemSL approach (CRAP) that can achieve superior performance than other SlfSL-based SemSL methods.
- We further extend CRAP with a SemSL technique and an improvement over the SlfSL task. The resulted new method achieves the state-of-the-art performance of SemSL.

2 Related Work

Our work CRAP is closely related to both SemSL and SlfSL.

SemSL is a long-standing research topic which aims to learn a predictor from a few labeled examples along with abundant of unlabeled ones. SemSL based on different principals are developed in the past decades, e.g., "transductive" models (Gammerman et al., 1998; Joachims, 2003), multi-view style approaches (Blum & Mitchell, 1998; Zhou & Li, 2005) and generative model-based methods (Kingma et al., 2014; Springenberg, 2016), etc. Recently, the consistency regularization based methods have become quite influential due to their promising performance in the context of deep learning. Specifically, Π-Model (Laine & Aila, 2017) requires model's predictions to be invariant when various perturbations are added to the input data. Mean Teacher (Tarvainen & Valpola, 2017) enforces a student model producing similar output as a teacher model whose weights are calculated through the moving average over the weight of student model. Virtual Adversarial Training (Miyato et al., 2018) encourages the predictions for input data and its adversarially perturbed version to be consistent. More recently, mixup (Zhang et al., 2018; Verma et al., 2019) has emerged as a powerful SemSL regularization method which requires the output of mixed data to be close to the output mixing of original images. In order to achieve good performance, most state-of-the-art approaches adopt the strategy of combining several existing techniques together. For example, Interpolation Consistency Training (Verma et al., 2019) incorporates Mean Teacher into the mixup regularization, MixMatch (Berthelot et al., 2019) adopts a technique that uses fused predictions as pseudo prediction target as well as the mixup regularization. Unsupervised data augmentation (Li et al., 2018) upgrades Π-Model with advanced data augmentation methods.

SlfSL is another powerful paradigm which learns feature representations through training on pretext tasks whose labels are not human annotated (Kolesnikov et al., 2019). Various pretext tasks are designed in different approaches. For example, image inpainting (Pathak et al., 2016) trains model to reproduce an arbitrary masked region of the input image. Image colorization (Zhang et al., 2016) encourages model to perform colorization of an input grayscale image. Rotation angle prediction (Gidaris et al., 2018) forces model to recognize the angle of a rotated input image. After training with the pretext task defined in a SlfSL method, the network is used as a pretrained model and can be fine-tuned for a downstream task on task-specific data. Generally speaking, it is still challenging for SlfSL method to achieve competitive performance to fully-supervised approaches. However, SlfSL provides many new insights into the use of unlabeled data and may have a profound impact to other learning paradigms, such as semi-supervised learning.

SlfSL based SemSL is an emerging approach which incorporates SlfSL into SemSL. The most straightforward approach is to first perform SlfSL on all available data and then fine-tune the learned model on labeled samples. S^4L (Zhai et al., 2019) is a newly proposed method which jointly train the downstream task and pretext task in a multi-task fashion without breaking them into stages. In

this paper, we further advance this direction through proposing a novel architecture which explicitly links these two tasks together and ensure that solving one task is beneficial to the other.

3 COUPLING SEMSL WITH SLFSL

In SemSL, we are given a set of training samples $\{x_1, x_2, \cdots, x_n\} \in X$ with only a few of them $X_l = \{x_1, x_2, \cdots, x_l\} \in X$ annotated with labels $\{y_1, y_2, \cdots, y_l\} \in Y_l$ (usually $l << n$ and y is considered as discrete class label here). The goal of a SemSL algorithm is to learn a better posterior probability estimator over y, i.e., $p(y|x, \theta)$ with θ denoting model parameters, from both labeled and unlabeled training samples. SlfSL aims to learn feature representations via a pretext task. The task usually defines a target z, which can be derived from the training data itself, e.g., rotation angle of the input image. Once z is defined, SlfSL is equivalent to training a predictor to model $p(z|x; \theta)$. There are two existing schemes to leverage SlfSL for SemSL. The first is to use SlfSL to learn the feature from the whole training set and then fine-tuning the network on the labeled part. The other is jointly optimizing the tasks of predicting y and z, as in the recently proposed S^4L method. As shown in Figure 1 (a), S^4L constructs a network with two branches and a shared feature extractor. One branch for modeling $p(y|x; \theta)$ and another branch for modeling $p(z|x; \theta)$. However, in both methods the pretext target z predictor $p(z|x; \theta)$ is implicitly related to the task of predicting y.

Our framework is different in that we explicitly incorporate y into the predictor for z. Specifically, we treat y as the latent factor in $p(z|x; \theta)$ and factorize $p(z|x; \theta)$ through marginalization:

$$p(z|x; \theta) = \sum_y p(z, y|x; \theta) = \sum_y p(z|x, y; \theta)p(y|x; \theta). \tag{1}$$

Eq. 1 suggests that the pretext target predictor $p(z|x; \theta)$ can be implemented as two parts: a model to estimate $p(y|x; \theta)$ and a model to estimate z conditioned on both x and y, i.e., $p(z|x, y; \theta)$. For the labeled samples, the ground-truth y is observed and can be used for training $p(y|x; \theta)$. For unlabeled samples, the estimation from $p(y|x; \theta)$ and $p(z|x, y; \theta)$ will be combined together to make the final prediction about z. Consequently, optimizing the loss for $p(z|x; \theta)$ will also provide gradient to back-propagate through $p(y|x; \theta)$. This is in contrast to the case of S^4L, where the gradient generated from the unlabeled data will not flow through $p(y|x; \theta)$. Theoretically, $p(z|x; \theta)$ and $p(y|x; \theta)$ can be two networks, but in practise we model them as two branches connecting to a shared feature extractor.

$p(z|x; \theta)$ suggested by Eq. 1 is essentially a pretext target predictor with a special structure and partial observations on its latent variable, i.e. y. The benefits of using such a predictor can be understood from three perspectives: (1) $p(y|x; \theta)$ in Eq. 1 acts as a soft selector to select $p(z|x, y; \theta)$ for predicting z. If the estimation of $p(y|x; \theta)$ is accurate, it will select $p(z|x, y = \hat{y}(x); \theta)$ for prediction and update, where $\hat{y}(x)$ is the true class of x. This selective updating will make $p(z|x, y; \theta)$ give more accurate prediction over z if y matches $\hat{y}(x)$. After such an update, $p(z|x, y; \theta)$ will in turn encourage $p(y|x; \theta)$ to attain higher value for $y = \hat{y}(x)$ since the prediction from $p(z|x, y = \hat{y}(x); \theta)$ is more likely to be accurate. Thus, the terms $p(y|x; \theta)$ and $p(z|x, y; \theta)$ will reinforce each other during training. (2) even if $p(y|x; \theta)$ is not accurate (this may happen at the beginning of the training process), $p(z|x, y; \theta)$ can still perform the pretext target prediction and act as an unsupervised feature learner. Thus, the features will be gradually improved in the course of training. With a better feature representation, the estimation of $p(y|x; \theta)$ will also be improved. (3) Finally, to predict z in Eq. 1, $p(z|x, y; \theta)$ needs to be evaluated for each candidate y. This in effect is similar to creating an ensemble of diversified pretext target predictors and with the combination weight given by $p(y|x; \theta)$ according to the marginalization rule. Thus, training features with Eq. 1 may enjoy the benefit from ensemble learning. Again, this will lead to better features and thus benefit the modelling of $p(y|x; \theta)$ and $p(z|x, y; \theta)$.

The above framework provides a guideline for turning a SlfSL method into a SemSL algorithm: (1) modifying a SlfSL predictor $p(z|x; \theta)$ by $p(z|x, y; \theta)$ and introducing a branch for $p(y|x; \theta)$ (2) optimizing the prediction of z on the SemSL dataset and update the branches $p(z|x, y; \theta)$, $p(y|x; \theta)$ and their shared feature extractor. (3) using $p(y|x; \theta)$ as downstream task predictor or adding an additional branch for training $p(y|x; \theta)$ only with the labeled data as in S^4L. More details about the additional branch will be explained in Section 4.

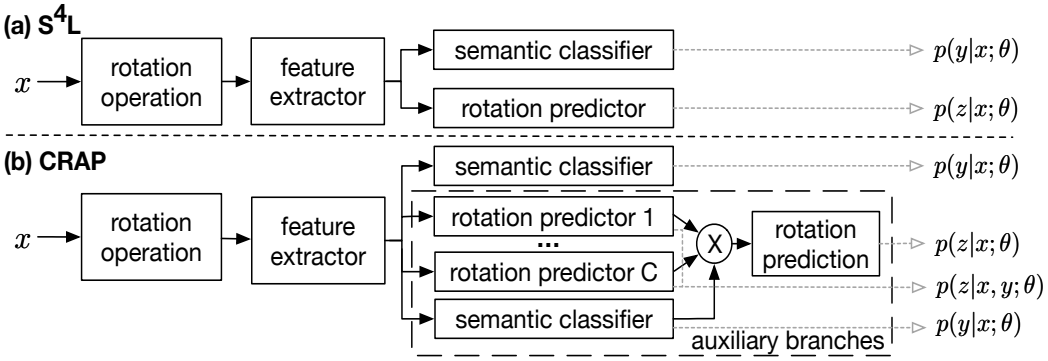

Figure 1: The structure comparison between S^4L and our proposed CRAP method. The symbol C in our model denotes the number of semantic classes.

4 CRAP: SemSL via conditional rotation angle prediction

In the following part, we will describe an implementation of this framework, which is realized by upgrading the rotation-angle prediction-based SlfSL to its conditional version.

Rotation angle prediction is a recently proposed SlfSL approach for image recognition. It randomly rotates the input image by one of the four possible rotation angles ($\{0°, 90°, 180°, 270°\}$) and requires the network to give a correct prediction of the rotation angle. Despite being extremely simple, this method works surprisingly well in practice. The underlying logic is that to correctly predict the rotation angle, the network needs to recognize the canonical view of objects from each class and thus enforces the network to learn informative patterns of each image category.

Following the proposed framework, we upgrade rotation angle prediction to conditional rotation angle prediction (CRAP) for semi-supervised learning. In this case, z in Eq. 1 is the rotation angle and y is the class label of input image x. We realize $p(z|x, y; \theta)$ by allocating a rotation angle prediction branch for each class. The prediction from each branch is then aggregated with the aid of $p(y|x; \theta)$ for the final prediction of z as shown in Eq. 1. A more detailed schematic illustration of the CRAP method is shown in Figure 1 (b). As seen, our method adopts a network with multiple branches and a shared feature extractor. Specifically, branches within the dashed box are called auxiliary branches since they are only used for training and will be discarded at the test stage. It contains C rotation predictors which corresponds to $p(z|x, y; \theta)$ and a semantic classifier which generates $p(y|x; \theta)$. The auxiliary branches and feature extractor are trained by using the procedure described in Section 3. Note that in CRAP, we do not directly use the semantic classifier from the auxiliary branches as the final classifier. Instead, we introduce an additional semantic classifier and learn it only via the loss incurred from the labeled data. This treatment is similar to S^4L and we find this strategy work slightly better in practice. We postulate the reason is that the $p(y|x; \theta)$ branch in auxiliary branches is mainly trained by the supervision generated from the optimization of $p(z|x; \theta)$. Such supervision is noisy comparing with the loss generated from the ground-truth y. It is better to use such a branch just for feature training since the latter is more tolerant to noisy supervision.

Remark: (1) One potential obstacle of our model is that the quantity of parameters in the auxiliary branches would increase significantly with a large C. To tackle this, we propose to perform dimension reduction for the features feeding into the rotation predictor. Results in Section 5.3 show that this scheme is effective as our performance will not drop even when the dimension is reduced from 2048 to 16. (2) The CRAP method is also highly expendable. In the following, we will extend CRAP from two perspectives: improving $p(y|x; \theta)$ and improving $p(z|x, y; \theta)$.

4.1 Extension 1: Incorporating an Additional SemSL loss

As discussed in Section 3, our method essentially introduces a network module with a special structure and partial observations on the latent variable y. Besides using labeled data to provide supervision for y, we can also use existing SemSL techniques to provide extra loss for modeling $p(y|x; \theta)$.

To implement such an extension, we employ a simple SemSL loss as follows: we rotate each image in four angles within one batch (the prediction of the rotated image can be obtained as the byproduct of CRAP) and obtain the arithmetic average $\bar{p}$ of the predicted distributions across these four rotated samples. Then we perform a sharpening operation over $\bar{p}$ as in MixMatch :

$$\hat{p_i} = \frac{(\bar{p}_i)^{\frac{1}{T}}}{\sum_{j=1}^{C}(\bar{p}_j)^{\frac{1}{T}}}, \tag{2}$$

where C is the number of classes and $T \in (0, 1]$ is a temperature hyper-parameter. Then we use the cross entropy between $\hat{p_i}$ and $p(y|x;\theta)$ (in auxiliary branches) as an additional loss.

Note that other (more powerful) SemSL can also apply here. We choose the above SemSL technique is simply because its operation, i.e. image rotation, has already been employed in the CRAP algorithm and thus could be reused to generate the additional SemSL loss without increasing the complexity of the algorithm.

4.2 Extension 2: using conditional denoising rotation prediction

We also make another extension over CRAP by introducing an improved version of the conditional rotation prediction task. Specifically, we require the rotation prediction branch to predict rotation angle for a mixed version of the rotated image, that is, we randomly mix the input image x_i with another randomly sampled rotated image x_j via $x_{\text{mix}} = \alpha x_i + (1-\alpha)x_j$, with α sampled from $[0.5, 1]$. Meanwhile, the class prediction $p(y|x_i;\theta)$ is calculated from the unmixed version of the input x_i. In such a design, the network needs to recognize the rotation angle of the target object with the noisy distraction from another image, and we call this scheme denoising rotation prediction. The purpose of introducing this modified task is to make the SlfSL task more challenging and more dependent on the correct prediction from $p(y|x;\theta)$. To see this point, let's consider the following example. Letter 'A' is rotated with 270° and is mixed with letter 'B' with rotation 90°. Directly predicting the rotation angle for this mixed image encounters an ambiguity: whose rotation angle, A's or B's, is the right answer? In other words, the network cannot know which image class is the class-of-interest. This ambiguity can only be resolved from the output of $p(y|x;\theta)$ since its input is unmixed target image. Therefore, this improved rotation prediction task relies more on the correct prediction from the semantic classifier and training through CRAP is expected to give stronger supervision signal to $p(y|x;\theta)$. Note that although the denoising rotation prediction also uses mix operation, **it is completely different from mixup.** The latter constructs a loss to require the output of the mixed image to be mixed version of the outputs of original images. This loss is not applied in our method. For more algorithm details about CRAP and the extended CRAP, please refer to the Appendix A.1.

5 Experiments

In this section, we conduct experiments to evaluate the proposed CRAP method[1]. The purpose of our experiments is threefolds: (1) to validate if CRAP is better than other SlfSL-based SemSL algorithms. (2) to compare **CRAP** and extended CRAP (denoted as **CRAP+** hereafter) against the state-of-the-art SemSL methods. (3) to understand the contribution of various components in CRAP.

5.1 Experimental details

To make a fair comparison to recent works, different experimental protocols are adopted for different datasets. Specifically, for CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009) and SVHN (Netzer et al., 2011), we directly follow the settings in (Berthelot et al., 2019). For ILSVRC-2012 (Russakovsky et al., 2015), our settings are identical to (Zhai et al., 2019) except for data pre-processing operations for which we only use the inception crop augmentation and horizontal mirroring. We ensure that all the baselines are compared under the same setting. Followed the standard settings of SemSL, the performance with different amount of labeled samples are tested. For CIFAR-10 and SVHN, sample size of labeled images is ranged in five levels: $\{250, 500, 1000, 2000, 4000\}$. For CIFAR-100, 10000 labeled data is used for training. For ILSVRC-2012, 10% and 1% of images are

[1]Source code: https://www.dropbox.com/s/ciummonqkd5u3as/CRAP.zip?dl=0

Table 1: Comparison of error rates (%) for SlfSL based SemSL on CIFAR10.

# Labels	250	500	1000	2000	4000
Labeled-only	55.69±2.07	45.11±1.85	37.69±1.67	29.14±2.95	20.83±1.09
Fine-tune	53.30±4.05	36.23±0.96	28.31±0.67	21.94±0.20	17.77±0.35
S^4L	37.26±1.23	31.35±1.20	29.03±0.67	24.71±0.88	20.27±1.60
CRAP	**17.26±1.22**	**15.13±0.42**	**12.79±0.10**	**10.73±0.05**	**9.26±0.17**

Table 2: Comparison of error rates (%) for SlfSL based SemSL on SVHN and SVHN+Extra.

	# Labels	250	500	1000	2000	4000
SVHN	Labeled-only	25.20±2.69	16.66±1.48	13.05±0.64	9.64±0.14	7.66±0.40
	Fine-tune	47.17±1.78	32.64±3.33	23.63±0.82	17.78±0.08	13.83±0.45
	S^4L	28.97±0.98	25.09±4.50	18.74±0.94	15.21±0.48	12.70±0.28
	CRAP	**10.21±1.45**	**7.99±0.34**	**5.81±0.11**	**5.07±0.37**	**4.61±0.27**
+Extra	Fine-tune	39.23±2.23	24.89±1.31	17.67±0.63	12.16±0.53	9.38±0.39
	S^4L	19.19±1.26	14.80±1.23	12.39±0.63	11.00±0.39	9.16±0.29
	CRAP	**7.21±0.32**	**5.49±0.20**	**4.79±0.17**	**4.31±0.22**	**3.77±0.08**

labeled among the whole dataset. In each experiment, three independent trials are conducted for all datasets except for ILSVRC-2012. See more details in Table 8 in Appendix.

5.2 Compare with SlfSL-Based SemSL methods

Firstly, we compare CRAP to other SlfSL-based SemSL algorithms on five datasets: CIFAR-10, CIFAR-100, SVHN, SVHN+Extra and ILSVRC-2012.

Two SlfSL-based SemSL baseline approaches are considered: 1) **Fine-tune**: taking the model pre-trained on the pretext task as an initialization and fine-tuning with a set of labeled data. We term this method Fine-tune in the following sections. 2) $\boldsymbol{S^4L}$: S^4L method proposed in (Zhai et al., 2019). Note that we do not include any methods which combine other SemSL techniques. For this reason, we only use our basic CRAP algorithm in the comparison in this subsection. As a reference, we also report the performance obtained by only using the labeled part of the dataset for training, denoting as **Labeled-only**. The experimental results are as follows:

CIFAR-10 The results are presented in Table 1. We find that the "Fine-tune" strategy leads to a mixed amount of improvement over the "Labeled-only" case. It is observed that a large improvement can be obtained when the amount of labeled samples is ranged from 500 to 2000 but not on 250 and 4000's settings. It might be because on one hand too few labeled samples are not sufficient to perform an effective fine-tuning while on the other hand the significant improvement diminishes after the sample size increase. In comparison, S^4L achieves much better accuracy for the case of using few samples. This is largely benefited from its down-stream-task awareness design — the labeled training samples exerts impact at the feature learning stage. Our CRAP method achieves significantly better performance than those two ways of incorporating SlfSL for SemSL and always halves the test error of S^4L in most cases.

SVHN and SVHN+Extra Table 2 shows the results of each method. Somehow surprisingly, we find that the Fine-tune and S^4L do not necessarily outperform the Labeled-only baseline. They actually performs worse than Labeled-only on SVHN. With more training data in SVHN + Extra, S^4L tends to bring benefits for enhancing performance when the size of labeled samples are small e.g., with 250 samples. In comparison, the proposed CRAP still manages to produce significant improvement over Labeled-only in all those settings. This result clearly demonstrates that the simple combination of SlfSL and SemSL may not necessarily bring improvement and a properly-designed strategy of incorporating SlfSL with SemSL is crucial.

CIFAR-100 As shown in Table 3, it is obvious that all SlfSL-based SemSL methods can have better accuracy than that of Labeled-only. S^4L leads to a marginal improvement over Fine-tune although its performance is a little bit unstable on different partitions as shown by its higher variance. Again, the proposed CRAP achieves significant improvement over those baselines.

Table 3: Comparison of error rates (%) for SlfSL based SemSL on CIFAR-100.

Methods	Labeled-only	Fine-tune	S^4L	CRAP
CIFAR-100	46.83±0.42	45.40±0.60	42.98±3.91	**30.32±0.83**

Table 4: ILSVRC-2012 accuracies (%) of SlfSL based SemSL methods.

# Labels	10%		1%	
	Top1	Top5	Top1	Top5
Labeled-only	-	80.43	-	48.43
Fine-tune	-	78.53	-	45.11
S^4L	-	83.82	-	53.37
Labeled-only	59.16	83.07	41.26	69.07
S^4L	63.84	86.28	46.90	74.16
CRAP	**65.34**	**87.07**	**49.26**	**75.57**

Table 5: ILSVRC-2012 accuracies (%) for reducing the dimension of input feature channels.

# Labels	10%		1%	
	Top1	Top5	Top1	Top5
Dim-2048	65.34	87.07	49.26	75.57
Dim-512	65.46	87.04	49.25	75.79
Dim-256	65.45	86.95	49.08	75.62
Dim-128	65.53	86.93	48.94	75.53
Dim-64	65.45	87.04	48.98	75.60
Dim-32	65.33	87.08	48.96	75.41
Dim-16	65.39	86.85	48.78	75.36

ILSVRC-2012 Table 4 presents the results of each method. The top block of Table 4 shows the reported results in the original S^4L paper and we also re-implement S^4L based on the code of (Kolesnikov et al., 2019). Due to the difference of data pre-processing, results in the upper block cannot be directly compared to those below. Again, we have observed that CRAP is consistently superior to S^4L in all settings. As mentioned in Section 4, for saving the computational cost, we propose to reduce the dimensionality of features fed into the rotation angle predictor when there is a large number of classes. In Table 5, we demonstrates the effect of this scheme. As seen, the test performance stays the same when the feature dimensions is gradually reduced from 2048 to only 16 dimensions. This clearly validates the effectiveness of the proposed scheme.

5.3 Compare with the state-of-the-art SemSL

In the following section, we proceed to demonstrate the performance of CRAP+, that is, the extended CRAP method by incorporating the two extensions discussed in Section 4.1 and 4.2. We compare its performance against the current state-of-the-art methods in SemSL. Similar to (Berthelot et al., 2019), several SemSL baselines are considered: Pseudo-Label, Π-Model, Mean Teacher, Virtual Adversarial Training (VAT), MixUp and MixMatch [2]. Since a fair and comprehensive comparison has been done in (Berthelot et al., 2019) and we strictly follow the same experimental setting, we directly compare CRAP+ to the numbers reported in (Berthelot et al., 2019).

The experimental results are shown in Figure 2, Figure 3 and Table 6. As seen from those Figures and Table, the proposed CRAP+ is on-par with the best performed approaches, e.g., Mixmatch, in those datasets. This clearly demonstrates the power of the proposed method. Note that the current state-of-the-art in SemSL is achieved by carefully combining multiple existing successful ideas in SemSL. In contrast, our CRAP+ achieves excellent performance via an innovative framework of marrying SlfSL with SemSL. Conceptually, the latter enjoys greater potential. In fact, CRAP might be further extended by using more successful techniques in SemSL, such as MixUp. Since the focus of this paper is to study how SlfSL can benefit SemSL, we do not pursue this direction here.

Table 6: Error rate (%) comparison of CRAP+ to SOTA SemSL methods on CIFAR-100.

Methods	SWA	MixMatch	CRAP+
CIFAR-100	28.8	**25.88±0.30**	**25.97±0.40**

5.4 Ablation Study

Since there are several components in CRAP and CRAP+, we study the effect of adding or removing some components in order to provide additional insight into the role of each part. Specifically, we

[2]For CIFAR-100, we only compare CRAP+ against SWA (Tarvainen & Valpola, 2017) and MixMatch, since those methods achieve the best reported performance in literature.

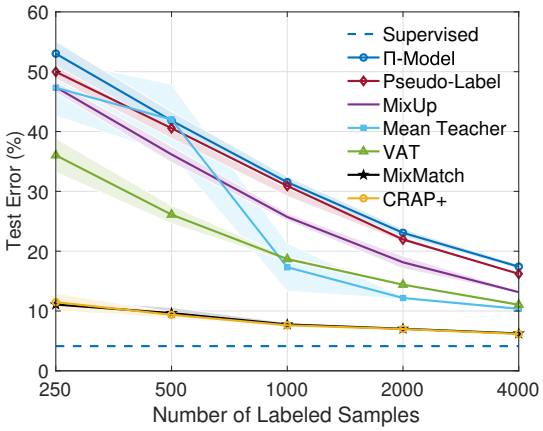

Figure 2: CIFAR-10 error rates of CRAP+ and SOTA SemSL methods.

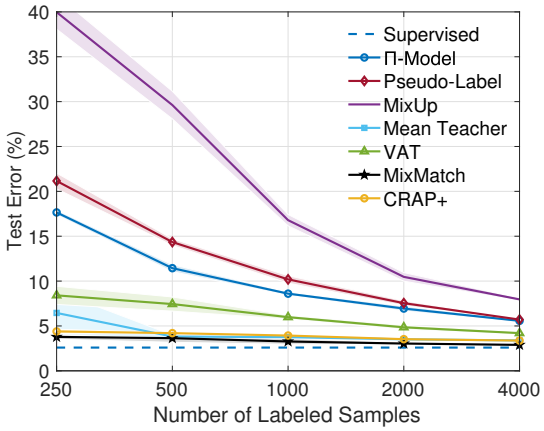

Figure 3: SVHN error rates of CRAP+ and SOTA SemSL methods.

Table 7: Ablation study for CRAP on CIFAR-10 with 250 and 4000 labels.

# Labels	250	4000
CRAP	17.07	9.07
CRAP + extension 1	12.68	7.05
CRAP + extension 1 + extension 2 (CRAP+)	10.76	5.98
CRAP w/o semantic classifier in the main branch	20.10	10.25
CRAP w/o rotation prediction branches + extension 1	54.09	14.38
CRAP w/o whole auxiliary branches	62.73	27.31

measure the effect of (1) only adding extension 1 to CRAP, i.e., incorporating an additional SemSL loss through sharpening operations on the semantic classifier in auxiliary branches (2) further adding extension 2 to CRAP. The resulted model is identical to CRAP+ (3) removing semantic classifier of main branch from CRAP. This is equivalent to using the semantic classifier in auxiliary branches for testing (4) removing rotation angle prediction branch from auxiliary branches and adding extension 1 to CRAP. The resulted structure can be seen as a variant of only using the SemSL technique in Extension 1 (but also with the classifier in main branch) (5) removing whole auxiliary branches from CRAP, i.e., pure supervised method with data rotated.

We conduct ablation studies on CIFAR-10 with 250 and 4000 labels with results presented in Table 7. The main observations are: (1) The two extensions in CRAP+ will bring varying degrees of improvement. Extension 1 in Section 4.1, i.e., a stronger $p(y|x;\theta)$ modeling, perhaps leads to greater improvement. (2) Using an additional semantic classifier leads to a slight performance improvement over the strategy of directly utilizing $p(y|x;\theta)$ in the auxiliary branches for testing (method in third line from the bottom). (3) Using the sharpening strategy as in our extension 1 and training a SemSL method alone does not produce good performance. This indicates the superior performance of CRAP+ is not simply coming from a strong SemSL method but its incorporation with the CRAP framework. (4) Applying rotation as a data augmentation for labeled data (the last method in Table 7) will not lead to improved performance over the labeled-only baseline as by cross referring the results in Table 9. This shows that the advantage of CRAP is not coming from the rotation data augmentation.

6 Conclusion

In this work, we introduce a framework for effectively coupling SemSL with SlfSL. The proposed CRAP method is an implementation of this framework and it shows compelling performance on several benchmark datasets compared to other SlfSL-based SemSL methods. Furthermore, two extensions are incorporated into CRAP to create an improved method which achieves comparable performance to the state-of-the-art SemSL methods.

REFERENCES

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, 2019.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100. Citeseer, 1998.

Alexander Gammerman, Volodya Vovk, and Vladimir Vapnik. Learning by transduction. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pp. 148–155. Morgan Kaufmann Publishers Inc., 1998.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

Thorsten Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 290–297, 2003.

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *The IEEE International Conference on Robotics and Automation*, pp. 7286–7291, 2018.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *The IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems*, 2011.

Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems*, pp. 3235–3246, 2018.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2536–2544, 2016.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *International Conference on Learning Representations*, 2016.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pp. 1195–1204, 2017.

Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 3635–3641, 7 2019.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S^4l: Self-supervised semi-supervised learning. In *The IEEE International Conference on Computer Vision*, October 2019.

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

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pp. 649–666, 2016.

Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge & Data Engineering*, (11):1529–1541, 2005.

A APPENDIX

A.1 ALGORITHM DETAILS OF CRAP AND CRAP+

Algorithm 1: CRAP Pseudocode

Inputs:

$\{X_l, Y_l\}$: collection of labeled samples
X_u: collection of unlabeled samples
$f_{\text{main\_cls}}$: semantic classifier of main branch
$f_{\text{aux\_cls}}$: semantic classifier of auxiliary branches
$f^i_{\text{aux\_rot}}$: rotation angle classifier of auxiliary branches, where $i \in [1, 2, \cdots, C]$
T: total number of iterations
B: minibatch size

Outputs:

$f_{\text{main\_cls}}$: semantic predictor for test set

Process:

1: **for** $t \leftarrow 1$ to T **do**
2: Sample B examples from $\{X_l, Y_l\}$ and X_u respectively
3: Obtain **semantic class prediction** on both main and auxiliary branches:
$p(y_{\text{main}}|x_l) = f_{\text{main\_cls}}(x_l), \quad p(y_{\text{aux}}|x_l) = f_{\text{aux\_cls}}(x_l)$ and train with
CrossEntropy$(p(y_{\text{main}}|x_l), y_l)$ and CrossEntropy$(p(y_{\text{aux}}|x_l), y_l)$ losses
4: Obtain **rotation angle prediction**
$p(z|x) = \sum_y p(z|y, x) \cdot p(y|x) = \sum_{i=1}^{C} f^i_{\text{aux\_rot}}(x) \cdot f_{\text{aux\_cls}}(x)$ and train with
CrossEntropy$(p(z|x), \hat{z})$ loss `// ẑ is rotation angle ground truth`
5: **end for**

Algorithm 2: CRAP+ Pseudocode

Inputs:

$\{X_l, Y_l\}$: collection of labeled samples
X_u: collection of unlabeled samples
$f_{\text{main\_cls}}$: semantic classifier of main branch
$f_{\text{aux\_cls}}$: semantic classifier of auxiliary branches
$f^i_{\text{aux\_rot}}$: rotation angle classifier of auxiliary branches, where $i \in [1, 2, \cdots, C]$
T: total number of iterations
B: minibatch size

Outputs:

$f_{\text{main\_cls}}$: semantic predictor for test set

Process:

1: **for** $t \leftarrow 1$ to T **do**
2: Sample B examples from $\{X_l, Y_l\}$ and X_u respectively
3: Obtain **semantic class prediction** on both main and auxiliary branches:
$p(y_{\text{main}}|x_l) = f_{\text{main\_cls}}(x_l), \quad p(y_{\text{aux}}|x_l) = f_{\text{aux\_cls}}(x_l)$ and train with
CrossEntropy$(p(y_{\text{main}}|x_l), y_l)$ and CrossEntropy$(p(y_{\text{aux}}|x_l), y_l)$ losses
4: Construct sharpened target $\hat{p}$ for unlabeled data `// see extension 1 in Section 4.1`
5: Obtain **semantic class prediction for unlabeled data** on auxiliary branch
$p(y_u|x_u) = f_{\text{aux\_cls}}(x_u)$ and train with CrossEntropy$(p(y_u|x_u), \hat{p})$ loss
6: Construct mixed images $\{x_m\}$ from $\{X_l, X_u\}$ `// see extension 2 in Section 4.2`
7: Obtain **rotation angle prediction**
$p(z_m|x_m) = \sum_y p(z_m|y, x_m) \cdot p(y|x) = \sum_{i=1}^{C} f^i_{\text{aux\_rot}}(x_m) \cdot f_{\text{aux\_cls}}(x)$ and train with
CrossEntropy$(p(z_m|x_m), \hat{z})$ loss `// ẑ is rotation angle ground truth`
8: **end for**

A.2 EXPERIMENTAL DETAILS

The experimental details are presented in Table 8.

Table 8: Experimental details.

datasets	CIFAR-10	SVHN/+Extra	CIFAR-100	ILSVRC2012-1%	ILSVRC2012-10%
architecture	WRN-28-2	WRN-28-2	WRN-28-8.4375	ResNet50v2	ResNet50v2
# training set	50000	73257/+531131	50000	1281167	1281167
# labeled set	{250, 500, 1000, 2000, 4000}		10000	13762	128866
# validation set	5000	7325	5000	5005	50046
minibatch size	64	64	64	256	256
optimizer	Adam	Adam	Adam	SGD	SGD
LR	0.002	0.002	0.002	0.01	0.1
weight decay	0.02	0.02/0.0001	0.04	0.01	0.001
# epoch	1024	500	300	1000	200
# iteration/epoch	1024	1024	1024	53	503
LR rampup	✗	✗	✗	10 epoch	5 epoch
LR decay	✗	✗	✗	10	10
LR decay at	✗	✗	✗	{700,800,900}	{140,160,180}
EMA model	✓	✓	✓	✗	✗

A.3 TABULAR RESULTS

Table 9 and 10 presents a summary for error rate comparison of CRAP and CRAP+ to existing SemSL methods on CIFAR-10 and SVHN respectively. Results in top block are reported in literature where mark † means that the results come from (Oliver et al., 2018), mark ‡ means that the results come from (Verma et al., 2019) and others are from (Berthelot et al., 2019). Results locating in the bottom block are achieved by our implementation.

Table 9: Error rate (%) comparison of CRAP to existing SemSL methods on CIFAR-10.

# Labels	250	500	1000	2000	4000
Labeled-only†	-	-	-	-	20.26±0.38
Π-Model	53.02±2.05	41.82±1.52	31.53±0.98	23.07±0.66	17.41±0.37
Pseudo-Label	49.98±1.17	40.55±1.70	30.91±1.73	21.96±0.42	16.21±0.11
Mixup	47.43±0.92	36.17±1.36	25.72±0.66	18.14±1.06	13.15±0.20
VAT	36.03±2.82	26.11±1.52	18.68±0.40	14.40±0.15	11.05±0.31
MeanTeacher	47.32±4.71	42.01±5.86	17.32±4.00	12.17±0.22	10.36±0.25
MixMatch	11.08±0.87	9.65±0.94	7.75±0.32	7.03±0.15	6.24±0.06
ICT‡	-	-	15.48±0.78	9.26±0.09	7.29±0.02
Labeled-only	55.69±2.07	45.11±1.85	37.69±1.67	29.14±2.95	20.83±1.09
Fine-tune	53.30±4.05	36.23±0.96	28.31±0.67	21.94±0.20	17.77±0.35
S^4L	37.26±1.23	31.35±1.20	29.03±0.67	24.71±0.88	20.27±1.60
CRAP	17.26±1.22	15.13±0.42	12.79±0.10	10.73±0.05	9.26±0.17
CRAP+	11.48±1.46	9.37±0.46	7.61±0.16	6.98±0.19	6.16±0.17

Table 10: Error rate (%) comparison of CRAP to existing SemSL methods on SVHN.

# Labels	250	500	1000	2000	4000
Labeled-only†	-	-	12.83±0.47	-	-
Π-Model	17.65±0.27	11.44±0.39	8.60±0.18	6.94±0.27	5.57±0.14
Pseudo-Label	21.16±0.88	14.35±0.37	10.19±0.41	7.54±0.27	5.71±0.07
Mixup	39.97±1.89	29.62±1.54	16.79±0.63	10.47±0.48	7.96±0.14
VAT	8.41±1.01	7.44±0.79	5.98±0.21	4.85±0.23	4.20±0.15
MeanTeacher	6.45±2.43	3.82±0.17	3.75±0.10	3.51±0.09	3.39±0.11
MixMatch	3.78±0.26	3.64±0.46	3.27±0.31	3.04±0.13	2.89±0.06
ICT‡	4.78±0.68	4.23±0.15	3.89±0.04	-	-
Labeled-only	25.20±2.69	16.66±1.48	13.05±0.64	9.64±0.14	7.66±0.40
Fine-tune	47.17±1.78	32.64±3.33	23.63±0.82	17.78±0.08	13.83±0.45
S^4L	28.97±0.98	25.09±4.50	18.74±0.94	15.21±0.48	12.70±0.28
CRAP	10.21±1.45	7.99±0.34	5.81±0.11	5.07±0.37	4.61±0.27
CRAP+	4.39±0.25	4.20±0.04	3.93±0.06	3.52±0.07	3.37±0.01