Semi-Supervised Learning with Scarce Annotations

Sylvestre-Alvise Rebuffi⋆   Sebastien Ehrhardt⋆   Kai Han∗
Andrea Vedaldi   Andrew Zisserman
Visual Geometry Group, University of Oxford

{srebuffi,hyenal,khan,vedaldi,az}@robots.ox.ac.uk

Abstract

While semi-supervised learning (SSL) algorithms provide an efficient way to make use of both labelled and unlabelled data, they generally struggle when the number of annotated samples is very small. In this work, we consider the problem of SSL multi-class classification with very few labelled instances. We introduce two key ideas. The first is a simple but effective one: we leverage the power of transfer learning among different tasks and self-supervision to initialize a good representation of the data without making use of any label. The second idea is a new algorithm for SSL that can exploit well such a pre-trained representation. The algorithm works by alternating two phases, one fitting the labelled points and one fitting the unlabelled ones, with carefully-controlled information flow between them. The benefits are greatly reducing overfitting of the labelled data and avoiding issue with balancing labelled and unlabelled losses during training. We show empirically that this method can successfully train competitive models with as few as 10 labelled data points per class. More in general, we show that the idea of bootstrapping features using self-supervised learning always improves SSL on standard benchmarks. We show that our algorithm works increasingly well compared to other methods when refining from other tasks or datasets.

1. Introduction

The success of Deep Learning (DL) in computer vision comes at the cost of collecting large quantities of labelled data. In many applications, data collection is increasingly inexpensive, but data annotation still involves manual and thus expensive labor. Semi-supervised learning (SSL) can significantly reduce the cost of learning new models by using large datasets of which only a small proportion comes with manual labels.

When only some data samples are annotated, one can

∗Authors contributed equally

exploit the structure in the data to infer, implicitly or explicitly, the missing annotations. For example, the class of an image generally does not change if the camera is slightly shifted while maintaining the focus on the same point. The resulting viewpoint change can be approximated by a deformation or warp of the image. A change in illumination, which also generally leaves the image class unchanged, can instead be approximated by a linear transformation of the image range. Hence, an effective way of propagating labels is to enforce consistency in prediction: if one of these perturbations is applied to the image, then the prediction should remain the same.

A simple way to incorporate such invariance while training a deep neural network is to consider image augmentations such as rotations and flips and other perturbations such as Dropout [32], which is injected directly at the level of the features. Such perturbations are used as a form of regularization for supervised learning to avoid overfitting. The idea is that a perturbed image should maintain the same ground-truth label as the original image. In the SSL setting, where most labels are unknown, augmentation can be used to encourage the predictions of unlabelled data instances to be 'consistent', i.e. to remain stable across small variations of the input image, regardless of which specific label the data point takes. If data samples are dense enough, stability to local perturbations may be sufficient to propagate the known labels to the rest of the dataset. Similar ideas are used in many approaches to semi-supervised classification, including the recent works of [15, 20, 33]. Still, performance degrades quickly with a diminishing number of labelled instances per class [23].

We can explain this performance drop as follows. Perturbations can bridge some of the gaps between labelled data points, but the effect is ultimately only local. The ability to transport information across points that are farther apart (since there are fewer of them) depends mainly on the smoothness of the data representation. For example, the Euclidean distance is essentially meaningless in image space, so that transferring labels between images that are close as vectors of pixels is very ineffective. On the other hand, if
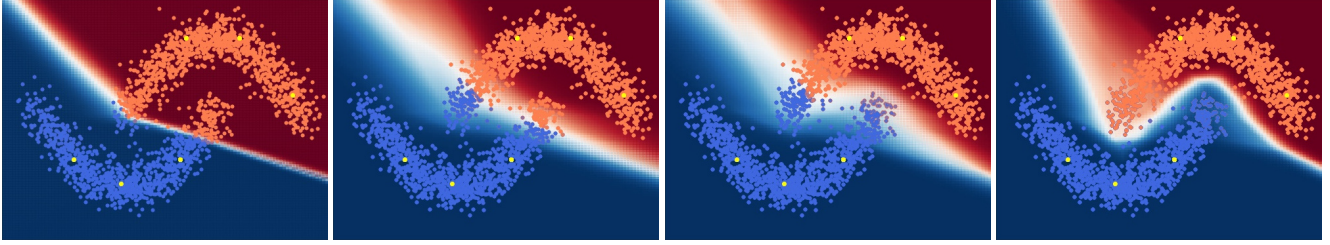
Figure 1. **Consistency loss with the ramp-up strategy.** The Π-model[15] is applied to the 'two moons' problem of [20] and various steps of the optimization are shown. At first, the learned network creates a hard border (first panel) based on the labelled anchor points (in yellow). Then the border is progressively stretched as the consistency loss is enforced more strongly (second and third panels). Overall, the Π-model works well to label connected components of the data space. (Darker regions are more confident; best viewed in color.)

images are first encoded via a representation function in a low dimensional space whose smoothness intrinsically captures some of the relevant invariances, then augmentation can be much more effective.

Unfortunately, simply training a representation such as a deep neural network on a semi-supervised dataset is unlikely to solve the problem. In fact, it is difficult to learn effectively from a label-deficient dataset with a small number of data points with known labels and a very large number of data points with unknown labels. In general, labelled points would work as "anchors": since their label is known with certainty, the neural network is forced to fit those points with confidence, so that the information extracted from them can be reliably propagated. However, when the ratio of labelled vs unlabelled points is very unbalanced, the most likely result is that the neural network would overfit the few labelled points, which would then cease to have an influence on the unlabelled points, thus preventing effective propagation.

We make several contributions to address such shortcomings. The first contribution is a simple but effective one: we propose to use self-supervision to bootstrap a good representation of the data before running our SSL algorithm. Self-supervision defines a pretext-task where a full set of labels can be generated. While these labels are unrelated to the task at hand, we exploit the ability of deep network to *transfer effectively between tasks* and re-use the pre-trained feature to initialize SSL. Note that the data are the same, it is only the task which is being transferred.

The other contribution is a new SSL algorithm that can leverage the bootstrapped representation well. The core idea is that, instead of fitting labelled and unlabelled data points simultaneously, we alternate between two phases of optimization. The information flow between the two phases is carefully controlled to minimize the risk of overfitting. Namely, during *phase one* we fit the labelled data, but only change the final layer of the representation. Then, the resulting classifier is used to generate pseudo-labels for *phase two*, where the unlabelled data are fitted. At this point, the representation is reset to initial state (from the pre-text task)

and fine-tuned using the pseudo-labels only. Finally, the fine-tuned representation is passed back to *phase one* to retrain the classifier and obtain new-pseudo labels. In this way, the few available data points are only used to fit a small number of model parameters (a classifier layer only), whereas the large unlabelled dataset is used to fine-tune the representation. We show empirically that this greatly reduces overfitting.

We add other technical contributions to this basic scheme. The most significant in terms of final performance is inspired by cross-validation, Tri-Training [39] and weakly supervised localization [8], and amounts to split our dataset in different subsets, considering only part of the unlabelled data at each cycle of the alternate optimization above.

Empirically, we show that our method achieve *state-of-the-art* results in a large number of benchmark cases we test. Furthermore, in addition to showing that our method is effective as a standalone algorithm, we also show two other potential utilities of it. First it is able to refine results obtained from others SSL methods and second it works best among other SSL algorithm when transferring information from one dataset to another. Finally, further studies on architectures and self-supervised task were carried out to assess their importance in improving SSL accuracy.

2. Related work

There exist a vast number of classic works on SSL [3] across many disciplines [6, 30, 34]. In the classification context, one common approach is to optimise the conventional cross-entropy loss on the labelled data together with a regularisation term that propagates information to the unlabelled data. One popular form of 'regularisation' is the consistency of predictions to perturbation [1, 28]. Rasmus *et al*. [25] introduced a ladder network that minimises the reconstruction loss between the network outputs from a given sample and its stochastically-perturbed counterpart. Laine and Aila [15] simplified the ladder network into two temporal methods: Π-Model and Temporal Ensembling. Both encourage the output of the network for each unlabelled in-

stance to be as similar as possible between different training epochs, by penalising the inconsistency between current network predictions and past predictions. More recently, the Mean Teacher [33] extended these methods by ensembling over the parameter space: instead of recalling past predictions they run an exponential moving average over the weights of the network to build a Teacher model. The Teacher network is then used to enforce consistency in predictions with the main model, called the Student. Mean Teacher was extended in [18] by a certainty-driven consistency loss that focuses on the unlabelled training samples whose labels are the most uncertain. Similarly, [20] considered instead adversarial perturbations that maximise the change in model prediction. Jackson *et al.* [11] also used this idea together with gradient alignment between labelled and unlabelled data to further improve classification results.

SSL approaches alternative to perturbations include GANs [29, 31]. Chen *et al.* [5] introduced a memory-assisted deep neural network (MA-DNN) which uses an external memory module to maintain the category prototypes and provide guidance for learning with unlabelled data. Alternatively, [19] uses a label graph to refine the Mean Teacher's model predictions. When using graphs, label propagation [12, 17, 40] is also commonly used. Lee [16] proposed to iteratively assign pseudo-labels for unlabelled data and pick the high-confidence assignments as training data in the subsequent learning steps, combined with the entropy loss to further regularise training. The idea of pseudo labelling is also used alongside with interpolation in the recent work of Verma *et al.* [35]. Tri-training [39] and its deep network extension Tri-Net[4] use co-training [2] to generate three classifiers from three different portions of the data and let them label the unlabelled data. Oliver *et al.* [23] shows that consistency is generally the most resilient method as the number of labelled instances decreases, but all such methods suffer significantly when this number is decreased past a certain point.

Transfer learning is not often used in SSL. Gidaris *et al.* [7] and Oliver *et al.* [23] used a pre-trained network to fine-tune it on a restricted part of the labelled set. Closer to our approach, Zhou *et al.* [38] also combined both SSL algorithms and transfer learning. However, their approach used a pre-trained network from ImageNet[27] to train SSL methods on non-standard datasets for SSL. Our work extends their study to transfer learning from self-supervision and propose a method that surpasses current SSL technique when transferring representation across dataset.

3. Method

Our main goal is to push the limits of SSL to work with very few annotated data points. Next, we introduce several ideas (sections 3.1 to 3.3) to improve SSL algorithms in this regime and combine them in the new algorithm 1. This al-

---

Algorithm 1 Proposed Alternative Optimisation Algorithm

1: **Preparation phase:**
2: Train a self-supervised method on the whole dataset and freeze the first blocks' weights. Discard the last layer and replace with suitably dimensioned one. The trainable weights now form a network N_t
3: **Main Loop:**
4: **for** $i \in \{1, \ldots, N\}$ **do**
5:     **Supervised-training**: Fine-tune N_t classification layer on the labelled subset with cross-entropy loss $\mathcal{L}_{labelled}$.
6:     **Labels assignement**: Use N_t to assign a label y_i to each unlabelled sample x_i.
7:     **Dataset split**: Create a training set T_i from a random split of the unlabelled data.
8:     **Restart**: Reassign N_t to the weights extracted from the preparation phase.
9:     **Unsupervised-training**: Train N_t on the unlabelled metaset T_i with consistency and pseudo labelling loss: $\mathcal{L}_{unlabelled} = 0.5\mathcal{L}_{temp} + 0.5\mathcal{L}_{pseudo}$.
10: **end for**

---

gorithm consists of *preparation phase* (section 3.1, l.1 in algorithm 1) where the weights of the model are initialized from a self-supervision training and an alternate optimisation between two phases (section 3.2, l.5-9 in algorithm 1), *phase one* fitting the labelled data and *phase two* fitting the unlabelled data.

3.1. Transfer learning and self-supervision

Overfitting the labelled data points is a major issue when SSL is applied to the low-labelled-data regime. Data augmentation and other forms of regularisation can help to some extent, but their effectiveness is limited. In order to address these issues, we propose to work instead on how the data representation is initialised.

A good representation should project the images into a space that better reflects the semantic similarity between them, suppressing irrelevant differences due to nuisance factors in the imaging process and other intra-class variations, and thus reducing the need for more data augmentation and heavy regularisation. In this work, we propose to use transfer learning to initialise the representation. The idea is to pre-train the model on an auxiliary dataset or task for which labels are available, and then exploit the ability of deep networks to move from one domain to another effortlessly.

In our experiments, we consider (mostly) the case in which no external data is available. Thus, rather than transferring from one dataset to another, we propose to use self-supervision to define a pretext task on the available data, and then use the intermediate feature representation learnt

from it to bootstrap our SSL model. In fact, we show that excellent results can be obtained by just freezing a large part of the model to this initial setting and only train the rest via SSL.

3.2. Alternate optimisation

During training, current state-of-the-art algorithms generally use mini-batches containing a mix of labelled and unlabelled data. They use a sum of two losses corresponding to two different tasks: the first one enforces the classification of the labelled samples with a high confidence while the second one enforces prediction consistency across the unlabelled samples. In general, a ramp-up term is used for the consistency loss [15, 33] so that in the first iterations of training the model focuses on fitting the labelled data.

The effect of this ramp-up can be visualized using a simple 2D toy clustering problem, the 'two moons' from [20]. In fig. 1, we apply the Π-model algorithm [15] to this dataset. We see that before consistency is applied the network learns a simple boundary fitting just the anchor points with a sharp distribution enforced by the cross entropy loss. When consistency is applied, the network stretches the boundary to satisfy the consistency criterion, while becoming less certain near the boundary, but becoming more accurate overall. A drawback of this approach, however, is that since both losses are trained jointly, they must be carefully balanced with hyper parameter tuning. This is particularly true when using very few label instances as there is a large information imbalance between the two losses. Furthermore, since mini batches are generally divided between labelled and unlabelled data, the same samples from the annotated set will be seen very frequently, thus overfitting these samples.

Inspired by incremental learning techniques that try to avoid overfitting on past exemplars while learning new classes [26], we propose to disentangle the two losses by alternating the optimisation of labelled and unlabelled data points while maintaining soft constraints between them. This method reduces the number of hyper-parameters and avoids strong overfitting of the labelled data while still extracting useful knowledge from the labelled set.

In practice, after the preparatory phase of section 3.2 (l.1 in algorithm 1), each task is learnt separately in an alternate fashion (fig. 2) with a regularisation term that works as a soft constraint borrowing information from the other task.

***Phase one*: fitting the labelled data.** In the first part of our training, we optimise the cross-entropy loss on the labelled set $\mathcal{L}_{labelled}$ and train the model for a few epochs. In this part we only fine-tune the final classification layer of the network (l.5 in algorithm 1). This way the labelled samples benefit the most from the feature learnt during optimisation on the unlabelled set while not modifying the intermediate

representation. We then we use this trained network to assign soft or hard pseudo-labels to the unlabelled data. These pseudo-labels are used in the next phase to fit the unlabelled data (l.6 in algorithm 1).
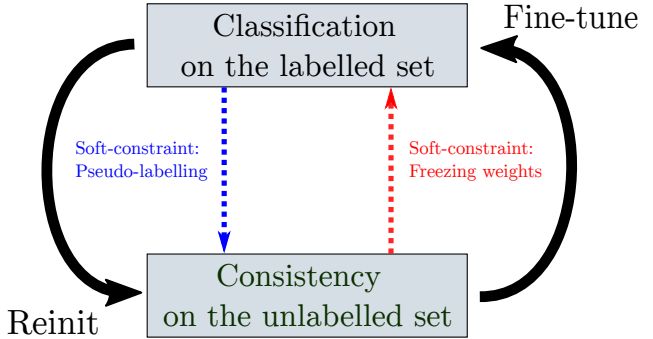

Figure 2. **Overview of our alternating optimisation method.** Starting from a pre-trained network with a self-supervised learning method we train successively on the labelled and unlabelled set. Every time a soft constraint is enforced from previous optimisation.

***Phase two*: fitting the unlabelled data.** In the second phase we reset the model with the parameters learnt during the preparatory phase of section 3.2 (see l.8 in algorithm 1). Then, we fine-tune the whole architecture on the unlabelled set using a loss $\mathcal{L}_{unlabelled}$ that is a weighted average of a term $\mathcal{L}_{pseudo}$ fitting the pseudo-labels estimated in the first phase, and a second temporal consistency term $\mathcal{L}_{temp}$, borrowed from the Π-model of [15]. The idea of the second term is to match the probability distribution p_i^{t-1} obtained from the network for a sample x_i at epoch $t-1$ with the probability p_i^t assigned at iteration t to the same sample. We use the KL-divergence between p_i^{t-1} and p_i^t during *phase two*. (l.9 in algorithm 1). The loss $\mathcal{L}_{pseudo}$, besides being a soft constraint tying *phase two* to *phase one*, can also be viewed as another consistency condition [38] that also helps reduce the entropy on the final prediction.

Information flow. Note that the model is reinitialised every time *phase two* is entered. This is done to avoid getting stuck in local minima. Hence, the information that flows (fig. 2) from *phase one* to *phase two* is only the pseudo-labels on the unlabelled data, whereas the information that flows from *phase two* back to *phase one* are the weights of the network fine-tuned on the unlabelled data.

While these choices may seem ad-hoc, they constitute the primary reason why our algorithm is able to work well in a low-labelled-data regime by avoiding overfitting. Thanks to this technique, the algorithm focuses on fitting one loss at a time while still allowing information to be exchanged between the two subtasks. We show in section 4 this approach robustly improve the quality of the overall classifier

at every step.

3.3. Dataset split

One potential pitfall of alternating optimisation is the drift that may be induced by training for a long time on the unlabelled data. Inspired by cross-validation and co-training [2], we also suggest to split the dataset into several parts. However, differently from [2] we do not train different models from each split. Instead, at the beginning of each training cycle we randomly choose two-thirds of the unlabelled training data to learn from (l.7 in algorithm 1). The rest of the unlabelled data is held out to reduce the risk of overfitting to it in the current training cycle. This way, the held-out samples are more likely to have their labels swapped during pseudo-label generation in the next cycle. Roughly two-thirds of this "fresher" held-out data will be used to fit the model in the next cycle. The benefits of this reshuffling procedure diminish as training nears completion, so all the data is used in the final few phases of training. In section 4 we demonstrate empirically the benefits of this approach.

4. Experiments

Datasets. Following [5] we mostly use standard image classification tasks suitably modified to the semi-supervised context. Each dataset is randomly split in two sets: one small set containing images with their corresponding labels evenly divided among classes and another set with the rest of the image without annotations. For each dataset we evaluated our method on the publicly available test set. We used the same datasets as [5]:

SVHN [22]. A Street View House Numbers dataset including 10 classes (0-9) of coloured digit images from Google Street View. The classification task is to recognise the central digit of each image. We use the format-2 version that provides cropped images sized at 32×32, and the standard 73,257/26,032 training/test data split. We centered and scaled all images RGB channels based on the extracted dataset statistics and augmented the data with random cropping.

CIFAR-10 [14]. A natural images dataset containing 50,000 training and 10,000 test image samples from 10 object classes. Images have a 32×32 resolution and are evenly divided among classes. We again pre-processed the data with centering and scaling and augmented the data with random cropping and flipping.

CIFAR-100 [14]. A dataset (with the same image size as CIFAR-10) containing 50,000/10,000 training/test images from 100 more fine-grained classes with subtle inter-class visual discrepancy. Same as before, we centered and scaled the images and used the same data augmentation as for CIFAR-10.

Baselines. We used Mean-Teacher [33] as the main baseline to experiment alongside with our proposed method. Additionally we also used MA-DNN [5] on a more restricted set of experiments. For both we used publicly available code provided by the authors[1].

Implementation details. Unless stated otherwise, we used the ResNet-18 [10] model with then average pooling and skip connections. We chose RotNet [7] as the self-supervision method as it is easy to implement and integrate in any pipeline. For all networks we trained this proxy task on 200 epochs with a step-wise decaying schedule. A more thorough study is done in section 4.6. All networks are trained using Nesterov accelerated gradient method [21]. We use the same hyper-parameters when training with and without transfer learning, which have every time been tuned on the specific task.

4.1. Impact of self-supervision

We first compare Mean-Teacher and our algorithm on standard classification benchmarks. For every dataset we trained the network with 10/25/50/100 labels per class, with an extra experiment of 400 labels per class on CIFAR-10 as is common practice [5, 15, 33]. In every experiment we keep the same split of the data for each method. In the default case, models are trained from scratch. For every method we also used transfer learning from self-supervision, with only the last two blocks and the last classification layer used for semi-supervised training (Legends containing 'with T.L.'). Results are shown in fig. 3.

This experiment clearly demonstrates the importance of transfer learning as the number of labelled data per class decreases. For instance, on CIFAR-10, transfer learning from self-supervision used with semi-supervision makes an improvement of over 50 points in classification error for both our method and Mean Teacher. On SVHN and CIFAR-10 the performance from 100 labels per class to 10 labels per class decreases by less than 5 points in error. Performances on CIFAR-100 dropped by a larger amount when having fewer labels due to the higher complexity of the dataset, although the performance deteriorate noticeably more slowly when using self-supervision pre-training. Overall this means that self-supervision enables networks to train with competitive results in the extremely low data setting where standard SSL methods fail. We also emphasise that we only use the dataset provided and did not use any regularisation method and very limited data augmentation. This is particularly crucial and demonstrates that a carefully chosen self-supervision method can be sufficient to extract very good features which are sufficient to train good models with few labels.

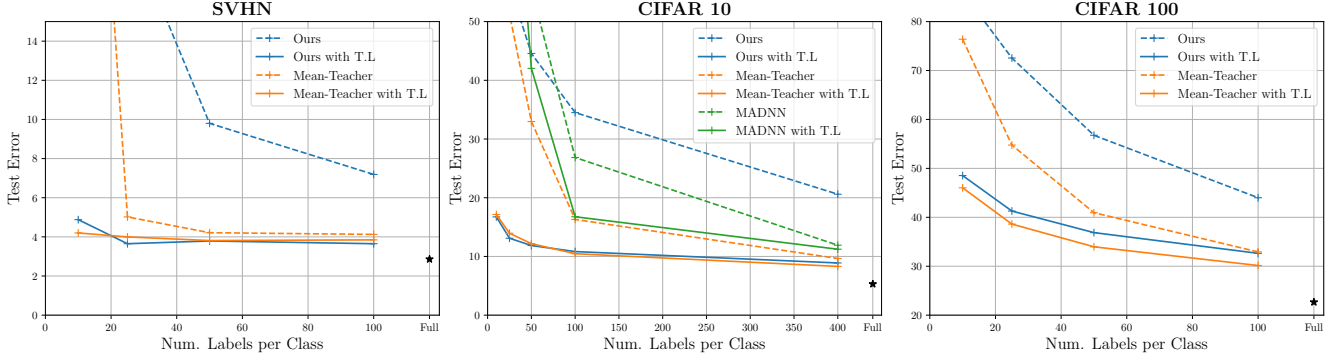[1] https://github.com/CuriousAI/mean-teacher; https://github.com/yanbeic/semi-memory

Figure 3. **Impact of Tranfer learning for SSL classification benchmark.** For each dataset we vary the number of labelled data per class in $10, 25, 50, 100$ with an additional experiment on CIFAR-10 with 400 labels per class. We note that for every method, models benefit from self-supervision. The 'with T.L' indicates a model pretrained with RotNet[7]. In addition we can see that our method is getting competitive results with Mean Teacher. The star point '*' denotes our network trained with supervision only on the full training set. The shown results are the average of 10 runs per setting.

For every dataset we see in fig. 3 that our method and Mean-Teacher obtain similar performance when they both use transfer from self-supervised networks. However our method needs a more careful initialisation as its performance dropped for models trained from scratch. This can easily be understood, as fine-tuning on the labelled set would be more efficient with either an increased number of labelled data or better intermediate representation. In the rest of the paper we call a 'good' representation of an image, a representation from which it is easier to discriminate different classes.

In addition, we show that self-supervision can also improve the state-of-the-art performance of a more recent SSL method. To this end we use MA-DNN [5] with the 10-layers model proposed in their paper. Again we used RotNet [7] as the self-supervision method and only fine-tune the last block and linear layer on the SSL task. We see in fig. 3 that again, transfer from self-supervision consistently achieves the best performance, improving their *state-of-the-art* results on CIFAR-10 from 11.9% error to 11.1% using 400 labels per class. With the hyper-parameters given in their paper, we demonstrate again that under equivalent settings, self-supervision is beneficial to SSL. Finally we found that MA-DNN was more sensitive to the number of labels per class. This means that building a reliable template of every class requires more annotations, while we observed that consistency based methods are seemingly showing better robustness overall.

4.2. Ablation study

To assess the impact of every component of our method we also conducted an ablation study on the CIFAR-10 SSL task with 10 labels per class.

We first measure the gain of using random splitting of the unlabelled set through training cycles as opposed to using the whole set. We observe in fig. 4 that random dataset

splitting allows much faster progress in the course of training. When only using pseudo-labelling on the unlabelled set and no temporal consistency, this technique also leads to a better optimum.

Then we analyse the effect of the consistency in the training cycle. The dashed curves correspond to cases where the temporal consistency is not used. It means that, when training on the unlabeled set, we only optimise the cross-entropy on the pseudo-labels previously assigned during the training phase on the labelled set. Interestingly, we also note that the performance is still increasing over cycles even without an explicit consistency loss. In fact, this corroborates the statement in Zhou *et al.* [38], that the assigned pseudo-labels act as an implicit consistency across samples (instead of the explicit consistency across augmented versions of the same sample) which improves the representation for the next training phase on the labelled set.

Finally we can observe for the different settings that the performance is steadily increasing over cycles. The training indeed does not stall after a few cycles but instead keeps improving the feature representation over the course of alternating optimisation cycles. Alternating optimisation therefore refines the results every cycle with each optimisation helping the next one to reach a better optimum.

4.3. A refining SSL algorithm

In fig. 3 we note that our alternating training method is particularly efficient when using a good representation rather than starting from scratch. Hence we hypothesise that our method could make an even bigger difference when starting from a better network representation. One way of doing it is to build our method on top of others in order to refine their results. To validate this hypothesis, we apply our method as a refinement on CIFAR-10 and CIFAR-100 datasets for networks pre-trained with Mean-Teacher [33]. It should be noted that, as the starting features are already
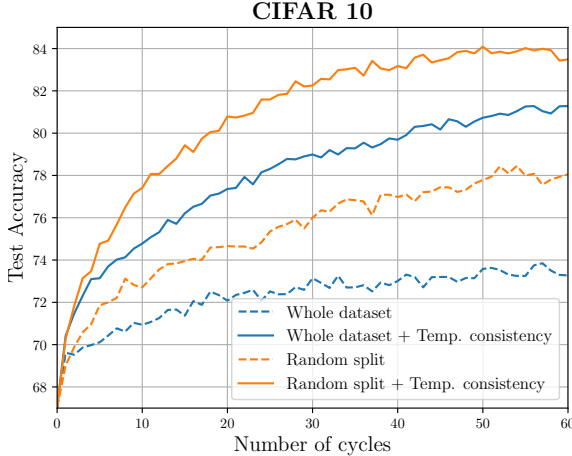
Figure 4. **Ablation study.** Starting from a pre-trained model, our alternating optimisation method makes steady progress over cycles. Dataset random splitting is also effectively speeding-up training. Default case uses $\mathcal{L}_{unlabelled} = \mathcal{L}_{pseudo}$. 'Temp. consistency' indicates that we use both $\mathcal{L}_{temp}$ and $\mathcal{L}_{pseudo}$.

	CIFAR-10	CIFAR-100	
Labels per class	10	10	25
MT	32.4	23.6	45.7
MT + Self-Sup	86.4	53.7	60.7
MT + Self-Sup + Refinement	**87.7**	**58.3**	**63.9**

Table 1. **Our method applied as a refinement of Mean Teacher ('MT').** For different numbers of labels per class our method always improve the testing accuracy by a big margin.

competitive, we do not use the dataset random splitting in our training procedure and we choose soft labels assignment in l.6 of algorithm 1.

In table 1, we see that in every case our method used as refinement consistently improves the test accuracy. The synergy of alternating training is enabling the network to gain as much as 4.6 points in test accuracy on CIFAR-100 and 1.3 points on CIFAR-10. The refinement technique combined with self supervision allows us on CIFAR-10 to almost match previous *state-of-the-art* results with methods trained with 400 labels per class (11.9% reported in [5]) whereas we only used 10 labels. These results confirm our previous hypothesis and motivate us to see the impact of increasingly better representations in the next section.

4.4. Transfer learning from different tasks

So far we have only used the dataset at hand to extract our intermediate representation with self-supervised learning. In this section we use a different type of transfer learning used in [38]: we learn our representation on a different classification dataset. To this end, we use networks pretrained on ImageNet [27]. As is done in the other sections, we then fix the first two blocks of our ResNet-18 network and train the rest of the weights on the hardest semi-

	C-10	C-100	P-10	MIT-67
Fully Sup.	96.6	82.5	83.3	75.5
Labelled Set	82.5±1.0	62.8±0.6	66.4±1.7	55.1±0.8
M.T.[33]	88.0±1.5	63.3±0.7	72.8±1.2	58.8±1.3
Ours	**93.5±0.4**	**69.2±0.4**	**75.8±1.7**	**63.0±0.7**

Table 2. **Transfer Learning: Test accuracy on the target task.** Transfer is from training on ImageNet. We compare our method with Mean Teacher (M.T.) and supervision on the labelled set only ('Labelled set') for 10 labelled sample per class. Our method consistently outperforms both method on this task and achieves results nearly as good as the one obtained by training on the full labelled set.

supervised learning task of 10 labels per class using either Mean Teacher or our alternating optimisation method (see section 3). We experimented with CIFAR-10 and CIFAR-100 with pretraining on a downsampled 32×32 version of ImageNet, and also MIT Indoor-67 [24], and 'Places10' where we only kept 10 classes from Places [37] using a pretraining on the standard version of ImageNet. Results are shown in table 2 where we perform 10 runs per setting. Fine-tuning on the full set is given as a reference.

We note that our method consistently outperforms Mean Teacher for such scenarios. This result corroborates our assumption in the previous section that our method performs increasingly well with better representation. In fact, as also noted in [23], since ImageNet has redundant classes with the datasets, we can consider the representation learnt from ImageNet as an upper bound. Thus it is not entirely surprising to see our method obtaining nearly as good results as if it were using the complete set with 93.12% accuracy compared to 96.56% when fine-tuning on the full labelled set on CIFAR-10. As our method aims to disentangle the two losses by using distillation to preserve the effect of the very reduced labelled set, its effect is magnified when using a representation as good as the one learnt on ImageNet. In contrast, techniques like Mean Teacher overfit on a small labelled set, which in turns will lead to the degradation of the representation coming from ImageNet.

4.5. Evaluating representation quality with SSL

In the context of SSL for images classification, practitioners usually use shallow networks. If some works [33, 35] also experimented with deeper and wider architectures, then those were generally applied when the number of labels at hand was large enough (4,000 on CIFAR-10, ~100,000 on ImageNet). So far in this work, we have shown that it is possible to train a deeper architecture by using transfer-learning from self-supervision. In practice, what matters in our context is the quality of the extracted intermediate representation that we will train our model from.

In this section we experiment with two other architectures and use SSL to measure the quality of their repre-

N	TempEns[15]	RevNet-18[9]	ResNet-18[10]
10	18.4±1.9	20.0±2.4	**16.4±2.6**
50	16.3±0.9	13.7±0.7	**11.9±0.4**
100	14.8±0.6	12.2±0.5	**10.8±0.4**
400	12.8±0.2	10.5±0.4	**8.9±0.2**
Full	6.7	5.8	5.3

Table 3. **Model variation.** Test error on CIFAR-10 with different number of labels per class 'N'. In all cases networks, were pre-trained with RotNet [7]. First two block are then frozen while the rest of the network is fine-tuned on the classification task. 'Full' indicates that we used the full training set with a fully supervised method. All methods uses the same hyper-parameters.

	Self-supervised training stopped at epoch				FT
	60	120	150	200	
Rotation accuracy	74.1	83.5	85.9	86.1	**86.8**
Labelled set	37.6	40.4	**40.9**	40.7	40.4
SSL (our method)	46.5	50.2	50.4	50.7	**52.3**

Table 4. **Rotation accuracy vs Classification accuracy.** Correlation between rotation accuracy and final SSL classification accuracy on CIFAR-100 with 10 labels per class. 'FT' is fine-tuned model for 500 epochs. 'Labelled' set indicates fine-tuning on label set only.

sentation. As is done in [13], we compare with one fully-convolutional model and a RevNet[9] model. We chose the fully convolutional 10-layers/3-blocks model from [15] which we named 'TempEns'. We provided our own RevNet implementation named 'RevNet-18' aimed to match the ResNet-18 we used in this work. More specifically, the RevNet-18 has four main blocks and uses the same type of downsampling mechanism as our ResNet-18. We pre-trained all models with RotNet and trained our SSL method on CIFAR-10 with different numbers of annotated instances. For all architectures the first two blocks of the networks are fixed during SSL training, as we found it was giving best results for every architecture. We report results in table 3.

We found that the ResNet-18 has better performance over all methods. TempEns surprisingly remains very competitive for all cases with less weights trained than both methods, with good robustness to extreme cases. While RevNet is almost always able to reach ResNet-18 level of performances. With such sparse data, since RevNet has the most parameters to fine-tune it is thereby the most prone to overfit. Overall, the RevNet results slightly contrasts with the one found by Kolesnikov *et al.* [13], where a RevNet representation surpasses ResNet when training with rotation. However, we note that their methods of evaluation differs significantly from ours. These results emphasise that while K-means neighbors or full supervision can be used to evaluate self-supervised learning [13, 36], SSL could also be used as a way to evaluate the intermediate representation learnt from self-supervised learning algorithms.

4.6. Self-supervision performance vs classification accuracy

In general, different self-supervision tasks would lead to different intermediate representation. However one could also wonder how much solving the proxy task could help to find a representation that would be best suited for the classification task. Intuitively, heavy fine-tuning on the proxy task could lead to a more specialized network which would be harder to adapt to the classification task. To ver-

ify this, we studied the rotation accuracy impact on the final SSL performance. As previously mentioned we generally trained our self-supervised method for 200 epochs with a step-wise decaying learning rate starting from 0.1. The learning-rate is decayed by a factor 10 at epochs 60, 120, and 160. We saved snapshot of the network before decaying the learning rate and measure the performance reached by our method on the CIFAR-100 classification task with 10 labelled instances per class. In addition, we also trained a RotNet for 500 epochs with a learning rate schedule directly extracted from [5]. We summarise the results in table 4.

Overall we found that fine-tuning on the proxy task was beneficial to the SSL algorithm. Interestingly, we note that the accuracy on the labelled set does not entirely correlate with the final SSL accuracy. While the model with best accuracy on the labelled set obtained 50.4% SSL accuracy, the fine-tuned model obtained an overall SSL accuracy of 52.3%. This also corroborates the results found in section 4.5, better accuracy on the complete or partial set does not necessarily mean a better final SSL accuracy. We believe this result also motivates a new method of self-supervised learning evaluation using SSL.

5. Conclusion and discussion

In this paper, we have pushed the limits of SSL to work with very few annotations. We achieved this goal with several innovations. We first proposed to leverage the power of transfer learning among different tasks and self-supervision to provide a good initial feature representation for SSL. We also proposed a new SSL algorithm that can exploit well such a pre-trained representation, achieving state-of-the-art results on a large number of public benchmarks. We showed that self-supervised learning can not only benefit our proposed SSL algorithm, but also existing state-of-the-art methods. Equipped with the representation learned from self-supervised learning, we found that the performance of state-of-the-art methods can be notably boosted, especially the extreme cases with sparsely annotated data. More generally, we found that SSL could be a new and sensitive downstream task for ranking self-supervised learning methods. Besides working as a standalone method showing su-

perior performance, our method can also serve as an effective refinement method on a network pre-trained either with an existing SSL algorithm or another dataset. Although we have demonstrated our SSL method here on classification, it is equally applicable to other tasks. As future work, we plan to apply the same technique to other tasks and datasets such as tracking or medical imaging.

References

[1] P. Bachman, O. Alsharif, and D. Precup. Learning with pseudo-ensembles. In *NeurIPS*, 2014.

[2] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Conference on Computational learning theory*, 1998.

[3] O. Chapelle and A. Scholkopf, B.and Zien. *Semi-Supervised Learning*. MIT Press, 2006.

[4] D.-D. Chen et al. Tri-net for semi-supervised deep learning. In *IJCAI*, 2018.

[5] Y. Chen, X. Zhu, and S. Gong. Semi-supervised deep learning with memory. In *ECCV*, 2018.

[6] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NeurIPS*, 2009.

[7] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

[8] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2409–2416, 2014.

[9] A. N. Gomez et al. The reversible residual network: Backpropagation without storing activations. In *NeurIPS*, 2017.

[10] K. He et al. Deep residual learning for image recognition. In *CVPR*, 2016.

[11] J. Jackson, J.and Schulman. Semi-supervised learning by label gradient alignment. *ArXiv e-prints*, 2019.

[12] K. Kamnitsas et al. Semi-supervised learning via compact latent space clustering. In *ICML*, 2018.

[13] X. Kolesnikov, A. andZhai and L. Beyer. Revisiting self-supervised visual representation learning. In *ArXiv e-prints*, 2019.

[14] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Technical report*, 2009.

[15] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *ICLR*, 2017.

[16] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013.

[17] Q. Li, X.-M. Wu, and Z. Guan. Generalized label propagation methods for semi-supervised learning. *ICLR*, 2019.

[18] Y. Li, L. Liu, and R. Tan. Certainty-driven consistency loss for semi-supervised learning. In *ArXiv e-prints*, 2019.

[19] Y. Luo, J. Zhu, et al. Smooth neighbors on teacher graphs for semi-supervised learning. In *CVPR*, 2018.

[20] T. Miyato et al. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE TPAMI*, 2018.

[21] Y. Nesterov. A method for solving the convex programming problem with convergence rate o (1/k^ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[22] Y. Netzer et al. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[23] A. Oliver et al. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, 2018.

[24] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[25] A. Rasmus et al. Semi-supervised learning with ladder networks. In *NeurIPS*, 2015.

[26] S.-A. Rebuffi et al. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.

[27] O. Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[28] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NeurIPS*, 2016.

[29] T. Salimans et al. Improved techniques for training gans. In *NeurIPS*, 2016.

[30] M. Shi and B. Zhang. Semi-supervised learning improves gene expression-based prediction of cancer recurrence. *Bioinformatics*, 27(21):3017–3023, 2011.

[31] J. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *ArXiv e-prints*, 2015.

[32] N. Srivastava et al. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research (JMLR)*, 2014.

[33] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017.

[34] A. Teichman and S. Thrun. Tracking-based semi-supervised learning. *The International Journal of Robotics Research*, 31(7):804–818, 2012.

[35] Verma.V et al. Interpolation consistency training for semi-supervised learning. In *ArXiv e-prints*, 2019.

[36] L. Zhang et al. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *CVPR*, 2019.

[37] B. Zhou et al. Learning deep features for scene recognition using places database. In *NeurIPS*, 2014.

[38] H. Zhou et al. When semi-supervised learning meets transfer learning: Training strategies, models and datasets. In *ArXiv e-prints*, 2018.

[39] Z.-H. Zhou and M. Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge & Data Engineering*, 2005.

[40] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.