Smooth Pseudo-Labeling

Anonymous authors
Paper under double-blind review

Abstract

Semi-Supervised Learning (SSL) seeks to leverage large amounts of non-annotated data along with the smallest amount possible of annotated data in order to achieve the same level of performance as if all data were annotated. A fruitful method in SSL is Pseudo-Labeling (PL), which, however, suffers from the important drawback that the associated loss function has discontinuities in its derivatives, which cause instabilities in performance when labels are very scarce. In the present work, we address this drawback with the introduction of a Smooth Pseudo-Labeling (SPL) loss function. In our experiments, we test our improvements on FixMatch and show that it significantly improves the performance in the regime of scarce labels, without addition of any modules, hyperparameters or computational overhead. Robustness with respect to variation of hyperparameters and training parameters is also significantly improved. We also introduce a new benchmark, where labeled images are selected randomly from the whole dataset, without imposing representation of each class proportional to its frequency in the dataset. We see that the smooth version of FixMatch does appear to perform better than the original, non-smooth implementation. However, more importantly, we see that both implementations do not necessarily see their performance improve when labeled images are added, an important issue in the design of SSL algorithms that should be addressed so that Active Learning algorithms become more reliable and explainable.

1 Introduction

The massive successes of modern supervised Machine Learning techniques in all standard benchmarks in Computer Vision, Foret et al. (2020); Kabir et al. (2020); Dai et al. (2021); Dosovitskiy et al. (2020), have made possible the relaxation for need of supervision. The field of Semi-Supervised Learning (SSL) thus gained considerable attention, as it promises, under some conditions, to lighten the need for annotated data, condition sine qua non for supervision. In such settings, one generally assumes a large dataset, the greatest part of which is non-annotated, the remaining part having been annotated for the relevant task. In our exposition we will fix the task as Image Classification, so as to avoid having to introduce general and complex notions and notation. Annotations, even for Image Classification, can be costly to obtain, not to mention more difficult problems, in some cases demanding man-hours from highly trained professionals. On the other hand, non-annotated images are orders of magnitude cheaper to obtain. SSL thus seeks to alleviate the cost of training by transforming man-hours to machine-time, while keeping the performance of the model at the same level. The core quest of SSL is, thus, to lower the percentage of data that needs to be annotated in order to keep the same performance as if the whole dataset had been annotated, for a given dataset and task.

SSL approaches rely on a variant of Siamese networks, where two branches of neural networks consume different views of the same image, augmented to a different degree. Some kind of consistency is imposed at the output, stemming from the intuition that if the augmentations are not so violent as to totally distort the content of the image, the output of the network should remain invariant. A popular strategy was inaugurated by the introduction of Pseudo-Labeling Lee et al. (2013) consisting in using the model being trained to infer labels for non-annotated data and using them in conjunction with a small amount of annotated data. Crucially, the score of the dominant class at inference on non-annotated data must exceed a given threshold

to be considered as reliable enough and define a pseudo-label. Such a cut-off for admitting pseudo-labels introduces a discontinuity in the derivative of the loss. The fact that the loss function is iteratively defined during training leads to an accumulation of discontinuities and can generate instabilities that can be described in terms of a signal-to-noise ratio problem. The signal in this case comes from supervision, which feeds the network with information independent of the current state of the network. Self-supervision (on unlabelled data), which depends on the current state of the network and contributes to the determination of the next state, is inherently noisy, especially so in the early stages of training, before some level of confirmation bias kicks in Arazo et al. (2020). The fact that instabilities due to the accumulation of discontinuities have not yet been observed widely can be attributed to the strength of the signal needed for performances close to the fully supervised baseline to be obtained in most benchmarks.

FixMatch Sohn et al. (2020) is built upon the Pseudo-labelling principle but obtains the pseudo-label by inference on a weak augmentation of the image and learns it on a strong augmentation. It was the first improvement on Pseudo-Labeling to reach a performance close to the fully supervised baseline on CIFAR-10 with only 40 labels, 4 per class, and thus a very weak supervision signal. As show our experiments, in the best cases, FixMatch can reach $\sim 7\%$ error rate, close to the 4% achieved by the fully supervised baseline. However, it can also go as high as $\sim 15.5\%$, presenting a standard deviation of $\sim 3.00\%$ over 6 independent experiments (cf. table 1 or Sohn et al. (2020)). This was indeed observed in Sohn et al. (2020), but not attributed to any particular factor. We establish that this kind of instability can be attributed to a considerable extent to the discontinuity of the derivative of the loss function minimized in Pseudo-Labeling and, consequently, in FixMatch.

We note that such instabilities are not present in the experiments on CIFAR-100 or ImageNet, as performance remains further from the fully supervised baseline, since the dataset is more difficult, even with considerably stronger supervision. The phenomenon is nevertheless likely to appear in the future, if the methods that bring the performance of SSL and supervised methods closer together still rely on non continuously differentiable (i.e in $C^0 \setminus C^1$) loss functions. It is also bound to create problems in Active Learning scenarios where labels are added during training. In such cases, the correct response of the model to the strengthening of the signal is not guaranteed, even in the absence of discontinuities in the derivative. In our experiment section we see, however, that, even though smoothing the discontinuities does not solve all issues with PL, it does result in a better overall performance.

Subsequent improvements on FixMatch added modules, resulting in additional computational cost, and/or discontinuities, and/or hyperparameters, or hypotheses on label distribution of non-annotated data Zhang et al. (2021), but none addressed the discontinuity issue.

The fact that the loss minimized by algorithms using Pseudo-Labeling Lee et al. (2013) is $C^0 \setminus C^1$ has, to our best knowledge, not been observed so far in the literature on the subject, despite the latter being quite abundant. Minimization of a function whose derivative presents discontinuities is a serious flaw from the point of view of optimization theory, as it can lead to a sensitive dependence of the point of convergence of the related minimization algorithm with respect to the irreducible stochastic factors of training (random initialization, order and composition of batches, etc.). Non-Differentiable Optimization is an active field of research, relying on specific approaches such as subgradients Lemaréchal (1989). Arguably, failure to observe this important drawback in the method is due to the probabilistic and information-theoretic point of view that is predominant in the field of Machine Learning. We argue that, in view of the cost-less improvement of PL that this observation brings about, as well as the success of the Focal loss function in some settings Lin et al. (2020), the optimization point of view should be be given more consideration in the design and the construction of machine learning algorithms. We note that neither the Focal loss, nor the SPL loss introduced in this work have any information theoretical content.

We thus focus in the regime of scarce labels and high performance, where SGD should be expected to be less robust due to the discontinuities in the derivative of the loss. The usual way of comparing methods is by comparison of average performance over a certain number of runs and its standard deviation. However, in our study we found that this is insufficient, mainly due to the fact that standard deviation is agnostic of the sign.

We argue that when one compares two methods over several runs, it is more rigorous to estimate their difference with an actual statistical test, as is usually done in other scientific fields. The Wilcoxon test was introduced in Wilcoxon (1945) and tests whether the median of an increasing array of numbers is significantly non-zero, positive or negative, depending on the particular version of the test (two-sided, one-sided greater or one-sided less). It produces a confidence level, a p-value which measures the probability of having the given result if methods A and B are equivalent, method A is better than B or vice versa (for the two-sided, one-sided greater or one-sided less tests, resp.).

The main contribution of this article is to investigate the consequence of using a continuously differentiable loss in a regime of scarce labels for SSL, leading to "smooth pseudo-labelling" (SPL). A subsequent contribution is to introduce a novel protocol to compare the methods in such settings. Using the Wilcoxon one-sided test, we found that the proposed smoothing improves the performances of FixMatch ten times more significantly ($p \sim 0.0156$) than FlexMatch does ($p \sim 0.1562$).

As we discuss in the related work section, §2, a number of models in the literature on both the balanced and imbalanced class setting use in a crucial way knowledge of the class distribution of the relevant datasets. This is information that is, by the very definition of SSL, not accessible, and use of such information is bound to lead to increased preformance, but to the detriment of applicability of the method in real-life conditions, where its assumptions are not verifyable.

This kind of assumption is also hardcoded in the benchmarks on which FixMatch is tested in the original article Sohn et al. (2020), where labeled images are selected so that their distribution agrees with the one of the total dataset. In a real-life scenario, the class distribution of the total dataset is unknown and such sampling is impossible. In order to bridge the gab between experimental and applied settings, we added a benchmark with random sampling of labeled images, and compared our method with FixMatch on it. We refrained from testing other models proposed in the literature, since their application would not be out-of-the-box as their assumptions on class distribution of the (un)labeled dataset were not directly verifiable.

2 Related work

Two main branches can be distinguished in the field of SSL. The first one, heavier in computational cost, relies on Self-Supervised pretraining, using the whole dataset without annotations in order to learn general, possibly even task-agnostic features. The pretrained model then learns the annotated data in a traditional supervised manner and teaches a simpler model its predictions on both annotated and non-annotated data Caron et al. (2021); Grill et al. (2020); Chen et al. (2020). The second one, considerably cheaper in computational cost, consists in a combination of supervised learning for annotated data coupled with self-supervision for non-annotated data, all at the same time Lee et al. (2013); Sohn et al. (2020); Berthelot et al. (2019b). As should be expected, the first type of strategy performs better in more difficult datasets such as Imagenet, but strategies of the second kind have already been successful in reducing the annotation cost to a minute percentage of simpler datasets such as CIFAR-10 or CIFAR-100, Krizhevsky et al. (2009). Naturally, a combination of both approaches, i.e. using a strategy of the second type in order to train a pretrained model should be expected to boost performance.

To the best knowledge of the authors, all recent improvements on PL in general and on FixMatch in particular Pham et al. (2021); Lee et al. (2022); Yang et al. (2022); Zhang et al. (2022); Xie et al. (2020); Berthelot et al. (2019a); Kim et al. (2021); Li et al. (2020); Xu et al. (2021), have either preserved the issue of discontinuities in the derivative of the loss function or even worsened it by adding thersholding. In any case, the general tendency is to add modules on top of FixMatch, which results in additional hypermarameters and sometimes discontinuities in the derivative. Moreover, the value of some additional hyperparameters is often changed when passing to more difficult datasets without an a priori, i.e. agnostic of the test-set of the new dataset, justification of the new value. All this comes with computational overhead and/or additional hyperparameter fine-tuning.

FlexMatch, Zhang et al. (2021), did obtain high performance in certain benchmarks but this came at the cost of sneaking in a hypothesis on the distribution of classes in the unlabeled part of the dataset. CPL, the pseudo-labeling strategy used in FlexMatch, lowers the threshold for accepting pseudo-labels for classes that

are less represented in pseudo-labels. The weaker representation of a class in the number of pseudo-labels is interpreted by the authors of Zhang et al. (2021) as a difficulty of the model to learn the given class. Even though this can definitely be the case, another possible reason can be that the given class is indeed less represented in the dataset. Knowing which of the two is the actual reason can only be known for sure (as is assumed in the construction of CPL and therefore of FlexMatch) only by annotating the unlabeled dataset. For the heuristic to work, therefore, a hypothesis on class distribution is indispensable, and the one imposed in Zhang et al. (2021) is uniform class distribution. In §3 we argue that FlexMatch is for this reason not an SSL algorithm on par with the rest in the literature, and in §4.3 we establish to what point FlexMatch depends crucially on the alignment of class distribution between labeled and unlabeled datasets. These observations and results question the validity of the remark of the authors of FlexMatch concerning the improvement on the Imagenet 10% benchmark

> `This result indicates that when the task is complicated, despite the class imbalance issue (the number of images within each class ranges from 732 to 1300), CPL can still bring improvements.`

Firstly, this is as an explicit admission of the fact that class distribution is assumed to be uniform as we were able to find in the article, another one being the discussion in paragraph "Comparison with class balancing objectives". We were unable, however, to find a trully explicit mention of this assumption in the construction of the algorithm, or in the abstract. In the abstract the fact that

> `CPL does not introduce additional parameters or computations (forward or backward propagation)`

is mentioned, but the fact that additional assumptions are imposed on the (unlabeled) dataset is omitted. We think that making this issue clear is important for the community, as it seems to have gone under the radar of both the peer-review process and the general knowledge of the community. We would expect this kind of assumptions to be made explicit by the authors of the paper whenever they are introduced.

We note that in the CIFAR-100 and Imagenet 10% benchmarks FixMatch (and FlexMatch, as well) collapses a number of classes, i.e. the corresponding columns in the confusion matrix have 0.00 entries, which mechanically implies that some classes are overloaded. Imposing uniformity of class distribution with pseudo-label purity of the order of 60% should be expected to remove more False Positives from the overloaded classes than True Positives, since already random selection is a TP with a rate of only $\sim 0.1\%$. Transferring FP to the formerly collapsed classes will then, in an almost mechanical way, entail some improvement in pseudo-label purity and consequently in accuracy on the test-set, even by pure chance. The reasons for the marginal improvement of the error rate by 2%, from $\sim 43.5\%$ to $\sim 41.5\%$, in the Imagenet 10% benchmark should be sought in an adapted variant of this argument and not assumed to indicate robustness of CPL with respect to class imbalance in any way.

In order to establish the cruciality of this assumption for the improvement of performance of FlexMatch over FixMatch, we needed to address the class imbalance regime in order to establish empirically that FlexMatch overfits the standard benchmarks by hardcoding the uniformity of class distribution in the method. In the setting where datasets are not a priori supposed to have uniform class distribution, as is the case in the CIFAR datasets or Imagenet, recent advances include Kim et al. (2020); Lai et al. (2022); Wei et al. (2021).

In Kim et al. (2020) the class distribution of both labeled and unlabeled data is by construction assumed to be geometric, and a very quick sensitivity analysis is carried out only with respect to the ratio of the geometric progression. In particular, when the ratio of the geometric progression is assumed to be the same for both datasets, the class distribution is assumed to be identical, which is a very strong and unverifiable hypothesis. Moreover, it is assumed that the ordering of classes is the same for the labeled and unlabeled datasets, a hypothesis that cannot be verified, and can only be assumed to be reasonable if the labeled dataset is a large proportion of the total data. The method is validated only with abundant labeled data, and our study of FlexMatch in §4.3 already shows the limits of methods based on such hypotheses when subjected to the stress test of scarce labels.

In Lai et al. (2022), the authors claim that assuming identical ordering of classes with respect to frequency in the labeled and unlabeled datasets entails no loss of generality. This is an obvious mistake, and the ordering can only be trusted (still with a certain probability of error) again when labeled data are assumed abundant. Again, class distributions are assumed to follow the same mono-parametric law (geometric distribution) and the only, limited, sensitivity analysis is carried out with respect to this unique parameter. Again, 1/3 of the dataset is assumed to be labeled, so that the assumptions become plausible, even though no calculation is provided as to the reliability of the hypotheses when labeled data are drawn *iid* from the total dataset.

In Wei et al. (2021) at least 10% of the dataset comes with labels and to our best understanding the class distribution is by construction geometric and the sampling rate for keeping pseudo-labels is also very conveniently supposed to be geometric of opposite monotonicity, while no sensitivity analysis with respect to this hypothesis is included.

We see, therefore, that improvements on FixMatch in the class imbalanced setting either add assumptions considering the class distribution of the unlabeled dataset in the same way as FlexMatch, or they add modules, in which case they need more hyperparameters, or eventually both.

Needless to say, the only way in which exact knowledge of the class distribution can be obtained is by labeling the whole unlabeled dataset, which defeats the purpose of SSL and would beg the question "why not use the labels instead". In the case where the unlabeled dataset is indeed assumed to be unlabeled, as the wording suggests, only an estimation of the class distribution of the unlabeled dataset is available. The quality of the estimation, as can be seen by simple probability arguments, depends crucially on the proportion of labeled images compared with the size of the dataset, and deteriorates as the proportion becomes small. An exhaustive sensitivity analysis with respect to the misalignment between estimation and reality is then needed in order to establish the usefulness of the method in real-life scenarios where the dataset is given and a part of it is selected iid and then annotated, as opposed to the current practice of constructing the labeled and unlabeled datasets the one next to the other assuming some shared properties.

To the end of clarifying the confusion in the order by which the objects in SSL are given (first the whole dataset and then the labeled part), and to bridge the gap between benchmarks and real-life conditions, we introduce the CIFAR-10_random benchmark, in which we only impose a number of labels and no uniformity of class distribution or any other assumption on the labeled dataset.

Finally, since our method has the same computational overhead and hyperparameters as FixMatch and no additional assumptions, and since, more crucially, we improve one of the bases upon which FixMatch is constructed (namely PL), we will only compare our method only with FixMatch. Benchmarking the other methods mentioned here above would demand a significant amount of work for lifting or, to the least, adapting their assumptions using only the knowledge provided by the labeled data.

3 Method

We will introduce the new loss function in §3.6 after having established some notation and discussed the issue of discontinuity of the derivative in PL Lee et al. (2013).

3.1 Notation

Let $\mathcal{D} \subset \mathbb{R}^d$ be a dataset, split in $\mathcal{L}$, the annotated part, and $\mathcal{U}$, the non-annotated one, so that $\mathcal{L} \cap \mathcal{U} = \emptyset$ and $\mathcal{L} \cup \mathcal{U} = \mathcal{D}$. We consider a classification problem with n classes, $n \geq 2$, and $\mathcal{Y} \in \mathcal{L} \times \{0,1\}^n$ the one-hot labels for images in $\mathcal{L}$. The test-set is denoted by $\mathcal{T}$. We also define the function GT assigning to the image $x \in \mathbb{R}^d$ its ground-truth label $\mathrm{GT}(x)$. This function is defined on $\mathcal{L} \cup \mathcal{U} \cup \mathcal{T}$ but accessible only on $\mathcal{L}$ for training and on $\mathcal{T}$ for inference. We therefore have that for every $(u, l) \in \mathcal{Y}$, $\mathrm{GT}(u) = l$, but for $x \in \mathcal{L}$, $\mathrm{GT}(x)$ is inaccessible during training.

Throughout the article, we consider f_θ to be a family of models with softmax output, parametrized by $\theta \in \mathbb{R}^m$, with a fixed architecture. The goal of SSL is to use $\mathcal{L}$, $\mathcal{Y}$ and $\mathcal{U}$ so as to obtain a value $\theta_0 \in \mathbb{R}^m$ for which the accuracy of f_{θ_0} on $\mathcal{T}$ is maximized.

Under this definition, FlexMatch Zhang et al. (2021) is not an SSL algorithm as it makes a hypothesis on the values of $\mathrm{GT}(\mathcal{U})$, namely on the distribution of labels

$$\mathbb{E}_{u\in\mathcal{U}}(\mathbb{1}(\arg\max(\mathrm{GT}(u)) = i)), i \in [0, \cdots, n-1]. \tag{1}$$

In the FlexMatch article this distribution is supposed to be uniform and equal to $\frac{1}{n}$, and in our experiment section, §4 we establish that FlexMatch depends crucially on this assumption.

Finally, we will denote by CE the Cross-Entropy loss function, defined by

$$\mathrm{CE}(x, y) = -\sum_{y_i=1} \log x_i, \tag{2}$$

where $x = (x_i)$ and $y = (y_i)$ are both vectors in $\mathbb{R}^n$. The vector x satisfies $x_i \geq 0$ and $\sum x_i = 1$, and y is a one-hot label vector.

3.2 Pseudo-Labeling

Let $\tau > \frac{1}{2}$. PL, as introduced in Lee et al. (2013), ignoring batching and regularization, consists in minimizing

$$L_{PL}(\theta; \theta, \tau, \mathcal{L}, \mathcal{U}) = \mathbb{E}_{x\in\mathcal{L}}[\mathrm{CE}(f_\theta(x), \mathrm{GT}(x))] - \lambda_u \mathbb{E}_{x\in\mathcal{U}}[\mathbb{1}(f_\theta(x) > \tau) \log(f_\theta(x))] \tag{3}$$

with respect to θ for $x \in D$. The hyperparameter $\lambda_u > 0$ is the weight of the loss on unlabeled images. The hyperparameter $\tau \in (0.5, 1]$ is the threshold for accepting a pseudo-label for any given image. The first factor is the supervised loss on labeled data, and the second one the self-supervised loss on the unlabeled data.

The intuition behind the method is that, if $\mathcal{L}$ is large enough and if τ is close enough to 1., then

$$\frac{1}{\#\mathcal{U}} \sum_{u\in\mathcal{U}} \mathbb{1}(\arg\max(f_\theta(u) > \tau) = \arg\max \mathrm{GT}(u)) \approx 1 \tag{4}$$

i.e. the coverage of the unlabeled set by pseudo-labeled images and the purity of pseudo-labels will be high, so that the network will extrapolate sufficiently and correctly from the labeled dataset and will eventually learn correctly the whole dataset.

The derivative of L_{PL} is discontinuous at all images x and parameters θ such that $\max f_\theta(x) = \tau$, see fig. 1. L_{PL} is, thus, in $C^0 \setminus C^1$.

3.3 A remark on the dependence of L_{PL} on θ

The L_{PL} loss function, as well as the rest of the loss functions defined in this paper and, in general, all loss functions based on pseudo-labeling, depend on θ, the parameters of the network, in two ways. The parameters of the network enter the loss function as differentiated variables in

$$\Theta \mapsto \mathbb{E}_{x\in\mathcal{L}}[\mathrm{CE}(f_\Theta(x), \mathrm{GT}(x))] \tag{5}$$

and in

$$\Theta \mapsto \mathbb{E}[\mathbb{1}(f_\theta(x) > \tau) \log(f_\Theta(x))], \tag{6}$$

where θ in eq. equation 6 is considered as a non-differentiated parameter. This accounts for the first occurrence of θ in the signature of $L_{PL}(\theta; \theta, \tau, \mathcal{L}, \mathcal{U})$. The parameters θ also enter the loss as non-differentiated parameters in the mapping

$$\Theta \mapsto \mathbb{E}[\mathbb{1}(f_\Theta(x) > \tau) \log(f_\theta(x))] \tag{7}$$

determining the cut-off of pseudo-labels. This accounts for the second occurrence of θ in the signature of L_{PL}.

The double occurrence has an important consequence. Since training a neural network consists in the iterative minimization of a loss function, the loss function L_{PL} at step i depends on the value of θ at step

$i-1$ on the side of non-differentiated parameters: at step i, the parameters θ_i of the network are updated following

$$\theta_{i+1} = \theta_i - \eta \nabla L_{PL}(\theta_i; \theta_i, \tau, \mathcal{L}, \mathcal{U}), \tag{8}$$

However, the same relation holds at step i, namely

$$\theta_i = \theta_{i-1} - \eta \nabla L_{PL}(\theta_{i-1}; \theta_{i-1}, \tau, \mathcal{L}, \mathcal{U}). \tag{9}$$

Since the second occurrence of θ_i in eq. equation 8 is not differentiated and the dependence of L_{PL} on this occurrence is non-trivial, we can substitute this second occurence of θ_i by its value given by eq. equation 9 and obtain

$$\theta_{i+1} = \theta_i - \eta \nabla L_{PL}(\theta_i; \theta_{i-1}, \tau, \mathcal{L}, \mathcal{U}), \tag{10}$$

where we have kept the same notation for the loss function L_{PL} by a slight abuse of notation in order to keep things simple.

This relation is made possible by the second occurrence of the parameter θ in eq. equation 3, and the iterative nature of gradient descent. This results in the equations equation 8 and equation 9 giving the same value when differentiated, while they depend on different quantities.

In this derivation we have ignored regularization and momentum (for the details related to momentum, see §4.3.4). The loss function is therefore defined in an iterative manner, as θ_i itself is obtained by

$$\theta_i = \theta_{i-1} - \eta \nabla L_{PL}(\theta_{i-1}; \theta_{i-2}, \tau, \mathcal{L}, \mathcal{U}), \tag{11}$$

depending on θ_{i-2}, and it's turtles all the way down (fortunately, down only to zero). This fact introduces a very delicate dependence of θ_i on all previous $\theta_{i-1}, \cdots, \theta_0$ with discontinuities accumulating all the way.

Consequently, any discontinuities in the derivative accumulate as i grows, rendering the algorithm very sensitive to the inherent stochasticity of training (initialization, batch order and composition).

This complication is inherited by all applications of PL known to the authors, and, as will be established in the experiment section, poses serious instability issues in the limit of weak supervision and high performance. Regarding our signal-to-ratio terminology, see §1, we can now formally distinguish between the signal, given by the supervised loss, independent of θ, and the self-supervised loss, depending intractably on θ_0 as the number of iterations i grows. This intractable dependence is inherent in the method, but the accumulation of discontinuities in the derivative and their extremely delicate mitigation by EMA along the way are not.

3.4 Smooth Pseudo-Labeling

Let us introduce the function $\Phi : [0,1] \to [0,1]$,

$$\Phi(\sigma; \tau) = \begin{cases} \frac{\sigma - \tau}{1 - \tau}, \tau \leq \sigma \leq 1 \\ 0, \quad 0 \leq \sigma \leq \tau, \end{cases} \tag{12}$$

or, in more traditional notation, $\Phi(\sigma; \tau) = \mathrm{ReLU}(\frac{\sigma - \tau}{1 - \tau})$. We also define the stop-gradient operator, $\mathrm{sg}(\cdot)$, whose arguments enter calculations as parameters and not as variables differentiated in backpropagation. Hence the function

$$L_{SPL}(\theta; \theta, \tau, \mathcal{L}, \mathcal{U}) = \mathbb{E}_{x \in \mathcal{L}}[\mathrm{CE}(f_\theta(x), \mathrm{GT}(x))] - \lambda_u \mathbb{E}_{x \in \mathcal{U}}[\Phi(\mathrm{sg}(f_\theta(x)); \tau) \log(f_\theta(x))] \tag{13}$$

has continuous derivative, since the factor of the loss increases continuously from 0 to 1 instead of having a jump at τ. Taking the derivative of the function $\sigma \mapsto \Phi(\mathrm{sg}(\sigma); \tau) \log(\sigma)$ considering $\mathrm{sg}(\sigma)$ as a parameter and integrating with respect to all instances of σ yields

$$\widetilde{SPL}(\sigma; \tau) = \begin{cases} \frac{1 - \tau + \tau \log(\tau)}{1 - \tau}, 0 \leq \sigma \leq \tau \\ \frac{1 - \sigma + \tau \log(\sigma)}{1 - \tau}, \tau \leq \sigma \leq 1, \end{cases} \tag{14}$$

where the constant of integration was chosen so as to resolve the non-essential discontinuity at $\sigma = \tau$. This is the function plotted in fig. 1(b). Hence, minimizing L_{SPL} in eq. 13 is equivalent (at the C^1 level) to minimizing

$$\tilde{L}_{SPL}(\theta; \theta, \tau, \mathcal{L}, \mathcal{U}) = \mathbb{E}_{x \in \mathcal{L}}[\mathrm{CE}(f_\theta(x), \mathrm{GT}(x))] - \lambda_u \mathbb{E}_{x \in \mathcal{U}}[\widetilde{SPL}(f_\theta(x); \tau)] \tag{15}$$

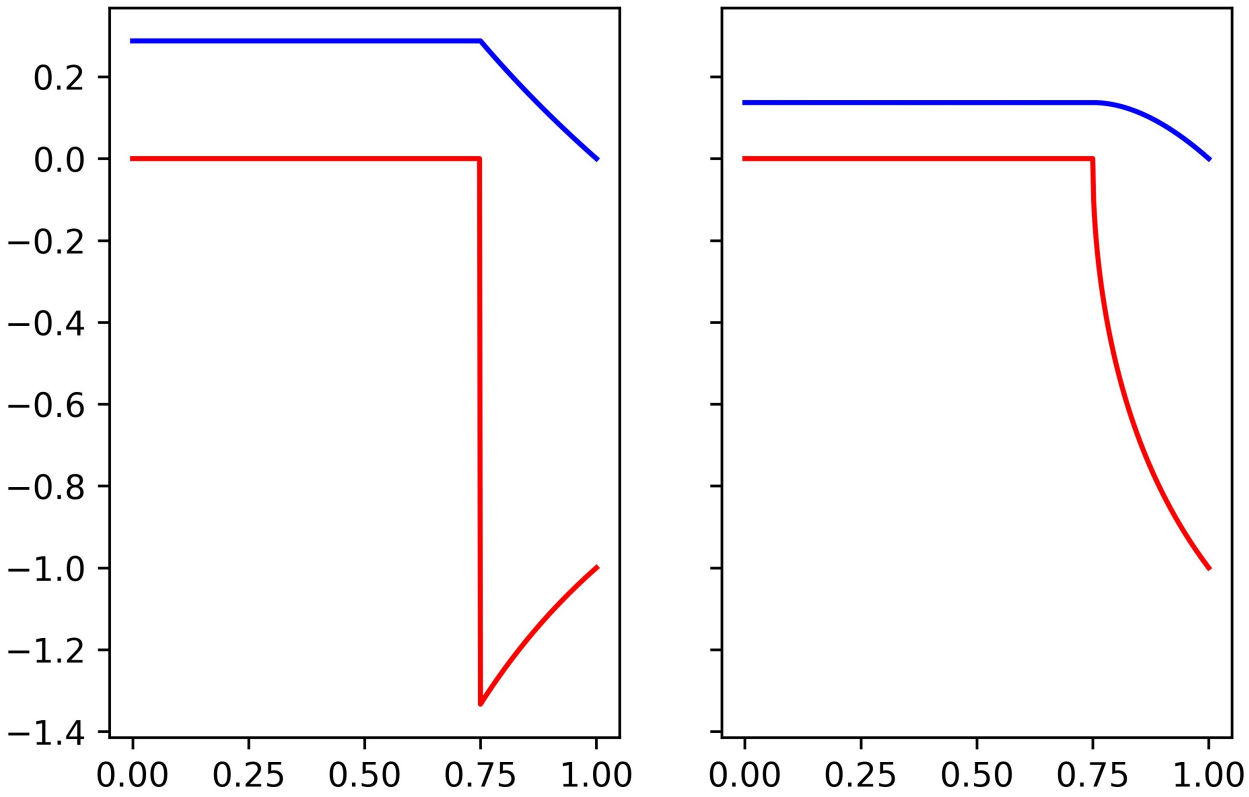

Figure 1: Loss functions minimized by PL (left) and SPL (right) in blue, and their derivatives in red, for $\tau = 0.75$.

3.5 The shape of the factor

The shape of the smoothing factor Φ need not be linear, but it needs to be increasing. Smoothness of the resulting loss function imposes that $\Phi(\tau;\tau) = 0$, and normalization that $\Phi(1;\tau) = 1$.

The linear model can be naturally embedded in a one parameter family $\mu \mapsto \Phi^\mu(\cdot;\cdot)$, where Φ is as in equation 12, $\mu \in [0, \infty]$, and the linear model corresponds to $\mu = 1$. As $\mu \to 0^+$, L_{SPL} degenerates to L_{PL} (though training with L_{SPL} does not degenerate to training with L_{PL} since $L_{SPL}(\mu)$ converges to L_{PL} only in a space of distributions). As $\mu \to \infty$, L_{SPL} degenerates to training on unlabeled images for which the score of the dominant class is already 1. This is *not* equivalent to training only on labeled images, as the derivative of the self-supervised loss function on such unlabeled images is equal to $-1 \neq 0$. Degeneracy is formal also in this limit.

In §4.3.1, we explore three important cases, the linear, $\mu = 1$; the quadratic $\mu = 2$, with an increasing convex shape function; and the square root, $\mu = 1/2$, with an increasing concave function. In any case, the function Φ represents the confidence of the network on the pseudo-label, and its shape the rate at which the factor of the self-supervised loss should increase as the score of the dominant class increases from τ to 1: slower than linear for the quadratic function, and faster than linear for the square root.

3.6 Smooth FixMatch

FixMatch consists in minimizing the loss function

$$L_{FM}(\theta;\theta,\tau,\mathcal{L},\mathcal{U}) = \mathbb{E}_{x\in\mathcal{L}}[\mathrm{CE}(f_\theta(x),\mathrm{GT}(x))] - \lambda_u \mathbb{E}_{x\in\mathcal{U}}[\mathbb{1}(f_\theta(x_w) > \tau)\log(f_\theta(x_s))], \tag{16}$$

where x_w is a weak augmentation of the image x, and x_s is a strong augmentation of the same image. This loss function presents discontinuities at all images x and parameters θ such that $\max f_\theta(x_w) = \tau$, just as in

PL. Replacing the loss function by

$$L_{SFM}(\theta;\theta,\tau,\mathcal{L},\mathcal{U}) = \mathbb{E}_{x\in\mathcal{L}}[\mathrm{CE}(f_\theta(x),\mathrm{GT}(x))] - \lambda_u \mathbb{E}_{x\in\mathcal{U}}[\Phi(\mathrm{sg}(f_\theta(x_w));\tau)\log(f_\theta(x_s))] \tag{17}$$

yields a smooth loss function, again just as in PL. The loss function depends on two parameters, making visualization less attractive, but the principle is the same as in PL.

In the experiment section we compare Smooth FixMatch with the original implementation and establish that in the limit of weak supervision and high performance the smooth version performs significantly better, while the same level of performance is maintained when supervision is stronger and/or performance is far from the fully supervised baseline. Concerning the shape of the factor Φ, we found that the linear function performs better overall, when compared with FixMatch with the same set of hyperparameters. Different hyperparameters gave good results for the quadratic and the square root factors, so their usefullness should not be excluded. Naturally, the quadratic factor performed better when the threshold was lowered, and the square root performed better with a higher threshold, see table 1 and fig. 2.

In FixMatch, the weight λ_u was set equal to 1 for all experiments. In order to keep results comparable, we kept the magnitude of the equilibrium value of the unsupervised loss of L_{SFM}, ℓ_{SFM}, equal to the one of L_{FM}, ℓ_{FM}, see §C.1 for the relevant calculation.

3.7 Learning the smoothness factor

It is tempting to add the factor

$$L_\Phi = -\lambda_\Phi \Phi(\mathrm{sg}(f_\theta(x_w));\tau)\Phi(f_\theta(x_w);\tau) \tag{18}$$

to the L_{SFM} loss function, so that the factor can be maximized, as it learns x_w, contrary to x_s learned in L_{SFM}. The smoothness of L_Φ is once again assured by the multiplication factor with sg.

FixMatch established that the network benefits more from learning the strong augmentation of the image than the weak one. In §C.2 of the supplementary material we present a calculation producing the value $\bar{\lambda}_\Phi$ beyond which the model is more incited to learn the weak augmentation than the strong one. A natural choice for λ_Φ is thus in the interval $[0, \bar{\lambda}_\Phi]$.

Our related ablation study (cf. §A.3.6) showed instabilities when the upper limit is reached. Surprisingly, even values of the order of 1% of $\bar{\lambda}_\Phi$ were detrimental to performance. We thus kept the default value $\lambda_\Phi = 0$ and did not add this loss to Smooth FixMatch.

4 Experiments

In the experiment section, we use FixMatch as baseline, a common practice in current literature in the subject, in order to test the gain of imposing the continuity of the derivative of the loss function without the additional complications of added modules and hyperparameters, which, to boot, worsen the problem of the discontinuity of the derivative of the loss.

We thus compare our method with the baseline, but also with FlexMatch, holding the SOTA performance in the benchmark CIFAR-10-40 which is of particular interest in our study, due to the high volatility of results. We establish the superiority of our method, and observe that FlexMatch presents the same kind of volatility as FixMatch in the standard benchmark, but, what is more important, is extremely sensitive with respect to the statistics of class distribution, contrary to our method and to the original application of FixMatch.

In all tables, we report the error rate of the last checkpoint. Sohn et al. (2020) reports the median over the last 20 checkpoints, while Zhang et al. (2021) reports best error rate, both not accessible in real conditions. In particular, FixMatch presented some sudden and significant drops in a few experiments, see §4.4, establishing that the best performance is not a good measure (and reminding why it is not used in the literature).

All tables with detailed results, as well as some additional ablation studies, are presented in the appendix §A, the structure of which is the same as the one of §4 of the article, for ease of cross-reference.

Our runs of FixMatch and FlexMatch use and adaptation of the codebase of Zhang et al. (2021). Apart from adding our implementation of Smooth FixMatch, most notably we ensured that batch composition is correctly controlled by the random seed, and we separated the random condition for the creation of the labeled datasets of the different folds from the one controlling training (initialization of the backbone and batch composition) in order to bring the evaluation protocol closer to real-life conditions, where the constitution of the labeled dataset is independent of training.

4.1 **CIFAR**-10-40

This benchmark consists in keeping 4 labeled images from each of the 10 classes of the dataset CIFAR-10, 40 in total. In order to check the compatibility of our results with both the original FixMatch paper, Sohn et al. (2020), and the FlexMatch paper, Zhang et al. (2021), we run 6 experiments in total, indexed $0-6$: $0-2$ for FlexMatch and $1-5$ for FixMatch. Our results for indexes $0-2$ agree with those of Zhang et al. (2021), and those indexed $1-5$ agree with those of Sohn et al. (2020). However, the additional experiments indexed $3-5$ give significantly worse results for FlexMatch than in Zhang et al. (2021), see tables 1 and 5. We note that, while the results for FixMatch and Smooth FixMatch were reproducible after some necessary modifications assuring control of batch composition by the random condition, we were unable to get reproducible results for FlexMatch.

For FixMatch we observe two basins of attraction for the accuracy, in accordance with Sohn et al. (2020): one low, $\sim 83-87\%$ (roughly 1 in 6 experiments, accounting for the lower mean and high std of FixMatch and for the very high std of FlexMatch) and one high, $> 90\%$. For FixMatch, convergence to either basin seems to be determined by both the labeled dataset as well as the random condition, as observed in the original paper.

Convergence to the low basin of attraction occurs when one out of the 10 classes of the dataset is totally or partially collapsed, i.e. when the corresponding column in the confusion matrix has 0 or small ($\lessapprox 0.20$) elements. This can happen either because of bad representativeness of labeled examples, unfavorable batch order and/or composition, unfavorable initialization or a combination of the above. The instabilities observed in FixMatch training are a manifestation of a delicate competition between the tendency of the model to be more expressive and not have null columns in its confusion matrix, and the L^2 regularization factor, which can overcome the tendency for expressiveness and totally or partially collapse one class when the drive for expressiveness, coming from supervision or CR, is overcome. The phenomenon of collapsed classes is stronger in the more complicated datasets (CIFAR-100 and Imagenet) where the confusion matrices are very sparse, which accounts to a considerable extent for the high error rates.

In practice, the collapse can be detected and ad-hoc solutions could be given such as pausing training and extending the labeled set, but this could also be problematic in the present state of the art, see our discussion in §4.5. A better understanding of the behavior of SSL algorithms seems to be necessary so that Active Learning can be based on more solid foundations.

Continuity with linear coefficient improves performance in the high basin of attraction and approaches the low basin to the high one. Continuity with square factor and low threshold improves performance in both basins, but the low basin remains in the same window. Likewise for square root and higher threshold. Shapes alternative to linear (square and square root) with the standard threshold switched experiments converging to the high basin for FixMatch to the low basin for Smooth FixMatch with the respective factor shape.

Some experiments kept learning until the very end of training, hinting that a higher learning rate or a different scheduler with less steep decline of the learning rate could be more adapted. However, since the fact that our method improves on FixMatch at a significant level is well-established without this eventual additional improvement, we favoured more important ablation studies instead.

4.2 Strong supervision regime

We verify that in the more stable benchmark of CIFAR-100 with 2500 labels, 25 per class, as well as in the Imagenet-10% benchmark ($100K$ labeled out of $\sim 1000K$ images) our improvement does not deteriorate performance when applied on FixMatch. The performance on PL-based methods in these benchmarks is far

	mean ± std	gain wrt FixMatch	p-value
FixMatch*	9.28 ±2.95		
Smooth FixMatch	**7.24** **±1.94**	**2.04** ±1.27	**0.015625**
Smooth FixMatch-sq†	7.71 ±2.98	1.57 **±0.91**	0.031250
Smooth FixMatch-sqrt‡	7.72 ±2.60	1.55 ±1.02	**0.015625**

Table 1: Error rate on 6 folds of CIFAR-10 with 40 labels of our runs of FixMatch without and with the smoothness factor. The p-value line corresponds to the result of the Wilcoxon one-sided test for the given method having lower error rate than FixMatch, which is used as baseline (median being significantly positive, lower is better). * Our runs. † Factor shape is quadratic, threshold set at 0.82. ‡ Factor shape is square root, threshold set at 0.98.

from the fully-supervised baseline, and supervision is considerably stronger. This combination results in a std $\sim$0.2% for FixMatch on CIFAR-100-2500 and $\sim$0.5% on Imagenet, an order of magnitude less than the one observed in CIFAR-10-40 experiments. The error rates are $\sim 30\%$ and $\sim 45\%$, respectively, quite far from full supervision.

The instabilities caused by the discontinuity of the derivative are thus not significant in this regime, and we only need establish that our method does not harm performance outside the regime where it is supposed to improve it. The relevant results are presented in the appendix for completeness.

4.3 Ablation studies

4.3.1 Shape of continuity factor

We tried shapes for the continuity factor alternative to the simplest linear one, two natural choices being the square root and the square of the linear continuity factor.

The square root attributes higher weights for scores closer to the threshold but introduces a rather harsh discontinuity in the second derivative of the loss. Experiments with the standard configuration of FixMatch gave a high error rate on seed 5 of CIFAR-10-40, ~ 14.00 against 8.01 for FixMatch, while on other seeds it improved on FixMatch. Increasing the threshold to 0.98 from 0.95 was found to consistently improve on FixMatch, see table 1, confirming our intuition. The threshold for more realistic datasets like Imagenet is lower, 0.70 by default for FixMatch, which relaxes the sensitivity of this configuration with respect to the value of the threshold, even though it remains more sensitive than the linear factor.

The square of the linear factor increases slower than the linear one, and thus attributes smaller weights to the loss when the score is close to the threshold. It also results in the second derivative of the loss being continuous. This shape performed worse than FixMatch on seed 2 with standard threshold, giving an error rate of 12.04 against 7.48. However, lowering the threshold to 0.82 gave the same significance level for improving on FixMatch as the linear factor with the standard threshold, only with a smaller run-wise gain, see table 1.

Obtaining a better performance when lowering the threshold for the square and when increasing it for the square root is easily found to be intuitive.

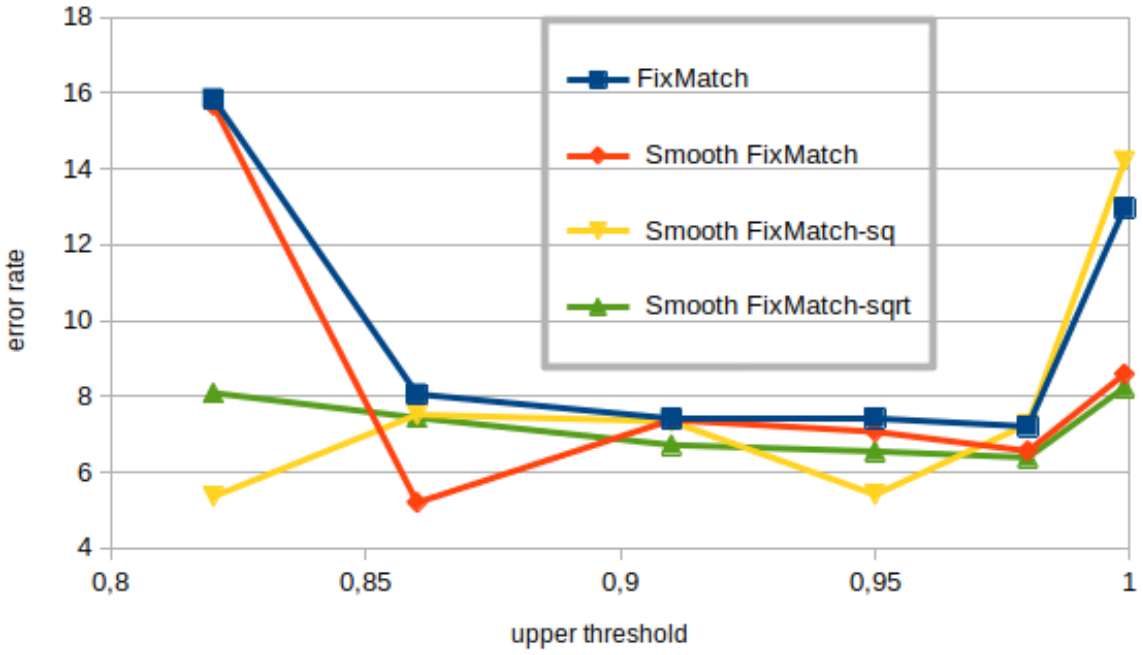

Figure 2: Error rate on the 1st fold of CIFAR-10-40 when the threshold varies. The smooth versions of FixMatch with all three shape factors (linear, quadratic, square root, reliably outperform the baseline.

4.3.2 Threshold

We vary the threshold for accepting pseudo-labels on the first fold of CIFAR-10-40, and report results in figure 2. FixMatch is found to depend in a much more sensitive way with respect to the value of the threshold τ for accepting a pseudo-label than the smooth version with linear factor. The quadratic factor gives better results for lower values of τ, and the square-root better for higher ones, confirming intuition. We provide an indicative plot of linear and quadratic factors in the appendix.

4.3.3 Class imbalance

We introduce a non-alignment between the class distribution of the labeled and unlabeled datasets. For each fold, one class is chosen randomly and its unlabeled dataset is reduced to 60% of the rest of the classes by dropping randomly 40% of the unlabeled images in that class. This is done in order to establish the sensitivity of FlexMatch with respect to the uniformity of class distribution, which shows clearly in our results, reported in table 2.

We see that even in this mild scenario FlexMatch is found to perform worse than FixMatch at a < 0.05 significance level, while Smooth FixMatch is found to perform better at a ~ 0.11 significance level. FixMatch and Smooth FixMatch do not see their performance drop as much as FlexMatch. This ablation confirms our claim that FlexMatch is not an SSL algorithm on the same standing as others, and, indeed is not an SSL algorithm according to our definition.

4.3.4 SGD momentum parameter

In FixMatch, a rapid degradation of performance was observed in their experiments on CIFAR-10 with 250 labels (and therefore considerably stronger supervision signal) when the momentum parameter of the SGD optimizer is increased beyond the default value $\beta = 0.90$.

Momentum is in general beneficial to the stability of convergence because it introduces an exponential moving average (EMA) smoothing of the gradients and reduces noise in the direction of gradient descent. Since the

	FixMatch	FlexMatch	Smooth FixMatch
	9.30 ±3.43	11.59 ±5.90	**7.78** ±**2.71**
gain wrt FixMatch		−2.28 ±3.05	**1.51** ±**1.94**
p-value		0.953125	**0.109375**
drop due to imbalance	**0.03** ±1.83	4.80 ±7.15	0.55 ±**0.89**
p-value	**0.421875**	0.109375	0.15625

Table 2: Ablation study on class imbalance. Error rates with one class reduced to 60%. FlexMatch is very sensitive on the class distribution in the unlabelled dataset, while FixMatch and SmoothFixMatch are significantly more robust. Drop due to imbalance is the difference with respect to the performance of the same model without class imbalance (smaller is better). The p-value of drop due to imbalance is the significance level at which the method is affected by the introduction of imbalance (bigger is better).

derivative of the loss of FixMatch is discontinuous and, as we have observed, the loss is actually defined in an iterative manner due to its dependence on the state of the model at the each step, momentum in this case results in an accumulation of discontinuities, which becomes more significant as the value of β increases. On the other hand, these discontinuities are smoothed out to a certain extent by EMA.

Imposing smoothness on the derivative of the loss function removes the additional stochasticity due to discontinuities, but does not alter the fact that the loss is defined in an iterative manner, and thus inherently highly stochastic.

The trade-off between accumulation of discontinuities, which adds stochasticity in the gradient at a given step, and the benefits of EMA smoothing is far too complicated to analyze in a realistic setting. It should, nonetheless, be expected to introduce a very sensitive dependence on β in the limit of weak supervision and high performance when the loss is not C^1-smooth.

It is not surprising to observe that increasing β beyond a certain tipping point becomes detrimental to performance, and, to boot, in a quite catastrophic manner. This is observed for both algorithms, but seems to harm ever so slightly more the smooth version, which suffers significantly less from randomness in the value and direction of the gradient. This tipping point manifests itself at the value where the benefits of smoothing the noise in the direction of gradient descent via EMA are overpowered by the accumulation of discontinuities (if present) and the long memory of the gradient from iterations where pseudo-labels were less abundant and/or reliable than at the given step. Naturally, the value of the tipping point depends on the strength of supervision signal and the particularities of the dataset in a convoluted way, making it impossible to determine theoretically, but only accessible empirically.

We verify this trade-off hypothesis by repeating the experiment of Sohn et al. (2020) in the more difficult setting of CIFAR-10-40 (instead of 250), comparing the behavior of FixMatch without and with the smoothness factor, and report results in figure 3. The default parameter for the rest of our experiments is $\beta = 0.90$.

What happens when momentum is reduced, adding stochasticity to gradient descent is more interesting. Already for $\beta = 0.85$, instead of the default value $\beta = 0.90$, FixMatch presents instabilities with the model spiking to an error rate $\sim 25.00\%$ once at $\sim 40\%$ of iterations, but then regressing to $\sim 30.00\%$ and never recovering, practically doubling its error rate for a $\sim 5\%$ decrease in β. The smooth version also saw its performance drop, but by $\sim 30\%$, as opposed to $\sim 90\%$ for FixMatch.

This confirms experimentally the delicate nature of the stochasticity of the loss and its derivative both in the the non-smooth and the smooth setting. At the same time it proves beyond any doubt that non-C^1-smooth loss functions are highly problematic in the limit of weak supervision and high performance.

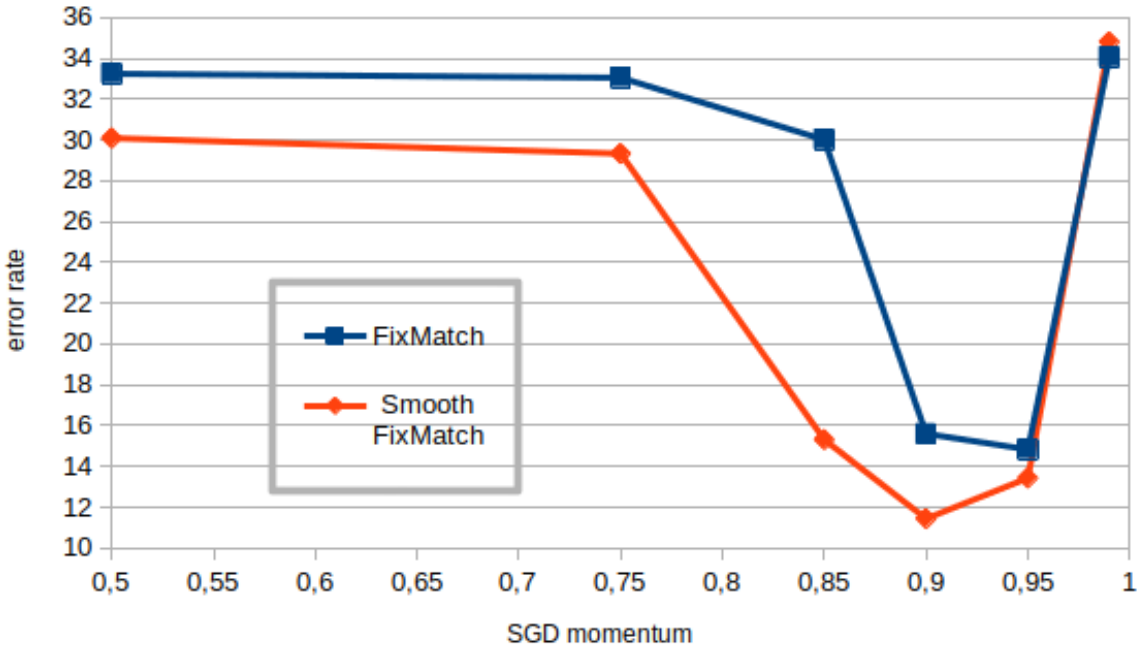

Figure 3: Error rate on the 4th fold of CIFAR-10-40 when SDG momentum varies. The window allowing a good performance for FixMatch is very narrow, but becomes significantly wider upon introduction of the smooth loss function.

4.4 Aggregate comparison and some comments

When all our experiments on CIFAR-10-40 on FixMatch versus Smooth FixMatch with linear continuity factor performed under the same conditions are compared run for run, including ablations, Smooth FixMatch gains 2.61 on average on FixMatch, with a standard deviation of 3.28. The p-value for Smooth FixMatch performing better than FixMatch is 4.35e−5. The maximum gain is 14.70, the minimum is −1.30. Smooth FixMatch obtained a better performance in 23 out of 26 experiments, including for reasons of transparency experiments where both performed poorly.

The experiment with FixMatch on CIFAR-10-40 with the threshold set at 0.82, see fig. 2, had a best error rate of 6.46 attained at $\sim$90% of iterations, but presented an instability and dropped to 15.85%. Similarly, on the same dataset with SGD momentum $\beta = 0.85$, FixMatch spiked to an error rate $\sim$25.00% once at $\sim$40% of iterations, but then regressed to $\sim$ 30.00% and never recovered. The $\sim$ 30.00% error rate corresponds to 3 collapsed classes with one column summing up to $\sim$0.37%, another up to $\sim$0.20% and a third up to practically 0.00.

Such instabilities were observed only once in our experiments where the continuity factor has been integrated into the unsupervised loss, and the discrepancy between the best error rate and the one of the last checkpoint were insignificant. The only exception was a discrepancy of $\sim$1.5% between best and final error rates for the square root factor and $\tau = 0.82$, which again is in agreement with our intuition. In a considerable portion of our experiments, actually, best error rate was achieved late in training, hinting that the learning rate and scheduling are not optimal for our method, but investigating this direction as well would lead us astray from the main argument of the paper.

In the standard CIFAR-10-40 benchmark, FlexMatch is found to gain 2.10 on FixMatch on average, run for run, a greater average improvement than our 2.04. The standard deviation of FlexMatch's gains on FixMatch is $\sim$5.7% while ours $\sim$1.3%. This however still hides the fact that our method is better than FixMatch in every single experiment of table 1, while FlexMatch only in 5 out of 6 (cf. table 5). Such intricacies made necessary the introduction of the Wilcoxon test, which is a more adapted measure of comparison.

4.5 Random choice of labeled dataset

4.5.1 Presentation of the benchmark

In order to contrast with current practices of hardcoding properties of the benchmark datasets into the learning procedure of proposed methods, as we discussed in our Related Work section, §2, we introduce the following benchmarks which bring evaluation protocols closer to real-world practice.

We consider a dataset, say CIFAR-10, and fix the number of labeled images, say 40. Contrary to the widely studied benchmark CIFAR-10-40, we *do not* chose 4 labels for each class, imitating the uniform class distribution of CIFAR-10, but we randomly choose 40 images to label, uniformly from the whole dataset, and fix 6 such folds, numbered $0-5$. This brings the evaluation protocol closer to real-life scenarios, as in practice class distribution is unknown and can only be extrapolated from that of the labeled dataset, subject to an uncertainty that grows as the number of labels becomes smaller.

Indicatively, the class distribution of the 0-th fold features 1 class with frequency 2.5%, 3 classes with frequency 5%, 2 classes with frequency 10%, 3 classes with frequency 12.5% and 1 class with frequency 25%, which is considerably far from the real uniform distribution of 10% for each class. Such phenomena should not be expected to be rare in the regime of weak supervision, provided that the sampling of labeled data is truly random and has not artificial similarities with the whole dataset, imposed by construction.

We note that the 3rd fold has two classes with no labeled image associated to them. Concretely, we think that since the probability of the existence of a class with 0 frequency in the labeled dataset is not negligible, the benchmark with 40 images labeled out of $50K$ is not relevant. We decided to keep the fold in the benchmark nonetheless and train a 10-class classifier on it in order to showcase the difficulties of the regime, technical, conceptual and protocol-related, and for reasons of comparison with the standard benchmark with uniform class distribution imposed on the labeled dataset.

A single fold with one class with 0 labeled representatives persists in the 50 label setting, but all classes are represented in all folds starting from the 60 label setting. Class distribution remains quite far from the uniform 10% for all classes, however, as can be seen in fig 4. For each fold of for each benchmark of 40 up to 150 labels, we calculate the standard deviation of class frequencies around the mean frequency 1/10, and we normalize by 1/10, expressing the standard devation as a fraction of the mean value. The plot features the mean of this normalized standard deviation over the 6 folds of each benchmark as well as that of 3 folds for which the models had difficulty converging. It starts at $\sim 50\%$ of the mean value for 40 labels and decreases slowly to $\sim 25\%$ for 140 labels.

Methods using assumptions on class distribution tested on this benchmark, which can be carried over to the imbalanced class setting, can only use the observed class distribution of each labeled fold, and not the uniformity of overall class distribution in our case (or the actual class distribution in the general case).

Extrapolation of the class distribution from a few labeled data is easily seen to be completely unjustifiable in an empirical way. A rigorous calculation of confidence intervals of class distribution should have made part of any method based on such an extrapolation, along with a rigorous sensitivity analysis of the effect of misestimation.

The goal of the benchmark is to reach stable performance equal to the fully supervised baseline with the minimal number of labels using only the properties of the each labeled dataset for each run. This protocol imitates an Active Learning scenario for which each time labels are randomly selected and added to the labeled dataset the model is trained from scratch.

The naive expectation for SSL models would be that performance should not worsen when labels are added in this incremental way. However, we observe that CR based on PL behaves badly in this scenario. Smoothing out the discontinuities in the derivative does improve the behavior of the problem, since the smooth model obtains a stable performance close to the fully supervized baseline before the non-smooth one. Smoothness of the loss function nevertheless is not the root of the problem, as the smooth version still presents drops in performance when labels are added.

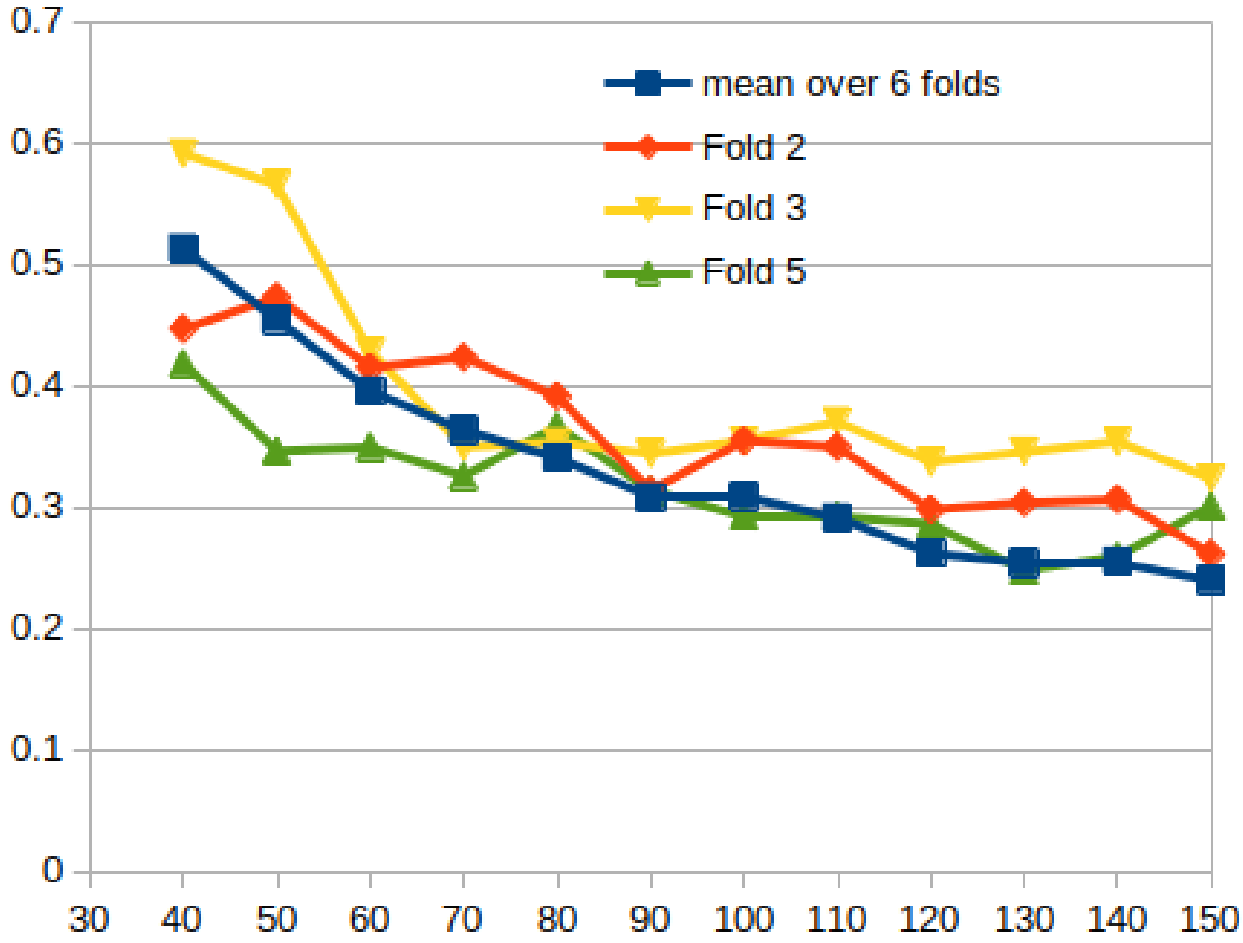

Figure 4: Standard deviation of class frequency from uniform class distribution of CIFAR-10 with random sampling normalized by the uniform frequency 1/10, over 6 folds. The graph shows that with truly random sampling the class distribution of the labeled dataset remains quite far from the true (uniform) class distribution of the whole dataset. We also include the evolution of the standard deviation of 3 folds that proved difficult for the model to converge. In particular, the difficulty of fold 5 shows that standard deviation is a crude measure, and that representativeness of images does play a role in convergence.

This observation raises the question as to how the model will react if labels are added and training is resumed. Will adding labeled data result in the model correcting its confirmation bias or will it ignore the new labels and stick to its predictions? If our benchmark allows for a prediction, it should be that the smooth version would be faster to correct its biases than the non-smooth one, but both would react in a non-monotonic way.

4.5.2 Results

We test Smooth FixMatch against the baseline FixMatch with a progressively increasing number of labels by 10 at each step until one algorithm reaches a stable performance close to the supervised baseline across the 6 folds. The goal is to reach stable performance equal to the fully supervised baseline with the minimal number of labels. Results are presented in table 3.

In the regime where the performance of both models is far from the fully supervised baseline, the performances of the models seem to be comparable. In addition, both models may regress to worse performance upon increase of labeled images. However, Smooth FixMatch reaches the point of diminishing returns earlier than FixMatch. The smooth model obtains a stable mean error rate under 8.00% and a standard deviation smaller than 1% with 120 labels and stays in that regime up to 150 labels. However, FixMatch can regress to a

	CIFAR-10 with random sampling											
labels	40	50	60	70	80	90	100	110	120	130	140	150
FixMatch*	16.34 ±8.23	14.83 ±9.23	11.54 ±5.02	8.02 ±3.71	10.15 ±4.73	9.26 ±3.93	9.56 ±4.27	7.49 ±2.94	6.01 ±0.70	6.69 ±2.26	5.64 ±0.64	5.63 ±0.43
Smooth FixMatch	17.51 ±7.21	15.37 ±10.06	9.09 ±3.99	7.67 ±3.21	8.08 ±3.76	8.91 ±4.84	8.40 ±4.03	8.35 ±4.02	5.96 ±0.78	5.35 ±0.26	6.12 ±0.88	5.61 ±0.48
gain wrt FixMatch	−1.17 ±8.07	−0.54 ±6.28	2.46 ±3.47	0.36 ±5.68	2.07 ±4.23	0.36 ±3.00	1.14 ±3.14	−0.85 ±4.59	0.05 ±0.20	1.34 ±2.00	−0.47 ±1.08	0.02 ±0.71
p-value	0.5781	0.4219	0.2188	0.5	0.2813	0.5	0.2813	0.7813	0.3422	0.0157	0.7188	0.5

Table 3: Error rate on 6 folds of CIFAR-10 with varying number of randomly selected labels of our runs of FixMatch without and with the linear smoothness factor. *Our runs.

worse performance even in the passage from 120 to 130 labels, and, then reach a competitive error rate at 140 labels, which is than that of the smooth version for the same number of labels (but worse than that of Smooth FixMatch with 130 labels).

As show the results of table 3 and, in more detail, those of the Excel sheet accessible here, the response to the expansion of the labeled dataset of both FixMatch and Smooth FixMatch, even if to a slightly lesser

	Effect of adding 10 labels in CIFAR-10 with random sampling												
labels	40 ↑ 50	50 ↑ 60	60 ↑ 70	70 ↑ 80	80 ↑ 90	90 ↑ 100	100 ↑ 110	110 ↑ 120	120 ↑ 130	130 ↑ 140	140 ↑ 150		
FixMatch*	1.54 ±3.37	3.29 ±9.45	3.52 ±3.66	−2.12 ±4.41	0.88 ±1.28	−0.30 ±1.78	2.07 ±2.93	1.48 ±3.18	−0.67 ±2.41	1.04 ±2.50	0.02 ±0.85		
p-value	0.15625	0.65625	0.03125	0.71875	0.15625	0.57813	0.03125	0.5	0.42188	0.28125	0.5		
Smooth FixMatch	2.14 ±10.27	6.29 ±9.83	1.42 ±5.49	−0.41 ±1.42	−0.83 ±3.27	0.51 ±1.50	0.06 ±4.05	2.39 ±3.59	0.61 ±0.82	−0.77 ±0.96	0.50 ±0.92		
p-value	0.34375	0.42188	0.21875	0.78125	0.34375	0.25009	0.28125	0.34375	0.07813	0.93099	0.34375		

Table 4: Mean gain and p-value for the improvement on the error rate on 6 folds of CIFAR-10 with 10 labels added incrementally. *Our runs.

extent for the latter, is not the expected one. The p-value for the algorithm gaining in performance after labeling $10 \times n$ images (cf. the Excel sheet accessible here for more detailed results) is not consistently low. We indicatively present in fig. 5 the evolution of the error rate on the 5th fold of the benchmark, the one

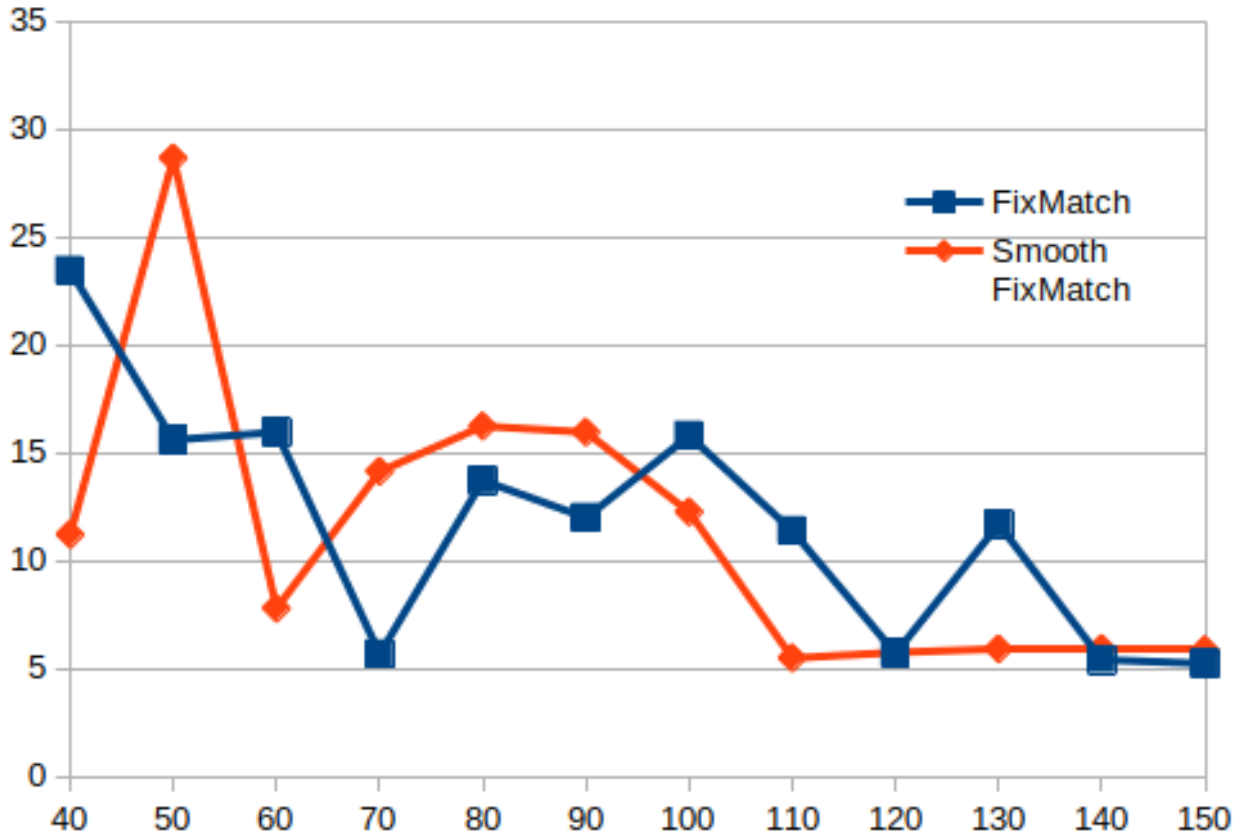

Figure 5: Evolution of the error rate on the 5th fold of CIFAR-10 with random sampling. Both models can regress to higher error rates when the number of labeled images is increased, but this occurs for greater number of labels for FixMatch than for Smooth FixMatch.

that proved to be the most difficult for the models to solve. We remind the reader that the benchmarks with 40 and 50 labels are in fact irrelevant (as some classes are not represented in the labeled set) but are included only for comparison with the standard benchmark.

We assert that a good response to labeling images should be a built-in feature of SSL algorithms. More precisely, consider two splits

$$\mathcal{L} \cup \mathcal{U} = \mathcal{D} \text{ and } \mathcal{L}' \cup \mathcal{U}' = \mathcal{D} \tag{19}$$

of the same dataset $\mathcal{D}$ with $\mathcal{L} \subsetneq \mathcal{L}'$ (and consequently $\mathcal{U}' \subsetneq \mathcal{U}$), cf. §3.1 for notation. All other parameters being equal, the performance on the test-set of a model trained on the split $\mathcal{L}' \cup \mathcal{U}'$ should be expected to be worse than the performance of the same model trained on $\mathcal{L} \cup \mathcal{U}$ only with a small probability. This is a notion of "probabilistic monotonicity" of the training strategy with respect to the labeled dataset.

We were unable to find even a heuristic as to why consistency should satisfy such a natural property. This does is not directly related to the discontinuity of the loss function, even if the smooth version seems to respond better when the labeled dataset is expanded.

Table 4 shows that the smooth version of the model is more likely to respond correctly to the addition of labeled examples. The p-value for decreasing the error rate is above 0.5 only once for the smooth version, at the passage $70 \to 80$ labels. Its is above 0.5 for the non-smooth version 3 times, the last one being observed at the passage $90 \to 100$ labels, and remains high even at the passage $120 \to 130$ labels, while the one of Smooth FixMatch is already < 0.08 but deteriorates again at $130 \to 140$, even though it remains $< 10\%$.

We can remark, however, that both methods, already with 60 labels have enough information to classify more or less correctly 9 out of 10 classes of CIFAR-10. The difference between a model converging to an

accuracy $\geq 92\%$ and one converging to an accuracy $\leq 88\%$ can be read in the confusion matrix: low accuracy is obtained when one class is totally collapsed, while high accuracy when all diagonal entries are significantly > 0.

Already in the 80-label benchmark, Smooth FixMatch displays two folds that manage to populate all 10 diagonal elements of the confusion matrix, while FixMatch populates only 9/10, leading to an accuracy smaller than 90%.

5 Conclusions

We imposed smoothness of the loss function in PL as it is applied in CR. Our improvement is generic, entails no additional hyperparameters, marginal computational cost and uses no assumption on label distribution, making it more robust than the SOTA when mild misalignment between label distribution between annotated and non-annotated data is introduced. When applied on FixMatch, our method significantly improves the performance in the regime where very weak supervision can achieve performance closer to the fully supervised baseline.

The factor imposing smoothness on the loss function can be interpreted as a measure of the confidence of the network producing the pseudo-labels on the pseudo-labels, ranging from 0 to 1 in a continuous fashion.

Our improvement becomes significant in the limit of weak supervision and performance close to the fully supervised baseline. In the present state of the art, it is therefore significant only in simple datasets, since for datasets close to real-life applications our techniques either demand strong supervision (which overpowers the instabilities caused by the discontinuities in the derivative of the loss), or their performance stays far from the fully supervised baseline, where the instabilities do not manifest themselves and volatility of the performance remains low.

The issue of instabilities should not be expected to be fixed by itself when moving on to more difficult datasets. If anything, instabilities should be expected to be more important when the limit of weak supervision and high performance is reached. This work is therefore a word of caution for the direction that SSL should take, in order to anticipate this kind of difficulty when the limit is reached.

Replacement of a step-wise function by a ReLU factor wherever thresholding is involved could lead to improving performance in other contexts.

References

Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2019a.

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, 2019b.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.

Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems*, 34:3965–3977, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Jean-Bastien Grill, Florian Strub, Florent Altché, C. Tallec, Pierre H. Richemond, Elena Buchatskaya, C. Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, B. Piot, K. Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.

HM Kabir, Moloud Abdar, Seyed Mohammad Jafar Jalali, Abbas Khosravi, Amir F Atiya, Saeid Nahavandi, and Dipti Srinivasan. Spinalnet: Deep neural network with gradual input. *arXiv preprint arXiv:2007.03347*, 2020.

Byoungjip Kim, Jinho Choo, Yeong-Dae Kwon, Seongho Joe, Seungjai Min, and Youngjune Gwon. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*, 2021.

Jaehyung Kim, Youngbum Hur, Sejun Park, Eunho Yang, Sung Ju Hwang, and Jinwoo Shin. Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. *Advances in neural information processing systems*, 33:14567–14579, 2020.

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

Zhengfeng Lai, Chao Wang, Henrry Gunawan, Sen-Ching S Cheung, and Chen-Nee Chuah. Smoothed adaptive weighting for imbalanced semi-supervised learning: Improve reliability against unknown distribution data. In *International Conference on Machine Learning*, pp. 11828–11843. PMLR, 2022.

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 3-2, pp. 896, 2013.

Doyup Lee, Sungwoong Kim, Ildoo Kim, Yeongjae Cheon, Minsu Cho, and Wook-Shin Han. Contrastive regularization for semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3911–3920, 2022.

Claude Lemaréchal. Chapter vii nondifferentiable optimization. In *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, pp. 529–572. Elsevier, 1989. doi: https://doi.org/10.1016/S0927-0507(89)01008-X. URL `https://www.sciencedirect.com/science/article/pii/S092705078901008X`.

Junnan Li, Caiming Xiong, and Steven Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. *arXiv preprint arXiv:2011.11183*, 2020.

T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2020. doi: 10.1109/TPAMI.2018.2858826.

Hieu Pham, Zihang Dai, Qizhe Xie, Minh-Thang Luong, and Quoc V. Le. Meta pseudo labels. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. URL `https://arxiv.org/abs/2003.10580`.

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.

Chen Wei, Kihyuk Sohn, Clayton Mellina, Alan Yuille, and Fan Yang. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10857–10866, 2021.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. URL http://www.jstor.org/stable/3001968.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33, 2020.

Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. Dash: Semi-supervised learning with dynamic thresholding. In *International Conference on Machine Learning*, pp. 11525–11536. PMLR, 2021.

Fan Yang, Kai Wu, Shuyi Zhang, Guannan Jiang, Yong Liu, Feng Zheng, Wei Zhang, Chengjie Wang, and Long Zeng. Class-aware contrastive semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14421–14430, 2022.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34, 2021.

Wenqiao Zhang, Lei Zhu, James Hallinan, Shengyu Zhang, Andrew Makmur, Qingpeng Cai, and Beng Chin Ooi. Boostmis: Boosting medical image semi-supervised learning with adaptive pseudo labeling and informative active annotation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20666–20676, 2022.

A Detailed results of experiments

We provide the full tables of all experiments reported in the paper. The title of each paragraph is the same as the one in the paper with reference to the corresponding table.

In tables corresponding to ablation studies, default parameters are in bold.

All experiments were run using the codebase of Zhang et al. (2021).

A.1 **CIFAR**-10-40

Detailed results reported in table 5.

A.2 Strong supervision regime

Detailed results on CIFAR-100-2500 reported in table 6. The results confirm that, outside the regime of weak supervision coupled with high performance, the smoothing factor has no significant influence on the performance of the model.

For experiments on Imagenet10%, cf. §B.2.

A.3 Ablation studies

A.3.1 Shape of continuity factor

Figure 6 compares the linear with the quadratic factor. Detailed results are reported in table 5. As mentioned in the main part of the paper (§4.3.1 and figure 2), Smooth FixMatch outperforms Fixmatch more significantly than FlexMatch. In other words, while FlexMatch may obtain better performances on average (in particular thanks to the *a priori* information on class distribution that is not used in other methods) it is much worse than FixMatch on some runs, while our methods systematically performs better.

Fold	FixMatch*	FlexMatch*	Smooth FixMatch	Smooth FixMatch-sq†	Smooth FixMatch-sqrt††
0	9.77	5.33	6.25	7.14	6.27
1	7.43	5.13	7.07	5.36	6.39
2	7.48	5.13	5.45	5.22	7.14
3	7.36	14.73	6.32	7.48	6.48
4	15.60	5.07	11.44	14.08	13.51
5	8.01	5.32	6.47	6.96	6.54
	9.28	6.79	7.24	7.71	7.72
	±2.95	±3.55	±1.94	±2.98	±2.60
max	15.60	14.73	11.44	14.08	13.51
min	7.36	5.07	5.45	5.22	6.27
range	8.24	9.66	5.99	8.86	7.24
gain wrt FixMatch		2.10 ±5.68	2.04 ±1.27	1.57 ±0.91	1.55 ±1.02
p-value		0.15625	0.015625	0.03125	0.015625

Table 5: Error rate on 6 folds (with each fold result) of CIFAR-10 with 40 labels of our runs of FixMatch without and with the smoothness factor. The p-value line corresponds to the result of the Wilcoxon one-sided test for the given method having lower error rate than FixMatch, which is used as baseline (median being significantly positive, lower is better). Range is maximum minus minimum, smaller is better. * Our run using the codebase of Zhang et al. (2021). † Factor shape is square, threshold set at 0.82. †† Factor shape is square root, threshold set at 0.98.

	C100-2500					
Fold	FixMatch*†	FlexMatch*†	Smooth FixMatch†	FixMatch*††	FlexMatch*††	Smooth FixMatch††
0	30.27	30.48	30.04	28.84	28.65	28.84
1	30.18	29.84	29.57	29.29	27.65	29.50
2	30.61	30.19	30.73	29.30	28.51	29.30
	30.35	30.17	30.11	29.14	28.27	29.21
	±0.19	±0.26	±0.48	±0.21	±0.44	±0.28
gain wrt FixMatch		0.18 ±0.28	0.24 ±0.30		0.87 ±0.59	−0.07 ±0.10
p-value		0.25	0.25		0.125	0.8413‡

Table 6: Accuracy on 3 folds reporting error rate of last checkpoint. Baselines. C100-400: FixM 46.42, FlexM 39.94. C100-2500: FixM 28.03, FlexM 26.49. FullySup 19.30. * Our runs. † Weight decay rate 0.0005 as for CIFAR-10. †† Weight decay rate 0.001 as in experiments carried out in Sohn et al. (2020); Zhang et al. (2021). ‡ Two out of three are ties, p-value not reliable.

A.3.2 Threshold

Detailed results reported in table 7. Indicative plot of linear and quadratic factors for $\tau = 0.82, 0.91, 0.95$ in fig. 6. Smooth FixMatch with linear shape factor can be seen to have a milder dependence on the value of the threshold.

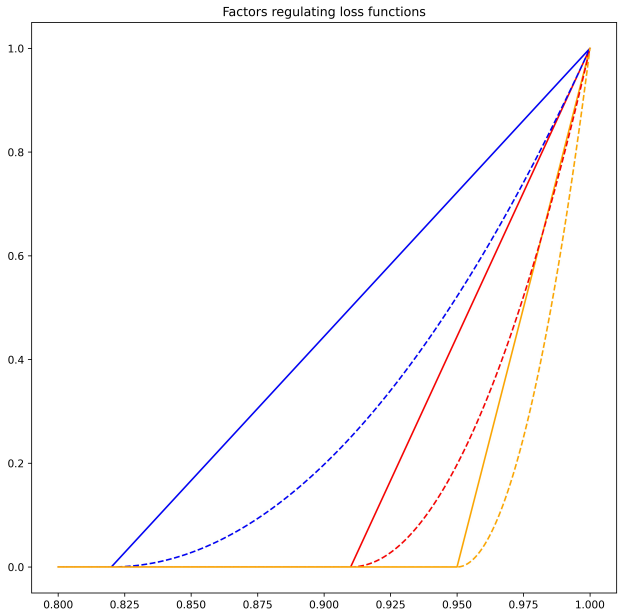

Figure 6: Plot of linear (contiguous) and quadratic (dashed line) factors for $\tau = 0.82, 0.91, 0.95$.

τ	0.75	0.82	0.86	0.91	**0.95**	0.98	0.999
FixMatch		15.85	8.06	7.41	7.43	7.21	12.97
Smooth FixMatch		15.68	5.20	7.38	7.07	6.56	8.59
Smooth FixMatch-sq	7.38	5.36	7.52	7.34	5.41	7.27	14.21
Smooth FixMatch-sqrt		8.09	7.44	6.72	6.55	6.39	8.24

Table 7: Ablation study on upper threshold: error rate on the first fold of CIFAR-10-40 when the upper threshold varies.

A.3.3 Class imbalance

We introduce a class imbalance by reducing one class, chosen randomly, to 60%. Detailed results are reported in table 2. When the implicit hypothesis of balanced classes does not hold (even lightly in this case), it is clear that the performance of FlexMatch drops significantly, even below that of FixMatch. On the contrary, our method still exhibits better performance than FixMatch.

A.3.4 SGD momentum parameter

Detailed results reported in table 9. The default parameter for the rest of our experiments is $\beta = 0.90$.

The value of the momentum has a significant influence on the results. However, the method we propose is less sensitive to this hyperparameter than the original FixMatch.

A.3.5 Aggregate comparison

In the corresponding section of the paper, the results on FixMatch and Smooth FixMatch contained in tables 5, 7, 2, 9 and 11 are compared. Smooth FixMatch is found to outperform FixMatch at a high significance level, in almost all experiments.

Fold	FixMatch	FlexMatch	Smooth FixMatch
0	7.18	7.96	6.70
1	5.36	7.95	6.66
2	10.26	8.83	5.25
3	8.06	10.65	6.99
4	16.21	24.61	13.65
5	8.75	9.52	7.50
	9.30 ±3.43	11.59 ±5.90	7.78 ±2.71
max	16.21	24.61	13.65
min	5.36	7.95	5.25
range	10.85	16.66	8.40
gain wrt FixMatch		−2.28 ±3.05	1.51 ±1.94
p-value		0.953125	0.109375
drop due to imbalance	0.03 ±1.83	4.80 ±7.15	0.55 ±0.89
p-value	0.421875	0.109375	0.15625

Table 8: Ablation study on class imbalance: error rates with one class reduced to 60%. Drop due to imbalance is the difference with respect to the performance of the same model without class imbalance (smaller is better).

β	0.50	0.75	0.85	**0.90**	0.95	0.99
FixMatch	33.24	33.05	30.01	15.60	14.83	34.05
Smooth FixMatch	30.09	29.33	15.31	11.44	13.43	34.80

Table 9: Ablation study on momentum: error rate when the momentum of SGD varies, reporting error rate of last checkpoint on the fourth fold of CIFAR-10-40.

λ_Φ	**0.**	0.0005	0.005	0.01	0.05
Smooth FixMatch	11.44	13.42	13.55	13.25	12.97

Table 10: Ablation study on weight of factor: error rate when the weight of factor varies, reporting error rate of last checkpoint on the fourth fold of CIFAR-10-40.

A.3.6 Factor as loss

We study the effect of learning the smoothing factor by varying the corresponding weight λ_Φ and report results in table 10.

We added the smoothing factor in the total loss with factors 0.0005, 0.005, 0.01, 0.05. This addition amounts to learning the pseudo-label on the weak augmentation, which, as we argue in §C, should contribute to back-propagation with a smaller factor than the unsupervised loss on the strong augmentation.

The upper limit $\bar{\lambda}_\Phi$, determined experimentally by calculating the equilibrium values of the relevant losses, was found to be 0.05 for CIFAR-10-40. In a somewhat counter-intuitive way, we found a weight factor of even 1% of this upper limit increased the error rate, and that on the 4th fold of CIFAR-10-40 the error rate was more or less constant for all tested values of λ_Φ (even though it remained below the error rate of FixMatch).

Experiments on other folds with $\lambda_\Phi = \bar{\lambda}_\Phi = 0.05$ confirmed that this value is indeed a tipping point for the error rate, as the experiment on the 1st fold gave an error rate $\sim$14%, with FixMatch having an error rate of $\sim$7.5%.

The value of λ_Φ was thus set to 0 for all other experiments.

A.3.7 Ablation study on initial condition

In Sohn et al. (2020) the authors observed that the model has the same variance across folds, as on the same fold of CIFAR-10-40 with varying random seed. They obtained a mean error $9.03 \pm 2.92\%$ over 5 runs.

We repeated the ablation using the 4th fold, which was the only one to converge to the low basin of attraction for FixMatch in our standard experiments with an error rate of 15.60%, cf. table 11.

In our ablation, FixMatch was found to have an error rate of 16.95 ± 3.27, while Smooth FixMatch 12.97 ± 0.78, a striking difference. The performance of FixMatch in the low basin of attraction seems to be very sensitive when an unfavorable network initialization and/or unfavorable order and composition of batches is coupled with a less representative labeled dataset. We note that the initial condition controls not only the initialization of the network, but also the augmentations applied to the images throughout training. More precisely, FixMatch lost up to $\sim$7.%, giving an error rate as high as $\sim$22.50%, while the smooth version of the algorithm did not lose more than 2.%. The worst performance of the smooth version is on par with the best one of the original, non-smooth, implementation. This means that the maximum experimentally observed error rate for Smooth FixMatch is as high as the lowest error observed for FixMatch, assuming convergence to the low basin of attraction for the standard CIFAR-10-40 benchmark.

Cross comparison of these results with those of table 5 shows that FixMatch indeed has a comparable statistical behavior across folds and across runs. On the contrary, Smooth FixMatch is less sensitive to the random condition than to the representativity of the labeled dataset, which is a more desirable behavior. The representativity of the labeled dataset, coupled with an unfavorable random initial condition can severely penalize FixMatch, while Smooth FixMatch behaves in a significantly more stable manner, as it does also when the labeled dataset varies. Once again, this proves our main argument, as smoothing out the discontinuities in the derivative of the loss accounts for a considerable reduction of the volatility of the error rate, and a pointwise improvement, when each possible factor introducing stochasticity is taken into account.

seed	1917	2001	2019	**2046**	2067	
FixMatch	17.71	22.63	13.26	15.60	15.56	16.95 ± 3.27
Smooth FixMatch	13.14	13.28	13.55	11.44	13.43	12.97 ± 0.78

Table 11: Ablation study on random seed: error rate on the 4th fold of CIFAR-10-40 when the random seed varies. The default random seed for our experiments is 2046.

The seed 1917 produced an instability for FixMatch, with a minimum error rate $\sim$10.% reached at $\sim 800K$ iterations, with a final error rate $\sim$17.%. Likewise, the best error rate for seed 2001 was $\sim$13.5%, reached at $\sim$95% of iterations, but then rose to $\sim$22.5%.

Smooth FixMatch with linear factor did not exhibit this kind of instability in the numerous experiments carried out under the same conditions.

Detailed results are presented in table 11. Results not included in the aggregate result comparison of §4.4 of the paper.

A.4 Random choice of labeled dataset

We compare Smooth FixMatch with FixMatch on CIFAR-10 with an increasing number of randomly selected labeled images until one algorithm reaches stable performance on par with the fully supervised baseline. We start with 40 labels and proceed with increments of 10 labels. The choice for each labeled fold is made independently, depending only on the random condition, and incrementally, i.e. the fold of 40 labels for the seed 0 is contained by construction in the fold of 50 labels for the seed 0. We present detailed results in an Excel sheet available here. This benchmark is therefore a dummy version of an Active Learning scenario with an increment of the size of one label per class, but random selection.

B Details on setup of experiments

Our experiments were run using the codebase of Zhang et al. (2021).

B.1 CIFAR datasets

The values for hyperparameters used in our experiments are provided in table 12.

Experiments on CIFAR-10 were executed on single GPU P5000 16Go. Experiments on CIFAR-100 were executed on 2 GPUs A100 40Go.

B.2 Experiments on Imagenet

Training with the standard FixMatch protocol on Imagenet gave equally low accuracy for both FixMatch and Smooth FixMatch, see figure 7. The reason for this low performance is that the network struggles throughout training to learn the labeled dataset, as is made clear by figures 8 and 9. Since accuracy on the labeled dataset is not satisfactory for the greatest part of training, the quality of pseudo-labels, especially in the beginning of training, is low, which results in a high degree of confirmation bias and explains the low performance.

These results are, for obvious reasons, irrelevant to the (non-)smoothness of the loss function. The algorithm fails to interpolate sufficiently the labeled images, and therefore its extrapolation properties on the unlabeled dataset are irrelevant. The discontinuity in the derivative of the loss function of FixMatch occurred in the unsupervised loss, and our improvement therefore does not improve the interpolation properties of FixMatch. Consequently, our improvement should not be expected to have any impact on performance, which is precisely what we observed in our experiment.

	CIFAR-10	CIFAR-100
Optimizer	SGD	
learning rate	0.03	
momentum	0.9	
weight decay	5e−4	1e−3
ema	0.999	
λ_u	1.1	
λ_Φ	0.	
τ	0.95	
random seed	2046	
Architecture	WideResNet	
Depth	28	
Width	2	8
Num. iterations	2^{20}	
Labeled Batch-size	64	
Unlabeled Batch-size	448	
Folds	$0-5$	$0-2$

Table 12: Hyperparameter values of Smooth FixMatch for CIFAR datasets. We follow the values used in Sohn et al. (2020) except for λ_u which is determined by the calculations of §C

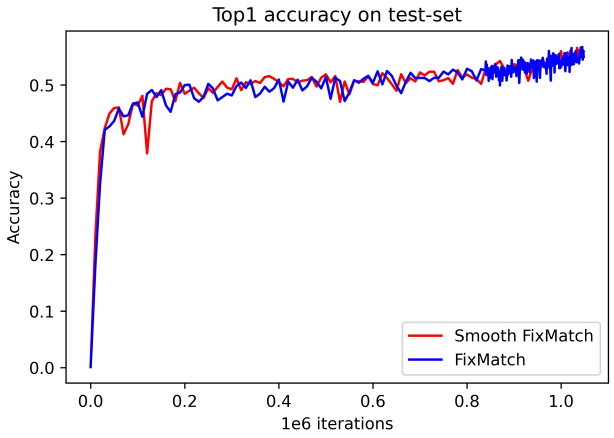

Figure 7: Accuracy on the test-set of Imagenet when training with standard FixMatch protocol.

This experiment being of low practical value, we decided to not include it in the main part of the article. We only note that in this regime our method does not hinder the performance of FixMatch, as the curves (red for Smooth- and blue for FixMatch) are practically indistinguishable

For completeness, we give a comparative graph of the evolution of accuracy on labeled images of the original FixMatch algorithm, trained with standard parameters on CIFAR-10-40, CIFAR-100-2500 and Imagenet-10%. The deterioration is clear and establishes our point.

Experiments on ImageNet were run on one node with 8 GPUs A100 40Go.

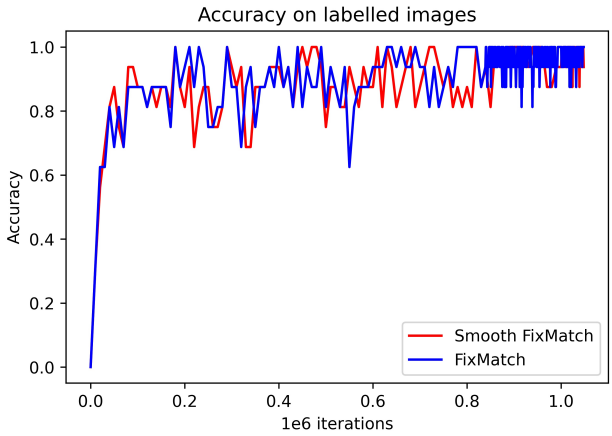

Figure 8: Accuracy on the labelled dataset of Imagenet when training with standard FixMatch protocol.

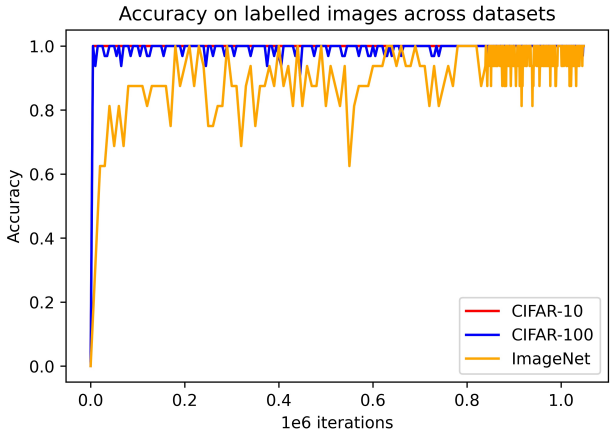

Figure 9: Comparison of the accuracy of FixMatch on the current batch of labelled images across datasets.

C Gauging the weight of factor as loss

C.1 Calculation for λ_u

We follow notation of §3.8 of the paper and provide the calculation for the factor of the unsupervised loss used in our experiments.

Both the unsupervised losses and Φ reach equilibrium values very early in training. If ℓ_Φ is the one for Φ, we impose that

$$\lambda_{u,FM}\ell_{FM} = \lambda_{u,SFM}\ell_\Phi\ell_{SFM} \tag{20}$$

The value of $\lambda_{u,SFM}$ for $\lambda_{u,FM} = 1$ can be determined experimentally ($\sim$1.1 for CIFAR datasets and $\sim$1.8 for Imagenet) and is therefore not a hyperparameter.

C.2 Calculation of $\bar{\lambda}_\Phi$

We follow the notation of §3 of the paper.

Imposing that the derivative for the unsupervised loss be equal to the derivative of L_Φ gives, in the linear case,

$$\frac{\lambda_{u,SFM}\ell_\Phi}{\ell_{SFM}} = \frac{\lambda_\Phi\ell_\Phi}{1-\tau} \tag{21}$$

	FixMatch	Smooth FixMatch
Optimizer	SGD	
learning rate	0.03	
momentum	0.9	
weight decay	3e−4	
ema	0.999	
λ_u	1.	1.8
λ_Φ	0.	
τ	0.70	
random seed	2046	
Architecture	ResNet50	
Num. iterations	2^{20}	
Labeled Batch-size	128	
Unlabeled Batch-size	896	
Fold	0	

Table 13: Hyperparameter values for Imagenet, for FixMatch and Smooth FixMatch.

The original implementation of FixMatch established that the network benefits more from learning the strong augmentation than the weak one. Therefore, a break in performance should be expected when equality is satisfied, with better performance in the regime

$$\lambda_\Phi < \bar{\lambda}_\Phi = \frac{(1-\tau)\lambda_{u,SFM}}{\ell_{SFM}} \tag{22}$$

The quantity $\bar{\lambda}_\Phi$ is not a hyperparameter, as it can be determined experimentally.