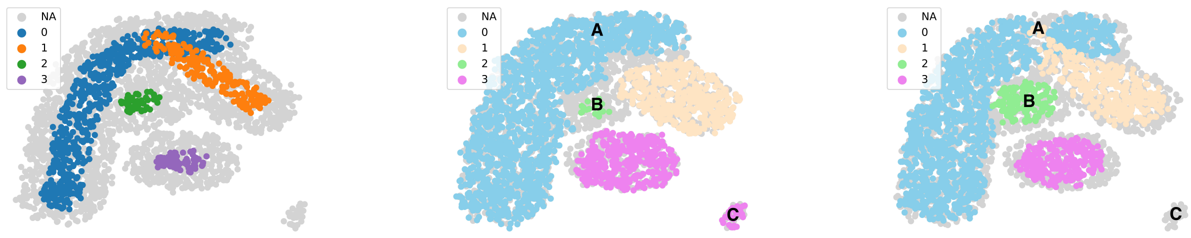
SemiGPC: Distribution-Aware Label Refinement for Imbalanced Semi-Supervised Learning Using Gaussian Processes

Abdelhak Lemkhenter, Manchen Wang, Luca Zancato,
Gurumurthy Swaminathan, Paolo Favaro, Davide Modolo
AWS AI Labs
abdelhak.lemkhenter@inf.unibe.ch (internship), {zancato, pffavaro, dmodolo}@amazon.com



(a) Initial labels (b) Similarity-based predictions (c) SemiGPC predictions

Figure 1. **SemiGPC pseudo-labeling with class imbalance**. (a) Four-class dataset containing unlabeled (gray) and labeled samples (in color). (b) and (c) show the labels propagated according to the similarity-based aggregate [17] and SemiGPC methods. (c) In A, the initial labels are mixed so SemiGPC is more conservative there (many samples are not pseudo-labeled at the current threshold level). In B, SemiGPC is able to propagate the labels of the minority green class despite being surrounded by the majority blue one. In C, SemiGPC assigns low confidence to the set of outliers (labels are not propagated). In contrast, the similarity-based approach expands the majority classes at the expense of the minority ones, *cf*. B and C.

Abstract

In this paper we introduce SemiGPC, a distribution-aware label refinement strategy based on Gaussian Processes where the predictions of the model are derived from the labels posterior distribution. Differently from other buffer-based semi-supervised methods such as CoMatch [17] and SimMatch [34], our SemiGPC includes a normalization term that addresses imbalances in the global data distribution while maintaining local sensitivity. This explicit control allows SemiGPC to be more robust to confirmation bias especially under class imbalance. We show that SemiGPC improves performance when paired with different Semi-Supervised methods such as FixMatch [23], ReMixMatch [4], SimMatch [34] and FreeMatch [32] and different pre-training strategies including MSN [2] and Dino [5]. We also show that SemiGPC achieves state of the art results under different degrees of class imbalance on standard CIFAR10-LT/CIFAR100-LT especially in the low data-regime. Using SemiGPC also results in about 2% avg. accuracy increase compared to a new competitive baseline on the more challenging benchmarks SemiAves, SemiCUB, SemiFungi [27] and Semi-iNat [26].

1. Introduction

Semi-Supervised Learning offers a more cost effective alternative to fully supervised learning when scaling up the data collection process. Current state the of the art semi-supervised methods rely on self-learning by generating pseudo-labels for the unlabeled samples. However, pseudo-labels can also hurt the final performance when they introduce persistent incorrect predictions, a problem known as confirmation bias. In particular, self-learning can bias the label distribution if the data is imbalanced. To address this, recent works such as CoMatch [17] and SimMatch [34] rely on a buffer of samples to refine the predicted pseudo-labels. However, no counter measure is adopted to globally balance the data in the memory bank. As such, the resulting *refined pseudo-labels* are plagued by the class imbalances present in the unlabeled data.

To overcome these limitations we introduce SemiGPC, a novel semi-supervised learning method that generates pseudo-labels using a distribution-aware label-refinement strategy. This distribution awareness stems from the use of Gaussian Processes, which accounts for local data concentration disparities and counteracts them. This results in more robust pseudo-labels especially for minority classes and outliers as shown in Figure 1. In particular, SemiGPC correctly assigns nearby points to the minority classes despite the larger count of the majority class at a bigger scale, *i.e.* it has a better local sensitivity, while remaining faithful to the global data distribution. SemiGPC is flexible and can be used on-top of previous label-refinement schemes on other semi-supervised methods. Furthermore, we show that the similarity-based pseudo-labels heuristics used in SimMatch and CoMatch can be cast as a special case of SemiGPC. To improve computational efficiency of our method and allow for fast batched updates essential for Semi-Supervised learning methods, we pair SemiGPC with a batched online update rule that significantly reduces its forward pass cost ($\times$7.5 speed-up). We show the benefit of using SemiGPC on top of semi-supervised algorithms such as FixMatch [23], ReMixMatch [4], SimMatch [34] and FreeMatch [32] ($\sim$ 0.8% avg. improvement) and different self-supervised pre-training strategies such as MSN [2] and Dino [5] resulting in a $\sim$ 1.3% avg. improvement. This highlights the general purpose nature of SemiGPC as a relevant extension for semi-supervised methods based on label refinement strategies. We experimentally show that SemiGPC is able to achieve state of the art results on CIFAR10-LT ($\geq$ $+7.65\%$) and CIFAR100-LT ($\geq$ $+1.84\%$) as well as the more challenging semi-supervised benchmarks SemiAves, SemiCUB, SemiFungi and Semi-iNat ($\sim$ $+1.92\%$ compared to our baseline and $\sim$ $+20.52\%$ compared to numbers reported in the literature [26, 27]). We also show that SemiGPC is able to narrow the gap between the high and low data regimes with 10-100x fewer labeled samples as we report a 45% and 32% relative improvement over the baseline across regimes for CIFAR10-LT and CIFAR100-LT respectively.

2. Related Works

Consistency-based Semi-Supervised Learning. Semi-supervised learning methods such as FixMatch [23], ReMixMatch [4], SimMatch [34] and FreeMatch [32] share the common design choice of enforcing the consistency of the model predictions across augmentations of different strength levels on the unlabeled samples. FixMatch [23] enforces this consistency as a cross-entropy loss applied using the one-hot encoding of the model predictions on weakly augmented unlabeled samples as pseudo-labels for their strongly augmented counterpart. Weak augmentations consist of random image flipping and translation while strong augmentation combine AutoAugment [6] and Cutout [8]. The consistency loss is only applied on high confidence predictions where the predicted probability value is higher than a fixed threshold. ReMixMatch [4] instead opts for using temperature sharpened model predictions as pseudo-labels for its consistency regularization in addition to a rotation prediction regularization loss. FreeMatch [32] introduces a per-class confidence threshold update rule based on the models predictions combined with an entropy-based diversity loss making it more suitable for imbalanced settings. Methods such as CoMatch [17] and SimMatch [34] choose to enforce consistency across additional data representations. In particular, CoMatch [17] encourages the consistency between the pseudo-labels and embeddings similarity graphs while SimMatch [34] encourages the consistency between semantic-level and instance-level pseudo-labels. Both methods choose to smooth the predicted pseudo-labels based on a similarity based aggregate of a memory buffer of samples in order to mitigate confirmation bias. However, these pseudo-label refinement strategies fail to eliminate data biases w.r.t. the class balance.

Probabilistic Models for Semi-Supervised Learning. Gaussian Processes (GP) are a class of non-parametric function approximation methods fully characterized by their mean and kernel functions. Given a set of observations and their corresponding measurements, GPs define a posterior distribution over measurements for new observations. Their ability to explicitly model uncertainty makes them a natural fit for Semi-supervised learning. Early works such as [15] introduce GPs in the context of Semi-supervised learning by assuming that the data density in regions between the class-conditional densities should be low while [22] leverages GPs to model the relationship between labeled and unlabeled samples by incorporating the geometry of the latter in the construction of the global kernel function. However such early works are limited to toy datasets due to the computational cost of GP. The more recent UaGGP work [18] proposed to address uncertainty caused by erroneous neighborhood relationships in the context of graph-based semi-supervised learning by leveraging the ability of GPs to generalize well from few samples. NP-Match [29] proposed Neural Processes instead of GPs as a probabilistic model for uncertainty estimation which, in turn, allows for better computational efficiency compared to MCDropout [12].

Gaussian Processes and Deep Learning. Beyond Semi-Supervised Learning, Gaussian Processes have been used alongside neural networks in multiple other fields. DG-PNet [14] relies on GPs in the context of dense few-shot segmentation to capture complex appearance distributions while [16] leverages GPs for fast and accurate uncertainty estimates in robotics systems. Furthermore, different works draw parallels between Gaussian Processes and Neural Networks by interpreting the action functions of the latter as as interdomain inducing features [10] or by proving a correspondence between the two classes of models [33].

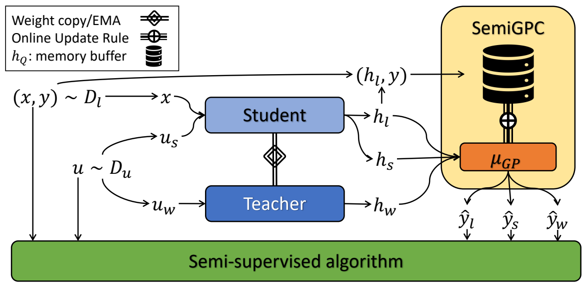


Figure 2. **SemiGPC Outline.** x, u_s, u_w, h and y are the labeled, strongly/weakly augmented unlabeled samples, their feature vector and the ground truth labels respectively. The SemiGPC buffer is used to derive the model predictions $\hat{y}$.

3. SemiGPC

In this section, we present the basic framework of consistency-based semi-supervised learning, how it can be extended with a memory buffer and analyze where confirmation bias comes into play. We then introduce our Gaussian Processes-based classifier, SemiGPC, and highlight its advantages over other classifiers using a toy example. The general outline of SemiGPC is shown in Figure 2. In the following, we shall denote the labeled dataset with $\mathcal{D}_l = \{(x_l^i, y_l^i)\}_{i=1}^{n_l}$ and the unlabeled one with $\mathcal{D}_u = \{(x_u^i)\}_{i=1}^{n_u}$, where $x \in X$ are RGB images and $y \in \mathbb{R}^C$ are labels belonging to a fixed set of concepts C. We indicate a feature extractor with $h : X \to Z$ where $Z = \mathbb{R}^d$ and d is the dimension of the feature space, and call the classification head $g : Z \to \mathbb{R}^C$. The model predictions are defined as $\hat{y}(x) = g \circ h(x)$.

3.1. Consistency-based Semi-Supervised Learning

Given labeled and unlabeled datasets $\mathcal{D}_l$ and $\mathcal{D}_u$, consistency based Semi-Supervised methods [4, 17, 23, 32, 34] rely on the labeled set and high-confidence pseudo-labels computed on the unlabeled set. Pseudo-labels are computed using strongly $\text{aug}_s(x)$ and weakly $\text{aug}_w(x)$ augmented views of a given image x, as follows: $\hat{y}_s(x) = g(h(\text{aug}_s(x)))$ and $\hat{y}_w(x) = g(h(\text{aug}_w(x)))$. We refer to [17, 23, 34] for typical strong and weak data augmentations. In this work, we adopt those used in FixMatch [23]. More precisely, the labeled and unlabeled losses are defined as

$$\begin{aligned} \mathcal{L}_l &= H(\hat{y}_w(x), y); \quad (x, y) \in \mathcal{D}_l \\ \mathcal{L}_u &= \mathbb{1}_{[\text{conf}(x) > \tau]} H(\hat{y}_s(x), f(\hat{y}_w(x))); \quad x \in \mathcal{D}_u \\ \text{with } \text{conf}(x) &= \max[\text{softmax}(\hat{y}_w(x))] \end{aligned} \tag{1}$$

where H is the cross-entropy loss, τ is the confidence threshold specifying which unlabeled samples to use and $f : \mathbb{R}^C \to \mathbb{R}^C$ is a label refinement function (*e.g.* see [4]). The model confidence is defined as the maximum of the softmax vector. Different choices of f include the identity function (no refinement), one hot encoding (hard pseudo-labels used in FixMatch [23]), temperature sharpening [4], *etc*.

For a linear classification head, such pseudo-labels are sensitive to outliers in the sense that a new unlabeled sample located far from the labeled data can have high confidence, *cf*. Figure 3a. To overcome such limitation, works such as SimMatch [34] and CoMatch [17] propose to ground their pseudo-labels using a memory buffer during training. The buffer, (h_Q, y_Q), is a set of N_Q feature vectors of weakly augmented labeled samples using aug_w, *i.e.*, $h_Q := \{h(\text{aug}_w(x_l)); x_l \sim \mathcal{D}_l\} \in \mathbb{R}^{n \times d}$, and their associated labels. Then, the smoothed pseudo-labels (output of f) on any given input x are defined as

$$f(\hat{y}(x))) = (1 - \alpha)\hat{y}(x) + \alpha \hat{y}^{sim} \tag{2}$$

$$\text{with } \hat{y}^{sim} = k(h(x), h_Q) y_Q \tag{3}$$

where α, k and $h(x)$ are a smoothing factor, a kernel similarity function and the feature representation of the input x. The pseudo-labels $\hat{y}^{sim}$ introduced in eq. (3) closely reflects the data distribution. However, biases present in the data, if not addressed, could be amplified due to the un-weighted kernel average (*e.g*., by favoring the majority classes *cf*. Figure 3b over minority ones). In this work, we address short-comings of previous approaches by introducing a normalization term, computed leveraging Gaussian Processes, that automatically counteracts class imbalances in the data.

3.2. Gaussian Processes-based Label Refinement

We now introduce SemiGPC, our label refinement strategy based on Gaussian Processes (GPs). Our key motivation is that the normalized kernel similarity used in the GP posterior mean helps address class imbalance by equalizing the local contribution of each sample in the buffer w.r.t. each class population. Similarly to previous methods [17, 23, 34], SemiGPC aggregates global information for the refinement of each pseudo-label's input-location. However, SemiGPC retains local sensitivity by favoring the minority classes when appropriate despite the global aggregate favoring the majority ones as first shown in Figure 1.

Given a memory buffer containing features and labels (h_Q, y_Q), we define SemiGPC refined pseudo-labels as

$$\hat{y}^{GP}(x) = \lambda\, \mu^{GP}(h(x)) = \lambda\, k(h(x), h_Q) K^{-1} y_Q \tag{4}$$
$$\text{with } K = k(h_Q, h_Q) + \sigma^2 I$$

where μ^{GP} is the posterior mean of the GP, λ is the logit scaling factor, σ a regularization parameter of the GP which represents how much we trust labels in the memory bank and k is the GP kernel function (*e.g*., the RBF kernel). By comparing equations (4) and (3) we see that the GP approach aggregates all labels in the memory bank and reweights them according to the inverse covariance matrix

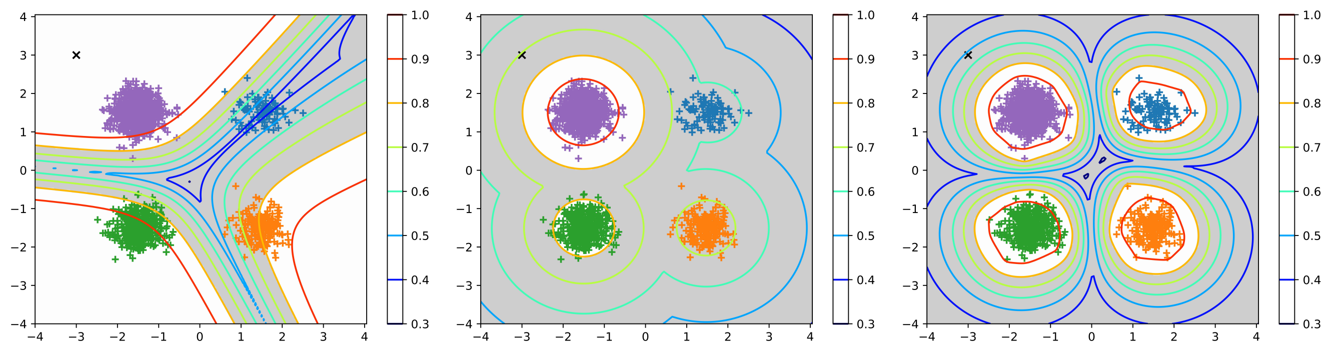

(a) Linear classifier confidence (b) Similarity-based classifier confidence (c) GP classifier confidence

Figure 3. **Comparison of confidence maps**: (a) a linear model, (b) a similarity-based classifier [17] and (c) a GP classifier are represented using the contour lines. The number of samples per class grows clockwise by a factor of 2 starting from the top right cluster. We grey out regions that are below 80% confidence. The outlier at (-3,3) is indicated with an $\times$. (c) Only the GP classifier is able to define confidence levels that are not biased toward the majority classes and ignore the outlier.

K^{-1}. Such normalization is particularly useful to counteract class imbalance as we show in Figure 3.

In the following, we use the RBF kernel defined as $k(x, y) = \eta \exp(-\|x-y\|^2/2l^2)$ where η and l are the kernel scale factor and length scale respectively. Note that eq. (4) characterizes the posterior mean of a GP whose likelihood function is Gaussian, in general, other non-Gaussian choices are available and typically applied to build GP-based classifiers [21]. However, when non-Gaussian likelihood are used, no closed-form solution exists and approximation schemes which entail higher computational costs are required [21]. Thus, we choose to refine pseudo-labels by directly regressing the logits $\hat{y}$ using a Gaussian likelihood.

Connection with other label refinement methods. Eq. (4) can also be rewritten as $\mu^{GP}(x) = k(h(x), h_Q)y_K$, a similarity-based aggregation of y_K, the propagated version of y_Q through the graph defined by K. When $\eta/\sigma^2 \rightarrow 0$ $K \rightarrow \sigma^2 I$. In this setting, eq. (4) becomes equivalent to eq. (3). Thus, we obtain the similarity-based aggregation strategy of works such as SimMatch [34] and CoMatch [17]. Furthermore, clipping the kernel below a given threshold results in a matrix K equivalent to an epsilon graph.

SemiGPC robustness to class imbalance. We now illustrate with a toy example how SemiGPC is more robust to class imbalance than previous methods. In particular, we compare SemiGPC with a linear classifier and the similarity-based classifier used in CoMatch [17] in Figure 3. We build a dataset with 4 normally distributed classes centered at (1, 1), (1, -1), (-1, -1) and (-1, 1) resp., and plot the model *confidence* as defined in eq. (1). To simulate class imbalance, the number of samples per class grows by a factor of 2 starting from the top right cluster and going clock-wise.

First, note how samples far from the data distribution, *e.g*. (-3, 3), are assigned very high confidence by the linear model although such points are locally isolated from the others and therefore should not be considered well supported by evidence. Second, note that the minority class (in blue) is a low confidence region for both the linear and similarity-based classifier, despite locally containing many samples supporting that class. On the other hand, the GP-based classifier defines an appropriate high confidence region for each supported class and its sensitivity to the confidence threshold is much smaller than the similarity-based classifier used in CoMatch [17] as highlighted by the contour plots.

Summarizing, thanks to the use of a GP-based label refinement strategy, SemiGPC is confident if: (1) the considered sample is close to a subset of h_Q regardless of whether it belongs to the majority or minority classes since K^{-1} reweighs the kernel similarity to counteract disparities in class populations while all samples far from the data are considered outliers, and (2) the sample is located in a high purity region w.r.t. y_Q since the average of conflicting results in a model confidence that is spread between classes.

3.3. Efficient GP update

As we mentioned in section 3.2, applying GPs in a classification setting requires approximation schemes that are in general computational expensive. To reduce the forward time of SemiGPC we choose to model the refined pseudo-labels using a Gaussian likelihood. In this way, computing the posterior mean for each input image only requires solving a quadratic optimization problem available in closed-form. However, computing μ^{GP} is still computationally expensive since we need to update the set h_Q after each model update and invert the covariance matrix K which scales with the cube of the memory bank size (N_Q) at each mini-batch forward pass. To speed up computations we start from the key observation that at each optimization iteration, most of the samples in the queue do not change. Therefore,

Table 1. **Complexity Comparison of GP updates.**We observe a $\times$**7.5** speedup in practice.

Classic GP update	Efficient GP update
$\mathcal{O}(N_Q^3 + BN_Q^2)$	$\mathcal{O}(B^3 + BN_Q^2 + B^2 N_Q)$

at the t-th iteration after observing the new batch of data of size B, we update the previously computed covariance at step $t-1$ with an incremental update rule. In the following, we implement SemiGPC using the well-know matrix inversion lemma [3] (Woodbury identity) which provides a simple batched iterative rank-B correction to the inverse of a given invertible matrix.

In particular, for each labeled training mini-batch of size B, we replace the B oldest samples in the buffer with the features computed using the current mini-batch. Let K_{t-1} and K_t be the covariance matrices at iteration $t-1$ and t respectively. We now show how to compute K_t by updating K_{t-1} after having updated the memory bank with the new samples from the current mini-batch. Let,

$$K_t = \begin{bmatrix} k(h_o, h_o) & k(h_o, h_n) \\ k(h_o, h_n)^\top & k(h_n, h_n) \end{bmatrix} + \sigma^2 I = \begin{bmatrix} A & C \\ C^\top & D \end{bmatrix},$$

where h_o and h_n denote the old samples that were kept in the buffer and the new samples added to the buffer. I is the identity matrix. The inverse of K_t is given by

$$\begin{aligned} K_t^{-1} &= \begin{bmatrix} A & C \\ C^\top & D \end{bmatrix}^{-1} = \begin{bmatrix} K_{11} & K_{12} \\ K_{12}^\top & K_{22} \end{bmatrix} \\ \text{where } K_{22} &= (D - C^\top A^{-1} C)^{-1} \\ K_{12} &= -A^{-1} C K_{22} \\ K_{11} &= A^{-1} + A^{-1} C K_{22} C^\top A^{-1} \end{aligned}$$

Note that computing K_t^{-1} only requires inverting the two matrices A and $(D - C^\top A^{-1} C)$. The latter is of size $B \times B$ while the former is still a relatively large matrix of size $(N_q - B) \times (N_q - B)$. However, A does not depend on the newly added samples, yet it is not equal to K_{t-1}^{-1}. Therefore, A^{-1} can be computed efficiently only requiring the inverse of a $B \times B$ matrix as follows:

$$\begin{aligned} K_{t-1}^{-1} &= \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^\top & M_{22} \end{bmatrix} \\ A^{-1} &= M_{11} - M_{12} M_{22}^{-1} M_{12}^\top \end{aligned}$$

For simplicity, we assume that the new samples are located at the end of the buffer, however the derivation remains true for an arbitrarily ordered buffer up to a permutation matrix.

To summarize, using block matrix linear algebra, the cost of the inverting K_t can be reduced to computing the inverse of a couple of $B \times B$ matrices which is in turn much more efficient when $B \ll N_q$ as is the case in our setting. The detailed derivation is provided in the supplementary material. For example, for a buffer size $N_Q \sim 16k$ and a batch size $B = 8$, using our efficient update rule results in $\times 7.5$ speedup.

Class-balanced SemiGPC. SemiGPC has the additional benefit of allowing us to explicitly address the class imbalance without altering the training scheme. We split h_Q into C class buffers and insert the new samples based on their labels, thus ensuring a balanced h_Q. We compare this approach to the classic class rebalancing in the supplementary material.

4. Experimental Settings

4.1. Implementation details

We use the semi-supervised training recipe of USB [31][1]. It uses an ImageNet [7] pre-trained ViT to initialize the student model. This training scheme allows for faster training time and better performance overall. We use a ViT Small/Tiny with a patch size of 2 and a resolution of 32 for CIFAR100/10 respectively. For our other experiments, we use a ViT Small with a patch size of 16 and a resolution of 224. All our experiments can be ran on a single V100 GPU. All our model are trained using AdamW [19] for 200 epochs using a batch size of 8. The detailed set of hyperparameters are provided in the supplementary material. For most of our experiments, we use SimMatch as our baseline. We include a comparison of SemiGPC across different algorithms in section 6.1. For all SemiGPC experiments, we use a buffer size $N_q = 16300$. Following most works in the literature, we adopt the Top 1 Accuracy as our main evaluation metric and report the mean and standard deviation across 3 random seeds.

4.2. Datasets

CIFAR10-LT, CIFAR100-LT. We evaluate SemiGPC on imbalanced versions of CIFAR10 and CIFAR100. The class distribution of these datasets can be fully described using the imbalance ratio γ and the number of samples in the majority class N_1. For each class $1 < i \leq K$ its number of samples N_i is defined as

$$\begin{aligned} N_i &= N_1 \gamma^{-\frac{i-1}{K-1}} \\ \text{where } \gamma &= {}^{N_1}/_{N_K} \end{aligned}$$

where γ, N_K, and K are the imbalance ratio, the cardinality of the minority class and the number of classes respectively.

FGVC Benchmarks. We also evaluate SemiGPC on the fine-grained semi-supervised benchmarks introduces in [26,

[1]https://github.com/microsoft/semi-supervised-learning

27]. These challenging benchmarks contain naturally long-tailed distributions with highly similar class pair. Note that both works [26, 27] argue that Semi-supervised methods struggle on such benchmarks. These datasets include a labeled set L_{in} and two unlabeled U_{in} and U_{out} with seen and unseen classes.

SemiAves. This dataset [24] is built using the Aves kingdom in iNaturalist 2018 dataset [13]. L_{in}, U_{in} and U_{out} include 200/200/800 species and 5959/26640/122208 images respectively. The test set is balanced and contains 40 samples per class. Its reported imbalance ratio is $\gamma = 7.9$.

SemiFungi. This dataset is based on the CVPR 2018 FGVCx Fungi challenge dataset [1]. L_{in}, U_{in} and U_{out} include 200/200/1194 species and 4141/13166/64871 images respectively. The test set is balanced and contains 20 samples per class. Its reported imbalance ratio is $\gamma = 10.1$.

Semi-iNat. This dataset was introduced at CVPR 2021 FGVC8 workshop [25]. L_{in}, U_{in} and U_{out} include 810/810/1629 species and 13771/91336/221912 images respectively. The test set is balanced and contains 100 samples per seen class. Its imbalance ratio is $\gamma = 8.5$.

SemiCUB. This dataset is based on the Caltech-UCSD Birds-200-2011 (CUB) dataset [28]. L_{in}, U_{in} and U_{out} include 100/100/100 species and 1000/3853/5903 images respectively. Unlike the other three, only the unlabeled sets are imbalanced with $\gamma \sim 2$ for U_{in}. The test set is balanced and contains 1000 samples.

We use $U = U_{in}$ as our unlabeled dataset. Results for $U = U_{in} \cup U_{out}$ are shown in the supplementary material.

5. Experimental Results

In this section, we showcase SemiGPC's robustness under various degrees of class imbalance across different data regimes on CIFAR10-LT and CIFAR100-LT. We then report the performance on the more challenging long-tailed semi-supervised benchmarks SemiAves, SemiCUB, SemiFungi and Semi-iNat. Lastly, we benchmark SemiGPC on the classic balanced semi-supervised splits of CIFAR10 and CIFAR100. For all our imbalanced experiments, we forgo using techniques such as CReST as they don't necessarily improve performance when combined with the USB [31] training recipe. These results can be found in the supplementary material.

5.1. Imbalanced Semi-Supervised Learning

In this section, we evaluate the robustness of SemiGPC under different degrees of class imbalance on CIFAR10-LT and CIFAR100-LT. More specifically, we explore two imbalanced settings based on whether one has access to a balanced labeled dataset or not:

- **Setting A** ($\gamma_l = \gamma_u > 1$)**.** Both the labeled and unlabeled sets are imbalanced using the same factor. Following prior works, we use $(N_1^l, N_1^u) = (150, 500)$ for the imbalanced version CIFAR100, *i.e.* CIFAR100-LT, and $(N_1^l, N_1^u) = (1500, 5000)$ for the imbalanced version CIFAR10, *i.e.* CIFAR10-LT. N_1^l and N_1^u are the number of samples for the majority class in the labeled and unlabeled datasets respectively.
- **Setting B** ($\gamma_l = 1; \gamma_u > 1$)**.** Only the unlabeled set is imbalanced. For the labeled setting, we use 4 samples per class resulting $\times 100/\times 10$ fewer labeled samples compared to A for CIFAR10-LT and CIFAR100-LT respectively. We argue that this setting is more challenging and better represents real-world scenarios. Indeed, realistically, a small set of balanced labeled samples can be curated while one cannot make any assumptions on the distribution of the unlabeled dataset based on its labeled counterpart.

Table 2. Top1 Accuracy obtained on CIFAR100-LT for different values of $\gamma_l = 1$ and γ_u. †: A class balanced buffer is used for SemiGPC. $*$: as reported by [11]. The difference to the baseline is highlighted in green/red.

Model	γ_l	γ_u	n_{lb}	**Top1 Acc**
CoSSL [11]$*$	20	20	4741	55.80 ± 0.62
SimMatch	20	20	4741	83.38 ± 0.48
w/ SemiGPC†	20	20	4741	**83.76** ± 0.26 **(+0.37)**
CoSSL [11]$*$	50	50	3751	48.90 ± 0.61
SimMatch	50	50	3751	78.82 ± 0.60
w/ SemiGPC†	50	50	3751	**79.79** ± 0.08 **(+0.97)**
CoSSL [11]$*$	100	100	3218	44.10 ± 0.59
SimMatch	100	100	3218	73.90 ± 0.75
w/ SemiGPC†	100	100	3218	**74.48** ± 0.98 **(+0.58)**
SimMatch	1	20	400	76.28 ± 0.28
w/ SemiGPC	1	20	400	**77.79** ± 0.51 **(+1.53)**
SimMatch	1	50	400	72.78 ± 0.29
w/ SemiGPC	1	50	400	**75.21** ± 0.53 **(+2.43)**
SimMatch	1	100	400	70.19 ± 0.43
w/ SemiGPC	1	100	400	**73.47** ± 0.63 **(+3.28)**

CIFAR100-LT (Table 2). For Setting A, we use the class-balanced version of SemiGPC. We also include the numbers reported by [11] for CoSSL+ReMixMatch as they represent the current state of the art. For setting A, we observe that SemiGPC outperforms the baseline across all values of γ_u. This highlights SemiGPC's robustness with respect to class imbalance and its inherent ability to address it explicitly using a balanced buffer. The results obtained for setting B further support the robustness of SemiGPC with respect to class imbalance. Indeed, when provided with balanced samples that are $10\times$ fewer than setting A, SemiGPC is able to outperform our baseline across all values of γ_u by a margin greater than $+1.5\%$. Additionally, for each model and value γ_u we measure the gap $\Delta(\gamma_u) = \text{Acc}(A) - \text{Acc}(B)$ between the accuracies $\text{Acc}(A)$ and $\text{Acc}(b)$ in setting A and B

Table 3. Top1 Accuracy obtained on CIFAR10-LT for different values of $\gamma_l = 1$ and γ_u. †: A class balanced buffer is used for SemiGPC. $*$: as reported by [11]. The difference to the baseline is highlighted in green.

Model	γ_l	γ_u	n_{lb}	Top1 Acc
CoSSL [11]$*$	50	50	4196	87.70 $\pm$0.21
SimMatch	50	50	4196	96.48 $\pm$0.26
w/ SemiGPC†	50	50	4196	**96.80** $\pm$0.12 **(+0.32)**
CoSSL [11]$*$	100	100	3720	84.10 $\pm$0.56
SimMatch	100	100	3720	94.59 $\pm$0.59
w/ SemiGPC†	100	100	3720	**95.74** $\pm$0.37 **(+1.15)**
CoSSL [11]$*$	150	150	3496	81.30 $\pm$0.83
SimMatch	150	150	3496	94.07 $\pm$1.46
w/ SemiGPC†	150	150	3496	**95.41** $\pm$0.56 **(+1.34)**
SimMatch	1	50	40	80.59 $\pm$2.24
w/ SemiGPC	1	50	40	**88.22** $\pm$2.38 **(+7.63)**
SimMatch	1	100	40	76.69 $\pm$2.13
w/ SemiGPC	1	100	40	**86.86** $\pm$4.48 **(+10.17)**
SimMatch	1	150	40	75.68 $\pm$4.31
w/ SemiGPC	1	150	40	**84.25** $\pm$8.92 **(+8.57)**

respectively. When comparing Δ averaged over all γ_u values, we observe a gap of 5.62% and 3.85% for SimMatch and SemiGPC respectively. In addition to improving performance across both settings, SemiGPC is better at bridging the gap between the two data regimes by about 32%.

CIFAR10-LT (Table 3). For setting A, we observe the SemiGPC outperforms the baseline across different values of γ_u especially for the more challenging setting $\gamma_u = 150$ where we observe a gap of $+1.34\%$. This highlights the robustness of SemiGPC to class imbalance. SemiGPC also largely outperforms our baseline in the setting B. We observe an accuracy increase of at least 7.63% across all values of γ_u with the gap growing bigger for higher values of γ_u up to $+10.17\%$. Additionally, we report an average gap across settings of 17.39% and 9.54% for SimMatch and SemiGPC respectively, *i.e.*, a relative improvement of 45%. Thanks to its normalization scheme, SemiGPC reduces the risk of confirmation bias which is more prominent when the labeled data is scarce.

5.2. Semi-Supervised FGVC Benchmarks

In the section, we evaluate the performance of SemiGPC on the naturally long-tailed semi-supervised benchmarks such as SemiAves, SemiFungi and SemiCUB and for Semi-iNat. In addition to the class imbalance, these datasets includes highly similar classes, *cf*. supplementary material.

We report the obtained results on Table 4. For reference, we report the numbers obtained by [27] for SemiAves, SemiFungi and SemiCUB and obtained by [26] for Semi-iNat. We show that using the USB [31] training recipe produces a strong semi-supervised baseline as opposed to the numbers reported by [26, 27]. Furthermore, SemiGPC outperforms the baseline on all fine-grained benchmarks. This is especially true for the most imbalanced dataset Semi-Fungi ($\gamma = 10.1$) where SemiGPC improves upon the baseline accuracy by $+3.49\%$. This shows that SemiGPC is not only more robust with respect to class imbalance on artificially skewed benchmarks such CIFAR10/100-LT but is also better suited for naturally imbalanced datasets containing fine-grained classes where it establishes a new state of the art.

Table 4. Top1 Accuracy obtained on the considered fine-grained semi-supervised benchmarks.$*$: as reported by [26, 27]. The difference to the baseline is highlighted in green.

Dataset	Model	Top1 Acc
SemiCUB	FixMatch [27]$*$	53.20
	SimMatch	84.53 $\pm$0.45
	w/ SemiGPC	**85.43** $\pm$0.67 **(+0.90)**
SemiAves	FixMatch [27]$*$	57.40 $\pm$0.80
	SimMatch	68.47 $\pm$0.43
	w/ SemiGPC	**69.59** $\pm$0.09 **(+1.12)**
SemiFungi	FixMatch [27]$*$	56.30 $\pm$0.50
	SimMatch	68.01$\pm$0.19
	w/ SemiGPC	**71.50** $\pm$0.49 **(+3.49)**
Semi-iNat	FixMatch [26]$*$	44.10
	SimMatch	64.95 $\pm$0.11
	w/ SemiGPC	**66.54** $\pm$0.85 **(+1.59)**

5.3. Standard CIFAR10/CIFAR100

Lastly, we evaluate our SemiGPC method on different split of CIFAR100 and CIFAR10. We report the obtained performance on Table 5 when using 200/400 and 40/250 labeled samples for CIFAR100 and CIFAR10 respectively. For reference we include the numbers reported by USB [31] and FreeMatch [32] as the current state of the art. We observe that SemiGPC improves performance across different amount of available labeled samples on CIFAR100, with the biggest improvement $+0.83\%$ in the low data regime. However, SemiGPC is simply on par with the baseline on CIFAR10. We argue that the semi-supervised performance is already saturated on this benchmark when using the USB training recipe.

6. Ablations

In order to establish the general purpose nature of SemiGPC, throughout this section we highlight the impact of SemiGPC on top of different underlying algorithms and/or pre-training strategies.

Table 5. Top1 Accuracy obtained on CIFAR10 and CIFAR100 for different numbers of labeled samples. $*$: reported by [31, 32].

Dataset	**Model**	n_{lb}	**Top1 Acc**
CIFAR100	USB [31]$*$	200	79.15
CIFAR100	SimMatch	200	79.18
CIFAR100	w/ SemiGPC	200	**80.01** **(+0.83)**
CIFAR100	USB [31]$*$	400	83.20
CIFAR100	SimMatch	400	83.25
CIFAR100	w/ SemiGPC	400	**83.87** **(+0.62)**
CIFAR10	FreeMatch [32]$*$	40	95.10
CIFAR10	SimMatch	40	**97.32**
CIFAR10	w/ SemiGPC	40	97.14 **(-0.18)**
CIFAR10	FreeMatch [32]$*$	250	95.12
CIFAR10	SimMatch	250	97.21
CIFAR10	w/ SemiGPC	250	**97.39** **(-0.18)**

Table 6. Comparison of the Top1 Accuracy on SemiAves when using different Semi-supervised algorithms.

Model	**Dataset**	**Top1 Acc**
FreeMatch	SemiAves	66.97
w/ SemiGPC	SemiAves	**67.93** **(+0.96)**
FixMatch	SemiAves	67.36
w/ SemiGPC	SemiAves	**68.31** **(+0.95)**
ReMixMatch	SemiAves	67.9
w/ SemiGPC	SemiAves	**68.31** **(+0.41)**
SimMatch	SemiAves	68.45
w/ SemiGPC	SemiAves	**69.30** **(+0.85)**

6.1. Semi-Supervised Learning Algorithms

The design of SemiGPC is agnostic to the underlying choice of the semi-supervised algorithms. In this section, we evaluate the impact of our proposed GP-based classifier on different Semi-Supervised methods including FixMatch [23], ReMixMatch [4], SimMatch [34] and FreeMatch [32] on the SemiAves benchmark. The obtained results are reported in Table 6. Despite FreeMatch [32] being designed to better tackle class imbalance, we observe that SimMatch [34] outperforms it when using the USB [31] training recipe. This justifies why we use SimMatch as our baseline throughout this work. Not only does SemiGPC improve performance across all considered methods, but its performance also improves monotonically with respect to the performance of the base method. This allows SemiGPC to remain relevant to future better semi-supervised algorithms.

6.2. Pre-training Strategy

As stated in section 4.1, we used a pre-trained ViT [9] to initialize our semi-supervised models. We evaluate the impact of SemiGPC across different pre-training strategies by training SimMatch on the SemiAves benchmark using supervised pre-training, Dino [5] and MSN [2] pre-training on ImageNet [7]. Both Dino [5] and MSN [2] are self-supervised methods that produce competitive performance on ImageNet with MSN being the top performer out of the two. We report the obtained results in Table 7. Not only does the SemiGPC performance scale based on the performance of the pre-training methods, it also improves performance across all considered pre-training strategies.

Table 7. Comparison of the Top1 Accuracy on SemiAves when using different pre-training strategies.

Model	**Pretraining**	**Top1 Acc**
SimMatch	DINO	64
w/ SemiGPC	DINO	**65.32** **(+1.32)**
SimMatch	MSN	64.7
w/ SemiGPC	MSN	**67.73** **(+3.03)**
SimMatch	Supervised	68.45
w/ SemiGPC	Supervised	**69.30** **(+0.85)**

7. Conclusion and Future Work

Our method SemiGPC is able to achieve state of the art results across different benchmarks and settings thanks to its ability to counteract imbalances in the data distribution. However, SemiGPC still has a few limitations. We observe in Tables 3 and 5 that SemiGPC shows mixed results when used on top of an already strong baseline ($> 94\%$ accuracy) such as on CIFAR10. Also, although our update rule greatly speeds up the matrix inversion, it does not fully eliminate the additional computational overhead. Furthermore, the quadratic scaling of the memory cost of this matrix limits the maximum size of the buffer in SemiGPC to around $N_Q = 16K$. However, this limitation can be addressed using an ensemble of GPs each using a separate buffer. This would allow us to scale SemiGPC to $N_Q \sim 80K$. Furthermore, our update rule is not compatible with using trainable kernel hyper-parameters since it relies on reusing previous values of the kernel matrix. Leveraging matrix-vector-matrix solvers [30] fixes both these limitations. Indeed, by enabling efficient GP inference with trainable hyper-parameters, SemiGPC would forgo sharing the kernel hyper-parameters across classes and adapt the geometry induced by the kernel function on a per-class basis. We leave deriving an online update rule using matrix-vector-matrix solvers to future work. Lastly, combining SemiGPC with alternative definitions of confidence to eq. (1) by either using the sample-wise posterior covariance provided by GP or by leveraging recent advances in efficient Neural Tangent Kernel (NTK) computation [20, 35] remains an open area of research.

References

[1] 2018 fgvcx fungi classification challenge. https://github.com/visipedia/fgvcx_fungi_comp. 6

[2] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pages 456–473. Springer, 2022. 1, 2, 8

[3] D Bernstein. Matrix mathematics (princeton university press. page 45, 2005. 5

[4] David Berthelot, Nicholas Carlini, Ekin Dogus Cubuk, Alexey Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020. 1, 2, 3, 8

[5] 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*, pages 9650–9660, 2021. 1, 2, 8

[6] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019. 2

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5, 8

[8] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 8

[10] Vincent Dutordoir, James Hensman, Mark van der Wilk, Carl Henrik Ek, Zoubin Ghahramani, and Nicolas Durrande. Deep neural networks as point estimates for deep gaussian processes. In *NeurIPS*, 2021. 2

[11] Yue Fan, Dengxin Dai, Anna Kukleva, and Bernt Schiele. Cossl: Co-learning of representation and classifier for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14574–14584, 2022. 6, 7

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016. 2

[13] iNaturalist 2018 competition dataset. iNaturalist 2018 competition dataset. https://github.com/visipedia/inat_comp/tree/master/2018, 2018. 6

[14] Joakim Johnander, Johan Edstedt, Michael Felsberg, Fahad Shahbaz Khan, and Martin Danelljan. Dense gaussian processes for few-shot segmentation. In *ECCV*, 2022. 2

[15] Neil Lawrence and Michael Jordan. Semi-supervised learning via gaussian processes. *Advances in neural information processing systems*, 17, 2004. 2

[16] Jongseok Lee, Jianxiang Feng, Matthias Humt, Marcus Gerhard Müller, and Rudolph Triebel. Trust your robots! predictive uncertainty estimation of neural networks with sparse gaussian processes. In *CoRL*, 2022. 2

[17] Junnan Li, Caiming Xiong, and Steven CH Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9475–9484, 2021. 1, 2, 3, 4

[18] Zhao-Yang Liu, Shao-Yuan Li, Songcan Chen, Yao Hu, and Sheng-Jun Huang. Uncertainty aware graph gaussian process for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4957–4964, 2020. 2

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[20] Roman Novak, Jascha Sohl-Dickstein, and Samuel S Schoenholz. Fast finite width neural tangent kernel. In *International Conference on Machine Learning*, pages 17018–17044. PMLR, 2022. 8

[21] Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*. Springer, 2006. 4

[22] Vikas Sindhwani, Wei Chu, and S Sathiya Keerthi. Semi-supervised gaussian process classifiers. In *IJCAI*, pages 1059–1064, 2007. 2

[23] 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. 1, 2, 3, 8

[24] Jong-Chyi Su and Subhransu Maji. The semi-supervised inaturalist-aves challenge at fgvc7 workshop, 2021. 6

[25] Jong-Chyi Su and Subhransu Maji. The semi-supervised inaturalist challenge at the fgvc8 workshop, 2021. 6

[26] Jong-Chyi Su and Subhransu Maji. Semi-supervised learning with taxonomic labels. In *British Machine Vision Conference (BMVC), 2021*, 2021. 1, 2, 5, 6, 7

[27] Jong-Chyi Su, Zezhou Cheng, and Subhransu Maji. A realistic evaluation of semi-supervised learning for fine-grained classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12966–12975, 2021. 1, 2, 6, 7

[28] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6

[29] Jianfeng Wang, Thomas Lukasiewicz, Daniela Massiceti, Xiaolin Hu, Vladimir Pavlovic, and Alexandros Neophytou.

Np-match: When neural processes meet semi-supervised learning. In *International Conference on Machine Learning*, pages 22919–22934. PMLR, 2022. 2

[30] Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact gaussian processes on a million data points. *Advances in neural information processing systems*, 32, 2019. 8

[31] Yidong Wang, Hao Chen, Yue Fan, Wang Sun, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, Heli Qi, Zhen Wu, Yu-Feng Li, Satoshi Nakamura, Wei Ye, Marios Savvides, Bhiksha Raj, Takahiro Shinozaki, Bernt Schiele, Jindong Wang, Xing Xie, and Yue Zhang. Usb: A unified semi-supervised learning benchmark for classification. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 5, 6, 7, 8

[32] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Yue Fan, , Zhen Wu, Jindong Wang, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Bernt Schiele, and Xing Xie. Freematch: Self-adaptive thresholding for semi-supervised learning. 2023. 1, 2, 3, 7, 8

[33] Greg Yang. Wide feedforward or recurrent neural networks of any architecture are gaussian processes. In *NeurIPS*, 2019. 2

[34] Mingkai Zheng, Shan You, Lang Huang, Fei Wang, Chen Qian, and Chang Xu. Simmatch: Semi-supervised learning with similarity matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14471–14481, 2022. 1, 2, 3, 4, 8

[35] Yufan Zhou and Zhenyi Wang. Meta-learning with neural tangent kernels. In *The International Conference on Learning Representations (ICLR)*, 2021. 8