Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm CVPR 2023

Abstract.

Previous research : unsupervised pretraining + supervised finetuning, but finetuning annotation budget 에는 관심없음.

Ours : 우리는 finetuning annotation budget 에 관심있고, 그래서 active finetuning task 를 새로 공식적으로 정의하겠음.
Active finetuning task : selection of samples for annotation in the pretraining-finetuning paradigm.
ActiveFT : 우리가 제안하는 active finetuning task 를 위한 새로운 방법 :
select a subset of data distributing similarly with the entire unlabeled pool and maintaining enough diversity by optimizing a parametric model in the continuous space. (We prove that the Earth Mover's distance between the distributions of the selected subset and the entire data pool is also reduced in this process.)

Extensive experiments show the leading performance and high efficiency of ActiveFT superior to baselines on both image classfication and semantic segmentation.

1. Introduction.

Expensive annotation budget problem inspires a popular pretraining-finetuning paradigm where models are pretrained on a large amount of data in an unsupervised manner and finetuned on a small labeled subset.

Existing literature : unsupervised pretraining + supervised finetuning, but *these researches build upon an unrealistic assumption that we already know which samples should be labeled.*

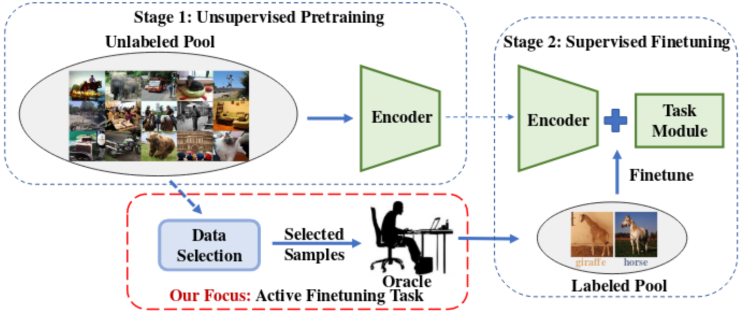


Figure 1. **Pretraining-Finetuning Paradigm:** We focus on the selection strategy of a small subset from a large unlabeled data pool for annotation, named as active finetuning task, which is under-explored for a long time.

전유진

Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm

CVPR 2023

Pretraining-finetuning paradigm 에 기존의 active learning algorithm 적용시 실패 (Sec 4.1) 하는데 그 이유로 현재 대부분 active learning methods에서 사용하는 batch-selection strategy를 꼽을 수 있음.
Starting from a random initial set, active learning algorithm repeats the model training and data selection processes multiple times until the annotation budget runs out. Despite their success in from-scratch training, it does not fit this pretraining-finetuning paradigm well due to the typically low annotation budget, where too few samples in each batch lead to harmful bias inside the selection process.

To fill in this gap in the pretraining-finetuning paradigm, we formulate a new task called active finetuning, concentrating on the sample selection for supervised finetuning. In this paper, a novel method, ActiveFT, is proposed to deal with this task. Starting from purely unlabeled data, ActiveFT fetches a proper data subset for supervised finetuning in a negligible time. Without any redundant heuristics, we directly bring close the distributions between the selected subset and the entire unlabeled pool while ensuring the diversity of the selected subset. This goal is achieved by continuous optimization in the high-dimensional feature space, which is mapped with the pretrained model.

We design a parametric model p_{θ_S} *to estimate the distribution of the selected subset. Its parameter* θ_S *is exactly the high-dimensional features of those selected samples. We optimize this model via gradient descent by minimizing our designed loss function. Unlike traditional active learning algorithms, our method can select all the samples from scratch in a single-pass without iterative batch-selections. We also mathematically show that the optimization in the continuous space can exactly reduce the earth mover's distance (EMD) [36, 37] between the entire pool and selected subset in the discrete data sample space.*

Extensive experiments are conducted to evaluate our method in the pretraining-finetuning paradigm. After pretraining the model on ImageNet-1k [38], we select subsets of data from CIFAR-10, CIFAR-100 [23], and ImageNet1k [38] for image classification, as well as ADE20k [51] for semantic segmentation. Results show the significant performance gain of our ActiveFT in comparison with baselines.

Contributions:
- *To our best knowledge, we are the first to identify the gap of data selection for annotation and supervised finetuning in the pretraining-finetuning paradigm, which can cause inefficient use of annotation budgets as also verified in our empirical study. Meanwhile, we formulate a new task called active finetuning to fill in this gap.*
- *We propose a novel method, ActiveFT, to deal with the active finetuning task through parametric model optimization which theoretically reduces the earth mover's distance (EMD) between the distributions of the selected subset and entire unlabeled pool. To our best knowledge, we are the first to directly optimize samples to be selected in the continuous space for data selection tasks.*
- *We apply ActiveFT to popular public datasets, achieving leading performance on both classification and segmentation tasks. In particular, our ablation study results justify the design of our method to fill in the data selection gap in the pretraining-finetuning paradigm. The source code will be made public available.*

2. Related work

Unsupervised learning

For both kinds of methods, prior research has well investigated their positive roles in downstream supervised finetuning. Of particular interest, they can bring significant performance gain in semi-supervised learning settings, where only a small part (e.g. 1%) of data samples are annotated.

Active learning

Most above active learning algorithms are designed for from-scratch training. Prior research [4] reveals their negative effect in finetuning after unsupervised pretraining.

3. Methodology

We first formulate this new task called active finetuning in Sec. 3.1. Our novel method, ActiveFT, to solve this problem based on continuous space optimization is proposed in Sec. 3.2. Afterward, we elaborate on how to optimize this model by minimizing the loss function in Sec. 3.3. An illustration of our method is shown in Fig. 2. We also clarify the correlation between our method and earth mover's distance in Sec. 3.4. Finally, the implementation of this method to deep learning model is explained in Sec. 3.5.

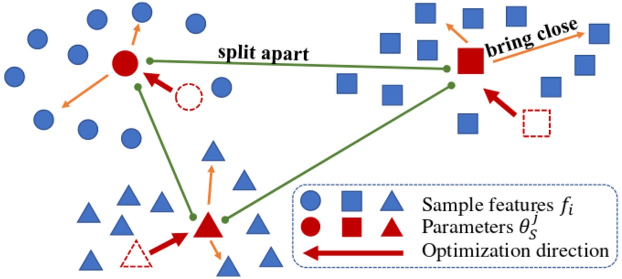


Figure 2. **Parametric Model Optimization Process:** By optimizing the loss in Eq. 11, each parameter θ_S^j is appealed by nearby sample features (orange in the figure, Eq. 9) and repelled by other parameters $\theta_S^k, k \neq j$ (green in the figure, Eq. 10).

Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm

3.1. Formulation of Active Finetuning Task

$f(\cdot; w_0) : \mathcal{X} \rightarrow \mathbb{R}^C$: a deep neural network model with pretrained weight w_0

$\mathcal{X}$: data space

$\mathbb{R}^C$: normalized high dimensional feature space

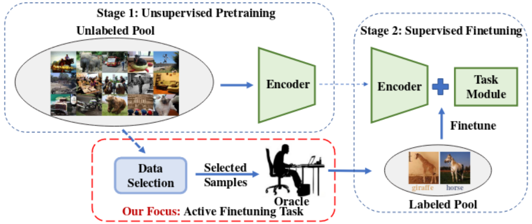


Figure 1. **Pretraining-Finetuning Paradigm:** We focus on the selection strategy of a small subset from a large unlabeled data pool for annotation, named as active finetuning task, which is under-explored for a long time.

$\mathcal{P}^u = \{\mathbf{x}_i\}_{i\in[N]} \sim p_u$: large unlabeled data pool inside data space $\mathcal{X}$ with distribution p_u

$[N] = \{1, 2, \ldots, N\}$

$\mathcal{P}^u_{\mathcal{S}}$: subset for supervised finetuning is selected from $\mathcal{P}^u$

$f(\cdot; w_0)$ can be pretrained either on $\mathcal{P}^u$ or other data sources, e.g. pretrained on ImageNet1k [38] and finetuned on a subset of CIFAR-10 [23].

$\mathcal{S} = \{s_j \in [N]\}_{j\in[B]}$: sampling strategy to select a subset $\mathcal{P}^u_{\mathcal{S}} = \{\mathbf{x}_{s_j}\}_{j\in[B]} \subset \mathcal{P}^u$

B : annotation budget size for supervised finetuning

The model would have access to the labels $\{\mathbf{y}_{s_j}\}_{j\in[B]} \subset \mathcal{Y}$ of this subset through the oracle

, obtaining a labeled data pool $\mathcal{P}^l_{\mathcal{S}} = \{\mathbf{x}_{s_j}, \mathbf{y}_{s_j}\}_{j\in[B]}$

$\mathcal{Y}$: label space

전유진

Afterward, the model f is finetuned on $\mathcal{P}^l_{\mathcal{S}}$ supervisedly and the model parameter is updated to $w_{\mathcal{S}}$ after the finetuning.

The goal of active finetuning is to find the sampling strategy $\mathcal{S}_{opt}$ minimizing the expected model error $error(f(\mathbf{x}; w_{\mathcal{S}}), \mathbf{y})$

$$\mathcal{S}_{opt} = \arg\min_{\mathcal{S}} \underset{\mathbf{x},\mathbf{y}\in\mathcal{X}\times\mathcal{Y}}{E} [error(f(\mathbf{x}; w_{\mathcal{S}}), \mathbf{y})] \quad (1)$$

- Our active finetuning 의 traditional active learning 과의 차이점

1) We have access to the pretrained model $f(\cdot; w_0)$, which will be finetuned, before data selection.

2) The selected samples are applied to the finetuning of the pretrained model $f(\cdot; w_0)$ instead of from-scratch training.

3) The sampled subset size $|\mathcal{P}^l_{\mathcal{S}}|$ is relatively small, less than 10% in most cases.

4) We have no access to any labels such as a random initial labeled set before data selection.

3.2. Data Selection with Parametric Model

-샘플 선택의 바탕이 되는 2가지 기본 직관

1) bringing close the distributions between the selected subset $\mathcal{P}^u_{\mathcal{S}}$ and the original pool $\mathcal{P}^u \sim p_u$

2) maintaining the diversity of $\mathcal{P}^u_{\mathcal{S}}$

1) 의 효과 : model finetuned on the subset performs similarly with that trained on the full set
2) 의 효과 : allow the subset to cover corner cases in the full set

Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm CVPR 2023

In comparison to distribution $p_u(\mathbf{x})$ in the data space, it is more feasible to work on its corresponding distribution $p_{f_u}(\mathbf{f})$ in the feature space.

Through the agency of pretrained model $f(\cdot; w_0)$, we map each data sample $\mathbf{x}_i$ to the high dimensional feature space as $\mathbf{f}_i = f(\mathbf{x}_i; w_0)$

$\mathbf{f}_i$: normalized feature of $\mathbf{x}_i$

As a result, we can derive the pool $\mathcal{F}^u = \{\mathbf{f}_i\}_{i\in[N]}$ from $\mathcal{P}^u$ and corresponding distribution p_{f_u} of $\mathcal{F}^u$

Similarly, $\mathcal{F}^u_{\mathcal{S}}$: feature pool of selected data subset $\mathcal{P}^u_{\mathcal{S}}$ and corresponding distribution $p_{f_{\mathcal{S}}}$ of $\mathcal{F}^u_{\mathcal{S}}$

Our goal is to find the optimal selection strategy

$$\mathcal{S}_{opt} = \arg\min_{\mathcal{S}} D(p_{f_u}, p_{f_{\mathcal{S}}}) - \lambda R(\mathcal{F}^u_{\mathcal{S}}) \qquad (2)$$

$D(\cdot, \cdot)$: some distance metrics between distributions

$R(\cdot)$: measure the diversity of a set

λ : a scale to balance these two terms

첫번째 항 : aims to bring close these two distributions $p_{f_u}, p_{f_{\mathcal{S}}}$

두번째 항 : to ensure the diversity of subset

Unfortunately, it is difficult to directly optimize the discrete selection strategy $\mathcal{S}$, so we alternatively model $p_{f_{\mathcal{S}}}$ with $p_{\theta_{\mathcal{S}}}$

$\theta_{\mathcal{S}} = \{\theta_{\mathcal{S}}^{j}\}_{j \in [B]}$: continuous parameters

B : annotation budget size

Each $\theta_{\mathcal{S}}^{j}$ after optimization corresponds to the feature of a selected sample $\mathbf{f}_{s_j}$

We would find $\mathbf{f}_{s_j}$ closest to each $\theta_{\mathcal{S}}^{j}$ after optimization to determine the selection strategy $\mathcal{S}$

Therefore, our goal in Eq. 2 is written as follows.

$$\mathcal{S}_{opt} = \arg\min_{\mathcal{S}} D(p_{f_u}, p_{f_{\mathcal{S}}}) - \lambda R(\mathcal{F}_{\mathcal{S}}^{u}) \tag{2}$$

$$\theta_{\mathcal{S},opt} = \arg\min_{\theta_{\mathcal{S}}} D(p_{f_u}, p_{\theta_{\mathcal{S}}}) - \lambda R(\theta_{\mathcal{S}}) \ s.t.\ ||\theta_{\mathcal{S}}^{j}||_2 = 1 \tag{3}$$

The difference between extracted sample features $\mathcal{F}_{\mathcal{S}}^{u} = \{\mathbf{f}_{s_i}\}$ *and our define parameters* $\theta_{\mathcal{S}} = \{\theta_{\mathcal{S}}^{j}\}$ *is that* $\mathbf{f}_{s_i}$ *is a discrete feature corresponding to a sample in the dataset while* $\theta_{\mathcal{S}}^{j}$ *is continuous in the feature space.*

3.3. Parametric Model Optimization

In the parametric model p_{θ_S} , the distribution is represented by B parameters $\{\theta_S^j\}_{j\in[B]}$

- mixture model with B components

$$p_{\theta_S}(\mathbf{f}) = \sum_{j=1}^{B} \phi_j p(\mathbf{f}|\theta_S^j) \qquad (4)$$

ϕ_j : mixture weight or prior probability $p(\theta_S^j)$ of the j-th component $\sum_{j=1}^{B} \phi_j = 1$

distribution of each component is formulated based on their similarity as Eq. 5, since $\mathbf{f}$ and θ_S^j both lie in the feature space

$$p(\mathbf{f}|\theta_S^j) = \frac{\exp(sim(\mathbf{f}, \theta_S^j)/\tau)}{Z_j} \qquad (5)$$

Z_j : normalizing constant

$sim(\cdot,\cdot)$: similarity metric

τ : temperature scale

전유진

We follow the protocol in [6, 47] to apply the cosine similarity between normalized features,

as the metric $sim(\mathbf{f}_1, \mathbf{f}_2) = \mathbf{f}_1^\top \mathbf{f}_2, ||\mathbf{f}_1||_2 = ||\mathbf{f}_2||_2 = 1$ *and set the temperature τ = 0.07 [6,47] throughout the paper.*

For each $\mathbf{f}_i \in \mathcal{F}^u$, there exists a $\theta_{\mathcal{S}}^{c_i}$ most similar (and closest) to $\mathbf{f}_i$ where we keep updating c_i in the optimization process.

$$c_i = \arg\max_{j \in [B]} sim(\mathbf{f}_i, \theta_{\mathcal{S}}^{j}) \qquad (6)$$

Since there is a very low temperature (τ = 0.07), the gap between the exponential similarity $\exp(sim(\mathbf{f}_i, \theta_{\mathbf{S}}^{j})/\tau)$ *with different* $\theta_{\mathbf{S}}^{j}$ *is significant.*

Assumption 1

$\forall i \in [N], j \in [B],$ if τ is small, the following far-more-than relationship holds that

$$\exp(sim(\mathbf{f}_i, \theta_{\mathcal{S}}^{c_i})/\tau) \gg \exp(sim(\mathbf{f}_i, \theta_{\mathcal{S}}^{j})/\tau), j \neq c_i$$

When the optimization is finished, we find feature $\{\mathbf{f}_{s_j}\}_{j \in [B]}$ with the highest similarity to $\theta_{\mathcal{S}}^{j}$

$$\mathbf{f}_{s_j} = \arg\max_{\mathbf{f}_k \in \mathcal{F}^u} sim(\mathbf{f}_k, \theta_{\mathcal{S}}^{j}) \qquad (12)$$

The corresponding data samples $\{\mathbf{x}_{s_j}\}_{j \in [B]}$ are selected as the subset $\mathcal{P}_{\mathcal{S}}^{u}$ with selection strategy $\mathcal{S} = \{s_j\}_{j \in [B]}$

3.4. Relation to Earth Mover's Distance

Optimizing the loss in Eq. 11 is actually minimizing the earth mover's distance between the distributions of selected subset and full set.
This justifies that our optimization in the continuous space is equivalent with bringing close the distribution gap in the discrete data sample space.

After the optimization, we get the features $\mathbf{f}_{s_j}$ *of selected samples.*

We deliberately assign the discrete probability distribution p_{f_S} *as Eq. 13.*

$$p_{f_S}(\mathbf{f}_{s_j}) = \frac{|C_j|}{N}, C_j = \{\mathbf{f}_i | c_i = j\}, \mathbf{f}_{s_j} \in \mathcal{F}^u_{\mathcal{S}} \qquad (13)$$

C_j : the set of features closest to f_{s_j} with c_i defined in Eq. 6

The distribution p_{f_u} is modeled as a uniform distribution over $\mathcal{F}^u$, i.e. $p_{f_u}(\mathbf{f}_i) = \frac{1}{N}, \mathbf{f}_i \in \mathcal{F}^u$

The earth mover's distance (EMD) between p_{f_u}, p_{f_S} is written as [24]:

$$EMD(p_{f_u}, p_{f_S}) = \underset{\gamma \in \Pi(p_{f_u}, p_{f_S})}{inf} \underset{(\mathbf{f}_i, \mathbf{f}_{s_j}) \sim \gamma}{E} [||\mathbf{f}_i - \mathbf{f}_{s_j}||_2] \qquad (14)$$

$\Pi(p_{f_u}, p_{f_S})$: the set of all possible joint distributions whose marginals are p_{f_u}, p_{f_S}

It is intuitive to come up with the infimum, i.e. each $\mathbf{f}_i \sim p_{f_u}$ transports to their closest $\mathbf{f}_{s_j} \sim p_{f_S}$

$$\gamma_{f_u,f_S}(\mathbf{f}_i,\mathbf{f}_{s_j}) = \begin{cases} \frac{1}{N} & \mathbf{f}_i \in \mathcal{F}^u, \mathbf{f}_{s_j} \in \mathcal{F}^u_{\mathcal{S}}, c_i = j \\ 0 & otherwise \end{cases} \tag{15}$$

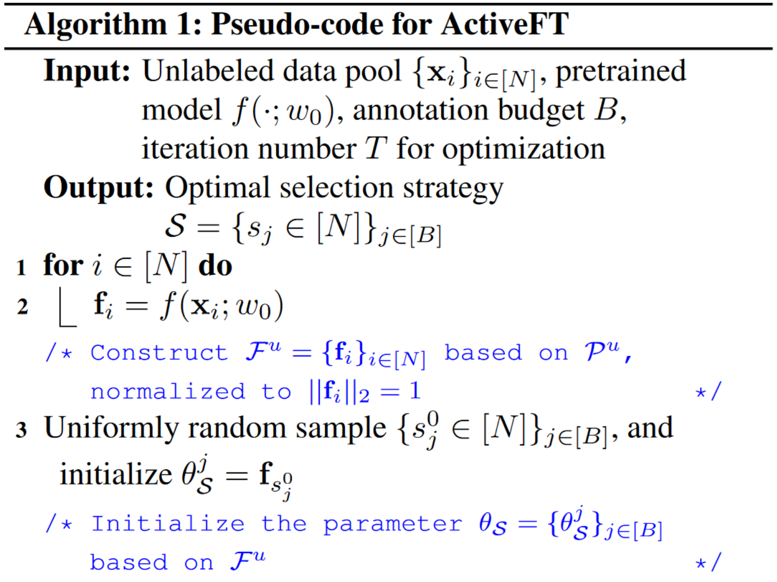
In this case, the distance in Eq. 14 becomes

$$EMD(p_{f_u},p_{f_S}) = \underset{\gamma\in\Pi(p_{f_u},p_{f_S})(\mathbf{f}_i,\mathbf{f}_{s_j})\sim\gamma}{inf \qquad\qquad E} [||\mathbf{f}_i - \mathbf{f}_{s_j}||_2] \tag{14}$$

$$\begin{aligned} EMD(p_{f_u},p_{f_S}) &= \underset{(\mathbf{f}_i,\mathbf{f}_{s_{c_i}})\sim\gamma}{E} [||\mathbf{f}_i - \mathbf{f}_{s_{c_i}}||_2] \\ &= \frac{1}{N}\sum_{i=1}^{N}\left[\sqrt{2 - 2sim(\mathbf{f}_i,\mathbf{f}_{s_{c_i}})}\right] \end{aligned} \tag{16}$$

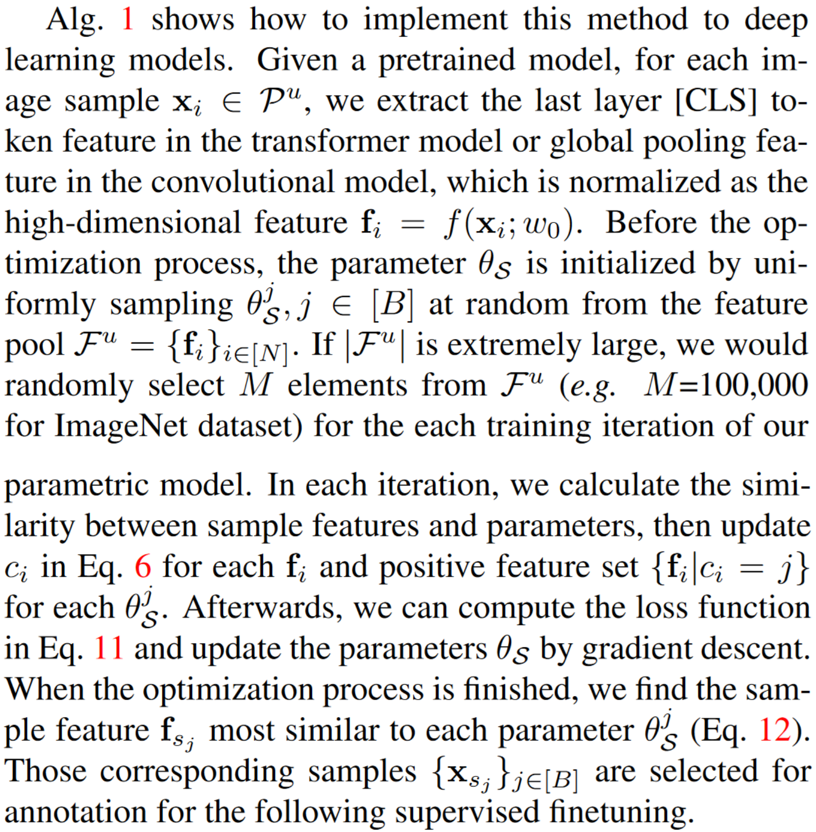
Therefore, our optimization method in Sec. 3.3 is equivalent with reducing the earth mover's distance between the distributions of the original unlabeled pool and selected subset.

3.5. Implementation as a Learning Model



Algorithm 1: Pseudo-code for ActiveFT

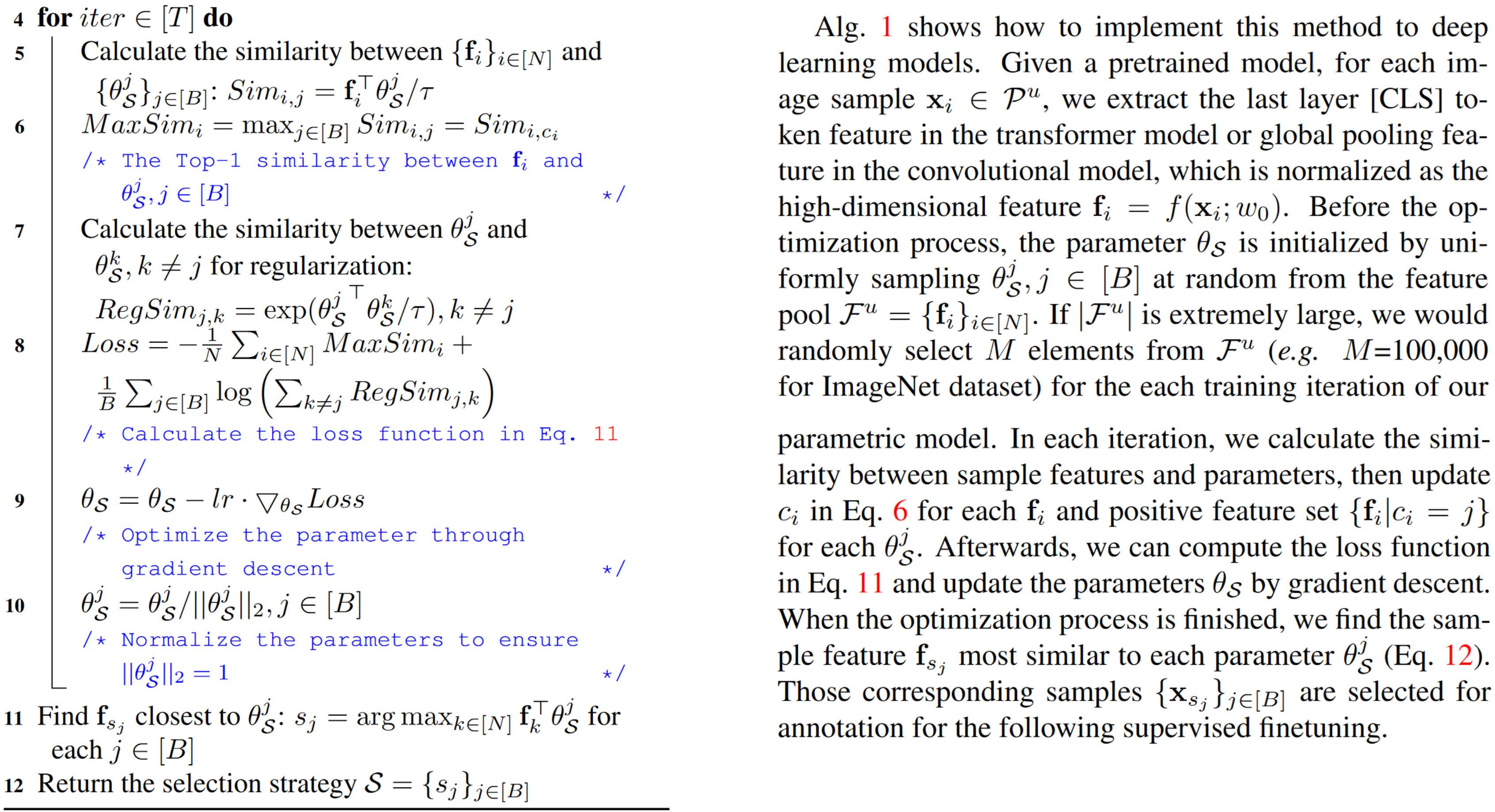
```
Input: Unlabeled data pool $\{\mathbf{x}_i\}_{i\in[N]}$, pretrained
       model $f(\cdot;w_0)$, annotation budget $B$,
       iteration number $T$ for optimization
Output: Optimal selection strategy
        $\mathcal{S}=\{s_j\in[N]\}_{j\in[B]}$
for $i\in[N]$ do
    $\mathbf{f}_i=f(\mathbf{x}_i;w_0)$
/* Construct $\mathcal{F}^u=\{\mathbf{f}_i\}_{i\in[N]}$ based on $\mathcal{P}^u$,
   normalized to $||\mathbf{f}_i||_2=1$                          */
Uniformly random sample $\{s_j^0\in[N]\}_{j\in[B]}$, and
 initialize $\theta_\mathcal{S}^j=\mathbf{f}_{s_j^0}$
/* Initialize the parameter $\theta_\mathcal{S}=\{\theta_\mathcal{S}^j\}_{j\in[B]}$
   based on $\mathcal{F}^u$                                      */
```



Alg. 1 shows how to implement this method to deep learning models. Given a pretrained model, for each image sample $\mathbf{x}_i\in\mathcal{P}^u$, we extract the last layer [CLS] token feature in the transformer model or global pooling feature in the convolutional model, which is normalized as the high-dimensional feature $\mathbf{f}_i=f(\mathbf{x}_i;w_0)$. Before the optimization process, the parameter $\theta_\mathcal{S}$ is initialized by uniformly sampling $\theta_\mathcal{S}^j, j\in[B]$ at random from the feature pool $\mathcal{F}^u=\{\mathbf{f}_i\}_{i\in[N]}$. If $|\mathcal{F}^u|$ is extremely large, we would randomly select M elements from $\mathcal{F}^u$ (*e.g.* M=100,000 for ImageNet dataset) for the each training iteration of our parametric model. In each iteration, we calculate the similarity between sample features and parameters, then update c_i in Eq. 6 for each $\mathbf{f}_i$ and positive feature set $\{\mathbf{f}_i|c_i=j\}$ for each $\theta_\mathcal{S}^j$. Afterwards, we can compute the loss function in Eq. 11 and update the parameters $\theta_\mathcal{S}$ by gradient descent. When the optimization process is finished, we find the sample feature $\mathbf{f}_{s_j}$ most similar to each parameter $\theta_\mathcal{S}^j$ (Eq. 12). Those corresponding samples $\{\mathbf{x}_{s_j}\}_{j\in[B]}$ are selected for annotation for the following supervised finetuning.

Active Finetuning: Exploiting Annotation Budget in the Pretraining-Finetuning Paradigm

CVPR 2023



```
for $iter \in [T]$ do
    Calculate the similarity between $\{\mathbf{f}_i\}_{i\in[N]}$ and
      $\{\theta^j_\mathcal{S}\}_{j\in[B]}$: $Sim_{i,j} = \mathbf{f}_i^\top \theta^j_\mathcal{S} / \tau$
    $MaxSim_i = \max_{j\in[B]} Sim_{i,j} = Sim_{i,c_i}$
    /* The Top-1 similarity between $\mathbf{f}_i$ and
       $\theta^j_\mathcal{S}, j \in [B]$                                   */
    Calculate the similarity between $\theta^j_\mathcal{S}$ and
      $\theta^k_\mathcal{S}, k \neq j$ for regularization:
      $RegSim_{j,k} = \exp(\theta^j_\mathcal{S}{}^\top \theta^k_\mathcal{S} / \tau), k \neq j$
    $Loss = -\frac{1}{N} \sum_{i\in[N]} MaxSim_i +$
      $\frac{1}{B} \sum_{j\in[B]} \log \left(\sum_{k\neq j} RegSim_{j,k}\right)$
    /* Calculate the loss function in Eq. 11
       */
    $\theta_\mathcal{S} = \theta_\mathcal{S} - lr \cdot \nabla_{\theta_\mathcal{S}} Loss$
    /* Optimize the parameter through
       gradient descent                                                      */
    $\theta^j_\mathcal{S} = \theta^j_\mathcal{S} / ||\theta^j_\mathcal{S}||_2, j \in [B]$
    /* Normalize the parameters to ensure
       $||\theta^j_\mathcal{S}||_2 = 1$                                         */
Find $\mathbf{f}_{s_j}$ closest to $\theta^j_\mathcal{S}$: $s_j = \arg\max_{k\in[N]} \mathbf{f}_k^\top \theta^j_\mathcal{S}$ for
  each $j \in [B]$
Return the selection strategy $\mathcal{S} = \{s_j\}_{j\in[B]}$
```

Alg. 1 shows how to implement this method to deep learning models. Given a pretrained model, for each image sample $\mathbf{x}_i \in \mathcal{P}^u$, we extract the last layer [CLS] token feature in the transformer model or global pooling feature in the convolutional model, which is normalized as the high-dimensional feature $\mathbf{f}_i = f(\mathbf{x}_i; w_0)$. Before the optimization process, the parameter $\theta_\mathcal{S}$ is initialized by uniformly sampling $\theta^j_\mathcal{S}, j \in [B]$ at random from the feature pool $\mathcal{F}^u = \{\mathbf{f}_i\}_{i\in[N]}$. If $|\mathcal{F}^u|$ is extremely large, we would randomly select M elements from $\mathcal{F}^u$ (*e.g.* M=100,000 for ImageNet dataset) for the each training iteration of our parametric model. In each iteration, we calculate the similarity between sample features and parameters, then update c_i in Eq. 6 for each $\mathbf{f}_i$ and positive feature set $\{\mathbf{f}_i | c_i = j\}$ for each $\theta^j_\mathcal{S}$. Afterwards, we can compute the loss function in Eq. 11 and update the parameters $\theta_\mathcal{S}$ by gradient descent. When the optimization process is finished, we find the sample feature $\mathbf{f}_{s_j}$ most similar to each parameter $\theta^j_\mathcal{S}$ (Eq. 12). Those corresponding samples $\{\mathbf{x}_{s_j}\}_{j\in[B]}$ are selected for annotation for the following supervised finetuning.

전유진