FINE-GRAINED IMAGE RETRIEVAL WITH NEIGHBOR-ATTENTION LABEL CORRECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper studies noise-resistant deep model training for the fine-grained image retrieval task, which has an unconstrained target label space and suffers from the difficulty of acquiring accurate fine-grained labels. A Neighbor-Attention Label Correction (NALC) model is proposed based on the meta-learning framework to correct labels during the training stage. A training batch and a validation batch are sampled from the training set, which hence allows to optimize the NALC model by referring to the validation batch. We also propose a novel nested optimization for the meta-learning framework to enhance the optimization efficiency. The training procedure consistently boosts the label accuracy in the training batch, which in turn ensures a more accurate training set. Experiments results show that our method boosts the label accuracy from 70% to 97+% and it outperforms recent works up to 11.5% in rank1 accuracy on various fine-grained image retrieval tasks, *e.g.*, fine-grained instance retrieval on CUB200 and CARS, as well as person re-identification, respectively. Ablation studies also show the NALC generalizes well on different types of noises, *e.g.*, Asymmetric, Pair-Flip, Pattern noises, *etc.*

1 INTRODUCTION

Fine-grained image retrieval aims to query images from the gallery with the same fine-grained label, *e.g.* person identities (Zheng et al., 2015; Wei et al., 2018), bird species (Wah et al., 2011), or car models (Krause et al., 2013), *etc*. It is more appealing than general image retrieval due to the capability of differentiating visually similar objects. Compared with image classification, it features an unconstrained target label space, and thus is expected to present better generalization capability on large-scale test sets. This task has attracted increasing attention in recent years. For instance, lots of efforts are conducted on instance re-identification (re-id) (Luo et al., 2019; He et al., 2020; Liu et al., 2019).

Most current fine-grained image retrieval works train deep models as feature extractors. Clean datasets can be hard to acquire in real scenarios, because of the difficulty in differentiating fine-grained labels. This issue has been noticed by the community, where many unsupervised training strategies are proposed (Wei et al., 2018; Zhong et al., 2019). Unsupervised methods have achieved significant performance gains (Xuan & Zhang, 2021; Chen et al., 2021). Fig. 1 shows the performance of supervised training, where 20% label noise leads to a lower performance than a recent unsupervised training method (Chen et al., 2021). Another category of works tends to eliminate noisy samples during training (Wang et al., 2019; Yu et al., 2019; Ye & Yuen, 2020; Zhang et al., 2021; Liu et al., 2021). Fig. 1 shows their performance upper bound, which outperforms unsupervised training till 40% noisy images are removed. However, the upper-bound drops substantially for higher noisy rates due to the lack of training data. It is hence appealing to study noise-resistant training to 1) leverage correct annotations, meanwhile 2) correct noisy labels.

Most current noise-resistant training research focuses on the classification task and can be summarized into two categories, *i.e.*, noise ignoring and noise correcting, respectively. Noisy ignoring reduces the influence of noisy samples by either explicitly discriminating noisy samples (Wu et al., 2020; Lee et al., 2018) or implicitly eliminating their interference (Li et al., 2020a; Han et al., 2018). Noise correction recovers noisy samples by either calibrating the training loss (Patrini et al., 2017; Hendrycks et al., 2018) or correcting the target labels (Zheng et al., 2021; Yi & Wu, 2019). Noise

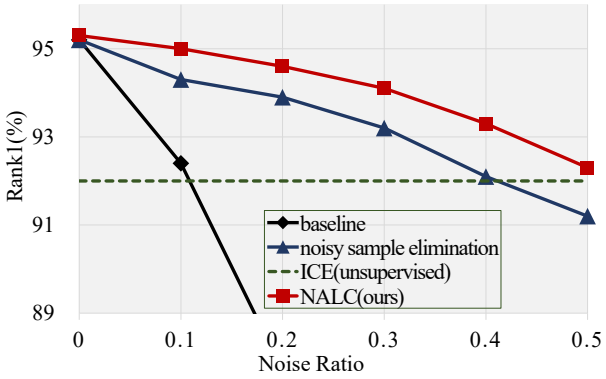


Figure 1: Comparison of rank-1 accuracy on person re-id dataset Market1501. **Baseline** trains ResNet50 on the noisy training set with various noise ratios. **ICE** (Chen et al., 2021) is a recent unsupervised method. **Noisy sample elimination** removes noisy labels referring to ground truth annotations. **NALC** is our method, which performs the best. See Sec. 6 for more details.

ignoring could deal with out-of-distribution noises. Noise correction could make better use of training data, especially when a large portion of noise exists. Sec. 2 presents a more detailed review.

Different from classification, retrieval commonly features an unconstrained target label space. This difference leads to the failure of existing noise-resistant training methods on retrieval tasks. As training and testing sets in retrieval do not share the same label space, it degrades the effectiveness of methods (Patrini et al., 2017; Hendrycks et al., 2018) which optimize the classifier layers. Methods (Zheng et al., 2021; Yi & Wu, 2019) which use parametric methods to recover the complete label space could be expensive to compute and hard to converge on a large number of training categories in retrieval. Moreover, due to the high annotation cost, lots of training categories in retrieval may contain a few samples. Too few samples lead to biased distribution (Li et al., 2019a) and degrade the effectiveness of noise ignoring methods (Wu et al., 2020; Lee et al., 2018).

This paper aims to correct labels for fine-grained image retrieval by referring to neighbor cues of each training sample. CNN pre-trained on large image classification datasets like ImageNet can initialize a reasonable feature space, which guarantees the reasonable and robust neighbor cues among sample features. It is also indicated that the memorization effect of CNN tends to ignore hard noisy labels during the initial training stage (Li et al., 2020a). We hence propose a Neighbor-Attention Label Correction (NALC) model to directly generate corrected labels in the label space. In other words, the corrected label of each sample is computed by referring to labels of its neighbors, a more efficient way than previous works that build a complicated mapping from feature space to label space.

The NALC model is end-to-end optimized in a meta-learning framework to chase better performance and generalization capability. We sample a training batch and a validation batch from the training data and optimize the NALC on the validation batch. As NALC trains a better feature extractor by correcting noisy labels, it in turn decreases the loss on validation batch, which makes nested optimization to NALC possible. Implicit function theorem (IFT) and Neumann approximation are used in this procedure. Former works (Lorraine et al., 2020; Gudovskiy et al., 2021) use a fixed hyperparameter α in Neumann approximation, which cannot adapt the variance of the Hessian matrix during the training stage. We propose the adaptive Neumann approximation according to the estimation of L2-norm, leading to a more stable and accurate nested optimization of NALC.

The proposed methods are evaluated on various fine-grained image retrieval datasets under different noise rates ranging from 0% to 50%. As shown in Fig. 1, NALC achieves promising performance, and outperforms the upper bound of noise elimination methods. NALC is capable of boosting label accuracy from 70% to 97.9% and enhances the rank1 accuracy from 82.3% to 93.8% on Market1501 dataset. Experiments on other datasets suggest similar conclusions. To the best of our knowledge, this is an original research on label correction for fine-grained image retrieval. It leverages neighbour cues to correct labels in the label space, which enjoys high efficiency and substantially outperforms previous noise-resistant fine-grained retrieval methods. Our proposed nested optimization algorithm also guarantees a stable and efficient optimization of NALC in the meta-learning framework.

2 RELATED WORK

This work is related to research on noise-resistant learning and Differentiable hyper-parameter optimization. This section briefly views those works.

Noise-resistant learning: Most noise-resistant methods are designed for classification. We divide them into two categories, *i.e.*, noise correction and noise ignoring. Former noise correction methods (Patrini et al., 2017; Hendrycks et al., 2018) correct the loss functions by multiplying the Sym-flipping transition matrix (Rooyen et al., 2015) to the original loss function. Some recent methods try to correct labels. PENCIL (Yi & Wu, 2019) and Joint Optimization (Tanaka et al., 2018) set corrected labels as parameters, and apply regularization between corrected loss and original loss to prevent collapsing. MLC (Zheng et al., 2021) and AutoDO (Gudovskiy et al., 2021) use meta-learning to supervise backbone and correction models with different data. Noise ignoring for classification includes global, class-based, and neighbour-based methods. Global methods directly takes use of the memorization effect. A deep model is more likely to learn from clean samples than from noisy ones in the early training stage (Li et al., 2020a). Class-based methods (Li et al., 2019a; Han et al., 2018; Li et al., 2019b; Ren et al., 2018) identify label noises according to the membership in their classes. Neighbour-based methods (Li et al., 2019b; Wu et al., 2020) use neighbor clues to identify label noise.

Some noise-resistant methods are designed for fine-grained image retrieval. Most of them work by ignoring noisy samples. PurifyNet (Ye & Yuen, 2020) uses a regularization term to refine the falsely annotated labels and fine-tunes the model with hard-aware instance re-weighting. OSM-CAA (Wang et al., 2019) trains a proxy for each class and adjusts the weight of outliers to eliminate their effects. PRISM (Liu et al., 2021) uses memory features of the same category to identify noisy samples. One4More (Zhang et al., 2021) learns a data sampler to reduce the sampling frequency on noisy samples. DNet (Yu et al., 2019) introduce variance to make noisy samples have less influence on the training process. Unsupervised learning is another way to deal with noisy labels. Related works fall into two categories: domain transfer and pseudo label based methods. Domain transfer either transfers images from the source domain to the target domain (Wei et al., 2018), or transfers images from the original camera style to other camera styles (Zhong et al., 2019). Pseudo label based methods (Ge et al., 2020a;b; Chen et al., 2021) use clustering to generate labels for training.

Differentiable hyper-parameter optimization: Our NALC affects the feature extractor by correcting labels, which in turn changes the validation loss. The optimization to NALC can be achieved by a nested optimization, which is widely used in AutoML (Liu et al., 2018; Li et al., 2020b; Gudovskiy et al., 2021). Hyper-parameter can be updated after several iterations or after several epochs. The former (Zheng et al., 2021; Shu et al., 2019) computes gradients of hyper-parameters more easily, but degrades the training stability. The latter (Lorraine et al., 2020; Gudovskiy et al., 2021; Bi et al., 2019) is more stable, but requires an inverse Hessian matrix for gradient computation, which is expensive to compute. Some works (Lorraine et al., 2020; Gudovskiy et al., 2021) use Neumann series to approximate it.

Relationship with previous works: This method trains a fine-grained image retrieval model by 1) leveraging correct labels and 2) correcting noisy ones. It differs from previous noise-resistant retrieval methods and unsupervised methods, which mostly ignore noisy labels, or correct labels. The MLC (Zheng et al., 2021) and neighbour-based methods (Li et al., 2019b; Wu et al., 2020) designed for image classification are related to our work. As the fine-grained retrieval task involves a larger category number, NALC uses neighbour features rather than label embeddings as the input, and introduces attention parameters to compute neighbour cues. Experiments with different setups show the promising performance of NALC.

3 PROBLEM STATEMENT

Given a training set $\mathcal{T}$ containing images and their labels, fine-grained image retrieval aims to train a backbone model $\mathrm{F}(\cdot;\theta)$, where θ denotes learnable parameters. For any query image x_q, the backbone model is expected to produce a feature vector $f_q = \mathrm{F}(x_q;\theta)$ to retrieve the gallery image x_g having the same label with x_q from a gallery set $\mathcal{G}$. The backbone model should be optimized to guarantee features of x_q and x_g to be more similar than other image pairs. The training objective of

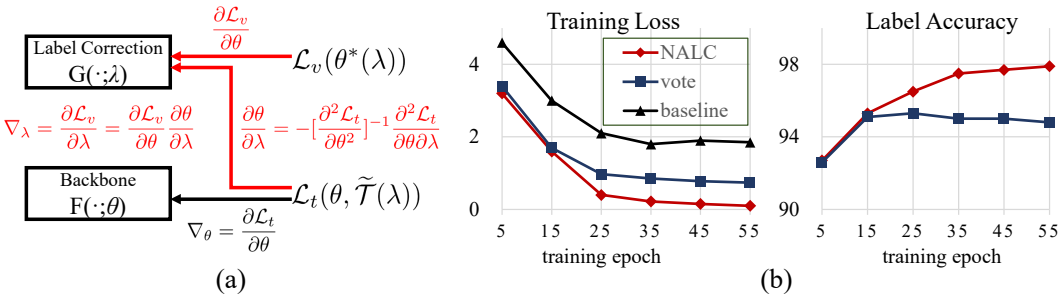


Figure 2: (a): Illustration of data flow to compute the gradients of label correction model $\mathrm{G}(\cdot)$ (red arrows) and backbone $\mathrm{F}(\cdot)$ (black arrows). (b): Training loss and corrected label accuracy at different training epochs on Market1501 with noise ratio=30%.

fine-grained image retrieval can be conceptually denoted as,

$$\theta^* = \arg\min_\theta(\mathrm{dist}(f_q, f_g) - \mathrm{dist}(f_q, f_i)), x_i \in \mathcal{G}, i \neq g, \tag{1}$$

where $\mathrm{dist}(\cdot)$ is the distance metric, *e.g.*, the L2 distance, f_g and f_i are features of gallery images.

Given a training set $\mathcal{T}$ containing images and their labels, fine-grained image retrieval aims to train a backbone model $\mathrm{F}(\cdot;\theta)$, where θ denotes learnable parameters. The general training objective is to During training, labels and images in $\mathcal{G}$ are commonly unknown. We reasonably assume that $\mathcal{T}$ and $\mathcal{G}$ present similar feature distributions and optimize θ by assuming $\mathcal{T} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ as the gallery set, where n denotes image number and image label $y_i \in \{0, 1\}^C$ is a C-dim one-hot vector if $\mathcal{T}$ contains C classes. The general training loss function on training set, *i.e.*, $\mathcal{L}_t(\theta, \mathcal{T})$ can be denoted as

$$\theta^* = \arg\min_\theta \mathcal{L}_t(\theta, \mathcal{T}). \tag{2}$$

Eq. (2) is commonly used in previous retrieval works, where $\mathcal{T}$ is accurately annotated. For the case with noisy labels, we denote the training set as $\bar{\mathcal{T}} = \{(x_1, \bar{y}_1), (x_2, \bar{y}_2), \cdots, (x_n, \bar{y}_n)\}$, where the annotated label $\bar{y}_i$ may differ from the correct label y_i. As shown in Fig. 1, noisy labels substantially degrade the performance of supervised training as in Eq. (2).

This work aims to train a label correction model $\mathrm{G}(\cdot;\lambda)$ with parameters λ to generate corrected labels for each training sample, *i.e.*, $\tilde{y}_i = \mathrm{G}(i;\lambda)$. It produces a corrected training set $\widetilde{\mathcal{T}}$, which hence replaces the original $\bar{\mathcal{T}}$ for supervised training. The training objective of the backbone can be denoted as,

$$\theta^* = \arg\min_\theta \mathcal{L}_t(\theta, \widetilde{\mathcal{T}}(\lambda)), \tag{3}$$

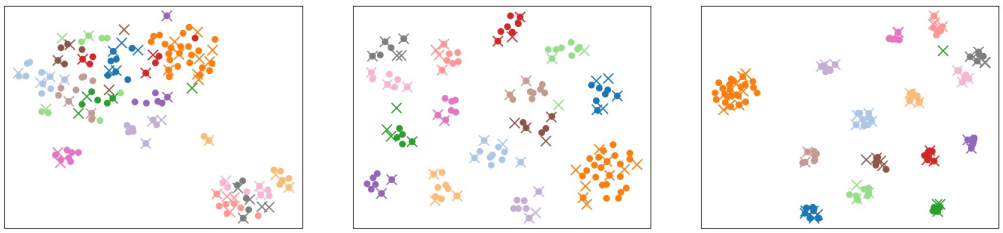
where $\widetilde{\mathcal{T}}$ is written as an implicit function of parameter λ in label correction model $\mathrm{G}(\cdot)$.

Eq. (3) indicates that the training loss $\mathcal{L}_t$ on $\widetilde{\mathcal{T}}$ is affected by both θ and λ. Direct end-to-end optimization to θ and λ on the same training set will lead to the collapse of the model, *e.g.*, corrected labels by $\mathrm{G}(\cdot)$ and predicted labels by $\mathrm{F}(\cdot)$ could converge to a trivial solution like an identical label or all zero vector.

The above issue can be addressed by optimizing λ and θ on different datasets. We sample a training batch and a validation batch from the original training set, and optimize the λ on the validation batch. As λ is updated, $\mathrm{G}(\cdot;\lambda)$ corrects labels on the training batch, which supervise the learning of θ. We hence also denote θ^* as an implicit function of λ, *i.e.*, $\theta^* = \theta^*(\lambda)$. The training of λ can be regarded as a nested optimization and the corresponding validation loss $\mathcal{L}_v$ is denoted as

$$\lambda^* = \arg\min_\lambda \mathcal{L}_v(\theta^*(\lambda)). \tag{4}$$

θ and λ are optimized in two different loops, respectively. The training loop only optimizes the backbone parameter θ. The validation loop fixes θ and updates label correction parameter λ. The data flows to compute the gradients of $\mathrm{G}(\cdot)$ and $\mathrm{F}(\cdot)$ are illustrated in Fig. 2(a). Following sections present details of $\mathrm{G}(\cdot)$ and its optimization, respectively.



(a) initialization (b) baseline (c) NALC

Figure 3: t-SNE visualization of features in a training set containing 30% noises. Color denotes the correct category label, '$\times$' and '$\cdot$' denote noisy and clean labels, respectively. Features in (a) and (b) are extracted by the initialized model and baseline model after training for 30 epochs, respectively. 15 categories are sampled from the Market1501 dataset.

4 Neighbour-Attention Label Correction

As discussed in previous works (Li et al., 2020a), CNN pre-trained on ImageNet gains reasonably good discriminative power. *E.g.*, features of the same class in Fig. 3 (a) are roughly gathered together even though they are labeled differently. CNN also tends to ignore hard noisy labels at the early training stage. Trained by the baseline method for 30 epochs in Fig. 3 (b), those features can be clustered together. This observation indicates robust neighbour cues in the early training stage. We hence propose label correction model referring to neighbour cues. For an image x_i, we denote its K-Nearest Neighbour set as KNN_i. Referring to Fig. 3 (b), a simple way of label correction for x_i is voting labels according to the feature similarities in the neighbor. We denote the corrected label y_i^{vote} as,

$$y_i^{vote} = \sum_{j \in KNN_i} \exp(\tau f_i^T f_j) \bar{y}_j. \tag{5}$$

This simple voting model is not robust in leveraging neighbour cues, *e.g.*, it is difficult to tune parameter K and temperature τ. A smaller K should be set for categories containing a few samples and larger τ is required to prevent the corrected labels over smooth.

We implement the label correction model $\mathrm{G}(\cdot)$ based on neighbor-attention, which produces corrected labels for x_i referring to labels and features of its neighbouring samples. We first transform features of x_i and its neighbours with fully connected layers ϕ, ψ, respectively, then compute the feature similarities. After learning ϕ, ψ through end-to-end training to enhance the retrieval performance, we adopt the resulting feature similarities to represent neighbour relation cues. The corrected labels can be computed by taking feature similarities as fusion weights, *i.e.*,

$$\tilde{y}_i = \sum_{j \in KNN_i} \exp(\phi(f_i)^T \psi(f_j)) \bar{y}_j. \tag{6}$$

Because Eq. (6) uses learnable similarity as voting weights, it is not sensitive to the parameter K. We fix K as 9, and have tested other selections in the appendices. As shown in Fig. 3 (c) our NALC presents a promising performance in label correction.

Optimizing ϕ, ψ in $\mathrm{G}(\cdot)$ leads to continuously updated $\mathrm{F}(\cdot)$ and sample features. Repetitively extracting and updating all sample features makes KNN computation expensive. To reduce the computational complexity, we maintain a first-in first-out memory bank $\mathcal{M}$ to store historic features of training samples. The length of $\mathcal{M}$ equals to the size of the training set. After every iteration in the training loop, we use features in current iteration to update $\mathcal{M}$, and search for their KNNs from $\mathcal{M}$.

5 Nested Optimization to NALC

Training Loss: We follow fine-grained retrieval methods and fuse proxy-based loss and pair-based loss as the training loss $\mathcal{L}_t$. Since our corrected labels are soft labels, we use KL loss $\mathcal{L}_{KL}$ and soft softmax-triplet loss $\mathcal{L}_{ST}$ (Ge et al., 2020a) to implement $\mathcal{L}_t$, *i.e.*, $\mathcal{L}_t = \mathcal{L}_{KL} + \mathcal{L}_{ST}$, and

$$\mathcal{L}_{KL} = \tilde{y}^T(\log(\tilde{y}) - \log(o)),\ \mathcal{L}_{ST} = \mathcal{L}_{bce}(\frac{\exp(d_n)}{\exp(d_n) + \exp(d_p)}, \max(\tilde{y})), \tag{7}$$

Algorithm 1 NALC

Input : Noisy training set $\bar{\mathcal{T}} = \{(x_1, \bar{y}_1), (x_2, \bar{y}_2), \cdots, (x_n, \bar{y}_n)\}$, backbone model $\mathrm{F}(\cdot;\theta)$, label correction model $\mathrm{G}(\cdot;\lambda)$, memory bank $\mathcal{M}$;
Output : optimal backbone model parameter θ^*;
Initialize θ, λ and $\mathcal{M}$;
for epoch = 1...maximum epoch **do**
  if epoch $>$ E **then**
    // Update label correction model
    for batch = 1...maximum batch number **do**
      sample two mini-batches $\{(x_t, \bar{y}_t)\}$ and $\{(x_v, \bar{y}_v)\}$ from $\mathcal{T}$;
      extract features $f_t = F(x_t;\theta)$, $f_v = F(x_v;\theta)$;
      calculate validation loss $\mathcal{L}_v$ on $(f_v, \bar{y}_v)$
      select KNN_t from $\mathcal{M}$;
      calculate $\tilde{y}_t$ using Eq. (6);
      calculate training loss $\mathcal{L}_t$ on $(f_t, \tilde{y}_t)$;
      update λ with $\nabla_\lambda L_v$ using Eq. (9);
    end for
  end if
  // Update backbone model
  for batch = 1...maximum batch number **do**
    sample a mini-batch $(x_t, \bar{y}_t)$ from $\mathcal{T}$;
    extract features $f_t = F(x_t;\theta)$;
    enqueue $(f_t, \bar{y}_t)$ to $\mathcal{M}$;
    select KNN_t from $\mathcal{M}$;
    calculate $\tilde{y}_t$ using Eq. (6);
    calculate training loss $\mathcal{L}_t$ on $(f_t, \tilde{y}_t)$;
    update θ with $\nabla_\theta L_t$;
  end for
end for

where o is the prediction output of the classifier, $\mathcal{L}_{bce}$ denotes the binary cross entropy loss, d_n and d_p denote the distance of hardest negative and positive pairs, respectively. $\max(\cdot)$ returns the maximum value in a vector. $\mathcal{L}_t$ is hence adopted to optimize parameter θ through back-propagated with Eq 3.

Validation Loss: In Eq. (4), θ^* is written as an implicit function of λ, *i.e.*, $\theta(\lambda)$. λ influences the $\mathcal{L}_v$ through the backbone. The gradient of $\mathcal{L}_v$ with respect to λ can be computed as,

$$\frac{\partial \mathcal{L}_v(\theta)}{\partial \lambda} = \frac{\partial \mathcal{L}_v(\theta^*)}{\partial \lambda} + \frac{\partial \mathcal{L}_v(\theta^*)}{\partial \theta^{*T}} \frac{\partial \theta}{\partial \lambda} = \frac{\partial \mathcal{L}_v}{\partial \theta^{*T}} \frac{\partial \theta}{\partial \lambda}, \tag{8}$$

where $\frac{\partial \mathcal{L}_v(\theta^*)}{\partial \lambda} = 0$ and $\frac{\partial \mathcal{L}_v}{\partial \theta^{*T}}$ can be easily computed.

To compute $\frac{\partial \theta}{\partial \lambda}$, we assume that $\frac{\partial \mathcal{L}_t}{\partial \theta} = 0$ and $\frac{\partial^2 \mathcal{L}_t}{\partial \theta \partial \theta^T} \neq 0$. To make this assumption reasonable, we update λ after getting the optimal θ on the training stage, which effectively guarantees $\frac{\partial \mathcal{L}_T}{\partial \theta} \approx 0$. According to the Implicit Function Theorem (IFT) (Lorraine et al., 2020; Gudovskiy et al., 2021) $\frac{\partial \theta}{\partial \lambda} = -[\frac{\partial^2 \mathcal{L}_t}{\partial \theta \partial \theta^T}]^{-1} \frac{\partial^2 \mathcal{L}_t}{\partial \theta \partial \lambda}$. So the gradient of $\mathcal{L}_v$ *w.r.t.* λ can be written as

$$\frac{\partial \mathcal{L}_v(\theta)}{\partial \lambda} = -g_v^T \mathcal{H}^{-1} \frac{\partial^2 \mathcal{L}_t}{\partial \theta \partial \lambda}, \tag{9}$$

where g_v denotes $\frac{\partial \mathcal{L}_v}{\partial \theta^*}$, $\mathcal{H}$ denotes the Hessian matrix $\frac{\partial^2 \mathcal{L}_t}{\partial \theta \partial \theta^T}$.

Adaptive Neumann Approximation of $\mathcal{H}^{-1}$: It is difficult to directly compute the inverse of the whole Hessian matrix. Since vector-Hessian products (Pearlmutter, 1994) have been implemented by PyTorch (Paszke et al., 2017), we use Neumann Series to approximate the inverse Hessian Matrix:

$$g_v^T \mathcal{H}^{-1} = g_v^T \alpha (I - (I - \alpha \mathcal{H}))^{-1} = \alpha \sum_{k=0}^{k_0 - 1} \binom{k_0}{k+1} g_v^T (-\alpha \mathcal{H})^k + g_v^T (I - \alpha \mathcal{H})^{k_0} \mathcal{H}^{-1}, \tag{10}$$

where k_0 denotes the number of iterations to approximate the inverse Hessian, and α is a key weight for convergence. Recent works (Lorraine et al., 2020; Gudovskiy et al., 2021) set α as a fixed value, *i.e.* 0.01. But a fixed value cannot approximate the inverse Hessian throughout the whole training process. In this paper, we use an adaptive way to determine the value of α to further approximate the inverse Hessian.

To decrease the residual term in Eq. (10), we minimize the L2-norm of $(I - \alpha \mathcal{H})$. Sagun *et al.* (Sagun et al., 2016) discovered that almost all eigenvalues of the Hessian after training are no less than zero. So we can diagonalize $\mathcal{H}$ to $U \Sigma U^T$. Then $I - \alpha \mathcal{H} = U(I - \alpha \Sigma) U^T$. We can set α as approximated maximum eigenvalue of $\mathcal{H}$. Here we use power method to approximate the maximum eigenvalue. The final weight $\alpha = \|g_v^T \mathcal{H}^{k_0 - 1}\| / \|g_v^T \mathcal{H}^{k_0 - 2}\|$.

Table 1: Rank@1/mAP(%) under different noise ratios on Market1501 and MSMT17.

Method	Market1501				MSMT17			
	0%	10%	20%	50%	0%	10%	20%	50%
BOT	94.4/86.1	90.4/75.4	87.0/67.4	69.2/43.2	74.1/50.2	68.3/41.2	61.2/33.1	33.2/13.8
SBS	**95.4/88.2**	88.6/69.7	77.4/50.5	61.6/34.4	**81.8/58.4**	68.6/41.2	60.9/32.5	33.1/ 13.0
SpCL	88.1/73.1				42.3/19.1			
ICE	92.0/79.5				59.0/29.8			
Forward	94.0/84.9	90.5/74.8	87.6/67.8	71.5/45.0	71.0/45.3	66.1/44.8	61.5/32.4	34.6/14.5
GLC	94.1/85.3	90.6/75.2	88.4/71.0	75.4/49.7	74.9/49.5	68.1/51.2	62.2/36.3	45.7/15.3
Co-teach	72.9/52.0	73.2/52.3	72.7/52.3	/	/	/	/	/
UbiW	87.9/71.2	84.2/64.3	83.1/63.1	/	/	/	/	/
DNet	87.3/70.8	82.3/61.5	77.0/53.4	65.1/35.1	/	/	/	/
PurifyNet	88.4/72.1	84.2/64.3	83.1/63.1	/	/	/	/	/
CORE	89.6/74.6	85.5/67.7	84.1/66.2	/	/	/	/	/
One4More	94.9/88.9	88.2/69.7	/	/	/	/	/	/
BOT+NALC	94.6/86.0	94.3/**84.9**	94.0/**84.3**	**91.2/79.4**	74.3/49.3	73.2/48.0	72.3/47.2	65.6/37.4
SBS+NALC	95.3/86.7	**95.0**/84.8	**94.6**/83.6	90.9/78.6	79.3/56.2	**76.3/51.6**	**75.0/50.6**	**70.3/44.2**

Algorithm 1 summarizes the training procedure for our backbone and label correction model. Fig. 2(b) visualizes the training loss and label accuracy at different training epochs. It also compares with the supervised training baseline and the label correction method in Eq. (5). It is clear that, our method is more effective in decreasing the training loss and correcting labels.

6 EXPERIMENTS

6.1 IMPLEMENTATION DETAILS

We evaluate the proposed methods in fine-grained image retrieval on Market1501 (Zheng et al., 2015), MSMT17 (Wei et al., 2018), CUB (Wah et al., 2011) and CARS (Krause et al., 2013). More details are described in the appendices. We randomly select a certain percentage of training images and assign them with wrong labels. The above modifications are only applied to the training set. The original testing sets are adopted to evaluate the retrieval performance. ResNet50 pre-trained on ImageNet is adopted as our backbone. We use two one-layer fully-connection (fc) layers to implement feature transforms ϕ and ψ in Eq. (6). Their output dimension is equal to the input feature dimension. Those two fc layers are initialized as identity matrices. Therefore, the corrected label by the initialized NALC is equivalent to the feature similarity weighted soft label. The Memory Bank $\mathcal{M}$ is randomly initialized. The loss computation in baseline is adopted as our validation loss. The max training epoch is set to 60 and meta optimization starts after the 20th epoch, *i.e.*, 'E' in Algorithm 1 is set to 20. We set batch size to 32/64 and input size to 256$\times$128/224$\times$ 224 for person re-identification/other datasets, respectively. ADAM is used as our optimizer for the backbone and meta-learning framework. Its initial learning rate is set to 3×10^{-4}. At the 20th and 40th epoch, learning rate is multiplied by 0.1.

6.2 COMPARISON WITH RECENT METHODS

We test our method with different noise ratios ranging from 0 to 50% on four datasets, and compare it against recent noise-resistant learning methods. Note that, our method is a label correction method, it is hence compatible to different baselines and backbones.

On person re-identification datasets, we adopt two baselines, *i.e.* BOT (Luo et al., 2019) and SBS (He et al., 2020), respectively. Table 1 summarizes the comparison. We compare with two recent unsupervised re-id methods SpCL (Ge et al., 2020b) and ICE (Chen et al., 2021). Five noise-resistant methods designed for classification tasks are also compared, including Forward Correction (FC) (Patrini et al., 2017), GLC (Hendrycks et al., 2018), MLC (Zheng et al., 2021), co-teach (Han et al., 2018) and UbiW (Li et al., 2019a), respectively. The first three are label correction methods and the rest two are noise ignoring methods. Table 1 also compares three noise-resistant re-id methods including DNet (Yu et al., 2019), PurifyNet (Ye & Yuen, 2020), CORE (Ye et al., 2022) and One4More (Zhang et al., 2021). The performance of co-teach and UbiW is reported in PurifyNet (Ye & Yuen, 2020). It is interesting to observe that, strong algorithms for clean person re-id datasets like

Table 2: Precision@1 (%) under different noise ratio on CUB and CARS.

Method	CUB				CARS			
	0%	10%	20%	50%	0%	10%	20%	50%
nSoftmax	67.8	65.6	63.6	55.5	87.5	85.4	80.7	64.7
Forward	66.2	64.1	63.4	59.5	84.9	81.6	78.4	66.8
GLC	67.5	66.0	64.9	61.3	87.1	85.7	81.0	69.9
Co-teach	/	53.7	51.1	45.0	/	73.5	70.4	59.6
Proxy Anchor	**69.2**	67.1	65.3	59.0	**87.6**	85.6	80.8	65.7
Circle Loss	66.7	47.5	45.3	13.0	83.4	71.0	56.2	15.2
PRISM	/	58.8	58.7	56.0	/	80.0	78.0	72.9
NALC	67.5	**67.2**	**66.9**	**63.5**	87.0	**86.0**	**82.1**	**74.9**

SBS (He et al., 2020) cannot maintain their advantages on noisy training data. Their performances become worse than unsupervised methods when the noise ratio is larger than 20%. This could be because those methods do not differentiate noisy labels, which leads to biased models during training. Unsupervised methods perform worse on MSMT17 than on Market1501, indicating the increased difficulty of label prediction on more challenging datasets. NALC outperforms these unsupervised methods by clear margins even at noise ratio = 50%.

Table 2 summarizes the comparison on two fine-grained instance retrieval datasets. We apply NALC to nSoftmax implemented by Pytorch Metric Learning (Musgrave et al., 2020). In the table, FC (Patrini et al., 2017), GLC (Hendrycks et al., 2018), MLC (Zheng et al., 2021), co-teach (Han et al., 2018) are noise-resistant methods designed for classification. Table 2 also compares with three deep metric learning methods. Proxy Anchor (Kim et al., 2020) and Circle Loss (Sun et al., 2020) are proxy-based and pair-based methods, respectively. PRISM (Liu et al., 2021) is a noise-resistant method which replaces individual data points with class centers. Among those three methods, Proxy Anchor shows the best performance on a clean training set. It also demonstrates better label noise robustness than Circle Loss. It is interesting to observe that, higher noise ratio severely degrades the performance of Circle Loss. This is because pair-based loss is more sensitive to label noises, *e.g.*, on a dataset containing 50% noisy labels, the portion of correct sample pairs is 25%. In Table 2, our NALC also achieves the best performance. Table 1 and Table 2 present similar conclusions, *i.e.*, our method gets the best performance for different datasets and noise ratios.

6.3 Ablation Study

Comparison with other label correction methods: Table 3a compares NALC with several neighbor-aware label correction methods. "Label vote" is the simple label correction in Eq. (5). It follows the similar process of 'Update backbone model' in Algorithm 1. "Single projection" shares the parameters of two projection functions, *i.e.* ϕ on current features and ψ on memory features. "Label vote" brings a significant improvement over the baseline, *e.g.*, boosts the label accuracy from 70% to 94.8%. This shows the effectiveness of neighbour cues in label correction. "Single projection" achieves better label accuracy and rank1 accuracy by learnable projection. NALC achieves the best performance. It indicates that model features and memory features requires separate projections.

Test on meta learning strategy: Eq. (9) and Eq. (10) implement our nested optimization to the label correction model. Table 3b compares our algorithm against several variants. "No meta" updates the label correction model through end-to-end training without using meta-learning. It leads to model collapse. $\mathcal{H}^{-1} \rightarrow I$ replaces the inverse Hessian with an identity matrix. "Fixed α" fixes α as 0.01 referring to (Lorraine et al., 2020). "Adaptive α" denotes our method. Our training strategy achieves the best performance.

Test on loss function: Table 3c tests the validity of our loss function, which fuses proxy-based loss and pair-based loss. It is clear that, fusing those two types of loss functions achieves the best performance. Note that, corrected labels in Eq. (6) are soft labels. As the conversion from soft-labels to one-hot labels is not differentiable. Our loss functions are computed with soft labels. Table 3c further tests the hard-label version of our loss function, which only updates θ but can not be adopted to optimize λ. It is clear that, the hard version leads to lower performance.

Table 3: Ablation studies of our proposed NALC on individual components. Market1501 with 30% noise ratio is adopted as the dataset. "Label Accuracy" is computed by first converting each predicted soft label into a one-hot label, then comparing it with its ground-truth.

(a) different label correction methods.

Model	Rank@1	Label Accuracy
Baseline	82.3	70.0
Label vote	91.9	94.8
Single projection	92.7	97.2
NALC	**93.8**	**97.9**

(b) different nested optimization strategies.

Method	Rank@1	Label Accuracy
No meta	0.1	0.2
$\mathcal{H}^{-1} \to I$	92.0	95.5
Fixed α	92.3	95.7
Adaptive α	**93.8**	**97.9**

(c) different loss functions.

Proxy-based	Pair-based	Rank@1	mAP
Hard	Hard	92.8	82.5
Hard	Soft	92.5	82.6
Soft	Hard	93.4	83.1
Soft	Soft	**93.8**	**83.2**

(d) different validation set selection methods.

Validation set	Rank@1
10% clean	93.2
10% separated	92.5
20% separated	92.3
100% shared	**93.8**

Table 4: Rank-1 accuracy and mAP on Market1501 under different noise types and ratios.

Noise Type	Noise Ratio	10%	20%	30%	50%
Asymmetric	Baseline	90.8/77.3	89.1/72.8	88.3/70.4	87.1/67.6
	Label vote	93.1/82.7	91.6/79.3	89.6/73.9	85.8/67.3
	NALC	**94.2/84.5**	**94.0/82.4**	**93.0/78.4**	**91.1/74.2**
Pair-Flip	Baseline	91.1/76.3	89.0/72.4	87.8/69.9	87.9/69.2
	Label vote	92.6/82.4	91.4/78.6	88.8/74.0	86.2/67.5
	NALC	**94.0/84.1**	**92.4/81.6**	**91.2/75.6**	**89.3/71.8**
Pattern	Baseline	92.1/78.4	89.0/73.9	87.0/70.3	83.0/63.9
	Label vote	94.7/84.6	94.2/84.1	93.9/82.6	91.1/75.4
	NALC	**95.3/86.1**	**95.0/84.7**	**94.5/83.0**	**92.3/77.3**

Test on validation set selection methods: "10% clean" in Table 3d samples 10% clean samples with true labels and uses the rest 90% as training set. "x% separated" randomly selects x% samples for validation and uses the rest for training. "10% separated" performs worse than "10% GT". Further enlarging the validation set decreases the size of training set, hence is harmful to the performance. "100% shared" uses two independent data loaders on a shared noisy training set to select training samples and validation samples. "100% shared" performs the best and is adopted in our method.

Test on other noise types: To test the generalization capability of NALC to different label noises, Table 4 compares it with several label correction methods under different noise types and ratios. We follow CDR (Xia et al., 2021) to generate Asymmetric and Pair-Flip noises and follow DNet (Yu et al., 2019) to generate Pattern noises. Asymmetric noise flips labels within a set of class pairs. Pair-Flip noise flips each class to its adjacent classes in the feature space. Pattern noise flips the label of the sample to its nearest negative class, and leads to the worst baseline performance. Under these noise types, our method consistently outperforms baseline and "Label vote". For instance, on the Asymmetric noise, our method outperforms the "Label vote" by clear margins at various noise ratios. Especially for the case with 50% noises, NALC outperforms "Label vote" by 6.9% in mAP.

7 Conclusion

This paper proposes Neighbor-Attention Label Correction (NALC) for noise-resistant fine-grained image retrieval. Different from previous works that ignore noisy labels, this work corrects and leverages those labels to chase better retrieval performance. Our label correction model computes neighbor relationships to infer corrected labels and is optimized by a meta learning framework on a validation batch. We also propose a novel nested optimization referring to Implicit Function Theorem and adaptive Neumann approximation to enhance the optimization efficiency. Extensive experiments on four datasets and various types of noises show the remarkable performance of our method.

REFERENCES

Kaifeng Bi, Changping Hu, Lingxi Xie, Xin Chen, Longhui Wei, and Qi Tian. Stabilizing darts with amended gradient estimation on architectural parameters. *arXiv preprint arXiv:1910.11831*, 2019.

Hao Chen, Benoit Lagadec, and François Bremond. Ice: Inter-instance contrastive encoding for unsupervised person re-identification. In *ICCV*, pp. 14960–14969, October 2021.

Yixiao Ge, Dapeng Chen, and Hongsheng Li. Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification. In *ICLR*, 2020a.

Yixiao Ge, Dapeng Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-paced contrastive learning with hybrid memory for domain adaptive object re-id. *NeurIPS*, 2020b.

Denis Gudovskiy, Luca Rigazio, Shun Ishizaka, Kazuki Kozuka, and Sotaro Tsukizawa. Autodo: Robust autoaugment for biased data with label noise via scalable probabilistic implicit differentiation. In *CVPR*, pp. 16601–16610, 2021.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, pp. 8527–8537, 2018.

Lingxiao He, Xingyu Liao, Wu Liu, Xinchen Liu, Peng Cheng, and Tao Mei. Fastreid: A pytorch toolbox for general instance re-identification. *arXiv*, 2020.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *NeurIPS*, 2018.

Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *CVPR*, pp. 3238–3247, 2020.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV workshops*, pp. 554–561, 2013.

Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *CVPR*, pp. 5447–5456, 2018.

Jia Li, Yafei Song, Jianfeng Zhu, Lele Cheng, Ying Su, Lin Ye, Pengcheng Yuan, and Shumin Han. Learning from large-scale noisy web data with ubiquitous reweighting for image classification. *TPAMI*, 2019a.

Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *CVPR*, pp. 5051–5059, 2019b.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *AISTATS*, pp. 4313–4324. PMLR, 2020a.

Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M Robertson, and Yongxin Yang. Dada: Differentiable automatic data augmentation. *arXiv preprint arXiv:2003.03780*, 2020b.

Chang Liu, Han Yu, Boyang Li, Zhiqi Shen, Zhanning Gao, Peiran Ren, Xuansong Xie, Lizhen Cui, and Chunyan Miao. Noise-resistant deep metric learning with ranking-based instance selection. In *CVPR*, pp. 6811–6820, 2021.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Xiaobin Liu, Shiliang Zhang, Xiaoyu Wang, Richang Hong, and Qi Tian. Group-group loss-based global-regional feature learning for vehicle re-identification. *TIP*, 29:2638–2652, 2019.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *AISTATS*, pp. 1540–1552. PMLR, 2020.

H. Luo, W. Jiang, Y. Gu, F. Liu, X. Liao, S. Lai, and J. Gu. A strong baseline and batch normalization neck for deep person re-identification. *TMM*, pp. 1–1, 2019. ISSN 1941-0077. doi: 10.1109/TMM.2019.2958756.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. Pytorch metric learning, 2020.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pp. 1944–1952, 2017.

Barak A Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, pp. 4334–4343. PMLR, 2018.

Brendan van Rooyen, Aditya Krishna Menon, and Robert C Williamson. Learning with symmetric label noise: the importance of being unhinged. In *NeurIPS*, pp. 10–18, 2015.

Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *NeurIPS*, 32:1919–1930, 2019.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *CVPR*, pp. 6398–6407, 2020.

Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, pp. 5552–5560, 2018.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, and Neil M Robertson. Deep metric learning by online soft mining and class-aware attention. In *AAAI*, volume 33, pp. 5361–5368, 2019.

Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *CVPR*, pp. 79–88, 2018.

Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris N Metaxas, and Chao Chen. A topological filter for learning with label noise. In *NeurIPS*, 2020.

Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *ICLR*, 2021.

Shiyu Xuan and Shiliang Zhang. Intra-inter camera similarity for unsupervised person re-identification. In *CVPR*, pp. 11926–11935, 2021.

Mang Ye and Pong C Yuen. Purifynet: A robust person re-identification model with noisy labels. *TIFS*, 15:2655–2666, 2020.

Mang Ye, He Li, Bo Du, Jianbing Shen, Ling Shao, and Steven C. H. Hoi. Collaborative refining for person re-identification with label noise. *IEEE Transactions on Image Processing*, 31:379–391, 2022. doi: 10.1109/TIP.2021.3131937.

Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *CVPR*, pp. 7017–7025, 2019.

Tianyuan Yu, Da Li, Yongxin Yang, Timothy M Hospedales, and Tao Xiang. Robust person re-identification by modelling feature uncertainty. In *ICCV*, pp. 552–561, 2019.

Enwei Zhang, Xinyang Jiang, Hao Cheng, Ancong Wu, Fufu Yu, Ke Li, Xiaowei Guo, Feng Zheng, Weishi Zheng, and Xing Sun. One for more: Selecting generalizable samples for generalizable reid model. In *AAAI*, volume 35, pp. 3324–3332, 2021.

Guoqing Zheng, Ahmed Hassan Awadallah, and Susan Dumais. Meta label correction for noisy label learning. In *AAAI*, 2021.

Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *ICCV*, pp. 1116–1124, 2015.

Zhun Zhong, Liang Zheng, Zhiming Luo, Shaozi Li, and Yi Yang. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *CVPR*, pp. 598–607, 2019.

Table 5: Rank-1 accuracy of different neighbor size K for both the non-parametric label voting method and NALC

neighbor size K	Label vote		NALC	
	30	50	30	50
5	**91.9**	87.0	93.5	90.1
9	91.3	**88.4**	**93.8**	**91.2**
13	90.4	87.8	93.7	91.1

A DETAILED DESCRIPTION OF THE DATASETS

We evaluate the proposed methods in fine-grained image retrieval on four datasets, including two person re-identification datasets and two fine-grained instance retrieval datasets, respectively. They are Market1501 (Zheng et al., 2015), MSMT17 (Wei et al., 2018), CUB (Wah et al., 2011) and CARS (Krause et al., 2013). Among them, Market1501,CUB and CARS can be directly downloaded from the homepage of the project. MSMT17 (Wei et al., 2018) is available after we sign the agreement. We are required not to further distribute, publish, copy, or further disseminate the database.

Person re-id datasets: Market1501 (Zheng et al., 2015) contains 32,668 images of 1,501 identities captured by 6 cameras. The training set contains 12,936 images of 751 identities. The testing set contains the rest 750 identities, and 3,368 images for query, 19,732 images for the gallery. MSMT17 (Wei et al., 2018) contains 126,441 images of 4,101 identities captured by 15 cameras. Numbers of training/query/gallery images are 30,248/11,659/82,161, respectively.

Fine-grained instance retrieval datasets: CUB (Wah et al., 2011) contains 11,788 images of 200 bird species. We use the first 100 species for training and the rest for testing. CARS (Krause et al., 2013) contains 16,185 images of 196 car models. We use the first 98 models for training and the rest for for testing.

B DISCUSSION OF THE NEIGHBOR SIZE K

We compare NALC with the the non-parametric label voting method with several selections of neighbor size K in Table 5. The hyperparameter K is hard to tune in label voting method. When the noise ratio is fixed, the performances of label voting is sensitive to neighbor sizes. And when the noise ratio varies, its optimal K also varies.