Label Noise Robust Crowd Counting with Loss Filtering Factor

Zhengmeng Xu, Hai Lin, Yufeng Chen & Yanli Li

To cite this article: Zhengmeng Xu, Hai Lin, Yufeng Chen & Yanli Li (2024) Label Noise Robust Crowd Counting with Loss Filtering Factor, Applied Artificial Intelligence, 38:1, 2329859, DOI: 10.1080/08839514.2024.2329859

To link to this article: https://doi.org/10.1080/08839514.2024.2329859

© 2024 The Author(s). Published with license by Taylor & Francis Group, LLC.

View supplementary material

Published online: 21 Mar 2024.

Submit your article to this journal

Article views: 370

View related articles

View Crossmark data
Label Noise Robust Crowd Counting with Loss Filtering Factor

Zhengmeng Xu\textsuperscript{a}, Hai Lin\textsuperscript{b}, Yufeng Chen\textsuperscript{c}, and Yanli Li\textsuperscript{d}

\textsuperscript{a}Postdoctoral Research Center, Industrial and Commercial Bank of China, Beijing, China; \textsuperscript{b}School of Big Data Science, Hebei Finance University, Baoding, China; \textsuperscript{c}Department of Information Technology, Uppsala University, Uppsala, Sweden; \textsuperscript{d}School of Electrical and Information Engineering, The University of Sydney, Sydney, Australia

\textbf{ABSTRACT}

Crowd counting, a crucial computer vision task, aims at estimating the number of individuals in various environments. Each person in crowd counting datasets is typically annotated by a point at the center of the head. However, challenges like dense crowds, diverse scenarios, significant obscuration, and low resolution lead to inevitable label noise, adversely impacting model performance. Driven by the need to enhance model robustness in noisy environments and improve accuracy, we propose the Loss Filtering Factor (LFF) and the corresponding Label Noise Robust Crowd Counting (LNRCC) training scheme. LFF innovatively filters out losses caused by label noise during training, enabling models to focus on accurate data, thereby increasing reliability. Our extensive experiments demonstrate the effectiveness of LNRCC, which consistently improves performance across all models and datasets, with an average enhancement of 3.68\% in Mean Absolute Error (MAE), 6.7\% in Mean Squared Error (MSE) and 4.68\% in Grid Average Mean Absolute Error (GAME). The universal applicability of this approach, coupled with its ease of integration into any neural network model architecture, marks a significant advancement in the field of computer vision, particularly in addressing the pivotal issue of accuracy in crowd counting under challenging conditions.

\textbf{Introduction}

Crowd counting represents a prominent computer vision task, aimed at automatically estimating the number of individuals in unconstrained scenes (Idrees et al. 2013; Laradji et al. 2018; Liu et al. 2019; Ma et al. 2019). This task has garnered significant attention in recent years, with extensive research and implementation across various real-world scenarios, including smart buildings (Zou et al. 2018), traffic monitoring (Marsden et al. 2018; Zhang et al. 2017), and public spaces in Saudi Arabia (Alotibi et al. 2019). By harnessing real-time...
image data, crowd counting facilitates applications such as video surveillance (Wang, Hou, and Chau 2019), enhanced security (Chan, John Liang, and Vasconcelos 2008), and efficient bandwidth allocation (Zou et al. 2017).

In general, crowd counting is challenging due to heavy overlaps and occlusions, complex and noisy backgrounds, and variations in perspective and illumination. In the past decade, a number of crowd counting algorithms have been proposed in the literature. Most early works estimate crowd counts via the detection of people, bodies, or heads in the image (Li et al. 2008; Rabaud and Belongie 2006), which may have inaccurate estimates and considerable computational complexity of dense crowds due to the heavy overlaps and occlusions of people. Currently, methods that mainly cast crowd counting as a density map estimation problem and combine it with convolutional neural networks (CNNs) have made remarkable progress (Boominathan, Kruthiventi, and Venkatesh Babu 2016; Chen et al. 2021; Cheng et al. 2019; Lin et al. 2022; Ma et al. 2019; Wang et al. 2020). By this method, the values of the crowd density map regressed by CNNs are summed to give the total size of the crowd.

Training crowd counting models effectively poses a challenge due to the nature of most available datasets, which provide only point annotations for each training image, typically denoting the center of a person’s head (Idrees et al. 2013, 2018; Zhang et al. 2016a). One prevalent approach involves transforming the point annotations into density maps using a Gaussian kernel, treating these density maps as the ‘ground truth,’ and training the model by regressing values at each pixel within the density map. Furthermore, recent studies (Ma et al. 2019; Wang et al. 2020; Wan, Liu, and Chan 2021; Zhang et al. 2016a) have explored alternative methods to enhance this point-to-density-map conversion process, yielding promising performance improvements.

The quality of datasets and the accuracy of point annotations have a profound impact on the performance of crowd count estimators in crowd counting learning tasks (Gao et al. 2020). Presently, point annotations for crowd counting are primarily acquired through manual labor, requiring the meticulous labeling of each person in every image of the dataset (Jingying 2021; Li et al. 2021). However, due to factors such as overlapping individuals, low image resolution, and extremely high crowd densities – especially near the vanishing point of the image – inevitably, there exists a significant likelihood of mislabeled annotations by annotators. Furthermore, given that these point annotations represent only a small fraction of individuals’ heads, there is inherent spatial error; in other words, not every point annotation precisely corresponds to the center of a person’s head. As a result, these “unknown” mislabeled point annotations and spatial errors can also hurt the training of crowd count estimators.

To alleviate the impact of annotation noises and bridge the research gap, we introduce the Loss Filtering Factor (LFF) to assist the model in filtering
out losses (at the pixel level) that are most likely due to annotation noises and further propose the Label Noise Robust Crowd Counting (LNRCC) training scheme based on LFF which enables the model focus on more critical non-noise losses during the training process. Specifically, LNRCC first initializes a crowd counting model and trains it on the given training images and annotations to obtain an initial model. Then, predictions are made on test images using this initial model, and the deviation between predictions and annotations is calculated to represent the loss for each data point. LNRCC then sorts this loss vector to rank the losses from small to large. After that, it generates a binary mask vector based on the sorted losses and a hyperparameter theta, which acts as a filter to remove certain losses. Using this mask, LNRCC calculates the filtered losses by zeroing out some elements of the original loss vector so that only selected losses are used for supervision. Finally, the initial model is updated using LFF as the loss function. The steps are repeated until the model converges. In this way, LNRCC selectively filters out likely noise-induced losses, enabling the model to focus more on critical non-noise data during training and enhancing robustness against label noise.

We evaluate the performance of our proposed LNRCC method across various backbone networks (Liu et al. 2021; Ma et al. 2019; Wang et al. 2020; Zhang, Choi, and Hong 2022) using multiple datasets (Idrees et al. 2013, 2018; Liu et al. 2021; Zhang et al. 2016a). Our comprehensive experimental results demonstrate that our LNRCC effectively enhances the robustness of crowd counting models in label-noise environments while significantly improving their overall learning performance.

Recent progress in crowd counting has seen a variety of innovative approaches tailored to specific challenges in the field. The Scale Region Recognition Network (Guo et al. 2023) and the Scale-Context Perceptive Network (Zhai et al. 2023) address scale variations and context-aware processing, crucial for applications in intelligent transportation systems and smart cities. Attention mechanisms have been pivotal in enhancing accuracy within dense crowds, as seen in the Group-split Attention Network (Zhai et al. 2022), Spatial-Frequency Attention Network (Guo et al. 2022), and FPANet (Zhai et al. 2023). These models employ attention to manage complex crowd scenes effectively.

In contrast to these developments, our research introduces the Loss Filtering Factor (LFF) and the Label Noise Robust Crowd Counting (LNRCC) training scheme. While the aforementioned SOTA models focus on structural, attentional, and scale-aware aspects of the networks, our LFF approach specifically targets the challenge of label noise in crowd counting datasets. By filtering out noise-impacted losses, LFF enables a more accurate and reliable training process, addressing a key challenge that has been less explored in these recent advancements.
This paper is a significant extension of our prior conference paper, our contributions are summarized as follows:

- We identify that label noise, encompassing both spatial noise and quantity noise within the training data, exerts a significant influence on the reduction of learning performance in crowd counting models.
- We propose the loss filtering factor (LFF) to alleviate the negative impact of label noise by filtering losses that are likely caused by noise during model training.
- We propose the Label Noise Robust Crowd Counting (LNRCC) Training Scheme based on LFF. LNRCC is a comprehensive training pipeline designed to bolster the robustness of crowd counting models in label-noise environments.

Related Work

Crowd Counting

Crowd counting has been extensively studied as a fundamental issue in computer vision. The approaches to this problem can be categorized into three types: detection, direct count regression, and point supervision.

Initially, most methods (Li et al. 2008; Liu et al. 2019; Rabaud and Belongie 2006) focused on detecting individuals, heads, or upper bodies in images. However, this approach faces significant challenges in dense crowds, primarily due to the extensive occlusions and the labor-intensive nature of bounding box annotation.

Transitioning to the next phase of development, later methods (Idrees et al. 2018; Jiang et al. 2019; Li, Zhang, and Chen 2018) moved away from detection-based approaches. Instead, they regress to a “ground truth” density map created from point annotations. These methods use location information to learn a density map for each training sample. Nonetheless, they often assume an even crowd distribution, which is not always reflected in images due to factors like camera angles and imaging techniques. In addressing these challenges (Hu et al. 2022) introduced RDC-SAL, a framework that combines refine distance compensating with quantum scale-aware learning, significantly enhancing feature extraction in dense scenes.

Further evolving the methodology, recent works (Dong et al. 2020; Ma et al. 2019; Wang et al. 2021) have suggested using point supervision directly, bypassing the need for generating density maps. These advancements have led to novel approaches like optimal transport (Ma et al. 2021) and divergent measuring techniques, focusing on weak supervision without relying on Gaussian distribution assumptions. In this context (Hafeezallah, Al-Dhamari, and Abd Rahman Abu-Bakar 2022) introduces an innovative
approach using a multi-scale network with an integrated attention unit, further enhancing the accuracy and robustness of crowd counting in challenging scenarios. The QE-DAL framework by (Hu, Tang, and Yang 2023) leverages quantum computing for feature extraction in dense crowd scenes, providing a new dimension in crowd counting methodologies.

Further innovations include the integration of multiple attention mechanisms and scale-aware strategies, such as in the Triple Attention and Scale-Aware Network (Guo et al. 2022), designed for remote sensing, and the Dense Attention Fusion Network (Guo et al. 2023), which focuses on IoT systems. The Multiscale Aggregation Network (Guo et al. 2022) offers a unique approach with its smooth inverse mapping technique. Additionally, the comprehensive analysis of crowd counting methodologies in IoT by Gao et al (Gao et al. 2023) provides valuable insights into the applicability of various approaches in IoT scenarios.

Other notable works include the Lightweight Ghost Attention Pyramid Network (Guo et al. 2023), which offers an efficient solution for smart city applications, the DA2Net (Zhai et al. 2023), a dual attention-aware network designed for robust crowd counting, and the Attentive Hierarchy ConvNet (Zhai et al. 2023), which emphasizes hierarchical convolutional approaches for smart city environments.

In crowd counting tasks, selecting an effective loss function is critical. Initially, Euclidean loss was commonly used, focusing on minimizing the MSE between predicted density maps and ground truth ((Li, Zhang, and Chen 2018; Zhang et al. 2016a)). While simple and flexible, this approach overlooks the correlation between adjacent pixels, limiting the quality of density maps.

To address these limitations, newer methods introduced structural similarity-based losses, like the SSIM loss in SANet (Cao et al. 2018). These allow models to learn local correlations at various scales, but they struggle with scale variations. Here, the approach of (Hafeezallah, Al-Dhamari, and Abd Rahman Abu-Bakar 2022) demonstrates the efficacy of leveraging multi-scale features and attention mechanisms to address scale variations in crowd scenes.

Another development in this field involves the use of multi-task learning frameworks (Sindagi and Patel 2019; Wei, Yuan, and Wang 2020), (Hafeezallah, Al-Dhamari, and Rahman Abu-Bakar Hafeezallah, Al-Dhamari, and Abd Rahman Abu-Bakar 2022). These frameworks, while effective in crowded scenes, are sensitive to hyperparameters and require precise tuning. Strategies like divide-and-conquer (Xiong et al. 2019) also emerged, offering efficient segmentation but at higher computational costs.

Additionally, diverse loss optimization strategies have been explored. For instance, CNN-Boosting uses layered boosting and selective sampling (Walach and Wolf 2016), and D-ConvNet employs deep negative correlation learning (Shi et al. 2018), enhancing counting robustness. In the context of scene
classification and motion pattern analysis (Mohammed et al. 2023) introduces the concept of adaptive synthetic oversampling and fully connected deep neural networks, providing new insights into the classification of crowd scenes based on motion patterns.

**Noisy Labels**

The effectiveness of deep neural networks is dependent on having access to high-quality labeled training data because label mistakes (label noise) in training data can significantly impair model performance on clean test data (Zhang et al. 2021). Unfortunately, samples with faulty or incorrect labels are virtually always present in big training datasets. Recently, an increasing number of academics have focused on this issue. Han et al. proposed a Co-teaching model for combating noisy labels (Han et al. 2018), Jiang et al. contributed to a deeper understanding of deep learning using non-synthetic noisy labels (Jiang et al. 2020). To combat overfitting on faulty labels, Jiang et al. proposed the MentorNet method, a strategy for learning another neural network (Jiang et al. 2018).

However, these methods can’t transfer well to the crowd counting problem due to insufficient datasets for controlled tests and high computing costs. On the other hand, studies have shown that deep learning models outperform humans in a variety of activities, such as Image classification (Zoph et al. 2018), go (Silver et al. 2016), and speech recognition (Pham et al. 2019). Therefore, for crowd counting problems, particularly high-density crowd counting tasks, we may expect that, under certain conditions, existing deep learning models predict more true signals than human-labeled annotations.

**Proposed Method**

**Background and Motivation**

Unlike other tasks, label noise in datasets is unavoidable for crowd counting tasks because of the dense crowd, variety of scenarios, and significant obscuration.

On the one hand, most popular crowd benchmarks have large crowd density, making consistency and accuracy in point annotations challenging. The statistics of the multi-scene datasets for dense crowd counting are summarized in Table 1. The majority of datasets feature an average of more than a hundred persons per image.

On the other hand, based on observation, the label noise of the popular crowd benchmark may be separated into two main categories: spatial noise and quantity noise. The relative locations of labeled points vary for the same
images in the dataset. Some are the pixel in the center of the head. In contrast, others are just a random pixel within the person (e.g., in people’s chest or waist); in some cases, points outside the person are also annotated. Such annotated inaccuracies are referred to as spatial noise, as shown in Figure 1a. Moreover, due to the limited availability of high-resolution crowd images, there is quantity noise in the dataset, particularly for images with a high crowd density and low resolution, such as missing annotated and duplicate annotated, as shown in Figure 1b. Both of the aforementioned label noises will undoubtedly have an impact on the training and performance of deep learning models to some extent (Zhang et al. 2021).

**Loss Filtering Factor**

The Loss Filtering Factor is proposed to alleviate the negative effects of label noise based on the assumption that the trained neural network model predicts more correct signals under certain conditions than human-labeled annotations (Khan, Menouar, and Hamila 2023a). The proposed Loss Filtering Factor can filter out the losses assumed to be caused by label noise (quantity noise and spatial noise) during the training process.

**Figure 2** shows a simple (2-D) example of training with and without Loss Filtering Factor. During the training process, the model’s predicted values in epoch $T$ will be closer to the label values than in epoch $T - 1$, i.e., the total of losses of epoch $T$ is less than epoch $T - 1$. When training without using the loss filter factor, the model will consider the losses caused by label noise as regular losses; therefore, even though the total losses are reduced, the non-noise losses are not always minimized; When training using the Loss Filtering Factor, the Loss Filtering Factor can filter out the losses believed to be caused by label noise. As a result, both total losses and non-noise losses decrease.

The inspiration for the LFF concept is rooted in the principle of selective attention in human perception, analogous to focusing on relevant information while filtering out the less pertinent. This approach is critical in environments with label noise, a prevalent issue in computer vision tasks like crowd counting. LFF is designed to dynamically adjust the weight of each data point in the loss function, based on its estimated level of noise. This selective filtering allows the model to prioritize learning from non-noisy, reliable data, thereby
enhancing the overall model performance and robustness against label noise. The development of LFF is a response to the need for simple yet effective solutions in machine learning models, emphasizing tailored approaches to address specific challenges effectively, rather than applying generic methods.
The proposed method employs the mask $M = [m^i]_N^j$ to selectivity supervise the training of the neural network model. It can be be defined as:

$$LFF = \sum_{j}^{N} m^j \cdot \ell^j$$

where $L = [\ell^j]_N$ is the deviation between the predicted value and label value. For example, for methods based on point supervision, $\ell^j$ is the difference between each predicted point value and the corresponding label point value. In contrast, for density-maps-based methods, $\ell^j$ is the difference between predicted and generated density maps. The value of sizes $N$ is determined by the loss function. For example, in MSE Loss $N$ equals the size of the density map, while in Bayesian Loss $N$ equals the number of annotated points.

Given that there is unavoidably some label noise in datasets and that the trained model sometimes predicts more accurate signals than annotations, the mask $M$ provides a mechanism to filter out some losses, preventing them from participating in the back-propagation process of model training. We consider the deviation $\ell^j$ between predictions and labels to be a type of label uncertainty. According to the foregoing argument, if the $\ell^j$ is excessively large, it is most likely generated by label noise. In this case, we shall dynamically diminish or eliminate its weight in back-propagation. For efficient computation, we adopt $M$ as binary vectors, after calculating all losses $[\ell^j]_N$, We get the sorted list $S = [\ell^j]_N$ by sorting $[\ell^j]_N$ in ascending order. Then $m^j$ is as follows:

$$m^j = \begin{cases} 
1, & \text{if } \ell^j \in \{S(1), S(2), S(3), \ldots, S([\theta N])\} \\
0, & \text{otherwise} 
\end{cases}$$

where $\theta$ is the parameter of Loss Filtering Factor and $[\theta N]$ denotes the largest integer no more than $\theta N$. 

*Figure 2.* Comparison of training with(right) and without(left) using loss filter factor.
If $F = F(\hat{y})$ is the loss function used by the model. Obviously, when Loss Filtering Factor $\theta = 1.0$, it is the same as not using Loss Filtering Factor and putting all losses $[\hat{y}]$ into the loss function $F$ directly; when Loss Filtering Factor $\theta = 0.85$, it means that there are 15% of label values are considered to be noise, and only 85% of label values with the lowest deviations from the prediction will be involved in supervision. Finally, the overall loss function with the Loss Filtering Factor is as follows:

$$F_{\text{LFF}} = F(L\text{FF}) = F\left(\sum_{j}^{N} m^j \cdot \hat{y}^j\right)$$

(3)

**Label Noise Robust Crowd Counting Training Scheme**

Based on LFF we design Label Noise Robust Crowd Counting (LNRCC) training scheme to enhance crowd counting models robustness and improve learning performance under the label noise environment. Algorithm 1 illustrates the steps of LNRCC.

LNRCC first initializes a crowd counting model and trains it on the given training images and annotations to obtain an initial model. Then, predictions are made on test images using this initial model, and the deviation vector between predictions and annotations is calculated to represent the loss for each data point. Then, LNRCC sorts this loss vector in ascending order to rank the losses from small to large. After that, it generates a binary mask vector based on the sorted losses and a hyper parameter $\theta$, which acts as a filter to remove certain losses. Using this mask, LNRCC calculates the filtered losses by zeroing out some elements of the original loss vector, so that only selected losses are used for supervision. Finally, the initial model is updated using LFF as the loss function. The steps are repeated until the model converges. In this way, LNRCC selectively filters out likely noise-induced losses, enabling the model to focus more on critical non-noise data during training and enhancing robustness against label noise.

**Algorithm 1** Label Noise Robust Crowd Counting Training Scheme

**Require:** Training images $I$ and corresponding crowd point annotations $A$

1. Train crowd counting model $F$ on $I$ and $A$
2. Make prediction on test images, get predicted points $B$
3. Calculate deviation vector $L$ between $B$ and $A$
4. Sort $L$ in ascending order, get sorted vector $S$
5. Generate mask vector $M$: For the first $[\theta N]$ elements in $S$, $M$ at corresponding position is 1, else 0
6. Calculate filtered loss $L_{\text{FF}} = \sum_{j}^{N} (M^j \cdot L^j)$
7. Update model $F$ using LFF as loss function
8. Update model $F$ using LFF as loss function
9. **while** model not converged **do**
10. Repeat steps 2–8
11. **end while**
12. **return** Updated model $F$

**Parameters:** $\theta$: Loss filtering ratio factor, $L$: Deviation vector between prediction and label, $M$: Mask vector, $S$: Sorted vector
Figure 3 presents the flowchart of the Low-Noise Robust Crowd Counting (LNRCC) framework. The process commences with the input of a crowd counting dataset, which feeds into a deep learning model exemplified by architectures such as VGG19 or HRNet. These models are adept at capturing complex spatial features from high-density crowd images. The convolutional layers, marked by their depth and the size of the convolutional kernels, progressively reduce the spatial dimensions while increasing the depth of feature maps. Following feature extraction, the flowchart delineates the application of a Bayesian loss function, $L_{\text{Bayes}}$, which is computed as a sum over the transformation of individual errors, $F(1 - E_{cn})$, to enhance the model’s robustness to noise. Subsequently, the overall loss function incorporates a Loss Filtering Factor (LFF), which is designed to mitigate the impact of noisy labels during training. The LFF is applied through a selective weighting mechanism, where certain data points are emphasized or disregarded based on their estimated noise levels. Finally, the denoising block represents the process of cleansing the output, further refining the count estimates by filtering out the noise-induced inconsistencies. This integrative approach facilitates a more accurate and noise-resilient crowd counting model, as evidenced by the reduced noise in the denoised output compared to the initial model predictions.

**Experiments**

In this section, we introduce our LFF in multiple networks with different datasets, the experimental results show the LFF can effectively enhance the crowd counting model robustness in label noise environment and improve its learning performance.
**Evaluation Metrics**

MAE and MSE are two extensively used metrics for evaluating crowd count estimate methods. They are defined as follows:

\[
MAE = \frac{1}{K} \sum_{k=1}^{K} |N_{k}^{GT} - N_{k}|
\]  

(4)

\[
MSE = \sqrt{\frac{1}{K} \sum_{k=1}^{K} |N_{k}^{GT} - N_{k}|^2}
\]  

(5)

where \( K \) is the number of test images, \( N_{k}^{GT} \) and \( N_{k} \) are the value of label count and the value of estimated count for the \( k \) – \( th \) image, respectively.

Another important metric is the GAME (Khan, Menouar, and Hamila 2023b). GAME aims to improve the assessment of localization in crowd counting. It divides each image into \( L^2 \) non-overlapping grids and calculates the MAE within each grid, summing up the errors. The GAME metric is particularly useful for evaluating the spatial distribution accuracy of the estimated count. The formula for GAME is given by:

\[
GAME(L) = \frac{1}{K} \sum_{k=1}^{K} \sum_{l=1}^{L^2} |N_{k,l}^{GT} - N_{k,l}|
\]  

(6)

where \( L^2 \) represents the total number of grids the image is divided into, \( K \) is the number of test images, \( N_{k,l}^{GT} \) is the ground truth count in the \( l \) – \( th \) grid of the \( k \) – \( th \) image, and \( N_{k,l} \) is the estimated count in the same grid.

**Datasets**

There are numerous crowd counting public datasets available nowadays. Based on the popularity of crowd counting datasets, ShanghaiTech (Zhang et al. 2016a), UCF-QNRF (Idrees et al. 2018), UCF_CC_50 (Idrees et al. 2013) will be used in this paper. Additionally, RGBT-CC (Liu et al. 2021), the first large-scale RGBT Crowd Counting benchmark, published in 2021, will also be used in this paper.

**ShanghaiTech**

(Zhang et al. 2016a) consists of parts A and B. Part A contains 300 training images and 182 testing images. At the same time, Part B includes 400 training images and 316 testing images. According to Table 1, Part A has a significantly higher density than Part B. It is the most popular Crowd Counting benchmark with more than 90% usage rate in relevant research.
**UCF-QNRF**
(Idrees et al. 2018) is one of the largest crowd counting datasets, with 1,535 images and 1.25 million point annotations. It is a difficult dataset to analyze since it has a wide variety of counts, image resolutions, lighting conditions, and viewpoints and contains a dense crowd. The training set includes 1,201 images, while the remaining 334 images are used for testing. It also has a usage rate of more than 70% in relevant research.

**Ucf_cc_50**
(Idrees et al. 2013) includes 50 grayscale images with varying but high resolutions. Each image has an average resolution of $2013 \times 2902$. The average count for each image is 1,279, and the minimum and maximum counts are 94 and 4,532, respectively. It is the second most popular Crowd Counting benchmark with more than 80% usage rate in relevant research.

**RGBT-CC**
(Liu et al. 2021) is the first publicly available RGBT dataset for crowd counting, containing 2,030 pairs of representative RGB-thermal images, 1,013 of which are captured in light and 1,017 of which are captured in darkness. Each image is the same size ($640 \times 480$), and 1,030 pairs are used for training, 200 pairs for validation, and 800 for testing. A total of 138,389 persons are marked with point annotations, on average 68 people per image.

**Neural Network Model**

In this experiment, we select several representative crowd counting models and add different Loss Filtering Factors to examine the efficiency of the proposed method.

**BL**
(Ma et al. 2019) is a loss function that builds a density contribution probability model using point annotations. Rather than restricting the value at each pixel in the density map, the BL training loss uses more reliable supervision on the count expectation at each annotated point. It significantly outperforms the baseline loss on many crowd counting datasets, including UCF-QNRF, ShanghaiTech, and UCF_CC_50, and exceeds the previous best approaches on the UCF-QNRF dataset. Additionally, many recent studies use BL as the loss function for their methods.

**DMCC**
(Wang et al. 2020) uses Distribution Matching for crowd counting. It uses the Optimal Transport method to evaluate the similarity between the normalized predicted density map and the normalized ground truth density map, as well
as a Total Variation loss to stabilize the OT computation. DMCC method also outperforms the previous state-of-the-art results on the ShanghaiTech and UCF_CC_50 datasets.

**CSCA** (Zhang, Choi, and Hong 2022) are modular building pieces that can be simply integrated into any modality-specific architecture. Through spatial-wise cross-modal attention, the CSCA blocks first spatially capture global functional connections across multimodality with less overhead. Then, cross-modal features with spatial attention are refined using adaptive channel-wise feature aggregation. This method greatly improves performance across various backbone networks and outperforms the previous state-of-the-art results on the RGBT-CC dataset.

**IADM** (Liu et al. 2021) is a crossmodal collaborative representation learning framework, which consists of multiple modality-specific branches, a modality-shared branch, and an Information Aggregation Distribution Module to capture the complementary information of different modalities fully. It incorporates two collaborative information transfers to dynamically enhance the modality-shared and modality-specific representations with a dual information propagation mechanism. Moreover, this method is universal for multimodal crowd counting, and experiment results demonstrate its effectiveness.

Figure 4 illustrates the detailed architecture of the VGG19 model, a deep convolutional neural network widely utilized in the fields of image recognition and processing. The VGG19 model comprises 19 layers, including 16 convolutional layers and 3 fully connected layers. A distinctive feature of this model is the use of numerous small-sized (3×3) convolutional kernels, allowing the network to learn image features more deeply while maintaining the receptive field. Multiple pooling layers are interspersed between the convolutional layers, serving to reduce dimensions and computational load. The VGG19 demonstrates exceptional performance in image recognition tasks, owing its effectiveness to its deep structure and the application of small-sized convolutional kernels.

**Implementation Details**

In this experiment, we use the standard image classification network VGG-19 as the backbone and build the model respectively according to the official implementation of the methods in Sec.4.3. The backbone is pre-trained on ImageNet, and the Adam optimizer with an initial learning rate $10^{-5}$ is used to update the parameters.
For training, images from various datasets are randomly cropped into different sizes. The crop size is $256 \times 256$ for ShanghaiTech Part A, UCF_CC_50 and RGBT-CC where images resolutions are smaller, and $512 \times 512$ for ShanghaiTech Part B and UCF-QNRF. And we perform five-fold cross validations to obtain the average test result of the UCF_CC_50 dataset since it is a small-scale dataset with no data split designated for training and testing.

**Experimental Evaluations**

We compare the models with and without using the proposed Loss Filtering Factor on the benchmark datasets described in Sec.4.2. We set the model without Loss Filtering Factor ($\theta = 1$) as the baseline and compare models’ performances at various Loss Filtering Factor values. To make a fair comparison, for the same model, the same random seeds are used for each set of experiments and other parameters. The experimental results are shown in Table 2.

We conduct five experiments on each dataset separately for each model structure, including a model without the proposed method ($\theta = 1$) and four models using the Loss Filtering Factor with different values of $\theta$.

**Quantitative Results**

In all experiments, the proposed Loss Filtering Factor consistently improves the performances of all four models in all datasets used in experiments by an average of 3.68% for MAE, 6.7% for MSE and 4.68% for GAME. For most models and datasets, using the Loss Filtering Factor with a $\theta$ of 0.9 or 0.95 may increase the best performance. When $\theta = 0.9$, it provides the greatest performance improvement, boosting the counting accuracy of DMCC on the ShanghaiTechB dataset by 11.38% and 9.12% for MAE and MSE, respectively.

From the perspective of the model, using the appropriate Loss Filtering Factor makes around 6.81% improvements in DMCC and 5.27% of BL on all
Table 2. (a): benchmark evaluations on five benchmark crowd counting datasets using the MAE, MSE and GAME(2) metrics. All of the models use the VGG19 neural network.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ShanghaiTechA</th>
<th>ShanghaiTechB</th>
<th>UCF-QNRF</th>
<th>UCF-CC-50</th>
<th>RGBT-CC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>θ</td>
<td>MAE</td>
<td>MSE</td>
<td>GAME</td>
<td>θ</td>
</tr>
<tr>
<td>1.00</td>
<td>61.16</td>
<td>100.85</td>
<td>205.29</td>
<td>8.53</td>
<td>13.29</td>
</tr>
<tr>
<td>0.95</td>
<td><strong>58.98</strong></td>
<td><strong>95.33</strong></td>
<td><strong>180.62</strong></td>
<td>8.12</td>
<td><strong>12.76</strong></td>
</tr>
<tr>
<td>0.90</td>
<td>62.86</td>
<td>99.77</td>
<td>189.84</td>
<td><strong>7.47</strong></td>
<td>12.99</td>
</tr>
<tr>
<td>0.85</td>
<td>64.85</td>
<td>100.4</td>
<td>194.92</td>
<td>8.21</td>
<td>12.52</td>
</tr>
<tr>
<td>0.80</td>
<td>66.23</td>
<td>100.8</td>
<td>199.20</td>
<td>8.61</td>
<td>13.60</td>
</tr>
<tr>
<td>BL</td>
<td>0.75</td>
<td>67.43</td>
<td>102.3</td>
<td>219.22</td>
<td>10.04</td>
</tr>
<tr>
<td>1.00</td>
<td>62.08</td>
<td>98.59</td>
<td>201.12</td>
<td>7.91</td>
<td>14.91</td>
</tr>
<tr>
<td>0.95</td>
<td>65.28</td>
<td>97.96</td>
<td>190.51</td>
<td>7.54</td>
<td>16.65</td>
</tr>
<tr>
<td>0.90</td>
<td><strong>61.33</strong></td>
<td><strong>96.14</strong></td>
<td><strong>185.69</strong></td>
<td><strong>7.01</strong></td>
<td><strong>13.55</strong></td>
</tr>
<tr>
<td>0.85</td>
<td>65.24</td>
<td>99.60</td>
<td>206.51</td>
<td>7.22</td>
<td>13.86</td>
</tr>
<tr>
<td>0.80</td>
<td>67.88</td>
<td>104.02</td>
<td>212.59</td>
<td>7.24</td>
<td>13.75</td>
</tr>
<tr>
<td>DMCC</td>
<td>0.75</td>
<td>70.42</td>
<td>109.86</td>
<td>221.08</td>
<td>8.33</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RGBT-CC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>θ</td>
</tr>
<tr>
<td>CSCA</td>
<td>1.00</td>
</tr>
<tr>
<td>0.95</td>
<td><strong>14.35</strong></td>
</tr>
<tr>
<td>0.90</td>
<td>14.57</td>
</tr>
<tr>
<td>0.85</td>
<td>14.75</td>
</tr>
<tr>
<td>0.80</td>
<td>15.51</td>
</tr>
<tr>
<td>0.75</td>
<td>16.40</td>
</tr>
<tr>
<td>IADM</td>
<td>1.00</td>
</tr>
<tr>
<td>0.95</td>
<td>15.52</td>
</tr>
<tr>
<td>0.90</td>
<td><strong>15.20</strong></td>
</tr>
<tr>
<td>0.85</td>
<td>15.36</td>
</tr>
<tr>
<td>0.80</td>
<td>16.06</td>
</tr>
<tr>
<td>0.75</td>
<td>17.18</td>
</tr>
</tbody>
</table>

The two models in (b) correspond to the RGB image datasets, and the datasets other than the RGBT-C dataset do not provide additional RGB images, so the CSCA and IADM models are only evaluated for the RGBT-C dataset. The best performance is shown in **underlined**.
Table 3. For the BL model, we modified the backbone network by replacing VGG19 with HRNet and ResNeXt, respectively, to test the generalization and validity of LFF.

<table>
<thead>
<tr>
<th>Model</th>
<th>$\theta$</th>
<th>MAE</th>
<th>MSE</th>
<th>GAME</th>
<th>MAE</th>
<th>MSE</th>
<th>GAME</th>
<th>MAE</th>
<th>MSE</th>
<th>GAME</th>
<th>MAE</th>
<th>MSE</th>
<th>GAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRNet</td>
<td>1.00</td>
<td>53.34</td>
<td>89.29</td>
<td>185.10</td>
<td>8.20</td>
<td>11.30</td>
<td>30.58</td>
<td>83.42</td>
<td>143.26</td>
<td>294.21</td>
<td>219.32</td>
<td>301.35</td>
<td>632.43</td>
</tr>
<tr>
<td></td>
<td>0.95</td>
<td>46.77</td>
<td>84.20</td>
<td>176.49</td>
<td>7.45</td>
<td>10.83</td>
<td>27.30</td>
<td>79.42</td>
<td>139.02</td>
<td>281.17</td>
<td>209.26</td>
<td>296.29</td>
<td>51.29</td>
</tr>
<tr>
<td></td>
<td>0.90</td>
<td>49.33</td>
<td>87.39</td>
<td>184.89</td>
<td>7.97</td>
<td>11.25</td>
<td>28.48</td>
<td>76.40</td>
<td>135.81</td>
<td>277.43</td>
<td>204.23</td>
<td>287.30</td>
<td>54.63</td>
</tr>
<tr>
<td></td>
<td>0.85</td>
<td>56.05</td>
<td>89.24</td>
<td>191.48</td>
<td>8.19</td>
<td>12.93</td>
<td>29.99</td>
<td>81.07</td>
<td>142.49</td>
<td>283.51</td>
<td>212.32</td>
<td>294.84</td>
<td>59.79</td>
</tr>
<tr>
<td></td>
<td>0.80</td>
<td>60.28</td>
<td>95.19</td>
<td>193.00</td>
<td>8.00</td>
<td>12.80</td>
<td>32.51</td>
<td>84.06</td>
<td>146.53</td>
<td>289.94</td>
<td>220.45</td>
<td>300.59</td>
<td>63.24</td>
</tr>
<tr>
<td></td>
<td>0.75</td>
<td>64.93</td>
<td>102.03</td>
<td>196.82</td>
<td>9.15</td>
<td>13.03</td>
<td>36.32</td>
<td>88.32</td>
<td>151.12</td>
<td>297.36</td>
<td>225.83</td>
<td>309.64</td>
<td>67.84</td>
</tr>
<tr>
<td>ResNeXt</td>
<td>1.00</td>
<td>96.08</td>
<td>204.49</td>
<td>549.92</td>
<td>30.62</td>
<td>107.39</td>
<td>254.26</td>
<td>172.49</td>
<td>385.71</td>
<td>730.35</td>
<td>489.36</td>
<td>931.32</td>
<td>324.58</td>
</tr>
<tr>
<td></td>
<td>0.95</td>
<td>94.98</td>
<td>200.32</td>
<td>530.28</td>
<td>31.94</td>
<td>114.37</td>
<td>242.40</td>
<td>163.62</td>
<td>373.37</td>
<td>724.58</td>
<td>480.56</td>
<td>924.49</td>
<td>319.72</td>
</tr>
<tr>
<td></td>
<td>0.90</td>
<td>95.50</td>
<td>205.33</td>
<td>538.68</td>
<td>30.11</td>
<td>110.95</td>
<td>236.01</td>
<td>157.63</td>
<td>364.63</td>
<td>721.51</td>
<td>488.43</td>
<td>942.68</td>
<td>321.07</td>
</tr>
<tr>
<td></td>
<td>0.85</td>
<td>100.20</td>
<td>215.70</td>
<td>547.21</td>
<td>34.48</td>
<td>116.26</td>
<td>240.67</td>
<td>167.85</td>
<td>369.08</td>
<td>759.79</td>
<td>497.13</td>
<td>947.56</td>
<td>330.58</td>
</tr>
<tr>
<td></td>
<td>0.80</td>
<td>109.88</td>
<td>231.48</td>
<td>553.09</td>
<td>37.21</td>
<td>123.57</td>
<td>254.12</td>
<td>174.21</td>
<td>379.39</td>
<td>772.50</td>
<td>505.42</td>
<td>969.99</td>
<td>339.48</td>
</tr>
<tr>
<td></td>
<td>0.75</td>
<td>115.42</td>
<td>239.52</td>
<td>2570.63</td>
<td>40.32</td>
<td>130.57</td>
<td>264.39</td>
<td>180.75</td>
<td>392.41</td>
<td>779.57</td>
<td>512.79</td>
<td>986.07</td>
<td>353.59</td>
</tr>
</tbody>
</table>
five datasets. It makes around 5.91% improvements of CSCA and 5.11% of IADM on RGBT-CC datasets.

From the perspective of the dataset, using the appropriate Loss Filtering Factor makes an average of 5.36% improvements of models trained on UCF-QNRF, 3.18% on ShanghaiTechA, 9.77% on ShanghaiTechB, 5.93% on UCF_CC_50 and 5.74% on RGBT-CC, respectively.

In order to compare the impact of different networks on the LNRCC scheme, we conducted experiments to replace the backbone network. Our experiments results in Table 3 indicate that while HRNet integration with our LNRCC scheme showed significant performance improvements, ResNeXt did not achieve similar success, facing issues with error margins and convergence. This highlights the necessity for model-specific adaptations of our methods, as evidenced by the effectiveness of VGG19 and the adaptability of LFF across various network architectures.

**Key Issues and Discussion**

The change trends of MAE, MSE and GAME of model BL with different θ values compared to the baseline model are shown in Figure 5 (Upper left). As θ decreases, MAE first drops then rises, reaching the lowest point at θ = 0.95 and improving 2.2% compared to the baseline model. MSE exhibits a similar trend, hitting the lowest point at θ = 0.95 with a 4% improvement. GAME also follows a similar pattern, with the lowest value at θ = 0.90, indicating a 4.2% decrease compared to the baseline. In Figure 5 (Lower left), the change trends of MAE and MSE of model DMCC with varying θ values against the baseline model are presented. Both MAE and MSE first fall then climb as θ reduces, sharing the same lowest point at θ = 0.90. MAE sees a 6.4% increase and MSE has a 6.8% gain. GAME trends similarly, reaching its minimum at θ = 0.90 with a 3.4% improvement over the baseline. Figure 5 (Upper right) illustrates how MAE and MSE of model CSCA alter with different θ values versus the baseline model. The variations of MAE and MSE are analogous, declining initially then rising when θ decreases. The lowest spots emerge at θ = 0.95 and θ = 0.90 separately. MAE gains 3.1% while MSE increases by 8.7%. For GAME, the minimum value is attained at θ = 0.95, which is 3.7% higher than the baseline. The change trends of MAE and MSE of model IADM across different θ values in contrast to the baseline model are depicted in Figure 5 (Lower right). MAE and MSE exhibit similar tendencies, both dropping first and then growing as θ reduces, with the lowest point occurring at θ = 0.90. MAE improves by 3% and MSE by 7.3%. GAME follows a similar pattern, the most substantial reduction of 7.4% happening at θ = 0.90.
Effect of $\theta$. To analyze the impact of $\theta$ selection on model performance, we divide the results of each model with Loss Filtering Factor($\theta \neq 1$) by the corresponding result of the baseline model($\theta = 1$) and take the average. The result is shown in Figure 6.

Although the best value of $\theta$ varies for different models and datasets, the proposed Loss Filtering Factor increases model performance when the $\theta$ are

**Figure 5.** The illusion of the effectiveness of parameter $\theta$ under LFF in different models.

**Figure 6.** Effect of loss filtering factor $\theta$. 
between 0.85 and 1, and for most models, selecting the value of \( \theta \) to 0.95 or 0.9 can maximize the model’s performance. However, as \( \theta \) is selected smaller, the performance of models decreases significantly.

The effect of \( \theta \) can be explained in the following ways:

- There are about 10% point annotations in the dataset, which will negatively influence the model’s performance in counting when adopted in training. In other words, there is about 10% label noise in the dataset, and reducing this label noise during training can improve the model performance in counting. This may also explain why the best \( \theta \) choices for the same model vary across datasets because the levels of label noise vary among datasets.
- By using the proposed Loss Filtering Factor, the model will not use all data points for supervision in each training epoch, similar to the dropout layer in the neural network model, and even if the data removed from training by a factor \( \theta \) is not the label noise, it can still improve generalization and decrease the overfitting of the model to training data.
- Setting \( \theta \) too small will filter out too much training data. When \( \theta \) is extremely small, only a tiny portion of data may be selected for training during each iteration. In other words, the model can only see partial information of the training data each time, making it hard to grasp the overall distribution. This will lead to the model’s inability to sufficiently learn the patterns in the training data, resulting in underfitting. The experimental results show that when \( \theta \) is 0.75, the performance of most models is inferior compared to \( \theta \) values of 0.85–0.95, indicating that a too small \( \theta \) causes underfitting. In general, regularization techniques improve generalization capability while sacrificing fitting accuracy on the training data. Only by finding a balancing point can both overfitting and underfitting be avoided. An excessively small \( \theta \) disrupts this balance by over-regularizing, thus leading to underfitting.

The above explanation shows that a suitable \( \theta \) can significantly increase model performance because it can reduce the dataset’s label noise during training, improve generalization, and decrease the overfitting of the model. However, a too small \( \theta \) (\( \theta \leq 0.80 \)) results in lower model accuracy due to insufficient use of ground truth, and the model is poorly supervised.

**Effect on Model Convergence Speed**

Through experiments, we observe that after adding the Loss Filtering Factor, the model’s training time keeps relatively constant for each epoch. The model’s convergence speed, on the other hand, is slow and varies widely between models. In general, the total number of epochs required to complete the training for the model using the Loss Filtering Factor is around 5% more than for the model without using it.
Ablation Studies

We perform the ablation study on the UCF-QNRF dataset by comparing the proposed Loss Filtering Factor with removing the same number of points randomly during the training. Table 4 provides quantitative results of it.

When 10% of the data were randomly removed to supervision in training, The overall performance of the model improved slightly, but the increase was tiny. It indicates that randomly removing the data supervision in each epoch during the training process has limited influence on model performance, showing that the proposed Loss Filtering Factor improves model performance by filtering label noise that will negatively influence the performance of the model in counting instead of just removing some of data supervision randomly in training.

Conclusions

In this work, we propose the LFF and design the corresponding framework, namely the LNRCC training scheme to mitigate the label noise impact in crowd counting learning tasks. Our proposed approach allows models to filter out likely noise-induced losses during training, enabling them to focus on more reliable signals in the data. We evaluate the LNRCC using crowd images in multiple scenarios like malls, airports and streets. The experimental results show our LNRCC can effectively improve the learning performance of crowd counting models and mitigate the noise label impact in such tasks. As the existing LNRCC relies on manually setting, introducing reinforcement learning in LNRCC and achieving self-tune could be an appealing research trend in the future crowd analytics.

Disclosure statement

This paper does not have potential conflict of interest.
Data availability statement

The data that support the findings of this study are openly available in ShanghaiTech shanghaitech, UCF-QNRF QNRF, UCF_CC_50 Idrees_2013.CVPR and RGBT-CC RGBTCC.

References


