TANGO: Text-Anchored Guided Optimization for Robust Fine-tuning Vision-Language Models under Label Noise

Supplementary Material

7. Datasets

We conduct extensive experiments on two standard benchmarks with synthetic label noise, CIFAR-100 [30] and Tiny-ImageNet [31], and on four real-world noisy datasets: CIFAR-100N [53], WebVision [37], Food-101N [32], and Animal-10N [48].

7.1. Synthetic Noise

Following established setups [41, 44, 56], we evaluate TANGO under three types of synthetic label noise: symmetric, asymmetric, and instance-dependent.

Symmetric Noise: This noise is generated by uniformly corrupting labels across all classes, meaning each label has an equal probability of being flipped to any other incorrect class.

Asymmetric Noise: This approach uses a semantically-aware corruption scheme, where labels are intentionally misassigned to visually similar categories. We follow the same schema as in prior works [28, 41]. Specifically for CIFAR-100, its 100 classes are grouped into 20 semantic super-classes, and circular replacement rules are applied within each super-class to create the noise.

Instance-Dependent Noise: Introduced by [56], this noise model provides a more realistic scenario where the probability of label corruption is dependent on the features of the instance itself.

7.2. Real-World Noise

We evaluate our method on four real-world datasets with naturally occurring label noise.

- **CIFAR-100N** [53] is a version of the CIFAR-100 dataset that includes noisy labels annotated by humans via Amazon Mechanical Turk. It has an approximate noise rate of 40%.
- **WebVision** [37] contains 2.4 million images crawled from the web, with an estimated noise rate of 30% (25% out-of-distribution and 5% in-distribution noise) [1]. Following common practice [41], we train our models on the mini-WebVision subset, which includes the first 50 classes from the Google Images subset (approximately 66,000 images). We evaluate performance on both the WebVision and the ImageNet ILSVRC12 validation sets.
- **Food-101N** [32] is a large-scale dataset used as a reliable benchmark, chosen as an alternative to Clothing1M [58] which has known issues that can lead to biased evaluations [37]. Food-101N contains 310,009 images of food recipes across 101 categories with an estimated noise rate of 20%. While it shares the same classes as the original Food-101 dataset [2], it is larger and has a higher level of noise. Following prior work [32], we report performance on the clean Food-101 test set.
- **Animal-10N** [48] consists of 50,000 training images and 10,000 test images distributed across 10 animal classes. The noise in this dataset, estimated to be around 8%, originates from the manual labeling process.

8. Implementation Details for Visual-Only Baselines

As referenced in the main paper, our sample refinement mechanism fuses a visually-derived signal with a semantically-derived one. To provide a clear context for our contributions and for the ablation studies, we detail the computation of the purely visual components below. These mechanisms are analogous to their cross-modal counterparts but operate exclusively within the visual feature space of the training samples.

8.1. Visually-Derived Consistency Distribution

For the task of sample selection, a consistency score distribution $\boldsymbol{p}_i^{\text{vis}} \in [0,1]^C$ is computed for each training sample $\boldsymbol{x}_i$. This vector represents a full probability distribution over the C classes, as determined by the consensus of the sample's neighbors in the visual feature space. This approach is more robust than a single score as it captures the nuanced support for all classes within the local neighborhood.

The computation proceeds as follows:

1. k-NN Identification: For each sample's visual feature $\boldsymbol{v}_i = f_v(\boldsymbol{x}_i)$, we first identify its set of k nearest neighbors within the entire training set, denoted as $\mathcal{N}(i)$.

2. Weighted Label Aggregation: We aggregate the noisy labels of these neighbors. To counteract the bias introduced by class imbalance in the dataset, each neighbor's vote is weighted inversely to its class frequency. First, we compute the class distribution vector $\boldsymbol{\pi} \in \mathbb{R}^C$, where each element π_c is the total number of training samples with the noisy label c. Then, for each sample $\boldsymbol{x}_i$, we compute an unnormalized score vector $\boldsymbol{s}_i \in \mathbb{R}^C$:

$$\boldsymbol{s}_i = \sum_{j \in \mathcal{N}(i)} \frac{\hat{\boldsymbol{y}}_j}{\boldsymbol{\pi}}, \tag{9}$$

where $\hat{\boldsymbol{y}}_j$ is the one-hot vector corresponding to the noisy label of neighbor $\boldsymbol{x}_j$, and the division is performed element-

wise. This ensures that votes from minority classes are given higher significance.

3. Normalization: The resulting score vector $\boldsymbol{s}_i$ is then L1-normalized to produce the final probability distribution $\boldsymbol{p}_i^{\text{vis}}$:

$$\boldsymbol{p}_i^{\text{vis}} = \frac{\boldsymbol{s}_i}{\|\boldsymbol{s}_i\|_1}. \tag{10}$$

This vector $\boldsymbol{p}_i^{\text{vis}}$ serves as the visually-derived soft label, providing a balanced consensus from the local neighborhood that is subsequently fused with the semantic signal.

8.2. Visually-Propagated Soft Labels

To generate corrected pseudo-labels for the regularization term, we employ a label propagation strategy that aggregates information from a sample's local neighborhood. This approach is inspired by methods that leverage structural consistency within the data manifold [27]. Specifically, we adopt a reverse k-NN scheme to generate robust soft labels.

Instead of a sample inheriting labels from its neighbors, it receives label information from all the other samples that consider *it* to be a neighbor. This "reverse" flow helps mitigate the influence of outliers.

First, we construct a binary visual neighborhood graph, $\mathbf{A}^{\text{vv}} \in \{0,1\}^{N\times N}$, where N is the number of training samples. An element $(\mathbf{A}^{\text{vv}})_{ij} = 1$ if sample $\boldsymbol{x}_j$ is among the k-nearest neighbors of sample $\boldsymbol{x}_i$, and 0 otherwise.

Next, we compute an aggregated score matrix $\mathbf{S}^{\text{vis}}$ by multiplying the transpose of the neighborhood graph with the one-hot noisy label matrix $\mathbf{Y}_{\text{noisy}} \in \{0,1\}^{N\times C}$:

$$\mathbf{S}^{\text{vis}} = (\mathbf{A}^{\text{vv}})^\top \mathbf{Y}_{\text{noisy}}. \tag{11}$$

Each row $(\mathbf{S}^{\text{vis}})_i$ now represents the sum of the one-hot labels of all samples for which $\boldsymbol{x}_i$ is a neighbor.

Finally, the matrix of visually-propagated soft labels, $\mathbf{Y}^{\text{vis}} \in [0,1]^{N\times C}$, is obtained by row-normalizing $\mathbf{S}^{\text{vis}}$:

$$\mathbf{Y}^{\text{vis}} = (\mathbf{D}^{\text{vis}})^{-1}\mathbf{S}^{\text{vis}}, \tag{12}$$

where $\mathbf{D}^{\text{vis}}$ is a diagonal matrix whose elements $D_{ii} = \sum_j (\mathbf{A}^{\text{vv}})_{ji}$ contain the in-degree of each node—that is, the number of samples that selected sample $\boldsymbol{x}_i$ as a neighbor. This matrix $\mathbf{Y}^{\text{vis}}$ provides a soft label distribution for each sample, refined by the consensus of its local visual context

9. Additional Analysis

To provide a more comprehensive validation of our TANGO framework, we present further analyses that supplement the results in the main paper. Specifically, we evaluate the computational efficiency of our method, provide a quantitative assessment of our sample selection performance, measure the quality of the pseudo-labels generated by our relabeling mechanism, and test the framework's generalizability on a different model backbone.

9.1. Computational Cost Analysis

Regarding computational cost per iteration, Table 8 shows that TANGO consistently operates faster than LSL [27] but slightly slower than SSR [12] across all datasets. Crucially, TANGO requires significantly less runtime compared to the co-teaching-based DivideMix [33], highlighting its practical efficiency for training.

9.2. Sample Selection Performance

We evaluated the performance of our selection mechanism on the CIFAR100N dataset, which has ground-truth labels available for the real-world noisy training set. We use four standard metrics: Area Under the ROC Curve (AUROC), Precision, Recall, and F1 Score. As shown in Table 9, TANGO demonstrates state-of-the-art performance across the most critical metrics. It achieves the highest AUROC (94.43), indicating a superior ability to discriminate between clean and noisy samples compared to all baselines. Furthermore, TANGO obtains the highest Precision (88.97) and F1 Score (94.42). This highlights that our method not only correctly identifies clean samples but also achieves an excellent balance between precision and recall. While the SSR method achieves a slightly higher Recall, its lower Precision suggests that it does so at the cost of including more noisy samples in its selected "clean" set. TANGO's top-ranking F1 Score confirms it provides the best trade-off, validating that our cross-modal consistency check is more reliable than purely uni-modal heuristics.

9.3. Relabeling Performance Analysis

The quality of the pseudo-labels used for regularization is critical to the success of any LNL method. We measured the ground-truth accuracy of the labels generated by our full refinement mechanism on the CIFAR100N dataset and compared it against strong baselines. Our anchor-guided approach, TANGO, achieves a relabeling accuracy of **84.26%**. This significantly surpasses the performance of LSL (73.74%). This substantial improvement of over 10% compared to LSL underscores the immense benefit of injecting clean, external semantic information into the label correction process, validating a core premise of our work.

10. Detailed Training Procedure

As discussed in the main paper, by integrating our anchor-guided refinement mechanism into the standard learning-with-noisy-labels pipeline, the visual encoder is progressively trained with higher-quality supervision signals. The complete training procedure of TANGO is formally summarized in Algorithm 1.

Table 8. Computational cost (in seconds) per epoch. The analysis covers four key stages: feature extraction, sample selection, relabeling, and model training. TANGO demonstrates superior efficiency with a total training time significantly lower than key baselines like LSL and DivideMix, validating its practicality for large-scale use.

Dataset	Method	Feature Extraction	Sample Selection	Relabeling	Model Training	Total Time
CIFAR100	DivideMix	66.35	3.60	0.00	492.89	562.84
	SSR	30.13	0.72	0.03	75.54	106.42
	LSL	30.09	0.73	0.19	278.08	309.09
	TANGO	**29.64**	**0.43**	**0.27**	**175.97**	**206.31**
Tiny-ImageNet	DivideMix	131.52	5.02	0.00	1025.31	1161.85
	SSR	58.19	1.57	0.03	142.01	201.80
	LSL	58.48	1.54	0.59	546.82	607.43
	TANGO	**58.26**	**1.61**	**0.78**	**352.19**	**412.84**
WebVision	DivideMix	77.10	6.32	0.00	602.82	686.24
	SSR	39.10	0.74	0.03	156.66	196.53
	LSL	35.99	0.68	0.22	344.19	381.08
	TANGO	**35.01**	**0.30**	**0.53**	**220.63**	**256.47**
Food101N	DivideMix	484.01	6.35	0.00	3558.30	4048.66
	SSR	231.51	4.29	0.03	746.08	981.91
	LSL	223.32	4.10	4.46	1687.97	1919.85
	TANGO	**234.44**	**4.62**	**4.28**	**1080.13**	**1323.47**

Table 9. Quantitative comparison of sample selection performance on CIFAR100N. TANGO achieves the highest AUROC, Precision, and F1 Score, demonstrating the best overall balance for accurately identifying clean samples.

Method	CIFAR100N			
	AUROC	Precision	Recall	F1 Score
Label-match	81.30	84.37	88.39	86.33
Small-loss	84.01	84.87	82.14	83.49
SSR	92.20	86.94	**96.72**	91.57
AUM	89.04	83.25	92.99	87.85
DeFT	/	88.43	92.84	90.58
TANGO	**94.43**	**88.97**	96.38	**94.42**

Algorithm 1 TANGO Training Procedure

Input: Noisy dataset $\mathcal{D}_{\text{noisy}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, visual encoder f_v, frozen text encoder f_t, hyperparameters.
— Pre-computation —
Pre-compute semantic anchor set $\mathcal{A}$ using f_t.
for each training epoch $e = 1, ..., E$ **do**
  # — Epoch-level Sample Refinement —
  Extract features for all samples: $\boldsymbol{V} = \{f_v(\boldsymbol{x}_i)\}_{i=1}^{N}$ (using the model from epoch $e-1$).
  Generate clean set indices $\mathcal{I}_c$ and corrected labels $\tilde{\mathbf{Y}}$ via Anchor-Guided Sample Refinement using $\boldsymbol{V}$ and $\mathcal{A}$.
  # — Batch-level Model Training —
  for each batch $\{\boldsymbol{x}_b, \boldsymbol{y}_b\}_{b=1}^{B} \subset \mathcal{D}_{\text{noisy}}$ **do**
    Get predictions $\{\boldsymbol{l}_b\}$ using the TAC on features from the current model f_v.
    Identify clean samples in batch: $\mathcal{B}_c = \{b \mid \text{index}(b) \in \mathcal{I}_c\}$.
    Calculate supervised loss L_{clean} on $\mathcal{B}_c$.
    Get corrected labels for the batch: $\{\tilde{\boldsymbol{y}}_b\}$.
    Calculate regularization loss L_{reg} on the entire batch.
    Compute total loss $L = L_{\text{clean}} + L_{\text{reg}}$.
    Update visual encoder parameters θ_v via back-propagation on L.
  end for
end for