IterIS: Iterative Inference-Solving Alignment for LoRA Merging

Supplementary Material

In the supplementary materials, we provide:

- **Mathematical Proofs** (Section A): We present the mathematical proofs for the equations and properties of our proposed algorithm, as discussed in the main text.
- **Datasets and Experimental Setup** (Section B): This section details the datasets employed in our experiments, the LoRA training configuration, and the key settings of the baseline implementations.
- **Ablation Studies** (Section C): We report the results of ablation studies, demonstrating the necessity of the three mechanisms integrated into our algorithm.
- **Experimental Results** (Section D): A detailed presentation of the experimental results is provided in this section.
- **Workflow and Analysis** (Section E): Finally, we illustrate the complete workflow of IterIS and conduct an analysis of the efficiency and limitations of our approach.

A. Detailed Mathematical Derivation

A.1. Linear Merging

Consider the following optimization problem:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \mathbb{E}_{\mathcal{X}}[\sum_{i=1}^{N} \lambda_i \|\boldsymbol{W}_i^T \boldsymbol{\mathcal{X}}_i - \boldsymbol{W}^T \boldsymbol{\mathcal{X}}_i\|_F^2] \tag{9}$$

where $\mathbb{E}_{\mathcal{X}}[\cdot]$ denotes the expectation with respect to $\mathcal{X}$ ($\mathcal{X}$=($\mathcal{X}_1, \ldots, \mathcal{X}_N$)), and λ_i is a constant. and $\|\cdot\|_F$ represents the Frobenius norm. Assuming each $\boldsymbol{\mathcal{X}}_i$ follows an isotropic distribution and $\boldsymbol{\mathcal{X}}_1 \ldots \boldsymbol{\mathcal{X}}_N$ are mutually independent, we have:

$$\mathbb{E}_{\mathcal{X}}[\|\boldsymbol{W}_i^T \mathcal{X}_i - \boldsymbol{W}^T \mathcal{X}_i\|_F^2] = \frac{1}{D} \mathbb{E}_{\mathcal{X}_i}[\|\mathcal{X}_i\|_F^2] \|\boldsymbol{W}_i - \boldsymbol{W}\|_F^2 \tag{10}$$

where D is the dimension of each $\mathcal{X}_i$. we have the following based on Eq. 10:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \mathbb{E}_{\mathcal{X}}[\sum_{i=1}^{N} \lambda_i \|\boldsymbol{W}_i^T \mathcal{X}_i - \boldsymbol{W}^T \mathcal{X}_i\|_F^2] \tag{11}$$

$$= \arg\min_{\boldsymbol{W}} \sum_{i=1}^{N} \mathbb{E}_{\mathcal{X}_i}[\lambda_i \|\boldsymbol{W}_i^T \mathcal{X}_i - \boldsymbol{W}^T \mathcal{X}_i\|_F^2] \tag{12}$$

$$= \arg\min_{\boldsymbol{W}} \sum_{i=1}^{N} \frac{\lambda_i}{D} \mathbb{E}_{\mathcal{X}_i}\left[\|\mathcal{X}_i\|_F^2\right] \|\boldsymbol{W}_i - \boldsymbol{W}\|_F^2 \tag{13}$$

This expression admits a closed-form solution.

$$\boldsymbol{W}^* = \sum_{i=1}^{N} \tilde{\lambda}_i \boldsymbol{W}_i, \tilde{\lambda}_i = \frac{\lambda_i \mathbb{E}_{\mathcal{X}_i}[\|\mathcal{X}_i\|_F^2]}{\Sigma_{j=1}^{N} \lambda_j \mathbb{E}_{\mathcal{X}_j}[\|\mathcal{X}_j\|_F^2]} \tag{14}$$

A.2. Real-distribution-based Merging

We can derive the following from the linear merging derivation in Subsection A.1, and set each λ_i to 1:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{\mathcal{X}}_i}[\|\boldsymbol{W}_i^T \boldsymbol{\mathcal{X}}_i - \boldsymbol{W}^T \boldsymbol{\mathcal{X}}_i\|_F^2]. \tag{15}$$

To compute each expectation in Eq. 15, we can sample from the distribution of $\boldsymbol{\mathcal{X}}_i$, and using the law of large numbers, approximate the expectation as:

$$\mathbb{E}_{\boldsymbol{\mathcal{X}}_i}[\|\boldsymbol{W}_i^T \boldsymbol{\mathcal{X}}_i - \boldsymbol{W}^T \boldsymbol{\mathcal{X}}_i\|_F^2] \approx \frac{1}{S} \sum_{s=1}^{S} \|\boldsymbol{W}_i^T \boldsymbol{x}_{is} - \boldsymbol{W}^T \boldsymbol{x}_{is}\|_F^2 \tag{16}$$

$$= \frac{1}{S} \|\boldsymbol{W}_i^T \boldsymbol{X}_i - \boldsymbol{W}^T \boldsymbol{X}_i\|_F^2 \tag{17}$$

where $\boldsymbol{x}_{is} \in \text{Sample}(\boldsymbol{\mathcal{X}}_i)$ and $\boldsymbol{X}_i = (\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{iS})$. Therefore, we can reformulate the optimization objective based on the real distribution:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \sum_{i=1}^{N} \|\boldsymbol{W}_i^T \boldsymbol{X}_i - \boldsymbol{W}^T \boldsymbol{X}_i\|_F^2. \tag{18}$$

In fact, this optimization problem can be viewed as a linear regression problem, where $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_N]^T$ is mapped to $\boldsymbol{Y} = [\boldsymbol{W}_1^T \boldsymbol{X}_1, \boldsymbol{W}_2^T \boldsymbol{X}_2, \cdots, \boldsymbol{W}_N^T \boldsymbol{X}_N]^T$. Thus, the problem has a closed-form solution:

$$\boldsymbol{W}^* = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y} \tag{19}$$

$$= (\sum_{i=1}^{N} \boldsymbol{X}_i \boldsymbol{X}_i^T)^{-1} (\sum_{i=1}^{N} \boldsymbol{X}_i \boldsymbol{X}_i^T \boldsymbol{W}_i) \tag{20}$$

Similarly, for the optimization problem corresponding to our algorithm in Eq. 5 of Section 3, we can also obtain its solution.

A.3. Maximum Iteration Limit for IterIS

Starting with the topology of the generative model, we construct the gradient computation graph G_∇, which is a directed acyclic graph (DAG). To create the new graph G_M, we extract all nodes corresponding to the input features for each LoRA. We follow a specific rule to construct G_M: if there is a directed path from node A to node B in G_∇, we establish a directed edge from A to B in G_M. The resulting graph G_M is also a directed acyclic graph. We will demonstrate that *the length of the longest directed path in G_M, originating from any input node (of which there may be multiple), minus one, provides the maximum number of iterations required for the IterIS's convergence.*

Proof: Let s denote the length of the longest path in G_M originating from the input node O_{in}. We define a function $g : v(G_M) \setminus \{O_{\text{in}}\} \to \{1, 2, \ldots, s\}$, where $g(A)$ indicates the length

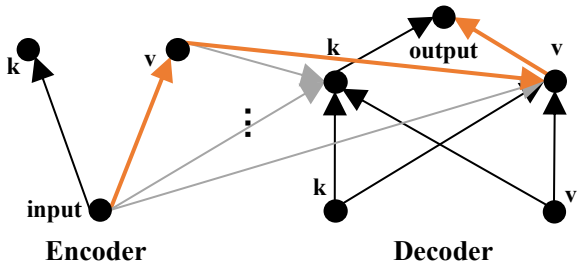

Figure 7. **Illustration of the maximum iteration count in graph G_M.** For a transformer composed of one encoder and one decoder, the abstracted G_M after LoRA fine-tuning on the k and v matrices allows IterIS to converge within two iterations.

	B-1	B-2	B-3	B-4	RougeL	CIDEr	ACC_{POS}	ACC_{NEG}
$LoRA_1$	0.570	0.369	0.241	0.159	0.426	0.808	0.848	0.018
$LoRA_2$	0.550	0.354	0.234	0.157	0.414	0.811	0.138	0.867

Table 4. **Performance of style-specific LoRAs for multi-style caption on vision-language model.**

of the longest path from O_{in} to node A. We will use mathematical induction to demonstrate that at the k-th iteration, all nodes in $g^{-1}(\{1, 2, \ldots, k+1\})$ and their corresponding updated matrices remain unchanged in subsequent iterations. This observation relies on the fact that if the unified adapter's input $\tilde{X}_i$ remains constant during iterations, the computed W_i will also remain constant.

1. Base Case ($k = 0$): For the initial iteration, it is clear that $g^{-1}(1)$ is non-empty and contains only nodes with O_{in} as their sole parent. Therefore, $g^{-1}(1)$ is entirely dependent on the input, indicating that these nodes and their corresponding matrices will remain unchanged in subsequent iterations.

2. Inductive Step (Assume true for $k = k_0$): Assume that at iteration $k = k_0$, all nodes in $g^{-1}(\{1, 2, \ldots, k_0 + 1\})$ and their associated matrices remain constant. Since each node in $g^{-1}(k_0 + 2)$ has its parent nodes contained within $g^{-1}(\{1, 2, \ldots, k_0+1\})$, it follows that the nodes in $g^{-1}(k_0+2)$ are reliant on the unchanged nodes in $g^{-1}(\{1, 2, \ldots, k_0 + 1\})$. Consequently, these nodes and their corresponding updated matrices will also remain constant in subsequent iterations.

By this reasoning, when $k = s - 1$, it holds that $g^{-1}(\{1, 2, \ldots, s\}) = v(G_M) \setminus \{O_{\text{in}}\}$ will remain unchanged in subsequent iterations. Thus, we prove and compute the upper bound on the maximum number of iterations required for the convergence of the IterIS algorithm.

For a transformer model consisting of L layers of encoders and L layers of decoders, if we apply LoRA fine-tuning to the k and v matrices within the attention modules, our algorithm will converge in no more than $3L - 1$ iterations. As illustrated in Figure 7, the red-marked path has a length of 3, indicating that the algorithm converges after two iterations.

COLA	RTE	SST2	MNLI	QQP	QNLI	MRPC
0.532	0.812	0.946	0.855	0.858	0.920	0.831

Table 5. **Performance of task-specific LoRAs for multi-task integration on large language model.**

TEC	Emoint	ISEAR	EC
0.669	0.828	0.767	0.947

Table 6. **Performance of task-specific LoRAs for in-domain task integration on large language model.**

B. Detailed Experimental Setting

B.1. Datasets

DreamBooth [19]. The DreamBooth dataset consists of 30 subjects across 15 different classes. Among these, 9 are live subjects (dogs and cats), while the remaining 21 are objects. Each subject is represented by a variable number of images (ranging from 4 to 6), typically captured under diverse conditions, in various environments, and from different angles.

Customconcept101 [11]. The customconcept101 dataset, introduced by custom diffusion [11], comprises 101 concepts, each represented by 3 to 15 images, designed to evaluate model customization methods.

SentiCap [13]. The SentiCap dataset comprises thousands of images with sentiment-labeled captions. The POS subset includes 2,873 positive sentences paired with 998 images for training and 2,019 sentences paired with 673 images for testing. The NEG subset contains 2,468 negative sentences with 997 images for training and 1,509 sentences with 503 images for testing.

GLUE [23]. The GLUE dataset serves as a benchmark for evaluating natural language understanding models. It encompasses multiple tasks, including sentiment analysis, textual entailment, and sentence similarity. For our evaluation, we utilize the SST-2 [21], MRPC [3], MNLI [25], QNLI [17], RTE [7], COLA [24], and QQP datasets. Evaluations are conducted on the official development sets, as the test labels remain hidden.

Emotion. For emotion classification, we use the preprocessed datasets provided by Oberlander & Klinger [1], specifically TEC [14], ISEAR [20], Emoint [15], and Emotion-Cause (EC) [6], for domain-specific training. All four datasets include the emotion classes of anger, fear, joy, and sadness in their label space, encompassing various scenarios and sources. In our experiments, each dataset is randomly divided into training and testing sets, with five parts for training and one part for testing.

FlickrStyle10K [5]. The FlickrStyle10K dataset is derived from the Flickr30K [16] image caption dataset. It originally contained 10,000 image-caption pairs with stylized captions in humorous and romantic styles. However, only 7,000 pairs from the official training set are currently publicly available.

B.2. LoRA Training

LoRAs for Multi-concept Customization. We applied the widely-used DreamBooth-LoRA [19] tuning to generate multiple LoRAs for stable diffusion v1.5 [18], with each LoRA dedicated to a single new concept. The LoRA rank was set to 32, and LoRA

Iteration	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
ACC$_{mean}$	0.674	0.919	0.681	0.862	0.729	0.806	0.789	0.788	0.786	0.784	0.784	0.784	0.784	0.784	0.784
CIDEr$_{mean}$	0.785	0.381	0.793	0.651	0.798	0.794	0.791	0.770	0.789	0.790	0.791	0.790	0.790	0.790	0.790
Score$_{aver}$	0.730	0.650	0.737	0.756	0.764	0.800	0.790	0.779	0.788	0.787	0.788	0.787	0.787	0.787	0.787

Table 7. **Performance across different maximum iterations.**

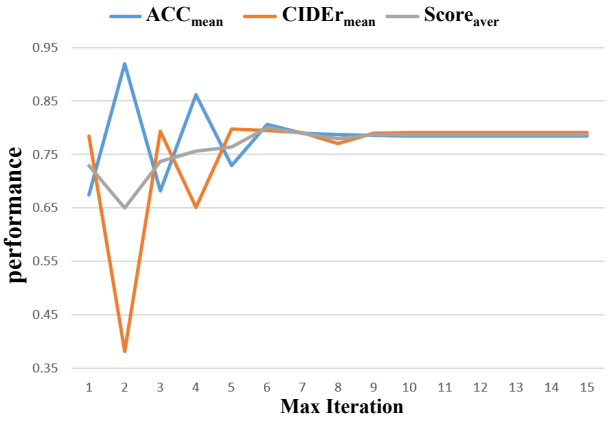

Figure 8. **Performance across different maximum iterations**

alpha was set to 1. During the textual inversion phase, we used a learning rate of 5×10^{-4}, with each new concept represented by a single learnable token, all initialized uniformly. In the subsequent fine-tuning phase, the UNet and text encoder were trained with learning rates of 10^{-4} and 10^{-5}, respectively, while the textual inversion learning rate was maintained at 10^{-4}. Both phases involved training for 800 steps, with a batch size of 1 and gradient accumulation steps set to 4.

LoRAs for Multi-style Caption. We applied LoRA tuning to fine-tune the BLIP-image-caption-base [12]. LoRAs were integrated into all self-attention modules within the text encoder, specifically targeting the q, v, and k matrices [22]. The LoRA rank was set to 32, with α set to 32, and a dropout rate of 0.05. The training was conducted with a batch size of 8 for a maximum of 15 epochs, using an initial learning rate of 3e-5. The AdamW [10] optimizer was employed for optimization. Table 4 summarizes the key metrics of multiple LoRAs trained in this part.

LoRAs for Multiple NLP tasks Integration. 1) In-domain task integration: We fine-tuned FLAN-t5-large [2] using LoRA tuning, targeting the q and v matrices of all attention layers [22]. The learning rate was set to 5e-5, with a training duration of 10 epochs and a batch size of 12. A warm-up phase [8] of 100 steps was applied. Optimization was performed using the AdamW optimizer [10], with a weight decay of 0.001. **2)** Multi-task integration: We fine-tuned FLAN-T5-base [2] using LoRA on the q and v matrices across all attention layers [22]. The learning rate was set to either 5×10^{-5} or 1×10^{-5}, with training durations of 8 or 10 epochs and batch sizes of 12 or 16. A 100-step warm-up phase [8] was applied. The AdamW optimizer [10] was used with a weight decay of 0.001. For further detailed training configurations for each subset, please refer to the forthcoming release of our code. Table 5 and Table 6 summarize the key metrics of the LoRAs trained in this part.

	B-1	B-2	B-3	B-4	Rough-L	CIDEr
POS (Ours w/o Reg)	0.480	0.291	0.180	0.112	0.368	0.534
POS (Ours w/ Reg)	**0.561**	**0.359**	**0.232**	**0.151**	**0.420**	**0.794**
NEG (Ours w/o Reg)	0.356	0.206	0.127	0.079	0.334	0.476
NEG (Ours w/ Reg)	**0.540**	**0.348**	**0.230**	**0.153**	**0.406**	**0.794**

Table 8. **Performance comparison of our algorithm with and without the regularization term for vision-language model**. The **bold** numbers highlight the best performance.

B.3. Baseline Training Details

Textual Inversion [4]. We employed uniformly initialized learnable tokens with a batch size of 1 and set the gradient accumulation steps to 4. The learning rate was configured to 5×10^{-4}, and the training was carried out over a total of 800 steps. Each new concept was represented as "[adj_n] n," where "[adj_n]" denoted a learnable token. For instance, when introducing the new concept "cat," we represented the new concept as "[adj_cat] cat".

Custom Diffusion [11]. We utilized the official implementation of custom diffusion. To ensure a fair comparison, we adopted its optimization method as the baseline. The number of DDPM steps was set to 200, and the unconditional guidance scale was set to 6.

Linear Merging [26]. We performed linear merging using LoRAs with identical weights (e.g., average merging).

RegMean [9]. We utilized the official implementation of RegMean, with modifications to the interface. In our experiments, we reproduced RegMean by adopting its official regularization settings, with the regularization coefficient α set to 0.1. Following the recommendations in the original paper, we set the number of inference samples to 100–200, with a batch size of 16. In cases where the dataset contained fewer than the recommended number, we used all available data. All other hyperparameters and LoRAs were kept consistent with those in our method.

C. Ablation Study of IterIS

Influence on Maximum Iteration. As shown in Figure 8 and Table 7, we evaluated the impact of varying maximum iterations in IterIS on vision-language model. Mean ACC, CIDEr, and score$_{aver}$ (defined as (ACC$_{mean}$ + CIDEr$_{mean}$) / 2) were computed on the validation sets of the POS and NEG datasets. When limited to 4 or fewer iterations, IterIS exhibits notable instability, starting with a low score$_{aver}$ that progressively improves. Peak performance is observed at 5 iterations, beyond which a slight decline suggests potential overfitting. Performance stabilizes after 11 iterations, aligning with our theoretical expectations.

Influence on Regularization Term. Using 50 samples for inference, we evaluated the effect of the introduced regularization term

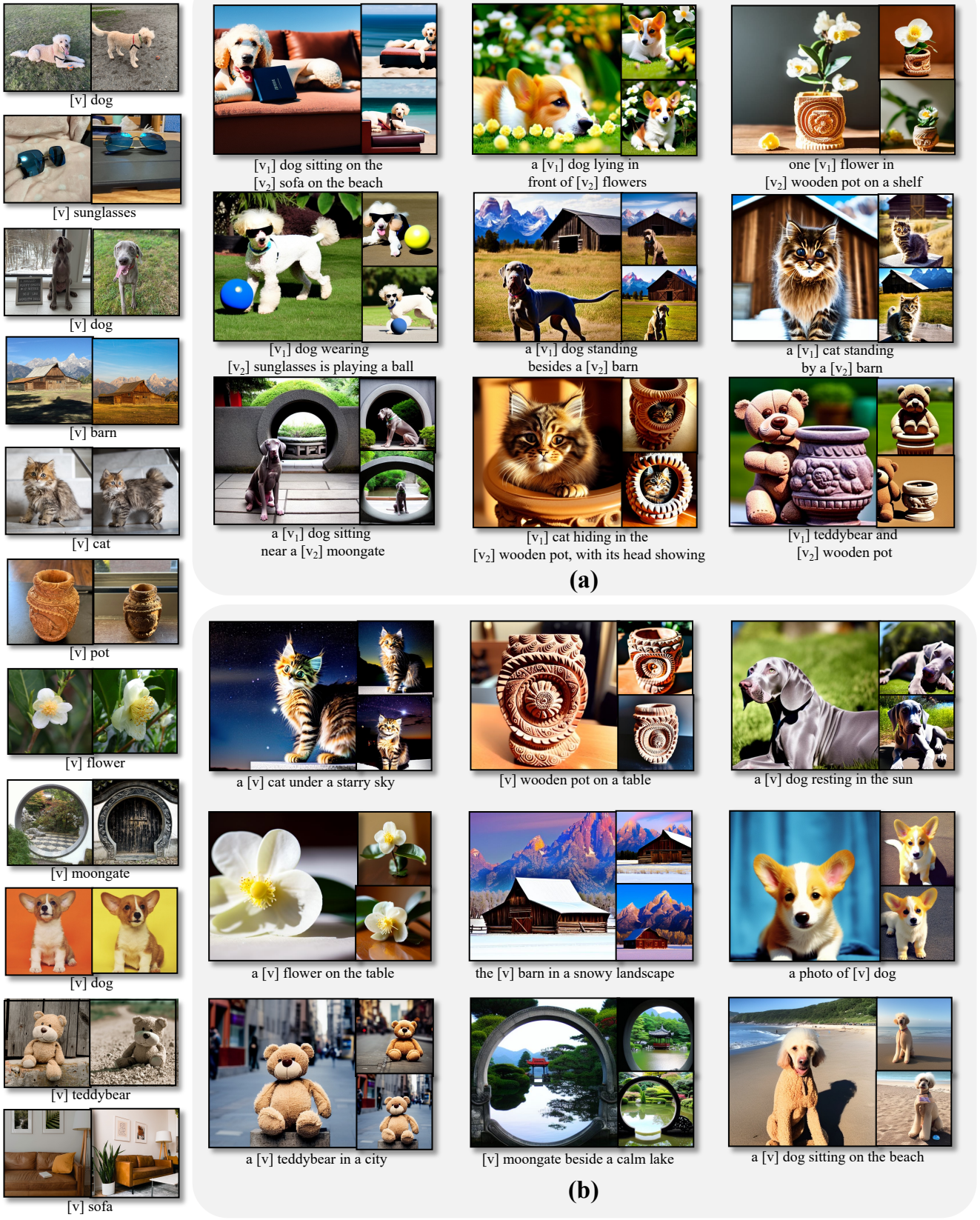

Figure 9. **Additional qualitative results for multi-concept customization.** Target images represent individual concepts used in the compositions. (a) Examples of pairwise compositions generated by our method. (b) Examples of individual concepts generated by IterIS.

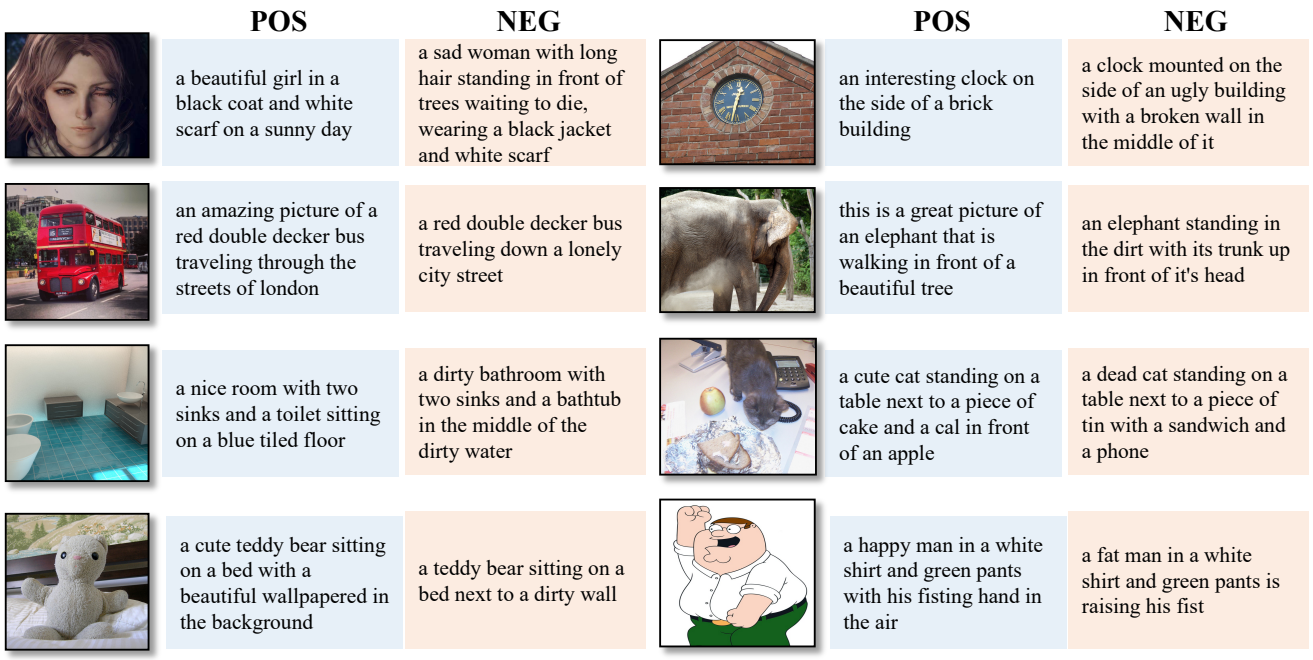

Figure 10. **More examples of style caption generated by our algorithm.**

Composition	**(MRPC, SST2)**	**(COLA, MNLI)**	**(RTE, MRPC)**
Ours w/o Reg	(0.272, 0.177)	(0.0, 0.0)	(0.671, 0.667)
Ours w/ Reg	**(0.824, 0.951)**	**(0.299, 0.780)**	**(0.805, 0.814)**

Table 9. **Performance comparison of our algorithm with and without the regularization term for LLM.** The first dataset pair consists of MRPC and SST2, and so on. The **bold** numbers indicate the best performance within each dataset pair.

Composition	**(COLA, MNLI)**	**(MNLI, SST2)**	**(COLA, SST2)**
Ours w/o Weights	(0.271, 0.728)	(0.761, 0.945)	(0.356, 0.943)
Ours w/ Weights	**(0.299, 0.780)**	**(0.764, 0.945)**	**(0.360, 0.943)**

Table 10. **Performance comparison of our algorithm with and without adaptive weights for LLM.** The first dataset pair consists of COLA and MNLI, and so on. The **bold** numbers indicate the best performance within each dataset pair.

on the performance of IterIS for both the vision-language model and the large language model. As demonstrated in Table 8 and Table 9, incorporating the regularization term results in substantial performance improvements, highlighting its pivotal role in enhancing the effectiveness of IterIS.

Influence on Adaptive Weights. We assessed the impact of adaptive weights on algorithm performance, as summarized in Table 10. The results demonstrate that adaptive weights can enhance algorithm performance to a certain extent. Moreover, we observe that when there are greater discrepancies between different tasks, the application of our proposed adaptive weights leads to a more noticeable improvement (*e.g.*, COLA and MNLI).

Style	**Method**	**B-1**	**B-2**	**B-3**	**B-4**	**Cider**	**ACC**
POS	RegMean	0.576	0.368	0.239	0.156	**0.801**	0.624
	Linear	0.561	0.357	0.230	0.151	0.771	0.522
	IterIS	0.561	0.359	0.232	0.151	0.794	**0.830**
NEG	RegMean	0.540	0.341	0.223	0.144	0.779	0.692
	Linear	0.509	0.319	0.206	0.133	0.733	0.557
	IterIS	0.540	0.348	0.230	0.153	**0.794**	**0.781**
HUM	RegMean	0.303	0.174	0.103	0.061	0.528	-
	Linear	0.281	0.161	0.095	0.055	0.504	-
	IterIS	0.300	0.174	0.103	0.060	**0.530**	-
ROM	RegMean	0.307	0.172	0.101	0.061	0.523	-
	Linear	0.266	0.151	0.054	0.047	0.487	-
	IterIS	0.308	0.172	0.101	0.062	**0.524**	-

Table 11. **Detailed performance comparison for multi-style caption generation** includes two combinations: **(1)** "positive" (POS) + "negative" (NEG), and **(2)** "humor" (HUM) + "romance" (ROM). The **bold** values highlight the best performance.

D. More Results

More Results on Text-to-Image Diffusion. Figure 9(a) showcases additional combinations of pairwise concepts from our experiments, while Figure 9(b) illustrates the individual concepts generated by the composed LoRAs. Our method exhibits robust performance across a wide range of multi-concept combinations.

More Results on V&L Model. As shown in Figure 6 of Section 4, we provide additional examples of positive and negative captions generated by our algorithm. A detailed version of Table 2 of Section 4 from the main text is presented in Table 11. Additionally, we conducted experiments on FlickrStyle10K [5], exploring combinations of two other distinct styles. The results are also included

Composition	Score$_1$			Score$_2$			Score$_3$			Score$_4$		
	IterIS	RegMean	Linear	IterIS	RegMean	Linear	IterIS	RegMean	Linear	IterIS	RegMean	Linear
Emoint/EC	**0.810**	0.808	0.784	**0.938**	0.922	0.912	-	-	-	-	-	-
Emoint/TEC	**0.807**	0.758	0.731	**0.551**	0.544	0.474	-	-	-	-	-	-
Emoint/ISEAR	**0.811**	0.786	0.733	**0.665**	0.650	0.603	-	-	-	-	-	-
EC/TEC	**0.945**	0.926	0.881	0.603	**0.630**	0.592	-	-	-	-	-	-
EC/ISEAR	**0.953**	0.938	0.900	0.709	**0.719**	0.679	-	-	-	-	-	-
TEC/ISEAR	0.574	**0.592**	0.526	**0.710**	0.661	0.631	-	-	-	-	-	-
Emoint/EC/TEC	**0.782**	0.753	0.727	**0.942**	0.923	0.877	**0.540**	0.510	0.460	-	-	-
Emoint/EC/ISEAR	**0.795**	0.774	0.720	**0.935**	0.926	0.887	**0.670**	0.641	0.593	-	-	-
Emoint/TEC/ISEAR	**0.786**	0.721	0.683	**0.500**	0.499	0.455	**0.646**	0.607	0.593	-	-	-
EC/TEC/ISEAR	**0.950**	0.910	0.857	0.552	**0.555**	0.486	**0.683**	0.649	0.614	-	-	-
Emoint/EC/TEC/ISEAR	**0.776**	0.714	0.688	**0.935**	0.901	0.873	**0.508**	0.481	0.445	**0.657**	0.607	0.590

Table 12. **Performance of all possible compositions in in-domain task integration.** Score$_i$ denotes the performance metric for the i-th task within the composition. The **bold** values highlight the best performance.

Composition	Score$_1$			Score$_2$		
	IterIS	RegMean	Linear	IterIS	RegMean	Linear
MNLI/RTE	**0.831**	0.803	0.751	**0.794**	0.787	0.776
MNLI/COLA	**0.780**	0.755	0.765	0.299	0.279	**0.383**
MNLI/SST2	**0.764**	0.747	0.645	**0.945**	**0.945**	0.942
MNLI/QQP	**0.821**	0.813	0.687	**0.855**	0.853	0.842
MNLI/QNLI	**0.821**	0.806	0.797	**0.916**	0.914	0.905
MNLI/MRPC	**0.825**	0.816	0.744	**0.821**	0.792	0.772
RTE/COLA	**0.787**	0.783	0.733	0.311	0.307	**0.388**
RTE/SST2	**0.819**	0.809	0.773	0.946	**0.948**	0.943
RTE/QQP	0.816	0.812	**0.819**	0.856	**0.856**	0.853
RTE/QNLI	0.805	0.805	**0.812**	0.919	**0.920**	0.913
RTE/MRPC	0.805	**0.812**	0.805	**0.814**	0.801	0.757
COLA/SST2	**0.360**	0.291	0.233	**0.943**	0.938	0.891
COLA/QQP	0.386	0.332	**0.397**	**0.839**	0.831	0.734
COLA/QNLI	0.297	0.305	**0.346**	**0.906**	0.904	0.818
COLA/MRPC	0.341	0.307	**0.383**	0.811	**0.816**	0.765
SST2/QQP	**0.947**	**0.947**	0.936	**0.855**	0.854	0.839
SST2/QNLI	**0.943**	**0.943**	**0.943**	**0.920**	0.916	0.882
SST2/MRPC	**0.951**	0.947	0.939	**0.824**	0.809	0.794
QQP/QNLI	**0.856**	0.856	0.846	**0.920**	0.920	0.915
QQP/MRPC	**0.855**	0.854	0.846	**0.816**	0.814	0.801
QNLI/MRPC	**0.922**	0.920	0.907	**0.821**	**0.821**	0.797

Table 13. **Performance of all pairwise compositions in multi-task integration.** Score$_i$ denotes the performance metric for the i-th task within the composition. The **bold** values highlight the best performance.

in Table 11. Due to the subjective nature of evaluating humor and romance, as well as the lack of the open test set, we didn't quantify humor(romance)-specific metrics. As a result, aside from the CIDEr score, the advantages of our algorithm for this style combination are less evident.

Composition	(MRPC, SST2)	(SST2, QQP)	(QQP, MRPC)
Task arithmetic	(0.806, 0.937)	(0.937, 0.841)	(0.845, 0.811)
Ties merging	(0.765, 0.943)	(0.944, 0.823)	(0.846, 0.806)
Hyper-linear	(0.809, 0.933)	(0.931, 0.849)	(0.846, 0.801)
IterIS (Ours)	**(0.824, 0.951)**	**(0.947, 0.855)**	**(0.855, 0.816)**

Table 14. **Performance for multi-task integration for additional three baselines.**

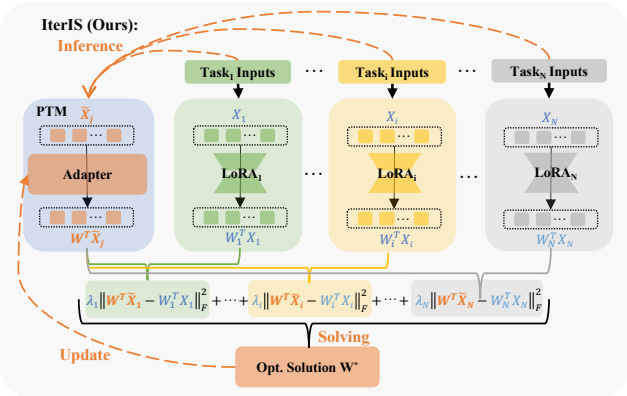

Figure 11. **The workflow diagram of our algorithm.**

More Results on LLM. A detailed version of Table 3 in Section, as discussed in the main text, is provided for reference in this part. Table 12 summarizes the experimental results of 11 combinations for in-domain task integration, while Table 13 presents the results of 21 combinations for multi-task integration. Our proposed method outperforms both RegMean and linear merging in the majority of cases, highlighting its robustness and effectiveness. Additionally, Table 14 shows the superiority of our method compared to three extra baselines in the multi-task integration subset.

E. Others

Illustration of IterIS. To facilitate a better understanding of IterIS, we present Figure 11, which effectively illustrates the layer-

wise characteristics and inference-solving process of our method.

Efficiency Analysis. IterIS ensures computation and memory efficiency through sample efficiency and limited iterations. In the V&L experiments, we utilized 50 prompt-image pairs as inference samples and performed 6 iterations. For each layer, the total computational cost for computing a single inner product matrix is 50 (sample) $\times$ 6 (iteration) $=$ 300 one-dimensional tensor inner products. However, that cost for RegMean is $1600 \times 1 = 1600$, which is much higher than IterIS. Moreover, sample efficiency of IterIS allows us to perform all computations without batching, ensuring memory efficiency. Notably, all experiments for IterIS were conducted using a single RTX 3090 GPU.

Detailed Analysis for the Regularization Term and Adaptive Weights. 1) The regularization term prevents irreversible situations in solutions, which often occur in few-sample cases. In such cases, at least one solution to the optimization objective in Eq.(5) renders it zero. This means the output features in each layer of the merged model are equal to those in the corresponding layer of each individual model. Due to the limited number of inference samples, the risk of overfitting to these samples is high. Notably, we can infer that the optimal solution to Eq.(1) is close to the linear solution based on Eq.(2). Applying identity matrices as regularization terms aligns the solution more closely with the linear solution, enhancing robustness. However, diagonal matrices in RegMean cannot achieve the same effect. **2)** From a dimensional analysis perspective, the scale of the adaptive weights corresponds to that of $\|\boldsymbol{X}_i\|_F^{-2}$. This neutralizes the scale of $\boldsymbol{X}_i$ in the optimization objective ($\|\boldsymbol{W}_i^T\boldsymbol{X}_i - \boldsymbol{W}^T\tilde{\boldsymbol{X}}_i\|_F^2$), thereby ensuring that the optimization objective remains unbiased to the scale of $\boldsymbol{X}_i$.

Limitations. Similar to other LoRA merging methods, performance degradation on individual tasks is inevitable when merging tasks of different types. This degradation becomes more pronounced as the diversity of task types increases or when parameter conflicts arise. Furthermore, since our method does not include domain-specific enhancements for text-to-image diffusion, it shares a limitation with custom diffusion—namely, the confusion of object concepts.

References

[1] Laura Ana Maria Bostan and Roman Klinger. An analysis of annotated corpora for emotion classification in text. 2018. 2

[2] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. In *arXiv*, 2022. 3

[3] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *IWP2005*, 2005. 2

[4] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 3

[5] Chuang Gan, Zhe Gan, Xiaodong He, Jianfeng Gao, and Li Deng. Stylenet: Generating attractive visual captions with styles. In *CVPR*, pages 3137–3146, 2017. 2, 5

[6] Diman Ghazi, Diana Inkpen, and Stan Szpakowicz. Detecting emotion stimuli in emotion-bearing sentences. In *CICLing*, pages 152–165. Springer, 2015. 2

[7] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. The third pascal recognizing textual entailment challenge. In *ACL-PASCAL*, pages 1–9, 2007. 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3

[9] Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022. 3

[10] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[11] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *CVPR*, pages 1931–1941, 2023. 2, 3

[12] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, pages 12888–12900. PMLR, 2022. 3

[13] Alexander Mathews, Lexing Xie, and Xuming He. Senticap: Generating image descriptions with sentiments. In *AAAI*, 2016. 2

[14] Saif Mohammad. # emotional tweets. In *SEM*, pages 246–255, 2012. 2

[15] Saif M Mohammad and Felipe Bravo-Marquez. Wassa-2017 shared task on emotion intensity. *arXiv preprint arXiv:1708.03700*, 2017. 2

[16] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, pages 2641–2649, 2015. 2

[17] P Rajpurkar. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016. 2

[18] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 2

[19] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, pages 22500–22510, 2023. 2

[20] Klaus R Scherer and Harald G Wallbott. Evidence for universality and cultural variation of differential emotion response patterning. *Journal of personality and social psychology*, 66 (2):310, 1994. 2

[21] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher

Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013. 2

[22] A Vaswani. Attention is all you need. *NeurIPS*, 2017. 3

[23] Alex Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. 2

[24] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Cola: The corpus of linguistic acceptability (with added annotations). 2019. 2

[25] Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017. 2

[26] Jinghan Zhang, Junteng Liu, Junxian He, et al. Composing parameter-efficient modules with arithmetic operation. *NeurIPS*, 36:12589–12610, 2023. 3