CrossGET: Cross-Guided Ensemble of Tokens for Accelerating Vision-Language Transformers

Dachuan Shi [1 2] **Chaofan Tao** [3] **Anyi Rao** [4] **Zhendong Yang** [1] **Chun Yuan** [1 †] **Jiaqi Wang** [2 †]

Abstract

Recent vision-language models have achieved tremendous advances. However, their computational costs are also escalating dramatically, making model acceleration exceedingly critical. To pursue more efficient vision-language Transformers, this paper introduces **Cross-G**uided **E**nsemble of **T**okens (***CrossGET***), a general acceleration framework for vision-language Transformers. This framework adaptively combines tokens in real-time during inference, significantly reducing computational costs while maintaining high performance. *CrossGET* features two primary innovations: 1) *Cross-Guided Matching and Ensemble*. *CrossGET* leverages cross-modal guided token matching and ensemble to effectively utilize cross-modal information, achieving wider applicability across both modality-independent models, *e.g.*, CLIP, and modality-dependent ones, *e.g.*, BLIP2. 2) *Complete-Graph Soft Matching*. *CrossGET* introduces an algorithm for the token-matching mechanism, ensuring reliable matching results while facilitating parallelizability and high efficiency. Extensive experiments have been conducted on various vision-language tasks, such as image-text retrieval, visual reasoning, image captioning, and visual question answering. The performance on both classic multimodal architectures and emerging multimodal LLMs demonstrates the framework's effectiveness and versatility. The code is available at https://github.com/sdc17/CrossGET.

[1]Tsinghua University [2]Shanghai AI Laboratory [3]The University of Hong Kong [4]Stanford University. Work was done when Dachuan Shi was an intern at Shanghai AI Laboratory. [†]Correspondence to: Chun Yuan <yuanc@sz.tsinghua.edu.cn>, Jiaqi Wang <wjqdev@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

The AI community is currently witnessing the bloom of vision-language models (Kiros et al., 2014; Karpathy et al., 2014; Antol et al., 2015; Vinyals et al., 2015; Yang et al., 2016; Huang et al., 2017; Radford et al., 2021; Wang et al., 2022a; Li et al., 2022; 2023b), with Transformer-based models such as CLIP (Radford et al., 2021), BLIP/BLIP2 (Li et al., 2022; 2023c), and GPT-4V (OpenAI, 2023) emerging as prominent in recent research. These models are capable of tackling a broad range of vision-language tasks, such as Image-Text Retrieval (Jia et al., 2015), Vision Reasoning (Suhr et al., 2018), Image Captioning (Lin et al., 2014), and Visual Question Answering (Antol et al., 2015). Nevertheless, the notable improvement is at the expense of significantly increased computational cost, making it less accessible for consumers with limited resources.

The computational cost of Transformers increases monotonically with the input tokens. Token reduction, which reduces the number of tokens processed during forward, is an effective strategy to mitigate high computational costs for both vision Transformers (Rao et al., 2021; Liang et al., 2022b; Bolya et al., 2023) and language Transformers (Goyal et al., 2020; Wang et al., 2021; Kim et al., 2022). Although studied on the acceleration of unimodal models, a non-negligible research gap persists in multimodal contexts.

In vision-language transformers, a straightforward idea involves leveraging cross-modal information to guide token reduction. This concept can be intuitively applicable in modality-independent frameworks, such as CLIP (Radford et al., 2021). However, recent popular vision-language Transformers like BLIP/BLIP2 (Li et al., 2022; 2023c) and LLaVA (Liu et al., 2023b;a) are modality-dependent, with the vision encoder processing first. While vision features can guide token reduction in the following networks, incorporating language priors to guide token reduction in the vision encoder poses a challenge. The issue of facilitating bidirectional guidance, which would allow the preceding modality to benefit from information in the succeeding modality for token reduction, remains an open question.

This paper introduces *CrossGET*, a general acceleration framework designed to efficiently reduce the number of to-

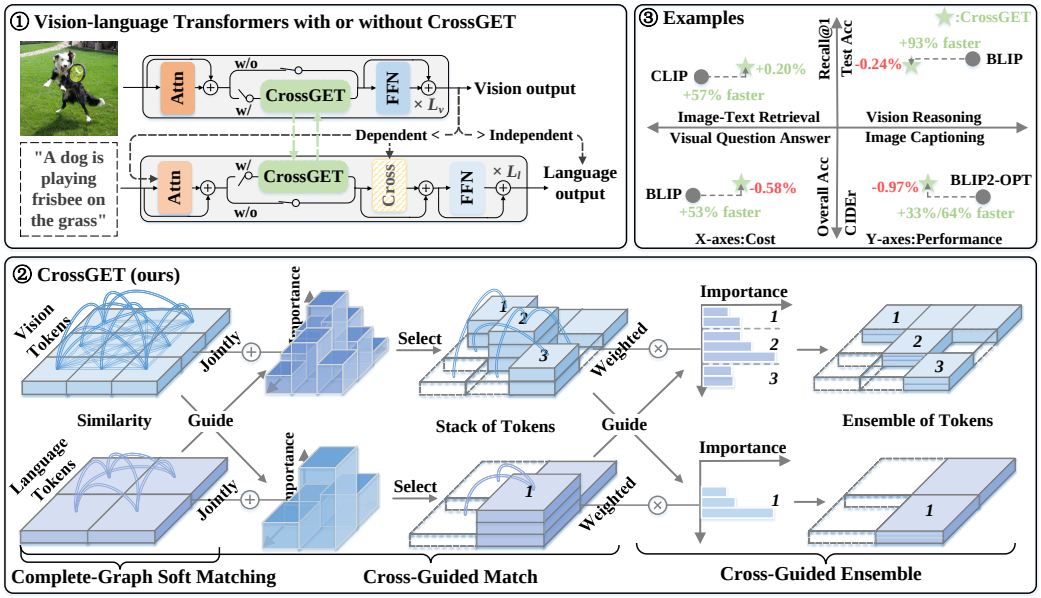

Figure 1: **Overview of *CrossGET*.** ① *CrossGET* is a general multimodal token reduction framework that applies to both modality-independent and modality-dependent models. ② *CrossGET* jointly considers the token similarity derived from intra-modal complete-graph soft matching and the token importance indicated by cross-modal guidance to determine which tokens should be combined. The cross-modal importance is subsequently utilized to weight tokens within each stack and output their ensembles. ③ Compared with the original models, *CrossGET* achieves considerable computation saving and acceleration with negligible performance degradation.

kens for both *modality-independent* and *modality-dependent* vision-language Transformers with bidirectional guidance. *CrossGET* features two primary innovations: *cross-guided matching and ensemble* and *complete-graph soft matching*.

Firstly, *CrossGET* utilizes *cross-guided matching and ensemble* to identify and ensemble redundant tokens, which applies to both modality-independent and modality-dependent models. *CrossGET* incorporates cross tokens into both vision and language branches to facilitate learning of cross-modal importance and to guide the selection of redundant tokens.[1] Secondly, for the underlying mechanism of token matching, *CrossGET* formulates it as a discrete optimization problem and proposes an approximate algorithm *complete-graph soft matching* to secure reliable matching results while maintaining parallelizability for high efficiency. The contributions of this paper are summarized as follows:

- It is one of the pioneering efforts in token ensemble framework for vision-language Transformers, achieving general applicability across both modality-independent and modality-dependent models. The approach is also validated in zero-shot scenarios.
- It introduces *cross-guided matching and ensemble*, a novel approach for effectively leveraging cross-modal information. It also proposes a *complete-graph soft matching* algorithm for reliable token-matching results while maintaining parallelizability.
- Its versatility has been validated across a broad range of vision-language tasks, datasets, and model architectures. This is also the *first* application of token ensemble approaches to the modality-dependent pipeline of BLIP2 (Li et al., 2023c), which is a widely adopted paradigm among recent large vision-language Transformers, *e.g.*, LLaVA (Liu et al., 2023b), MiniGPT-4 (Zhu et al., 2023), and mPLUG-Owl (Ye et al., 2023).

[1] A naive solution is to calculate the similarity between vision and language tokens directly. However, in modality-dependent models, different modality branches are aligned crosswise or sequentially, rendering cross-modal similarity inaccessible to preceding branches. *CrossGET* enables cross tokens within each modality to act as proxies for other modalities, allowing the preceding modality to leverage information from the succeeding modality without being constrained by the order of calculations.

2. Related Work

Vision-Language Transformers According to the dependency on calculation order across different modalities, existing vision-language Transformers can be classified into two main categories: 1) Modality-independent models (Li et al., 2020; 2021; Radford et al., 2021; Kim et al., 2021; Singh et al., 2022). For example, CLIP (Radford et al., 2021) is a representative model. These models allow for both the visual and language branches to be calculated simultaneously. 2) Modality-dependent models (Li et al., 2021; Yu et al., 2022; Li et al., 2022; Alayrac et al., 2022), exemplified by BLIP-based models (Li et al., 2022) and BLIP2/LLaVA-based (Li et al., 2023c; Zhu et al., 2023; Dai et al., 2023; Liu et al., 2023b; Gao et al., 2023) multimodal LLMs. In

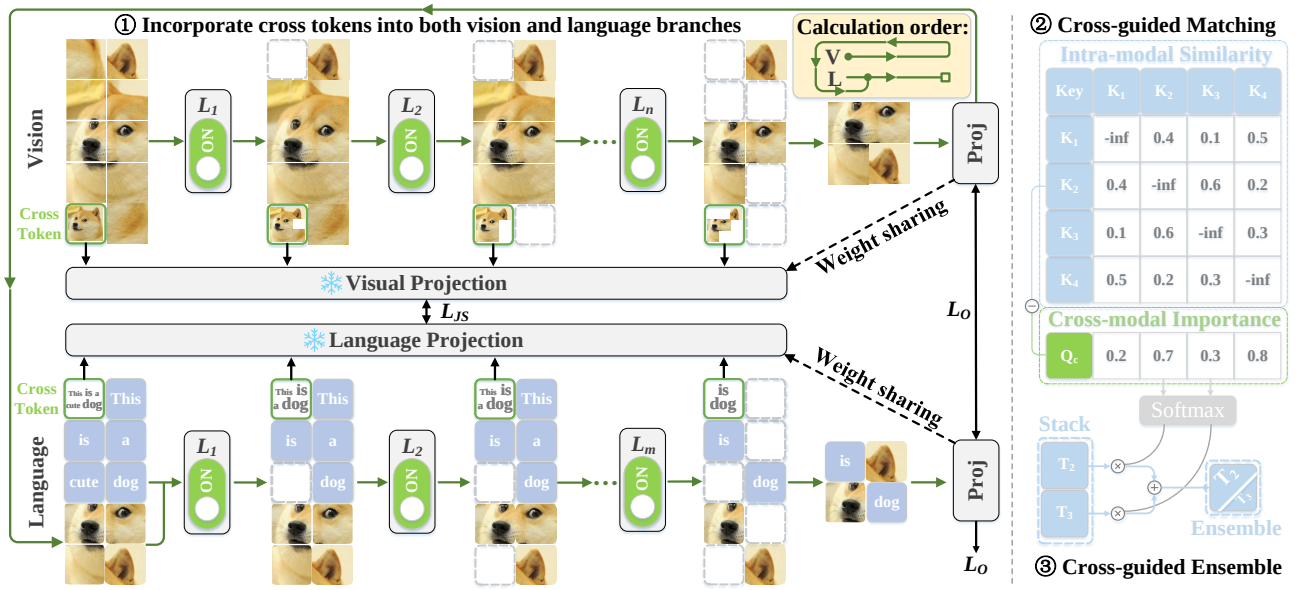

Figure 2: **Diagram of introducing and leveraging cross-model guidance for vision-language Transformers.** ① Cross tokens learn cross-modal information by closing the after-projection distance between cross tokens of different modalities. The switches indicate that it is free to choose whether to reduce tokens in different modalities and layers. ② Cross tokens provide cross-modal importance as a metric to guide token matching. ③ The metric also guides the weighted summation of the stacked tokens to produce token ensemble results.

these models, calculation must commence with the visual branch, as the language branch relies on outputs from the visual branch as part of its inputs. *CrossGET* applies to both modality-independent and modality-dependent scenarios.

Model Acceleration Techniques Numerous model acceleration techniques exist, for example, knowledge distillation (Hinton et al., 2015; Zhang et al., 2019; Jiao et al., 2019; Wang et al., 2020b; Touvron et al., 2021; Yang et al., 2022), model pruning (Han et al., 2015; He et al., 2017; Fan et al., 2019; Zhu et al., 2021; Chavan et al., 2022; Tao et al., 2023), and quantization (Xiao et al., 2022; Tao et al., 2022; Frantar & Alistarh, 2022; Yuan et al., 2023; Frantar et al., 2023) *CrossGET* is orthogonal to these techniques and does not seek to quantitatively surpass them. Instead, being orthogonal indicates that these techniques can be used together with *CrossGET* to further enhance their acceleration effect. Besides, *CrossGET* offers distinct advantages, including 1) Unlike knowledge distillation that necessitates tuning, *CrossGET* offers the flexibility to be utilized both with and without tuning. This is particularly beneficial when tuning large models is costly or when data are publicly unavailable. 2) The effectiveness of model pruning is heavily dependent on granularity. Unstructured and semi-structured pruning hardly delivers practical speedup without special hardware support, which is unnecessary for *CrossGET*. 3) Low-bit quantization may result in unstable training and necessitate custom CUDA kernel implementations, which are unnecessary for *CrossGET*. Furthermore, a recent advance, TRIPS (Jiang et al., 2022), employs text feature extracted from the Bert (Devlin et al., 2018) encoder to unidirectionally guide the token reduction in image encoder, which is limited to modality-independent models. In contrast, *CrossGET* is not only applicable to both modality-independent and modality-dependent scenarios, but also executes the modality-dependent token reduction in a more effective bidirectional manner.

3. Methodology

Figure 1 demonstrates that *CrossGET* accelerates vision-language Transformers by ensembling tokens. It is inserted into the middle of Self-Attention and FFN layers in both the vision and language branches. To effectively leverage cross-modal information, *CrossGET* proposes *cross-guided matching and ensemble* (Section 3.1). To achieve reliable token-matching results, *CrossGET* utilizes a parallelizable *complete-graph soft matching* algorithm (Section 3.2).

3.1. Cross-Guided Matching and Ensemble

Dependencies of Calculation Order For multimodal models, in addition to utilizing intra-modal similarity as guidance, token-matching results can further benefit from cross-modal guidance. However, effectively introducing cross-modal guidance is challenging, particularly when dependencies exist on the calculation order of modalities.

For example, if modality $\mathbb{A}$ requires guidance from modality $\mathbb{B}$, then $\mathbb{B}$ should perform inference, output features as cross-modal guidance, and send these to $\mathbb{A}$. However, if

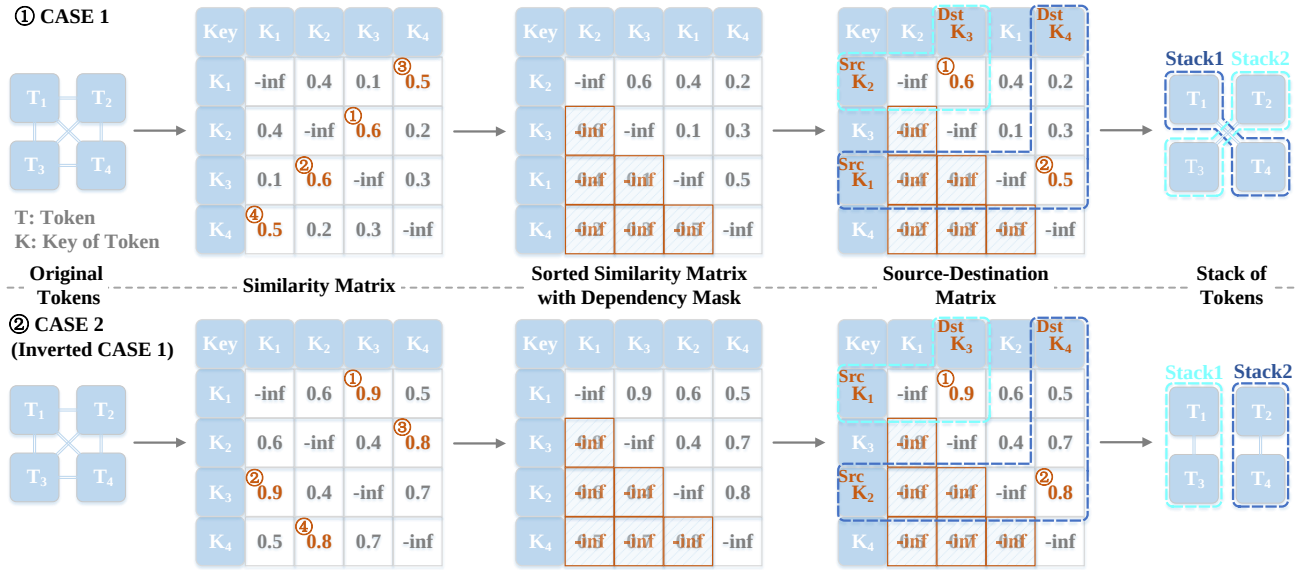

Figure 3: **Illustration of complete-graph soft matching on two examples.** Case2 is an inverted version of case1 in which the similarity between token pairs in case2 equals (1− similarity of corresponding pairs in case1).

a calculation dependency exists (*e.g.*, the output of $\mathbb{A}$ is a necessary input for $\mathbb{B}$), $\mathbb{B}$ cannot initiate inference before $\mathbb{A}$ completes its inference process. Therefore, $\mathbb{A}$ cannot leverage the cross-modal guidance provided by $\mathbb{B}$.

Breaking Dependencies To allow $\mathbb{A}$ to leverage information from the succeeding modality $\mathbb{B}$ without being constrained by order of calculations, *CrossGET* decouples the capability to guide preceding modalities from the inference process on succeeding modalities, *i.e.*, $\mathbb{B}$ can offer guidance to $\mathbb{A}$ before $\mathbb{B}$'s inference. As illustrated in Figure 2, this is achieved by injecting learnable cross tokens into each modality, driving them to learn cross-modal information from each other. When conducting inference within a modality, cross tokens act as proxies for other modalities, offering cross-modal guidance on behalf of other modalities.

Cross-Guided Matching Cross tokens provide cross-modal importance $\boldsymbol{I}$ as a metric to guide *complete-graph soft matching*. $\boldsymbol{I}$ is calculated as the cosine similarity between the query of the cross-token $\boldsymbol{T}_c \in \mathbb{R}^{1\times d}$ where d is the embedding size and the key of other tokens $\boldsymbol{T}_i \in \mathbb{R}^{1\times d}$, $i \neq c$:

$$\boldsymbol{I}_i = \frac{(\boldsymbol{T}_c\boldsymbol{W}^q)(\boldsymbol{T}_i\boldsymbol{W}^k)^\top}{\|\boldsymbol{T}_c\boldsymbol{W}^q\|_2\|\boldsymbol{T}_i\boldsymbol{W}^k\|_2}, \tag{1}$$

where $\boldsymbol{W}^q, \boldsymbol{W}^k \in \mathbb{R}^{d\times d}$ are weights of query and key layers, respectively. $\|\cdot\|_2$ denotes L2-norm.

Cross-Guided Ensemble *CrossGET* can be further enhanced by incorporating cross-modal guidance into the ensemble process. More specifically, employing the softmax value of cross-modal importance to produce a weighted summation of the stacked tokens as the ensemble results:

$$\boldsymbol{T}_i = \sum\nolimits_{\boldsymbol{T}_j \in \boldsymbol{S}_i} \mathrm{softmax}(\boldsymbol{I})_j \boldsymbol{T}_j, \tag{2}$$

where $\boldsymbol{S}_i$ represents the set of the stacked tokens, and $\boldsymbol{T}_i$ signifies the corresponding ensembled token.

Loss Function JS divergence $\mathcal{L}_{\mathcal{JS}}$ (*i.e.*, a symmetrized KL divergence $\mathcal{L}_{\mathcal{KL}}$) between after-projection cross tokens T^i_{cv} from vision and T^i_{cl} from language in layer i is [2]:

$$\mathcal{L}^i_{\mathcal{JS}} = \mathcal{L}_{\mathcal{JS}}[(\boldsymbol{T}^i_{cv}\tilde{\boldsymbol{W}}^v)||(\boldsymbol{T}^i_{cl}\tilde{\boldsymbol{W}}^l)] \tag{3}$$

$$= \frac{1}{2}\left[\mathcal{L}_{\mathcal{KL}}[(\boldsymbol{T}^i_{cv}\tilde{\boldsymbol{W}}^v)||\boldsymbol{T}^i_m] + \mathcal{L}_{\mathcal{KL}}[(\boldsymbol{T}^i_{cl}\tilde{\boldsymbol{W}}^l)||\boldsymbol{T}^i_m]\right], \tag{4}$$

$$\boldsymbol{T}^i_m = \frac{1}{2}(\boldsymbol{T}^i_{cv}\tilde{\boldsymbol{W}}^v + \boldsymbol{T}^i_{cl}\tilde{\boldsymbol{W}}^l), \tag{5}$$

where $\tilde{\boldsymbol{W}}^v$ and $\tilde{\boldsymbol{W}}^l$ represent the detached weights of the existing projection layers used for alignment in vision modality and language modality, respectively. Being detached implies that $\mathcal{L}^i_{\mathcal{JS}}$ produce gradients solely with respect to cross tokens $\boldsymbol{T}^i_{cv}$ and $\boldsymbol{T}^i_{cl}$, not affecting the projection layers. The weight of projection layers $\boldsymbol{W}^v$ and $\boldsymbol{W}^l$ are updated exclusively based on the gradients from the original loss. $\mathcal{L}^i_{\mathcal{JS}}$ is introduced to encourage cross tokens to learn cross-modal information from different modalities:

$$\mathcal{L} = \mathcal{L}_{\mathcal{O}} + \alpha \sum\nolimits_{i=0}^{L-1} \mathcal{L}^i_{\mathcal{JS}}, \tag{6}$$

where $\mathcal{L}_{\mathcal{O}}$ denotes the original loss for learning a multi-modal model, α is a hyperparameter to align the loss items

[2]For modalities with different number of layers, order-preserving mappings between layer indices can be employed.

closely in order of magnitude, and L is the number of model layers, which means cross tokens are inserted into each layer of the model and $\mathcal{L}_{\mathcal{JS}}$ should be calculated for each layer.

3.2. Complete-Graph Soft Matching

Problem Formulation for Token Matching Token matching is aimed at determining which tokens should be combined. Suppose there are $N \in \mathbb{N}^+$ tokens in total, and $r \in \mathbb{N}^+$ ($r < N$) tokens among them should be eliminated (*i.e.*, combined together with other tokens), then the token matching problem can be formulated as a discrete optimization problem that is to find a set of feasible token pairs:

$$\boldsymbol{P} = \{(\boldsymbol{T}_i, \boldsymbol{T}_j) \,|\, 0 \leq i, j \leq N, i \neq j\}, \quad |\boldsymbol{P}| = r, \tag{7}$$

where $\boldsymbol{T}_i$ denotes tokens i, and $|\cdot|$ denotes the size of the set, to maximize the objective function

$$S = \sum\nolimits_{(\boldsymbol{T}_i, \boldsymbol{T}_j) \in P} \mathcal{D}(\boldsymbol{T}_i, \boldsymbol{T}_j), \tag{8}$$

where $\mathcal{D}$ is a function (*e.g.*, cosine similarity) that calculates the similarity between the key of the token $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$. Appendix C.1 provides examples to elaborate.

Parallelizability While iterative clustering can be utilized for token matching, it cannot be parallelized and is time-consuming. To facilitate the parallelizability, an additional constraint $\boldsymbol{T}^S \cap \boldsymbol{T}^D = \phi$, should be met. *i.e.*, the source set $\boldsymbol{T}^S$ and destination set $\boldsymbol{T}^D$ should be disjointed, where

$$\boldsymbol{T}^S = \{\boldsymbol{T}_i \,|\, (\boldsymbol{T}_i, \boldsymbol{T}_j) \in P\}, \quad |\boldsymbol{T}^S| = r, \tag{9}$$

$$\boldsymbol{T}^D = \{\boldsymbol{T}_j \,|\, (\boldsymbol{T}_i, \boldsymbol{T}_j) \in P\}, \quad |\boldsymbol{T}^D| \leq r. \tag{10}$$

Algorithm Procedure *Complete-Graph Soft Matching* is designed as a non-iterative, approximate algorithm to ensure parallelizability and high efficiency. It enables each token to consider its similarity with all other tokens, as shown in Figure 3 (Appendix C.2 provides an implementation):

- **Step 1**: Calculate the cosine similarities $\frac{\boldsymbol{K}\boldsymbol{K}^\top}{\|\boldsymbol{K}\|_2^2}$ between the keys $\boldsymbol{K}$ of every two tokens to generate the similarity matrix $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ (Diagonal self-similarities are ignored).
- **Step 2**: Sort the rows and columns of the similarity matrix in descending order based on their maximum similarity $\max\limits_{1 \leq j \leq N} \boldsymbol{D}_{ij}$ and $\max\limits_{1 \leq i \leq N} \boldsymbol{D}_{ij}$ to other tokens.
- **Step 3**: Upon the sorted similarity matrix $\boldsymbol{D}^\star$, a lower triangle dependency mask $\boldsymbol{M}_{ij} = \begin{cases} -\infty & \text{for } i \geq j \\ 0 & \text{for } i < j \end{cases}$ is applied to disjoint the sets $\boldsymbol{T}^S$ and $\boldsymbol{T}^D$. It explicitly prioritizes the matching among tokens based on similarity values, ensuring source tokens with higher priority do not become targets for those with lower priority.
- **Step 4**: Select r rows with the highest similarity $\max\limits_{1 \leq j \leq N} \boldsymbol{D}^\star_{ij}$ to other tokens as the source set $\boldsymbol{T}^S$. For every token in $\boldsymbol{T}^S$, select tokens from $\boldsymbol{T} \backslash \boldsymbol{T}^S$ that exhibit the highest similarity as the destination set $\boldsymbol{T}^D$.
- **Step 5**: The matching among tokens leads to multiple connected components (*i.e.*, stacks), and tokens in each stack are ensembled by averaging.

This procedure is non-iterative and parallelizable. As depicted in Figure 3, *complete-graph soft matching* achieves optimal solutions in both case1 and case2.

Incorporation with Cross-Guided Matching and Ensemble Modifications are as follows to leverage cross-modal guidance (Appendix C.3 provides an implementation):

• **Step 4**: Select r rows via metric $\max\limits_{1 \leq j \leq N} \boldsymbol{D}^\star_{ij} - \boldsymbol{I}$ (*i.e.*, highest similarity to other tokens - cross-modal importance) instead of $\max\limits_{1 \leq j \leq N} \boldsymbol{D}^\star_{ij}$.

• **Step 5**: Instead of averaging, ensembling tokens via weighted summation based on $\mathrm{softmax}(\boldsymbol{I})$.

Appendix C.4 and C.5 provide additional discussions on the sub-optimal cases of this method and analyses regarding the expectation of optimal matching probability, respectively.

4. Experiments

We report the performance on modality-independent model CLIP (Radford et al., 2021) as well as modality-dependent models BLIP/BLIP2 (Li et al., 2022; 2023c), and mainstream tasks such as Image-Text Retrieval, Visual Reasoning, Image Captioning, and Visual Question Answering.

4.1. Experiments with CLIP on Image-Text Retrieval

We conduct experiments on the CLIP model, and Flickr30K datasets (Young et al., 2014) with Karpathy split (Karpathy & Fei-Fei, 2015) of Image-Text Retrieval and Text-Image Retrieval task. The number of tokens is reduced to half with the same reduction number for each layer. For example, suppose one of the modalities of a 12-layer CLIP has 100 tokens as input, then $\lfloor \frac{100}{12} \rfloor = 8$ tokens will be eliminated from each layer so that the number of tokens left in the last layer is $100 - 12 \times 8 = 4$, and the total number of tokens across all layers is roughly reduced to half. If not specified, the number of tokens to be reduced in other experiments is also determined by this strategy.

Table 1: **Accelerate CLIP on the Flickr30K dataset of the Image-Text Retrieval task.** R: Recall. R@1, R@5 and R@10 are the higher the better. Experimental results are reported *after training for all approaches*. CrossGET$^{\blacktriangle}$ only uses *complete-graph soft matching* (CGSM) (Section 3.2), CrossGET$^{\blacklozenge}$ adds *cross-guided matching* (CGM) (Section 3.1) on $^{\blacktriangle}$, and CrossGET* further adds *cross-guided ensemble* (CGE) (Section 3.1) on $^{\blacklozenge}$. Here *UPop* uses a larger CLIP as its original model, and therefore GFLOPs is higher.

Approach	Image → Text			Text → Image			Avg.	GFLOPs	Throughput
	R@1	R@5	R@10	R@1	R@5	R@10	$\overline{\textbf{R@1}}$	↓	↑
CLIP (Radford et al., 2021)	92.1	99.1	99.7	79.3	95.7	98.0	85.7	20.6	255.2
TRIPS (Jiang et al., 2022)	90.4	98.9	99.5	76.8	94.4	97.2	83.6	16.4	316.9
UPop (Shi et al., 2023)	82.9	95.7	97.8	67.3	89.5	93.5	75.1	51.3	-
Hourglass (Liang et al., 2022a)	90.5	99.0	99.7	77.9	94.8	97.3	84.2	15.0	342.3
DynamicViT (Rao et al., 2021)	89.4	98.8	99.3	75.7	94.2	97.0	82.6	12.2	422.1
EViT (Liang et al., 2022b)	89.9	98.6	99.4	76.7	94.5	97.4	83.3	12.4	413.2
ToMe (Bolya et al., 2023)	$90.8_{\downarrow 1.3}$	$99.2_{\uparrow 0.1}$	$99.5_{\downarrow 0.2}$	$78.1_{\downarrow 1.2}$	$95.3_{\downarrow 0.4}$	$97.7_{\downarrow 0.3}$	$84.5_{\downarrow 1.2}$	11.8	417.4
ToMe+Extra Token	$90.8_{\downarrow 1.3}$	$98.7_{\downarrow 0.4}$	$99.6_{\downarrow 0.1}$	$78.8_{\downarrow 0.5}$	$95.1_{\downarrow 0.6}$	$97.6_{\downarrow 0.4}$	$84.8_{\downarrow 0.9}$	11.9	412.9
ToMe+CGM&CGE	$91.5_{\downarrow 0.6}$	$99.0_{\downarrow 0.1}$	$99.6_{\downarrow 0.1}$	$78.6_{\downarrow 0.7}$	$95.4_{\downarrow 0.3}$	$97.8_{\downarrow 0.2}$	$85.1_{\downarrow 0.6}$	11.9	409.9
CrossGET$^{\blacktriangle}$ (CGSM)	$90.9_{\downarrow 1.2}$	$99.2_{\uparrow 0.1}$	$\mathbf{99.9}_{\uparrow 0.2}$	$79.1_{\downarrow 0.2}$	$95.1_{\downarrow 0.6}$	$97.6_{\downarrow 0.4}$	$85.0_{\downarrow 0.7}$	11.9	408.9
CrossGET$^{\blacklozenge}$ (CGSM+CGM)	$\mathbf{92.1}_{\uparrow 0.0}$	$99.3_{\uparrow 0.2}$	$99.7_{\uparrow 0.0}$	$79.5_{\uparrow 0.2}$	$95.3_{\downarrow 0.4}$	$97.7_{\downarrow 0.3}$	$85.8_{\uparrow 0.1}$	12.0	402.1
CrossGET* (CGSM+CGM&CGE)	$\mathbf{92.1}_{\uparrow 0.0}$	$\mathbf{99.7}_{\uparrow 0.6}$	$99.8_{\uparrow 0.1}$	$\mathbf{79.6}_{\uparrow 0.3}$	$\mathbf{95.7}_{\uparrow 0.0}$	$\mathbf{98.0}_{\uparrow 0.0}$	$\mathbf{85.9}_{\uparrow 0.2}$	$12.0_{\downarrow 42\%}$	$401.8_{\uparrow 57\%}$

Table 2: **Accelerate BLIP on the NLVR2 dataset of the Vision Reasoning task.** BLIP is the original model for all approaches.

Approach	Dev Acc	Test Acc	GFLOPs	Throughput
BLIP (Li et al., 2022)	82.3	83.4	132.5	39.8
UPop (Shi et al., 2023)	$80.3_{\downarrow 2.0}$	$81.1_{\downarrow 2.3}$	89.4	-
ToMe (Bolya et al., 2023)	$81.7_{\downarrow 0.6}$	$82.2_{\downarrow 1.2}$	59.0	81.9
CrossGET$^{\blacktriangle}$ (CGSM)	$\mathbf{82.2}_{\downarrow 0.1}$	$82.6_{\downarrow 0.8}$	60.8	77.7
CrossGET* (Ours)	$82.1_{\downarrow 0.2}$	$\mathbf{83.2}_{\downarrow 0.2}$	$61.1_{\downarrow 57\%}$	$76.8_{\uparrow 93\%}$

Comparison with Baselines Unless stated otherwise, all reported experimental results are after training. Table 1 demonstrates that *CrossGET* outperforms both the SOTA multimodal model pruning approach UPop (Shi et al., 2023), token reduction approach TRIPS (Jiang et al., 2022), and other unimodal token reduction approachs (Bolya et al., 2023; Liang et al., 2022b; Rao et al., 2021; Liang et al., 2022a) without extra learnable parameter other than negligible cross tokens [3]. It can also be observed that simply adding an extra learnable token to unimodal approach ToMe does not bring a notable improvement. In particular, the average of Recall@1 is significantly lower than *CrossGET*, which indicates that the improvement given by *cross-guided matching and ensemble* is mainly from learning cross-modal information instead of the increase of learnable tokens.

Effect of individual components As highlighted by grey in Table 1, *complete-graph soft matching* (CGSM) brings improvements on most of the metrics and a significant improvement on text-to-image retrieval (recall@1 increases from 78.1 to 79.1). Since the complete graph has more similarity of token pairs to compute than the bipartite graph, GFLOPs also slightly increase by 0.1. *Cross-guided matching* (CGM) brings further improvement on most metrics and a significant improvement on image-to-text retrieval (recall@1 increases from 90.9 to 92.1). Since cross tokens interact with other tokens during the forward, GFLOPs again slightly increase by 0.1. *Cross-guided ensemble* brings final improvement on all metrics with negligible extra GFLOPs. Moreover, consistent improvements can also be observed when *Cross-Guided Matching and Ensemble* is applied to *ToMe*. Compared with the original CLIP, *CrossGET* achieves the same image-to-text recall@1 and 0.3 higher text-to-image recall@1 while saving 42% GFLOPs and improving throughput by 57%.

[3]For fairness of comparison, methods that require additional learnable parameters exceeding the level of several tokens are not taken into comparison (*e.g.*, simply adding a new linear projection layer with weight $W \in \mathbb{R}^{768\times768}$ already needs 768 times the number of our cross token's parameters $T_c \in \mathbb{R}^{1\times768}$)

4.2. Experiments with BLIP on Visual Reasoning

Table 2 shows *CrossGET* also achieves very competitive performance on the BLIP model and NLVR2 dataset of a vision reasoning task that requires predicting whether a given sentence can describe a pair of given images. Compared with the original BLIP, *CrossGET* gets only 0.2 lower accuracies on the dev set and test set while saving 57% GFLOPs and improving throughput by 93%.

4.3. Experiments at Different Reduction Ratios

Figure 4 illustrates experimental results at various reduction ratios under three different settings: (1) Comparisons without training (left subfigure). Note that the only part of *CrossGET* that requires training is learning cross tokens. However, they are initialized with informative features (see Appendix A.8) and already contain representative informa-

Table 3: **Accelerate BLIP on the COCO Caption dataset of the Image Caption task.** The suffix -F denotes GFLOPs and throughput for the forward, while -G denotes GFLOPs and throughput for the generation.

Approach	CIDEr	SPICE	GFLOPs-F	Throughput-F	GFLOPs-G	Throughput-G
BLIP (Li et al., 2022)	133.3	23.8	65.7	106.4	330.7	17.2
UPop (Shi et al., 2023)	128.9$_{\downarrow 4.4}$	23.3$_{\downarrow 0.5}$	39.8	-	-	-
ToMe (Bolya et al., 2023)	130.3$_{\downarrow 3.0}$	23.3$_{\downarrow 0.5}$	29.2	209.3	43.8	77.7
CrossGET (Ours)	**131.6**$_{\downarrow 1.7}$	**23.8**$_{\uparrow 0.0}$	30.1$_{\downarrow 54\%}$	183.5$_{\uparrow 72\%}$	46.7$_{\downarrow 86\%}$	73.9$_{\uparrow 330\%}$

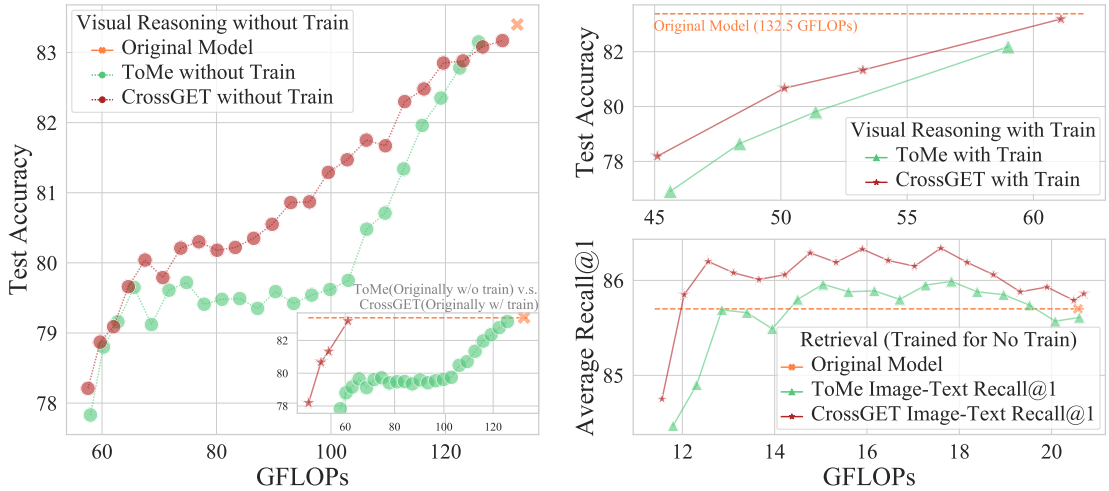

Figure 4: **Performance-Cost tradeoffs in three situations:** 1) The left subfigure illustrates the tradeoff for BLIP on the NVLR2 dataset of the Visual Reasoning task without training. 2) The upper-right subfigure illustrates the tradeoff for BLIP on the NVLR2 dataset of the Visual Reasoning task with training. 3) The lower-right subfigure illustrates the tradeoff for CLIP on the Flickr30K dataset of the Image-Text Retrieval task are trained with 50% token reduced and then re-evaluated under other token reduction ratios without training.

Table 4: **Accelerate BLIP on the NoCaps dataset of the Novel Object Caption task.** All metrics are the higher the better, and the evaluation uses the same model finetuned on the COCO Caption dataset as in Table 3, and therefore the GFLOPs and throughput of models are the same as in Table 3.

Approach	in-domain		near-domain		out-domain		**entire**	
	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE
BLIP (Li et al., 2022)	111.9	14.9	108.8	14.8	112.1	14.2	109.9	14.7
ToMe (Bolya et al., 2023)	107.9$_{\downarrow 4.0}$	14.8$_{\downarrow 0.1}$	105.1$_{\downarrow 3.7}$	14.4$_{\downarrow 0.4}$	106.4$_{\downarrow 5.7}$	**14.1**$_{\downarrow 0.1}$	105.7$_{\downarrow 4.2}$	14.4$_{\downarrow 0.3}$
CrossGET (Ours)	**113.2**$_{\uparrow 1.3}$	**15.1**$_{\uparrow 0.2}$	**107.2**$_{\downarrow 1.6}$	**14.6**$_{\downarrow 0.2}$	**107.4**$_{\downarrow 4.7}$	**14.1**$_{\downarrow 0.1}$	**108.1**$_{\downarrow 1.8}$	**14.6**$_{\downarrow 0.1}$

tion even though they are not trained. Therefore, *CrossGET* can also be used without training (certainly worse than with training); (2) Comparisons with training (upper-right subfigure). (3) Re-evaluate a trained model (50% token reduced) under other token reduction ratios without training (lower-right subfigure). These subfigures demonstrate that *CrossGET* achieves superior Pareto frontiers in all three situations. Appendix A.17, A.18, and A.19 provide detailed data. Besides, the original ToMe method does not require training, and the comparison with it is illustrated in the small plot at the lower right corner of the left subfigure.

4.4. Experiments with BLIP on Image Captioning

On auto-regressive models performing cross-modal interactions at each layer and forward via Cross-Attentions, such as the BLIP-Captioning (Li et al., 2022) model, *CrossGET* achieves higher speedups. As shown in Table 3, reducing the total tokens by half for the generation brings 86% saving of GFLOPs and improving 330% throughput.

We also conduct experiments on the NoCaps (Agrawal et al., 2019) datasets of the Novel Object Caption task, and the model accelerated by *CrossGET* again achieves superior performances on the entire task and all sub-tasks.

Table 5: **Accelerate BLIP on the VQA2.0 dataset of the Visual Question Answer task.** "yes/no", "number", "other", and "overall" denote accuracy on the corresponding types of questions. These four metrics are the higher the better. The suffix -F denotes GFLOPs and throughput for the forward that a single image may be accompanied by multiple questions and answers during training, while -T denotes GFLOPs and throughput for the test that a single image is accompanied by only one question and answer.

Approach	yes/no	number	other	**overall**	GFLOPs-F	Throughput-F	GFLOPs-T	Throughput-T
BLIP (Li et al., 2022)	92.6	60.6	68.3	77.4	186.1	67.2	106.8	53.0
UPop (Shi et al., 2023)	-	-	-	$76.3_{\downarrow 1.1}$	109.4	-	-	-
ToMe (Bolya et al., 2023)	$92.1_{\downarrow 0.5}$	$59.3_{\downarrow 1.3}$	$67.1_{\downarrow 1.2}$	$76.5_{\downarrow 0.9}$	119.0	141.1	46.7	90.1
CrossGET (Ours)	**92.4**$_{\downarrow 0.2}$	**59.7**$_{\downarrow 0.9}$	**67.7**$_{\downarrow 0.6}$	**77.0**$_{\downarrow 0.4}$	$124.5_{\downarrow 33\%}$	$120.4_{\uparrow 79\%}$	$49.0_{\downarrow 54\%}$	$81.3_{\uparrow 53\%}$

Table 6: **Accelerate multimodal LLM BLIP2-OPT6.7B on the COCO Caption dataset of the Image Caption task.** The suffix -F denotes GFLOPs and throughput for the forward, while -G denotes GFLOPs and throughput for the generation. * indicates using greedy decoding instead of beam search for generation. Experimental results on BLIP2-OPT2.7B are provided in Appendix A.15.

Approach	Tuning	CIDEr	BLEU@4	GFLOPs-F	Throughput-F	GFLOPs-G	Throughput-G	Throughput-G*
BLIP2-OPT6.7B	-	144.5	42.5	1042.6	47.4	2461.1	16.2	46.2
ToMe (Bolya et al., 2023)	w/o tuning	144.7	**42.4**	957.6	-	2342.7	-	-
	w/o tuning	144.3	42.3	868.1	-	2086.7	-	-
	w/o tuning	142.4	41.9	780.7	-	2232.4	-	-
	w/o tuning	135.5	40.1	695.1	-	2046.9	-	-
	w/ tuning	$141.7_{\downarrow 2.8}$	$41.4_{\downarrow 1.1}$	544.8	92.6	1510.0	21.5	75.1
CrossGET(Ours)	w/o tuning	144.5	42.3	973.8	-	2392.3	-	-
	w/o tuning	144.6	**42.4**	881.1	-	2266.2	-	-
	w/o tuning	143.3	**42.1**	790.9	-	2176.1	-	-
	w/o tuning	137.5	**40.6**	703.4	-	2121.8	-	-
	w/ tuning	**143.1**$_{\downarrow 1.4}$	**42.0**$_{\downarrow 0.5}$	$558.2_{\downarrow 49\%}$	$91.0_{\uparrow 92\%}$	$1583.2_{\downarrow 36\%}$	$21.6_{\uparrow 33\%}$	$75.7_{\uparrow 64\%}$

4.5. Experiments with BLIP on Visual QA

We conduct experiments on the BLIP model (Li et al., 2022) and the test-dev set of the VQA2.0 dataset (Goyal et al., 2017). Table 5 demonstrates that *CrossGET* can also considerably save computational cost and improve throughput for the Visual Question Answering task. For example, when compared with the original model, *CrossGET* gets only 0.4 lower overall accuracy on all three types of questions while saving 33% GFLOPs and improving throughput by 79% for the multiple-question scenario, and saving 54% GFLOPs and improving throughput by 53% for the single-question scenario.

4.6. Experiments with BLIP2 on Image Captioning

We apply CrossGET to the multimodal LLM BLIP2 (Li et al., 2023c). Following the original strategy of BLIP2, which tunes the ViT and Q-Former (a BERT) while freezing the LLM, we conduct experiments with and without tuning. Table 6 demonstrates that CrossGET consistently achieves promising performance on multimodal LLMs. Additionally, compared with BLIP, the language branch of BLIP2 receives fewer tokens from the vision branch, resulting in less generation speedup.

4.7. Experiments with LLaVA-1.5 on Various Datasets

For experiments on LLaVA-1.5 (Liu et al., 2023a), we followed its supervised fine-tuning (SFT) strategy, which tunes the LLM (*i.e.*, Vicuna (Chiang et al., 2023)) and projector (*i.e.*, MLP) while freezing the ViT. We evenly sampled 10% of data from the SFT dataset of LLaVA-1.5 as our training dataset. We observed that using more data provided limited improvement in performance recovery for models after acceleration. As shown in Table 7, with only 10% of the SFT data, CrossGET nearly doubles the throughput of the model forward and improves the throughput of generation by nearly 50%, while maintaining more than 98% of the original models' capabilities on average. Table 7 also indicates that with similar computational cost and throughput, LLaVA-1.5-13B after acceleration achieves better overall performance than LLaVA-1.5-7B without acceleration, which further demonstrates that instead of training smaller models from scratch, CrossGET can efficiently create more capable models from large-scale ones.

4.8. Experiments on CoOp Benchmark for Few-Shot Image Classification

We followed the same settings as CoOp (Zhou et al., 2022b), which uses 16 shots and freezes the backbone model CLIP

Table 7: **Accelerate multimodal LLM LLaVA-1.5-7B and LLaVA-1.5-13B.** Quantitative evaluation is conducted on ten widely used datasets. Tput represents throughput. The superscript ^{F} denotes GFLOPs and throughput for the forward, while ^{G} denotes GFLOPs and throughput for the generation. Details of each dataset are provided in Appendix A.4.

Approach	VQA^{v2}	GQA	VisWiz	SQA^{I}	VQA^{T}	POPE	MME	MMB	MMB^{CN}	$SEED^{I}$	$GFLOPs^{F}$	$Tput^{F}$	$GFLOPs^{G}$	$Tput^{G}$
LLaVA-1.5-7B	78.5	62.0	50.0	66.8	58.2	85.9	1510.7	64.3	58.3	66.2	4480.9	32.0	6216.7	1.7
with CrossGET	77.3	61.4	47.7	66.7	54.9	83.9	1510.2	64.7	55.2	64.4	$2382.5_{\downarrow 31\%}$	$60.4_{\uparrow 89\%}$	$4098.4_{\downarrow 34\%}$	$2.5_{\uparrow 47\%}$
LLaVA-1.5-13B	80.0	63.3	53.6	71.6	61.3	85.9	1531.3	67.7	63.6	68.2	8505.3	18.6	11862.0	1.1
with CrossGET	78.7	62.6	51.8	71.4	58.0	84.9	1548.8	66.3	62.0	67.5	$4500.3_{\downarrow 47\%}$	$37.0_{\uparrow 99\%}$	$7825.9_{\downarrow 34\%}$	$1.6_{\uparrow 45\%}$

Table 8: **Accelerate CLIP on the CoOp benchmark for the few-shot Image Classification task.** Following the same settings as CoOp, we use 16 shots and report top-1 accuracy on each of the 11 datasets. Details of each dataset are provided in Appendix A.3.

Approach	ImageNet	Caltech101	OxfordPets	StanfordCars	Flowers102	Food101	FGVCAircraft	SUN397	DTD	EuroSAT	UCF101	***Average***	GFLOPs
CoOp (Zhou et al., 2022b)	71.1	95.4	93.3	77.5	95.6	86.5	37.3	75.1	65.8	82.6	83.7	78.5	20.6
CoOp with CrossGET (Ours)	70.8	94.6	90.8	81.9	95.8	82.0	43.7	74.1	65.7	88.4	82.2	$79.1_{\uparrow 0.6}$	$16.5_{\downarrow 20\%}$
	70.2	94.9	90.1	81.1	95.0	81.5	43.1	73.5	65.9	86.9	81.9	$78.6_{\uparrow 0.1}$	$14.2_{\downarrow 31\%}$
	67.6	93.9	89.5	76.6	93.3	79.7	41.3	72.1	64.2	84.5	80.5	$76.7_{\downarrow 1.8}$	$12.0_{\downarrow 42\%}$

while conducting prompt tuning. The experimental results in Table 8 demonstrate that CrossGET achieves notable computational cost savings on few-shot Image Classification task[4]. For example, CrossGET achieves a 31% lossless computational cost saving according to the average top-1 accuracy over 11 datasets. Besides, the performance-cost trade-off on the CoOp benchmark is relatively worse than other experiments we have reported, which should be attributed to 1) most of the model parameters (*i.e.*, the whole backbone) are frozen, resulting in worse convergence than the full-parameter fine-tuning we have used for other experiments; 2) only a portion of the datasets are used for few-shot learning, resulting in more severe overfitting than when using the entire datasets in other experiments.

5. Conclusion

In this paper, we introduce *CrossGET*, a general token ensemble framework tailored for accelerating vision-language Transformers. *CrossGET* effectively utilizes bidirectional cross-modal guidance to make informed decisions on token selection and ensemble. Notably, our token-matching method is grounded on an approximate complete-graph matching algorithm, ensuring superior token-matching reliability in comparison to bipartite-graph approaches while maintaining parallelizability for high efficiency. In summary, *CrossGET* provides favorable performance-cost tradeoffs and demonstrates robust applicability, as evidenced through extensive empirical evaluations on a multitude of vision-language tasks, datasets, and model architectures.

[4]We report overall accuracy on all classes as in CoOp, rather than splitting classes into two groups and reporting separate accuracy as in CoCoOp (Zhou et al., 2022a).

Acknowledgements

This work was supported by the National Key R&D Program of China (2022YFB4701400/4701402), SSTIC Grant (KJZD20230923115106012), Shenzhen Key Laboratory (ZDSYS20210623092001004), Beijing Key Lab of Networked Multimedia, the National Key R&D Program of China (2022ZD0160201), and Shanghai Artificial Intelligence Laboratory (JF-P23KK00072).

Impact Statement

This paper introduces work aimed at advancing the field of model acceleration for vision-language Transformers. It outlines numerous potential positive societal impacts, such as saving electrical energy and reducing carbon dioxide emissions. Concerning negative aspects, while we believe that our work does not explicitly introduce any harmful impacts, it is important to consider potential indirect consequences. These may include reliance on technology that could reduce human involvement in certain tasks or the possibility of misusing accelerated models in ways that were not intended. However, these concerns are not directly linked to the inherent nature of our work.

References

Agrawal, H., Desai, K., Wang, Y., Chen, X., Jain, R., Johnson, M., Batra, D., Parikh, D., Lee, S., and Anderson, P. nocaps: novel object captioning at scale. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 8948–8957, 2019.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C., and Hoffman, J. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023.

Bossard, L., Guillaumin, M., and Van Gool, L. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pp. 446–461. Springer, 2014.

Chavan, A., Shen, Z., Liu, Z., Liu, Z., Cheng, K.-T., and Xing, E. P. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4931–4941, 2022.

Chen, D., Tao, C., Hou, L., Shang, L., Jiang, X., and Liu, Q. Litevl: Efficient video-language learning with enhanced spatial-temporal modeling. *arXiv preprint arXiv:2210.11929*, 2022a.

Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., and Luo, P. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022b.

Chen, T., Cheng, Y., Gan, Z., Yuan, L., Zhang, L., and Wang, Z. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021.

Chen, X., Fang, H., Lin, T.-Y., Vedantam, R., Gupta, S., Dollár, P., and Zitnick, C. L. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613, 2014.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Dai, W., Li, J., Li, D., Tiong, A. M. H., Zhao, J., Wang, W., Li, B., Fung, P., and Hoi, S. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Fan, A., Grave, E., and Joulin, A. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

Fang, Y., Wang, W., Xie, B., Sun, Q., Wu, L., Wang, X., Huang, T., Wang, X., and Cao, Y. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19358–19369, 2023.

Fang, Z., Wang, J., Hu, X., Wang, L., Yang, Y., and Liu, Z. Compressing visual-linguistic model via knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1428–1438, 2021.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pp. 178–178. IEEE, 2004.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Frantar, E. and Alistarh, D. Optimal brain compression: A framework for accurate post-training quantization and pruning. *arXiv preprint arXiv:2208.11580*, 2022.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023.

Gan, Z., Chen, Y.-C., Li, L., Chen, T., Cheng, Y., Wang, S., Liu, J., Wang, L., and Liu, Z. Playing lottery tickets with vision and language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 652–660, 2022.

Gao, P., Han, J., Zhang, R., Lin, Z., Geng, S., Zhou, A., Zhang, W., Lu, P., He, C., Yue, X., et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

Goyal, S., Choudhury, A. R., Raje, S., Chakaravarthy, V., Sabharwal, Y., and Verma, A. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pp. 3690–3699. PMLR, 2020.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Gurari, D., Li, Q., Stangl, A. J., Guo, A., Lin, C., Grauman, K., Luo, J., and Bigham, J. P. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3608–3617, 2018.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., and Neubig, G. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

He, Y., Zhang, X., and Sun, J. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

Helber, P., Bischke, B., Dengel, A., and Borth, D. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hoare, C. A. Quicksort. *The computer journal*, 5(1):10–16, 1962.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Huang, Y., Wang, W., and Wang, L. Instance-aware image and sentence matching with selective multimodal lstm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2310–2318, 2017.

Hudson, D. A. and Manning, C. D. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6700–6709, 2019.

Hyeon-Woo, N., Ye-Bin, M., and Oh, T.-H. Fedpara: Low-rank hadamard product for communication-efficient federated learning. *arXiv preprint arXiv:2108.06098*, 2021.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B., and Lim, S.-N. Visual prompt tuning. In *European Conference on Computer Vision*, pp. 709–727. Springer, 2022.

Jia, X., Gavves, E., Fernando, B., and Tuytelaars, T. Guiding long-short term memory for image caption generation, 2015. URL https://arxiv.org/abs/1509.04942.

Jiang, C., Xu, H., Li, C., Yan, M., Ye, W., Zhang, S., Bi, B., and Huang, S. Trips: Efficient vision-and-language pre-training with text-relevant image patch selection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 4084–4096, 2022.

Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., and Liu, Q. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

Karimi Mahabadi, R., Henderson, J., and Ruder, S. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34: 1022–1035, 2021.

Karpathy, A. and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3128–3137, 2015.

Karpathy, A., Joulin, A., and Fei-Fei, L. F. Deep fragment embeddings for bidirectional image sentence mapping. *Advances in neural information processing systems*, 27, 2014.

Khattak, M. U., Rasheed, H., Maaz, M., Khan, S., and Khan, F. S. Maple: Multi-modal prompt learning. *arXiv preprint arXiv:2210.03117*, 2022.

Kim, G. and Cho, K. Length-adaptive transformer: Train once with length drop, use anytime with search. *arXiv preprint arXiv:2010.07003*, 2020.

Kim, S., Shen, S., Thorsley, D., Gholami, A., Kwon, W., Hassoun, J., and Keutzer, K. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 784–794, 2022.

Kim, W., Son, B., and Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pp. 5583–5594. PMLR, 2021.

Kiros, R., Salakhutdinov, R., and Zemel, R. S. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Lassance, C., Maachou, M., Park, J., and Clinchant, S. A study on token pruning for colbert. *arXiv preprint arXiv:2112.06540*, 2021.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Li, B., Wang, R., Wang, G., Ge, Y., Ge, Y., and Shan, Y. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023a.

Li, D., Li, J., Le, H., Wang, G., Savarese, S., and Hoi, S. C. LAVIS: A one-stop library for language-vision intelligence. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 31–41. Association for Computational Linguistics, July 2023b.

Li, J., Selvaraju, R., Gotmare, A., Joty, S., Xiong, C., and Hoi, S. C. H. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34: 9694–9705, 2021.

Li, J., Li, D., Xiong, C., and Hoi, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *arXiv preprint arXiv:2201.12086*, 2022.

Li, J., Li, D., Savarese, S., and Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023c.

Li, J., Pan, K., Ge, Z., Gao, M., Ji, W., Zhang, W., Chua, T.-S., Tang, S., Zhang, H., and Zhuang, Y. Fine-tuning multimodal llms to follow zero-shot demonstrative instructions. In *The Twelfth International Conference on Learning Representations*, 2023d.

Li, X., Yin, X., Li, C., Zhang, P., Hu, X., Zhang, L., Wang, L., Hu, H., Dong, L., Wei, F., et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pp. 121–137. Springer, 2020.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W. X., and Wen, J.-R. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023e.

Liang, W., Yuan, Y., Ding, H., Luo, X., Lin, W., Jia, D., Zhang, Z., Zhang, C., and Hu, H. Expediting large-scale vision transformer for dense prediction without fine-tuning. *Advances in Neural Information Processing Systems*, 35:35462–35477, 2022a.

Liang, Y., Ge, C., Tong, Z., Song, Y., Wang, J., and Xie, P. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022b.

Lin, J., Tang, J., Tang, H., Yang, S., Dang, X., and Han, S. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023a.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023b.

Liu, Y., Duan, H., Zhang, Y., Li, B., Zhang, S., Zhao, W., Yuan, Y., Wang, J., He, C., Liu, Z., et al. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023c.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M., and Vedaldi, A. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

OpenAI. Gpt-4v(ision) system card, 2023. URL https://cdn.openai.com/papers/GPTV_System_Card.pdf.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., and Hsieh, C.-J. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Shi, D., Liu, R., Tao, L., He, Z., and Huo, L. Multi-encoder parse-decoder network for sequential medical image segmentation. In *2021 IEEE international conference on image processing (ICIP)*, pp. 31–35. IEEE, 2021.

Shi, D., Liu, R., Tao, L., and Yuan, C. Heuristic dropout: An efficient regularization method for medical image segmentation models. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1101–1105. IEEE, 2022.

Shi, D., Tao, C., Jin, Y., Yang, Z., Yuan, C., and Wang, J. UPop: Unified and progressive pruning for compressing vision-language transformers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 31292–31311. PMLR, 2023.

Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8317–8326, 2019.

Singh, A., Hu, R., Goswami, V., Couairon, G., Galuba, W., Rohrbach, M., and Kiela, D. Flava: A foundational language and vision alignment model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15638–15650, 2022.

Soomro, K., Zamir, A. R., and Shah, M. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., and Xu, C. Vitas: vision transformer architecture search. In *European Conference on Computer Vision*, pp. 139–157. Springer, 2022.

Suhr, A., Zhou, S., Zhang, A., Zhang, I., Bai, H., and Artzi, Y. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018.

Sung, Y.-L., Cho, J., and Bansal, M. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35: 12991–13005, 2022a.

Sung, Y.-L., Cho, J., and Bansal, M. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5227–5237, 2022b.

Tao, C., Hou, L., Zhang, W., Shang, L., Jiang, X., Liu, Q., Luo, P., and Wong, N. Compression of generative pre-trained language models via quantization. *arXiv preprint arXiv:2203.10705*, 2022.

Tao, C., Hou, L., Bai, H., Wei, J., Jiang, X., Liu, Q., Luo, P., and Wong, N. Structured pruning for efficient generative pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10880–10895, 2023.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

Wang, H., Zhang, Z., and Han, S. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 97–110. IEEE, 2021.

Wang, J., Hu, X., Zhang, P., Li, X., Wang, L., Zhang, L., Gao, J., and Liu, Z. Minivlm: A smaller and faster vision-language model. *arXiv preprint arXiv:2012.06946*, 2020a.

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *arXiv preprint arXiv:2202.03052*, 2022a.

Wang, T., Zhou, W., Zeng, Y., and Zhang, X. Efficientvlm: Fast and accurate vision-language models via knowledge distillation and modal-adaptive pruning. *arXiv preprint arXiv:2210.07795*, 2022b.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33: 5776–5788, 2020b.

Xiao, G., Lin, J., Seznec, M., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. *arXiv preprint arXiv:2211.10438*, 2022.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 3485–3492. IEEE, 2010.

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 21–29, 2016.

Yang, Z., Li, Z., Shao, M., Shi, D., Yuan, Z., and Yuan, C. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022.

Ye, Q., Xu, H., Xu, G., Ye, J., Yan, M., Zhou, Y., Wang, J., Hu, A., Shi, P., Shi, Y., et al. mplug-owl: Modularization empowers large language models with multimodality. 2023.

Yin, H., Vahdat, A., Alvarez, J. M., Mallya, A., Kautz, J., and Molchanov, P. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10809–10818, 2022.

Yin, S., Fu, C., Zhao, S., Li, K., Sun, X., Xu, T., and Chen, E. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

Young, P., Lai, A., Hodosh, M., and Hockenmaier, J. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Yuan, Z., Niu, L., Liu, J., Liu, W., Wang, X., Shang, Y., Sun, G., Wu, Q., Wu, J., and Wu, B. Rptq: Reorder-based post-training quantization for large language models, 2023.

Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., and Ma, K. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3713–3722, 2019.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V.,

et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16816–16825, 2022a.

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022b.

Zhu, D., Chen, J., Shen, X., Li, X., and Elhoseiny, M. Minigpt-4: Enhancing vision-language understanding with advanced large language models. 2023.

Zhu, M., Tang, Y., and Han, K. Vision transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021.

A. Supplementary Experiments and Details

A.1. Diagram of Adding Cross Tokes to Different Models

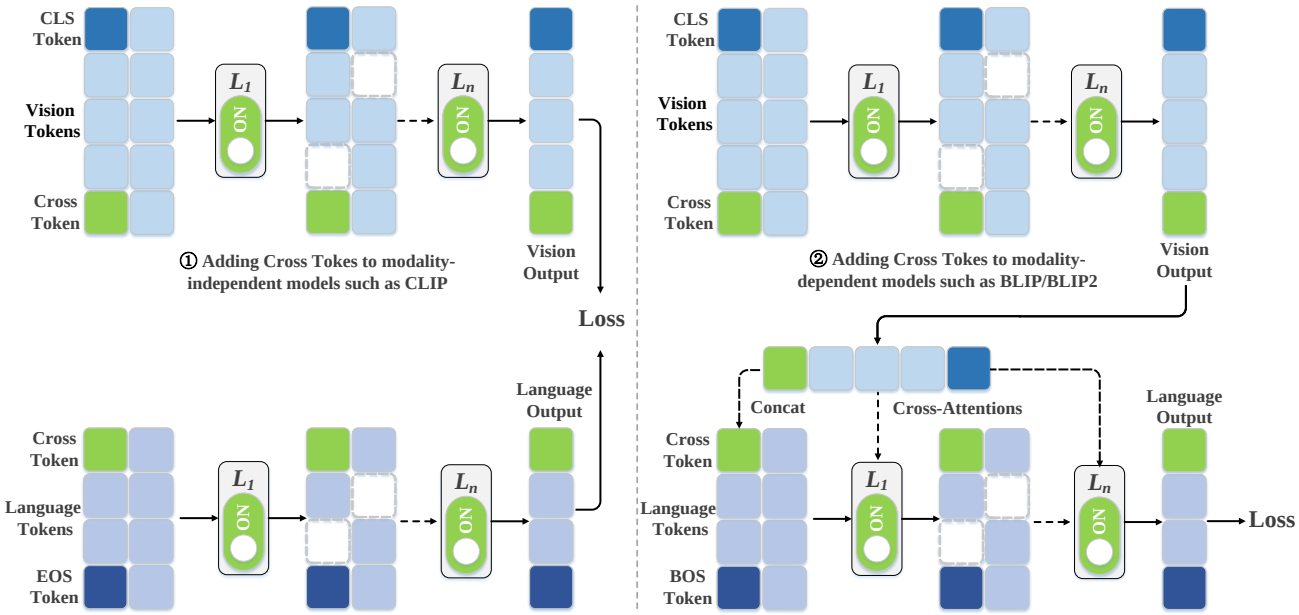

Figure 5: Diagram of adding cross tokes to modality-independent models such as CLIP (Radford et al., 2021) (left) and modality-dependent models such as BLIP/BLIP2 (Li et al., 2022; 2023c) (right).

Figure 5 demonstrates that *CrossGET* is designed to be a general framework that can be used for accelerating both modality-independent vision-language models such as CLIP (Radford et al., 2021) model and modality-dependent vision-language models such as BLIP/BLIP2 (Li et al., 2022; 2023c) -based models.

Two different types of dependencies exist for modality-dependent vision-language models. The first type that BLIP (Li et al., 2022) belongs to is that the succeeding modality interacts with the final output of the preceding modality through Cross-Attention modules. For this type, in addition to reducing the number of its own tokens, the succeeding modality can also be accelerated by reducing the number of tokens output from the preceding modality to speed up Cross-Attentions.

The second type BLIP2 (Li et al., 2023c) -based models such as InstructBLIP (Dai et al., 2023), MiniGPT-4 (Zhu et al., 2023), and mPLUG-Owl (Ye et al., 2023) belong to is that the succeeding modality takes the final output of the preceding modality as part of its input sequence to the first layer, and the cross-modal interaction is conducted by Self-Attentions in the succeeding modality. For the second type, the succeeding modality can be accelerated by reducing the length of the cross-modal input sequence to speed up Self-Attentions and FFNs in the succeeding modality.

Take BLIP2 as an example. BLIP2 consists of a ViT for processing visual input, a Q-Former (a Bert) for bridging modalities, and an LLM for taking inputs from Q-Former and generating text output accordingly. During fine-tuning, ViT and Q-Former are tunable while the LLM is frozen. To accelerate BLIP2, we can reduce the number of tokens in both ViT and Q-Former. It is worth noting that there are two ways to reduce the number of tokens processed by LLM. The first one is reducing the number of tokens in Q-Former. Since LLM takes Q-Former's output as its input, the token reduction conducted in Q-Former leads to LLM acceleration. This setting gives the default performance we reported in the paper. The second one is directly ensembling tokens in LLM. We have also tested this setting and discussed it in Appendix A.16.

Recently, vision-language models that take interleaved sequences as inputs, such as Flamingo (Alayrac et al., 2022) and VPG-C (Li et al., 2023d), also attracted much attention. CrossGET can be further extended to these models since they can still be categorized as either modality-dependent or modality-independent models. However, some modifications should be made to the concrete strategy for matching tokens. More specifically, for interleaved vision-language inputs, we also need to consider cross-fragment guidance between different image/text fragments within the same modality, which will be more complicated than single image-text scenarios where we only consider cross-modal guidance between different modalities.

A.2. Hyperparameter Settings

Table 9: Training hyperparameters for accelerating BLIP-based models.

Hyperparameters	BLIP-NLVR (Li et al., 2022)	BLIP-Captioning (Li et al., 2022)		BLIP-VQA (Li et al., 2022)
	NLVR2 (Suhr et al., 2018)	COCO Caption (Chen et al., 2015)	NoCaps (Agrawal et al., 2019)	VQAv2 (Goyal et al., 2017)
Optimizer	AdamW(Loshchilov & Hutter, 2017)			
AdamW β	(0.9, 0.999)			
Batch size	512			
Weight decay	0.05	0.05	0.05	0.05
Epochs	15	5	5	10
Initial learning rate	3×10^{-6}	1×10^{-5}	1×10^{-5}	2×10^{-5}
Learning rate schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)			
Data augmentation	RandomAugment (Cubuk et al., 2020)			
Training Precision	Mixed Precision (Micikevicius et al., 2017)			
Matching loss coefficient	10^1	10^2	10^2	10^1

Table 10: Training hyperparameters for accelerating CLIP and BLIP2-based models.

Hyperparameters	CLIP-Retrieval (Radford et al., 2021)	BLIP2-OPT2.7B-Captioning (Li et al., 2023c)	BLIP2-OPT6.7B-Captioning (Li et al., 2023c)
	Flickr30K (Young et al., 2014)	COCO Caption (Chen et al., 2015)	COCO Caption (Chen et al., 2015)
Optimizer	AdamW (Loshchilov & Hutter, 2017)		
AdamW β	(0.9, 0.999)		
Batch size	512	1024	512
Weight decay	0.2	0.05	0.05
Epochs	12	5	5
Initial learning rate	1×10^{-5}	1×10^{-5}	1×10^{-5}
Learning rate schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)		
Data augmentation	RandomAugment (Cubuk et al., 2020)		
Training Precision	Mixed Precision (Micikevicius et al., 2017)		
Matching loss coefficient	10^0	10^{-1}	10^{-1}

Table 11: Training hyperparameters for accelerating CoOp and LLaVA-1.5 models.

Hyperparameters	CLIP-CoOp (Zhou et al., 2022b)	LLaVA-1.5-7B (Liu et al., 2023a)	LLaVA-1.5-13B (Liu et al., 2023a)
	See Appendix A.3	SFT data of LLaVA-1.5	SFT data of LLaVA-1.5
Optimizer	AdamW (Loshchilov & Hutter, 2017)		
AdamW β	(0.9, 0.999)		
Batch size	EuroSAT: 128. Others: 256	128	128
Weight decay	0.0005	0	0
Epochs	ImageNet: 50. Others: 200	1	1
Initial learning rate	ImageNet: 2×10^{-2}. Others: 5×10^{-2}	2×10^{-5}	2×10^{-5}
Learning rate schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)		
Matching loss coefficient	10^1	10^{-3}	10^{-1}

The hyperparameters about model training are listed in Table 9, Table 10, and Table 11. The hyperparameters about model structures are listed in Table 12.

Table 12: Structure hyperparameters for all models used in our experiments. The superscript * indicates 2 Transformers share parameters. The superscript † indicates hyperparameters are from (OPT, Q-Former).

Model	Input resolution	Vision Transformer (Touvron et al., 2021; Fang et al., 2023)				Language Transformer (Devlin et al., 2018; Zhang et al., 2022)			
		number	layers	width	heads	number	layers	width	heads
CLIP-Retrieval	336×336	1	12	768	12	1	12	512	8
CLIP-CoOp	336×336	1	12	768	12	1	12	512	8
BLIP-NLVR	384×384	2*	(12, 12)	(768, 768)	(12, 12)	1	12	768	12
BLIP-Captioning	384×384	1	12	768	12	1	12	768	12
BLIP-NoCaps	384×384	1	12	768	12	1	12	768	12
BLIP-VQA	480×480	1	12	768	12	2	(12, 12)	(768, 768)	(12, 12)
BLIP2-OPT2.7B	364×364	1	39	1408	16	2†	(32, 12)	(2560, 768)	(32, 12)
BLIP2-OPT6.7B	364×364	1	39	1408	16	2†	(32, 12)	(4096, 768)	(32, 12)
LLaVA-1.5-7B	336×336	1	24	1024	16	1	32	4096	32
LLaVA-1.5-13B	336×336	1	24	1024	16	1	40	5120	40

A.3. Evaluation Datasets for CoOp

The CoOp benchmark (Zhou et al., 2022b) consists of 11 datasets, which are ImageNet (1000 classes) (Deng et al., 2009), Caltech101 (100 classes) (Fei-Fei et al., 2004), OxfordPets (37 classes) (Parkhi et al., 2012), StanfordCars (196 classes) (Krause et al., 2013), Flowers102 (102 classes) (Nilsback & Zisserman, 2008), Food101 (101 classes) (Bossard et al., 2014), FGVCAircraft (100 classes) (Maji et al., 2013), SUN397 (397 classes) (Xiao et al., 2010), DTD (47 classes) (Cimpoi et al., 2014), EuroSAT (10 classes) (Helber et al., 2019), and UCF101 (101 classes) (Soomro et al., 2012). For each dataset, we randomly sample 16 images in each class as its training set for few-shot learning.

A.4. Evaluation Datasets for LLaVA-1.5

After applying CrossGET to LLaVA-1.5 (Liu et al., 2023a), we used 10 datasets to evaluate model performance, including VQA-v2 (Goyal et al., 2017), GQA (Hudson & Manning, 2019), VisWiz (Gurari et al., 2018), ScienceQA-IMG (Lu et al., 2022), TextVQA (Singh et al., 2019), POPE (Li et al., 2023e), MME (Yin et al., 2023), MMBench (Liu et al., 2023c), MMBench-CN (Liu et al., 2023c), and SEED-Bench-Image (Li et al., 2023a).

A.5. Ablation Study on Training Hyperparameters

The hyperparameters are basically inherited from original models and do not need a specific tune. The particular case is that batch sizes are adjusted to fit our computational resources. The only additional hyperparameter introduced by *CrossGET* is the matching loss coefficient α, which is used to balance the original loss items and the matching loss item. For simplicity, α can be determined as the number $\alpha \in \{10^i\}_{i\in\mathbb{N}}$ that makes the original loss items and the matching loss item have the closest order of magnitude, and therefore, it does not need to be tuned either.

Table 13: Ablation study about batch size on BLIP-NLVR.

Batch size	Dev Acc	Test Acc
128	$82.0_{\downarrow 0.1}$	$82.8_{\downarrow 0.4}$
256	$\mathbf{82.2}_{\uparrow 0.1}$	$83.0_{\downarrow 0.2}$
512	82.1	**83.2**
1024	$\mathbf{82.2}_{\uparrow 0.1}$	$83.0_{\downarrow 0.2}$

Table 14: Ablation study about learning rate on BLIP-NLVR.

Learning rate	Dev Acc	Test Acc
1×10^{-6}	$81.8_{\downarrow 0.3}$	$82.5_{\downarrow 0.7}$
3×10^{-6}	82.1	**83.2**
1×10^{-5}	$\mathbf{82.2}_{\uparrow 0.1}$	$82.7_{\downarrow 0.5}$
3×10^{-5}	$82.0_{\downarrow 0.1}$	$82.6_{\downarrow 0.6}$

Table 15: Ablation study about coefficient α for matching loss on BLIP-NLVR.

Coefficient	Dev Acc	Test Acc
10^0	$82.0_{\downarrow 0.1}$	$82.5_{\downarrow 0.7}$
10^1	**82.1**	**83.2**
10^2	$81.8_{\downarrow 0.3}$	$82.7_{\downarrow 0.5}$

Table 13, Table 14, and Table 15 investigate how hyperparameters affect the model performance. Experimental results show that the performance is insensitive to batch size and slightly sensitive to the learning rate. As for the matching loss coefficient $\alpha \in \{10^i\}_{i\in\mathbb{N}}$, set it to the value that makes the original loss items and the matching loss item have the closest order of magnitude as mentioned above will work well.

A.6. Ablation Study on Different Modalities

Table 16: Ablation study about applying *CrossGET* on different modalities.

Modality	I2T R@1	T2I R@1	GFLOPs
vision only	92.1	79.6	12.0
language only	**92.8**$_{\uparrow 0.7}$	**80.4**$_{\uparrow 0.8}$	19.3$_{\uparrow 61\%}$
vision and language	91.4$_{\downarrow 0.7}$	78.3$_{\downarrow 1.3}$	**10.6**$_{\downarrow 12\%}$

As shown in Figure 2, it is flexible that *CrossGET* can be applied on both vision and language modalities or only on one of the modalities. Table 16 investigates the trade-off between model performance and computational cost of application on different modalities. Experimental results show that *CrossGET* only on the vision modality achieves the best trade-off.

A.7. Ablation Study on the Strategy of Adding Cross Token

Table 17: Ablation study about the strategy of adding cross tokens.

Depth	I2T R@1	T2I R@1	GFLOPs
shallow	91.5$_{\downarrow 0.6}$	79.5$_{\downarrow 0.1}$	12.0
deep	**92.1**	**79.6**	12.0
share	90.7$_{\downarrow 1.4}$	78.8$_{\downarrow 0.8}$	12.0

There are several strategies for injecting cross tokens into the model. For example, (1) deep: adding different cross tokens for each layer; (2) shallow: only adding one cross token into the first layer; (3) share: adding one cross token but jointly optimized in each layer. Table 17 shows that adding different cross tokens for each layer achieves the best performance.

A.8. Ablation Study on the Initialization of Cross Token

Table 18: Ablation study about initializing cross tokens.

Initialization	I2T R@1	T2I R@1	GFLOPs
zero	91.7$_{\downarrow 0.4}$	77.9$_{\downarrow 1.7}$	12.0
normal random	90.4$_{\downarrow 1.7}$	77.6$_{\downarrow 2.0}$	12.0
uniform random	90.2$_{\downarrow 1.9}$	77.6$_{\downarrow 2.0}$	12.0
informative tokens	**92.1**	**79.6**	12.0

For fine-tuning, cross tokens are kind of sensitive to the initialization strategy. Using informative tokens to initialize cross tokens is recommended. More specifically, for the vision modality, [CLS] token can be used to initialize the cross token. For the language modality, the cross token can be initialized by [CLS]/[EOS]/[EOT] tokens for discriminative tasks (it depends on which token is ultimately used to calculate the loss) and by [BOS] token for auto-regressive tasks (if there is no, we use the first token of the input sequence to initialize instead).

Table 18 shows that zero initialization and random initialization perform worse. We think the sensitivity should be attributed to the limited training time for fine-tuning and the purpose of quickly adapting to downstream tasks. More specifically, random/zero initialization may work well for pre-training since there is enough time for cross tokens to learn informative guidance. However, it will be difficult for random/zero initialization to learn well with limited iterations for fine-tuning. Therefore, initializing the cross token with [CLS] token in the vision modality and [CLS]/[BOS]/[EOS]/[EOT] token in the language modality implies that the cross token already contains some informative guidance of the modality it is in, and would be easier to form more informative guidance with this good starting point.

A.9. Ablation Study on the Projection Layer Detach

The final projection layers are initially used to project features from the different modalities into aligned representations. In *CrossGET*, the final projection layers are detached from the original model and used for aligning cross tokens. The detach operation prevents gradients with respect to cross tokens from updating the projection layers. Table 19 shows that detaching both vision and language projection improves performance.

Table 19: Ablation study about projection layer detach.

Projection detach	I2T R@1	T2I R@1	GFLOPs
neither	91.5$_{\downarrow 0.6}$	78.9$_{\downarrow 0.7}$	12.0
vision only	91.4$_{\downarrow 0.7}$	79.4$_{\downarrow 0.2}$	12.0
language only	90.5$_{\downarrow 1.6}$	78.9$_{\downarrow 0.7}$	12.0
both	**92.1**	**79.6**	12.0

A.10. Ablation Study on the Number of Cross Tokens

Table 20: Ablation study on number of cross tokens.

Number	Dev Acc	Test Acc	GFLOPs
1	82.1	**83.2**	**61.1**
2	**82.2**$_{\uparrow 0.1}$	**83.2**$_{\uparrow 0.0}$	61.4$_{\uparrow 0.3}$
3	81.9$_{\downarrow 0.2}$	**83.2**$_{\downarrow 0.0}$	61.8$_{\uparrow 0.7}$
4	82.0$_{\downarrow 0.1}$	82.9$_{\downarrow 0.3}$	62.2$_{\uparrow 1.1}$

Table 20 investigates how the performance is impacted by the number of cross tokens on the Vision Reasoning Task and BLIP (Li et al., 2022) model. It can be observed that the performance is not sensitive to the increase in the number of tokens, which is unlike prompt tuning (Lester et al., 2021; Jia et al., 2022) that model performance can be boosted by increasing the number of tokens. Considering the additional computational cost of multiple cross tokens, using only one cross token is recommended.

A.11. Ablation Study on Tokens for Computing Importance

For BLIP (Li et al., 2022) on the Visual Reasoning task, a different setting from default is that not cross tokens alone, but all tokens are used to compute importance. By default, in the modality-independent model CLIP (Radford et al., 2021), only the [CLS] and [EOS] tokens are ultimately used for computing loss. In contrast, for the modality-dependent model BLIP-NLVR, all tokens output from the vision modality are parts of the inputs for the language modality and matter.

Table 21: Ablation study about tokens for computing importance.

used tokens	Dev Acc	Test Acc
cross token	82.1$_{\uparrow 0.0}$	82.9$_{\downarrow 0.3}$
other tokens	81.9$_{\downarrow 0.2}$	82.2$_{\downarrow 1.0}$
all tokens	82.1	**83.2**
importance	**82.2**$_{\uparrow 0.1}$	83.0$_{\downarrow 0.2}$

Four settings about which tokens are used for computing importance are tested as shown in Table 21: (1) cross token: cross tokens contribute all; (2) other tokens: other tokens contribute all; (3) all tokens (adopted): cross tokens contribute to $\frac{1}{2}$ importance while other tokens contribute to the other $\frac{1}{2}$; (4) importance: we can reuse the dot product between the query and key of each token (including cross tokens) that has already been calculated in the Self-Attention as the importance metric to avoid extra computational cost for introducing other tokens' importance.

A.12. Ablation Study on Tokens for Computing JS divergence

For BLIP (Li et al., 2022) on the Image Caption task, a different setting from default is that not only the loss of JS divergence between the pairs of cross tokens but also the JS divergence between the cross tokens and other tokens should be added as loss items. By default, in the language modality of the discriminative model CLIP (Radford et al., 2021), only the [EOS] token matters for the final output. In contrast, for the auto-regressive model BLIP-Captioning, tokens are generated based on their previous tokens, and therefore, every token matters.

Table 22 shows that combined JS divergence between pairs of cross tokens as well as between cross tokens and other tokens as loss performs best. Besides, weighting the loss between cross tokens and other tokens according to the generation order also helps. The weight for the i-th generated token is $1 - \frac{i}{L}$ where L is the maximum generation length, which means the first generated token is more important than the later ones since they are generated based on former ones.

Table 22: Ablation study about which tokens are used for computing JS divergence as additional loss items.

JS divergence as loss	CIDEr	SPICE
only between pairs of cross tokens	130.2$_{\downarrow 1.4}$	23.7$_{\downarrow 0.1}$
only between cross tokens and other tokens	131.0$_{\downarrow 0.6}$	23.5$_{\downarrow 0.3}$
w/o weighting loss according to generation order	131.2$_{\downarrow 0.4}$	23.7$_{\downarrow 0.1}$
between cross tokens and all tokens	**131.6**	**23.8**

A.13. Comparison Experiments with Text-Relevant Image Patch Selection

Table 23: **Accelerate CLIP on the Flickr30K dataset of the Image-Text Retrieval task.** R: Recall. R@1, R@5, and R@10 are the higher the better. The TRIPS represents Text-Relevant Image Patch Selection (Jiang et al., 2022). The -L indicates using an additional learnable projection to align the text [CLS] token with vision tokens.

Approach	Image → Text			Text → Image			Avg.	GFLOPs	Throughput
	R@1	R@5	R@10	R@1	R@5	R@10	$\overline{\mathbf{R@1}}$	↓	↑
CLIP (Radford et al., 2021)	92.1	99.1	99.7	79.3	95.7	98.0	85.7	20.6	255.2
TRIPS (Default FLOPs)	87.6	98.7	99.4	76.6	94.4	97.0	82.1	16.4	317.7
TRIPS-L (Default FLOPs)	90.4	98.9	99.5	76.8	94.4	97.2	83.6	16.4	316.9
TRIPS (Same FLOPs)	75.5	94.3	97.8	63.9	88.5	93.8	69.7	12.0	423.5
TRIPS-L (Same FLOPs)	70.1	92.4	96.9	61.2	86.8	92.1	65.7	12.0	423.1
CrossGET (Ours)	**92.1**$_{\uparrow 0.0}$	**99.7**$_{\uparrow 0.6}$	**99.8**$_{\uparrow 0.1}$	**79.6**$_{\uparrow 0.3}$	**95.7**$_{\uparrow 0.0}$	**98.0**$_{\uparrow 0.0}$	**85.9**$_{\uparrow 0.2}$	12.0$_{\downarrow 42\%}$	401.8$_{\uparrow 57\%}$

In Table 23, the TRIPS (Default FLOPs) (Jiang et al., 2022) indicates we follow the recommended setting of the original TRIPS, *i.e.*, we take the 5th and 10th as the patch-selection layer and set the keep ratio of each layer to 70%. The TRIPS (Same FLOPs) indicates we decrease the keep ratio of each patch-selection layer to achieve similar GFLOPs with ToMe (Bolya et al., 2023) and CrossGET. Overall, the experimental results demonstrate that CrossGET outperforms TRIPS under similar computational costs.

When compared with TRIPS, one of the *CrossGET*'s advantages is that it can more easily deal with the models in which the embedding sizes of vision and language branches are different. More specifically, TRIPS is not directly applicable to the models with different embedding sizes of the vision branch and language branch, and without a projection layer that projects the language embedding size into the vision embedding size as well.

For example, the vision and language embedding sizes in the CLIP model we used are 768 and 512, respectively. Besides, there is a 768 → 512 projection layer for vision projection and a 512 → 512 for language projection. TRIPS requires the projected text [CLS] token to have the same embedding size as the tokens in the vision branch. However, CLIP has no trained (*i.e.*, aligned) 512 → 768 projection layer to fulfill this. To overcome this problem, we propose two strategies: 1) The first one is to use the pseudo-inverse of the trained 768 → 512 projection layer to project the 512-dimensional text [CLS] token into a 768-dimensional token, whose experimental results are denoted without -L. 2) The second one is to add an additional 512 → 768 learnable projection layer into the original model and then jointly optimize, whose experimental results are denoted with -L. Note that this is not a problem for *CrossGET* since cross tokens are learned cross-modally while used intra-modally, and the embedding size of cross tokens is the same as other tokens in the same modality branch. Thus, CrossGET doesn't need a projection layer to align cross tokens when they are used as metrics to guide the token reduction.

The other advantage of CrossGET is that it can deal with both modality-independent and modality-dependent models, which is also an important contribution of CrossGET. We have discussed this in Section 3.1, and TRIPS can serve as an example to elaborate it. More specifically, TRIPS uses text [CLS] token, *i.e.*, the output of the language encoder in the ALBEF (Li et al., 2021) model as the metric to guide the token reduction in the vision branch. However, this paradigm cannot be used in multimodal models where the input of the language branches depends on the output of the vision branch.

For example, in the BLIP-NLVR (Li et al., 2022) model, the output of the vision branch is a necessary input for the language branch. And if we want to use the text [CLS] token *i.e.*, the output of the language branch as a metric to guide the token reduction in the vision branch, we have to first forward through the vision branch, get the last layer's output as the input of the language branch, forward through the language branch, get the last layer's output as the metric, *i.e.*, only after the forward of the vision branch is finished, we can get the required metric used for vision branch. CrossGET breaks this

paradox of cycles by using cross tokens as proxies for other modalities, providing cross-modal guidance on behalf of other modalities without being constrained by the order of calculations.

A.14. Comparison Experiments with Adpater

Table 24: **Accelerate CLIP on the Flickr30K dataset of the Image-Text Retrieval task.** R: Recall. R@1, R@5, and R@10 are the higher the better. The Adapter-x represents Adaptformer (Chen et al., 2022b), and the integer x in Adapter-x represents the middle dimension of the adapter.

Approach	Image → Text			Text → Image			Avg.	GFLOPs	Throughput
	R@1	R@5	R@10	R@1	R@5	R@10	**R@1**	↓	↑
CLIP (Radford et al., 2021)	92.1	99.1	99.7	79.3	95.7	98.0	85.7	20.6	255.2
ToMe (Bolya et al., 2023)	90.8	99.2	99.5	78.1	95.3	97.7	84.5	11.8	417.4
ToMe❄+Adapter-16🔥	89.2	98.7	99.6	75.9	94.2	97.1	82.6	11.9	404.1
ToMe❄+Adapter-64🔥	89.9	98.8	99.5	76.7	94.4	97.3	83.3	12.0	401.2
ToMe❄+Adapter-256🔥	90.2	99.0	99.4	76.7	94.5	97.5	83.5	12.5	386.4
ToMe❄+Adapter-1024🔥	90.3	99.0	**99.8**	78.0	94.6	97.4	84.2	14.5	346.2
ToMe❄+Adapter-4096🔥	91.4	98.8	99.6	78.2	95.0	97.6	84.8	22.7	243.5
CrossGET (Ours)	**92.1**$_{\uparrow 0.0}$	**99.7**$_{\uparrow 0.6}$	**99.8**$_{\uparrow 0.1}$	**79.6**$_{\uparrow 0.3}$	**95.7**$_{\uparrow 0.0}$	**98.0**$_{\uparrow 0.0}$	**85.9**$_{\uparrow 0.2}$	12.0$_{\downarrow 42\%}$	401.8$_{\uparrow 57\%}$

The experimental results demonstrate that when using ToMe (Bolya et al., 2023) with an adapter (Chen et al., 2022b), the middle dimension of the adapter needs to be very large(*e.g.*, around 4096) for the model to perform better than without using the adapter. However, the additional computation cost introduced by the adapter is significant (see GFLOPs and Throughput in the above Table 24), and the performance is still worse than CrossGET.

A.15. Experiments with BLIP2 on Image Captioning

Table 25: Accelerate multimodal LLM BLIP2-OPT2.7B (Li et al., 2023c) on the COCO Caption dataset of the Image Caption task. The suffix -F denotes GFLOPs and throughput for the forward, while -G denotes GFLOPs and throughput for the generation. * indicates using greedy decoding instead of beam search for generation.

Approach	Tuning	CIDEr	BLEU@4	GFLOPs-F	Throughput.-F	GFLOPs-G	Throughput.-G	Throughput.-G*
BLIP2-OPT2.7B	-	145.6	42.8	854.2	54.0	1379.3	22.3	52.4
ToMe (Bolya et al., 2023)	w/o tuning	145.1	42.6	769.1	-	1294.2	-	-
	w/o tuning	144.2	42.3	679.7	-	1218.1	-	-
	w/o tuning	142.8	42.2	592.2	-	1104.0	-	-
	w/o tuning	136.5	40.6	506.7	-	1018.5	-	-
	w/ tuning	142.4$_{\downarrow 3.2}$	41.7$_{\downarrow 1.1}$	404.6	107.5	855.1	30.5	86.7
CrossGET(Ours)	w/o tuning	145.9	**43.1**	785.4	-	1310.5	-	-
	w/o tuning	144.6	**42.6**	692.7	-	1204.5	-	-
	w/o tuning	144.2	**42.7**	602.5	-	1114.3	-	-
	w/o tuning	138.6	**41.2**	514.9	-	1053.3	-	-
	w/ tuning	**143.1**$_{\downarrow 2.5}$	**41.9**$_{\downarrow 0.9}$	413.9$_{\downarrow 52\%}$	104.5$_{\uparrow 94\%}$	822.0$_{\downarrow 40\%}$	30.5$_{\uparrow 37\%}$	84.1$_{\uparrow 60\%}$

Experimental results on BLIP2-OPT2.7B (Li et al., 2023c) are listed in Table 25, which demonstrates similarly promising performance of *CrossGET* as on BLIP2-OPT6.7B. Note that the performance of the original model we tested locally is slightly lower than the results reported in the original paper.

A.16. Experiments with BLIP2 about Where to Reduce Tokens

We conduct experiments on directly ensembling tokens on OPT. To elaborate, directly ensembling tokens on OPT leads to fewer tokens stored in the KV cache. Therefore, during each generation step, a smaller number of previous tokens' KV cache will attend to the current token and thus need less computational cost for Self-Attentions.

An intriguing finding from Table 26 is that the performance of OPT is positively affected by directly ensembling tokens on

Table 26: Accelerate multimodal LLM BLIP2-OPT on the COCO Caption dataset of the Image Caption task.

Method	Where to reduce tokens	CIDEr	GFLOPs
BLIP2-OPT2.7B	/	145.6	854.2
BLIP2-OPT2.7B with CrossGET	On ViT and Q-Former	143.1	413.9
BLIP2-OPT2.7B with CrossGET	On ViT and LLM	$142.4_{\downarrow 0.7}$	417.7
BLIP2-OPT6.7B	/	144.5	1042.6
BLIP2-OPT6.7B with CrossGET	On ViT and Q-Former	143.1	558.2
BLIP2-OPT6.7B with CrossGET	On ViT and LLM	$143.5_{\uparrow 0.4}$	566.1

the relatively larger OPT6.7B model while negatively affected by the same setting on the relatively smaller OPT2.7B model. A possible explanation for the contrasting behaviors is:

- To accelerate OPT, the smaller 2.7B model is more vulnerable to the disturbance brought by the token ensemble within the model (note that the OPT model is frozen, so it cannot adapt its weights of parameters when the number of tokens is getting smaller). Therefore, applying CrossGET on Q-Former is a better setting so that the OPT model is accelerated by taking fewer input tokens.
- The larger 6.7B model is more resilient to the disturbance brought by the token ensemble within the model. Moreover, ensembling tokens within the OPT model can help preserve the tokens' information as much as possible (if the number of tokens is reduced in the preceding Q-Former, the lost information due to the ensemble operation will be inaccessible to the succeeding OPT model).

Moreover, the experiments also indicate that after ensembling tokens, at least two competing factors determine the extent of the performance affected: 1) performance increases due to preserving more tokens' information within the OPT model. 2) performance decreases depending on frozen OPT's resilience to the disturbance brought by token ensemble within the model.

A.17. Evaluation at Different Reduction Ratios without Training

Exhaustive experimental results at different reduction ratios ***without*** training are listed in Table 27.

A.18. Evaluation at Different Reduction Ratios with Training

More experimental results at different reduction ratios with training are listed in Table 28.

A.19. Re-Evaluation Trained Model at Different Reduction Ratios

Once *CrossGET* has trained a model at a certain compression ratio, a series of models with different performance and computational costs are obtained simultaneously. More specifically, by simply adjusting the number of tokens reduced at inference, it is free to use different models without training based on the desired budget. Table 29 provides the relevant experimental results for CLIP model on the Flickr30K dataset of the Image-Text Retrieval task.

Table 27: Experimental results at different reduction ratios for BLIP on the NVLR2 dataset of the Visual Reasoning task ***without*** training.

Approach	Test Acc	Drop	GFLOPs	Reduction
BLIP (Li et al., 2022)	83.38	-	132.54	-
CrossGET (***without*** training)	83.34	-0.04	136.92	0.97x
	83.32	-0.06	135.18	0.98x
	83.40	+0.02	133.43	0.99x
	83.22	-0.16	131.69	1.01x
	83.17	-0.21	129.96	1.02x
	83.02	-0.36	128.23	1.03x
	83.08	-0.30	126.50	1.05x
	82.99	-0.39	124.78	1.06x
	82.88	-0.50	123.07	1.08x
	82.81	-0.57	121.36	1.09x
	82.85	-0.53	119.65	1.11x
	82.66	-0.72	117.95	1.12x
	82.48	-0.90	116.25	1.14x
	82.20	-1.18	114.56	1.16x
	82.30	-1.08	112.87	1.17x
	82.02	-1.36	111.19	1.19x
	81.67	-1.71	109.51	1.21x
	81.90	-1.48	107.84	1.23x
	81.75	-1.63	106.17	1.25x
	81.63	-1.75	104.51	1.27x
	81.47	-1.91	102.84	1.29x
	81.43	-1.95	101.19	1.31x
	81.29	-2.09	99.54	1.33x
	80.93	-2.45	97.89	1.35x
	80.87	-2.51	96.25	1.38x
	80.93	-2.45	94.61	1.40x
	80.86	-2.52	92.98	1.43x
	80.68	-2.70	91.35	1.45x
	80.55	-2.83	89.73	1.48x
	80.28	-3.10	88.12	1.50x
	80.35	-3.03	86.50	1.53x
	80.22	-3.16	84.89	1.56x
	80.22	-3.16	83.29	1.59x
	80.38	-3.00	81.69	1.62x
	80.17	-3.21	80.10	1.65x
	80.17	-3.21	78.51	1.69x
	80.30	-3.08	76.92	1.72x
	80.12	-3.26	75.34	1.76x
	80.21	-3.17	73.76	1.80x
	80.02	-3.36	72.19	1.84x
	79.79	-3.59	70.63	1.88x
	79.64	-3.74	69.07	1.92x
	80.02	-3.36	67.51	1.96x
	79.87	-3.51	65.96	2.01x
	79.68	-3.70	64.60	2.05x
	79.32	-4.06	63.33	2.09x
	79.10	-4.28	62.03	2.14x
	78.80	-4.58	60.78	2.18x
	78.85	-4.53	59.73	2.22x
	78.85	-4.53	58.65	2.26x

Table 28: Experimental results at different reduction ratios for BLIP on the NVLR2 dataset of the Visual Reasoning task with training.

Approach	Test Acc	Drop	GFLOPs	Reduction
BLIP (Li et al., 2022)	83.38	-	132.54	-
CrossGET (with training)	83.74	+0.36	118.34	1.12x
	83.31	-0.07	85.27	1.55x
	83.19	-0.19	61.09	2.17x
	82.28	-1.10	58.95	2.25x
	81.33	-2.05	53.25	2.49x
	80.67	-2.71	50.14	2.64x
	78.19	-5.19	45.11	2.94x

Table 29: Experimental results for re-evaluating a model trained by CrossGET (50% tokens reduced) at different reduction ratios ***without*** training.

Approach	Recall@1 - Trained	Change	GFLOPs	Increase
CLIP (Radford et al., 2021)	85.70	-0.15	20.57	1.71x
CrossGET (re-evaluate ***without*** training)	85.86	+0.01	20.70	1.72x
	85.79	-0.06	20.48	1.70x
	85.93	+0.08	19.90	1.65x
	85.88	+0.03	19.32	1.60x
	86.06	+0.21	18.74	1.56x
	86.19	+0.34	18.17	1.51x
	86.34	+0.49	17.60	1.46x
	86.15	+0.30	17.03	1.41x
	86.21	+0.36	16.46	1.37x
	86.33	+0.48	15.90	1.32x
	86.19	+0.34	15.33	1.27x
	86.29	+0.44	14.77	1.23x
	86.06	+0.21	14.22	1.18x
	86.01	+0.16	13.66	1.13x
	86.08	+0.23	13.11	1.09x
	86.20	+0.35	12.56	1.04x
CrossGET (with training)	85.85	-	12.04	-

B. Supplementary Related Works

Token Reduction Prior works have advanced token reduction in unimodal scenarios, such as for vision (Chen et al., 2021; Rao et al., 2021; Su et al., 2022; Chavan et al., 2022; Liang et al., 2022b; Yin et al., 2022; Liang et al., 2022a; Bolya et al., 2023) or language (Goyal et al., 2020; Kim & Cho, 2020; Kim et al., 2022; Lassance et al., 2021). *CrossGET* emerges as one of the pioneering efforts in token ensemble frameworks for multimodal scenarios. Additionally, It is one of the few approaches requiring no extra learnable parameters aside from negligible cross tokens. Although ToMe (Bolya et al., 2023) also does not require learnable parameters, it is limited to unimodal scenarios. For the convenience of parallelizability, it adopts a bipartite matching method that delivers relatively unreliable token-matching results.

Multimodal Transformer Acceleration A few studies have tried to accelerate multimodal Transformers. Gan et al. (2022) investigates unstructured pruning, discovering that winning tickets (Frankle & Carbin, 2018) also exist in multimodal Transformers. By structured pruning, UPop (Shi et al., 2023) proposes that small vision-language models can be unifiedly searched within large ones and then progressively pruned. DistillVLM (Fang et al., 2021) and EfficientVLM (Wang et al., 2022b) suggest knowledge distillation to mimic the distribution of large vision-language models. MiniVLM (Wang et al., 2020a) employs lightweight networks for its construction. AWQ (Lin et al., 2023) applies weight-only quantization on multimodal Transformers. *CrossGET* achieves acceleration through ensembling tokens, which is orthogonal to these existing strategies by shrinking model parameters. TRIPS (Jiang et al., 2022) utilizes text information for unidirectional guidance in reducing image patches and is limited to modality-independent models. In contrast, *CrossGET* enables bidirectional learning of guidance information between modalities and applies to both modality-independent and modality-dependent models. Appendix A.13 provides more comparisons and analyses on TRIPS.

Parameter-Efficient Fine-Tuning Parameter-efficient fine-tuning aims to reduce the number of learnable parameters during fine-tuning. It primarily encompasses adapters (Houlsby et al., 2019; Sung et al., 2022b), prompt tuning (Li & Liang, 2021; Khattak et al., 2022), low-rank adaptation (Hu et al., 2021; Hyeon-Woo et al., 2021), parameter sharing (Lan et al., 2019; Shi et al., 2021), dropout (Fan et al., 2019; Shi et al., 2022) and their combinations (He et al., 2021; Karimi Mahabadi et al., 2021). LST (Sung et al., 2022a) suggests a side tuning for enhanced memory efficiency. In multimodal scenarios, LiteVL (Chen et al., 2022a) proposes to inherit image-text pre-trained weights with some slight modifications to quickly adapt to video-text tasks without heavy pre-training, thereby reducing the training cost. While parameter-efficient fine-tuning enhances efficiency in the fine-tuning phase, it does not accelerate model inference. Conversely, *CrossGET* mainly focuses on improving efficiency during inference, and accordingly, the model inference can be significantly accelerated.

C. Supplementary Methodology Details

C.1. Examples for Demonstrating Token Matching

Optimal Objective Function and Solution For example, when the number of tokens in total is $N = 4$, and the number of tokens to be reduced is $r = 2$, by verifying and comparing all possible token-matching results, the optimal objective function for case1 in Figure 3 can be obtained:

$$S_1^* = \mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_4) + \mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_3) = 0.5 + 0.6 = 1.1, \tag{11}$$

and the corresponding optimal solution for token matching can be determined as

$$\boldsymbol{P}_1^* = \{(\boldsymbol{T}_1, \boldsymbol{T}_4), (\boldsymbol{T}_2, \boldsymbol{T}_3)\}. \tag{12}$$

Similarly, the optimal objective function for case2 (inverted case1) in Figure 3 is

$$S_2^* = \mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_3) + \mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_4) = 0.9 + 0.8 = 1.7 \tag{13}$$

and the corresponding optimal solution for token matching is

$$\boldsymbol{P}_2^* = \{(\boldsymbol{T}_1, \boldsymbol{T}_3), (\boldsymbol{T}_2, \boldsymbol{T}_4)\}. \tag{14}$$

Revisiting Bipartite Soft Matching ToMe (Bolya et al., 2023) suggests a non-iterative *bipartite soft matching* ensure parallelizability, which divides tokens into two disjoint sets alternately, for each token in the first set calculates the maximum similarity from it to each token in the other set, and the token pairs with the highest similarities will be merged.

Take case1 in Figure 3 as an example, tokens are firstly divided into $\{\boldsymbol{T}_1, \boldsymbol{T}_3\}$ and $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$, then the maximum similarity from $\boldsymbol{T}_1$ to $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_4) = 0.5$ and from $\boldsymbol{T}_3$ to $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_3, \boldsymbol{T}_2) = 0.6$. Therefore, the optimal objective function in Eq.11 and optimal solution in Eq.12 are achieved:

$$S_1^B = \mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_4) + \mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_3) = S_1^*, \quad \boldsymbol{P}_1^B = \{(\boldsymbol{T}_1, \boldsymbol{T}_4), (\boldsymbol{T}_2, \boldsymbol{T}_3)\} = \boldsymbol{P}_1^* \tag{15}$$

However, for case2 (inverted case1), *bipartite soft matching* leads to a worse solution: tokens are firstly divided into $\{\boldsymbol{T}_1, \boldsymbol{T}_3\}$ and $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$, then the maximum similarity from $\boldsymbol{T}_1$ to $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_2) = 0.6$ and from $\boldsymbol{T}_3$ to $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_3, \boldsymbol{T}_4) = 0.7$. Therefore, the optimal objective function in Eq.13 and optimal solution in Eq.14 are not achieved:

$$S_2^B = \mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_2) + \mathcal{D}(\boldsymbol{T}_3, \boldsymbol{T}_4) < S_2^*, \quad \boldsymbol{P}_2^B = \{(\boldsymbol{T}_1, \boldsymbol{T}_2), (\boldsymbol{T}_3, \boldsymbol{T}_4)\} \neq \boldsymbol{P}_2^* \tag{16}$$

This is attributed to the design of *bipartite soft matching* that for the convenience of ensuring parallelizability, each token only takes into account the similarity with half but not all other tokens, and the method degrades when tokens with high similarity are not divided into different sets.

Shifting to Complete-Graph Soft Matching An approximate algorithm *complete-graph soft matching* is proposed to tackle the above challenge. It enables each token to take into account the similarity with all other tokens while avoiding introducing iterative and non-parallelizable operations.

Take case2 in Figure 3 as an example, all tokens $\boldsymbol{T} = \{\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3, \boldsymbol{T}_4\}$ are sorted in descending order according to their maximum similarity to other tokens: $\boldsymbol{T}' = \{\boldsymbol{T}_1, \boldsymbol{T}_3, \boldsymbol{T}_2, \boldsymbol{T}_4\}$. After adding the dependency mask, the maximum similarity from top priority source token candidate $\boldsymbol{T}_1$ to its destination tokens $\{\boldsymbol{T}_3, \boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_3) = 0.9$, from second priority source token candidate $\boldsymbol{T}_3$ to its destination tokens $\{\boldsymbol{T}_2, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_3, \boldsymbol{T}_4) = 0.7$, and from third priority source token candidate $\boldsymbol{T}_2$ to its destination token $\boldsymbol{T}_4$ is $\mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_4) = 0.8$. The source token candidates among them corresponding to the two largest similarities are selected as the source token set $\boldsymbol{T}_s = \{\boldsymbol{T}_1, \boldsymbol{T}_2\}$ while remaining tokens form the destination token set $\boldsymbol{T}_D = \{\boldsymbol{T}_3, \boldsymbol{T}_4\}$. Then the maximum similarity from $\boldsymbol{T}_1$ to $\{\boldsymbol{T}_3, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_3) = 0.9$ and from $\boldsymbol{T}_2$ to $\{\boldsymbol{T}_3, \boldsymbol{T}_4\}$ is $\mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_4) = 0.8$. Therefore, the optimal objective function in Eq.13 and optimal solution in Eq.14 are achieved:

$$S_2^C = \mathcal{D}(\boldsymbol{T}_1, \boldsymbol{T}_3) + \mathcal{D}(\boldsymbol{T}_2, \boldsymbol{T}_4) = S_2^*, \quad \boldsymbol{P}_2^C = \{(\boldsymbol{T}_1, \boldsymbol{T}_3), (\boldsymbol{T}_2, \boldsymbol{T}_4)\} = \boldsymbol{P}_2^*. \tag{17}$$

Similarly, it can also be verified that *complete-graph soft matching* achieves the optimal objective function in Eq.11 and optimal solution in Eq.12 for case1 in Figure 3.

C.2. Algorithm Implementation of Complete-Graph Soft Matching

Algorithm 1 Complete-Graph Soft Matching

Input: Number of tokens N, number of tokens to be reduced r, original tokens $\boldsymbol{T} = \{\boldsymbol{T}_i\}_{i=1}^{N}$ and their corresponding keys $\boldsymbol{K} = \{\boldsymbol{K}_i\}_{i=1}^{N}$ where $|\boldsymbol{T}| = |\boldsymbol{K}| = N$

Output: Reduced tokens $\boldsymbol{T}^\star = \{\boldsymbol{T}_i^\star\}_{i=1}^{N-r}$ where $|\boldsymbol{T}^\star| = N - r$

1 # Step1: Calculate the cosine distance $\boldsymbol{D}_{ij}$ between the keys of tokens

2 $\boldsymbol{D} = \frac{\boldsymbol{K}\boldsymbol{K}^\top}{\|\boldsymbol{K}\|_2^2} + \mathrm{diag}(\underbrace{-\infty, -\infty, \cdots, -\infty}_{N}), \quad \boldsymbol{D} \in \mathbb{R}^{N\times N}, \ \mathrm{diag}: \mathbb{R}^N \to \mathbb{R}^{N\times N}$

3 # Step2: Descendingly sort similarity matrix $\boldsymbol{D}$ by maximum similarity

4 $\boldsymbol{A}^S = \mathrm{argsort}(\max_{1\le j\le N} \boldsymbol{D}_{ij}) \in \mathbb{R}^N, \quad \boldsymbol{A}^D = \mathrm{argsort}(\max_{1\le i\le N} \boldsymbol{D}_{ij}) \in \mathbb{R}^N$

5 $\boldsymbol{D}^\star = \mathrm{sort_d}(\mathrm{sort_s}(\boldsymbol{D}, \boldsymbol{A}^S), \boldsymbol{A}^D), \ \mathrm{sort_s}: \boldsymbol{D}_{ij}^\star \leftarrow \boldsymbol{D}_{\boldsymbol{A}_i^S j}, \ \mathrm{sort_d}: \boldsymbol{D}_{ij}^\star \leftarrow \boldsymbol{D}_{i\boldsymbol{A}_j^D}$

6 # Step3: Add a lower triangle dependency mask $\boldsymbol{M}$

7 $\boldsymbol{D}^\star = \boldsymbol{D}^\star + \boldsymbol{M}, \quad \boldsymbol{M}_{ij} = \begin{cases} -\infty & \text{for } i \ge j \\ 0 & \text{for } i < j \end{cases}$

8 # Step4: Pick source tokens $\boldsymbol{T}^S$ and destination tokens $\boldsymbol{T}^D$ by similarity

9 $\boldsymbol{A} = \mathrm{argsort}(\max_{1\le j\le N} \boldsymbol{D}_{ij}^\star) \in \mathbb{R}^N, \quad \boldsymbol{A}^S = (\boldsymbol{A}_i)_{1\le i\le r} \in \mathbb{R}^r, \quad \boldsymbol{T}^S = \{\boldsymbol{T}_i | i \in \boldsymbol{A}^S\}$

10 $\boldsymbol{A} = \underset{j\in(\{k\}_{k=1}^N \setminus \boldsymbol{A}^S)}{\mathrm{argmax}} \boldsymbol{D}_{ij}^\star \in \mathbb{R}^N, \quad \boldsymbol{A}^D = (\boldsymbol{A}_i)_{i\in\boldsymbol{A}^S} \in \mathbb{R}^r, \quad \boldsymbol{T}^D = \{\boldsymbol{T}_i | i \in \boldsymbol{A}^D\}$

11 # Step5: Average source and corresponding destination tokens

12 **return** $\boldsymbol{T}^\star = [\boldsymbol{T} \setminus (\boldsymbol{T}^S \cup \boldsymbol{T}^D)] \cup \{\frac{1}{2}(\boldsymbol{T}_i^S + \boldsymbol{T}_i^D)\}_{i=1}^r$

Algorithm 1 is the detailed implementation of the proposed *complete-graph soft matching*. The Step1 $\sim$ 5 in the comments correspond to the Step1 $\sim$ 5 described in Section 3.2 of the main text. Regarding parallelizability, there are no sequential loops in the algorithm procedure. Therefore, data can be processed in parallel within each step by parallelizable operations (such as *bmm*, *matmul*, *scatter* and *gather* in Pytorch (Paszke et al., 2019)).

C.3. Algorithm Implementation of Cross-Guided Matching and Ensemble

Algorithm 2 Cross-Guided Matching and Ensemble (improvements upon Algorithm 1)

Input: Same inputs as Algorithm 1, plus query of the cross token $\boldsymbol{Q}$

Output: Same as Algorithm 1

1 # Step1$\sim$3: Same as Algorithm 1

2 # Step4: Pick tokens $\boldsymbol{T}^S$ and $\boldsymbol{T}^D$ by similarity and importance

3 $\boldsymbol{I} = \frac{\boldsymbol{K}\boldsymbol{Q}^\top}{\|\boldsymbol{K}\|_2\|\boldsymbol{Q}\|_2} \in \mathbb{R}^N$

4 $\boldsymbol{A} = \mathrm{argsort}(\max_{1\le j\le N} \boldsymbol{D}_{ij}^\star - \boldsymbol{I}) \in \mathbb{R}^N, \quad \boldsymbol{A}^S = (\boldsymbol{A}_i)_{1\le i\le r} \in \mathbb{R}^r, \quad \boldsymbol{T}^S = \{\boldsymbol{T}_i | i \in \boldsymbol{A}^S\}$

5 $\boldsymbol{A} = \underset{j\in(\{k\}_{k=1}^N \setminus \boldsymbol{A}^S)}{\mathrm{argmax}} \boldsymbol{D}_{ij}^\star \in \mathbb{R}^N, \quad \boldsymbol{A}^D = (\boldsymbol{A}_i)_{i\in\boldsymbol{A}^S} \in \mathbb{R}^r, \quad \boldsymbol{T}^D = \{\boldsymbol{T}_i | i \in \boldsymbol{A}^D\}$

6 # Step5: Sum weighted source and corresponding destination tokens

7 $\boldsymbol{P} = \{(\boldsymbol{T}_i^S, \boldsymbol{T}_i^D)\}_{i=1}^r, \quad \boldsymbol{W} = \{\mathrm{softmax}(\boldsymbol{I}_i, \boldsymbol{I}_j) | (\boldsymbol{T}_i, \boldsymbol{T}_j) \in \boldsymbol{P}\}$

8 **return** $\boldsymbol{T}^\star = [\boldsymbol{T} \setminus (\boldsymbol{T}^S \cup \boldsymbol{T}^D)] \cup \{\sum_{j=1}^{|\boldsymbol{W}_i|} \boldsymbol{W}_{ij}\boldsymbol{P}_{ij}\}_{i=1}^r$

Algorithm 2 demonstrates how to improve *complete-graph soft matching* by adding *cross-guided matching and ensemble* upon it. It is worth noting that line7 $\sim$ 8 in Algorithm 2 does not imply that only two tokens are in each stack of tokens to be ensembled. This is because different source tokens in $\boldsymbol{T}^S$ may have the same destination token in $\boldsymbol{T}^D$, which implies that the size of the stack is allowed to be larger than two (in this case, the procedure of ensembling stacks with the different number of tokens can still be implemented by parallelizable operations such as *scatter_add* in Pytorch).

C.4. Sub-optimal Cases for Complete-Graph Soft Matching

Section 3.2 has already shown the cases that *complete-graph soft matching* achieves optimal matching, and here we provide more analyses on the sub-optimal cases of *complete-graph soft matching*.

The main sub-optimal cases come from the trade-off between parallelizability and matching accuracy. To achieve parallelizability, the set of source token $\boldsymbol{T}^S$ and destination tokens $\boldsymbol{T}^D$ have to be disjoint:

$$\boldsymbol{T}^S \cap \boldsymbol{T}^D = \phi. \tag{18}$$

Otherwise, consider

$$\boldsymbol{T}^S \cap \boldsymbol{T}^D = \{\boldsymbol{T}_x\} \neq \phi, \quad 1 \leq x \leq N \tag{19}$$

where N is the number of the original tokens, then

$$(\exists \boldsymbol{T}_i \in \boldsymbol{T}^S \ s.t. \ (\boldsymbol{T}_i, \boldsymbol{T}_x) \in \boldsymbol{P}) \wedge (\exists \boldsymbol{T}_j \in \boldsymbol{T}^D \ s.t. \ (\boldsymbol{T}_x, \boldsymbol{T}_j) \in \boldsymbol{P}) \tag{20}$$

where $\boldsymbol{P} = \{(\boldsymbol{T}_i^S, \boldsymbol{T}_i^D)\}_{i=1}^r$ is the set of the paired tokens to be ensembled, is true. However, there is a computational dependency between merging $\boldsymbol{T}_i$ into $\boldsymbol{T}_x$ and merging $\boldsymbol{T}_x$ into $\boldsymbol{T}_j$. The two operations of the merging require iterations and therefore cannot be parallelized.

$\boldsymbol{T}^S$ and $\boldsymbol{T}^D$ are disjoint (*i.e.*, Eq.18 holds) is equivalent to the constraint

$$\forall \boldsymbol{T}_i \in \boldsymbol{T}^S, \boldsymbol{T}_i \notin \boldsymbol{T}^D \tag{21}$$

is satisfied. In the Step1 of the Algorithm 1, computation is conducted on a complete graph. Therefore $\boldsymbol{T}^S$ and $\boldsymbol{T}^D$ are joint, and constraint 21 does not been satisfied. In Step3, the added lower triangle dependency mask ensures that source tokens with higher priority (*i.e.*, whose keys have higher maximum cosine similarity to keys of other tokens) will not become targets for other source tokens with lower priority, *i.e.*, a relaxed constraint

$$\forall \boldsymbol{T}_i \in \boldsymbol{T}^S, \boldsymbol{T}_i \notin (\boldsymbol{T}_j^D)_{i \leq j \leq N} \tag{22}$$

is satisfied. However, the unsatisfied part of the constraint 21

$$\forall \boldsymbol{T}_i \in \boldsymbol{T}^S, \boldsymbol{T}_i \notin (\boldsymbol{T}_j^D)_{1 \leq j < i} \tag{23}$$

indicates that source tokens with low priority may still become targets for other source tokens with high priority. To further satisfy constraint 23, the line10 of Step4 in Algorithm 1 explicitly removes all elements of the source token set from the set of all tokens to construct the set of the destination tokens. And the sub-optimal cases for *complete-graph soft matching* arise when

$$\underset{j \in (\{k\}_{k=1}^N \setminus \boldsymbol{A}^S)}{\operatorname{argmax}} \boldsymbol{D}_{ij}^\star \neq \underset{j \in \{k\}_{k=1}^N}{\operatorname{argmax}} \boldsymbol{D}_{ij}^\star, \tag{24}$$

which indicates a source token may exist whose closest destination token in $\boldsymbol{T}^D$ happens to be another source token in $\boldsymbol{T}^S$. For parallelizability, this destination token is removed from $\boldsymbol{T}^D$, resulting in the source token can only match the second closest destination token in the set of reduced $\boldsymbol{T}^D$.

C.5. Expectation of Optimal Matching Probability and Complexity Analysis

Expectation of Optimal Matching Probability For a token $\boldsymbol{T}_i \in \boldsymbol{T}$, assume that any other token $\boldsymbol{T}_j \in \boldsymbol{T} \backslash \{\boldsymbol{T}_i\}$ has the same probability of being its optimal destination token, *i.e.*

$$\forall 1 \leq j \leq N, \quad p((\boldsymbol{K}_i, \boldsymbol{K}_j) = \underset{\substack{1 \leq k \leq N \\ k \neq i}}{\operatorname{argmax}} \ s(\boldsymbol{K}_i, \boldsymbol{K}_k)) = \frac{1}{N-1} \tag{25}$$

where $s(x, y)$ is a function that calculates cosine similarity between x and y.

For *complete-graph soft matching*, in layer l ($1 \leq l \leq L$), supoose $\boldsymbol{X} \sim p(x)$ is a discrete random variable about whether a token from $\boldsymbol{T}^S$ ($|\boldsymbol{T}^S| = r$) can find its optimal destination token in $\boldsymbol{T}^D$ ($|\boldsymbol{T}^D| = N - lr$), and the probability distribution

$p(x)$ is:

$$p(\boldsymbol{X} = \text{can}) = \sum_{1}^{|\boldsymbol{T}^D|} p((\boldsymbol{K}_i, \boldsymbol{K}_j) = \underset{\substack{1 \le k \le N+(1-l)r \\ k \ne i}}{\operatorname{argmax}} s(\boldsymbol{K}_i, \boldsymbol{K}_k)) \tag{26}$$

$$= \frac{N - lr}{N + (1-l)r - 1}. \tag{27}$$

$$p(\boldsymbol{X} = \text{not}) = 1 - p(\boldsymbol{X} = \text{can}). \tag{28}$$

Denote $\boldsymbol{L} \sim p(l)$ as a discrete random variable ($\boldsymbol{L} \perp\!\!\!\perp \boldsymbol{X}$) about the current layer number, and

$$\forall 1 \le l \le L, \quad p(\boldsymbol{L} = l) = \frac{1}{L} \tag{29}$$

Denote $h(\boldsymbol{X}, \boldsymbol{L})$ as a indicator function

$$h(\boldsymbol{X}, \boldsymbol{L}) = \begin{cases} 1 & \text{for } \boldsymbol{X} = \text{can} \\ 0 & \text{for } \boldsymbol{X} = \text{not} \end{cases} \tag{30}$$

Then the expectation of a token from $\boldsymbol{T}^S$ can find its optimal destination token in $\boldsymbol{T}^D$ is

$$\mathbb{E}^C = \mathbb{E}_{XL}\left[h(\boldsymbol{X}, \boldsymbol{L})\right] = \sum_{l \in \boldsymbol{L}} \sum_{x \in \boldsymbol{X}} h(x, l) p_{XL}(x, l) \tag{31}$$

$$= \sum_{l \in \boldsymbol{L}} \sum_{x \in \boldsymbol{X}} h(x, l) p_{\boldsymbol{X}}(x) p_{\boldsymbol{L}}(l) \tag{32}$$

$$= \sum_{l=1}^{L} [1 \cdot p(\boldsymbol{X} = \text{can}) + 0 \cdot p(\boldsymbol{X} = \text{not})] \frac{1}{L} \tag{33}$$

$$= \frac{1}{L} \sum_{l=1}^{L} \frac{N - lr}{N + (1-l)r - 1} \tag{34}$$

Similarly, given by *bipartite soft matching* used in ToMe (Bolya et al., 2023), the expectation of a token from $\boldsymbol{T}^S$ ($|\boldsymbol{T}^S| = \lceil \frac{N+(1-l)r}{2} \rceil$) can find its optimal destination token in $\boldsymbol{T}^D$ ($|\boldsymbol{T}^D| = \lfloor \frac{N+(1-l)r}{2} \rfloor$) is

$$\mathbb{E}^B = \frac{1}{L} \sum_{l=1}^{L} \frac{1}{N + (1-l)r - 1} \lfloor \frac{N + (1-l)r}{2} \rfloor \tag{35}$$

Compare $\mathbb{E}^C$ given by *complete-graph soft matching* with $\mathbb{E}^B$ give by *bipartite soft matching*:

$$\mathbb{E}^C - \mathbb{E}^B = \frac{1}{L} \sum_{l=1}^{L} \frac{1}{N + (1-l)r - 1} (N - lr - \lfloor \frac{N + (1-l)r}{2} \rfloor) \tag{36}$$

$$\ge \frac{1}{L} \sum_{l=1}^{L} \frac{1}{N + (1-l)r - 1} (N - lr - \frac{N + (1-l)r}{2}) \tag{37}$$

$$= \frac{1}{L} [\underbrace{\sum_{l=1}^{L-1} \frac{1}{N + (1-l)r - 1} \frac{N - lr - r}{2}}_{Part1:\ 1 \le l \le L-1} + \underbrace{\frac{1}{N + (1-L)r - 1} \frac{N - Lr - r}{2}}_{Part2:\ l = L}] \tag{38}$$

For part1 in Eq.38, since the number of remaining tokens is always a positive integer, we have

$$N - Lr \ge 1, \tag{39}$$

and therefore for $1 \leq l \leq L-1$:

$$N - lr \geq r + 1 \Leftrightarrow N - lr - r \geq 1 > 0 \tag{40}$$

always holds. Morever

$$N + (1-l)r - 1 = (N - lr - 1) + r > 0 \tag{41}$$

always holds. Furthermore, we have part1 > 0 always holds, which indicates the expectation given by *complete-graph soft matching* is higher than *bipartite soft matching* except for the last layer.

For part2 in Eq.38, *bipartite soft matching* evenly divides tokens into two disjoint sets, and the size of each set is not less than r. However, the remaining tokens before the last layer may be less than $2r$. In such a situation, *bipartite soft matching* have to reduce the r to the $r^\star$ such that

$$N - Lr^\star = r^\star \tag{42}$$

complete-graph soft matching follows the same design, and therefore part2 $= 0$ holds. Overall, we have

$$\mathbb{E}^C - \mathbb{E}^B > 0 \tag{43}$$

always holds. For example, for a CLIP (Radford et al., 2021) model with

$$N = 197,\ L = 12,\ r = 16 \tag{44}$$

used in our experiments, given by *complete-graph soft matching*, the expectation of optimal matching probability for a token from $\boldsymbol{T}^S$ is

$$\mathbb{E}^C = \frac{1}{12}\sum_{l=1}^{12} \frac{197 - l \times 16}{197 + (1-l) \times 16 - 1} \approx 0.78, \tag{45}$$

while given by *bipartite soft matching*, the corresponding expectation is

$$\mathbb{E}^B = \frac{1}{12}\sum_{l=1}^{12} \frac{1}{197 + (1-l) \times 16 - 1} \lfloor \frac{197 + (1-l) \times 16}{2} \rfloor = 0.50 < \mathbb{E}^C \tag{46}$$

C.6. Complexity Analysis for Complete-Graph Soft Matching

Since the *sort* and *argsort* operations in Algorithm 1 and 2 can be solved by algorithms with $\mathcal{O}(N \log N)$ complexity such as *QuickSort* (Hoare, 1962), the major complexity $\mathcal{O}(N^2)$ comes from the computation of cosine similarities between each pair of tokens.

A comparison of different matching methods is listed in Table 30, which demonstrates that as a parallelizable method, *CrossGET* can achieve relatively high expectation of optimal matching probability for a certain token from $\boldsymbol{T}^S$ with relatively low complexity.

Table 30: **A comparison of different matching methods.** Denote N as the number of the original tokens, r as the number of tokens to be reduced, and T as the number of iterations for k-means.

Method	Iterative	Parallelizable	Expectation of Optimal Matching Probability	Complexity
Greedy Search	Yes	✗	$\{1\}$	$\mathcal{O}(rN^2)$
K-Means	Yes	✗	$[1-\epsilon, 1],\ \lim_{T\to\infty}\epsilon = 0$	$\mathcal{O}(rNT)$
Random	No	✓	$\frac{1}{N-1} \in [0, 0+\epsilon],\ \lim_{N\to\infty}\epsilon = 0$	$\mathcal{O}(r)$
ToMe (Bolya et al., 2023)	No	✓	$\frac{1}{L}\sum_{l=1}^{L} \frac{1}{N+(1-l)r-1} \lfloor \frac{N+(1-l)r}{2} \rfloor \in [\frac{1}{2}, \frac{1}{2}+\epsilon],\ \lim_{N\to\infty}\epsilon = 0$	$\mathcal{O}(N^2)$
CrossGET (Ours)	No	✓	$\frac{1}{L}\sum_{l=1}^{L} \frac{N-lr}{N+(1-l)r-1} \in [1-\epsilon, 1],\ \lim_{N\to\infty}\epsilon = 0$	$\mathcal{O}(N^2)$