No Head Left Behind - Multi-Head Alignment Distillation for Transformers

Tianyang Zhao[1,2*], **Kunwar Yashraj Singh**[1 †], **Srikar Appalaraju**[1 ‡], **Peng Tang**[1], **Vijay Mahadevan**[1], **R. Manmatha**[1], **Ying Nian Wu**[1,2]

[1]AWS AI Labs, [2]University of California, Los Angeles

tyzhao@ucla.edu, {sinkunwa, srikara, tangpen, vmahad, manmatha, wunyin}@amazon.com

Abstract

Knowledge distillation aims at reducing model size without compromising much performance. Recent work has applied it to large vision-language (VL) Transformers, and has shown that attention maps in the multi-head attention modules of vision-language Transformers contain extensive intra-modal and cross-modal co-reference relations to be distilled. The standard approach is to apply a one-to-one attention map distillation loss, i.e. the Teacher's first attention head instructs the Student's first head, the second teaches the second, and so forth, but this only works when the numbers of attention heads in the Teacher and Student are the same. To remove this constraint, we propose a new Attention Map Alignment Distillation (AMAD) method for Transformers with multi-head attention, which works for a Teacher and a Student with different numbers of attention heads. Specifically, we soft-align different heads in Teacher and Student attention maps using a cosine similarity weighting. The Teacher head contributes more to the Student heads for which it has a higher similarity weight. Each Teacher head contributes to all the Student heads by minimizing the divergence between the attention activation distributions for the soft-aligned heads. No head is left behind. This distillation approach operates like cross-attention. We experiment on distilling VL-T5 and BLIP, and apply AMAD loss on their T5, BERT, and ViT sub-modules. We show, under vision-language setting, that AMAD outperforms conventional distillation methods on VQA-2.0, COCO Captioning, and Multi30K translation datasets. We further show that even without VL pre-training, the distilled VL-T5 models outperform corresponding VL pre-trained VL-T5 models that are further fine-tuned by ground-truth signals, and that fine-tuning distillation can also compensate to some degree for the absence of VL pre-training for BLIP models.

Introduction

Recently, large pre-trained Transformers-based (Vaswani et al. 2017) models, such as BERT (Devlin et al. 2019), T5 (Raffel et al. 2020), and GPT (Radford et al. 2018), have shown great capabilities for language modeling. Researchers have further extended these language Transformers to multi-modal Transformers for visual-linguistic tasks, e.g. VL-BERT (Su et al. 2019), VL-T5 (Cho et al. 2021), Oscar (Li et al. 2020b), BLIP (Li et al. 2022b, 2023), OFA (Wang et al. 2022a), Flamingo (Alayrac et al. 2022), Florence (Yuan et al. 2021), PALI (Chen et al. 2022c). These large vision-language (VL) models exhibit promising performances on a variety of visual-linguistic tasks, including visual question answering (VQA), image captioning, visual grounding and image-text matching. Increasing model size (BERT-L (340M), OFA (930M), T5 (11B), GPT-3 (175B) (Brown et al. 2020)) leads to better performance, but also increases memory consumption during deployment and leads to large increases in inference latency.

To alleviate this problem, researchers (Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b; Fang et al. 2021; Sanh et al. 2019) have applied knowledge distillation (KD) (Hinton, Vinyals, and Dean 2015) approaches to large Transformers, aiming at compressing these large models into smaller ones without compromising much performance. In general, KD involves a large trained and frozen Teacher network, and a small Student network to be trained. The goal is to distill the knowledge from the larger Teacher into the smaller Student to bridge the performance gap between the two caused by difference of model sizes. In the distillation process, the Student learns to mimic the soft response and the latent representation of the Teacher. Specifically, this may involve minimizing the divergence between the Student's and the Teacher's output classification logits, and the divergence between their intermediate representations.

In case of distilling Transformers, their attention maps are often important and contain intermediate representations to be transferred. (Cao et al. 2020) show that certain attention matrices of the pre-trained vision-language Transformers contain extensive intra- and cross-modal co-reference relations. (Fang et al. 2021) further show that minimizing the divergence between these attention maps of Teacher and Student can boost distillation performance.

However, conventional attention map distillation methods for multi-head attention modules, either for language (Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b; Sanh et al. 2019) or VL (Fang et al. 2021) Transformers, directly minimize the divergence between the attention maps of Teacher and Student for each of their heads in a one-to-one fashion, i.e. the Teacher's first attention head instructs the Student's first head, the second teaches the second, and so forth, as in

[*]Work conducted during an internship at Amazon.

[†]Corresponding Author

[‡]Corresponding Author

the right side of Figure 1. Hence, these methods can only be applied where the Teacher and Student have an equal number of heads, and do not generalize to the more common case where the large Transformer and the small Transformer have different numbers of attention heads; Otherwise if the Teacher has more heads, its extra heads have to be discarded in the distilling process, causing knowledge loss.

This motivates the design of our approach, Attention Map Alignment Distillation (AMAD), to remove the same-number-of-heads restriction. In brief, AMAD soft aligns different heads in the Teacher and Student attention maps using cosine similarity. Each Teacher head teaches all the Student heads with the contribution being more for the Student heads with which it has higher weights (higher cosine similarities). The Teacher teaches the Student heads by minimizing the divergence between the attention activation distributions for the soft-aligned heads. We may view AMAD as operating like a cross-attention itself.

One intuition behind is that, unlike embeddings for which each vision/language token shares the same order for both Teacher and Student, attention heads do not have a semantic order. (Cao et al. 2020) found that different subsets of attention heads in VL Transformers may encode different co-reference knowledge, e.g. a subset of heads may evolve to pivot on cross-modal interaction between image and text regimes. Therefore, even in the case when Teacher and Student have the same number of attention heads, we still cannot assume that the Teacher's and the Student's heads are aligned semantically without reordering. Conventional attention map distillation methods force Student heads to have exactly the same order as Teacher's, while AMAD allows similarity alignment based distillation free of head order: each Teacher teaches all the Students, the contribution being proportional to the similarity weight between them.

We conduct experiments on distilling VL-T5 (Cho et al. 2021) *base* to *small*, and on distilling BLIP (Li et al. 2022b) *large* to *base*. AMAD loss is applied on all their Transformer sub-modules, including T5-Encoder + Decoder for VL-T5, and Vision Transformer (ViT) + BERT for BLIP. We evaluate on VQA-2.0 (Goyal et al. 2019), COCO Captioning (Chen et al. 2015), and Multi30K (Elliott et al. 2016) datasets. We show that AMAD boosts performance.

Our contributions include:

- We propose Attention Map Alignment Distillation (AMAD) to distill attention maps from a Teacher Transformer to a Student Transformer with different numbers of attention heads. AMAD uses a soft-alignment approach so that each Teacher head teaches all the Student heads but in proportion to how similar the Student is to the Teacher. We show, under vision-language setting, that AMAD narrows the performance gap between the large Teacher and the small Student. With AMAD, researchers are set free from the same-number-of-heads restriction and have more choices over Transformers for distillation.
- We show that even without VL pre-training, distilled VL-T5 models outperform VL pre-trained VL-T5 models of the same size further fine-tuned with ground-truth data.
- We conduct extensive experiments on distilling VL models, which contributes to this relatively under-explored field in the current literature.

Related Work

Vision-Language Pre-training. Recently, large pre-trained Transformers (Liu et al. 2019; Lan et al. 2020; Clark et al. 2020; Yang et al. 2019; Ho et al. 2022; Li et al. 2022a; Appalaraju et al. 2021) have started to show improved capability in a variety of language modeling tasks (Zellers et al. 2018; Wang et al. 2018; Williams, Nangia, and Bowman 2017). Researchers have further extended these models to large image-text and video-text pre-training multi-modal models (Huang et al. 2020; Li et al. 2020a; Cho et al. 2020; Zhang et al. 2021; Sun et al. 2019; Zhu and Yang 2020; Miech et al. 2020; Radford et al. 2021; Appalaraju et al. 2024) for visual-linguistic tasks (Goyal et al. 2019; Hudson and Manning 2019; Lei et al. 2018; Mao et al. 2016; Xu et al. 2016; Zhou, Xu, and Corso 2018). These pre-trained models outperform previous approaches (Yu et al. 2018; Yu, Kim, and Kim 2018; Kim, Jun, and Zhang 2018; Anderson et al. 2018). As their models sizes grow rapidly, recent works have also explored parameter-efficient learning and model compression methods, including adapters (Sung, Cho, and Bansal 2022; Houlsby et al. 2019; Rebuffi, Bilen, and Vedaldi 2018, 2017), prompt tuning (Gu et al. 2021b; Lester, Al-Rfou, and Constant 2021; Li and Liang 2021).

Knowledge Distillation. Knowledge distillation (KD) (Hinton, Vinyals, and Dean 2015) transfers knowledge from a stronger Teacher network (T) to a Student network (S) by minimizing the divergence of their soft response and intermediate features (Gou et al. 2021). Compared with recent approaches to distillation in vision (Zagoruyko and Komodakis 2017; Peng et al. 2019; Tung and Mori 2019; Yang et al. 2022; Chen et al. 2022b; Wu et al. 2022b; Andonian, Chen, and Hamid 2022; He et al. 2022; Wu et al. 2022a) or language tasks (Wang et al. 2020b; Li et al. 2022c; Wang et al. 2021; Ding et al. 2023), distilling VL models is a relatively under-explored field, as pointed out by the review paper of (Chen et al. 2022a). (Fang et al. 2021) claim to be the first to distill vision-language Transformers, and (Wang et al. 2022b) claim to be the first to use multi-modal distillation for VL models. These two papers both focus on distilling Encoder-only VL Transformers. (Gu et al. 2021a; Ma et al. 2022) also involve distilling knowledge from visual and linguistic domains, but their architectures are based upon Mask RCNN (He et al. 2017) and ResNet (He et al. 2016), and apply to the visual task of object detection.

Specifically, for distilling large Transformers to smaller ones, recent work in language distillation (Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b; Sanh et al. 2019), vision distillation (Qu et al. 2022), and VL distillation (Fang et al. 2021) all show that applying attention map distillation to transfer the rich co-reference relations to Student can boost performance. These approaches apply attention map distillation either on all attention layers or only on the last self/cross-attention layer and the last cross-attention layer. Specifically, for a given multi-head attention layer, most of these approaches (Fang et al. 2021; Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b; Sanh et al. 2019) minimize the sum of divergence between the attention matrices of each head of Teacher and Student. However, this formulation only applies when the Teacher and Student have the same number of attention heads. (Qu et al. 2022) minimizes the divergence between the mean attention matrices of all the heads of Teacher and Student. However, as pointed out by (Cao et al. 2020), different attention heads encode different co-reference knowledge, hence applying mean reduction may result in knowledge loss.

Different from most approaches for distilling intermediate features in a one-to-one fashion, (Lin et al. 2022) propose a one-to-all spatial matching strategy for distilling Convnets feature maps, allowing each pixel of the Teacher feature to be distilled to all spatial locations in the Student by similarity mapping; (Ji, Heo, and Park 2021) propose to learn to match Teacher and Student features maps in different ResNet layers. Our design is also inspired by these works.

Attention Map Alignment Distillation

In this section, we introduce our proposed Attention Map Alignment Distillation (AMAD) method. We use plain lower case letters x for scalars, bold lower case letters $\mathbf{x}$ for vectors, and bold upper case letters $\mathbf{X}$ for matrices.

In a multi-head attention layer of Transformer (Vaswani et al. 2017), each entry of the attention matrix for a head is a dot-product of query and key vectors followed by softmax normalization (Bahdanau, Cho, and Bengio 2014). In matrix form, for each head h, if we denote the number of query vectors as q, the number of key vectors as k, and the attention matrix as $\mathbf{A}_h \in \mathbb{R}^{q\times k}$, then we have

$$\mathbf{A}_h = \text{softmax}(\mathbf{Q}_h\mathbf{K}_h^T/\sqrt{d_k}) \tag{1}$$

where $\mathbf{Q}_h$ and $\mathbf{K}_h$ are the query and key matrices of head h, and d_k is the dimension of the key as a scaling factor.

For each training data sample (not batched), the attention maps of all heads for a given H-head multi-head attention layer form a tensor of $[\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_H] \in \mathbb{R}^{H\times q\times k}$. AMAD aims at distilling the attention maps of the H_t heads in the Teacher to those of the H_s heads in Student. Generally, $H_s \leq H_t$.

For representational simplicity, let $n = q \cdot k$, and $\mathbf{t}_i \in \mathbb{R}^n$ denote the column vector representing the flattened attention map $\mathbf{A}_i \in \mathbb{R}^{q\times k}$ of Teacher head i. Let $\mathbf{s}_j \in \mathbb{R}^n$ denote the column vector representing the flattened attention map $\mathbf{A}_j \in \mathbb{R}^{q\times k}$ of Student head j.

For a given Teacher head $\mathbf{t}_i$, compute its cosine similarity w_{ij} with each Student head $\mathbf{s}_j$:

$$w_{ij} = \mathbf{t}_i \cdot \mathbf{s}_j/(\|\mathbf{t}_i\|_2 \cdot \|\mathbf{s}_j\|_2) \tag{2}$$

For the given Teacher head, we then compute its distilling contribution a_{ij} to each of the Student heads $j \in \{1, 2, \cdots, H_s\}$ by applying softmax non-linearity on the similarity weights w_{ij}:

$$a_{ij} = \frac{\exp(w_{ij})}{\sum_{m=1}^{H_s}\exp(w_{im})} \tag{3}$$

Then, we minimize the mean squared error between the given normalized Teacher head attention map $\mathbf{t}_i$ and the weighted sum of soft-aligned Student head attention maps.

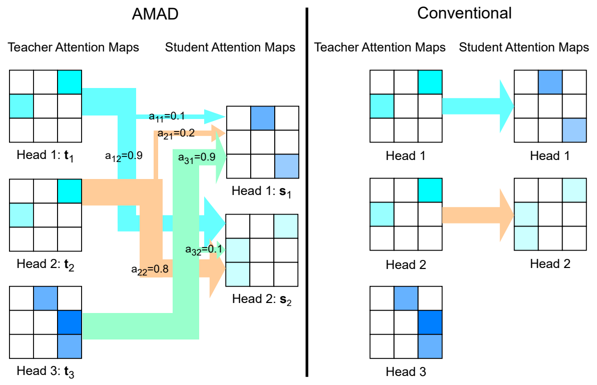


Figure 1: An illustration of AMAD in a toy case, corresponding to Equation 7, where the Teacher has $H_t = 3$ heads $(\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3)$ and Student has $H_s = 2$ heads $(\mathbf{s}_1, \mathbf{s}_2)$, all with self attention maps of dimension $n = q \times k = 3 \times 3$. Different colors of matrix entries denote different attention values. On the left, AMAD uses soft-alignment, so each Teacher head attention map teaches all the Student heads attention maps but in proportion to how close the Student head is to the Teacher head. As in the coloring of the matrices, Teacher heads 1 and 2 are similar to each other, and are relatively similar to Student head 2; While Teacher head 3 is similar to Student head 1. In this case, with AMAD, Teacher heads 1 and 2 instruct Student head 2 more (larger a_{12}, a_{22} and wider arrows above) and instruct Student head 1 less (smaller a_{11}, a_{21} and narrower arrows above), and Teacher head 3 mainly instructs Student head 1. Also note that the knowledge in the two similar Teacher heads $\mathbf{t}_1$ and $\mathbf{t}_2$ can be compressed mostly to a single Student head $\mathbf{s}_2$. While on the right, conventional attention map distillation method does not apply when the numbers of heads are different between Teacher and Student: Teacher head 3 has to be discarded in distilling, causing knowledge loss.

For each Attention head i of the Teacher,

$$\mathcal{L}_{\text{AMAD}_i} = \left\| \frac{\mathbf{t}_i}{\|\mathbf{t}_i\|_2} - \sum_{j=1}^{H_s} a_{ij} \cdot \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|_2} \right\|_2^2 \tag{4}$$

The total loss $\mathcal{L}_{\text{AMAD}}$ is the summation of $\mathcal{L}_{\text{AMAD}_i}$ over all the heads of the Teachers,

$$\mathcal{L}_{\text{AMAD}} = \sum_{i=1}^{H_t} \mathcal{L}_{\text{AMAD}_i} \tag{5}$$

Now we rewrite the above formulas using a matrix formulation to parallel computations. For each training data sample (not batched), recall that $n = q \cdot k$, let matrix $\mathbf{T} \in \mathbb{R}^{H_t\times n}$ represent the Teacher attention maps of all its heads, whose each row vector is the normalized flattened attention map $\mathbf{t}_i^T/\|\mathbf{t}_i\|_2$ of the i-th head. Similarly, let matrix $\mathbf{S} \in \mathbb{R}^{H_s\times n}$ represent the normalized Student attention maps of all its heads. We have (detailed derivation in Appendix),

$$\mathcal{L}_{\text{AMAD}} = \|\mathbf{T} - \text{softmax}_{\text{dim=row}}(\mathbf{T}\mathbf{S}^T)\mathbf{S}\|_2^2 \tag{6}$$

where the softmax function applies to each row. The pairwise similarity weight matrix $\mathbf{TS}^T$ before and after softmax are both of dimension $\mathbb{R}^{H_t \times H_s}$. Computation with respect to the i-th row of the matrix formulation corresponds to the operations regarding the i-th Teacher head as in Equation 4.

For instance, in the case as in Figure 1, we have,

$$\text{softmax}(\mathbf{TS}^T) = (a_{ij})_{H_t \times H_s} = \begin{pmatrix} 0.1 & 0.9 \\ 0.2 & 0.8 \\ 0.9 & 0.1 \end{pmatrix} \quad (7)$$

The above formulas focus on each training data sample (not batched); For implementation, we use batched tensor computations via PyTorch, and we L_2-normalize each row of the weighted sum softmax$(\mathbf{TS}^T)\mathbf{S}$ before calculating the mean squared error. **Code** is provided in the Appendix.

In contrast to previous attention map distillation methods directly minimizing $\|\mathbf{T}-\mathbf{S}\|_2^2$ or KL$(\mathbf{T}\|\mathbf{S})$, requiring $\mathbf{T}$ and $\mathbf{S}$ of the same shape (Fang et al. 2021; Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b), AMAD removes the limitation of requiring Teacher and Student to have the same number of attention heads, by letting each Teacher head teaches all the Student heads with the contribution being more for the Student heads with which it has a higher weight (higher cosine similarity), and supports flexible and smooth distillation because of the soft semantic alignment mechanism.

Formulation Variants

We refer to the above formulation as **Variant 1** and the corresponding loss as $\mathcal{L}_{\text{AMAD-1}}$, and we also explore the following ablative baselines and variants for multi-head attention-matrix distillation:

Baseline: One-to-one Distillation. In this baseline, following (Fang et al. 2021; Jiao et al. 2020; Sun et al. 2020; Wang et al. 2020b), we distill the attention maps in a one-to-one fashion. Note that different from previous work, we have more heads in the Teacher than in Student, $H_t \geq H_s$, so we distill the first H_s heads in Teacher to the H_s heads in Student, respectively, the extra $H_t - H_s$ Teacher heads are ignored during distillation, as in the right part of Figure 1:

$$\mathcal{L}_{\text{KD-ATT}} = \|\mathbf{S} - \mathbf{T}[:H_s, :]\|_2^2 \quad (8)$$

Variant 2: KL Divergence. In this variant, we minimize the sum of Kullback–Leibler divergence between the aligned weighted sum of Student multi-head attention map distributions and the Teacher distributions:

$$\mathcal{L}_{\text{AMAD-2}} = \text{KL}(\mathbf{T}\|\text{softmax}(\mathbf{TS}^T)\mathbf{S}) \quad (9)$$

where the Teacher $\mathbf{T}$ and the Student $\mathbf{S}$ are all L_1-normalized by each row in all KL variants, instead of L_2-normalized. Both input and target contain $q \cdot H_t$ distributions, and KL$(\cdot)$ is computed for each of these $q \cdot H_t$ distributions and then summed up.

Variants 3: Parameterized Projection. We borrow the idea from attention mechanisms to apply a learnable linear projection $\mathbf{W}$ on each flattened vector of attention map $\mathbf{s}_j$ of Student head j before computing similarity and alignment: $\mathbf{Ws}_j + \mathbf{b}$. In matrix form:

$$\tilde{\mathbf{S}} = \text{ReLU}(\mathbf{SW}^T + \mathbf{b}) \quad (10)$$

$$\mathcal{L}_{\text{AMAD-3}} = \text{KL}(\mathbf{T}\|\text{softmax}(\mathbf{T}\tilde{\mathbf{S}}^T)\tilde{\mathbf{S}}) \quad (11)$$

where $\mathbf{W} \in \mathbb{R}^{n \times n}$ is a learnable matrix and $\mathbf{b}$ is the bias.

Variant 4: Token-level Alignment. Here, we adopt a finer token-level granularity of aligning correspondence and allow independence for the soft alignment weights w_{ij} for different query attention vectors in Teacher / Student heads.

Formally, denote $\mathbf{T}_l \in \mathbb{R}^{H_t \times k}$ as the matrix whose h-th row vector is the l-th row vector $\mathbf{t}_{h,l}^T \in \mathbb{R}^k$ in the Teacher's h-th head attention map $\mathbf{A}_h$, and $\mathbf{S}_l \in \mathbb{R}^{H_s \times k}$ as the matrix whose each h-th row vector is $\mathbf{s}_{h,l}^T \in \mathbb{R}^k$. We have:

$$\mathcal{L}_{\text{AMAD-4}} = \sum_{l=1}^{q} \text{KL}(\mathbf{T}_l\|\text{softmax}(\mathbf{T}_l\mathbf{S}_l^T)\mathbf{S}_l) \quad (12)$$

where KL$(\cdot)$ is computed for each of the H_t distributions and then summed up. In this variant, the weight matrices softmax$(\mathbf{T}_l\mathbf{S}_l^T)$ are different for each l-th query attention vector, in contrast to the unified same weight matrix of softmax$(\mathbf{TS}^T)$ for all the queries in previous variants.

We report ablation results for each variant in Table 10. More theoretical analysis is in Appendix.

Experimental Setup

We distill VL-T5 *base* to *small*; and distill BLIP *large* to *base*. Table 1 summarizes their architectural backbones. Visualized architecture and distilling pipeline are in Appendix.

Knowledge Distillation (KD)

For training efficiency, we only apply distillation in downstream fine-tuning, no distillation involved in pre-training.

As in previous work (Hinton, Vinyals, and Dean 2015; Fang et al. 2021; Jiao et al. 2020; Sanh et al. 2019), we apply classification distillation loss on the classifier output logits. For VQA with a single classifier head,

$$\mathcal{L}_{\text{KD}} = \text{CE}(\mathbf{z}^S/\tau_d, \mathbf{z}^T/\tau_d) \quad (13)$$

where τ_d denotes the distillation temperature (Hinton, Vinyals, and Dean 2015), which we simply use 1, as in (Fang et al. 2021). $\mathbf{z}^S$ and $\mathbf{z}^T$ refer to the logits from Student and Teacher classifier. CE denotes Cross Entropy, i.e. $p_i = \frac{\exp(z_i/\tau_d)}{\sum_k \exp(z_k/\tau_d)}$ and $\mathcal{L}_{\text{KD}} = \sum_i p_i^T \cdot \log(p_i^S)$.

For auto-regressive captioning and translation tasks, the Teacher and the Student both take ground-truth answer token sequence as input in Teacher Forcing style to maintain consistency for distillation (Beyer et al. 2022),

$$\mathcal{L}_{\text{KD}} = \sum_{j=1}^{|y|} \text{CE}(\mathbf{z}_j^S/\tau_d, \mathbf{z}_j^T/\tau_d) \quad (14)$$

where $\mathbf{z}_j^S$ and $\mathbf{z}_j^T$ denote logits for the j-th output token from Student and Teacher classifier, and $|y|$ denotes length of seq.

The overall training objective for the Student $\mathcal{L}_{\text{TOTAL}}$ is a weighted sum of the classification distillation loss $\mathcal{L}_{\text{KD}}$ and the proposed loss $\mathcal{L}_{\text{AMAD}}$,

$$\mathcal{L}_{\text{TOTAL}} = \mathcal{L}_{\text{KD}} + \alpha\mathcal{L}_{\text{AMAD}} \quad (15)$$

	VL Model	#Learnable Params	Vision Stream Backbone	#Layers	d_{model}	#Heads	$\mathcal{L}_{\text{AMAD}}$	Language (Multi-modal) Stream Backbone	#Layers	d_{model}	#Heads	$\mathcal{L}_{\text{AMAD}}$
Teacher	VL-T5 *base*	220M	Faster R-CNN (frozen)	-	-	-		T5	12+12	768	12	
Student	VL-T5 *small*	60M	(Ren et al. 2015)	-	-	-		(Raffel et al. 2020)	6 + 6	512	8	✓
Teacher	BLIP *large*	446M	ViT	24	1024	16		BERT	12	768	12	
Student	BLIP *base*	210M	(Dosovitskiy et al. 2021)	12	768	12	✓	(Devlin et al. 2019)	12	768	12	✓

Table 1: Model architectures with details of their Transformer sub-modules. We distill VL-T5 *base* to *small*; and distill BLIP *large* to *base*. AMAD loss is applied on all Transformer modules, including T5-Encoder + Decoder for VL-T5, and ViT + BERT for BLIP. Conventional one-to-one attention map distillation does not apply to some of these modules when Teacher and Student have different numbers of attention heads; AMAD removes the same-number-of-heads constraint and works here.

We apply $\mathcal{L}_{\text{AMAD}}$ to distill the self/cross-attention maps of the last (Wang et al. 2020b; Fang et al. 2021) layers of each Transformer stack. α is tuned so that $\mathcal{L}_{\text{KD}}$ and $\mathcal{L}_{\text{AMAD}}$ scale similarly. We do not add ground-truth loss (Beyer et al. 2022).

Pre-training and Fine-tuning

As VL-T5 *small* is not released, we pretrain it by ourselves, adopting the same setting as how they pretrain *base*. It is pretrained on MS COCO (Lin et al. 2014; Chen et al. 2015), Visual Genome (Krishna et al. 2016), VQA-2.0 (Goyal et al. 2019), GQA (Hudson and Manning 2019), and Visual7W (Zhu et al. 2016). For VL-T5 *base* and BLIP, we directly use their released pretrained checkpoints.

After VL pre-training the Teacher and the Student, we finetune the Teacher on downtream tasks, adopting the same settings as in VL-T5 or BLIP. The Teacher model is then frozen and ready to be distilled. We then distill the Teacher to the Student with $\mathcal{L}_{\text{TOTAL}}$ on downstream tasks.

In some of our ablative settings, we do not conduct any VL pre-training for Student: After loading the language-only pre-trained language / multi-modal branch checkpoint and the vision-only pre-trained vision branch checkpoint, we directly finetune the Student on downstream VL tasks with distillation; Teacher is always pre-trained and finetuned.

Downstream Fine-tuning Datasets

We demonstrate visual question-answering performance on VQA-2.0 dataset. We report results on *Karpathy test*, *test-std* and *test-dev* via: https://visualqa.org/challenge.html.

We evaluate image captioning performance on MS COCO dataset (Chen et al. 2015). As in (Cho et al. 2021; Fang et al. 2021; Li et al. 2022b), we use the *Karparthy split* (Karpathy and Fei-Fei 2015), which re-splits train2014 and val2014 images (Lin et al. 2014) into ~11K / 5K / 5K for train / validation / test. We report BLEU@4 (B) (Papineni et al. 2002), CIDEr (C) (Vedantam, Zitnick, and Parikh 2015), METEOR (M) (Banerjee and Lavie 2005), SPICE (S) (Anderson et al. 2016) evaluation metrics.

We also evaluate multi-modal machine translation performance on Multi30K dataset (Elliott et al. 2016), where models translate English text to German given context images. We report BLEU@4 score using SacreBLEU (Post 2018).

We report our implementation details in Appendix.

Results and Analysis

Table 2 shows results on distilling VL-T5 and BLIP with AMAD, in comparison to recent vision-language models. The higher the better for all metrics. As in previous work (Fang et al. 2021; Cho et al. 2021), captioning performance are shown with their cross entropy optimization variants instead of CIDEr optimization variants. Figure 2 visualizes the effect of model size and number of VL pre-train images to VQA performance for recent models. Overall, Table 2 and Figure 2 support the following arguments:
1. Reducing VL model size within same family of models causes performance drops: If all trained with ground-truth supervision without distillation, VL-T5 *small* perform worse than *base* (row 13-14, 16-17); and BLIP *base* worse than *large* (row 19-20). The effect is also visualized in Figure 2.
2. For a same model, removing VL pre-training / Pre-training with fewer images degrades performance: Performance of VL-T5 *base* and *small* both drop significantly if not VL pre-trained (row 13 vs 16; 14 vs 17). Even when VL pretrained, smaller numbers of VL pre-training images cause performance drop in MiniVLM (row 7 vs 8) and in BLIP (pink triangles in Figure 2).

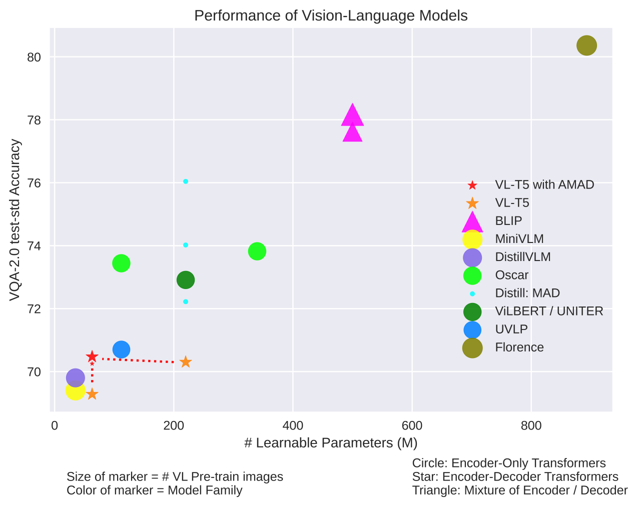


Figure 2: VL model performance with respect to # learnable params (X-axis) and # VL pre-train images (marker size). AMAD pushes VL-T5 towards upper-left (orange to red).

	Method	# Learnable Parameters	# VL pre-train Images	Distilled From Which Model	VQA-2.0 Acc Karpathy	test-std	test-dev	COCO Captioning B	C	M	S
	CNN-LSTM-based Models										
1	Up-Down (Anderson et al. 2018)	-	108K	✗	-	70.34		36.2	113.5	27.0	20.3
2	GLIED (Liu et al. 2020)	18.3M	-	*self distillation*	-	-	-	37.9	118.2	28.3	21.2
	Encoder-Only Transformers										
3	ViLBERT (Lu et al. 2019)	220M	3M	✗	-	70.92	70.55	-	-	-	-
4	UNITER *base* (Chen et al. 2020)	220M	4M	✗	-	72.91	72.70	-	-	-	-
5	Unified VLP (Zhou et al. 2020)	112M	3M	✗	-	70.7	-	36.5	116.9	28.4	21.2
6	OSCAR *base* (Li et al. 2020b)	112M	4M	✗	-	73.44	73.16	36.5	123.7	30.3	23.1
7	MiniVLM (Wang et al. 2020a)	35M	7M	✗	-	-	-	34.3	116.7	28.1	21.3
8	MiniVLM (Wang et al. 2020a)	35M	14M	✗	-	69.4	69.1	35.6	119.8	28.6	21.6
9	DistillVLM (Fang et al. 2021)	35M	7M	Oscar *base*	-	69.8	69.6	35.6	120.8	28.7	22.1
10	MAD-ViLBERT (Wang et al. 2022b)	220M	-	CLIP-V + CLIP-T	-	72.22	-	-	-	-	-
11	MAD-UNITER (Wang et al. 2022b)	220M	-	CLIP-V + CLIP-T	-	74.02	-	-	-	-	-
	Encoder-Decoder Transformers										
12	OFA *huge* (Wang et al. 2022a)	930M	15M	✗	-	82.0	82.0	43.9	145.3	31.8	24.8
13	VL-T5 *base* (Cho et al. 2021)	220M	180K	✗	67.9	70.30	-	34.5	116.5	28.7	21.9
14	VL-T5 *small*	63M	180K	✗	66.72	69.28	69.04	32.8	108.2	27.0	20.4
15	**Ours** VL-T5 *small* **w/ AMAD**	63M	180K	VL-T5 *base* reproduced	**68.06**	**70.47**	**70.41**	**33.9**	**114.4**	**28.3**	**21.5**
16	VL-T5 *base* (Cho et al. 2021)	220M	0	✗	-	-	-	32.6	109.4	28.2	21.0
17	VL-T5 *small*	63M	0	✗	56.44	-	58.47	30.8	101.4	26.3	19.5
18	**Ours** VL-T5 *small* **w/ AMAD**	63M	0	VL-T5 *base* reproduced	**67.79**	**70.25**	**70.06**	**33.3**	**112.9**	**28.0**	**21.3**
	Mixture of Encoder / Decoder Transformers										
19	BLIP *large* (Li et al. 2022b)	446M	129M	✗	-	-	-	40.4	136.7	-	-
20	BLIP *base* (Li et al. 2022b)	210M	129M	✗	-	78.32	78.25	39.7	133.3	-	-
21	**Ours** BLIP *base* **w/ AMAD**	210M	129M	BLIP *large*	-	-	-	**40.0**	**134.1**	**31.0**	**23.9**
22	BLIP *base* (Li et al. 2022b)	210M	14M	✗	-	77.62	77.54	38.6	129.7	-	-
23	BLIP *base*	210M	0	✗	-	-	-	34.7	115.8	28.5	21.5
24	**Ours** BLIP *base* **w/ AMAD**	210M	0	BLIP *large*	-	-	-	**38.7**	**129.6**	**30.4**	**23.3**

Table 2: Results on distilling VL-T5 and BLIP with AMAD, with comparisons to recent VL models. AMAD narrows the performance gaps caused by reducing model size or removing VL pre-training. Furthermore, to our surprise, the distilled VL-T5 *small* w/o VL pretraining (row 18) even outperforms VL pre-trained and GT fine-tuned VL-T5 *small* (row 14).

3. Distilling with AMAD narrows the aforementioned gaps caused by shrinking model size or by removing VL pre-training: Supported by results of distilling VL pre-trained VL-T5 (row 14 vs 15); distilling non-VL-pretrained VL-T5 (row 17 vs 18); and distilling BLIP (row 20 vs 21).

Note that, DistillVLM (row 9) distilled Oscar *base* (row 6) to MiniVLM (row 8) and achieved 0.4% VQA accuracy boost and 1.0% Captioning CIDEr score boost. And they (Fang et al. 2021) claimed to be the first work to apply knowledge distillation in training VL models. Neither MiniVLM nor DistillVLM has released their model or code.

4. Knowledge distillation compensates to some degree for the absence of VL pre-training: When we do not conduct any VL pre-training for VL-T5 *small*, fine-tuning distilled VL-T5 *small* (row 18) even outperforms the ground-truth supervised VL pre-trained and fine-tuned VL-T5 *small* baseline (row 14). It also outperforms non-VL-pretrained VL-T5 *base* (Cho et al. 2021) reported numbers (row 16). The performance is also rather comparable to other recent pre-trained and finetuned VL models. Also, for BLIP, fine-tuning distillation (row 24) compensates for 14M-scale VL pre-traning (row 22), but cannot fully compensate for 129M-scale pre-training (row 20).

One possible explanation is that the knowledge obtained in the pre-training stage of the Teacher can somehow be distilled to the Student in the downstream fine-tuning process when the Student tries to mimic the Teacher's classification logits and tries to align and mimic the Teacher's attention maps, even if the Student has no access to the pre-training data by itself. The Teacher's attention maps contain valuable intra-modal and cross-modal coreference relations learned from the pre-training dataset, and $\mathcal{L}_{\text{AMAD}}$ helps the Student to inherit the rich learned representation from the Teacher.

VL-T5 Model	#Params	test-2016	test-2017	test-2018
Teacher (reproduced)	220M	44.00	39.40	37.00
Student	60M	41.90	36.85	34.02
Student **w/ AMAD**	60M	43.88	38.70	36.64
Δ		**+1.98**	**+1.85**	**+2.62**

Table 3: Multi30K English-German translation BLEU@4 score. The VL-T5 *small* Student distilled with AMAD outperforms ground-truth (GT) fine-tuned VL-T5 *small*, and its performance is close to the VL-T5 *base* Teacher's.

Besides these observations 1-4, we show in Table 3 that AMAD can generalize well to language-heavy translation

BLIP Model	#Params	B	C	M	S
Teacher (Li et al. 2022b)	446M	40.4	136.7	-	-
Student	210M	22.3	65.1	20.9	13.5
Student **w/ AMAD**	210M	29.7	95.0	25.4	18.5
Δ		**+7.4**	**+29.9**	**+4.5**	**+5.0**

Table 4: Results on COCO Captioning karpathy test split. All BLIP Students are **w/o any pretraining**, i.e. the ViT and BERT modules are **randomly initialized** for all Students. Results for pretrained BLIP models are also reported in Appendix. Student distilled with AMAD outperforms the GT trained Student by a surprisingly large margin, although performance still degrades a lot because of neither VL nor uni-modal pre-trained. This indicates that uni-modal pretraining is still necessary even when distillation is applied.

VL-T5	#Params	$\mathcal{L}_{\text{KD}}$	$\mathcal{L}_{\text{AMAD}}$	B	C	M	S
Teacher	220M			34.2	115.1	28.3	21.6
Student	60M			32.8	108.2	27.0	20.4
Student	60M	✓		33.1	111.8	27.9	21.2
Student (RKD-D)	60M	✓		33.6	113.4	28.1	21.3
Student	60M	✓	✓	**33.9**	**114.4**	**28.3**	**21.5**

Table 5: Ablation results on COCO Captioning karpathy test split. All VL-T5 Students are first **VL pretrained**. Ablation results w/o VL pretraining are in Appendix. Distilling with $\mathcal{L}_{\text{AMAD}}$ outperforms logits distillation with $\mathcal{L}_{\text{KD}}$, and narrows the CIDEr gap between the *small* Student and the *base* Teacher to only 0.7% with 72% less parameters. The VL-T5 Teacher is reproduced, as their fine-tuned checkpoints are not released, also in following Tables. We also compare AMAD with RKD-D (Park et al. 2019), a distance-based similarity distillation method: We treat attention maps of all heads as a whole single feature and apply RKD-D on that, since #heads are different between Teacher and Student.

VL-T5	#Params	$\mathcal{L}_{\text{KD}}$	$\mathcal{L}_{\text{AMAD}}$	Karpathy	std	dev
Teacher	226M			68.75	71.34	71.23
Student	63M			66.72	69.28	69.04
Student	63M	✓		67.74	70.19	70.10
Student	63M	✓	✓	**68.06**	**70.47**	**70.41**

Table 6: VQA-2.0 test. All Students are first **VL pretrained**.

VL-T5	#Params	$\mathcal{L}_{\text{KD}}$	$\mathcal{L}_{\text{AMAD}}$	Karpathy	std	dev
Teacher	226M			68.75	71.34	71.23
Student	63M			56.44	-	58.47
Student	63M	✓		66.39	-	68.71
Student	63M	✓	✓	**67.79**	**70.25**	**70.06**

Table 7: Ablation results on VQA-2.0 test split. All Students are **w/o VL pretraining**, i.e. are initialized with language-only pre-trained T5 and vision-only pre-trained detector chechpoints. This shows that vanilla finetuning logits distillation already compensates to some degree for the absence of VL pre-training, and AMAD narrows the gap further.

VL-T5	#Params	Loss Variant	Acc	Δ
Teacher	226M		68.75	
Student	63M	Ground-Truth	66.72	
Student	63M	$\mathcal{L}_{\text{KD}}$	67.74	+1.02
Student	63M	$\mathcal{L}_{\text{KD}} + \mathcal{L}_{\text{KD-ATT}}$	67.73	+1.01
Student	63M	$\mathcal{L}_{\text{KD}} + \mathcal{L}_{\text{AMAD-1}}$	67.96	+1.24
Student	63M	$\mathcal{L}_{\text{KD}} + \mathcal{L}_{\text{AMAD-2}}$	**68.06**	+1.34
Student	63M	$\mathcal{L}_{\text{KD}} + \mathcal{L}_{\text{AMAD-3}}$	68.02	+1.30
Student	63M	$\mathcal{L}_{\text{KD}} + \mathcal{L}_{\text{AMAD-4}}$	68.05	+1.33
Student	63M	**w/ AMAD** mean	68.02	+1.30
		std-err of the mean	$\pm$ 0.02	$\pm$ 0.02

Table 8: Ablation results on VQA-2.0 Karpathy test split. All VL-T5 Students are first **VL pretrained**. The baseline of distilling the attention maps from the first 8 heads of Teacher to those of the 8 heads of Student in a one-to-one fashion with $\mathcal{L}_{\text{KD-ATT}} = \|\mathbf{S} - \mathbf{T}[: H_s, :]\|_2^2$ does not help improve performance than distilling with $\mathcal{L}_{\text{KD}}$ only, maybe because the extra $H_t - H_s$ Teacher heads are discarded during distillation, causing forced knowledge loss. Meanwhile, all $\mathcal{L}_{\text{AMAD}}$ variants help improve performance consistently with a small standard error. KL variants for $\mathcal{L}_{\text{AMAD}}$ (2, 3, 4) perform slightly better than MSE $\mathcal{L}_{\text{AMAD-1}}$. We have not observed significant performance change brought by learnable projection $\mathcal{L}_{\text{AMAD-3}}$ or token-level alignment $\mathcal{L}_{\text{AMAD-4}}$, compared to $\mathcal{L}_{\text{AMAD-2}}$.

task. Note that Table 4 unveils one of our limitations that, although fine-tuning distillation might close the performance gap of removing VL pretraining, uni-modal pretraining is still necessary.

Ablations

Effects of logits distillation $\mathcal{L}_{\text{KD}}$ and AMAD $\mathcal{L}_{\text{AMAD}}$: We ablates on captioning and VQA tasks in w/ and w/o VL pretrain settings in Table 5-7 and in Appendix, respectively. In w/o VL pretrain settings, Students are finetuned directly after loading the vision pretrained Faster R-CNN and language pretrained T5.

AMAD Variants and Baselines are analyzed in Table 8.

We present visualizations and qualitative analysis of $\mathcal{L}_{\text{AMAD}}$ distilled attention maps in Appendix.

Conclusion

We have proposed the Attention Map Alignment Distillation (AMAD) method to distill attention maps from a Teacher to a Student Transformer with different numbers of attention heads. Our experiments confirmed that AMAD narrows the performance gap between the large Teacher and the small Student in both discriminative VQA and auto-regressive generative captioning / translation tasks. Our ablation further suggested that fine-tuning knowledge distillation can compensate to some degree for the absence of VL pre-training for VL Transformers.

References

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.

Anderson, P.; Fernando, B.; Johnson, M.; and Gould, S. 2016. SPICE: Semantic Propositional Image Caption Evaluation. In *ECCV*.

Anderson, P.; He, X.; Buehler, C.; Teney, D.; Johnson, M.; Gould, S.; and Zhang, L. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6077–6086.

Andonian, A.; Chen, S.; and Hamid, R. 2022. Robust Cross-Modal Representation Learning with Progressive Self-Distillation. In *CVPR*, 16430–16441.

Appalaraju, S.; Jasani, B.; Kota, B. U.; Xie, Y.; and Manmatha, R. 2021. Docformer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 993–1003.

Appalaraju, S.; Tang, P.; Dong, Q.; Sankaran, N.; Zhou, Y.; and Manmatha, R. 2024. DocFormerv2: Local Features for Document Understanding - Full Paper. *AAAI*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Banerjee, S.; and Lavie, A. 2005. METEOR : An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *ACL Workshop*.

Beyer, L.; Zhai, X.; Royer, A.; Markeeva, L.; Anil, R.; and Kolesnikov, A. 2022. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF Conference on CVPR*, 10925–10934.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.

Cao, J.; Gan, Z.; Cheng, Y.; Yu, L.; Chen, Y.-C.; and Liu, J. 2020. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. In *ECCV*.

Chen, F.; Zhang, D.; Han, M.; Chen, X.; Shi, J.; Xu, S.; and Xu, B. 2022a. VLP: A Survey on Vision-Language Pretraining. *ArXiv*, abs/2202.09061.

Chen, X.; Cao, Q.; Zhong, Y.; Zhang, J.; Gao, S.; and Tao, D. 2022b. DearKD: Data-Efficient Early Knowledge Distillation for Vision Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12052–12062.

Chen, X.; Fang, H.; Lin, T.-Y.; Vedantam, R.; Gupta, S.; Dollar, P.; and Zitnick, C. L. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server.

Chen, X.; Wang, X.; Changpinyo, S.; Piergiovanni, A.; Padlewski, P.; Salz, D.; Goodman, S.; Grycner, A.; Mustafa, B.; Beyer, L.; et al. 2022c. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*.

Chen, Y.-c.; Li, L.; Yu, L.; Kholy, A. E.; Ahmed, F.; Gan, Z.; Cheng, Y.; and Liu, J. 2020. UNITER: UNiversal Image-TExt Representation Learning. In *ECCV*.

Cho, J.; Lei, J.; Tan, H.; and Bansal, M. 2021. Unifying Vision-and-Language Tasks via Text Generation. In *ICML*.

Cho, J.; Lu, J.; Schwenk, D.; Hajishirzi, H.; and Kembhavi, A. 2020. X-LXMERT: Paint, Caption and Answer Questions with Multi-Modal Transformers. In *EMNLP*.

Clark, K.; Luong, M.-T.; Le, Q. V.; and Manning, C. D. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

Ding, Z.; Jiang, G.; Zhang, S.; Guo, L.; and Lin, W. 2023. SKDBERT: Compressing BERT via Stochastic Knowledge Distillation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6): 7414–7422.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR*.

Elliott, D.; Frank, S.; Sima'an, K.; and Specia, L. 2016. Multi30K : Multilingual English-German Image Descriptions. In *ACL Workshop*, 70–74.

Fang, Z.; Wang, J.; Hu, X.; Wang, L.; Yang, Y.; and Liu, Z. 2021. Compressing Visual-linguistic Model via Knowledge Distillation. *ICCV*.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

Goyal, Y.; Khot, T.; Agrawal, A.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2019. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. *International Journal of Computer Vision*.

Gu, X.; Lin, T.-Y.; Kuo, W.; and Cui, Y. 2021a. Open-vocabulary Object Detection via Vision and Language Knowledge Distillation. In *ICLR*.

Gu, Y.; Han, X.; Liu, Z.; and Huang, M. 2021b. PPT: Pre-trained Prompt Tuning for Few-shot Learning. *ArXiv*, abs/2109.04332.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, R.; Sun, S.; Yang, J.; Bai, S.; and Qi, X. 2022. Knowledge Distillation as Efficient Pre-training: Faster Convergence, Higher Data-efficiency, and Better Transferability. In *CVPR*, 9161–9171.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. In *NIPS Deep Learning and Representation Learning Workshop*.

Ho, C.-H.; Appalaraju, S.; Jasani, B.; Manmatha, R.; and Vasconcelos, N. 2022. YORO-Lightweight End to End Visual Grounding. In *European Conference on Computer Vision - ECCV CAMP Workshop*.

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

Huang, Z.; Zeng, Z.; Liu, B.; Fu, D.; and Fu, J. 2020. Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers.

Hudson, D. A.; and Manning, C. D. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*. ISBN 9781728132938.

Ji, M.; Heo, B.; and Park, S. 2021. Show, attend and distill: Knowledge distillation via attention-based feature matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 7945–7952.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4163–4174. Online: Association for Computational Linguistics.

Karpathy, A.; and Fei-Fei, L. 2015. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *CVPR*. ISBN 9781467369640.

Kim, J.-h.; Jun, J.; and Zhang, B.-t. 2018. Bilinear Attention Networks. In *NeurIPS*, 1–12.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Jia-Li, L.; Shamma, D. A.; Michael Bernstein; and Fei-Fei, L. 2016. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Lei, J.; Yu, L.; Bansal, M.; and Berg, T. L. 2018. Tvqa: Localized, compositional video question answering. In *EMNLP*.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *EMNLP*.

Li, C.; Fehérvári, I.; Zhao, X.; Macêdo, I.; and Appalaraju, S. 2022a. SeeTek: Very Large-Scale Open-set Logo Recognition with Text-Aware Metric Learning. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 587–596.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *ICML*.

Li, J.; Li, D.; Xiong, C.; and Hoi, S. 2022b. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *ICML*.

Li, L.; Chen, Y.-C.; Yu Cheng, Z. G.; Yu, L.; and Liu, J. 2020a. HERO: Hierarchical Encoder for Video+Language Omni-representation Pre-training. In *EMNLP*.

Li, X.; Yin, X.; Li, C.; Zhang, P.; Hu, X.; Zhang, L.; Wang, L.; Hu, H.; Dong, L.; Wei, F.; Choi, Y.; and Gao, J. 2020b. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In *ECCV*.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation.

Li, Z.; Wang, Z.; Tan, M.; Nallapati, R.; Bhatia, P.; Arnold, A.; Xiang, B.; and Roth, D. 2022c. DQ-BART: Efficient Sequence-to-Sequence Model via Joint Distillation and Quantization. *arXiv preprint arXiv:2203.11239*.

Lin, S.; Xie, H.; Wang, B.; Yu, K.; Chang, X.; Liang, X.; and Wang, G. 2022. Knowledge Distillation via the Target-aware Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10915–10924.

Lin, T. Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft COCO: Common Objects in Context. In *ECCV*. ISBN 978-3-319-10601-4.

Liu, F.; Ren, X.; Liu, Y.; Lei, K.; and Sun, X. 2020. Exploring and distilling cross-modal information for image captioning. *arXiv preprint arXiv:2002.12585*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lu, J.; Batra, D.; Parikh, D.; and Lee, S. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *NeurIPS*.

Ma, Z.; Luo, G.; Gao, J.; Li, L.; Chen, Y.; Wang, S.; Zhang, C.; and Hu, W. 2022. Open-Vocabulary One-Stage Detection with Hierarchical Visual-Language Knowledge Distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14074–14083.

Mao, J.; Huang, J.; Toshev, A.; Camburu, O.; Yuille, A.; and Murphy, K. 2016. Generation and Comprehension of Unambiguous Object Descriptions. In *CVPR*.

Miech, A.; Alayrac, J.-B.; Smaira, L.; Laptev, I.; Sivic, J.; and Zisserman, A. 2020. End-to-end learning of visual representations from uncurated instructional videos. In *CVPR*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W. W.-j. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL*. ISBN 1-55860-883-4.

Park, W.; Kim, D.; Lu, Y.; and Cho, M. 2019. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3967–3976.

Peng, B.; Jin, X.; Liu, J.; Li, D.; Wu, Y.; Liu, Y.; Zhou, S.; and Zhang, Z. 2019. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5007–5016.

Post, M. 2018. A Call for Clarity in Reporting BLEU Scores. In *WMT*, volume 1, 186–191.

Qu, X.; Ding, C.; Li, X.; Zhong, X.; and Tao, D. 2022. Distillation Using Oracle Queries for Transformer-Based Human-Object Interaction Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19558–19567.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *JMLR*, 21: 1–67.

Rebuffi, S.-A.; Bilen, H.; and Vedaldi, A. 2017. Learning multiple visual domains with residual adapters. In *NIPS*.

Rebuffi, S.-A.; Bilen, H.; and Vedaldi, A. 2018. Efficient Parametrization of Multi-domain Deep Neural Networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8119–8127.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*.

Sanh, V.; Debut, L.; Chaumond, J.; and Wolf, T. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Su, W.; Zhu, X.; Cao, Y.; Li, B.; Lu, L.; Wei, F.; and Dai, J. 2019. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. In *ICLR*.

Sun, C.; Myers, A.; Vondrick, C.; Murphy, K.; and Schmid, C. 2019. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7464–7473.

Sun, Z.; Yu, H.; Song, X.; Liu, R.; Yang, Y.; and Zhou, D. 2020. MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2158–2170. Online: Association for Computational Linguistics.

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

Tung, F.; and Mori, G. 2019. Similarity-Preserving Knowledge Distillation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1365–1374.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. In *NIPS*.

Vedantam, R.; Zitnick, C. L.; and Parikh, D. 2015. CIDEr: Consensus-based Image Description Evaluation. In *CVPR*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Wang, J.; Hu, X.; Zhang, P.; Li, X.; Wang, L.; Zhang, L.; Gao, J.; and Liu, Z. 2020a. MiniVLM: A Smaller and Faster Vision-Language Model.

Wang, P.; Yang, A.; Men, R.; Lin, J.; Bai, S.; Li, Z.; Ma, J.; Zhou, C.; Zhou, J.; and Yang, H. 2022a. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *International Conference on Machine Learning*, 23318–23340. PMLR.

Wang, W.; Bao, H.; Huang, S.; Dong, L.; and Wei, F. 2021. MiniLMv2: Multi-Head Self-Attention Relation Distillation for Compressing Pretrained Transformers. In *ACL Findings*.

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

Wang, Z.; Codella, N.; Chen, Y.-C.; Zhou, L.; Dai, X.; Xiao, B.; Yang, J.; You, H.; Chang, K.-W.; Chang, S.-f.; et al. 2022b. Multimodal Adaptive Distillation for Leveraging Unimodal Encoders for Vision-Language Tasks. *arXiv preprint arXiv:2204.10496*.

Williams, A.; Nangia, N.; and Bowman, S. R. 2017. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

Wu, H.; Gao, Y.; Zhang, Y.; Lin, S.; Xie, Y.; Sun, X.; and Li, K. 2022a. Self-supervised Models are Good Teaching Assistants for Vision Transformers. In *ICML*.

Wu, K.; Zhang, J.; Peng, H.; Liu, M.; Xiao, B.; Fu, J.; and Yuan, L. 2022b. Tinyvit: Fast pretraining distillation for small vision transformers. In *ECCV*, 68–85. Springer.

Xu, J.; Mei, T.; Yao, T.; and Rui, Y. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R. R.; and Le, Q. V. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Yang, Z.; Li, Z.; Jiang, X.; Gong, Y.; Yuan, Z.; Zhao, D.; and Yuan, C. 2022. Focal and global knowledge distillation for detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4643–4652.

Yu, L.; Lin, Z.; Shen, X.; Yang, J.; Lu, X.; Bansal, M.; and Berg, T. L. 2018. MAttNet : Modular Attention Network for Referring Expression Comprehension. In *CVPR*.

Yu, Y.; Kim, J.; and Kim, G. 2018. A joint sequence fusion model for video question answering and retrieval. In *ECCV*.

Yuan, L.; Chen, D.; Chen, Y.-L.; Codella, N.; Dai, X.; Gao, J.; Hu, H.; Huang, X.; Li, B.; Li, C.; et al. 2021. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*.

Zagoruyko, S.; and Komodakis, N. 2017. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *ICLR*.

Zellers, R.; Bisk, Y.; Schwartz, R.; and Choi, Y. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *EMNLP*.

Zhang, P.; Li, X.; Hu, X.; Yang, J.; Zhang, L.; Wang, L.; Choi, Y.; and Gao, I. 2021. VinVL: Making Visual Representations Matter in Vision-Language Models.

Zhou, L.; Palangi, H.; Zhang, L.; Hu, H.; Corso, J. J.; and Gao, J. 2020. Unified Vision-Language Pre-Training for Image Captioning and VQA. In *AAAI*.

Zhou, L.; Xu, C.; and Corso, J. J. 2018. Towards automatic learning of procedures from web instructional videos. In *AAAI*.

Zhu, L.; and Yang, Y. 2020. ActBERT: Learning Global-Local Video-Text Representations. In *CVPR*.

Zhu, Y.; Groth, O.; Bernstein, M.; and Fei-Fei, L. 2016. Visual7W: Grounded Question Answering in Images. In *CVPR*. ISBN 978-1-4673-8851-1.

No Head Left Behind - Multi-Head Alignment Distillation for Transformers - Supplemental

Tianyang Zhao[1,2*], **Kunwar Yashraj Singh**[1 †], **Srikar Appalaraju**[1 ‡], **Peng Tang**[1], **Vijay Mahadevan**[1], **R. Manmatha**[1], **Ying Nian Wu**[1,2]

[1]AWS AI Labs, [2]University of California, Los Angeles

tyzhao@ucla.edu, {sinkunwa, srikara, tangpen, vmahad, manmatha, wunyin}@amazon.com

Appendix / Supplementary Material

In this Appendix, we provide a further illustration on the matrix-form loss derivation and formulation of different AMAD variants in Section A; more ablation results and a reverse experiment of distilling a smaller Teacher to a larger Student in Section B; implementation details including training environment, training time, and hyper-parameters in Section C; an additional illustration of the distillation workflow for our experimental setup and the architecture of VL-T5 in Section D; visualizations of AMAD distilled cross- and self- attention maps in Section E; and PyTorch code for the proposed AMAD loss in Section F.

A. Derivation and Discussion on AMAD Loss

We provide a further illustration on the matrix-form loss formulation of different AMAD variants in this subsection. The derivation and interpretation of Equation 6 and its equivalence to Equation 2 - Equation 5 in the main paper are straight-forward, and we illustrate it here for better readability to broader reader group.

For each data sample (not batched), for AMAD Variant 1 for instance, we first compute the pair-wise cosine similarity matrix $\mathbf{W} \in \mathbb{R}^{H_t \times H_s}$:

$$\mathbf{W} := \mathbf{T}\mathbf{S}^T = \begin{pmatrix} \mathbf{t}_1^T/\|\mathbf{t}_1\|_2 \\ \mathbf{t}_2^T/\|\mathbf{t}_2\|_2 \\ \vdots \\ \mathbf{t}_{H_t}^T/\|\mathbf{t}_{H_t}\|_2 \end{pmatrix} \begin{pmatrix} \frac{\mathbf{s}_1}{\|\mathbf{s}_1\|_2} & \frac{\mathbf{s}_2}{\|\mathbf{s}_2\|_2} & \cdots & \frac{\mathbf{s}_{H_s}}{\|\mathbf{s}_{H_s}\|_2} \end{pmatrix} \tag{1}$$

$$= \begin{pmatrix} w_{11} & \cdots & w_{1,H_s} \\ \vdots & \ddots & \vdots \\ w_{H_t,1} & \cdots & w_{H_t,H_s} \end{pmatrix} \tag{2}$$

where each of its entry $w_{ij} = \mathbf{t}_i \cdot \mathbf{s}_j / (\|\mathbf{t}_i\|_2 \cdot \|\mathbf{s}_j\|_2)$ is the cosine similarity between the i-th head flattened attention matrix $\mathbf{t}_i$ of Teacher and the j-th head flattened attention matrix $\mathbf{s}_j$ of Student.

Note that for Variant 4, instead of sharing a common $\mathbf{W}$ for all different query attention vectors, each group of l-th query attention vectors have their own weight matrix $\mathbf{W}_l = \mathbf{T}_l\mathbf{S}_l^T$, so that they can attend to the Student heads in different ways: Let each query attention vector $\mathbf{t}_{h,l}^T$ or $\mathbf{s}_{h,l}^T$ refers to the l-th row vector in the Teacher / Student attention map $\mathbf{A}_h$. In attention mechanism, for a given head h, the l-th query attention vector encodes how the l-th query vector in $\mathbf{Q}_h$ should attend to the key vectors in $\mathbf{K}_h$. In all previous variants, different query attention vectors in the same Teacher head share the same set of soft alignment weights for the corresponding query attention vectors in different Student heads. In this variant, different row vectors in the same Teacher head attention map can attend to the vectors of different Student heads independently.

After Equation 2, we then apply softmax by each row (for each Teacher head) to get the Teacher-Student soft-align weight matrix. Denote $a_{ij} := \frac{\exp(w_{ij})}{\sum_{m=1}^{H_s}\exp(w_{im})}$, we have:

$$\text{softmax}(\mathbf{T}\mathbf{S}^T) = \begin{pmatrix} a_{11} & \cdots & a_{1,H_s} \\ \vdots & \ddots & \vdots \\ a_{H_t,1} & \cdots & a_{H_t,H_s} \end{pmatrix} \tag{3}$$

We then multiply the Teacher-Student soft-align weight matrix on the left of the Student matrix to get the soft-aligned weighted sum of all Student heads for each Teacher head:

$$\text{softmax}(\mathbf{T}\mathbf{S}^T)\mathbf{S} = \begin{pmatrix} a_{11} & \cdots & a_{1,H_s} \\ \vdots & \ddots & \vdots \\ a_{H_t,1} & \cdots & a_{H_t,H_s} \end{pmatrix} \begin{pmatrix} \mathbf{s}_1^T/\|\mathbf{s}_1\|_2 \\ \mathbf{s}_2^T/\|\mathbf{s}_2\|_2 \\ \vdots \\ \mathbf{s}_{H_s}^T/\|\mathbf{s}_{H_s}\|_2 \end{pmatrix} \tag{4}$$

$$= \begin{pmatrix} \sum_{j=1}^{H_s} a_{1j} \cdot \frac{\mathbf{s}_j^T}{\|\mathbf{s}_j\|_2} \\ \vdots \\ \sum_{j=1}^{H_s} a_{H_t,j} \cdot \frac{\mathbf{s}_j^T}{\|\mathbf{s}_j\|_2} \end{pmatrix} \tag{5}$$

Finally, we compute the difference between the Teacher and the soft-aligned Student. Note that each Teacher head

*Work conducted during an internship at Amazon.
†Corresponding Author
‡Corresponding Author

is getting different weighted sums according to the soft-aligned weights. Each Teacher head contributes to all Student heads, but contributes more to the Student heads which are similar to it:

$$\mathbf{T} - \text{softmax}(\mathbf{T}\mathbf{S}^T)\mathbf{S} = \begin{pmatrix} \frac{\mathbf{t}_1^T}{\|\mathbf{t}_1\|_2} - \sum_{j=1}^{H_s} a_{1j} \cdot \frac{\mathbf{s}_j^T}{\|\mathbf{s}_j\|_2} \\ \vdots \\ \frac{\mathbf{t}_{H_t}^T}{\|\mathbf{t}_{H_t}\|_2} - \sum_{j=1}^{H_s} a_{H_t,j} \cdot \frac{\mathbf{s}_j^T}{\|\mathbf{s}_j\|_2} \end{pmatrix} \tag{6}$$

$$\mathcal{L}_{\text{AMAD}} := \|\mathbf{T} - \text{softmax}_{\text{dim=row}}(\mathbf{T}\mathbf{S}^T)\mathbf{S}\|_2^2 \tag{7}$$

$$= \sum_{i=1}^{H_t} \mathcal{L}_{\text{AMAD}_i} = \sum_{i=1}^{H_t} \left\| \frac{\mathbf{t}_i}{\|\mathbf{t}_i\|_2} - \sum_{j=1}^{H_s} a_{ij} \cdot \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|_2} \right\|_2^2 \tag{8}$$

We have also formulated a Student-to-Teacher (S2T) variant of AMAD during exploration. In this variant, instead of letting each Teacher attends to the Student heads, here we let each Student attends to the Teacher heads:

$$\mathcal{L}_{\text{AMAD}} = \|\mathbf{S} - \text{softmax}_{\text{dim=row}}(\mathbf{S}\mathbf{T}^T)\mathbf{T}\|_2^2 \tag{9}$$

Note that for this S2T Variant, when we attend Student to Teacher instead, we have,

$$\mathcal{L}_{\text{AMAD}} := \|\mathbf{S} - \text{softmax}_{\text{dim=row}}(\mathbf{S}\mathbf{T}^T)\mathbf{T}\|_2^2 \tag{10}$$

$$= \sum_{j=1}^{H_s} \mathcal{L}_{\text{AMAD}_j} = \sum_{j=1}^{H_s} \left\| \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|_2} - \sum_{i=1}^{H_t} \tilde{a}_{ij} \cdot \frac{\mathbf{t}_i}{\|\mathbf{t}_i\|_2} \right\|_2^2 \tag{11}$$

where $\tilde{a}_{ij} := \frac{\exp(w_{ij})}{\sum_{m=1}^{H_t}\exp(w_{mj})} \neq \frac{\exp(w_{ij})}{\sum_{m=1}^{H_s}\exp(w_{im})} = a_{ij}$.

The difference is that, in this formulation of S2T Variant, the optimization w.r.t. AMAD loss might lead to a degraded trivial solution that all Student heads are mostly soft-aligned to a same small subset of Teacher heads (or even to one single Teacher head). This might happen, e.g., in this example toy case: for each Student head $j \in \{1, 2, \cdots, H_s\}$, $\tilde{a}_{1,j} \approx 1.0$ and $\tilde{a}_{i,j} \approx 0.0$ for $\forall i \neq 1$. In this case, only the co-reference knowledge of Teacher head #1 is inherited to the Students, the knowledge in all other heads is lost. Variant 1, instead, will not degrade into this solution, because we always have $\sum_{j=1}^{H_s} a_{ij} = 1, \forall i$. In our experiments, we have not observe severe degradation with S2T Variant when the Student is pre-trained, but do observe a small performance drop. We sometimes also observe more severe degradation in S2T Variant when the Student is not pre-trained and thus might not be well-initiated and falls into the ill-conditioned solution.

B. More Ablation Results

More Ablation Results. We present more ablation results in different settings here in Table 1-3.

A Reverse Experiment: Distilling a Smaller Teacher to a Larger Student. Previous studies show that, especially for self-distillation when Teacher and Student are identical models, the distilled Student sometimes outperforms the Teacher on test / validation data. (Mobahi et al. 2020) explained this mathematically by regularization in Hilbert space: the number of basis functions for representation is limited during distillation, thus may reduce over-fitting. We hope to understand this in future work. However, here we do conduct a **reverse** experiment: We distill a **small** Teacher (VL-T5 small) to a **larger** Student (VL-T5 base), using only logits KD (w/o proposed AMAD, w/o supervision from ground-truth labels), and we surprisingly find that, on VQA-2.0 validation set, the small Teacher achieves 67.43% accuracy, but the large Student achieves 68.27%, higher than the Teacher. We hypothesis that this might attribute to the larger scale pre-training of larger model.

BLIP Model	#Params	B	C	M	S
Teacher (Li et al. 2022)	446M	40.4	136.7	-	-
Student (Li et al. 2022)	210M	39.7	133.3	-	-
Student (reproduced)	210M	39.6	132.9	30.7	23.7
Student **w/ AMAD**	210M	40.0	134.1	31.0	23.9
Δ		**+0.4**	**+1.2**	**+0.3**	**+0.2**

Table 1: Results on COCO Captioning karpathy test split. All BLIP Students are first **VL pretrained**. Student then distilled with AMAD outperforms the GT fine-tuned Student.

VL-T5	#Params	$\mathcal{L}_{\text{KD}}$	$\mathcal{L}_{\text{AMAD}}$	B	C	M	S
Teacher	220M			34.2	115.1	28.3	21.6
Student	60M			30.8	101.4	26.3	19.5
Student	60M	✓		32.4	107.2	26.9	20.3
Student	60M	✓	✓	**33.3**	**112.9**	**28.0**	**21.3**

Table 2: Results on COCO Captioning karpathy test split. All VL-T5 Students are **w/o VL pretrain**. AMAD boosts the Student's performance by a large margin.

	VL-T5	#Params	B ↑	C ↑	M ↑	S ↑
1	Teacher	220M	34.2	115.1	28.3	21.6
2	Student (**AMAD** on final layers)	60M	33.9	114.4	**28.3**	**21.5**
3	Student (**AMAD** on extra layers)	60M	**34.0**	**114.5**	**28.3**	**21.5**

Table 3: Results on COCO Captioning karpathy test split. All VL-T5 Students are first **VL pretrained**. Here, in row 3, we apply AMAD on additional layers. Note that the Teacher has 12+12 layers, but the Student has 6+6, so we distill Teacher layer #2,4,6,8,10,12 to Student layer #1-6, respectively.

C. Implementation Details

We build our code upon the open-source repository of VL-T5 (Cho et al. 2021) and BLIP (Li et al. 2022, 2023), based on PyTorch (Paszke et al. 2017) and Hugging Face (Wolf et al. 2019).

VL-T5 Architecture. We directly use the code from VL-T5. Briefly, The Teacher and the Student take same image features and question text as input, and output answer classification over all possible answer categories for the VQA task, or auto-regressively generates a sentence for captioning and translation tasks. The VL-T5 (Cho et al. 2021) architecture consists frozen Faster R-CNNs (Ren et al. 2015) for image detection and a multi-modal T5 (Raffel et al. 2020) Encoder-Decoder stack. For VQA task, they build a 3129-way answer classifier head upon the *start-of-sequence* token of the T5 Decoder, and they name it as the "discriminative variant". The classifier is an MLP on top of the Decoder representation of a *start-of-sequence* token following (Cho et al. 2021; Tan and Bansal 2019; Chen et al. 2020). The MLP has one hidden layer with a dimension twice as the Transformer hidden dimension d_{model}. The MLP contains additional 6M / 3M learnable parameters for *base* and *small*, respectively. For captioning and translation, we use the original VL-T5 *generative* language modeling head. During inference, tokens are generated auto-regressively with beam search.

BLIP Architecture. We directly use the code of BLIP. Briefly, for the captioning task we test on, it contains a trainable ViT (Dosovitskiy et al. 2021) image-encoding transformer and a trainable BERT (Devlin et al. 2019) text module, with causal self-attention layers in replace of the bi-directional self-attention layers. BLIP is detector-free, and both the ViT large (Teacher) and ViT base (Student) take an image patch size of 16×16. Using a multi-modal pre-trained encoder model (Kim, Son, and Kim 2021; Ho et al. 2022) could provide additional improvements.

Vision-Language Pre-training for Teacher. The VL-T5 and BLIP authors released their VL-pre-trained VL-T5 *base* / BLIP *large* models and we directly use their pre-trained checkpoints for our Teacher.

Vision-Language Pre-training for VL-T5 Student. For those experiments where vision-language pre-training for the Student is involved, we VL pre-train our own *small* model, since the VL-T5 authors have not released their VL-pre-trained *small* model. We adopt the same hyper-parameters as in the open-source code of VL-T5.

In some of our ablative settings, to further demonstrate the strength of AMAD, we do not conduct any VL-pre-training on our Student models. After loading the Student Transformer from the Huggingface language pre-trained checkpoint and loading the vision-pre-trained Faster RCNN, we directly finetune the Student Transformer on downstream tasks with distillation loss only.

Vision-Language Pre-training for BLIP Student. The BLIP authors released their VL-pre-trained *base* model and we directly use their pre-trained *base* checkpoint for our Student.

Fine-tuning Teacher. We adopt the same hyper-parameters as in the open-source code of VL-T5 for COCO Captioning. For VQA, we use the discriminative variant of VL-T5, and we find that training with more epochs can bring better performance than what the VL-T5 paper reported. We train 60 epochs, and remain all the other hyper-parameters the same as in the open-source code of VL-T5. We also adopt the same hyper-parameters as in the open-source code of BLIP.

Fine-tuning Student with Distillation: VL-T5. For both downstream distilling on VQA-2.0 and COCO Captioning, it takes roughly 1-1.5 days for fine-tuning distillation with AMAD on 8 NVIDIA A100 GPUs. We use batch size 300 and 200 for VQA and Captioning, respectively. Beyer et al., 2022 (Beyer et al. 2022) shown with extensive experiments that a patient and consistent training schedule will benefit distillation training. Following them, we adopt a patient training schedule and train 500 epochs and 400 epochs for VQA and Captioning, respectively. We use AdamW (Loshchilov and Hutter 2019) optimizer with a learning rate of 1e-4 with 2% linear warm-up schedule, with clipping gradient norm of 5. We always use $\tau_d = 1.0$ for logits distillation temperature in Equation 13 and Equation 14. For inference stage on COCO Captioning, we auto-regressively predict each output token with the beam search size at 5. For α in Equation 15, we tune it so that $\mathcal{L}_{\text{CLS}}$ and $\alpha\mathcal{L}_{\text{AMAD}}$ are in similar scales when initiated. For our implementation as in the code below, we use $\alpha = 0.2$ for all MSE variants, and $\alpha = 100$ for all KL variants. The different in scales of α is caused by the specific implementation of reshaping and normalization, but all resulting in similar scales of $\mathcal{L}_{\text{CLS}}$ and $\alpha\mathcal{L}_{\text{AMAD}}$. All hyper-parameters are not heavily-tuned.

Fine-tuning Student with Distillation: BLIP. We use a batch size of 32, total epochs of 20 (60 for training BLIP with randomly initialized ViT and BERT in w/o any pretrain setting). We use the same optimizer settings as in BLIP: using initial learning rate 1e-5, with AdamW optimizer with a weight decay of 0.05 and a cosine learning rate schedule. We use an image resolution of 384×384. During inference, we adopt the same setting as BLIP: we use beam search with a beam size of 3, and set the maximum generation length as 20. We use the same τ_d and α as in distilling VL-T5, so that $\mathcal{L}_{\text{CLS}}$ and $\alpha\mathcal{L}_{\text{AMAD}}$ scales similarly.

Augmentation: like MixGen (Hao et al. 2023) was not applied to make the comparison correct. In our experiments, we found adding such augmentation to teacher could make the teacher robust to noise and thereby guide a student better.

D. Illustration of Workflow and Architecture

We provide an additional illustration of the distillation workflow for our experimental setup and the architecture of VL-T5 (Cho et al. 2021) in Figure 1 for readers not familiar with literature on VL-T5 and knowledge distillation. The pipeline of distilling the ViT and BERT modules of BLIP with AMAD is similar.

E. Visualization of Attention Maps

We present visualization of AMAD distilled VL-T5 (for VQA-2.0) Decoder cross-attention maps in Figure 2, Figure 3, Figure 4, and Figure 5. We also present visualization of AMAD distilled Encoder self-attention maps in Figure 6, Figure 7, Figure 8, and Figure 9.

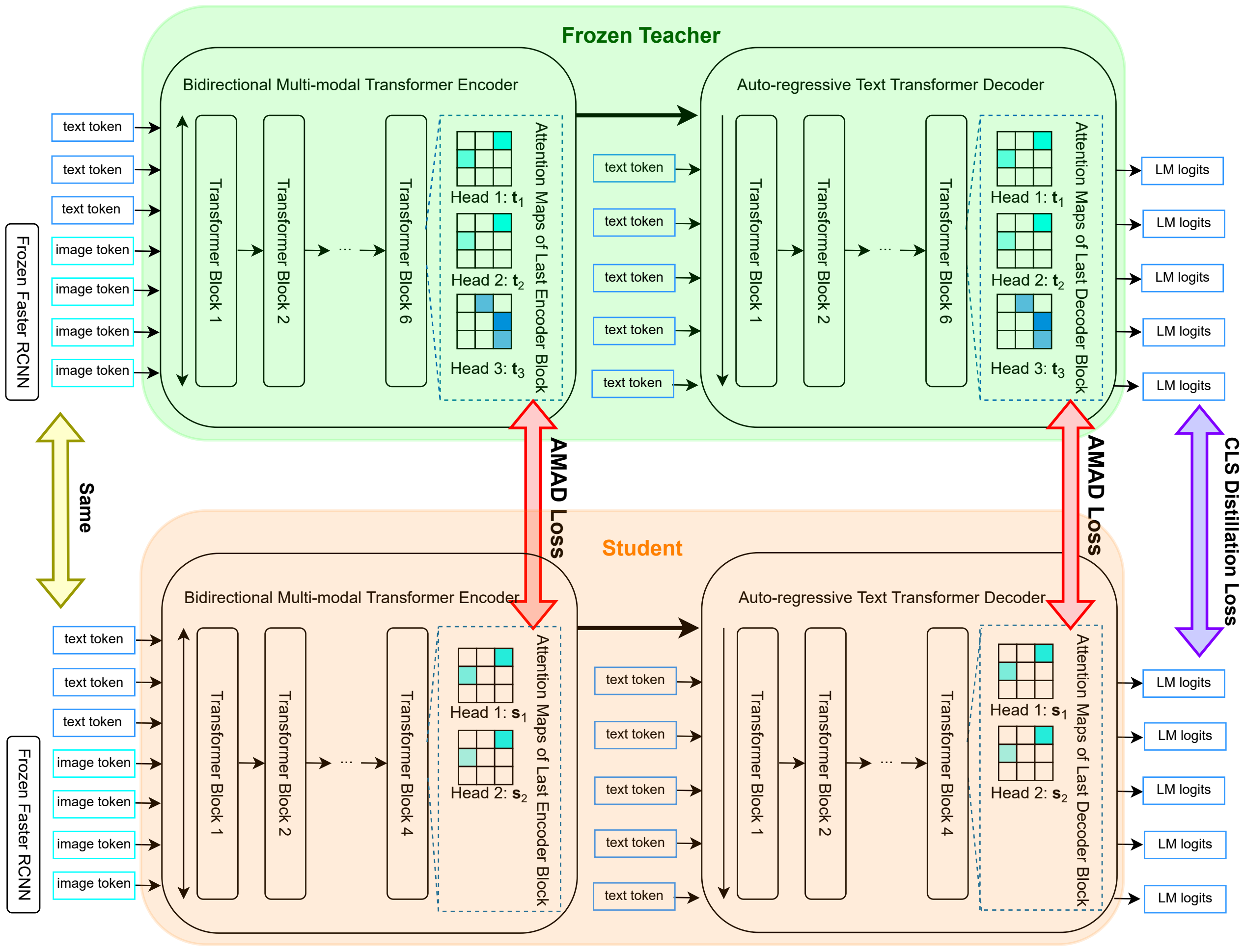


Figure 1: Illustration of the distillation workflow for our experimental setup and the architecture of VL-T5 (Cho et al. 2021), with logits classification (CLS) distillation loss $\mathcal{L}_{\text{KD}}$ and AMAD loss $\mathcal{L}_{\text{AMAD}}$.

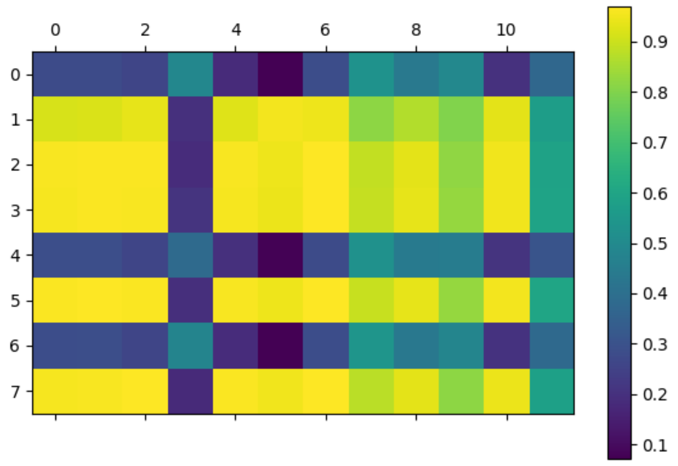
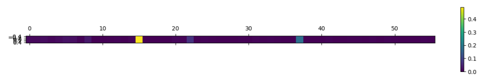
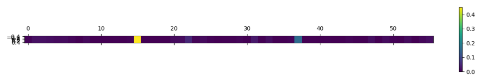
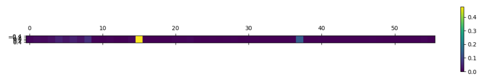
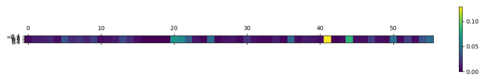


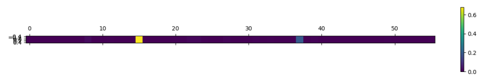
Figure 2: Visualization of the similarity matrix between each pair of Student and Teacher head attention maps in the last Decoder cross-attention layer in VQA: $\mathbf{W}^T \in \mathbb{R}^{H_s \times H_t}$, where $H_s = 8$ and $H_t = 12$ in this case. The visualization shows the similarity matrix $\mathbf{W}$ after distilled with AMAD, for a randomly-selected image-question pair sample. Each of its entry w_{ij} shows the cosine similarity between the i-th head attention map of AMAD-distilled Student $\mathbf{s}_i$ and the j-th head of the Teacher $\mathbf{t}_j$. Bright yellow represents higher similarity, and dark blue represents lower similarity. The corresponding 12 Teacher head attention maps and 8 Student head attention maps are shown in Figure 3, and Figure 4, respectively. We can see from this figure that after AMAD distillation, Student heads #1, #2, #3, #5, #7 are very similar to Teacher heads #0, #1, #2, #4, #5, #6, #8, #10 (cosine similarity > 0.8), and somewhat similar to Teacher heads #7, #9, #11 (cosine similarity > 0.5); While another group of Student heads, #0, #4, and #6, are quite similar to Teacher head #3, #7, #9 (cosine similarity > 0.5). This shows that with AMAD, different Student heads focus on learning from different soft-aligned Teacher heads. Each Student head inherits the co-reference knowledge from all Teacher heads, but different Student heads inherit more knowledge from different soft-aligned Teacher heads. Please also refer to Figure 3 and Figure 4 to see the detailed distilled effects for each specific head.

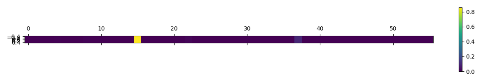

(a) Teacher Head 0

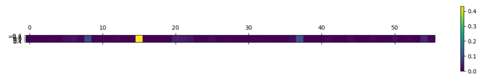

(b) Teacher Head 1

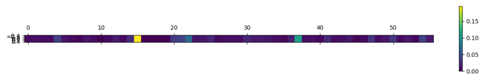

(c) Teacher Head 2

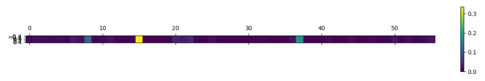

(d) Teacher Head 3

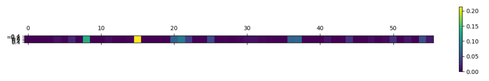

(e) Teacher Head 4

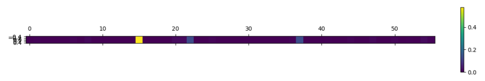

(f) Teacher Head 5


(g) Teacher Head 6


(h) Teacher Head 7


(i) Teacher Head 8


(j) Teacher Head 9


(k) Teacher Head 10

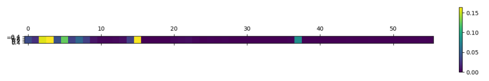
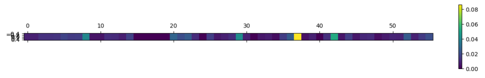
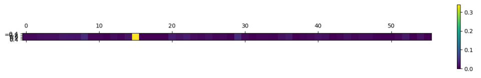

(l) Teacher Head 11

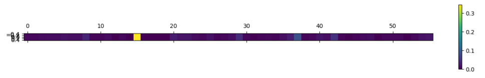
Figure 3: Visualization of the cross-attention maps of Teacher's last Decoder layer, for all its 12 heads. We are using the same image-question pair sample following Figure 2. Each cross attention map $\mathbf{t}_j$ is of shape $\in \mathbb{R}^{q \times k}$, where $q = 1$ in this VQA case, because we are using a 3129-way output classifier built on the Decoder *start-of-sequence* token. The attention maps can be divided into two similar groups, one group includes Teacher head #3; and the other group includes all other Teacher heads, featured by a strong attention to the 15-th token in bright yellow.



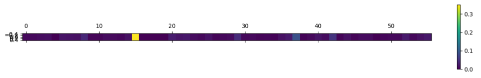
(a) Student Head 0



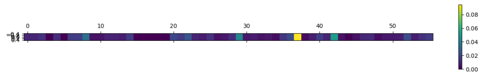
(b) Student Head 1



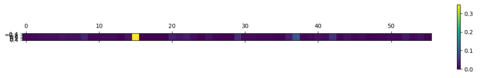
(c) Student Head 2



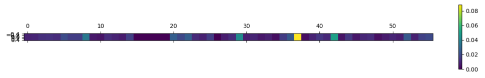
(d) Student Head 3



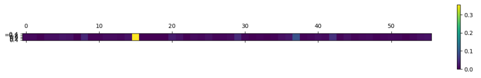
(e) Student Head 4



(f) Student Head 5

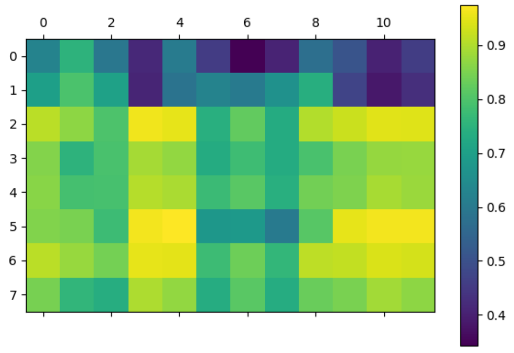


(g) Student Head 6

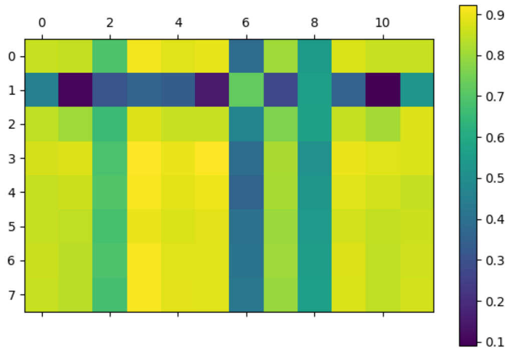


(h) Student Head 7

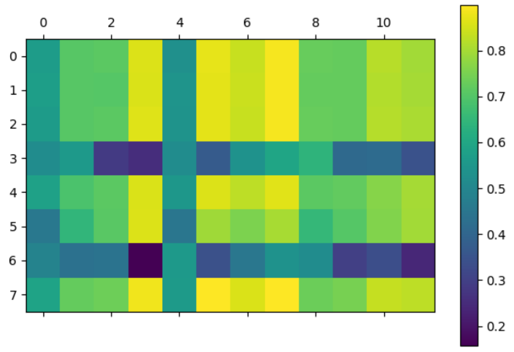
Figure 4: Visualization of the cross-attention maps of Student's last Decoder layer, for all its 8 heads. We are using the same image-question pair sample following Figure 2 and Figure 3. The attention maps of Student can also be divided into two similar groups, one group includes Student head #0, #4, #6, which are relatively similar to the first group of Teacher; and the other group includes the other Student heads, which are similar to the second group of Teacher heads, featured by a strong attention to the 15-th token in bright yellow.



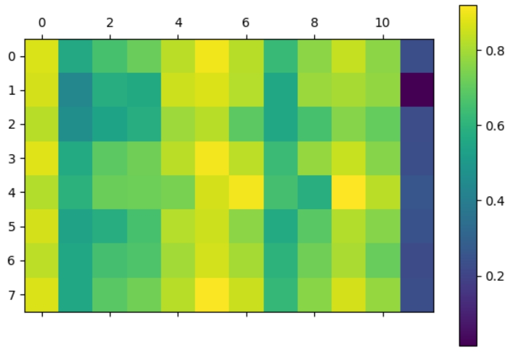
(a)



(b)



(c)



(d)

Figure 5: More visualizations for the similarity matrices between each pair of Student and Teacher head attention maps in the last Decoder cross-attention layer in VQA: $\mathbf{W}^T \in \mathbb{R}^{H_s \times H_t}$ for other image-question pair samples. In most cases (a, b, c), for each Teacher head, there are some Student head(s) similar to it, indicating that the co-reference knowledge in each Teacher head is inherited to the Student heads via distillation with AMAD to some extent. However, there are still some failure cases, e.g. in (d), none of the 8 Student heads is similar to the 11-th Teacher head.

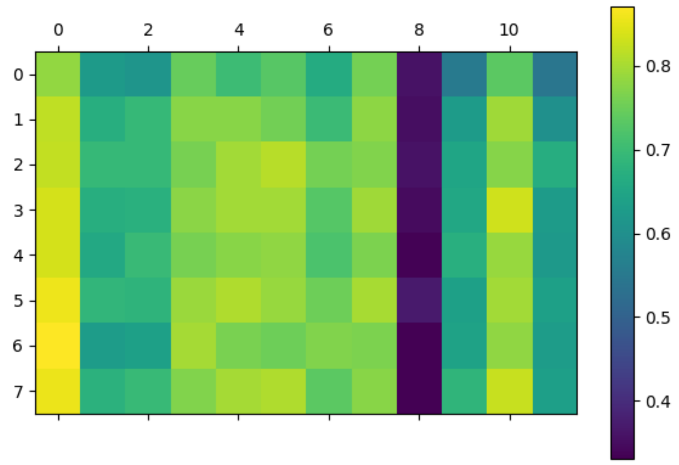
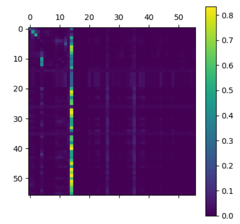
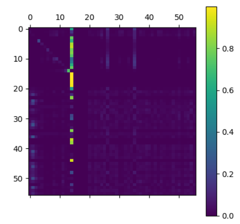


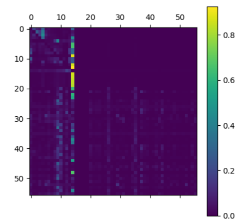
Figure 6: Visualization of the similarity matrix between each pair of Student and Teacher head attention maps in the last Encoder self-attention layer in VQA: $\mathbf{W}^T \in \mathbb{R}^{H_s \times H_t}$, where $H_s = 8$ and $H_t = 12$ in this case. The visualization shows the similarity matrix $\mathbf{W}$ after distilled with AMAD, for a randomly-selected image-question pair sample. Each of its entry w_{ij} shows the cosine similarity between the i-th head attention map of AMAD-distilled Student $\mathbf{s}_i$ and the j-th head of the Teacher $\mathbf{t}_j$. Bright yellow represents higher similarity, and dark blue represents lower similarity. The corresponding 12 Teacher head attention maps and 8 Student head attention maps are shown in Figure 7, and Figure 8, respectively. We can see from this figure that after AMAD distillation, for most Teacher heads, there are some soft-aligned Student heads similar to them. With AMAD, each Student head inherits the co-reference knowledge from all Teacher heads, but different Student heads inherit more knowledge from different soft-aligned Teacher heads. However, there is still a failure case as of Teacher head #8, for which none of the 8 Student heads is similar to it. Please also refer to Figure 7 and Figure 8 to see the detailed distilled effects for each specific head.



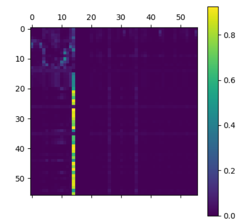
(a) Teacher Head 0



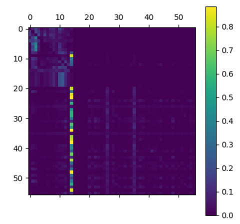
(b) Teacher Head 1



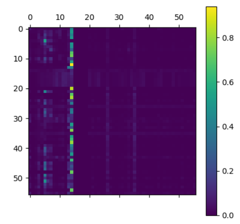
(c) Teacher Head 2



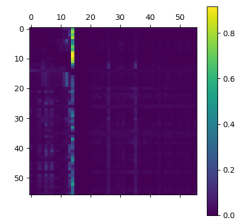
(d) Teacher Head 3



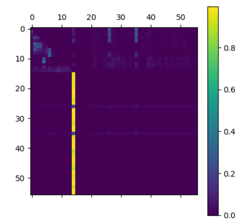
(e) Teacher Head 4



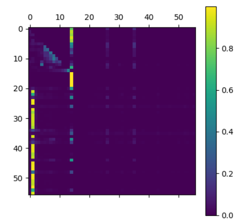
(f) Teacher Head 5



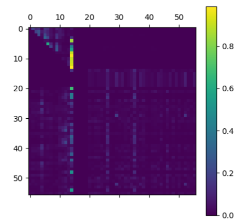
(g) Teacher Head 6



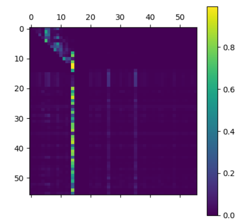
(h) Teacher Head 7



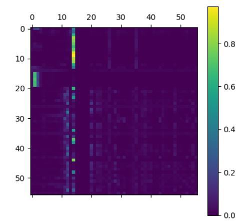
(i) Teacher Head 8



(j) Teacher Head 9

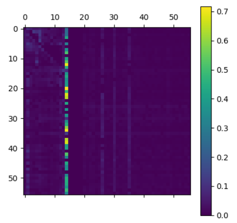


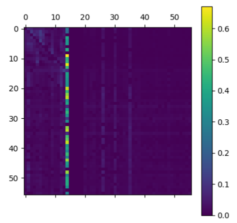
(k) Teacher Head 10

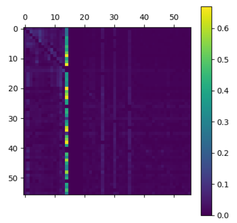


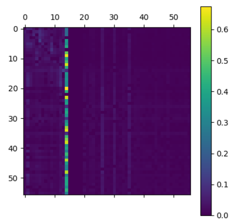
(l) Teacher Head 11

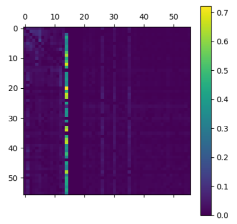
Figure 7: Visualization of the self-attention maps of Teacher's last Encoder layer, for all its 12 heads. We are using the same image-question pair sample following Figure 6. Each self attention map $\mathbf{t}_j$ is of shape $\in \mathbb{R}^{q \times k}$, where $q = k = 56$ in this self-attention case. Specifically, following VL-T5 (Cho et al. 2021), in the 56 tokens, the first (left / top) 20 tokens are question text tokens, and the last (right / bottom) 36 tokens are detected image tokens.

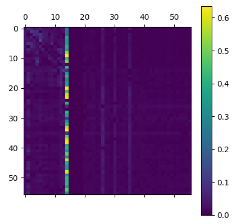

(a) Student Head 0

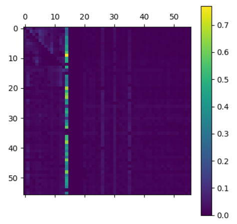

(b) Student Head 1


(c) Student Head 2


(d) Student Head 3


(e) Student Head 4


(f) Student Head 5


(g) Student Head 6

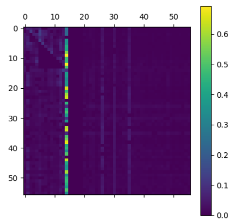
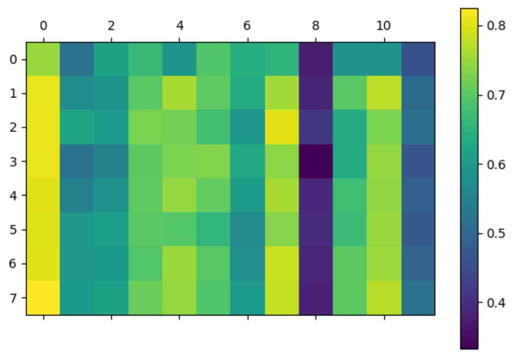
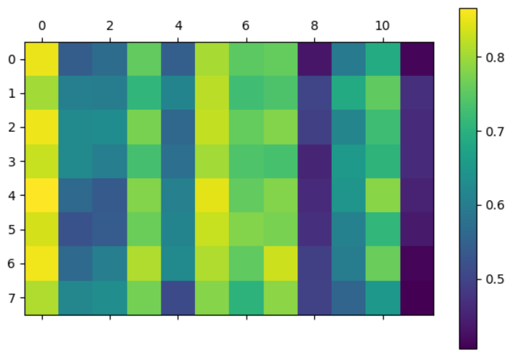

(h) Student Head 7

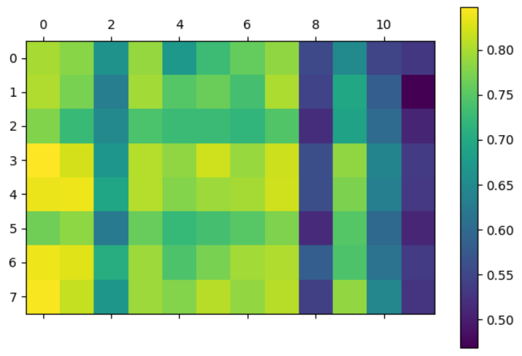
Figure 8: Visualization of the self-attention maps of Student's last Encoder layer, for all its 8 heads. We are using the same image-question pair sample following Figure 6 and Figure 7. Note the light yellow column of the 14-th token, the blue column of the 26-th and 35-th column, and the top-left corner of text intra-modal attentions, which are similar to those of the Teacher's attention maps.



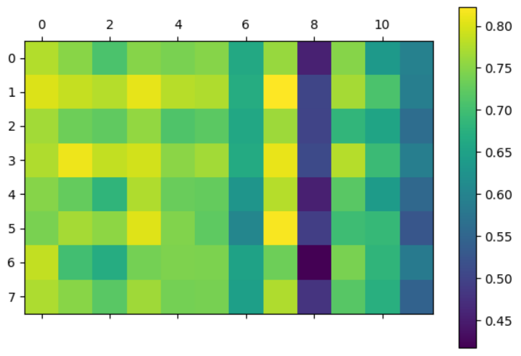
(a)



(b)



(c)



(d)

Figure 9: More visualizations for the similarity matrices between each pair of Student and Teacher head attention maps in the last Encoder self-attention layer in VQA: $\mathbf{W}^T$. For most of the Teacher heads, there are always some Student heads quite similar to it (light yellow or light green, cosine similarity > 0.7), indicating that their co-reference knowledge is inherited. However, the failure case is that Teacher head #8 always do not have Student heads very similar to it (max cosine similarity ≈ 0.5), and Teacher head #11 also have the similar problem, although slightly better in (a) and (d).

F. Code for AMAD Loss

PyTorch code for our implementation of the proposed AMAD loss is provided in the following pages.

```python
def attention_map_alignment_distillation_loss(teacher_attn, student_attn):
    # Arguments: teacher attention maps and student attention maps: last layer of Encoder / Decoder,
    # torch tensor of shape (batch_size, n_heads, seq_length, key_length);
    # Return: AMAD loss.

    (batch_size, n_heads_teacher, seq_length, key_length) = teacher_attn.shape
    n_heads_student = student_attn.shape[1]

    # Reshape
    teacher_attn = torch.reshape(teacher_attn, (batch_size, n_heads_teacher, seq_length * key_length))
    student_attn = torch.reshape(student_attn, (batch_size, n_heads_student, seq_length * key_length))

    # Compute cosine similarity: W = T S^t / norm(S) norm(T)
    teacher_attn = nn.functional.normalize(teacher_attn, dim=2)
    student_attn = nn.functional.normalize(student_attn, dim=2)
    student_attn_transpose = torch.transpose(student_attn, 1, 2)
    teacher_student_similarity_score = torch.matmul(teacher_attn, student_attn_transpose)

    # Compute Softmax(W) S by dim=2
    teacher_student_attention_weights = nn.functional.softmax(teacher_student_similarity_score, dim=2)
    weighted_sum_student_attn = torch.matmul(teacher_student_attention_weights , student_attn)

    # Normalize
    weighted_sum_student_attn = nn.functional.normalize(weighted_sum_student_attn, dim=2)

    # Compute loss = MSE(., T), differs from L2 error by a factor
    L2_loss = nn.MSELoss(reduction="mean")
    loss = L2_loss (input=weighted_sum_student_attn, target=teacher_attn)

    return loss

```

Figure 10: Variant 1.

```python
def attention_map_alignment_distillation_loss(teacher_attn, student_attn):
    # Arguments: teacher attention maps and student attention maps: last layer of Encoder / Decoder,
    # If provided with projected attention maps as arguments: Variant 5;
    # torch tensor of shape (batch_size, n_heads, seq_length, key_length);
    # Return: AMAD loss.

    (batch_size, n_heads_teacher, seq_length, key_length) = teacher_attn.shape
    n_heads_student = student_attn.shape[1]

    # Reshape
    teacher_attn = torch.reshape(teacher_attn, (batch_size, n_heads_teacher, seq_length * key_length))
    student_attn = torch.reshape(student_attn, (batch_size, n_heads_student, seq_length * key_length))

    # Compute similarity: W = T S^t / L1-norm(S) L1-norm(T)
    teacher_attn = nn.functional.normalize(teacher_attn, dim=2, p=1.0)
    student_attn = nn.functional.normalize(student_attn, dim=2, p=1.0)
    student_attn_transpose = torch.transpose(student_attn, 1, 2)
    teacher_student_similarity_score = torch.matmul(teacher_attn, student_attn_transpose)

    # Compute Softmax(A) S by dim=2
    teacher_student_attention_weights = nn.functional.softmax(teacher_student_similarity_score, dim=2)
    weighted_sum_student_attn = torch.matmul(teacher_student_attention_weights, student_attn)

    # Reshape for KL + normalize
    teacher_attn = torch.reshape(teacher_attn, (batch_size * n_heads_teacher * seq_length, key_length))
    weighted_sum_student_attn = torch.reshape(weighted_sum_student_attn, \
        (batch_size * n_heads_teacher * seq_length, key_length))
        # both reshaping to (batch_size, n_heads_teacher * seq_length * key_length) is also correct,
        # but will result in a linear scale different by n_heads_teacher * seq_length
        # since p/C log ((p/C) / (q/C)) = 1/C p log (p / q)

    # Normalize by L1
    teacher_attn = nn.functional.normalize(teacher_attn, dim=1, p=1.0) # redundant in this variant
    weighted_sum_student_attn = nn.functional.normalize(weighted_sum_student_attn, dim=1, p=1.0)

    # Compute loss
    eps = 1e-7
    weighted_sum_student_attn = torch.log(eps + weighted_sum_student_attn)
    KLD_loss = nn.KLDivLoss(reduction="mean")
    loss = KLD_loss(input=weighted_sum_student_attn, target=eps + teacher_attn)

    return loss

```

Figure 11: Variant 2.

```python
def attention_map_alignment_distillation_loss(teacher_attn, student_attn):
    # Arguments: teacher attention maps and student attention maps: last layer of Encoder / Decoder,
    # torch tensor of shape (batch_size, n_heads, seq_length, key_length);
    # Return: AMAD loss.

    (batch_size, n_heads_teacher, seq_length, key_length) = teacher_attn.shape
    n_heads_student = student_attn.shape[1]

    # Reshape
    teacher_attn = torch.transpose(teacher_attn, 1, 2)
    student_attn = torch.transpose(student_attn, 1, 2)
        # now: (batch_size, seq_length, n_heads, key_length)

    teacher_attn = torch.reshape(teacher_attn, (batch_size * seq_length, n_heads_teacher, key_length))
    student_attn = torch.reshape(student_attn, (batch_size * seq_length, n_heads_student, key_length))

    # Compute similarity with L1-normalization (redundant in this variant): W = T S^t / norm(S) norm(T);
    teacher_attn = nn.functional.normalize(teacher_attn, dim=2, p=1.0)
    student_attn = nn.functional.normalize(student_attn, dim=2, p=1.0)
    student_attn_transpose = torch.transpose(student_attn, 1, 2)
    teacher_student_similarity_score = torch.matmul(teacher_attn, student_attn_transpose)

    # Compute Softmax(W) S by dim=2
    teacher_student_attention_weights = nn.functional.softmax(teacher_student_similarity_score, dim=2)
    weighted_sum_student_attn = torch.matmul(teacher_student_attention_weights, student_attn)

    # Reshape for KL + normalize
    teacher_attn = torch.reshape(teacher_attn, (batch_size * seq_length * n_heads_teacher, key_length))
    weighted_sum_student_attn = torch.reshape(weighted_sum_student_attn, (batch_size * seq_length * n_heads_teacher, key_length))

    # Normalize by L1
    teacher_attn = nn.functional.normalize(teacher_attn, dim=1, p=1.0) # redundant in this variant
    weighted_sum_student_attn = nn.functional.normalize(weighted_sum_student_attn, dim=1, p=1.0)

    # Compute loss
    eps = 1e-7
    weighted_sum_student_attn = torch.log(eps + weighted_sum_student_attn)
    KLD_loss = nn.KLDivLoss(reduction="mean")
        # reduction="sum" is also correct, but will differ by a factor
    loss = KLD_loss(input=weighted_sum_student_attn, target=eps + teacher_attn)

    return loss

```
Figure 12: Variant 4.

References

Beyer, L.; Zhai, X.; Royer, A.; Markeeva, L.; Anil, R.; and Kolesnikov, A. 2022. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF Conference on CVPR*, 10925–10934. 3

Chen, Y.-c.; Li, L.; Yu, L.; Kholy, A. E.; Ahmed, F.; Gan, Z.; Cheng, Y.; and Liu, J. 2020. UNITER: UNiversal Image-TExt Representation Learning. In *ECCV*. 3

Cho, J.; Lei, J.; Tan, H.; and Bansal, M. 2021. Unifying Vision-and-Language Tasks via Text Generation. In *ICML*. 2, 3, 4, 10

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *North American Chapter of the Association for Computational Linguistics*. 3

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR*. 3

Hao, X.; Zhu, Y.; Appalaraju, S.; Zhang, A.; Zhang, W.; Li, B.; and Li, M. 2023. Mixgen: A new multi-modal data augmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 379–389. 3

Ho, C.-H.; Appalaraju, S.; Jasani, B.; Manmatha, R.; and Vasconcelos, N. 2022. Yoro-lightweight end to end visual grounding. In *European Conference on Computer Vision*, 3–23. Springer. 3

Kim, W.; Son, B.; and Kim, I. 2021. ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision. In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 5583–5594. PMLR. 3

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *ICML*. 2

Li, J.; Li, D.; Xiong, C.; and Hoi, S. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *ICML*. 2

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *ICLR*. 3

Mobahi, H. et al. 2020. Self-distillation amplifies regularization in hilbert space. *NeurIPS*. 2

Paszke, A.; Gross, S.; Chintala, S.; Chana, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch. In *NIPS Workshop*. 3

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *JMLR*, 21: 1–67. 3

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*. 3

Tan, H.; and Bansal, M. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *EMNLP*. 3

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; and Brew, J. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. 3