eP-ALM: Efficient Perceptual Augmentation of Language Models
Mustafa Shukor, Corentin Dancette, Matthieu Cord

To cite this version:

HAL Id: hal-04232603
https://hal.science/hal-04232603
Submitted on 8 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Abstract

Large Language Models (LLMs) have so far impressed the world, with unprecedented capabilities that emerge in models at large scales. On the vision side, transformer models (i.e., ViT) are following the same trend, achieving the best performance on challenging benchmarks. With the abundance of such unimodal models, a natural question arises: do we need also to follow this trend to tackle multimodal tasks? In this work, we propose to rather direct effort to efficient adaptations of existing models, and propose to augment Language Models with perception. Existing approaches for adapting pretrained models for vision-language tasks still rely on several key components that hinder their efficiency. In particular, they still train a large number of parameters, rely on large multimodal pretraining, use encoders (e.g., CLIP) trained on huge image-text datasets, and add significant inference overhead. In addition, most of these approaches have focused on Zero-Shot and In Context Learning, with little to no effort on direct finetuning. We investigate the minimal computational effort needed to adapt unimodal models for multimodal tasks and propose a new challenging setup, alongside different approaches, that efficiently adapts unimodal pretrained models. We show that by freezing more than 99% of total parameters, training only one linear projection layer, and prepending only one trainable token, our approach (dubbed eP-ALM) significantly outperforms other baselines on VQA and Captioning across Image, Video, and Audio modalities, following the proposed setup. The code will be available here: https://github.com/mshukor/eP-ALM.

1. Introduction

Going large scale has led to outstanding performances that consistently improve across tasks, modalities, and domains on current benchmarks. Most of the progress so far has been in the vision and language domains. For Computer Vision, the ViT family [21] starts from the tiny model with 5M parameters to the enormous ViT-e [15] with 4B parameters and the largest ViT-22B with 22B parameters [20]. More captivating, are the scales of Large Language Models (LLMs), such as the BLOOM [76] and OPT [107] families, ranging from hundreds of millions of parameters to 175B, in addition to other models that go beyond 100B [7, 19, 83] up to 1T parameters [27]. These huge scales come with a need for very large pretraining datasets and long training times.

The current prevalent paradigm to solve multimodal tasks, in particular, Vision-Language tasks is to leverage pretrained models, and then further train end-to-end [15, 56, 79, 81, 94] on large image-text datasets. However, the training cost is huge and unaffordable for a large portion of the community, as these approaches still train all model parameters, even after initialization, on a huge amount of data.

With the abundance of unimodal models, a natural question arises: Do we need also to follow this trend to tackle multimodal tasks? or rather direct effort to efficient adaptations of existing models?
Drawing inspiration from the recent work in Augmented Language Models (ALMs) [70], in this paper, we advocate for adapting pretrained LMs to solve multimodal tasks. Specifically, by augmenting LMs with perceptual encoders.

Several approaches have deviated from the end-to-end-training paradigm by freezing some pretrained modules and training only the adaptation parameters, such as, additional cross-attention [3], vision encoder [90] and Adapters [23].

Even though these approaches have taken a big step towards more parameter-efficient models, there are still many costly components that hinder their adoption by the large community, such as the training and inference memory and time cost.

In this work we argue that current approaches are far from optimal and it is possible to find more efficient approaches, in terms of the number of trainable parameters, training data, and compute, to adapt pretrained unimodal models for multimodal tasks. A better alignment of visual and language representations might help to devise extremely efficient adaptation approaches.

To investigate this hypothesis, we go a step further to efficiently leverage LLMs, and propose (1) a new technique to adapt unimodal models by freezing more than 99% (up to 99.94%) of their parameters, alongside (2) a minimal and challenging setup to adapt pretrained unimodal models for Image/Video/Audio-Language tasks (e.g., VQA [33, 97], Image and Audio Captioning [14, 48]). In this setup, we favor unimodal-only models, avoiding multimodal pretraining or massively trained multimodal encoders, and considering the typical LLMs architecture as the backbone. All that while freezing as much as possible of model parameters. The approach is illustrated in Fig. 1.

Specifically, we adopt the publicly released OPT model [107] and unimodal encoders (e.g., ViT, TimeSformer [6], AST [31]), which are kept frozen. We finetune directly the adaptation parameters on publicly available benchmarks of downstream tasks such as for VQA, GQA, Image Captioning, Video QA, Video Captioning, and Audio Captioning.

Based on this setup we investigate different design choices and propose very efficient approaches backed by the following interesting findings:

- Training a single linear layer directly on downstream multimodal datasets, and following the same setup, outperforms other work on Image/Video/Audio-Language tasks. With a few additional trainable parameters and a single learned prepended token, we can significantly improve the performance, while respecting a budget of 1% of trainable parameters, and keeping almost the same inference cost.

- Our approach enjoys better generalization (OOD, Zero-Shot) and is data-efficient (training on 1% of the data achieves 80% of performances) with better few-shot results than other approaches.

- While reaching good performance with small to mid-scale language models (i.e., 350M-2.7B) the improvement still increases by jointly scaling both vision and language models. When scaling both models, we can still outperform other approaches with only 0.06% of trainable parameters.

- Existing approaches do not behave well on the proposed challenging setup, without large multi-modal pretraining.

2. Related Work

Vision-Language Models (VLMs). Previously, vision-language tasks have been solved with models heavily customized for the particular task at hand [8, 26, 41, 44, 47]. The success in Self Supervised Learning [9, 34, 36, 87, 96] and the importance of good initialization have pushed researchers to transfer these ideas to VLMs and started Vision-Language Pretraining (VLP) on large scale video-text [28, 57, 92], image-text datasets in general domains [16, 51, 55, 56, 65, 79], as well as specific domains, such as Cooking [80], Medical Images [71] and Event Extraction [58]. VLP is a step to move away from the burden of customization by having one pretrained model, exploited for several downstream tasks. Recently, we have witnessed impressive work that go a step further towards more unification, by unifying the model, the training objective, and input-output format [15, 66, 93, 94]. All these models train most of the model parameters, even after initialization, which becomes more and more costly with the current trend in scaling data, model size, and compute [15, 103]. Another approach for VLM is to exploit existing pretrained models by keeping them frozen and training only the adaptation parameters [3, 23, 60]. This work advocates for the latter favoring training efficiency in terms of memory and time.

Adapting Language Models. Large Language Models (LLMs) [7, 19, 38, 76, 83, 107] have impressed the world in this last few years, showing unprecedented performance on a myriad of NLP tasks. Scaling LLMs to hundreds of billions of parameters has been motivated by the capabilities that surprisingly emerge [95] at this scale and lead to sudden jumps of relevant metrics on hard downstream tasks [37, 74, 84]. This generalization ability pushed researchers to start adapting these models for other modalities [3, 90], tasks [35, 88, 102, 106] and domains [82]. Currently, most of the focus is concentrated on exploiting LLMs for vision-language tasks, such as Flamingo [3] which trains 10B parameters to adapt a frozen 70B parameter language model, and other successful efficient techniques that are based on vision-conditioned prompt tuning (Frozen [90], Prompt-Fuse [60], LiMBeR [69]) and adapters (MAGMA [23]).
work has demonstrated good performance, showing that it is possible to devise very efficient approaches to adapt existing language models [38, 91]. On the video side, little work has been proposed, mostly based on Adapters [78, 101]. The closest to our approach is PromptFuse [60] which finetunes directly for VQA, however, they use encoder-decoder language models and train a soft prompt that is prepended to the input.

**Efficient Learning.** Parameter-Efficient learning is an interesting line of research that consists of adapting pretrained models using very few trainable parameters. Prompt Tuning [54] is one such approach that appends a few learnable tokens, or Soft Prompts to contextualize the input and steer the output of the frozen model toward the desired task. Other approaches use Adapters [4, 39], which are trainable MLP, consisting of 2 linear projection layers with activation in between and inserted inside the model to adapt the self-attention and feedforward layers. Many other approaches have been proposed in the context of NLP such as LoRa [40], Bitfit [104], Hyperformer [67], Compacters [46] and (IA)3 [62]. These approaches have been successfully adapted to other modalities such as image [13, 43], image-text [85, 86, 108], with very little work on video [72] and Audio [50].

Another line of research is Data-Efficient techniques, where the objective is to attain similar performance by significantly reducing the training datasets. Recently, some efforts have been proposed for vision [89], language [22] and vision-language [12, 17, 79], which mostly focus on designing better training objectives [79]. However, little work has been done to investigate the connection between parameter efficiency and data efficiency, which is considered in this work.

### 3. Framework

To solve multimodal tasks, we propose to augment pretrained LLMs with perception through unimodal perceptual encoders (Fig.1). We detail our approach in the following.

#### 3.1. eP-ALM

We augment a pretrained LM with perception through several modality-specific encoders. The encoders interact with LM through linearly projected, modality-specific [CLS] tokens. To ease the adaptation, we leverage some parameter-efficient techniques, such as Prompt Tuning. In this section, we detail the design principles of our approach, which is illustrated in Fig.2.

**Language Model (LM)** We adopt OPT models [107], which are autoregressive language decoders consisting of Self-Attention and Feed Forward layers. They are trained with *next token prediction* objective on 180B tokens mostly in English and gathered from different datasets [5, 29]. The authors released a family of models with different scales, starting from 125M up to 175B model size. Besides being open source and trained on English data, the different model sizes allow us to readily investigate the effect of scale, and help to devise new approaches with affordable model sizes.

**Perceptual Encoders** We favor only unimodal models. For images, we use the vanilla ViT model [21] which consists of Self Attention and FeedForward layers and is pretrained for image classification on ImageNet [75]. For Video, we use TimeSformer [6] that consists of a ViT-like model augmented with temporal attention and pretrained on kinetics [10]. For Audio, we adopt AST [31], a vanilla adaptation of ViT to digest spectrograms, that is pretrained on AudioSet [30]. Even though we consider only these 3 encoders, the extension of the approach to other types of encoders and modalities is straightforward.

**Perceptual Prompt Injection.** LMs are usually controlled via different textual prompts, such as questions and instructions. Here, the LM is controlled by both the text and the perceptual encoders. Specifically, the projected perceptual tokens are prepended to the textual tokens. Naively using all visual tokens, adds significant computation costs during training and inference, due to the quadratic complexity of attention layers with the number of tokens. This becomes
Connecting Models with Cross-Modal Hierarchical Linear layers. When freezing the perceptual encoders and language models, the minimal number of trainable parameters are those that amount to connecting these two models while adjusting the embedding dimensions in case of a mismatch. Therefore, we base our approach on this constraint and train only one linear projection layer (single connection, Fig. 2) to connect both models. To exploit the hierarchical representation encoded in pretrained models, instead of taking only the [CLS] token of the last output layer, we take the [CLS] tokens from several layers of the perceptual model, and we inject these tokens into several layers of the LM (shared connection). The tokens coming from early layers are injected earlier and are then replaced by those coming from deeper layers. We favor only the deeper layers (e.g., the last 6 layers of the ViT-B/16, and the last 12 layers of OPT-350M) where the representations are more abstract and less modality-specific. Moreover, using the same linear projection at different representation levels might not help to capture the particularity of such a hierarchy, to this end, we also experiment with different linear layers for each level (multiple connections).

Multimodal Adaptation with Parameter-Efficient Techniques. We explore several parameter-efficient techniques to ease the adaptation to multimodal tasks. The main technique we use is Prompt Tuning [54]: it consists of prepending trainable tokens or Soft Prompts to the textual tokens input of the LM. This gives useful context to steer the model output. Contrary to hard prompts that are manually engineered, this provides a more flexible and easier approach for task-dependant contextualization. For the sake of efficiency, we prepend only 10 learnable tokens. We also experiment Adapters [39] as detailed later. The approach can be formalized as follows (better read with Fig. 2):

\[
\begin{align*}
[CLS]_i &= C(E_i(X)), \\
\text{for } i &= N_E/2, \ldots, N_E, \\
t_j &= LM_j([CLS]_i, t_{j-1}, t_{j-2}), \\
\text{for } j &= N_L/2, \ldots, N_L, \\
(1)
\end{align*}
\]

where \([CLS]_i\) is the perceptual token of the input \(X\) extracted from the layer \(i\) of the perceptual encoder \(E_i\) with \(N_E\) layers. \([CLS]_i\) is projected using the linear connection \(C\) and prepended, alongside the Soft Prompt \(t\) to the embeddings of the textual tokens \(t_{j-1}\) coming from previous layer in the LM \(LM_{j-1}\). This operation is repeated each 2 layers in the LM (with \(N_L\) layers).

3.2. Efficiency-driven Training Framework Setup

Current approaches still rely on many costly components that hinder their adoption by the large community. Specifically: (1) still train a lot of parameters (e.g. vision encoders [90] and adapters [23] with \(\sim 325\)M params/5.11%), (2) still maintain the multimodal pretraining with image-text pair datasets on top of the unimodal pretraining [23, 69, 90], (3) leverage multimodal encoders such as CLIP, pretrained on 400M image-text pairs [23, 69]. (4) add significant computation overhead during inference, due to the long visual prompt, especially when evaluating with In Context Learning (ICL), that becomes common with LLMs [23, 69]. In this work, we propose a new setup to adapt unimodal models for multimodal downstream tasks. The setup is more challenging and is motivated by the quest for the least effort needed to exploit pretrained models. The setup is the following:

- Training only adaptation parameters (e.g., Soft Prompt, linear connection), while keeping as much as possible of pretrained parameters frozen (parameter efficient).
- Avoiding multimodal pretraining and finetuning directly on downstream multimodal datasets (data/compute efficient).
- Using only pretrained unimodal models, and avoid using multimodal encoders pretrained on huge datasets (data efficient).
- Keeping fast inference (e.g., 1 additional token), by avoiding long prompts, and using additional heavy modules (compute efficient).
- Using decoder-only language models (e.g., OPT), the current architecture adopted by LLMs (due to its pretraining efficiency and open-ended generation capacity).

Specifically, we train only the linear connection and the soft prompt directly on the downstream multimodal tasks. This amounts to less than 1% of trainable parameters that we can push further to 0.06% with big models.

The Pretrain Zero-shot Setup. The focus of this work is direct finetuning on target datasets. However, the proposed mechanism (Sec.3.1) can be adapted straightforwardly to the pretrain-zeroshot setup. In the appendix, we show that eP-ALM outperforms previous work and it is competitive with recent SoTA following the zero-shot evaluation.

4. Experiments

Implementation details. We use OPT-2.7B in our main model, eP-ALM, and we experiment in Section 4.2 with OPT models of various sizes. We extract the [CLS] tokens of the last 6 layers of perceptual encoders and prepend them, after
a linear projection, to the text tokens of the last 12 layers of the OPT. Note that we replace the previous [CLS] with the new one to keep the same number of tokens.

For VQA and VideoQA, we cast the problem as open-ended generation and compute the accuracy after a strict comparison between the output text (without truncation) and the ground truth one. Note that this setting is more challenging compared to classification-based VQA and not in favor of our approach as the model might generate semantically correct answers but using different words. We use a special token (‘</a>’) to separate the question from the answer. For captioning, we report the widely adopted CIDEr and BLUE@4 scores. We finetune with the classical cross-entropy loss used to train the original OPT for VQA and Captioning tasks. We use the AdamW optimizer with a learning rate (lr) of 1e-5 warmed up to 2e-5 then decreased to 1e-6 using a cosine scheduler. We train for 8 epochs with a batch size of 64 (128 for GQA) and an image resolution of 224. Training our approach with OPT-2.7B for VQA v2 can be done on a single V100 GPU 32GB for few hours. More details are given in the appendix. We find the method sensitive to the text decoding approach (Tab. 8). Following other work, we use greedy decoding with beam search for the main results (Sec. 4.1), and multinomial/random sampling for the ablation study (Sec. 4.2).

**eP-ALM Variants.** Our main model, eP-ALM (illustrated in Figure 2), has multiple linear connections; specific learned linear layers for each [CLS] token injected in the model. In addition to Prompt Tuning. We also test variants of this model: eP-ALM$_{ada}$ (eP-ALM with Adapters instead of Soft Prompts), eP-ALM$_{lin}$ (trains a shared linear connection with all [CLS] tokens, and no prompt tuning) and eP-ALM$_{pt}$ (Lin + Soft Prompt). For Adapters, we follow other work [23] and add sequentially one adapter module after self-attention and feedforward layers in all the blocks of OPT. While this might give better results, it adds a significant number of trainable parameters.

### 4.1. Main Results

In this section, we present the main comparison with other approaches. We present the results for the image modality in Section 4.1.1, the video modality in Section 4.1.2, and the audio modality in Section 4.1.3.

#### 4.1.1 Image-Text Results

We use a frozen ViT-B/16 pretrained on ImageNet1K as the image encoder. We consider the following image-text benchmarks; VQA v2 [33], GQA [42] and COCO Caption [14]. We use Karpathy splits for VQA v2 and COCO, unless specified otherwise.

**Baselines.** As we are the first to propose this setup, to have a fair comparison, we reimplemented some of the existing approaches and use the same vision (ViT-ImageNet) and language (OPT) models for all:

1) $B_{\text{PromptFuse}}$: which is equivalent to PromptFuse [60] and uses Prompt Tuning (N=10). We add a linear projection for the last [CLS] token. The [CLS] token is prepended to the input of the LM. Note that we could not avoid adding a trained linear projection as there is a mismatch between the dimensions of the vision and language model.

2) $B_{\text{MAGMA}}$: which is equivalent to MAGMA [23] and uses Adapters. We prepend the [CLS] token to the input of LM after linear projection. Note that, we consider only the [CLS] token as we find it better than prepending all image tokens (eP-ALM$_{pt}$). We also find that training the ViT degrades the performance, thus we keep it frozen in favor of their approach.

3) $B_{\text{LimBEr}}$: which is equivalent to LimBEr [69] and only trains the linear projection to project visual tokens and prepend them to the input text. Similarly, we only consider the [CLS] token as it gives better accuracy.

**Comparison to Other Work.** Based on our study (Sec. 4.2), we use ViT-B/16 and OPT-2.7B in our main model and in our replication of other approaches. In Table 1 we compare with other work on VQA v2, GQA, and COCO Caption. We significantly outperform other approaches with at least +10 points on VQA v2, +9 points on GQA and we double the scores on COCO Caption. eP-ALM$_{pt}$ with OPT-6.7B and ViT-L gives the best scores while training only 0.06% of model parameters.

Note that for COCO Caption, other works give very low scores (thus we did not report them).

<table>
<thead>
<tr>
<th>Method</th>
<th>VQA v2</th>
<th>GQA</th>
<th>COCO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Val</td>
<td>Test</td>
<td>Val</td>
</tr>
<tr>
<td>PromptFuse$^*$ [60]</td>
<td>34.1†</td>
<td>–</td>
<td>30.81</td>
</tr>
<tr>
<td>$B_{\text{LimBEr}}$</td>
<td>34.1</td>
<td>33.5</td>
<td>30.81</td>
</tr>
<tr>
<td>$B_{\text{PromptFuse}}$</td>
<td>40.4</td>
<td>39.5</td>
<td>33.74</td>
</tr>
<tr>
<td>$B_{\text{MAGMA}}$</td>
<td>32.2</td>
<td>31.8</td>
<td>30.98</td>
</tr>
<tr>
<td>eP-ALM$_{pt}$</td>
<td>48.8</td>
<td>47.8</td>
<td>43.8</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>50.7/53.9†</td>
<td>50.2</td>
<td>45.0</td>
</tr>
<tr>
<td>eP-ALM$_{pt}$-L$^*$</td>
<td>54.58/54.47†</td>
<td>54.47</td>
<td>46.86</td>
</tr>
</tbody>
</table>

Table 1. Comparison with other work after direct finetuning on VQA v2, GQA, and COCO Caption. eP-ALM significantly outperforms other approaches. eP-ALM uses ViT-B/16 and OPT-2.7B. eP-ALM-L uses OPT-6.7B and ViT-L/16. †: use standard split. †: trained more than 8 epochs.

**Few-shot Results: Are Parameter-Efficient Models also Data-Efficient?** In this section, we investigate how data-efficient our model can be. To this end, we train on a very small portion (randomly sampled) from the VQA training set...
and evaluate on the validation set. Table 2, shows the superiority of our approach over other baselines. Interestingly, we can achieve 80% (41.9 vs 52.77) of the performance when training on 1% of the data. This validates the approach on low resources scenarios and shows that, in addition to being parameter-efficient, our model is also data-efficient.

<table>
<thead>
<tr>
<th>Method</th>
<th>Train. data % (# of shots)</th>
<th>VQA v2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PromptFuse</td>
<td>0.12% (512)</td>
<td>29.40</td>
</tr>
<tr>
<td>BLimBEr*</td>
<td>1% (4.4K)</td>
<td>28.9</td>
</tr>
<tr>
<td>B_MAGMA</td>
<td>1% (4.4K)</td>
<td>31.9</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.12% (512)</td>
<td>30.36</td>
</tr>
<tr>
<td>eP-ALM*</td>
<td>1% (4.4K)</td>
<td>35.48</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.12% (44K)</td>
<td>47.4</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>100% (443K)</td>
<td>52.77</td>
</tr>
</tbody>
</table>

Table 2. Few-shot Results on VQA v2 validation set (standard split). *: longer training.

### Out of Distribution (OOD) Generalization: Do Parameter-Efficient Models Generalize Better?

Here we investigate whether our parameter-efficient approach can perform well in OOD scenarios. To this end, we follow other approaches [1] and train our model on the training set of a given benchmark, and evaluate it on the validation set of another benchmark, without multimodal pretraining. We measure the performance gap, i.e. the accuracy difference between a model trained on a different benchmark and the same model trained on the target benchmark. Tab.3 shows that eP-ALM, that trains 0.06% of total parameters, is very competitive in terms of OOD accuracy with other baselines, that train all model parameters and pretrain on large amount of data. Specifically, we outperform VILBERT on VQAv2 by more than 2 points. Interestingly, the OOD-IID gap for eP-ALM, is at least 2 times lower compared to ALBEF [56] and VilBERT [65]. This reveals that our parameter-efficient approach generalizes relatively well in OOD scenarios.

<table>
<thead>
<tr>
<th>Method</th>
<th>Multimodal PT data</th>
<th>Trained param. (%)</th>
<th>Train data</th>
<th>Test data</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALBEF [1]</td>
<td>14M</td>
<td>100%</td>
<td>QVA v2</td>
<td>GQA</td>
<td>-21.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50.3</td>
<td>-</td>
<td>-14.1</td>
</tr>
<tr>
<td>VILBERT [1]</td>
<td>3M</td>
<td>100%</td>
<td>QVA v2</td>
<td>GQA</td>
<td>-42.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>41.8</td>
<td>-</td>
<td>-22.7</td>
</tr>
<tr>
<td>eP-ALM*</td>
<td>0</td>
<td>0.06%</td>
<td>QVA v2</td>
<td>GQA</td>
<td>-41.39</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>45.19</td>
<td>-</td>
<td>-5.8</td>
</tr>
</tbody>
</table>

Table 3. Out-Of-Distribution Generalization on GQA and VQA v2 (standard split). The Gap shows the performance degradation when the model is trained on a different dataset.

### 4.1.2 Video-Text Results

We investigate how much our approach generalizes to other modalities. To this end, we evaluate eP-ALM for Video QA on MSRVTT-QA [97] and MSVD-QA [97] and for Video Captioning on MSR-VTT [98]. For the video encoding, we use the TimeFormer-base [6] model pretrained on Kinetics-600 [11]. We use 8 and 16 224x224 frames for VQA and captioning respectively.

#### Comparison to other work

to the best of our knowledge, FrozenBiLM [101] is the only parameter-efficient work proposing to adapt LMs for video-language tasks. It uses Adapters to adapt the frozen CLIP-ViT and Bidirectional LM for Video QA. We compare our approach to our re-implementation of this baseline; where we train only the Adapters and the linear projection layer to project the last [CLS] token and prepend it to the input text ones. The results in Tab. 4 show that eP-ALM outperforms this baseline by a significant margin. The reason why the latter does not give good results might be due to prepending the visual tokens to the input of OPT. We can reduce the number of parameters and slightly degrade the performance by using a shared linear connection (eP-ALM vs eP-ALM*).

<table>
<thead>
<tr>
<th>Method</th>
<th>Trained param (%)</th>
<th>MSVD-QA</th>
<th>MSRVTT-QA</th>
<th>MSRVTT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Test</td>
<td>Test</td>
<td>CIDEr</td>
</tr>
<tr>
<td>B_FrozenBiLM* (from [101])</td>
<td>3.72%</td>
<td>14.58</td>
<td>6.33</td>
<td>-</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.89%</td>
<td>38.64</td>
<td>36.16</td>
<td>47.31</td>
</tr>
<tr>
<td>eP-ALM*</td>
<td>0.54%</td>
<td>38.79</td>
<td>35.62</td>
<td>45.30</td>
</tr>
</tbody>
</table>

Table 4. Comparison with different approaches after direct finetuning on MSVD-QA, MSRVTT-QA, and MSRVTT Caption.

### Zero-Shot Results

To explore the generalization of our approach, we evaluate on Zero-Shot for VideoQA, where the model is trained on a dataset different from the target one. Table 5 shows a comparison with other approaches. eP-ALM, trained on VQA v2 (standard split), outperforms other approaches trained on significantly more data. Specifically, eP-ALM outperforms Flamingo-3B [3] on MSRVTT-QA by more than 2 points, and attains double the scores of FrozenBiLM [101]. Contrary to some of other approaches that cast the task as classification (similarity-based) [100] or constrained generation through masking, considering only a subset of answers (1k or 2k) [57, 101, 105], our approach is evaluated (with a character-wise comparison with the ground-truth) with unconstrained Open-ended Generation (OE Gen) and can generate answers with arbitrary lengths. This is more challenging and not in favor of our approach.
We find the approach sensitive to the text decoding strategy. We investigate the generalization of our approach to the work with both zero-shot (ZS) and Finetuning (FT) setups. The performance of eP-ALM is generally higher than ZS scores that train significant number of parameters, including methods with large-scale pretraining. Best and next best scores are bolded and underlined respectively. FT: Finetuning. ZS: Zero-shot.

### 4.1.3 Audio-Text Results

We compare with other end-to-end trained SoTA that takes only the audio signal as input. Tab. 6 shows that our approach is very competitive with previous work, showing the potential of efficient adaptation of LM for the audio modality.

### 4.1.4 Comparison with SoTA

To contextualize the work, we compare eP-ALM to other SoTA that train large number of parameters and most often with large-scale pretraining. Tab. 7 shows a comparison with both zero-shot (ZS) and Finetuning (FT) setups. The performance of eP-ALM is generally higher than ZS scores and still below FT ones. However, the performance gap with FT models, is smaller with the audio and video modalities.

### 4.2. Ablation Study

In this section, we ablate different component of our work.

### Comparison between different text generation methods.

We find the approach sensitive to the text decoding strategy. In Tab. 8, we compare with different text decoding methods; multinomial/random sampling [25] and greedy decoding

---

**Table 5.** Zero-Shot results on Video QA. OE Gen: unconstrained Open-Ended Generation. † evaluated on questions with top 1k answers.

<table>
<thead>
<tr>
<th>Method</th>
<th>Trained AudioCaps</th>
<th>Train. Param. (%)</th>
<th>OE Gen</th>
<th>MSRVTT-QA</th>
<th>MSVD-QA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liu et al. [63]</td>
<td>100%</td>
<td>0.671</td>
<td>0.498</td>
<td>0.232</td>
<td>0.667</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.90%</td>
<td>66.08</td>
<td>47.57</td>
<td>22.69</td>
<td>63.61</td>
</tr>
<tr>
<td>Kim et al. [32]</td>
<td>100%</td>
<td>0.699</td>
<td>0.446</td>
<td>0.203</td>
<td>0.593</td>
</tr>
<tr>
<td>Mei et al. [61]</td>
<td>100%</td>
<td>0.655</td>
<td>0.476</td>
<td>0.229</td>
<td>0.660</td>
</tr>
<tr>
<td>Gontier et al. [32]</td>
<td>100%</td>
<td>0.671</td>
<td>0.498</td>
<td>0.241</td>
<td>0.753</td>
</tr>
<tr>
<td>Liu et al. [63]</td>
<td>100%</td>
<td>0.671</td>
<td>0.498</td>
<td>0.232</td>
<td>0.667</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.90%</td>
<td>66.08</td>
<td>47.57</td>
<td>22.69</td>
<td>63.61</td>
</tr>
</tbody>
</table>

**Table 6.** Comparison with other work for Audio Captioning on AudioCaps Test set.

**Table 7.** Comparison of eP-ALM with text generation-based SoTA that train significant number of parameters, including methods with large-scale pretraining. Best and next best scores are bolded and underlined respectively. FT: Finetuning. ZS: Zero-shot.

**Table 8.** Comparison with different text generation mechanisms using eP-ALM-L.

In the following, we run some ablations for Image-Text tasks, mostly on VQA v2.

**Comparison with different variants and baselines.** We start by comparing the different variants to other work in Fig. 3. All models use OPT-350M and ViT-B/16. Other approaches lag significantly behind our model. B₇ MAGMA gives the best results (23.3% acc.) among them, followed by B₇ PromptFuse (18.82% acc.) and finally B₇ LimBEr (10.75% acc.). We also compare with another MAGMA baseline (B₇ Magma*), which has 30.72%, while the best has 34.34% (with Adapters eP-ALM). Interestingly, eP-ALM₇₇ with only one linear layer succeeds to get good performance on this challenging setup, revealing that the language and visual representation spaces are not very far. Other parameter-efficient techniques such as Prompt Tuning can help to get additional points (30.72 with eP-ALM₇₇ vs 31.27 with eP-ALM). Moreover, using different layers for each injected [CLS] token seems to give significant improvement (31.27 with eP-ALM₇₇ vs 33.08 with eP-ALM). Finally, we show that eP-ALM surpasses the “full finetuning” baseline (grayed line) that finetune all parameters by 1.27 points (31.79 vs 33.08). This reveals that training all weights of pretrained models on small datasets can reduce their generalization capability and degrades performance.
Figure 3. Comparison with other baselines on VQA v2. eP-ALM uses ViT-B/16 and OPT-350M. Our approach significantly outperforms other approaches. eP-ALM already surpasses the dashed "upper" baseline that trains all model parameters. Allocating more adaptation parameters help to increase the scores. Marker size indicates model size.

<table>
<thead>
<tr>
<th>[CLS] tokens</th>
<th>VQA v2</th>
</tr>
</thead>
<tbody>
<tr>
<td>From ViT layers</td>
<td>To OPT layers</td>
</tr>
<tr>
<td>12</td>
<td>12 to 23</td>
</tr>
<tr>
<td>6 to 12</td>
<td>12 to 23</td>
</tr>
<tr>
<td>1 to 12</td>
<td>12 to 23</td>
</tr>
<tr>
<td>6 to 12</td>
<td>1 to 23</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 9. Ablation study. Extracting the [CLS] tokens from the last layers of ViT (layers 6 to 12) is better than taking only the last token (layer 12). Injecting the [CLS] tokens lately (layers 12-24) in the OPT is better than injecting them in all layers or only in the input.

As a trade-off between performance and efficiency, we favor eP-ALM which we carry on for the following study.

**Extraction and Injection Level of [CLS] Tokens.** Here we investigate which [CLS] tokens to extract from the ViT and where is the best position to inject them inside the OPT model. Table 9 shows that extracting the last [CLS] tokens (from the last 6 layers) is better than using only the last one, as done in other approaches (Acc 30.53 vs 33.08). In addition, using all [CLS] tokens seems to degrade the performance. Moreover, prepending [CLS] tokens to all OPT layers degrades slightly (33.08 vs 32.15), and prepending to the input of OPT gives the worst results. This might indicate that it is easier to merge visual and textual tokens deeper in the model, where the representations are more abstract, compared to the first layers where we have more modality-specific features and higher representation mismatch.

**Scaling LM.** An interesting question that we investigate is the impact of scaling the language model’s parameters on our approach. Ideally, we would like to have an approach that efficiently exploits LLMs for other tasks and modalities, without having access to enormous computational resources. In Table 4, we show that the scores increase with the model size with the biggest jump being between OPT-350M (33.08 vs 37.29) and OPT-1.3B ($\sim \times 4$ the model size). The consistent improvement with scale shows the effectiveness of the approach when considering very big models.

As a trade-off between performance and model size, we favor OPT-2.7B and use it for all other experiments.

**Scaling Visual Model.** We also study how the model behaves when scaling the visual encoder. In Figure 5, we can notice that the scores increase with the size of the ViT. The best is ViT-L/16 (41.36) and the worst is ViT-S/16 (Acc 38.73). However, the ViT resolution or the number of image patches/tokens does not seem to have a significant effect on the final performance after a resolution of 16.
Scaling Compute. Table 10 shows that our approach scales with compute, as training for more epochs leads to 4 points gain in VQA accuracy. Interestingly with OPT-6.7B and ViT-L (eP-ALM_{pt-L}), we achieve a score of 43.6 by training only 0.06% of model parameters (~4M params).

<table>
<thead>
<tr>
<th>Method</th>
<th>number of epochs</th>
<th>VQA v2 Val</th>
</tr>
</thead>
<tbody>
<tr>
<td>eP-ALM</td>
<td>8</td>
<td>38.9</td>
</tr>
<tr>
<td>eP-ALM_{pt-L}</td>
<td>8</td>
<td>42.5</td>
</tr>
<tr>
<td>eP-ALM_{pt-L}</td>
<td>32</td>
<td>43.6</td>
</tr>
</tbody>
</table>

Table 10. Scaling Compute. Evaluation on VQA v2 standard split.

Qualitative Results. We show some qualitative results of our eP-ALM model with OPT-2.7B in Fig. 6. For VQA, we can notice that our model is able to correctly answer the questions. Moreover, some of the answers are richer and more accurate than the manually labeled ground truth in the dataset. This also reveals that the exact matching evaluation protocol is not in favor of the open-ended generation produced by our model. Interestingly, it seems that the model learned the answering style in the training set (i.e., short and concise answers). For Captioning, the model can generate coherent sentences describing the image globally. However, it still misses some details in the image.

5. Conclusion

In this work, we propose a new challenging setup to efficiently adapt unimodal models for multimodal tasks, which is centered around augmenting existing LMs with perception. Without multimodal pretraining, and with almost 4M trainable parameters consisting of a linear connection and a Soft Prompt, we can adapt a frozen 7B model and reach an accuracy of 54.5% on VQA v2, with unconstrained open-ended generation. We validate the effectiveness of the approach with Images, Video, and Audio modalities. This direct finetuning setup has several advantages; (a) training data/compute efficiency, (b) attains generally higher performance than pretrain-zeroshot setup, (c) easy to adapt to new tasks, modalities or other LLMs, where no costly pretraining is needed. However, the mechanism proposed in eP-ALM can be adapted in a straightforward manner to this setup.

Even though the results are still far from the state-of-the-art approaches that train most of the model parameters on much more data, the extremely small percentage of trainable parameters (0.06%) and the increasing scores with model size and compute make the work promising towards finding an intermediate point, between extremely efficient and extremely inefficient approaches, which is hopefully closer to the former.

The method has some limitations, which we illustrate in the appendix. In general, the model struggles to capture fine-grained details in the images, favors coherent generation over factual one, might hallucinate some objects not present in the image, and lacks common sense reasoning. Our approach inherits most of the limitations and biases of pretrained models, especially the LM, and training only a few adaptation parameters does not seem to avoid the transfer of these biases. Finally, the model is trained with next token prediction and is able to produce coherent text, however, it is still not clear how this paradigm can lead to real reasoning capabilities.

6. Acknowledgments

This work was partly supported by ANR grant VISA DEEP (ANR-20-CHIA-0022), and HPC resources of IDRIS under the allocation 2022-[AD011013415] and 2023-[AD011013415R1] made by GENCI. The authors would like to thank Theophane Vallaeys for fruitful discussion.

References


[59] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In CVPR, 2022. 16

[60] Sheng Liang, Mengjie Zhao, and Hinrich Schütze. Modular and parameter-efficient multimodal fusion with prompting. In Findings of the Association for Computational Linguistics: ACL 2022, pages 2976–2985, 2022. 2, 3, 5, 6, 14


[75] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy,


Appendix

The appendix is organized as follows; in Sec. A, we give more implementation details about the experiments that we conduct. We illustrate and explain the different variants of eP-ALM in Sec. B. We compare eP-ALM to other approaches following the pretrain-zero-shot setup (Sec.C). We then present more ablation studies on image-text and video-text tasks in Sec. D. Finally in Sec. E we show some qualitative results and discuss the limitation of the proposed approach.

A. Implementation Details

We use OPT-2.7B in our final model. We extract the [CLS] tokens of the last 6 layers of perceptual encoders and prepend them, after linear projection, to the text tokens of the last 12 layers of the OPT. Note that we replace the previous [CLS] with the new one to keep the same number of tokens. We finetune with the classical cross-entropy loss used to train the original OPT for VQA and Captioning. We use the AdamW optimizer with a learning rate of 1e-5 warmed up to 2e-5 then decreased to 1e-6 using a cosine scheduler. For Adapters, we use sequential Adapters after self-attentions and feedforward layers with a downsampling factor of 8 and ReLU activation. For Soft Prompt, we implement it as a linear embedding layer that takes numbers from 0 to the length of the prompt (here 10). We experiment also with adding an MLP after the prompts as done with other approaches [86]. We use the prompt with MLP for most of the experiments as we find that it gives slightly better results. The soft prompt and adapters are trained with a fixed lr of 1e-5. eP-ALM-\textit{pt} is trained with a light-weight prompt-adapter with language and feedforward layers with a downsampling factor of 8 and ReLU activation. VQA/GQA: we use a special token for VQA (‘<</a>’) to separate the question from the answer. We train for 8 epochs with a batch size of 64 (128 for GQA) and an image resolution of 224. Training our approach with OPT-2.7B for VQA v2 can be done on a single V100 GPU 32GB for 1.8 days (as the perceptual encoder is frozen, saving its output tokens can save a lot of training time). For Few-shot experiments, we train longer (for 64 epochs) with a higher starting learning rate (1e-4 warmed up to 2e-4 and decreased to 1e-5). Those marked by a * are trained for 100 epochs as in PromptFuse [60]. Image Captioning we train for 8 epochs with a batch size of 64 and an image resolution of 224.

Video QA: we sample randomly 8 frames of resolution 224x224 for each video and train for 25 epochs with a batch size of 32. For Zero-Shot experiments, we train only for 4 epochs with starting learning rate of 1e-4. We use only the spatial self-attention of TimeSformer to train on VQA v2.
Video Captioning: we sample randomly 16 frames of resolution 224x224 for each video and train for 25 epochs with a batch size of 64.

Audio Captioning: we train for 30 epochs with frequency and time masking of 24 and 96 respectively. The mel bins is 128 and the audio length is 1024. Batch size 32. For Deep Prompt, we inject new soft prompts in all 32 blocks of OPT (each with length 10).

B. eP-ALM Variants

We detail the different variants proposed in this paper (here we consider ViT-B/16 and OPT-350M for simplicity). These variants are illustrated in Fig. 7:

- **eP-ALM<sub>lin</sub>:** we extract the [CLS] tokens from the last 6 layers of the frozen ViT and inject them in the last 12 layers of the frozen OPT. To reduce inference cost, for each couple of layers (here 2), we replace the previous [CLS] with the new one (thus only increasing the number of tokens by 1 the whole process). All visual [CLS] tokens are projected by one trainable linear projection layer (shared) to fit their dimension to that of the OPT.

- **eP-ALM<sub>pt</sub>:** we augment eP-ALM<sub>lin</sub> with Prompt Tuning, which consists of prepending trainable tokens (i.e., soft prompt) to the input of the LM. This might help the model to adapt well to the new task by providing context to the text input. For the sake of efficiency, we prepend only 10 learnable tokens.

- **eP-ALM:** while one linear projection is appealing, it might not be able to capture all the particularity of different [CLS] tokens. To overcome this, we use different projections for each [CLS], while keeping the soft prompt.

- **eP-ALM<sub>ada</sub>:** another alternative to Prompt Tuning are Adapters. We follow other approaches [23] and add sequentially one adapter module (downsample, activation then upsample) after self-attention and feedforward layers in all the blocks of OPT. While this might give better results, it adds a significant number of trainable parameters.

C. Pretrain-Zeroshot Setup

The focus of this work is on direct finetuning, where we propose an efficient cross-modal interaction mechanism with low data regime. However, the proposed mechanism can be adapted in a straightforward manner to the pretrain-zeroshot evaluation setup. In this section, we show the effectiveness of eP-ALM with zero-shot evaluation after pretraining on CC3M. Specifically, we pretrain eP-ALM<sub>pt</sub> − L on CC3M for 4 epochs (which takes 35 hours on 2 gpus V100 32GB), and evaluate on COCO [61] and NoCaps (all) [2] datasets. We experiment with ViT-L, initialized from ImageNet and CLIP.

Tab. 11 show a comparison with other approaches. Without using CLIP encoder, eP-ALM significantly outperforms other work on both datasets. Using CLIP (which is trained to produce visual features aligned to text) reduces the improvement gap, where eP-ALM still outperforms all baselines on B@4 and METEOR metrics. This validates that in case of unaligned visual encoder (e.g., pretrained on ImageNet) our cross-modal interaction mechanism is efficient to align both modalities.

Note that, eP-ALM is significantly more efficient than LimBEr and MAGMA, as they train more parameters for very long time (~ 670 GPUhs). As MAGMA is trained on a lot more data, we compare with MAGMA trained on CC3M obtained from [69].

D. Ablation Study

Here we present an additional ablation study.
Table 11. Zero-shot comparison on COCO and NoCaps, after pretraining on CC3M (2.84M examples).

<table>
<thead>
<tr>
<th>Method</th>
<th>Trainable params</th>
<th>CLIP Enc.</th>
<th>COCO B@4 CIDEr</th>
<th>METEOR</th>
<th>NoCaps CIDEr</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAGMA (NFRN) [23]</td>
<td>243M</td>
<td>✓</td>
<td>8.2</td>
<td>22.4</td>
<td>-</td>
<td>4.5</td>
</tr>
<tr>
<td>LimBEr (NFRN) [69]</td>
<td>8.4M</td>
<td>✓</td>
<td>-</td>
<td>36.2</td>
<td>-</td>
<td>28.5</td>
</tr>
<tr>
<td>eP-ALMpt-L</td>
<td>4.2M</td>
<td>✓</td>
<td>11.50</td>
<td>42.47</td>
<td>15.44</td>
<td>12.53</td>
</tr>
<tr>
<td>MAGMA (on CC3M from [69])</td>
<td>243M</td>
<td>✓</td>
<td>9.7</td>
<td>47.5</td>
<td>14.6</td>
<td>-</td>
</tr>
<tr>
<td>LimBEr [69]</td>
<td>12.5M</td>
<td>✓</td>
<td>12.6</td>
<td>54.9</td>
<td>16.1</td>
<td>-</td>
</tr>
<tr>
<td>FROMAGe [52]</td>
<td>4.2M</td>
<td>✓</td>
<td>9.65</td>
<td>11.53</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>eP-ALMpt-L</td>
<td>4.2M</td>
<td>✓</td>
<td>12.97</td>
<td>51.29</td>
<td>16.23</td>
<td>13.30</td>
</tr>
</tbody>
</table>

Table 12. Ablation study: we study how much gain we can obtain by also training the pretrained vision and language models. We see slight improvement by training the pretrained models.

D.1. Image-Text

Training All Parameters Here we investigate how much gain we can obtain by unfreezing the pretrained models. We experiment on VQA v2 with eP-ALM. Table 12 shows that finetuning the pretrained models in our eP-ALM gives slight improvement, despite a large number of trainable parameters. Note that, we find that using a very small learning rate (lr=1e-7) is the only option (while keeping an lr of 1e-5 for the connectors) to unfreeze these models without significant degradation.

D.2. Video-Text

Video Encoder: here we compare different encoders to process the videos. We compare the TimeSformer [6] that has both spatial and temporal attention and trained for video classification with a simple baseline, ViT trained on ImageNet, that ignores the temporal dynamics. For ViT, we take the average of [CLS] tokens of the processed frames while for TimeSformer we consider the single [CLS] token. Table 13 shows that using video-specific encoders gives significantly better results for video captioning. In addition, we find that using 16 frames instead of 8 gives slight improvement.

Injection and Extraction level of [CLS] tokens: here we show the importance of leveraging the hierarchical representation in both the video encoder and language model. Table 14 shows the results on MSVD-QA. We show that keeping the interaction between cross-modal tokens to the last layers (layers 19 to 31) of the OPT leads to significantly better results. Extracting several tokens from different tokens of the TimeSformer gives slight improvement. However, using hierarchical video transformers [59, 64] might lead to better results. We noticed also that Adapters generally give better results than Prompt Tuning, this might be because when training on videos we sample randomly some frames, which prevents the model to overfit in the case of small datasets.

Table 13. Ablation (Caption) MSRVTT Caption.

<table>
<thead>
<tr>
<th>Method</th>
<th>MsRVTT CIDEr</th>
<th>B@4</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViT-B Avg.</td>
<td>17.96</td>
<td>12.77</td>
</tr>
<tr>
<td>ViT-B Avg. (16 f)</td>
<td>17.82</td>
<td>12.85</td>
</tr>
<tr>
<td>TimeSformer</td>
<td>20.11</td>
<td>13.53</td>
</tr>
<tr>
<td>TimeSformer (16 f)</td>
<td>20.58</td>
<td>14.12</td>
</tr>
</tbody>
</table>

Table 14. Ablation study: we investigate the extraction and injection position of [CLS] tokens for Video QA.

D.3. Audio-Text

Comparison with different variants. Here we compare different variants of our approach to different baselines for audio captioning. We evaluate on AudioCaps dataset [48], the largest benchmark for Audio Captioning. We train with mel spectrograms of 128 bins and frequency and time masking with a batch size of 8.

To the best of our knowledge, no prior work has been
proposed to adapt LM for audio-text tasks. However, there is some recent work adapting audio models using parameter-efficient techniques, such as Deep Prompts and Adapters [50]. Tab. 15 shows a comparison with different approaches. We find that prepending the audio tokens to the input of OPT does not give reasonable performance. To investigate this more, we train another baseline where the audio tokens are concatenated in the last 12 layers of OPT (eP-ALM$_{19−31+Adapter}$ and eP-ALM$_{19−31+DeepPT}$). This leads to significant improvement.

<table>
<thead>
<tr>
<th>Method</th>
<th>Trained param (%)</th>
<th>AudioCaps</th>
<th>CIDEr</th>
<th>B@4</th>
</tr>
</thead>
<tbody>
<tr>
<td>B$_{Adapter}$</td>
<td>3.76 %</td>
<td>2.96</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>eP-ALM$_{19−31+Adapter}$</td>
<td>3.76 %</td>
<td>31.17</td>
<td>8.09</td>
<td></td>
</tr>
<tr>
<td>eP-ALM$_{19−31+DeepPT}$</td>
<td>0.93 %</td>
<td>32.57</td>
<td>10.66</td>
<td></td>
</tr>
<tr>
<td>eP-ALM$_{pt}$</td>
<td>0.55 %</td>
<td>35.17</td>
<td>10.73</td>
<td></td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.90 %</td>
<td>37.14</td>
<td>11.37</td>
<td></td>
</tr>
</tbody>
</table>

Table 15. Comparison with other work for Audio Captioning on AudioCaps Test set.

Time and Frequency Masking: following other approaches [31, 73] we train eP-ALM with time and frequency masking on AudioCaps. Table 16 shows that masking significantly helps, however, using too much masking hurt the performance.

<table>
<thead>
<tr>
<th>Masking Window</th>
<th>AudioCaps</th>
<th>CIDEr</th>
<th>B@4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Frequency</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>64</td>
<td>33.94</td>
<td>10.21</td>
</tr>
<tr>
<td>192</td>
<td>48</td>
<td>35.67</td>
<td>10.40</td>
</tr>
<tr>
<td>96</td>
<td>24</td>
<td>37.14</td>
<td>11.37</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>36.01</td>
<td>10.23</td>
</tr>
</tbody>
</table>

Table 16. Ablation Study: time and frequency masking help for Audio Captioning.

E. Limitations

Even though we show appealing results for very efficient training, the method has several limitations, which we illustrate in Fig. 8. For VQA, we can notice that the model is unable to capture fine-grained details in the images (e.g., number of colors and the zebra in the first 2 examples), which might be due to constraining the interaction with the vision model through the [CLS] tokens, that generally capture global information about the image. In the case of hard questions, the model favors a coherent generation of a relevant question followed by its correct answer, instead of answering the main question ("A: what color is the phone?? black" in example 3).

For Captioning, the model seems to favor outputting a coherent sentence, even though it is not entirely correct ("many” cows in a ”crowded” city). Secondly, the model might hallucinate some objects that do not appear in the image (”apples” in example 2). Finally, the model lacks common sense reasoning, making him unable to understand that elephants are not small, and being far from the camera does not change this fact (example 3).

Our approach inherits most of the limitations and biases of pretrained models, especially the LM, and training only a few adaptation parameters does not seem to avoid the transfer of these biases. Finally, the model is trained with next token prediction and is able to produce coherent text, however, it is still not clear how this paradigm can lead to real reasoning capabilities.
Figure 8. Illustration of some limitations of eP-ALM: the model struggles to capture fine-grained details, favors coherence over factual responses, hallucinates some objects, and lacks common sense reasoning. Ground truth answers are highlighted in green. Results obtained using multinomial sampling.

Figure 9. Qualitative results of eP-ALM: the model is able to generate accurate answers and coherent descriptions of the image. Ground truth answers are highlighted in green. Results obtained using multinomial sampling.