eP-ALM: Efficient Perceptual Augmentation of Language Models

Mustafa Shukor\textsuperscript{1} Corentin Dancette\textsuperscript{1} Matthieu Cord\textsuperscript{1,2}
\textsuperscript{1}Sorbonne University \textsuperscript{2}Valeo.ai
\{firstname.lastname\}@sorbonne-universite.fr

Abstract

Large Language Models (LLMs) have so far impressed the world, with unprecedented capabilities that emerge in models at large scales. \textsuperscript{1}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 is available here: \url{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 \cite{vit} starts from the tiny model with 5M parameters to the enormous ViT-e \cite{vit-e} with 4B parameters and the largest ViT-22B with 22B parameters \cite{vit22b}. More captivating, are the scales of Large Language Models (LLMs), such as the BLOOM \cite{bloom} and OPT \cite{opt} families, ranging from hundreds of millions of parameters to 175B, in addition to other models that go beyond 100B \cite{gpt-3, gpt-4, bloom, opt} up to 1T parameters \cite{gpt-4}. 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 \cite{vl-models, vision-language-large} on large image-text datasets. However, the training cost is huge and unaffordable for much of the community, as these approaches always 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...
Drawing inspiration from the recent work in Augmented Language Models (ALMs) [65], 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 [2], vision encoder [85] and Adapters [22].

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% of their parameters, alongside (2) a minimal and challenging setup to adapt pretrained unimodal models for Image/Video/Audio-Language tasks (e.g., VQA [32, 92], Image and Audio Captioning [13, 47]). 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 [102] and unimodal encoders (e.g., ViT, TimeSformer [5], AST [30]), 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 [46, 25, 7, 43, 40]. The success in Self Supervised Learning [91, 33, 35, 81, 8] 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 [27, 55, 87], image-text datasets in general domains [60, 15, 73, 54, 50, 53], as well as specific domains, such as Cooking [74], Medical Images [66] and Event Extraction [56]. 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 [88, 89, 14, 61]. 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 [98, 14]. Another approach for VLM is to exploit existing pretrained models by keeping them frozen and training only the adaptation parameters [2, 22, 57]. This work advocates for the latter favoring training efficiency in terms of memory and time.

Adapting Language Models. Large Language Models (LLMs) [6, 77, 37, 18, 102, 70] 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 [90] at this scale and lead to sudden jumps of relevant metrics on hard downstream tasks [78, 68, 36]. This generalization ability pushed researchers to start adapting these models for other modalities [85, 2], tasks [101, 82, 97, 34] and domains [76]. Currently, most of the focus is concentrated on exploiting LLMs for vision-language tasks, such as Flamingo [2] 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 [85], PromptFuse [57], LiMBeR [64]) and adapters (MAGMA [22]). This work has demonstrated good performance, showing that it is possible to devise very efficient approaches to adapt existing language models [86, 37]. On the video side, little work has been proposed, mostly based on Adapters [96, 72]. The closest to our approach is PromptFuse [57] 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 [52] 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 [38, 3], 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 [39], Bitfit [99], Hyperperformer [62], Compacters [45] and (IA)³ [58]. These approaches have been successfully adapted to other modalities such as image [42, 12], image-text [80, 79, 103], with very little work on video [67] and Audio [49].

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 made for vision [83, 84], language [21] and vision-language [73, 16, 11], which mostly focus on designing better training objectives [73]. 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 [102], which are autoregressive language decoders consisting of

![Figure 2: Illustration of the adaptation mechanism in eP-ALM. The perceptual input (image/video/audio) is fed to the perceptual encoder E (e.g., ViT) and the corresponding text to the LM (e.g., OPT), which then generates a text conditioned on the perceptual input. The multimodal interaction is done via the [CLS] tokens acting as Perceptual Prompt, and are extracted from the last layers of the encoder, then injected in the last layers of LM, after passing by the Linear Connection C. The previous [CLS] token is replaced by the new one coming from a deeper layer, keeping the number of tokens fixed. The first layers (grayed) of each model are kept intact without any modality interaction. We ease the adaptation with a Soft Prompt that is prepended to the input of LM.](image-url)
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 more apparent with LLMs. To mitigate this, we consider only the [CLS] token of the perceptual encoders and prepend it to the text tokens. This increases the total number of tokens by 1 which maintains almost the same inference speed.

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 [52]: 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 with different linear layers for each level (multiple connections).


\[
\begin{align*}
[CLS]_i & = C(E_i(X)), \\
[CLS]_j & = \text{LM}_j([CLS]_i, p_{j-1}, t_{j-1}), \\
C & = \text{and prepended, alongside the Soft Prompt } p \text{ to the embeddings of the textual tokens } t_{j-1},
\end{align*}
\]

\[
\begin{align*}
& \text{coming from previous layer in the LM (LM}_{j-1}). \text{ This operation is repeated each 2 layers in the LM (with } N_L \text{ layers).}
\end{align*}
\]


3.2. Efficiency-driven Training Framework Setup

Current approaches still rely on many costly components that hinder their adoption by the large community. Specifically; they (1) still train a lot of parameters (e.g. vision encoders [85] and adapters [22] with \(\sim 325\text{M params/}5.11\%\)), (2) still maintain the multimodal pretraining with image-text pair datasets on top of the unimodal pretraining [64, 22, 85], (3) leverage multimodal encoders such as CLIP, pretrained on 400M image-text pairs [22, 64], (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 [22, 64]. 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-zero-shot 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-ALMada (eP-ALM with Adapters instead of Soft Prompts), eP-ALMlin (trains a shared linear connection with all [CLS] tokens, and no prompt tuning) and eP-ALMpt (lin + Soft Prompt). For Adapters, we follow other work [22] 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 [32], GQA [41] and COCO Caption [13]. 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) $B_{PromptFuse}$: which is equivalent to PromptFuse [57] 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) $B_{MAGMA}$: which is equivalent to MAGMA [22] 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-ALMptMAGMA). We also find that training the ViT degrades the performance, thus we keep it frozen in favor of their approach.
3) $B_{LimBEr}$: which is equivalent to LimBEr [64] 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-ALMpt-L 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).

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.
We investigate how much our approach generalizes to other scenarios. Specifically, we outperform VILBERT on VQAv2 by more than 2 points. Interestingly, the OOD-IID gap (i.e., the performance gap between a model trained on a different benchmark and the same model trained on a different dataset) does not give good results due to using the same 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_L).

### Table 1: Comparison with other work after direct finetuning on VQA v2, GQA, and COCO Caption.

<table>
<thead>
<tr>
<th>Method</th>
<th>VQA v2</th>
<th>GQA</th>
<th>COCO</th>
</tr>
</thead>
<tbody>
<tr>
<td>PromptFuse* [57]</td>
<td>34.1</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>B_Lin,L/Bi</td>
<td>34.1</td>
<td>33.5</td>
<td>30.81</td>
</tr>
<tr>
<td>B_FreezeF/Conv</td>
<td>40.4</td>
<td>39.5</td>
<td>33.74</td>
</tr>
<tr>
<td>B_AllGLA</td>
<td>32.2</td>
<td>31.8</td>
<td>30.98</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>48.8</td>
<td>47.8</td>
<td>43.8</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>50.7</td>
<td>53.3</td>
<td>50.2</td>
</tr>
<tr>
<td>eP-ALM_L</td>
<td>54.58/54.47</td>
<td>54.47</td>
<td>46.86</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Train. data % (# of shots)</th>
<th>VQA v2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PromptFuse* [57]</td>
<td>0.12% (512)</td>
<td>29.40</td>
</tr>
<tr>
<td>B_Lin,L/Bi</td>
<td>1% (4K)</td>
<td>28.9</td>
</tr>
<tr>
<td>B_FreezeF/Conv</td>
<td>1% (4K)</td>
<td>31.9</td>
</tr>
<tr>
<td>B_AllGLA</td>
<td>1% (4K)</td>
<td>34.5</td>
</tr>
<tr>
<td>eP-ALM_0%</td>
<td>0.12% (512)</td>
<td>31.3</td>
</tr>
<tr>
<td>eP-ALM_0.5%</td>
<td>0.12% (512)</td>
<td>30.36</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0.12% (512)</td>
<td>35.54</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>1% (4K)</td>
<td>41.9</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>10% (4K)</td>
<td>47.4</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>100% (443K)</td>
<td>52.77</td>
</tr>
</tbody>
</table>

### 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. Table 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 VQA v2 by more than 2 points. Interestingly, the OOD-IID gap (Δ) [1] for eP-ALM, at least 2 times lower compared to ALBEF [54] and VilBERT [60] (Note that, Δ should be considered alongside the accuracy, as a very poor model will have small Δ), this. This reveals that our parameter-efficient approach generalizes relatively well in OOD scenarios.

### 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 [92] and MSVD-QA [92] and for Video Captioning on MSR-VTT [93]. For the video encoding, we use the TimeFormer-base[5] model pretrained on Kinetics-600 [10]. We use 8 and 16 224x224 frames for VQA and captioning respectively.

**Comparison to other work** to the best of our knowledge, FrozenBiLM [96] 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 reimplementation 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. 3 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_L).

### 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Trained param (%)</th>
<th>MSVD-QA Val Test</th>
<th>MSRVTT-QA Val Test</th>
<th>MSRVTT CIDEr</th>
<th>B@4</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALBEF [1]</td>
<td>14M</td>
<td>100%</td>
<td>VQA v2</td>
<td>–72.1</td>
<td>50.1</td>
</tr>
<tr>
<td>VILBERT [1]</td>
<td>3M</td>
<td>100%</td>
<td>VQA v2</td>
<td>–62.2</td>
<td>46.2</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Multimodal Train. dim.</th>
<th>Trained Train data Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>eP-ALM</td>
<td>100%</td>
<td>VQA v2</td>
</tr>
<tr>
<td>eP-ALM</td>
<td>0%</td>
<td>VQA v2</td>
</tr>
</tbody>
</table>

**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 (with TimeFormer as encoder), trained on VQA v2 (standard split) by considering the image as a video with a single frame, outperforms other approaches trained on significantly more data. Specifically, eP-ALM outperforms Flamingo-3B [2] on MSRVTT-QA by more than 2 points, and attains double the scores of FrozenBiLM [96]. Contrary to some of other approaches that cast the task as classification (similarity-based) [95] or constrained generation through
masking, considering only a subset of answers (1k or 2k) [96, 100, 55], our approach is evaluated (with a character-wise comparison with the ground-truth) without unconstrained Open-ended Generation (OE Gen) and can generate answers with arbitrary lengths. This is more challenging and not in favor of our approach.

4.1.3 Audio-Text Results

We investigate the generalization of our approach to the audio domain. The encoder is AST-base model [30] pre-trained for classification on AudioSet [29]. We evaluate on AudioCaps dataset [47], 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 efficiently adapt LM for audio-text tasks, thus 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.

4.2. 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 [24] and greedy decoding with beam-search (1 to 5 beams). Greedy decoding significantly outperform multinomial sampling, and increasing the number of beams leads to additional improvements, to the detriment of increasing inference cost.

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. B_{MAGMA} gives the best results (23.3% acc.) among them, followed by B_{PromptFuse} (18.82% acc.) and finally B_{LinBEr} (10.75% acc.). We also compare with another MAGMA baseline (B_{MAGMA'} that prepends all visual tokens to the input, and we find a significant degradation compared to passing only the [CLS] token. This reveals that prepending all visual tokens directly to the input hinders the adaptation.

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. B_{MAGMA} gives the best results (23.3% acc.) among them, followed by B_{PromptFuse} (18.82% acc.) and finally B_{LinBEr} (10.75% acc.). We also compare with another MAGMA baseline (B_{MAGMA'} that prepends all visual tokens to the input, and we find a significant degradation compared to passing only the [CLS] token. This reveals that prepending all visual tokens directly to the input hinders the adaptation.

We can notice a consistent improvement of eP-ALM when adding more trainable parameters. The most parameter-efficient model is eP-ALM_{lin} which has 30.72%, while the best has 34.34% (with Adapters eP-ALM_{ada}). Interestingly, eP-ALM_{lin} with only one linear layer succeeds to get good performance on this challenging setup, revealing that the

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>Training Set</th>
<th>Test Param (%)</th>
<th>OE Gen</th>
<th>MSR-VTT-QA</th>
<th>MSVD-QA</th>
</tr>
</thead>
<tbody>
<tr>
<td>JustAsk [29]</td>
<td>ActivityNet-QA</td>
<td>80.6%</td>
<td>✓</td>
<td>17.4</td>
<td>14.2</td>
</tr>
<tr>
<td>JustAsk [29]</td>
<td>ActivityNet-QA</td>
<td>80.6%</td>
<td>✓</td>
<td>17.4</td>
<td>14.2</td>
</tr>
</tbody>
</table>

Table 6: Comparison with other work for AudioCaptioning on AudioCaps Test set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Trained param (%)</th>
<th>B@1</th>
<th>B@2</th>
<th>METEOR</th>
<th>CIDEr</th>
<th>SPICE</th>
<th>SPIDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kim et al. [45]</td>
<td>100%</td>
<td>0.614</td>
<td>0.446</td>
<td>0.203</td>
<td>0.593</td>
<td>0.144</td>
<td>0.369</td>
</tr>
<tr>
<td>Kozumi et al. [11]</td>
<td>100%</td>
<td>0.636</td>
<td>0.458</td>
<td>0.199</td>
<td>0.603</td>
<td>0.139</td>
<td>0.371</td>
</tr>
<tr>
<td>Eren et al. [23]</td>
<td>100%</td>
<td>0.710</td>
<td>0.490</td>
<td>0.290</td>
<td>0.750</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Xu et al. [46]</td>
<td>100%</td>
<td>0.650</td>
<td>0.476</td>
<td>0.229</td>
<td>0.660</td>
<td>0.168</td>
<td>0.414</td>
</tr>
<tr>
<td>Mei et al. [43]</td>
<td>100%</td>
<td>0.647</td>
<td>0.485</td>
<td>0.222</td>
<td>0.679</td>
<td>0.160</td>
<td>0.420</td>
</tr>
<tr>
<td>Goutier et al. [31]</td>
<td>100%</td>
<td>0.699</td>
<td>0.523</td>
<td>0.241</td>
<td>0.753</td>
<td>0.176</td>
<td>0.465</td>
</tr>
<tr>
<td>Lin et al. [55]</td>
<td>100%</td>
<td>0.671</td>
<td>0.498</td>
<td>0.232</td>
<td>0.667</td>
<td>0.172</td>
<td>0.420</td>
</tr>
<tr>
<td>eP-ALM (nh3)</td>
<td>0.90%</td>
<td>0.609</td>
<td>0.475</td>
<td>0.228</td>
<td>0.651</td>
<td>0.166</td>
<td>0.412</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Dataset (Metric)</th>
<th>SoTA (ZS)</th>
<th>eP-ALM (FT)</th>
<th>SoTA (FT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AudioCaps (CIDEr)</td>
<td>83.6</td>
<td>66.7</td>
<td>63.6</td>
</tr>
<tr>
<td>MSR-VTT-QA (Acc)</td>
<td>74.0</td>
<td>44.1</td>
<td>43.2</td>
</tr>
<tr>
<td>MSR-VTT (CIDEr)</td>
<td>29.7</td>
<td>50.0</td>
<td>47.7</td>
</tr>
<tr>
<td>MSVD-QA (Acc)</td>
<td>84.3</td>
<td>44.3</td>
<td>37.3</td>
</tr>
<tr>
<td>MSVD (VQA v2)</td>
<td>39.7</td>
<td>42.7</td>
<td>40.8</td>
</tr>
<tr>
<td>MSVD (VTP)</td>
<td>44.1</td>
<td>38.5</td>
<td>36.6</td>
</tr>
</tbody>
</table>

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

Decoding Method | # of beams | VQA v2 | COCO | MSRVTT-QA | MSVD-QA | MSVD (VQA v2) | MSVD (VTP) |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Multinomial</td>
<td>1</td>
<td>53.3</td>
<td>145.3</td>
<td>27.09</td>
<td>27.5</td>
<td>400M-CLIP</td>
<td>2.9</td>
</tr>
<tr>
<td>Greedy</td>
<td>1</td>
<td>54.47</td>
<td>12.6</td>
<td>44.1</td>
<td>38.5</td>
<td>40% (Flamingo80B)</td>
<td>2.9</td>
</tr>
<tr>
<td>Greedy</td>
<td>3</td>
<td>54.90</td>
<td>12.6</td>
<td>44.1</td>
<td>38.5</td>
<td>100% (FewVLM)</td>
<td>2.7</td>
</tr>
<tr>
<td>Greedy</td>
<td>5</td>
<td>54.92</td>
<td>12.6</td>
<td>44.1</td>
<td>38.5</td>
<td>100% (OFA)</td>
<td>2.7</td>
</tr>
</tbody>
</table>

In the following, we run some ablations for Image-Text tasks, mostly on VQA v2.
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-ALM\textsubscript{lin} vs 31.27 with eP-ALM\textsubscript{pt}). Moreover, using different layers for each injected [CLS] token seems to give significant improvement (31.27 with eP-ALM\textsubscript{pt} vs 33.08 with eP-ALM).

Finally, we show that eP-ALM surpasses the “full fine-tuning” 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.

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 5, 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 (≈×4 the model size). The consistent improvement with scale shows the effectiveness of the approach when considering very big models.

**Scaling Visual Model.** We also study how the model behaves when scaling the visual encoder. In Figure 4, we can notice that the scores increase with the size of the ViT. Increasing the patch resolution beyond 16 does not help.

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.

<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>

**Figure 4:** Scaling Vision Model; the score increases with the size of the ViT. Increasing the patch resolution beyond 16 does not help.

**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\textsubscript{pt-L}), we achieve a score of 43.6 by
training only **0.06%** of model parameters (~4M params).

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

<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</td>
<td>32</td>
<td>42.9</td>
</tr>
<tr>
<td>eP-ALM&lt;sub&gt;pt&lt;/sub&gt;-L</td>
<td>8</td>
<td>42.5</td>
</tr>
<tr>
<td>eP-ALM&lt;sub&gt;pt&lt;/sub&gt;-L</td>
<td>32</td>
<td>43.6</td>
</tr>
</tbody>
</table>

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

### 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.

### 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


Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Zhiwei Liu. Conditional prompt learning for vision-language...