\(\mathcal{E}\)-ViLM: Efficient Video-Language Model via Masked Video Modeling with Semantic Vector-Quantized Tokenizer

Jacob Zhiyuan Fang\(^\dagger\), Skyler Zheng\(^\dagger\), Vasu Sharma\(^\dagger\), Robinson Piramuthu\(^\dagger\)
\(^\dagger\)Amazon, \(^\dagger\)Meta
{zyfang, nzhengji, robinpir}@amazon.com vasusharma@meta.com

Abstract

To build scalable models for the challenging real-world tasks, it is important to learn from diverse, multi-modal data in various forms (e.g., videos, text, images). Amongst the existing works, a plethora of them have been focusing on leveraging large but cumbersome cross-modal architectures. Regardless of their effectiveness, larger architectures unavoidably prevent the models from being extended to real-world applications, so building a lightweight VL architecture and an efficient learning schema is of great practical value. In this paper, we propose an Efficient Video-Language Model (dubbed as \(\mathcal{E}\)-ViLM) and a masked video modeling (MVM) schema, assisted with a semantic vector-quantized tokenizer. In particular, our \(\mathcal{E}\)-ViLM learns to reconstruct the semantic labels of masked video regions, produced by the pre-trained vector-quantized tokenizer which discretizes the continuous visual signals into labels. We show that with our simple MVM task and regular VL pre-training modelings, our \(\mathcal{E}\)-ViLM, despite its compactness, is able to learn expressive representations from Video-Language corpus and generalize well to extensive Video-Language tasks including video question answering, text-to-video retrieval, etc. In particular, our \(\mathcal{E}\)-ViLM obtains obvious efficiency improvements by reaching competing performances with faster inference speed: i.e., our model reaches 39.3% Top-1 accuracy on the MSRVTT benchmark, retaining 91.4% of the accuracy of state-of-the-art larger VL architecture with only 15% parameters and 94.8% fewer GFLOPs. We also provide extensive ablative studies that validate the effectiveness of our proposed learning schema for \(\mathcal{E}\)-ViLM.

1. Introduction

The task of Video (V) and Language (L) pre-training aims to learn joint and robust cross-modal representations from video-text pairs. Recent advancements of VL pre-training have obtained great development and are primarily reflected in the aspects of leveraging more video-text pairs for scaling pre-training [52, 54, 90]; superior visual encoder for expressive video representations [41, 43, 77]; and unified VL architectures [23, 76, 78, 80, 84] etc. Despite this, the success of these prior arts unanimously relies on escalating parameters that impede their real-world applications due to the high latency and large memory footprint during inference.

Existing efforts to develop small VL architectures [20, 79, 82, 83] or end-to-end cross-modal architectures [38, 80] for image captioning [19], VQA [11], are restricted to image-text domains only. The challenge then becomes how to train these small VL architectures with increased accuracy. The majority of existing VL pre-training models stand on a common plateau of modeling, i.e., a cross-modal contrastive modeling [33, 59] in which visual representations...
are optimized to be aligned with the matched text while contrasting with the unmatched text; and a masked language modeling [14] schema in which the model learns to predict randomly masked textual tokens. The modeling of visual representations for VL learning, especially on small VL architectures, are typically less effective and still needs more investigations.

Recent studies also witness significant developments in masked token prediction for unsupervised representation learning in both natural language processing (e.g., BERT [14], GPT [9, 59, 60]) and vision domain (e.g., BEiT [6] and MAE [28]) from which the learned representations generalize well to diverse tasks. Despite remarkable headway made, in the regime of cross-modal tasks dealing with image/video and language, this “mask-and-predict” modeling is primarily limited to just the language modalities [17, 20, 42, 45, 68, 79, 91] (reconstructing solely masked language tokens). Hereby, we conjecture that a critical reason that imposes the masked visual token prediction on the VL domain is the inconsistent information density during joint VL pre-training, especially on smaller architectures where visual representations lack expressiveness. In particular: a unified VL architecture (e.g., ViLT [38] and [76]) encodes both modalities where continuous visual embedding contains redundant information yet embeddings from language are highly condensed with latent semantics. Such inconsistency prevents the model from inferring contextualized & aligned cross-modal relations, and it becomes even more visible on smaller VL architectures for highly redundant visual data like videos. Figure 2 shows the Multiple-Choice accuracy of VL models, pre-trained w./w.o Masked Video Token Modeling with different number of learnable parameters: we clearly observe that smaller VL architecture benefits more from discretized masked video token prediction. This clearly indicates that MVM can be the cure of lesion in the furtherance of a more tiny and powerful VL architecture.

In this paper, we investigate how to set up masked video modeling (MVM) with the goal of obtaining more robust and general VL representations on small VL architectures for diverse video and language tasks. We propose to use a semantic Vector-Quantized (VQ) module to discretize the visual embedding into a set of labels in order to address the aforementioned problem. To accomplish this, we choose the nearest code from a parameterized and learnable codebook as the target tokens for reconstruction. Different from previous efforts like VIOLLET [23] and [24], we train the video tokenizer without using pixel space, but by reconstructing the words from the video-level captions. As a result, the density gap is lessened because the input visuals have been converted into a smaller number of discrete labels. In order to encourage the learning of inferring the latent semantics of masked visual regions from contextualized visual contexts and languages, our proposed MVM is then formulated as predicting the semantic labels of a set of randomly masked video regions. Masked video/language modeling and widely used video-text matching modeling constitute our overall loss. We carry out extensive tests to verify the performance of our proposed MVM. We particularly experimented with VL pre-training on the WebVID [5] dataset and then assessed the learned representations on a multitude of benchmarks related to video and video-language, including generalization ability, video question answering, and video retrieval by text (linear probe of activity recognition). Our proposed \( \mathcal{E} \)-VLm achieves competing performances consistently across all benchmarks with less than 15% learnable parameters and 40 times fewer GFLOPs than the state-of-the-art (see Figure 1). The learned representations from the \( \mathcal{E} \)-VLm also exhibit excellent generalization abilities when applied to an activity recognition task.

To summarize our contribution:

- **We propose a small and novel video-language architecture, \( \mathcal{E} \)-VLm**, which achieves prominent accuracy & inference speed trade-off. To the best of our knowledge, this is the very initial attempt that leverages a small video-language model for VL representation learning.

- **To counter the pain point of training small VL architecture caused by inconsistent cross-modal information density**, we propose a novel semantic reconstruction based video tokenizer for discretized masked video modeling. This facilitates our small VL model to efficiently learn from large-scale video-text pairs for expressive representations.

- **Extensive quantitative experiments validate the effectiveness of our proposed model and learning schema.** Our proposed \( \mathcal{E} \)-VLm achieves competing results across extensive benchmarks with obvious inference advantages.

### 2. Related Work

**Efficient Vision-Language Models.** The remarkable performance of most VL models [25, 27, 29–31, 41, 57, 67, 69, 85, 88] is closely dependent on the great visual representations...
from expressive visual encoders like object detector [2, 45]. Zhang et al. [91] develop improved Faster-RCNN [63] for the sake of better visual representations that substantially uplifts the results on various VL tasks. Despite so, cumbersome visual encoder like object decoder arguably hinders the usefulness of VL models in real-world applications due to their demands of high computational resources and are de facto the major computational burden in VL architecture. To resolve this challenge, many recent efforts propose to optimize the detector [20, 79], utilize grid visual representations [17, 19, 34, 38, 78] with no regional operations in the detector. There also emerge many efforts to build efficient Vision-Language models. For instance, [79] proposes to utilize an EfficientNet variations [70] as the detector for visual feature extraction, which largely minimizes the inference bottleneck. Other follow-ups exploit knowledge distillation technique [20] to improve the accuracy of VL models. [20, 34, 38, 52] explore the feasibility of using end-to-end models to reduce the inference time on object detector than two-stage VL architectures.

**Masked Auto-encoder.** Recent studies also witness significant development in masked token prediction for unsupervised representation learning in both natural language processing, e.g., BERT [14] and GPT [9, 59, 60]. This mask-and-predict pre-training manner extends to the visual representation learning: BEiT [6] for the first time proposes to tokenize the continuous visual embedding into discrete tokens and then reconstruct the randomly masked token.

MAE [28] further simplifies this by directly predicting the masked patches and using their pixel values as the reconstruction target. Follow-up works further advance this to different architectures [44], various visual data (i.e., videos [21, 56, 71], depth image [87] etc). In particular, concurrent works from Tong et al. [72] and Feichtenhofer et al. [22] both show that video data, due to its high spatio-temporal redundancy, requires a significantly high masking ratio. Chen et al. [13] adopts a masked region modeling to reconstruct the masked regional representations of the image for VL pre-training but with limited improvement. Bachmann et al. [4] also [26] further extend this to a multi-modal-multi task setup where they show that additional modalities can be incorporated into the training objective by multi-modal masking. In particular, VIOLET [23] also proposes to leverage the video tokenizer to provide discretized video patch labels for masked video modeling. Our work differentiates with [23] majorly from two perspectives: 1. we focus on developing small and efficient architectures. 2. [23] leverages tokenizer of DALL-E [6] as video tagger, which is obtained from a pixel-level re-construction task while ours is trained from semantic-level re-construction.

3. Efficient Video-Language Modeling

In this section, we describe how we build and train our efficient Video-Language Model (E-ViLM) that is smaller and faster. This section first discusses in detail about the
architectural design of the model, followed by our proposed Masked Video Modeling aiming for more efficient VL pre-training and ends with our overall training schema.

### 3.1. Model Architecture

Figure 3 illustrates the overall architecture of our E-ViLM, which is an end-to-end architecture consisting of: a Language Encoder (ENC_L) that encodes the text (T) into a sequence of continuous embedding, Video Encoder (ENC_V) that encodes stack of sparsely sampled video frames (I), and a transformer decoder (DEC_VL) that allows cross-modal interaction across Visual and Language modalities:

\[
e^V = ENC_V(I), \quad e^L = ENC_L(T), \quad h = DEC_VL(e^V, e^L),
\]

where \( e^V/L \) denotes the visual/language embedding. In quest of a lightweight VL model, we purposefully constrain the size of each module in E-ViLM by leveraging a few efficient but well-performing deep architectures across both Vision and Language benchmarks.

**Video Encoder.** In our work, we resort to a lightweight 2D visual architecture as the visual encoder to obtain frame-wise grid representations as visual embedding. To be specific, given the sparsely sampled \( T \) video frames \( I = \{I_0, \ldots, I_T\} \), \( I_t \in \mathbb{R}^{H \times W \times 3} \), each frame is represented as continuous embedding \( e^V_t \in \mathbb{R}^{H/S \times W/S \times D} \), where \( S \) and \( D \) denote the down-sample ratio of the network and the dimension of embedding, respectively. We then flatten and concatenate all embeddings, followed by a linear transformation for dimension adjustment to produce the video-level visual representation \( e^V \).

**Language Encoder.** Transformer architecture [75] and its variations [14, 60] achieve exceptional performance on multiple natural language understanding tasks. Yet, transformer-like architectures suffer from slow inference speeds primarily because of the quadratic complexity of the self-attention mechanism and a large numbers of parameters. Although improved works [66, 73] take advantage of efficient/local self-attention mechanism or knowledge distillation [35] technique to address this, their performances still fall short of expectation. Instead, we propose to utilize a small BERT architecture with decreased hidden dimension and squeezed vocabulary where only frequent words are retained in favor of a more tiny tokenizer layer. A linear projection layer maps the language embedding to a consistent dimension at the end. We introduce detailed architectural settings in the experiment section.

**Multi-modal Decoder.** Our multi-modal decoder is composed of a stack of transformer blocks that takes the concatenation of visual and language embeddings as input, enabling the cross-modal interaction. To incorporate information for distinguishing between the two modalities, we add token type embeddings \( t^L \) and \( t^V \) to language embedding and video embedding, respectively. We also attach a special classification token \([CLS]\) (as shown in Figure 3):

\[
[h^{CLS}, h^L, h^V] = DEC_VL([e^{CLS}, e^L + t^L, e^V + t^V]).
\]

Similar to the language encoder, we decrease the hidden dimension in each block.

### 3.2. MVM with Semantic Video Tokenizer

We propose a novel masked video modeling (MVM) based on a semantic video tokenizer to assist VL representation learning. Masked language modeling (MLM) obtains great success [14] in a series of natural language comprehension tasks. Recent efforts on visual masked auto-encoder [28, 71] also show prominent effectiveness on self-supervised representation learning. In essence, the visual model learns to restore the masked raw pixels based on the observable pixels, which are continuous and high-dimensional, while the prediction target in MLM is the language token with high semantics and dense information. In particular, VL architecture encodes both modalities where continuous visual embedding contains redundant information yet semantics from language are highly condensed. Such inconsistency prevents the model from inferring out contextualized & aligned cross-modal relations, which play an essential role in VL pre-training [10], especially on highly redundant visual data like videos. For this reason, we propose to leverage an independent Tokenizer branch (described in a
later section) as a video tagger, providing one-hot semantic labels $y_i^V$ for the $i$-th masked video region. To be specific, we randomly mask out video frame patches by replacing them with zero-value pixels. Our proposed MVM aims to reconstruct the semantic labels of the masked regions via a classification task. And we use cross-entropy to measure this error:

$$\mathcal{L}_{\text{MVM}} = -\sum_{i=1}^{N} \frac{1}{N} \text{CE}(\hat{y}_i^V, y_i^V),$$  \hspace{1cm} \text{(3)}$$

where $N$ represents the number of masked regions and $y_i^V$ is the predicted semantic probability of the $i$-th masked video region, obtained by a semantic classification head: $\hat{y}_i^V = \text{HEAD}^V(h_i^V)$. In the later section, we explain how we construct our video tokenizer and discuss how it differentiates from previous efforts.

### 3.3. Semantic Vector-Quantized Video Tokenizer

A recent work leverages a visual tokenizer to discretize the visual regions for self-supervised representation learning [58], [23, 51] further extend this manner to Vision-Language learning and utilize a pre-trained visual tokenizer (i.e., tokenizer of DALL-E [62] and VQ-GAN [18]) as a tagger. However, a notable defect of such a tokenizer comes from its lack of semantics, as they are unanimously obtained from the pre-task of reconstructing raw image pixels in an auto-encoder fashion. To overcome this, we propose to construct our video tokenizer by reconstructing the semantics of the videos.

Following [18, 74], we first learn an asymmetric Encoder-Decoder model consisting of a ViT [59] or VideoSWIN [49] based encoder ($E$) and a decoder ($D$) that is composed of stacks of self-attention blocks. It learns to represent the video frames with codes from a learnable and discrete codebook with $M$ latent code embedding $\mathcal{C} = \{c_1, \ldots, c_M\}$, with $c_j \in \mathbb{R}^D$ (see Figure 3). Our vector-quantization module $q(\cdot)$ approximates a set of latent codes $\mathcal{C}_q$ for input video frames by a nearest neighbor look-up using the shared embedding space such that:

$$\mathcal{C}_q = q(E(\chi)) := \left(\arg\min_{c_j \in \mathcal{C}} \|E(x_i) - c_j\|\right),$$  \hspace{1cm} \text{(4)}$$

with $\chi = \{x_i \}_{i=1}^{T \times H/h \times W/w}$, and $x_i \in \mathbb{R}^{h \times w \times 3}$ denotes $i$-th the image patch obtained from $T$ sampled image frames via non-overlapping partitioning.

We introduce the semantic reconstruction task by predicting the discrete words $T = \{w_k\}$ of the video caption with latent codes by the decoder. We compute the multi-label probability using a classification head ($\text{HEAD}^S$) given the averaged output embedding from the decoder ($D$):

$$\hat{y}^S = \text{HEAD}^S \left(\frac{1}{|\chi|} \sum_{c_q} D(c_q)\right),$$  \hspace{1cm} \text{(5)}$$

Considering the extremely imbalanced word distribution and the rarity of positive samples, we adopt focal loss for tokenizer training [7, 20], which shows outstanding results on a multi-label classification task:

$$\mathcal{L}_{\text{TOKEN}} = \frac{1}{K} \sum_{k=1}^{K} \left\{ \log(\hat{y}_k^S) ; \hat{y}_k^S = 1, \hat{y}_k^S = 0 \right\},$$  \hspace{1cm} \text{(6)}$$

where $\hat{y}_k^S \in [0, 1]$ is the predicted probability for the $k$-th category, and $K$ denotes the size of the vocabulary. Intuitively, our tokenizer is trained to represent visual frames as a distribution over latent constituents, and these latent codes contain vivid semantics that are essential to Vision-Language pre-training task. Because back-propagation is not possible due to the non-differentiable quantization step, we copy the gradients over $\mathcal{C}_q$ directly to the $E(\chi)$ for encoder (straight-through gradient estimating [8]). Our Semantic Vector-Quantized Tokenizer is optimized by the objective:

$$\mathcal{L}_{\text{SVQ}} = \mathcal{L}_{\text{TOKEN}} + ||\text{sg}(E(\chi)) - \mathcal{C}_q||^2_2 + ||E(\chi) - \text{sg}(\mathcal{C}_q)||^2_2, \hspace{1cm} \text{(7)}$$

and $\text{sg}[\cdot]$ denotes the stop-gradient operation [74], which ensures the encoder commits to an embedding and its output does not grow arbitrarily. After the training, we retain only the encoder and the vector-quantized module as our tokenizer. Note that during VL pre-training, the tokenizer is capable of predicting the discrete semantic labels either online or off-the-shelf.

### 3.4. Video-Language Pre-Training

The training of $\mathcal{E}$-ViLM consists of two stages, where we first obtain the video tokenizer by optimizing it jointly with $\mathcal{L}_{\text{SVQ}}$ and $\mathcal{L}_{\text{TOKEN}}$. For the overall Video-Language pre-training, we use the common VL pre-training modeling objectives: masked language modeling loss ($\mathcal{L}_{\text{MLM}}$), where a certain ratio of the textual words are masked and the model predicts their label; also Video-Text matching loss ($\mathcal{L}_{\text{VTM}}$) where the model is expected to predict whether the video-text pair matches. Our overall loss at the pre-training stage is the combination of the aforementioned items:

$$\mathcal{L} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{VTM}} + \mathcal{L}_{\text{MVM}}.$$  \hspace{1cm} \text{(8)}$$

### 4. Experiments

We now introduce the detailed implementation of $\mathcal{E}$-ViLM and also discuss the empirical results that validate the effectiveness of our proposed training schema. To highlight the generalizability of $\mathcal{E}$-ViLM, we benchmark performances of $\mathcal{E}$-ViLM and compare them with prior arts on multiple video-language benchmarks. We then empirically study the benefits of different VL models and conduct ablative studies for various settings. To this end, we discuss in-depth about the efficiency of $\mathcal{E}$-ViLM and outline potential future works of efficient Vision-Language learning.
We build our Action, TGIF-Transition and TGIF-Frame) [32, 46] datasets; Table 1. Comparison of E-ViLM with existing methods with VL pre-training on video question answering and multiple-choice (MC) tasks. P.T. denotes the pre-trained data. # P and # F represent the number of parameters and number of frames. The bottom part of the table shows the results of small VL architectures (≤ 20M parameters). * denotes the results are re-produced using official released pre-trained checkpoint [36]. FP denotes the FLOPs of models (in G). Best results are underlined.

### 4.1. Datasets

**Pre-training Dataset.** In our experiment, we pre-train our model on WebVid-2.5M [5], which contains 2.5M video-text pairs scraped from the web with captions describing the holistic semantics of the video. Videos from WebVid are averagely around 18 seconds with 20 words per caption.

**Downstream Datasets.** To assess the representations learned by E-ViLM, we conduct extensive evaluations across multiple video-language downstream tasks with more than 10 benchmarks: spanning over video question answering task on MSRVTT [86], MSVD [12], TGIF-QA (including TGIF-Action, TGIF-Transition and TGIF-Frame) [32, 46] datasets; video retrieval by text task on MSRVTT, DiDeMo [3]. Meanwhile, in order to further evaluate the generalizability of E-ViLM, we continue to transfer the learned VL representations for linear-probe action recognition task on HMDB51 [40] and Kinetics-400 [37] datasets, as well as the zero-shot multiple-choice task on MSRVTT. We report the performance of Recall at K (R@K, K = 1, 5, 10) for text-to-video retrieval task; and report the accuracy on VQA task, action recognition and zero-shot multiple choice tasks.

**Architectural Details.** We build our E-ViLM based on an efficient visual architecture, EdgeNext [53], that involves both convolution layers and efficient local attention operation but with a light computational burden. Specifically, our visual encoder (EdgeNext_small) only consumes 1.26G FLOPs for one 256 × 256 frame with 5.59M parameters. Note that our E-ViLM is not architecture-specific and we choose EdgeNext [53] in our implementation due to their well-organized open-source code and good accuracy-speed trade-off. For the language encoder, we adopt a tiny BERT variation with decreased hidden size (128) and fewer attention heads per layer (4). In order to further compress the language branch, we truncate the vocabulary of the tokenizer by retaining only the top 30% most frequent words in COCO dataset [47] to avoid dataset bias. This additionally squeezes the language encoder with only 2.4M learnable parameters. Our cross-modal decoder is composed of 4 transformer blocks with decreased hidden size (384). In total, our E-ViLM accommodates 16M parameters (without a task specific head) and can encode video (4 frames)-caption pairs with 12.2 FLOPs(G), which is obviously advantageous compared with previous state-of-the-art. Table 1 also provides the specific number of parameters and FLOPs comparisons with other VL architectures in regard to efficiency.

We instantiate our video tokenizer with different visual backbones and compare their performances: namely, a ViT [16] architecture pre-trained by [59]. Our tokenizer also includes a decoder which consists of 8 layers of transformer architecture, yet this module is dropped after the training phase of the video tokenizer. Our semantic codebook contains 9, 420 learnable prototype embedding in the dimension of 32 initialized randomly. To compare the effects of various tokenizers, we also explore to utilize other architectures as the tokenizer, e.g., discrete variational auto-encoder (dVAE) from DALL-E [62] which is trained by reconstructing the RGB pixels, and a Video Swim Transformer (VideoSwin-Tiny) [49] pre-trained on Kinetics-400 [37] using our learning schema. Our video tokenizer “escorts” the pre-training of E-ViLM in the evaluation mode, so it barely takes up resources during VL pre-training.

**Training.** For VL pre-training, we sparsely sample 4 video frames per video and resize them into 256 × 256 after random augmentation. Different masking strategies and masking ratios, as previous works indicate [23,24,71,72], may affect the representation learning. We adopt block-wise masking [6]
with masking ratio 40%. We use optimizer adamW [50] with an initial learning rate of $1e^{-4}$, weight decay of 0.01 during pre-training and train \( \mathcal{E}\text{-ViLM} \) on WebViD for 20 epochs with \( L_{\text{MVM}} + L_{\text{VTM}} + L_{\text{MLM}} \) jointly. For MLM, we randomly mask 15% of the words during training. For downstream tasks, we stick to the sparsely sampled 4 frames and use task-specific learning rates/epochs. Same as the pre-training phase, we trained the video tokenizer on WebViD for 10 epochs with initial learning rate $2e^{-4}$. The embedding before/after look-up have all been \( l_2 \)-normalized for Euclidean distance computation. We further discuss the effect of tokenizer in the supplementary materials.

### 4.2. Results

#### Video Question Answering.

We compare \( \mathcal{E}\text{-ViLM} \) with previous state-of-the-art methods on Video Question Answering (VQA) task on multiple datasets and present the results in Table 1. We clearly observe that \( \mathcal{E}\text{-ViLM} \) reach on par performances or even surpasses most prior arts, where they unanimously employ large architectures. In particular, \( \mathcal{E}\text{-ViLM} \) achieves 39.3% Top-1 Acc. on MSRVT-QA task, with a significant gain of 2.5% Acc. over all-in-one-T [76], which is a tiny VL architecture. Remarkable improvements are observed on T-GIF and MSVD benchmarks consistently: \( \mathcal{E}\text{-ViLM} \) exceeds all-in-one-T for +9% Top-1 Acc. on MSVD, +4.1% on TGIIF-Action, +10.7% on TGIIF-Transition and +6.8% higher on TGIIF-Frame. MC denotes the multiple choice task and \( \mathcal{E}\text{-ViLM} \) also performs prominently better than its counterpart. Notably, we note that \( \mathcal{E}\text{-ViLM} \) even exceed most large VL architectures, i.e., ClipBERT [41], CoMVT [92] and SSML [1], indicating the superiority of our efficient architecture which has an obvious advantage regarding the efficiency and inference speed. Also, \( \mathcal{E}\text{-ViLM} \) is surprisingly more data efficient than the competitors and only exploits 2.5M video-text pairs, whereas the baseline approaches use much larger scale pre-training: for instance, JustAsk [88] adopts 69M video-question-answer triplets auto-extracted from narrated videos for training; All-in-one-T has been pre-trained on much larger VL corpus, WebViD2.5M and HowTo100M [55], which consists of over 100 million video clip-text pairs.

#### Text-to-Video Retrieval.

We study to extend \( \mathcal{E}\text{-ViLM} \) for video retrieval from text task. Table 2 summarizes results on MSRVT [86] and DiDeMo [3] for comparisons with previous methods. We follow [76, 78] to speed up retrieval efficiency by replacing the one-to-one image-text matching schema with encoding unimodal inputs during fine-tuning and inference. This additionally accelerates the notoriously slow retrieval process. For instance, our model only takes up 2 minutes for one round of text-to-video retrieval on MSRVT testing split with one V100 GPU device, while regular VL models, however, usually take up to several hours [76] or more. We explain more about how we conduct video retrieval in the Appendix. \( \mathcal{E}\text{-ViLM} \) achieves significant performance gain over a tiny VL variation of all-in-one [76], pre-trained across the aggregation of HowTo100M and WebViD2.5M datasets. Although \( \mathcal{E}\text{-ViLM} \) leverages only 16M parameters, it still surpasses some prevailing retrieval ar-

### Table 2. Comparison of \( \mathcal{E}\text{-ViLM} \) with state-of-the-art VL models on text-to-video-retrieval task. Results are reported on R@1 / R@5 / R@10. PT denotes the VL pre-training corpus.

<table>
<thead>
<tr>
<th>Methods</th>
<th>P.T.</th>
<th>MSRVT</th>
<th>DiDeMo</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>text-to-video</td>
<td>text-to-video</td>
</tr>
<tr>
<td>HT100M [55]</td>
<td>HT100M</td>
<td>14.9 / 40.2 / 52.8</td>
<td>-</td>
</tr>
<tr>
<td>HERO [43]</td>
<td>HT100M</td>
<td>16.8 / 43.4 / 57.7</td>
<td>2.1 / 11.4 / 36.1</td>
</tr>
<tr>
<td>ClipBERT [41]</td>
<td>COCO [47] + VG [39]</td>
<td>22.0 / 46.8 / 59.9</td>
<td>20.4 / 48.0 / 60.8</td>
</tr>
<tr>
<td>TACo [89]</td>
<td>HT100M</td>
<td>28.4 / 57.8 / 71.2</td>
<td>-</td>
</tr>
<tr>
<td>VideoCLIP [85]</td>
<td>HT100M</td>
<td>30.9 / 55.4 / 66.8</td>
<td>16.6 / 48.0 / -</td>
</tr>
<tr>
<td>Frozen [5]</td>
<td>CC [65] + Web2.5M</td>
<td>32.5 / 61.5 / 71.2</td>
<td>31.0 / 59.8 / 72.4</td>
</tr>
<tr>
<td>VIOLET [23]</td>
<td>CC [65] + Web2.5M</td>
<td>34.2 / 63.5 / 73.6</td>
<td>32.9 / 63.0 / 74.7</td>
</tr>
<tr>
<td>all-in-one-B [76]</td>
<td>HT100M + Web2.5M</td>
<td>39.5 / 63.3 / 71.9</td>
<td>-</td>
</tr>
<tr>
<td>all-in-one-T [76]</td>
<td>HT100M + Web2.5M</td>
<td>16.3 / 37.4 / 53.1*</td>
<td>12.5 / 26.4 / 38.9*</td>
</tr>
<tr>
<td>( \mathcal{E}\text{-ViLM} )</td>
<td>Web2.5M</td>
<td>27.0 / 51.7 / 64.4</td>
<td>23.8 / 44.9 / 53.2</td>
</tr>
</tbody>
</table>

### Table 3. Activity recognition via linear probe on Kinetics-400 [37] and HMDB51 [40] datasets.

<table>
<thead>
<tr>
<th>Methods</th>
<th>HMDB51 (Top-1)</th>
<th>K400 (Top-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frozen [5]</td>
<td>54.2</td>
<td>50.5</td>
</tr>
<tr>
<td>T-AVG. [76]</td>
<td>43.1</td>
<td>44.3</td>
</tr>
<tr>
<td>all-in-one-B</td>
<td>52.9</td>
<td>50.8</td>
</tr>
<tr>
<td>all-in-one-T</td>
<td>18.4$^*$</td>
<td>16.7$^*$</td>
</tr>
<tr>
<td>( \mathcal{E}\text{-ViLM} )</td>
<td>39.6</td>
<td>39.7</td>
</tr>
<tr>
<td>( \mathcal{E}\text{-ViLM} ) MVM</td>
<td>41.6</td>
<td>41.9</td>
</tr>
</tbody>
</table>

### Table 4. Zero-shot performances of \( \mathcal{E}\text{-ViLM} \) on text-to-video retrieval and multiple choice (MC) tasks. We report Top-1 Acc. on MC task and recall scores for retrieval task.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MSRVT</th>
<th>DiDeMo</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MC</td>
<td>text-to-video</td>
</tr>
<tr>
<td>HT100M</td>
<td>52.9</td>
<td>7.5 / 21.2 / 29.6</td>
</tr>
<tr>
<td>TACo</td>
<td>-</td>
<td>9.8 / 25.0 / 33.4</td>
</tr>
<tr>
<td>VideoCLIP</td>
<td>-</td>
<td>10.4 / 22.2 / 30.0</td>
</tr>
<tr>
<td>all-in-one-B</td>
<td>78.1</td>
<td>-</td>
</tr>
<tr>
<td>all-in-one-T</td>
<td>59.0</td>
<td>3.8 / 9.7 / 16.0$^*$</td>
</tr>
<tr>
<td>( \mathcal{E}\text{-ViLM} )</td>
<td>65.2</td>
<td>6.6 / 18.2 / 26.9</td>
</tr>
</tbody>
</table>


We choose HMDB51 [40] and Kinetics-400 datasets as our testbeds. HMDB51 contains 6,849 video clips extracted from commercial movies, divided into 51 human action categories, each containing a minimum of 101 clips. Kinetics-400 [37] dataset is a collection of 650,000 video clips that cover 400 human action classes. We conduct the classification training in a linear probing manner and report the Top-1 Acc. in Table 3. We observe that ViLM evidently outperforms all-in-one-T by a large margin: 41.6% vs. 18.4%, ViLM MVM denotes the pre-trained ViLM optimized with our masked video modeling. We observe that adopting MVM further improves the Top-1 Acc. by additional 2.0%. This verifies that the VL representations learned have great generalization ability, and the proposed \( L_{MVM} \) further strengthens that.

**Zero-shot Multiple Choice and Video Retrieval.** We now examine \( \mathcal{E} \text{-ViLM} \) from the perspective of zero-shot recognition and retrieval. We report performances \( \mathcal{E} \text{-ViLM} \) on both the MSRVTT and DiDeMo datasets, where the model is pre-trained and evaluated without any further fine-tuning. Table 4 shows the results of \( \mathcal{E} \text{-ViLM} \) with previous SOTA methods. \( \mathcal{E} \text{-ViLM} \) reaches 65.2% Top-1 accuracy on MSRVTT and 6.6/18.2/26.9 (R@1/R@5/R@10) for text-to-retrieval task, which is close to such larger VL architectures as VideoCLIP and TaCo. In comparison with all-in-one-T [76], our method noticeably exceeds it. This clearly verifies the great generalization ability of our proposed method.

<table>
<thead>
<tr>
<th>VTM</th>
<th>MLM</th>
<th>MVM</th>
<th>text-to-video</th>
<th>QA</th>
<th>MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>14.6 / 34.8 / 51.0</td>
<td>32.8</td>
<td>62.0</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>22.9 / 45.9 / 58.9</td>
<td>34.5</td>
<td>69.4</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>23.6 / 47.6 / 60.1</td>
<td>35.6</td>
<td>72.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>27.0 / 51.7 / 64.4</td>
<td>39.3</td>
<td>89.2</td>
</tr>
</tbody>
</table>

**Table 5.** Ablative studies for effects of different modeling items on text-to-video retrieval task. First row shows the results obtained w/o any pre-training (downstream task fine-tuning only).

Effect of MVM in VL Pre-training. In the furtherance of gaining more understanding of the impact of MVM throughout the different stages, we illustrate their performances on MSRVTT retrieval task in Table 5 and then compare the results w/o. MVM on HMDB and MSRVTT-MC tasks. We observe from the table that \( \mathcal{E} \text{-ViLM} \) with our proposed MVM clearly outperforms the counterpart (without MVM) and the combination of all modeling items leads to the best result. Also, we observe that the proposed MVM modeling is complementary to MLM and VTM across multiple tasks consistently. Tab 6 shows our comparison results leveraging different video tokenizers. We compare our optimized tokenizer to [48] (motion-level reconstruction), DALL-E (pixel-level reconstruction as [23]), and tokenizer optimized with BCE as reconstruction loss. Our focal-loss optimized tokenizer outperforms others on QA task (39.3 v.s. 38.5). \( \mathcal{E} \text{-ViLM} \) outperforms DALL-E’s pixel-level tokenizer to a large extent, confirming that semantically discretized labels are important for VL learning on small VL architectures. The table also shows the performance of [15, 81]’s official tokenizer and we observe a big performance gap (34.5 v.s. 41.6) with \( \mathcal{E} \text{-ViLM} \) mainly because of the lack of semantic optimization as \( \mathcal{E} \text{-ViLM} \). Different MVM target \( (L_{BCE}) \) suffers from imbalanced words distribution and yields a less vocabulary utilization rate and performances.

<table>
<thead>
<tr>
<th>Tokenizer</th>
<th>VOC. RATE</th>
<th>HMDB51</th>
<th>MSRVTT-QA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>-</td>
<td>33.9</td>
<td>37.9</td>
</tr>
<tr>
<td>( \mathcal{E} \text{-ViLM} + T_{DALL-E} )</td>
<td>-</td>
<td>34.2</td>
<td>38.1</td>
</tr>
<tr>
<td>( \mathcal{E} \text{-ViLM} + T_{BCE} )</td>
<td>-</td>
<td>34.5</td>
<td>38.3</td>
</tr>
<tr>
<td>( \mathcal{E} \text{-ViLM} + T_{BCE} )</td>
<td>64.2%</td>
<td>35.9</td>
<td>38.7</td>
</tr>
<tr>
<td>( \mathcal{E} \text{-ViLM} + T_{BCE} )</td>
<td>92.0%</td>
<td>41.6</td>
<td>39.3</td>
</tr>
</tbody>
</table>

**Table 6.** Performances of \( \mathcal{E} \text{-ViLM} \) with different tokenizers from [15,62,81] or obtained by different strategies. VOC. RATE denotes code-book utilization rate and higher value indicates a broader semantic space coverage.

5. Conclusion

In this paper, we propose the \( \mathcal{E} \text{-ViLM} \), a small Video-Language model that learns cross-modal representations efficiently and generalizes well to a series of VL downstream tasks. Compared with existing Video-Language models, \( \mathcal{E} \text{-ViLM} \) vastly increases the inference speed by over 40 times with only 15% of the parameters. With our proposed semantic vector-quantized video tokenizer, our VL pre-training is formulated as reconstructing the masked video region’s discrete labels as well as the masked language and video-text matching modeling. Extensive experiments indicate that \( \mathcal{E} \text{-ViLM} \) achieves competing performances, approaching most large Video-Language models and significantly outperforming its small counterpart. We anticipate that \( \mathcal{E} \text{-ViLM} \) will lead to more future works in building efficient Vision and Language models for broader real-world impact.
References


[22] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners, 2022. 3


[72] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training, 2022. 3, 6


[77] Jinpeng Wang, Yixiao Ge, Guanyu Cai, Rui Yan, Yudong Lin, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Object-aware


