An Empirical Study of End-to-End Video-Language Transformers with Masked Visual Modeling

Tsu-Jui Fu, Linjie Li, Zhe Gan, Kevin Lin, William Yang Wang, Lijuan Wang, Zicheng Liu

†UC Santa Barbara ‡Microsoft
{tsu-juifu, william}@cs.ucsb.edu
{lindsey.li, zhe.gan, keli, lijuanw, zli}@microsoft.com

Abstract

Masked visual modeling (MVM) has been recently proven effective for visual pre-training. While similar reconstructive objectives on video inputs (e.g., masked frame modeling) have been explored in video-language (VidL) pre-training, previous studies fail to find a truly effective MVM strategy that can largely benefit the downstream performance. In this work, we systematically examine the potential of MVM in the context of VidL learning. Specifically, we base our study on a fully end-to-end Video-Language Transformer (VIOLET) [15], where the supervision from MVM training can be backpropagated to the video pixel space. In total, eight different reconstructive targets of MVM are explored, from low-level pixel values and oriented gradients to high-level depth maps, optical flow, discrete visual tokens and latent visual features. We conduct comprehensive experiments and provide insights into the factors leading to effective MVM training, resulting in an enhanced model VIOLETv2. Empirically, we show VIOLETv2 pre-trained with MVM objective achieves notable improvements on 13 VidL benchmarks, ranging from video question answering, video captioning, to text-to-video retrieval.1

1. Introduction

Video, containing multiple modalities in nature, has been used as an epitome to test how AI systems perceive. Video-language (VidL) research aims at extending this ability to convey perception via language. Popular VidL tasks were introduced, such as text-to-video retrieval [29,54,75], video question answering [25,74], and video captioning [6,75]. Recent progresses in VidL learning mostly focus on VidL pre-training [49,58,83] with video-text matching [39,79] and masked language modeling [10]. There have also been attempts on similar masked modeling on vision inputs. For example, masked frame modeling [39] aims to recover masked frame representations. However, the pre-extracted video features cannot be refined during pre-training, which may limit its effectiveness. More recently, VIOLET [15] designs an end-to-end video-language transformer and proposes to reconstruct discrete visual tokens for masked frame patches. Though showing some promises in recovering visual semantics, the performance improvements on downstream VidL tasks are still marginal.

Meanwhile, self-supervised visual pre-training has been proven highly effective by reconstructing the masked image patches through raw pixel values [21,73], discrete visual tokens [3,81], or visual-semantic features [70,71]. However, they all only focus on the visual modality. It is unclear which variant of masked visual modeling (MVM) objectives can help VidL learning, especially given that the paired language inputs can already provide high-level semantics. Motivated by this, we conduct a comprehensive study of MVM for VidL learning. As illustrated in Figure 1, we base our study on the fully end-to-end Video-Language Transformer (VIOLET) [15], and study a broad spectrum of MVM targets, including RGB pixel values (Pixel), histogram of oriented gradients (HOG), depth maps (Depth), optical flow (Flow), discrete visual tokens (VQ), spatial-focused image features (SIF), temporal-aware video features (TVF), and multimodal features (MMF). During pre-training, we mask out some proportions of the video input along both spatial and temporal dimensions, and the model learns to recover the MVM targets for these masked patches. Equipped with another two standard pre-training tasks (i.e., video-text matching and masked language modeling), we empirically verify the effectiveness of different MVM variants on downstream VidL tasks.

Our study reveals that: (i) spatial-focused image features (SIF) is the most effective MVM target on video-text inputs; and (ii) the effects of different MVM targets on downstream VidL tasks are not shared between video-text and image-text inputs. For example, SIF extracted from the same model brings a large drop on downstream VidL

1Code has been released at https://github.com/tsujuifu/pytorch_empirical-mvm.
We systematically explore eight masked visual modeling (MVM) targets for end-to-end video-language (VidL) pre-training, including RGB pixel values (Pixel), histogram of oriented gradients (HOG), depth maps (Depth), optical flow (Flow), discrete visual tokens (VQ), spatial-focused image features (SIF), temporal-aware video features (TVF), and multimodal features from CLIP (MMF). Besides MVM, we pre-train VIOLET model [15] along with video-text matching (VTM) and masked language modeling (MLM).

In addition, we conduct comprehensive analyses of the masking strategy and ratio, combination of different MVM targets, to shed light on effective MVM training for VidL learning. We name the enhanced version of the original VIOLET [15] with the best MVM strategy as VIOLETv2.

Our contributions can be summarized as follows. We present an empirical study of masked visual modeling for video-language pre-training, with comprehensive analyses to reveal the ingredients for effective MVM training. VIOLETv2 with the best MVM recipe achieves strong performance on 13 VidL datasets. Concretely, compared to models pre-trained on the same 5M corpus, VIOLETv2 brings mean improvements of +5.4% accuracy on video question answering, +6.6% recall on text-to-video retrieval, and +11.4 CIDEr on video captioning. Direct comparison to VIOLET [15] also shows notable advantages of our model, even when pre-trained with much less data.

2. Related Work

Video-Language Understanding. Joint video-language (VidL) understanding [16,17,26,32,40,43,50] aims at interpreting the physical world via both vision and text perception. Researchers have explored such capability on VidL tasks including text-to-video retrieval [29,39,54,75], video question answering [25,35,36,74], moment retrieval [19,23,29,37], and video captioning [54,69,75,82]. Prior arts before the large-scale pre-training era [13,18,32,33,36,80] leverage offline extracted video features [1,5,9,14,22,27,30,67,72]. Later on, VidL pre-trained models [39,49,58,83] built on the above pre-extracted features have shown promising results. To enhance the performance, there have been parallel interests in bringing in more modalities from raw video inputs [16,42,55] and end-to-end training [2,34,48,79], aiming to elevate video representations.

Masked Visual Modeling (MVM). Aligned with the success of transformer-based language pre-training [31,44], image-text pre-training [7,59] and video-text pre-training [28,76,77] have shown promising results on diverse vision-language (VL) tasks. Popular VL pre-training tasks include visual-text matching (VTM) and masked language modeling (MLM), which are directly adapted from language pre-training [10]. Similar masked modeling on visual inputs [7,12,39] has also been introduced to VL pre-training, but are not as useful. Among the literature of vision pre-training itself, MAE [21,62] and SimMIM [73] reconstruct the pixels of the masked image patches to enhance visual representation. BEiT [3], iBOT [81], VIMPAC [60], and BEVT [68] adopt a BERT-like pre-training strategy to recover the missing visual tokens. On the other hand, MaskFeat [70] and MVP [71] consider latent features for MVM, including hand-crafted HOG features and image features extracted from pre-trained CLIP models [51]. Unlike previous studies exploring MVM on uni-modal data, in this study, we conduct a comprehensive investigation on how different MVM targets can help VidL learning.

The most relevant study to ours is VIOLET [15], which proposes to augment VidL pre-training with masked visual token modeling, while only showing marginal improvements on downstream performance. In contrast, our comprehensive investigation covers diverse MVM targets and
studies different combinations of masking strategies, which encompasses the design of MVM as well as shows large performance improvements on downstream VidL tasks.

3. Method

We first describe the base model VIOLET in Section 3.1, and then introduce the problem formulation of our investigation in Section 3.2. Section 3.3 discusses eight different target features for masked visual modeling (MVM).

3.1. End-to-End Video-Language Transformer

We conduct our empirical study using an end-to-end VIdeO-LanguagE Transformer (VIOLET) [15], with 3 components: Video Swim Transformer (VT), Language Embedder (LE), and Cross-modal Transformer (CT). VIOLET takes video \( V \) and sentence \( X \) as inputs. Sparse-sampled frames \( \{f_1, f_2, \ldots \} \) from \( V \) are first segmented into a set of video patches, and then processed by VT to compute video features \( v = \{v_1, v_2, \ldots\} \). LE extracts the word embeddings \( w = \{w_1, w_2, \ldots\} \) for each word token \( \{x_1, x_2, \ldots\} \) in \( X \). Then, CT performs cross-modal fusion on top of \( v \) and \( w \) to produce joint VidL representations \( h = [h^v, h^c, h^r] \), where \( h^v, h^c, h^r \) denote the hidden representations of video patches, the special \([CLS]\) token, and other word tokens.

3.2. Problem Setting

Given a large-scale video-language (VidL) dataset \( D \), we aim to pre-train a VidL transformer to learn effective video-text representations. The learned representations can be transferred to downstream tasks for performance improvement. Different from existing works that focus on MVM for pure vision problems [3, 21, 56], we study MVM as a VidL pre-training task. Given a video-text pair \((V, X)\) where \( V \) is a sequence of video frames and \( X \) is a sequence of word tokens. As shown in Figure 1, we randomly mask out some portions of the input frames \( V \), and learn to predict the target features corresponding to the masked patches. To output a correct prediction, the model will have to resort to other relevant video frames \( V \) and/or text tokens \( X \). This facilitates cross-modality learning for better VidL understanding.

In addition, we employ the commonly used VidL pre-training objectives, including video-text matching (VTM) and masked language modeling (MLM), where VTM aims to predict whether an input video-text pair is matched or not, while MLM aims to predict the masked word tokens from the surrounding context.\(^2\) Our overall pre-training objective can be written as:

\[
\mathcal{L} = \mathcal{L}_{\text{MVM}} + \mathcal{L}_{\text{VTM}} + \mathcal{L}_{\text{MLM}},
\]

where \( \mathcal{L}_{\text{MVM}}, \mathcal{L}_{\text{VTM}}, \mathcal{L}_{\text{MLM}} \) are the MVM, VTM and MLM objectives, respectively.

\(^2\)Refer to the Appendix for detailed formulation of VTM and MLM.

3.3. Target Features

Masked visual modeling (MVM) is a generic masked feature prediction task, where we mask out some of the visual input patches, and then predict the target features corresponding to the masked ones. Thus, a core design of MVM is the target features, which enables VIOLET learning a desired aspect of visual modeling. While MVM has been explored in pure vision tasks [3, 21, 56], it remains an open question whether MVM can facilitate the interactions between video and language modalities. In this study, we investigate what design of MVM is effective in the context of video-language pre-training?

Following [56, 73], we employ a simple linear layer or 2-layer MLP as the prediction head for MVM, to project the hidden video representations \( (h^v, \text{ of hidden size } 768) \) from CT to the same dimension as the MVM targets. The default MVM loss is the \( l_1 \) loss, unless specified otherwise. Next, we introduce the considered target features in details.

RGB Pixel Values (Pixel). We treat the normalized RGB pixel values as a candidate target feature. During MVM, VIOLET learns to reconstruct the pixel values of the masked patches. The linear MVM head projects \( h^v \) into the same dimension as the raw video frame patch \((H \times W \times 3)\).

Histogram of Oriented Gradients (HOG). HOG [8] is a pioneer feature descriptor that describes the gradients of orientations of the image. While HOG has been proven effective for visual pre-training [56], it is unknown whether it can benefit VidL pre-training. We extract HOG features in a dense grid level, and use such feature descriptors as the prediction targets of MVM. The HOG feature map is of the same size as the input video frame, but with channel size 1. The linear MVM prediction head projects \( h^v \) to the same dimension as HOG for the video frame patch \((H \times W \times 1)\).

Depth Maps (Depth). Since depth maps usually contain finer-grained details of the object shapes and general scene layout of the foreground objects, it is worth exploring whether depth maps can be used to improve the scene/object understanding capability of a VidL pre-trained model. To obtain such MVM target, we employ a pre-trained dense prediction transformer (DPT) [53] to perform monocular depth estimation given an input video frame. The linear prediction head used for Depth is the same as the one for HOG, as both targets are of channel size 1.

Optical Flow (Flow). Optical flow is commonly used in motion analysis and video understanding. Here, we analyze whether apparent velocity of objects can benefit VidL pre-training. We employ a pre-trained recurrent all-pairs field transforms (RAFT) [61] to compute optical flow given the consecutive video frames. We directly use the estimated optical flow values as the prediction target, and supervise the MVM training with \( l_1 \) loss. To obtain the MVM predictions, we concatenate the hidden video representations.
Table 1. Comparing target features for MVM applied to video-text data. All variants are pre-trained on WebVid [2] for 5 epochs. Masking is performed randomly (RM) with ratio of 15%. The final pre-training setting is highlighted in gray.

<table>
<thead>
<tr>
<th>Pre-training Tasks</th>
<th>MVM Target</th>
<th>TGIF-Frame Acc.</th>
<th>DiDeMo-Retrieval R1</th>
<th>R5</th>
<th>R10</th>
<th>AveR</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTM+MLM</td>
<td>None</td>
<td>68.1 (+0.2)</td>
<td>28.7</td>
<td>57.0</td>
<td>69.7</td>
<td>51.8</td>
</tr>
<tr>
<td></td>
<td>RGB Pixel Values</td>
<td>68.5 (+0.3)</td>
<td>29.2 (+0.5)</td>
<td>58.6 (+1.6)</td>
<td>70.1 (+0.4)</td>
<td>52.6 (+0.8)</td>
</tr>
<tr>
<td></td>
<td>Histogram of Oriented Gradients [8]</td>
<td>67.3 (+0.8)</td>
<td>26.6 (-2.1)</td>
<td>54.9 (-2.1)</td>
<td>68.1 (-1.6)</td>
<td>49.8 (-2.0)</td>
</tr>
<tr>
<td>+MVM</td>
<td>Depth Maps (DPTF-L [53])</td>
<td>68.0 (+0.1)</td>
<td>27.3 (+1.4)</td>
<td>55.0 (+2.0)</td>
<td>68.3 (+1.4)</td>
<td>50.2 (+1.6)</td>
</tr>
<tr>
<td></td>
<td>Optical Flow (RAFT-L [61])</td>
<td>67.6 (-0.5)</td>
<td>30.3 (+1.6)</td>
<td>58.0 (+1.0)</td>
<td>70.3 (+0.3)</td>
<td>52.9 (+1.1)</td>
</tr>
<tr>
<td></td>
<td>Spatial-focused Image Features (Swin-B [45])</td>
<td>68.8 (+0.7)</td>
<td>35.4 (+6.7)</td>
<td>62.4 (+5.2)</td>
<td>74.9 (+6.3)</td>
<td>57.6 (+5.8)</td>
</tr>
<tr>
<td></td>
<td>Temporal-aware Video Features (ViDSwin-L [46])</td>
<td>68.0 (+0.1)</td>
<td>32.8 (+4.1)</td>
<td>60.5 (+3.5)</td>
<td>73.0 (+3.3)</td>
<td>55.4 (+3.6)</td>
</tr>
<tr>
<td></td>
<td>Discrete Visual Tokens (DALL-E [52])</td>
<td>68.4 (+0.3)</td>
<td>28.1 (-0.6)</td>
<td>56.6 (-0.4)</td>
<td>69.4 (-0.5)</td>
<td>51.3 (-0.5)</td>
</tr>
<tr>
<td></td>
<td>Multimodal Features (CLIP-ViT-B [51])</td>
<td>67.7 (-0.4)</td>
<td>30.8 (+1.1)</td>
<td>57.8 (+0.8)</td>
<td>68.5 (-1.2)</td>
<td>52.1 (+0.3)</td>
</tr>
</tbody>
</table>

computed by CT on consecutive frames, and employ a linear layer to project the concatenated video representations (of hidden size $768 \times 2$) to the same dimension as the estimated optical flow target for a given patch ($H \times W \times 2$).

### Discrete Visual Tokens (VQ)
In addition to continuous MVM targets, we also consider the discrete variational autoencoder (dVAE) [52, 64] to quantize video inputs. dVAE is learned to tokenize images into discrete visual tokens $q$ from a finite dictionary, and then reconstruct the original visual scene based on $q$, where $q$ should have a one-to-one correspondence with the input image patches spatially. We first adopt dVAE to tokenize the $t^{th}$ video frame $f_t$ into $q_t$: $q_t = dVAE(f_t)$, and then a 2-layer MLP is used to project $h_v$ into the finite VQ vocabularies. As VQ token is discrete, we can model MVM with VQ as a classification problem, and adopt the cross-entropy loss to optimize the MVM training, following [3, 15].

### Spatial-focused Image Features (SIF)
We investigate whether image features can be useful for improving VidL pre-training. We employ a well-known vision transformer (such as Swin Transformer [45]) to extract the grid features given an input image. We then normalize the extracted grid features and consider them as ground-truth MVM targets. In the following sections, we conduct comprehensive investigation over MVM targets described above, and perform detailed analysis on MVM strategies. To avoid confusion, we denote the strongest model with the most effective MVM training as VIOLETv2.

### 4. Study: Target Features for MVM

#### Settings
We conduct pre-training on WebVid-2.5M [2] for 5 epochs, and report accuracy on TGIF-Frame [25] for video question answering and R1/R5/R10/AveR on DiDeMo [24] for text-to-video retrieval. We initialize our Video Swin Transformer (VT) with VideoSwin-Base [46], pre-trained on Kinetics-600 [27]. Language Embedder (LE) and Cross-modal Transformer (CT) are initialized from pre-trained BERT-Base [10]. During pre-training, we sparsely sample 4 video frames and randomly crop them into 224x224 to split into patches with $H = W = 32$. For all downstream tasks, we adopt the same video frame size and patch size but 5 sparse-sampled frames. We keep the training recipe (e.g., optimizer settings, masking ratio, training schedule, etc.) consistent across all targets, which we find generally good in practice. For MVM targets that involve a teacher model, we use official models released by the authors. We compare models pre-trained with 8 different MVM variants to the baseline pre-trained with only VTM and MLM. Our goal is to find the best MVM target features that can provide the largest performance improvement over this baseline. Results are summarized in Table 1. We first categorize the MVM targets into 4 groups, and discuss their performance in details.

#### One-stage Visual Targets
We include Pixel and HOG, as they do not require training a deep neural network in advance to extract these features. Compared to the baseline without MVM objective, regressing the explicit RGB colors contributes to a relatively small gain of +0.2% on TGIF-Frame and +0.8% on AveR for DiDeMo Retrieval. In

We base our ablation experiments on these two representative datasets for fast iteration, our main results are reported on 13 benchmarks in Section 6. Details about downstream adaptation are included in the Appendix. Refer to the Appendix for more on training details.
contrast, HOG renders degradation on downstream video-language (VidL) performance (-0.8% on TGIF-Frame and -2.0% on DiDeMo-Retrieval). We hypothesize that this is due to the missing color information in HOG features, which is critical in VidL understanding.

**Supervised Pseudo-label Targets.** We include *Depth Maps (Depth)* and *Optical Flow (Flow).* Intuitively, Depth and Flow can be considered as continuous pseudo “labels”, which are made by models trained to perform depth and optical flow estimation \[53,61\]. Depth does not improve over baseline with VTM+MLM. The nature of depth maps are to separate the foreground from the background, thus may guide the model to ignore information from the background, even when they are relevant for solving downstream VidL tasks (-0.1% on TGIF-Frame, -1.6% on DiDeMo Retrieval). Flow only focuses on the moving part between frames, while ignores the spatial details of static components, thus fail on more spatially-focused TGIF-Frame task (-0.5%). We also find that the optical flow estimation model easily fails with sparse sampling strategy, which is widely adopted in VidL pre-training.\(^5\)

**Supervised Visual Feature Targets.** We include continuous features extracted from the last layers of image classification model \[45\] (*i.e.*, Spatial-focused Image Features (SIF)) and action recognition model \[46\] (*i.e.*, Temporal-aware Video Features (TVF)). We consider regressing supervised features from Swin-B or VidSwin-L\(^6\) as a type of knowledge distillation from unimodal models to our model. SIF achieves significant improvement over baseline (+0.7% on TGIF-Frame and +5.8% on AveR for DiDeMo-Retrieval). In contrast, TVF fails to improve TGIF-Frame accuracy (-0.1%), though it brings notable improvement on retrieval performance (+3.6% on AveR). By distilling the knowledge from Swin-B, we enforce the model to focus more on spatial details of each frame, which we hypothesize is the main reason behind the large performance improvement. As previous study \[4\] pointed out, existing VidL benchmarks largely test on spatial understanding about the key frame of the video, with only a fractional of examples actually testing on temporal reasoning over multiple frames.

**Self-supervised Multimodal Feature Targets.** We use

---

\(^5\)Please find visualization examples in the Appendix.

\(^6\)VidSwin-L is trained on Kinetics-400 \[27\] with 83.1% accuracy.

---

### Table 2. Combining MVM targets. All variants are pre-trained on WebVid \[2\] for 5 epochs, using RM with 15% as the masking strategy. We highlight the final setting in gray.

<table>
<thead>
<tr>
<th>MVM Targets</th>
<th>TGIF-Frame</th>
<th>DiDeMo-Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc. R1 R5 R10 AveR</td>
<td>Acc. R1 R5 R10 AveR</td>
</tr>
<tr>
<td>Pixel</td>
<td>68.3</td>
<td>29.2</td>
</tr>
<tr>
<td>Flow</td>
<td>67.6</td>
<td>30.3</td>
</tr>
<tr>
<td>SIF</td>
<td><strong>68.8</strong></td>
<td><strong>35.4</strong></td>
</tr>
<tr>
<td>SIF + Pixel</td>
<td>68.8</td>
<td>31.8</td>
</tr>
<tr>
<td>SIF + Flow</td>
<td>68.7</td>
<td>34.4</td>
</tr>
</tbody>
</table>

### Table 3. Comparing different image feature targets for MVM. All variants are pre-trained on WebVid \[2\] with VTM+MLM+MVM (SIF) for 5 epochs, using RM with 15% as the masking strategy. The final pre-training setting is highlighted in gray.

<table>
<thead>
<tr>
<th>Image Features</th>
<th>Train IN-1K ACC@1</th>
<th>TGIF-Frame</th>
<th>DiDeMo-Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Acc. R1 R5 R10 AveR</td>
<td>Acc. R1 R5 R10 AveR</td>
<td></td>
</tr>
<tr>
<td>ResNet-50 [22]</td>
<td>In-1K 76.1</td>
<td>67.3</td>
<td>29.1</td>
</tr>
<tr>
<td>Swin-T [45]</td>
<td>In-1K 81.2</td>
<td>68.9</td>
<td>33.8</td>
</tr>
<tr>
<td>Swin-B</td>
<td>In-1K 83.5</td>
<td>68.3</td>
<td>34.9</td>
</tr>
<tr>
<td>Swin-B</td>
<td>In-22K 85.2</td>
<td>68.8</td>
<td>35.4</td>
</tr>
<tr>
<td>Swin-L</td>
<td>In-22K 86.3</td>
<td>68.2</td>
<td>33.2</td>
</tr>
</tbody>
</table>

**Discrete Visual Tokens (VQ)** from DALL-E \[52\] and continuous *Multimodal Features (MMF)* extracted from CLIP \[51\]. Both are pre-trained on large-scale image-text datasets, usually much more expensive than all other targets. Both targets improve the performance by a slight margin on only one task. VQ that can capture patch-level semantics, benefits TGIF-Frame (+0.3%) which mostly focuses on scene understanding. While MMF from CLIP, contrastively pre-trained to measure the high-level similarity between the entire image and text sentence, is helpful for DiDeMo-Retrieval (+0.3% on AveR).

**Summary.** We hypothesize that many factors could lead to the low performance of an MVM target, such as its own characteristics (*e.g.*, local vs. global features); the target model; the loss design; or the mismatch between pre-train objectives and downstream focus. We try our best to compare them rigorously with controlled experiments to find the best setting. Based on our experiments, regressing RGB values (Pixel) and distilling features from Swin-B \[45\] (SIF) are the only two that produce consistent gains over the baseline on both downstream tasks. MVM with SIF achieves the best performance, with a gain of +0.7% on TGIF-Frame and +5.8% on AveR for DiDeMo-Retrieval over the baseline. Therefore, we use SIF as the default target for MVM in the following sections, unless specified otherwise.

### 5. Analyses of MVM

**Combining MVM Targets.** As different MVM targets focus on different aspects of visual modeling, a naive way to enable models with different visual capabilities is to combine them together. Specifically, the model pre-training can be supervised by more than one MVM loss, which is simply added together to be backpropagated. In Table 2, we find there is no merit in combining different MVM targets, leading to worse downstream performance than using SIF alone. When combining the best two targets found in Table 1: Pixel+SIF, it performs better than Pixel only, but does not improve over using SIF alone. We hypothesize that the explicit details of pixel values may conflict with the high-level visual semantics summarized in the grid features from the image classifier. We further try to combine SIF with Flow in the hope of enforcing both temporal and spatial reasoning over video inputs. In addition, Flow is a better candidate than other targets, as it demonstrates some advantages
on retrieval performance in Table 1, and it is a different type of target from SIF, compared to temporal-aware video features. The results are consistent, with improvements over optical flow only; while the performance drops, compared to SIF alone. Though our results are not encouraging, we believe how to effectively combine different MVM targets is an interesting direction for future study.

**MVM Target Extractors vs. Downstream Performance.** In Table 3, we explore different image classification models as the MVM target extractor for SIF, and investigate whether stronger image classification model enables better VidL performance. We compare ResNet-50 [22], Swin-Tiny/Base/Large [45], trained on ImageNet-1K (IN1K) or ImageNet-22K (IN-22K) [9], and summarize the observations below:

- ResNet-50 performs lower than Swin variants. Two potential reasons are (i) ResNet-50 architecture is very different from ViDStwin (i.e., with different inductive bias); and (ii) the much lower ImageNet performance (~76 vs. >81) suggest the ResNet-50 features are not as strong.

- When the target model shares similar inductive biases to the video encoder (i.e., Swin-T/B/L), the downstream performance is not directly proportional to ImageNet accuracy, and is overall better than that of Res50. This suggests that the architecture design of both target model and video encoder should be similar.

- A key difference between different Swin targets is the feature dimension (768/1024/1568 for Swin-T/B/L), while the video tokens from CT are of size 768. Although we project them into the same dimension as the targets, the mismatch may lead to slightly lower performance (with Swin-L especially).

In short, we believe a SIF target model should share similar inductive biases as the video encoder.

**Masking Strategy.** We investigate the effect of different masking strategies in Table 4, including random masking (RM), blockwise masking (BM), attended masking (AM), and their combinations.

- **Random Masking (RM).** Following the conventional practice in MLM, we randomly select a certain percentage $p_m$ of video frame patches from the whole video inputs to be masked. In Table 5, we explore different masking ratios ($p_m$), and empirically find $p_m = 30\%$ gives the best downstream performance.

- **Blockwise Masking (BM).** To make MVM relying less on similar neighbor patches, we adopt blockwise masking [3, 60] that masks blocks of video patches along spatial-temporal dimension rather than independently masking randomly sampled patches for each frame. Specifically, we randomly sample an ($H', W', T'$) as a masking block, where all $H' \times W'$ visual patches in the following $T'$ consecutive frames will be masked; we repeat this process until $>p_m$ of video patches are masked to perform MVM pre-training.

- **Attended Masking (AM).** Attended masking tries to put more weights on the more important elements based on the attention weights computed by Cross-modal Transformer (CT). A similar idea has been explored in [79] for MLM. Here, we extend AM to both visual and textual modalities. We first keep the video-text inputs intact, feed them into CT to compute the attention weights, to decide which portions in video and text are more important. We then select the top $p_m$ of most-attended patches/tokens to be masked in video-text inputs for MVM and MLM.

To combine different masking strategies, we randomly apply one masking method for each video-text pair in a batch. Results in Table 4 suggest that TGFIFrame can slightly benefit from BM, and combining BM with RM leads to the best retrieval performance on DiDeMo. As video usually presents analogous visual patterns in spatial-temporal neighbors (i.e., nearby patches within current frame or neighboring frames), when masking patches independently (i.e., RM), these neighbors can make the masked patches easy to recover, and may lead to spurious success in MVM evaluation. By masking a block (i.e., BM) instead of individual patches, the model cannot merely rely on similar neighboring visual cues but requires actual visual reasoning to recover a group of missing patterns. Combining BM with RM leads to more diverse dropout patterns in video inputs, which is in analogy to data augmentation.

In addition, AM and combinations with AM are not effective for both downstream tasks. It is also worth noting that AM greatly increase the training time (4 times more than RM/BM), due to the additional forward pass needed.

<table>
<thead>
<tr>
<th>Masking Strategy</th>
<th>Time Cost (hours)</th>
<th>TGFIFrame Acc.</th>
<th>DiDeMo-Retrieval R1</th>
<th>R5</th>
<th>R10</th>
<th>AveR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM</td>
<td>8.0</td>
<td>68.8</td>
<td>33.4</td>
<td>62.4</td>
<td>74.9</td>
<td>57.6</td>
</tr>
<tr>
<td>BM</td>
<td>8.0</td>
<td>69.0</td>
<td>35.9</td>
<td>63.3</td>
<td>74.6</td>
<td>57.9</td>
</tr>
<tr>
<td>AM</td>
<td>34.5</td>
<td>68.4</td>
<td>31.5</td>
<td>59.9</td>
<td>72.0</td>
<td>54.7</td>
</tr>
<tr>
<td>RM+BM</td>
<td>8.0</td>
<td>68.7</td>
<td>36.4</td>
<td>64.2</td>
<td>74.4</td>
<td>58.3</td>
</tr>
<tr>
<td>RM+AM</td>
<td>20.5</td>
<td>68.8</td>
<td>33.7</td>
<td>63.2</td>
<td>73.5</td>
<td>56.8</td>
</tr>
<tr>
<td>BM+AM</td>
<td>20.5</td>
<td>69.9</td>
<td>35.6</td>
<td>61.9</td>
<td>74.4</td>
<td>57.3</td>
</tr>
<tr>
<td>RM+BM+AM</td>
<td>17.0</td>
<td>68.6</td>
<td>34.7</td>
<td>62.0</td>
<td>74.8</td>
<td>57.2</td>
</tr>
</tbody>
</table>

**Table 4. Impact of masking strategy of MVM.** All variants are pre-trained on WebVid [2] with VTM+MLM+MVM (SIF) for 5 epochs. The masking ratio is set as 15% for all masking strategies. The final pre-training setting is highlighted in gray.

<table>
<thead>
<tr>
<th>$p_m$</th>
<th>TGFIFrame Acc.</th>
<th>DiDeMo-Retrieval R1</th>
<th>R5</th>
<th>R10</th>
<th>AveR</th>
</tr>
</thead>
<tbody>
<tr>
<td>15%</td>
<td>68.8</td>
<td>35.4</td>
<td>62.4</td>
<td>74.9</td>
<td>57.6</td>
</tr>
<tr>
<td>30%</td>
<td>68.8</td>
<td>36.2</td>
<td>64.0</td>
<td>74.5</td>
<td>58.2</td>
</tr>
<tr>
<td>45%</td>
<td>68.9</td>
<td>35.6</td>
<td>61.9</td>
<td>74.4</td>
<td>57.3</td>
</tr>
<tr>
<td>60%</td>
<td>68.1</td>
<td>34.1</td>
<td>63.9</td>
<td>74.6</td>
<td>57.5</td>
</tr>
<tr>
<td>75%</td>
<td>68.3</td>
<td>35.4</td>
<td>62.4</td>
<td>74.2</td>
<td>57.3</td>
</tr>
</tbody>
</table>

**Table 5. Impact of masking ratio of MVM.** All variants are pre-trained on WebVid [2] with VTM+MLM+MVM (SIF) for 5 epochs, using RM as the masking strategy. The final pre-training setting is highlighted in gray.
Table 6. Comparing target features for MVM applied to image-text data. All variants are pre-trained on CC3M [57] for 5 epochs. Masking is performed randomly (RM) with ratio of 15%.

<table>
<thead>
<tr>
<th>Pre-training Tasks</th>
<th>MVM Target</th>
<th>TGIF-Frame</th>
<th>DiDeMo-Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Acc.</td>
<td>R1</td>
</tr>
<tr>
<td>VTM+MLM</td>
<td>69.8</td>
<td>36.4</td>
<td>64.3</td>
</tr>
<tr>
<td>+MVM</td>
<td>RGB Pixel Values</td>
<td>69.7 (-0.1)</td>
<td>35.8 (-0.6)</td>
</tr>
<tr>
<td></td>
<td>Histogram of Oriented Gradients [8]</td>
<td>69.8</td>
<td>34.9 (-1.5)</td>
</tr>
<tr>
<td></td>
<td>Depth Maps (DPT-L [53])</td>
<td>69.6 (-0.2)</td>
<td>32.3 (-4.1)</td>
</tr>
<tr>
<td></td>
<td>Spatial-focused Image Features (Swin-B [45])</td>
<td>69.7 (-0.1)</td>
<td>31.6 (-4.8)</td>
</tr>
<tr>
<td></td>
<td>Discrete Visual Tokens (DALL-E [52])</td>
<td>69.8</td>
<td>34.4 (-2.0)</td>
</tr>
<tr>
<td></td>
<td>Multimodal Features (CLIP-ViT-B [51])</td>
<td>69.8</td>
<td>33.6 (-2.8)</td>
</tr>
</tbody>
</table>

Table 7. Combining MVM target features for both video-text and image-text data. All variants are pre-trained on WebVid2.5M [2] + CC3M [57] for 5 epochs. The final pre-training setting is highlighted in gray.

<table>
<thead>
<tr>
<th>Pre-training Tasks</th>
<th>MVM Target</th>
<th>TGIF-Frame</th>
<th>DiDeMo-Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Acc.</td>
<td>R1</td>
</tr>
<tr>
<td>VTM+MLM</td>
<td>None</td>
<td>69.7</td>
<td>36.7</td>
</tr>
<tr>
<td>+MVM</td>
<td>Spatial-focused Image Features (Swin-B [45])</td>
<td>None</td>
<td>71.1</td>
</tr>
<tr>
<td></td>
<td>Spatial-focused Image Features (Swin-B)</td>
<td>Pixel</td>
<td>71.3</td>
</tr>
</tbody>
</table>

6. Main Results

To this end, we combine the most effective MVM strategies to pre-train VIOLETv2 and evaluate on 13 video-language (VidL) tasks. Table 8 shows the comparison to prior arts on video question answering (QA) and video captioning. We observe that VIOLETv2 is effective in learning transferable knowledge for the downstream tasks. For example, considering pre-training data at a similar scale (i.e., ≤ 5M, the top rows of Table 8), VIOLETv2 achieves better results than prior arts, including ALPRO [38], ClipBERT [34], and SwinBERT [41], across all considered video QA and video captioning benchmarks. Specifically, when pre-training with the exact same data (i.e., WebVid2.5M [2] + CC3M [57]), VIOLETv2 surpasses ALPRO by 2.4% accuracy on MSRVTT-QA and 8.4% accuracy on MSVD-QA, respectively. We also compare with other models pre-trained on significantly larger scale of video-text pairs. As shown in the bottom rows of Table 8, although...
we use less pre-training data than others, VIOLETv2 still achieves comparable or better performance.

We observe similar findings on video captioning. On MSRVTT captioning, VIOLETv2 is only 2 points behind MV-GPT [56] pre-trained with 53M video-text pairs, which is 10 times larger than ours (5M). In addition, MV-GPT leverages ASR transcripts to enhance the captioning performance, while our captioning model takes only video frames as inputs and outputs the video caption. We believe augmenting VIOLETv2 with additional modalities, such as audio or ASR transcripts, can further improve captioning performance, which we leave as future work.

Table 9 presents the comparison on text-to-video retrieval. When pre-training with the same datasets (i.e., WebVid2.5M [2] + CC3M [57]), VIOLETv2 shows across-the-board improvements with all metrics considered on DiDeMo and LSMDC. It is worth noting that our method performs comparably to BridgeFormer [20] on MSRVTT-Retrieval. BridgeFormer adopts a noun/verb masking strategy during pre-training, which is specially aligned to the simple sentences in MSRVTT. However, it cannot show similar effects on DiDeMo and LSMDC due to more complex texts with multiple nouns/verbs. In contrast, the studied MVM can achieve a comprehensive enhancement in VidL learning and lead to notable improvements (+10.9% R1 on DiDeMo and +6.1% R1 on LSMDC).

Direct Comparison to VIOLET [15]. Across Table 8 and 9, it is worth noting that VIOLETv2 outperforms VIOLET with notable margins, even when VIOLET is pre-trained with significantly more data (about 37 times more). Specifically, VIOLETv2 yields an average gain of +3.4% across 8 video QA datasets, and an absolute gain of +8.6% on R1 across all three retrieval benchmarks. These results suggest the importance of an appropriate MVM setting, which is the core belief in our study.

7. Conclusion

We initiate the first empirical study on adopting masked visual modeling (MVM) for video-language (VidL) learning. We explore diverse MVM objectives upon end-to-end Video-Language Transformer (VIOLETv2), including low-level pixel space, high-level visual semantics, and extracted latent features. Our results show that VIOLETv2 pre-trained on 5M video/image-text data with MVM objective achieves strong performance on 3 popular VidL tasks across 13 VidL benchmarks. Our comprehensive analyses on different combinations of MVM targets, various SIF target extractors, and varying masking strategies/ratios shed light on effective MVM design. We believe our study can guide future research on large-scale VidL pre-training and wish to study how MVM can generalize to larger-scale data. In addition, we vision that with the emergence of video/VidL foundation models in future works, better choices of MVM targets can be explored.
References


Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-Captions in Events. In International Conference on Computer Vision (ICCV), 2017. [1, 2]


[54] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A Dataset for Movie Description. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 8


[74] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwong Zhang, Xiangnan He, and Yueting Zhuang. Video Question Answering via Gradually Refined Attention over Appearance and Motion. In *ACM Multimedia (ACMMM)*, 2017. 1, 2, 8

[75] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 8


[81] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT Pre-Training with Online Tokenizer. In International Conference on Learning Representations (ICLR), 2012. 1, 2, 3
