

000 001 002 003 004 005 RICHSPACE: ENRICHING TEXT-TO-VIDEO PROMPT 006 SPACE VIA TEXT EMBEDDING INTERPOLATION 007 008 009

010 **Anonymous authors**
011 Paper under double-blind review
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053

ABSTRACT

Text-to-video generation models have made impressive progress, but they still struggle with generating videos with complex features. This limitation often arises from the inability of the text encoder to produce accurate embeddings, which hinders the video generation model. In this work, we propose a novel approach to overcome this challenge by selecting the optimal text embedding through interpolation in the embedding space. We demonstrate that this method enables the video generation model to produce the desired videos. Additionally, we introduce a simple algorithm using perpendicular foot embeddings and cosine similarity to identify the optimal interpolation embedding. Our findings highlight the importance of accurate text embeddings and offer a pathway for improving text-to-video generation performance.

1 INTRODUCTION

Text-to-video models have developed rapidly in recent years, driven by the advancement of Transformer architectures (Vaswani, 2017) and diffusion models (Ho et al., 2020). Early attempts at text-to-video generation focused on scaling up Transformers, with notable works such as CogVideo (Hong et al., 2022) and Phenaki (Villegas et al., 2022), which demonstrated promising results. More recently, the appearance of DiT (Peebles & Xie, 2023), which incorporates Transformers as the backbone of Diffusion Models, has pushed the capabilities of text-to-video generation models to new heights. Models like Sora (OpenAI, 2024), MovieGen (Meta, 2024), CogVideoX (Yang et al., 2024), and Veo 2 (Google, 2024) have further showcased the potential of these approaches. Despite the impressive progress made in recent years, current state-of-the-art text-to-video generation models still face challenges in effectively following complex instructions in user-provided text prompts. For example, when users describe unusual real-world scenarios, such as “a tiger with zebra-like stripes walking on grassland,” the text encoder may struggle to fully capture the intended meaning. This results in text embeddings that fail to guide the video generation model toward producing the desired output. This issue is also observed in the text-to-image generation domain, where a notable work, Stable Diffusion V3 (Esser et al., 2024), addresses this challenge by incorporating multiple text encoders to improve understanding. Although their approach, which combines embeddings from different encoders, yields effective results, it comes at a significant computational cost due to the need to compute embeddings from multiple sources.

In this work, we first study the problem that prompt space is not enough to cover all video space from a theoretical perspective. We provide an informal theorem of our theoretical findings as follows:

Theorem 1.1 (Word Embeddings being Insufficient to Represent for All Videos, informal version of Theorem 4.9). *Let n, d denote two integers, where n denotes the maximum length of the sentence, and all videos are in \mathbb{R}^d space. Let $V \in \mathbb{N}$ denote the vocabulary size. Let $\mathcal{U} = \{u_1, u_2, \dots, u_V\}$ denote the word embedding space, where for $i \in [V]$, the word embedding $u_i \in \mathbb{R}^k$. Let $\delta_{\min} = \min_{i,j \in [V], i \neq j} \|u_i - u_j\|_2$ denote the minimum ℓ_2 distance of two word embedding. Let $f : \mathbb{R}^{nk} \rightarrow \mathbb{R}^d$ denote the text-to-video generation model, which is also a mapping from sentence space (discrete space $\{u_1, \dots, u_V\}^n$) to video space \mathbb{R}^d . Let $M := \max_x \|f(x)\|_2$, $m := \min_x \|f(x)\|_2$. Let $\epsilon = ((M^d - m^d)/V^n)^{1/d}$. Then, we can show that there is a video $y \in \mathbb{R}^d$, satisfying $m \leq \|y\|_2 \leq M$, such that for any sentence $x \in \{u_1, u_2, \dots, u_V\}^n$, $\|f(x) - y\|_2 \geq \epsilon$.*

Additionally, we take a different approach by exploring whether we can obtain a powerful text embedding capable of guiding the video generation model through interpolation within the text em-

bedding space. Through empirical experiments, we demonstrate that by selecting the optimal text embedding, the video generation model can successfully generate the desired video. Additionally, we propose an algorithm that takes advantage of perpendicular foot embeddings and cosine similarity to capture both global and local information in order to identify the optimal embedding of interpolation text (Fig. 1 and Algorithm 1).

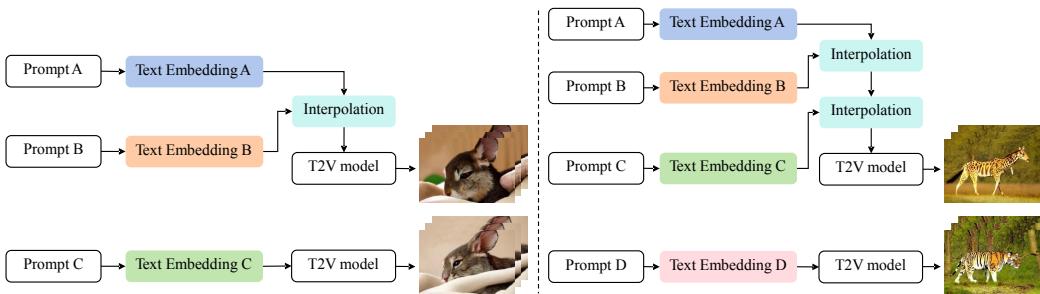


Figure 1: Two kinds of Text Prompts Mixture. **Left: Mixture of Two Prompts.** We set two prompts, A and B, and apply linear interpolation to two corresponding text embeddings. After that, we use one of the interpolation results to generate a video. To evaluate the effect of video interpolation, we set another prompt C, which describes the generated video to generate a video to compare with the interpolated video. **Right: Mixture of Three Prompts.** We set two prompts A and B and apply linear interpolation to two corresponding text embeddings. We manually choose one text embedding interpolated from A and B, then apply linear interpolation to this text embedding and text embedding C. After that, we use one of the interpolation results to generate a video. To evaluate the effect of video interpolation, we set another prompt D which describes the generated video to generate a video to compare with the interpolated video.

In summary, our main contributions are as follows:

- We demonstrate that selecting the correct text embedding can effectively guide a video generation model to produce the desired video.
- We propose a simple yet effective algorithm to find the optimal text embedding through the use of perpendicular foot embeddings and cosine similarity.

Roadmap. Our paper is organized as follows: Section 2 introduces our main algorithm for finding the optimal interpolation embedding. Section 3 presents the experiment result of this work. Section 4 presents the theoretical analysis, including the preliminary of our notation, key concepts of our video algorithm, model formulation, and our definition of an optimal interpolation embedding finder. In Section 5, we conclude our paper.

2 OUR METHODS

Section 2.1 introduces the problem formulation. In Section 2.2, we present our algorithm for finding the optimal interpolation embedding.

2.1 PROBLEM FORMULATION

In this section, we introduce the formal definition for finding the optimal interpolation embedding as follows:

Definition 2.1 (Finding Optimal Interpolation Embedding Problem). *Let P_a, P_b, P_c denote three text prompts. Our goal is to generate a video that contains features mentioned in P_a and P_b , and P_c is a text description of the feature combination of P_a and P_b . Let $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}$ denote the text embedding of P_a, P_b, P_c . Let $f_\theta(E_t, z)$ be defined in Definition 4.8. We define the “Finding optimal interpolation embedding” problem as: According to $E_{t_a}, E_{t_b}, E_{t_c}$, find the optimal interpolation embedding E_{opt} that can make the text-to-video generation model $f_\theta(E_{\text{opt}}, z)$ generate video contains features mentioned in P_a and P_b .*

```

108
109 Algorithm 1 Find Optimal Interpolation
110
111 1: datastructure OPTIMALINTERPFINDER
112 2: members
113 3:  $n \in \mathbb{N}$ : the length of input sequence.
114 4:  $n_{\text{ids}} \in \mathbb{N}$ : the ids length of input sequence.
115 5:  $d \in \mathbb{R}$ : the hidden dimension.
116 6:  $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}$ : the text embedding.
117 7:  $\phi_{\text{cos}}(X, Y)$ : the cosine similarity calculator.  $\triangleright$  Definition 4.2
118 8: end members
119 9:
120 10: procedure OPTIMALFINDER( $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}, n_{a_{\text{ids}}}, n_{b_{\text{ids}}}, n_{c_{\text{ids}}} \in \mathbb{N}$ )
121 11: /* Calculate the max ids length. */
122 12:  $n_{\text{ids}} \leftarrow \max\{n_{a_{\text{ids}}}, n_{b_{\text{ids}}}, n_{c_{\text{ids}}}\}$ 
123 13: /* Truncated text embeddings. */
124 14:  $E_{a_{\text{truc}}} \in \mathbb{R}^{n_{\text{ids}} \times d} \leftarrow E_{t_a}[:, n_{\text{ids}}, :]$ 
125 15:  $E_{b_{\text{truc}}} \in \mathbb{R}^{n_{\text{ids}} \times d} \leftarrow E_{t_b}[:, n_{\text{ids}}, :]$ 
126 16:  $E_{c_{\text{truc}}} \in \mathbb{R}^{n_{\text{ids}} \times d} \leftarrow E_{t_c}[:, n_{\text{ids}}, :]$ 
127 17: /* Calculate cosine similarity, Algorithm 2. */
128 18:  $L_{\text{CosTruc}} \leftarrow \text{COSINESIM}(E_{a_{\text{truc}}}, E_{b_{\text{truc}}}, E_{c_{\text{truc}}})$ 
129 19:  $L_{\text{CosFull}} \leftarrow \text{COSINESIM}(E_{t_a}, E_{t_b}, E_{t_c})$ 
130 20: /* Add CosineTruc and CosineFull. */
131 21:  $L_{\text{CosAdd}} \leftarrow []$ 
132 22: for  $i = 1 \rightarrow k$  do
133 23:  $L_{\text{CosAdd}}[i] \leftarrow L_{\text{CosTruc}}[i] + L_{\text{CosFull}}[i]$ 
134 24: end for
135 25: /* Find the optimal interpolation index. */
136 26:  $i_{\text{opt}} \leftarrow \text{maxindex}(L_{\text{CosAdd}})$ 
137 27: /* Calculate optimal interpolation embedding. */
138 28:  $E_{\text{opt}} \leftarrow \frac{i_{\text{opt}}}{k} \cdot E_{t_a} + \frac{k-i_{\text{opt}}}{k} \cdot E_{t_b}$ 
139 29: Return  $E_{\text{opt}}$ 
140 30: end procedure

```

We would like to refer the readers to Figure 2 (a) as an example of Definition 2.1. In Figure 2 (a), we set prompt P_a to “*The tiger, moves gracefully through the forest, its fur flowing in the breeze.*” and prompt P_b to: “*The zebra, moves gracefully through the forest, its fur flowing in the breeze.*”. Our goal is to generate a video that contains both features of “tiger” and “zebra”, where we set prompt P_c to “*The tiger, with black and white stripes like zebra, moves gracefully through the forest, its fur flowing in the breeze.*”, to describe the mixture features of tiger and zebra. However, the text-to-video model fails to generate the expected video. Therefore, it is essential to find the optimal interpolation embedding E_{opt} to make the model generate the expected video. In Figure 2 (a), the E_{opt} is the 14-th interpolation embedding of E_{t_a} and E_{t_b} .

2.2 OPTIMAL INTERPOLATION EMBEDDING FINDER

In this section, we introduce our main algorithm (Algorithm 3 and Algorithm 1), which is also depicted in Fig. 1. The algorithm is designed to identify the optimal interpolation embedding (as defined in Definition 2.1) and generate the corresponding video. The algorithm consists of three key steps:

1. Compute the perpendicular foot embedding (Line 9 in Algorithm 2).
2. Calculate the cosine similarity between the interpolation embeddings and the perpendicular foot embedding (Line 22 in Algorithm 2).
3. Select the optimal interpolation embedding based on the cosine similarity results (Algorithm 1).

We will now provide a detailed explanation of each part of the algorithm and the underlying intuitions.

Algorithm 2 Calculate Cosine Similarity

```

1: datastructure COSINESIMILARITYCALCULATOR
2: members
3:    $n \in \mathbb{N}$ : the length of input sequence.
4:    $d \in \mathbb{N}$ : the hidden dimension.
5:    $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}$ : the text embedding.
6:    $\phi_{\cos}(X, Y)$ : the cosine similarity calculator. ▷ Definition 4.2
7: end members
8:
9: procedure PERPENDICULARFOOT( $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}$ )
10:  /* Find perpendicular foot of  $E_{t_c}$  on  $E_{t_b} - E_{t_b}$ . */
11:   $E_{ac} \leftarrow E_{t_c} - E_{t_a}$ 
12:   $E_{ab} \leftarrow E_{t_b} - E_{t_a}$ 
13:  /* Calculate the projection length. */
14:   $l_{\text{proj}} \leftarrow \langle E_{ab}, E_{ac} \rangle / \langle E_{ab}, E_{ab} \rangle$ 
15:  /* Calculate the projection vector. */
16:   $E_{\text{proj}} \leftarrow l_{\text{proj}} \cdot E_{ab}$ 
17:  /* Calculate the perpendicular foot. */
18:   $E_{\text{foot}} \leftarrow E_{t_a} + E_{\text{proj}}$ 
19:  Return  $E_{\text{foot}}$ 
20: end procedure
21:
22: procedure COSINESIM( $E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}$ )
23:  /* Calculate perpendicular foot. */
24:   $E_{\text{foot}} \leftarrow \text{PENDICULARFOOT}(E_{t_a}, E_{t_b}, E_{t_c})$ 
25:  /* Init cosine similarity list. */
26:   $L_{\text{CosSim}} \leftarrow []$ 
27:  for  $i = 1 \rightarrow k$  do
28:    /* Compute interpolation embedding. */
29:     $E_{\text{interp}} \leftarrow \frac{i}{k} \cdot E_{t_1} + \frac{k-i}{k} \cdot E_{t_2}$ 
30:    /* Calculate and store cosine similarity. */
31:     $L_{\text{CosSim}}[i] \leftarrow \phi_{\cos}(E_{\text{interp}}, E_{\text{foot}})$ 
32:  end for
33:  Return  $L_{\text{CosSim}}$ 
34: end procedure

```

Perpendicular Foot Embedding. As outlined in the problem definition (Definition 2.1), our objective is to identify the optimal interpolation embedding that allows the text-to-video generation model to generate a video containing the features described in P_a and P_b . The combination of these features is represented by P_c , which typically does not lead to the desired video output. Consequently, we seek an interpolation embedding of E_{t_a} and E_{t_b} guided by E_{t_c} . The first step involves finding the perpendicular foot of E_{t_c} onto the vector $E_{t_b} - E_{t_a}$, also known as the projection of E_{t_c} . This perpendicular foot embedding, denoted as E_{foot} , is not the optimal embedding in itself, as the information within E_{t_c} alone does not enable the generation of the expected video. However, E_{foot} serves as a useful anchor, guiding us toward the optimal interpolation embedding. Further details of this approach will be discussed in the subsequent paragraph.

Cosine Similarity and Optimal Interpolation Embedding. To assess the similarity of each interpolation embedding to the anchor perpendicular foot embedding E_{foot} , we employ the straightforward yet effective metric of cosine similarity (Definition 4.2). It is important to note that the input text prompts are padded to a fixed maximum length, $n = 266$, before being encoded by the T5 model. However, in real-world scenarios, the actual length of text prompts is typically much shorter than $n = 266$, which results in a substantial number of padding embeddings being appended to the original text prompt. The inclusion or exclusion of these padding embeddings can lead to significant differences in the perpendicular foot embedding, as their presence introduces a shift in the distribution of the text embeddings. To account for this, we treat text embeddings with and without padding separately. Specifically, we define “full text embeddings” $E_{at}, E_{bt}, E_{ct} \in \mathbb{R}^{n \times d}$ to represent the

embeddings that include padding, and “truncated text embeddings” $E_{a_{\text{true}}}, E_{b_{\text{true}}}, E_{c_{\text{true}}} \in \mathbb{R}^{n_{\text{ids}} \times d}$ to represent the embeddings without padding (Line 13 in Algorithm 1). The full-text embeddings capture global information, whereas the truncated text embeddings focus on local information. We compute the perpendicular foot and cosine similarity separately for both types of text embeddings (Line 17) and then combine the results by summing the cosine similarities from the full and truncated embeddings. Finally, we select the optimal interpolation embedding based on the aggregated cosine similarity scores (Line 25).

3 EXPERIMENTS

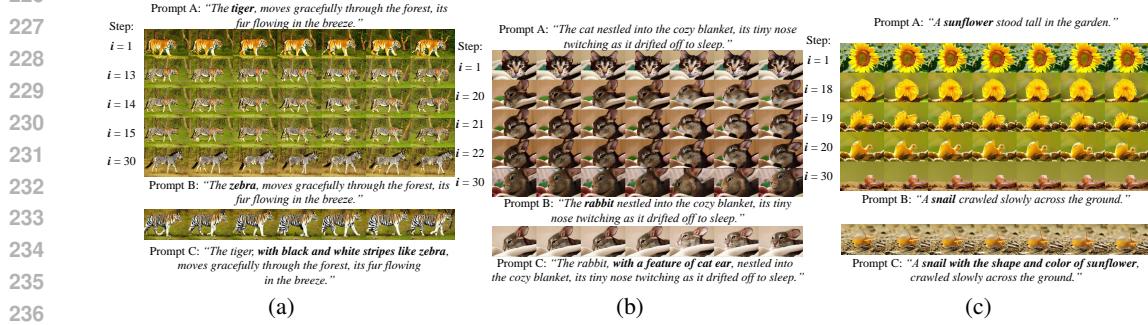


Figure 2: **Qualitative results of mixture of two features.** Figure (a): Mixture of [“Tiger”] and [“Zebra”]; Figure (b): Mixture of [“Cat”] and [“Rabbit”]; Figure (c): Mixture of [“Sunflower”] and [“Snail”]. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

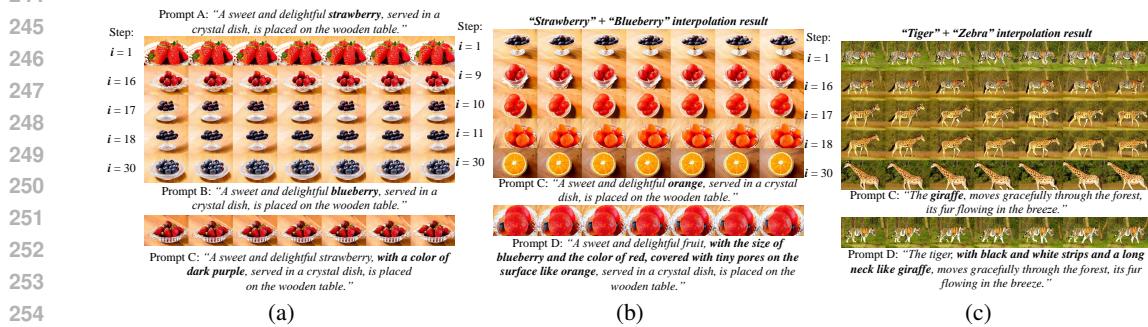


Figure 3: **Extending from two prompts mixture to three prompts mixture.** Figure (a): Mixture of [“Strawberry”] and [“Blueberry”]. Figure (b): Mixture of [“Strawberry” + “Blueberry”] and [“Orange”]. We further apply Algorithm 1 to that optimal embedding and Prompt C embedding, with the guidance of Prompt D. We identify 10-th interpolation embedding as the optimal embedding of [“Strawberry” + “Blueberry”] and [“Orange”] and generate the corresponding video. The video generated directly from Prompt D does not exhibit the desired mixed features. Figure (c): Mixture of [“Tiger” + “Zebra”] and [“Giraffe”]. We present another example of a mixture of three prompts to demonstrate the effectiveness of our algorithm.

In this section, we will first present our qualitative evaluation results of the proposed method in Section 3.1. Then, in Section 3.2, we present our quantitative evaluation.

3.1 QUALITATIVE EVALUATION

Our experiments are conducted on the CogVideoX-2B (Yang et al., 2024). We investigate the performance of our optimal embedding finder algorithm in the following two scenarios:

270 **Mixture of Features from Two Initial Prompts.** As outlined in Definition 2.1, we conduct experiments
 271 where the goal is to generate a mixture of features described in two text prompts, P_a and P_b .
 272 We construct a third prompt, P_c , to specify the desired features. Following Algorithm 1, we identify
 273 the optimal text embedding and use it for the text-to-video generation with our base model. We
 274 conducted experiments using a variety of text prompts. In Figure 2 (a) and Figure 2 (b), we investigate
 275 the mixture of features from different animals, demonstrating that a video containing the mixture of
 276 tiger and zebra features, as well as the mixture of rabbit and cat features can only be generated using
 277 the optimal embedding, not directly from the text prompts. Similarly, in Figure 3 (a), we show that a
 278 video combining features from strawberry and blueberry can only be generated through the optimal
 279 embedding, highlighting a similar phenomenon in the context of fruits. Furthermore, in Figure 2
 280 (c), we observe the same behavior in the domain of plants, specifically with the combination of rose
 281 and cactus features.
 282

283 **Mixture of Features from Three Initial Prompts.** We will investigate further to see if we can
 284 add one additional feature to the video. The high-level approach involves applying our optimal
 285 interpolation embedding algorithm (Algorithm 1) twice. Given three text embeddings, E_{t_a} , E_{t_b} ,
 286 and E_{t_c} , where we aim to blend their features in the generated video, we first apply Algorithm 1
 287 to E_{t_a} and E_{t_b} to obtain the optimal interpolation embedding $E_{\text{opt}_{ab}}$. Next, we apply Algorithm 1
 288 again, this time on $E_{\text{opt}_{ab}}$ and E_{t_c} , resulting in the final optimal interpolation embedding E_{opt} . We
 289 then use this embedding in our base model to generate the desired video. Following the method
 290 described above, we mix the giraffe feature with the tiger and zebra features, as shown in Figure 3
 291 (c). Only by using the optimal embedding identified by our algorithm can we enable the video
 292 generation model to produce the desired video. Directly generating the video from the text prompt
 293 results in the loss of at least one of the intended features. A similar phenomenon is observed in
 294 the case of mixing strawberry, blueberry, and orange features, as shown in Figure 3 (b). The video
 295 generated directly from the text prompt always renders each object separately, failing to combine
 296 the features into a single coherent entity.
 297

3.2 QUANTITATIVE EVALUATION

298 In the previous sections, we presented the qualitative results of our method. In this section, we
 299 provide a quantitative evaluation. Following the settings used by VBench (Huang et al., 2024), we
 300 evaluate the “subject consistency” and “aesthetic quality” of the generated videos. The results for
 301 mixtures of two prompts are presented in Table 1. The average Subject Consistency (SC) of the
 302 videos generated using optimal embeddings is 0.9787, higher than the SC of the videos generated
 303 directly from the prompt description, which is 0.9748. As for Aesthetic Quality (AQ), the videos
 304 generated by optimal embeddings achieve a score of 0.5163, which is lower than the 0.5519 obtained
 305 by the videos generated from prompts.
 306

307 Our method generates videos with higher “subject consistency” than those produced directly from
 308 the prompt description (i.e., Prompt C). This suggests that the optimal embedding enables the video
 309 generation model to better combine the desired features while maintaining coherence in the generated
 310 videos.
 311

312 Another observation is that the “aesthetic quality” of videos generated using the optimal embeddings
 313 is lower than that of videos generated directly from text prompts. This indicates that our method
 314 better blends the desired features. The aesthetic model is trained on real-world videos, which leads
 315 to a bias toward scoring videos that resemble those found in real-world datasets. However, in our
 316 setting, we aim to expand the prompt space of the video generation model, enabling it to generate
 317 videos that are rarely observed in real-world datasets. Therefore, a lower aesthetic score reflects that
 318 our method aligns better with this goal.
 319

4 THEORETICAL ANALYSIS

320 We first introduce some basic notations in Section 4.1. In Section 4.2, we introduce formal definitions
 321 of key concepts. Then, we introduce the formal definition of each module in the CogvideoX
 322 model in Section 4.3. In Section 4.4, we provide our rigorous theoretical analysis showing that word
 323 embedding space is not sufficient to represent all videos.
 324

324 Table 1: **Quantitative Evaluations.** We evaluate the videos generated using our optimal embed-
 325 dings and those generated directly from the text prompt with two metrics: “Subject Consistency”
 326 (SC) and “Aesthetic Quality” (AQ). Let f represent the optimal embedding finding algorithm, and
 327 g denote the video generation model. A higher SC score indicates better coherence in the video,
 328 which corresponds to higher quality. Conversely, a lower AQ score suggests that the video is rarely
 329 observed in the real world, implying that it aligns more closely with the mixture of desired features.
 330 We use A to denote PromptA, B to denote PromptB and C to denote PromptC.

Prompts	SC (\uparrow)	AQ (\downarrow)
$;g(f(\text{Tiger, Zebra}))$	0.9751	0.5472
$g(\text{Tiger, Zebra})$	0.9739	0.5424
$g(f(\text{Cat, Rabbit}))$	0.9688	0.4649
$g(\text{Cat, Rabbit})$	0.9608	0.4821
$g(f(\text{Strawberry, Blueberry}))$	0.9920	0.5957
$g(\text{Strawberry, Blueberry})$	0.9910	0.7256
$g(f(\text{Sunflower, Snail}))$	0.9790	0.4573
$g(\text{Sunflower, Snail})$	0.9734	0.4575
avg. $g(f(\text{A, B}))$	0.9787	0.5163
avg. $g(\text{C})$	0.9748	0.5519

345 4.1 NOTATIONS

346 For any $k \in \mathbb{N}$, let $[k]$ denote the set $\{1, 2, \dots, k\}$. For any $n \in \mathbb{N}$, let n denote the length of the
 347 input sequence of a model. For any $d \in \mathbb{N}$, let d denote the hidden dimension. For any $c \in \mathbb{N}$, let c
 348 denote the channel of a video. For any $n_f \in \mathbb{N}$, we use n_f to denote the video frames. For any $h \in \mathbb{N}$
 349 and $w \in \mathbb{N}$, we use h and w to denote the height and width of a video. For two vectors $x \in \mathbb{R}^n$ and
 350 $y \in \mathbb{R}^n$, we use $\langle x, y \rangle$ to denote the inner product between x, y . Namely, $\langle x, y \rangle = \sum_{i=1}^n x_i y_i$. For
 351 a vector $x \in \mathbb{R}^n$, we use $\|x\|_2$ to denote the ℓ_2 norm of the vector x , i.e., $\|x\|_2 := \sqrt{\sum_{i=1}^n x_i^2}$.
 352

353 Let \mathcal{D} represent a given distribution. The notation $x \sim \mathcal{D}$ indicates that x is a random variable drawn
 354 from the distribution \mathcal{D} .

356 4.2 KEY CONCEPTS

357 We will introduce some essential concepts in this section. We begin with introducing the formal
 358 definition of linear interpolation.

359 **Definition 4.1** (Linear Interpolation). *Let $x, y \in \mathbb{R}^d$ denote two vectors. Let $k \in \mathbb{N}$ denote the
 360 interpolation step. For $i \in [k]$, we define the i -th interpolation result $z_i \in \mathbb{R}$ as follows:*

$$362 \quad z_i := \frac{i}{k} \cdot x + \frac{k-i}{k} \cdot y$$

363 Next, we introduce another key concept used in our paper, the simple yet effective cosine similarity
 364 calculator.

365 **Definition 4.2** (Cosine Similarity Calculator). *Let $X, Y \in \mathbb{R}^{n \times d}$ denote two matrices. Let
 366 $X_i, Y_i \in \mathbb{R}^d$ denote i -th row of X, Y , respectively. Then, we defined the cosine similarity calcu-
 367 lator $\phi_{\text{cos}}(X, Y) : \mathbb{R}^{n \times d} \times \mathbb{R}^{n \times d} \rightarrow \mathbb{R}$ as follows $\phi_{\text{cos}}(X, Y) := \frac{1}{n} \sum_{i=1}^n \frac{\langle X_i, Y_i \rangle}{\|X_i\|_2 \|Y_i\|_2}$.*

368 Then, we introduce one crucial fact that we used later in this paper.

369 **Fact 4.3** (Volume of a Ball in d -dimension Space). *The volume of a ℓ_2 -ball with radius R in dimen-
 370 sion \mathbb{R}^d space is $\frac{\pi^{d/2}}{(d/2)!} R^d$.*

375 4.3 MODEL FORMULATION

376 In this section, we will introduce the formal definition for the text-to-video generation video we use.
 377 We begin with introducing the formal definition of the attention layer as follows:

378 **Algorithm 3** Video Interpolation

379 1: **datastructure** INTERPOLATION

380 2: **members**

381 3: $n \in \mathbb{N}$: the length of input sequence

382 4: $n_f \in \mathbb{N}$: the number of frames

383 5: $h \in \mathbb{N}$: the height of video

384 6: $w \in \mathbb{N}$: the width of video

385 7: $d \in \mathbb{N}$: the hidden dimension

386 8: $c \in \mathbb{N}$: the channel of video

387 9: $k \in \mathbb{N}$: the interpolation steps

388 10: $T \in \mathbb{N}$: the number of inference step

389 11: $E_{\text{opt}} \in \mathbb{R}^{n \times d}$: the optimal interpolation embedding

390 12: $E_t \in \mathbb{R}^{n \times d}$: the text embedding

391 13: $f_{\theta}(z, E_t, t) : \mathbb{R}^{n_f \times h \times w \times c} \times \mathbb{R}^{n \times d} \times \mathbb{N} \rightarrow \mathbb{R}^{n_f \times h \times w \times c}$: the text-to-video generation model

392 14: **end members**

393 15:

394 16: **procedure** INTERPOLATION($E_{t_a}, E_{t_b}, E_{t_c} \in \mathbb{R}^{n \times d}, k \in \mathbb{N}, T \in \mathbb{N}$)

395 17: /* Find optimal interpolation embedding, Algorithm 1. */

396 18: $E_{\text{opt}} \leftarrow \text{OPTIMALFINDER}(E_{t_a}, E_{t_b}, E_{t_c})$

397 19: /* Prepare initial latents. */

398 20: $z \sim \mathcal{N}(0, I) \in \mathbb{R}^{n_f \times h \times w \times c}$

399 21: **for** $t = T \rightarrow 0$ **do**

400 22: /* One denoise step. */

401 23: $z \leftarrow f_{\theta}(z, E_{\text{opt}}, t)$

402 24: **end for**

403 25: Return z

404 26: **end procedure**

405 **Definition 4.4** (Attention Layer). Let $X \in \mathbb{R}^{n \times d}$ denote the input matrix. Let $W_K, W_Q, W_V \in$
 406 $\mathbb{R}^{d \times d}$ denote the weighted matrices. Let $Q = XW_Q \in \mathbb{R}^{n \times d}$ and $K = XW_K \in \mathbb{R}^{n \times d}$. Let
 407 attention matrix $A = QK^{\top}$. Let $D := \text{diag}(A\mathbf{1}_n) \in \mathbb{R}^{n \times n}$. We define attention layer Attn as
 408 follows: $\text{Attn}(X) := D^{-1}AXW_V$.

409 Then, we define the convolution layer as follows:

410 **Definition 4.5** (Convolution Layer). Let $h \in \mathbb{N}$ denote the height of the input and output feature
 411 map. Let $w \in \mathbb{N}$ denote the width of the input and output feature map. Let $c_{\text{in}} \in \mathbb{N}$ denote the
 412 number of channels of the input feature map. Let $c_{\text{out}} \in \mathbb{N}$ denote the number of channels of the
 413 output feature map. Let $X \in \mathbb{R}^{h \times w \times c_{\text{in}}}$ represent the input feature map. For $l \in [c_{\text{out}}]$, we use $K^l \in$
 414 $\mathbb{R}^{3 \times 3 \times c_{\text{in}}}$ to denote the l -th convolution kernel. Let p denote the padding of the convolution layer.
 415 Let s denote the stride of the convolution kernel. Let $Y \in \mathbb{R}^{h \times w \times c_{\text{out}}}$ represent the output feature
 416 map. We define the convolution layer as follows: We use $\phi_{\text{conv}}(X, c_{\text{in}}, c_{\text{out}}, p, s) : \mathbb{R}^{h \times w \times c_{\text{in}}} \rightarrow$
 417 $\mathbb{R}^{h \times w \times c_{\text{out}}}$ to represent the convolution operation. Let $Y = \phi_{\text{conv}}(X, c_{\text{in}}, c_{\text{out}}, p, s)$. Then, for
 418 $i \in [h], j \in [w], l \in [c_{\text{out}}]$, we have $Y_{i,j,l} := \sum_{m=1}^3 \sum_{n=1}^3 \sum_{c=1}^{c_{\text{in}}} X_{i+m-1, j+n-1, c} \cdot K_{m,n,c}^l$

419 We introduce the formal definition of linear projection layer as follows:

420 **Definition 4.6** (Linear Projection). Let $X \in \mathbb{R}^{n \times d_1}$ denote the input data matrix. Let $W \in \mathbb{R}^{d_1 \times d_2}$
 421 denote the weight matrix. We define the linear projection $\phi_{\text{linear}} : \mathbb{R}^{n \times d_1} \rightarrow \mathbb{R}^{n \times d_2}$ as follows:

$$\phi_{\text{linear}}(X) := XW$$

422 And we define the 3D full attention layer as follows:

423 **Definition 4.7** (3D Attention). Let $\text{Attn}(X)$ be defined as in Definition 4.4. Let $\phi_{\text{conv}}(X, c_{\text{in}, \text{out}, p, s})$
 424 be defined in Definition 4.5. Let $\phi_{\text{linear}}(X)$ be defined as in Definition 4.6. We define the 3D attention
 425 $\phi_{\text{3DAttn}}(E_t, E_v)$ containing three components: $\phi_{\text{linear}}(X)$, $\text{Attn}(X)$, $\phi_{\text{conv}}(X, c_{\text{in}}, c_{\text{out}}, p, s)$. Its
 426 details are provided in Algorithm 4.

427 Finally, we provide the definition of the text-to-video generation model, which consists of a stack of
 428 multiple 3D attention layers, as introduced earlier.

432
 433 **Definition 4.8** (Text-to-Video Generation Model). Let $\phi_{3D\text{Attn}}$ be defined as Definition 4.7. Let
 434 $k_{3D} \in \mathbb{N}$ denote the number of 3D attention layers in the text-to-video generation model. Let
 435 θ denote the parameter in the text-to-video generation model. Let $E_t \in \mathbb{R}^{n \times d}$ denote the text
 436 embedding. Let $z \sim \mathcal{N}(0, I) \in \mathbb{R}^{n_f \times h \times w \times c}$ denote the initial random Gaussian noise. Then we
 437 defined the text-to-video generation model $f_\theta(E_t, z)$ as follows:

438
$$f_\theta(E_t, z) := \underbrace{\phi_{3D\text{Attn}} \circ \cdots \circ \phi_{3D\text{Attn}}}_{k_{3D} \text{ layers}}(E_t, z).$$

439

440

441 **4.4 WORD EMBEDDING SPACE BEING INSUFFICIENT TO REPRESENT FOR ALL VIDEOS**

442

443 Since the text-to-video generation model only has a finite vocabulary size, it only has finite wording embedding
 444 space. However, the space for all videos is infinite. Thus, word embedding space is insufficient to represent
 445 all videos in video space. We formalize this phenomenon to a rigorous math problem and provide our findings in
 446 the following theorem.

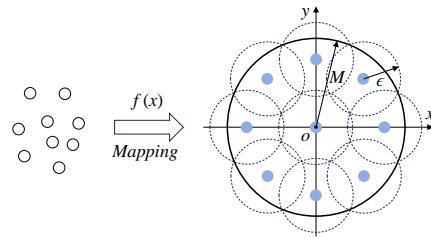
447 **Theorem 4.9** (Word Embeddings being Insufficient to
 448 Represent for All Videos, formal version of Theorem 1.1). Let n, d denote two integers, where n denotes the maximum length of the sentence, and all videos are in \mathbb{R}^d space. Let $V \in \mathbb{N}$ denote the vocabulary size. Let $\mathcal{U} = \{u_1, u_2, \dots, u_V\}$ denote the word embedding space, where for $i \in [V]$, the word embedding $u_i \in \mathbb{R}^k$. Let $\delta_{\min} = \min_{i, j \in [V], i \neq j} \|u_i - u_j\|_2$ denote the minimum ℓ_2 distance of two word embedding. Let $f : \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^d$ denote the text-to-video generation model, which is also a mapping from sentence space (discrete space $\{u_1, \dots, u_V\}^n$) to video space \mathbb{R}^d . Let $M := \max_x \|f(x)\|_2$, $m := \min_x \|f(x)\|_2$. Let $\epsilon = ((M^d - m^d)/V^n)^{1/d}$. Then, we can show that there exists a video $y \in \mathbb{R}^d$, satisfying $m \leq \|y\|_2 \leq M$, such that for any sentence $x \in \{u_1, u_2, \dots, u_V\}^n$, we have $\|f(x) - y\|_2 \geq \epsilon$.

468 Theorem 4.9 indicates that there always exists a video y , where its ℓ_2 distance to all videos can be
 469 represented by the prompt embeddings is larger than ϵ (Fig. 4). This means that there always exists
 470 a video that cannot be accurately generated by using only the prompt embeddings from the word
 471 embedding space. We defer the proof to Theorem C.6 which is the restatement of Theorem 4.9 in
 472 the Appendix.

473 **5 CONCLUSION**

474

475 In this work, we propose a novel algorithm to identify the optimal text embedding, enabling a
 476 video generation model to produce videos that accurately reflect the features specified in the initial
 477 prompts. Our findings reveal that the main bottleneck in text-to-video generation is the text
 478 encoder’s inability to generate precise text embeddings. By carefully selecting and interpolating text
 479 embeddings, we improve the model’s ability to generate more accurate and diverse videos. From
 480 the theoretical side, we show that text embeddings generated by the text encoder are insufficient
 481 to represent all possible video features, which explains why the text encoder becomes a bottleneck
 482 in generating videos with mixed desired features. Our proposed algorithm, based on perpendicular
 483 foot embeddings and cosine similarity, provides an effective solution to these challenges. These
 484 results highlight the importance of refining text embeddings to improve model performance and lay
 485 the foundation for future advancements in text-to-video generation by emphasizing the critical role
 486 of embedding optimization in bridging the gap between textual descriptions and video synthesis.



443 **Figure 4: Mapping from Prompt Space to Video Space.** This figure illustrates the mapping from a prompt space (with discrete prompts) to a video space (with continuous video embeddings) by a video generation model $f(x)$. Regardless of the specific form of the video generation model $f(x)$, there always exists a point in the video embedding space whose distance to all $f(x)$ is at least ϵ .

486
487
ETHIC STATEMENT488
489
490
491
This paper does not involve human subjects, personally identifiable data, or sensitive applications.
We do not foresee direct ethical risks. We follow the ICLR Code of Ethics and affirm that all aspects
of this research comply with the principles of fairness, transparency, and integrity.492
493
REPRODUCIBILITY STATEMENT494
495
496
497
We ensure reproducibility on both theoretical and empirical fronts. For theory, we include all formal
assumptions, definitions, and complete proofs in the appendix. For experiments, we describe models
and algorithms in the main text and appendix.498
499
REFERENCES500
501
502
Jie An, Songyang Zhang, Harry Yang, Sonal Gupta, Jia-Bin Huang, Jiebo Luo, and Xi Yin. Latent-
shift: Latent diffusion with temporal shift for efficient text-to-video generation. *arXiv preprint*
arXiv:2304.08477, 2023.503
504
505
506
Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten
Kreis, Miika Aittala, Timo Aila, Samuli Laine, et al. ediff-i: Text-to-image diffusion models with
an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.507
508
509
510
Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler,
and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion mod-
els. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*,
pp. 22563–22575, 2023.511
512
513
514
Rumeysa Bodur, Erhan Gundogdu, Binod Bhattarai, Tae-Kyun Kim, Michael Donoser, and Loris
Bazzani. redit: Localised text-guided image editing with weak supervision. In *Proceedings of the*
IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7426–7435, 2024.515
516
517
518
Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam
Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for
high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*,
2024.519
520
521
522
Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs,
Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for
video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer*
Vision, pp. 22930–22941, 2023.523
524
525
526
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,
Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information*
processing systems, 27, 2014.527
Google. Veo 2, 2024. URL <https://deepmind.google/technologies/veo/veo-2/>.528
529
530
531
Jiaxi Gu, Shicong Wang, Haoyu Zhao, Tianyi Lu, Xing Zhang, Zuxuan Wu, Songcen Xu, Wei
Zhang, Yu-Gang Jiang, and Hang Xu. Reuse and diffuse: Iterative denoising for text-to-video
generation. *arXiv preprint arXiv:2309.03549*, 2023.532
533
534
Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh
Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffu-
sion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023.535
536
537
Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in*
neural information processing systems, 33:6840–6851, 2020.538
539
Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P
Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition
video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

540 Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pre-
 541 training for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.
 542

543 Jerry Yao-Chieh Hu, Weimin Wu, Yi-Chen Lee, Yu-Chao Huang, Minshuo Chen, and Han Liu. On
 544 statistical rates of conditional diffusion transformers: Approximation, estimation and minimax
 545 optimality. *arXiv preprint arXiv:2411.17522*, 2024a.
 546

547 Jerry Yao-Chieh Hu, Weimin Wu, Zhuoru Li, Sophia Pi, , Zhao Song, and Han Liu. On statistical
 548 rates and provably efficient criteria of latent diffusion transformers (dits). *Advances in Neural
 549 Information Processing Systems*, 38, 2024b.
 550

551 Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianx-
 552 ing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for
 553 video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and
 Pattern Recognition*, pp. 21807–21818, 2024.
 554

555 Yupei Lin, Sen Zhang, Xiaojun Yang, Xiao Wang, and Yukai Shi. Regeneration learning of diffusion
 556 models with rich prompts for zero-shot image translation. *arXiv preprint arXiv:2305.04651*, 2023.
 557

558 Chengyi Liu, Jiahao Zhang, Shijie Wang, Wenqi Fan, and Qing Li. Score-based generative diffusion
 559 models for social recommendations. *arXiv preprint arXiv:2412.15579*, 2024.
 560

561 Xingchao Liu, Chengyue Gong, Lemeng Wu, Shujian Zhang, Hao Su, and Qiang Liu. Fused-
 562 dream: Training-free text-to-image generation with improved clip+ gan space optimization. *arXiv
 563 preprint arXiv:2112.01573*, 2021.
 564

565 Haoming Lu, Hazarapet Tunanyan, Kai Wang, Shant Navasardyan, Zhangyang Wang, and
 566 Humphrey Shi. Specialist diffusion: Plug-and-play sample-efficient fine-tuning of text-to-image
 567 diffusion models to learn any unseen style. In *Proceedings of the IEEE/CVF Conference on
 Computer Vision and Pattern Recognition*, pp. 14267–14276, 2023.
 568

569 Meta. Moviegen: Acastofmediafoundationmodels, 2024. URL [https://ai.meta.com/
 570 static-resource/movie-gen-research-paper](https://ai.meta.com/static-resource/movie-gen-research-paper).
 571

572 OpenAI. Sora, 2024. URL <https://openai.com/index/sora/>.
 573

574 William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of
 575 the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.
 576

577 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal,
 578 Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual
 579 models from natural language supervision. In *International conference on machine learning*, pp.
 580 8748–8763. PMLR, 2021.
 581

582 Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-
 583 resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF confer-
 584 ence on computer vision and pattern recognition*, pp. 10684–10695, 2022.
 585

586 Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Yanyu Li, Yifan Gong, Kai Zhang, Hao Tan, Jason
 587 Kuen, Henghui Ding, et al. Lazydit: Lazy learning for the acceleration of diffusion transformers.
 588 *arXiv preprint arXiv:2412.12444*, 2024.
 589

590 Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt:
 591 Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint
 592 arXiv:2010.15980*, 2020.
 593

594 Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry
 595 Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video
 596 data. *arXiv preprint arXiv:2209.14792*, 2022.
 597

598 Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben
 599 Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint
 600 arXiv:2011.13456*, 2020.
 601

594 A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.
595

596 Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang,
597 Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable
598 length video generation from open domain textual descriptions. In *International Conference on*
599 *Learning Representations*, 2022.

600 Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion
601 for prediction, generation, and interpolation. *Advances in neural information processing systems*,
602 35:23371–23385, 2022.

603 Yilin Wang, Haiyang Xu, Xiang Zhang, Zeyuan Chen, Zhizhou Sha, Zirui Wang, and Zhuowen Tu.
604 Omnicontrolnet: Dual-stage integration for conditional image generation. In *Proceedings of the*
605 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7436–7448, 2024a.

606

607 Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, and Zhuowen Tu. Tokencompose: Text-to-
608 image diffusion with token-level supervision. In *Proceedings of the IEEE/CVF Conference on*
609 *Computer Vision and Pattern Recognition*, pp. 8553–8564, 2024b.

610 Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang,
611 Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models
612 with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

613

614 Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo:
615 Efficient video generation with latent diffusion models. *arXiv preprint arXiv:2211.11018*, 2022a.

616

617 Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-
618 language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022b.

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648 649 650 651 **Appendix** 652 653 654

651 **Roadmap.** In Section A, we provide a detailed discussion of our work. In Section B, we review
652 related literature. In Section C, we provide detailed proofs for the theorem showing that word
653 embeddings are insufficient to represent all videos. In Section D, we provide more results of our
654 experiments. In Section E, we provide the algorithm for 3D attention.

655 A DISCUSSION 656

658 **Identifying the Actual Bottleneck of Generative Models.** Our work identifies that the primary
659 bottleneck hindering text-to-video generation models from producing the desired videos is the text
660 encoder’s inability to generate accurate text embeddings. Through our proposed algorithm, we can
661 guide the video generation model to produce the desired output. This insight helps the community
662 identify the true bottleneck within cutting-edge generative models, allowing for improvements in
663 model performance and capabilities.

664 B RELATED WORK 665

667 **Text-to-Video Generation.** Text-to-video generation (Singer et al., 2022; Voleti et al., 2022;
668 Blattmann et al., 2023), as a form of conditional video generation, focuses on the synthesis of high-
669 quality videos using text descriptions as conditioning inputs. Most recent works on video generation
670 jointly synthesize multiple frames based on diffusion models (Song et al., 2020; Ho et al., 2020; Liu
671 et al., 2024; Shen et al., 2024; Hu et al., 2024b;a). Diffusion models implement an iterative refine-
672 ment process by learning to gradually denoise a sample from a normal distribution, which has been
673 successfully applied to high-quality text-to-video generation. In terms of training strategies, one
674 of the existing approaches uses pre-trained text-to-image models and inserts temporal modules (Ge
675 et al., 2023; An et al., 2023), such as temporal convolutions and temporal attention mechanisms into
676 the pre-trained models to build up correlations between frames in the video (Singer et al., 2022; Gu
677 et al., 2023; Guo et al., 2023). PYoCo (Ge et al., 2023) proposed a noise prior approach and leveraged
678 a pre-trained eDiff-I (Balaji et al., 2022) as initialization. Conversely, other works (Blattmann et al.,
679 2023; Zhou et al., 2022a) build upon Stable Diffusion (Rombach et al., 2022) owing to the accessi-
680 bility of pre-trained models. This approach aims to leverage the benefits of large-scale pre-trained
681 text-to-image models to accelerate convergence. However, it may lead to unsatisfactory results due
682 to the potential distribution gap between images and videos. Other approaches are training the entire
683 model from scratch on both image and video datasets (Ho et al., 2022). Although this method can
684 yield high-quality results, it demands tremendous computational resources.

685 **Enrich Prompt Space.** In the context of conditional tasks, such as text-to-image and text-to-video
686 models, prompts worked as conditions can have a significant influence on the performance of the
687 models. For text-conditioned tasks, refining the user-provided natural provided natural language
688 prompts into keyword-enriched prompts has gained increasing attention. Several recent works have
689 explored the prompt space by the use of prompt learning, such as CoCoOp (Zhou et al., 2022b),
690 which uses conditional prompts to improve the model’s generalization capabilities. AutoPrompt
691 (Shin et al., 2020) explores tokens with the most significant gradient changes in the label likelihood
692 to automate the prompt generation process. Fusedream (Liu et al., 2021) manipulates the CLIP (Rad-
693 fford et al., 2021) latent space by using GAN (Goodfellow et al., 2014) optimization to enrich the
694 prompt space. Specialist Diffusion (Lu et al., 2023) augments the prompts to define the same image
695 with multiple captions that convey the same meaning to improve the generalization of the image
696 generation network. Another work (Lin et al., 2023) proposes to generate random sentences, includ-
697 ing source and target domain, in order to calculate a mean difference that will serve as a direction
698 while editing. The iEdit (Bodur et al., 2024) generates target prompts by changing words in the input
699 caption in order to retrieve pseudo-target images and guide the model. The TokenCompose (Wang
700 et al., 2024b) and OmniControlNet (Wang et al., 2024a) control the image generation in the token-
701 level space. Compared to the prior works, our work takes a different approach by exploring whether
we can obtain a powerful text embedding capable of guiding the video generation model through
interpolation within the text embedding space.

702 **C WORD EMBEDDING SPACE BEING INSUFFICIENT TO REPRESENT FOR ALL**
 703 **VIDEOS**
 704

705 In this section, we provide detailed proofs for Theorem C.8, showing that word embeddings are
 706 insufficient for representing all videos. We begin with a 1 dimensional case, where we assume all
 707 weights in function $f(x)$ are integers.
 708

709 **Lemma C.1** (Integer function bound in 1 dimension). *If the following conditions hold:*

- 710 • Let $V \in \mathbb{N}$ denote a positive integer.
- 711 • Let $f : [V]^n \rightarrow \mathbb{R}$ denote a linear function where weights are all integers.
- 712 • Let $x \in [V]^n$ denote the input of function f .
- 713 • Let $M := \max_x f(x)$, $m := \min_x f(x)$.
- 714 • Let $\epsilon = 0.5$.

715 Then we can show there exists a scalar $y \in [m, M]$ such that for any $x \in [V]^n$, $|f(x) - y| \geq \epsilon$.

720 *Proof.* Since $x \in [V]^n$, all entries of x are integers. Since function f is a linear function where all
 721 weights are integers, the output $f(x) \in \mathcal{Z}$ can only be integer.
 722

723 Therefore, $m, M \in \mathcal{Z}$. We choose $y = m + 0.5$. Since for all $f(x)$ are integers, then we have
 724 $|f(x) - y| \geq 0.5$. □
 725

726 Then, we extend the above Lemma to d dimensional case.

727 **Lemma C.2** (Integer function bound in d dimension). *If the following conditions hold:*

- 729 • Let $V \in \mathbb{N}$ denote a positive integer.
- 730 • Let $f : [V]^n \rightarrow \mathbb{R}^d$ denote a linear function where weights are all integers.
- 731 • Let $x \in [V]^n$ denote the input of function f .
- 732 • Let $M := \max_x \|f(x)\|_2$, $m := \min_x \|f(x)\|_2$.
- 733 • Let $\epsilon = 0.5\sqrt{d}$.

737 Then we can show there exists a vector $y \in \mathbb{R}^d$, satisfying $m \leq \|y\|_2 \leq M$, such that for any
 738 $x \in [V]^n$, $\|f(x) - y\|_2 \geq \epsilon$.
 739

740 *Proof.* Let $x_{\min} \in [V]^n$ denote the vector which satisfies $f(x_{\min}) = m$. Since all entries in x and
 741 f are integers, all entries in $f(x_{\min})$ are all integers.
 742

743 For $i \in [d]$, let $z_i \in \mathcal{Z}$ denote the i -th entry of $f(x_{\min})$.

744 Then, we choose the vector $y \in \mathbb{R}^d$ as

$$746 \quad y = \begin{bmatrix} z_1 + 0.5 \\ z_2 + 0.5 \\ \vdots \\ z_d + 0.5 \end{bmatrix}$$

751 Then, since all entries of $f(x)$ are integers, we have $\|f(x) - y\|_2 \geq 0.5\sqrt{d}$. □
 752

753 Then, we move on to a more complicated case, in which we do not make any assumptions about the
 754 function $f(x)$. We still begin by considering the 1 dimensional case.
 755

756 **Definition C.3** (Set Complement). *If the following conditions hold:*

756 • Let A, U denote two sets.
 757

758 Then, we use $U \setminus A$ to denote the complement of A in U :

759
$$U \setminus A := \{x \in U : x \notin A\}$$

 760

761 **Definition C.4** (Cover). If the following conditions hold:

762 • Let X denote a set.
 763
 764 • Let A denote an index set.
 765
 766 • For $\alpha \in A$, let $U_\alpha \subset X$ denote the subset of X , indexed by A .
 767
 768 • Let $C = \{U_\alpha : \alpha \in A\}$.

769 Then we call C is a cover of X if the following holds:

770
$$X \subseteq \bigcup_{\alpha \in A} U_\alpha$$

 771

772 **Lemma C.5** (Any function bound in 1 dimension). If the following conditions hold:

773 • Let $V \in \mathbb{N}$ denote a positive integer.
 774
 775 • Let $f : [V]^n \rightarrow \mathbb{R}$ denote a function.
 776
 777 • Let $x \in [V]^n$ denote the input of function f .
 778
 779 • Let $M := \max_x f(x)$, $m := \min_x f(x)$.
 780
 781 • Let $\epsilon = (M - m)/(2V^n)$.

782 Then we can show there exists a scalar $y \in [m, M]$ such that for any $x \in [V]^n$, $|f(x) - y| \geq \epsilon$.
 783

784 *Proof.* Assuming for all $y \in [m, M]$, there exists one $f(x)$, such that $|f(x) - y| < (M - m)/(2V^n)$.
 785

786 The overall maximum cover of all V^n points should satisfy

787
$$2 \cdot V^n \cdot |f(x) - y| < (M - m) \tag{1}$$

 788

789 where the first step follows from there are total V^n possible choices for $f(x)$, and each choice has a
 790 region with length less than $2|f(x) - y|$. This is because the y can be either left side of $f(x)$, or can
 791 be on the right side of $f(x)$, for both case, we need to have $|f(x) - y| < (M - m)/(2V^n)$. So the
 792 length for each region of $f(x)$ should at least be $2|f(x) - y|$.

793 Eq (1) indicates the overall regions of V^n points can not cover all $[m, M]$ range, i.e. cannot become
 794 a cover (Definition C.4) of $[m, M]$. This is because each points can cover at most $2|f(x) - y| <$
 795 $(M - m)/V^n$ length, and there are total V^n points. So the maximum region length is less than
 796 $V^n \cdot (M - m)/V^n = (M - m)$. Note that the length of the range $[m, M]$ is $(M - m)$. Therefore,
 797 V^n points cannot cover all $[m, M]$ range.

798 We use \mathcal{S} to denote the union of covers of all possible $f(x)$. Since the length of \mathcal{S} is less than
 799 $M - m$, there exists at least one y lies in $[m, M] \setminus \mathcal{S}$ such that $|f(x) - y| \geq (M - m)/(2V^n)$. Here
 800 \setminus denotes the set complement operation as defined in Definition C.3.

801 Then, we complete our proof. □
 802

803 Here, we introduce an essential fact that states the volume of a ℓ_2 -ball in d dimensional space.
 804

805 Then, we extend our 1 dimensional result on any function $f(x)$ to d dimensional cases.

806 **Theorem C.6** (Word embeddings are insufficient to represent for all videos, restatement of Theo-
 807 rem 4.9). If the following conditions hold:

808 • Let n, d denote two integers, where n denotes the maximum length of the sentence, and all
 809 videos are in \mathbb{R}^d space.

- Let $V \in \mathbb{N}$ denote the vocabulary size.
- Let $\mathcal{U} = \{u_1, u_2, \dots, u_V\}$ denote the word embedding space, where for $i \in [V]$, the word embedding $u_i \in \mathbb{R}^k$.
- Let $\delta_{\min} = \min_{i,j \in [V], i \neq j} \|u_i - u_j\|_2$ denote the minimum ℓ_2 distance of two word embedding.
- Let $f : \mathbb{R}^{nk} \rightarrow \mathbb{R}^d$ denote the mapping from sentence space (discrete space $\{u_1, \dots, u_V\}^n$) to video space \mathbb{R}^d .
- Let $M := \max_x \|f(x)\|_2$, $m := \min_x \|f(x)\|_2$.
- Let $\epsilon = ((M^d - m^d)/V^n)^{1/d}$.

Then, we can show that there exists a video $y \in \mathbb{R}^d$, satisfying $m \leq \|y\|_2 \leq M$, such that for any sentence $x \in \{u_1, u_2, \dots, u_V\}^n$, $\|f(x) - y\|_2 \geq \epsilon$.

Proof. Assuming for all y satisfying $m \leq \|y\|_2 \leq M$, there exists one $f(x)$, such that $|f(x) - y| < ((M^d - m^d)/V^n)^{1/d}$.

Then, according to Fact 4.3, for each $f(x)$, the volume of its cover is $\frac{\pi^{d/2}}{(d/2)!}((M^d - m^d)/V^n)$.

There are maximum total V^n $f(x)$, so the maximum volume of all covers is

$$V^n \cdot \frac{\pi^{d/2}}{(d/2)!}((M^d - m^d)/V^n) < \frac{\pi^{d/2}}{(d/2)!}(M^d - m^d) \quad (2)$$

The entire space of a d -dimensional ℓ_2 ball is $\frac{\pi^{d/2}}{(d/2)!}(M^d - m^d)$. However, according to Eq. (2) the maximum volume of the regions generated by all $f(x)$ is less than $\frac{\pi^{d/2}}{(d/2)!}(M^d - m^d)$. Therefore Eq. (2) indicates the cover of all V^n possible points does not cover the entire space for y .

Therefore, there exists a y satisfying $m \leq \|y\|_2 \leq M$, such that $\|f(x) - y\|_2 \geq ((M^d - m^d)/V^n)^{1/d}$.

Then, we complete our proof. □

Definition C.7 (Bi-Lipschitzness). We say a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is L -bi-Lipschitz if for all $x, y \in \mathbb{R}^n$, we have

$$L^{-1}\|x - y\|_2 \leq \|f(x) - f(y)\|_2 \leq L\|x - y\|_2.$$

Then, we state our main result as follows

Theorem C.8 (Word embeddings are insufficient to represent for all videos, with Bi-Lipschitz condition). If the following conditions hold:

- Let n, d denote two integers, where n denotes the maximum length of the sentence, and all videos are in \mathbb{R}^d space.
- Let $V \in \mathbb{N}$ denote the vocabulary size.
- Let $\mathcal{U} = \{u_1, u_2, \dots, u_V\}$ denote the word embedding space, where for $i \in [V]$, the word embedding $u_i \in \mathbb{R}^k$.
- Let $\delta_{\min} = \min_{i,j \in [V], i \neq j} \|u_i - u_j\|_2$ denote the minimum ℓ_2 distance of two word embedding.
- Let $f : \mathbb{R}^{nk} \rightarrow \mathbb{R}^d$ denote the text-to-video generation model, which is also a mapping from sentence space (discrete space $\{u_1, \dots, u_V\}^n$) to video space \mathbb{R}^d .
- Assuming $f : \mathbb{R}^{nk} \rightarrow \mathbb{R}^d$ satisfies the L -bi-Lipschitz condition (Definition C.7).

864 • Let $M := \max_x \|f(x)\|_2$, $m := \min_x \|f(x)\|_2$.
 865 • Let $\epsilon = \max\{0.5 \cdot \delta_{\min}/L, ((M^d - m^d)/V^n)^{1/d}\}$.

866 Then, we can show that there exists a video $y \in \mathbb{R}^d$, satisfying $m \leq \|y\|_2 \leq M$, such that for any
 869 sentence $x \in \{u_1, u_2, \dots, u_V\}^n$, $\|f(x) - y\|_2 \geq \epsilon$.

870
 871 *Proof.* Our goal is to prove that when the bi-Lipschitz condition (Definition C.7) holds for $f(x)$, the
 872 statement can be held with $\epsilon = \max\{0.5 \cdot \delta_{\min}/L, ((M^d - m^d)/V^n)^{1/d}\}$.

873 According to Lemma 4.9, we have that $\epsilon \geq ((M^d - m^d)/V^n)^{1/d}$. Then, we only need to prove that
 874 when $0.5 \cdot \delta_{\min}/L > ((M^d - m^d)/V^n)^{1/d}$, holds, $\epsilon = \max\{0.5 \cdot \delta_{\min}/L, ((M^d - m^d)/V^n)^{1/d}\} =$
 875 $0.5 \cdot \delta_{\min}$, our statement still holds.

876 Since we have assumed that the function $f(x)$ satisfies that for all $x, y \in \mathbb{R}^{nk}$, such that

$$877 \quad \|f(x) - f(y)\|_2 \geq \|x - y\|_2/L. \quad (3)$$

880 According to the definition of δ_{\min} , we have for all $i, j \in [V], i \neq j$, such that

$$882 \quad \|u_i - u_j\|_2 \geq \delta_{\min} \quad (4)$$

884 Combining Eq. (3) and (4), we have for all $i, j \in [V], i \neq j$

$$886 \quad \|f(u_i) - f(u_j)\|_2 \geq \delta_{\min}/L \quad (5)$$

888 We choose $y = f(\frac{1}{2}(u_i + u_j))$ for any $i, j \in [V], i \neq j$

889 Then, for all $k \in [V]$, we have

$$891 \quad \|y - f(u_k)\|_2 \geq \left\| \frac{1}{2}(u_i + u_j) - u_k \right\|_2/L \\ 892 \quad \geq 0.5 \cdot \delta_{\min}/L$$

894 where the first step follows from $f(x)$ satisfies the bi-Lipschitz condition, the second step follows
 895 from Eq. (5).

897 Therefore, when we have $0.5 \cdot \delta_{\min}/L > ((M^d - m^d)/V^n)^{1/d}$ holds, then we must have $\epsilon =$
 898 $0.5 \cdot \delta_{\min}/L$.

899 Considering all conditions we discussed above, we are safe to conclude that $\epsilon = \max\{0.5 \cdot$
 900 $\delta_{\min}/L, ((M^d - m^d)/V^n)^{1/d}\}$ \square

902
 903 **Table 2: Statement Reference Table.** This table shows the relationship between definitions and
 904 algorithms used in the paper, helping readers easily track where each term is defined and referenced.

906 Statements	907 Comment	908 Call	909 Called by
Def. 4.1	Define linear interpolation	None	Alg. 2, Alg. 1
Def. 4.2	Define cosine similarity calculator	None	Alg. 2, Alg. 1
Def. 4.4	Define attention layer	None	Alg. 4, Def. 4.7
Def. 4.5	Define convolution layer	None	Alg. 4, Def. 4.7
Def. 4.6	Define linear projection	None	Alg. 4, Def. 4.7
Def. 4.7	Define 3D attention	Def. 4.4, Def. 4.5, Def. 4.6	Alg. 4, Def. 4.8
Def. 4.8	Define text to video generation model	Def. 4.7	Def. 2.1
Def. 2.1	Define optimal interpolation embedding	Def. 4.8	Alg. 3
Alg. 4	3D Attention algorithm	Def. 4.4, Def. 4.5, Def. 4.6, Def. 4.7	None
Alg. 2	Cosine similarity calculator algorithm	Def. 4.1, Def. 4.2	Alg. 1
Alg. 1	Find optimal interpolation algorithm	Def. 4.1, Def. 4.2, Alg. 2	Alg. 3
Alg. 3	Video interpolation algorithm	Alg. 1	None

Step:

i = 1



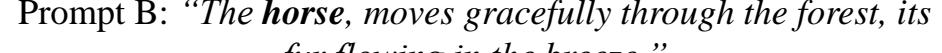
i = 16



i = 17



t = 18



Prompt B: *"The horse, moves gracefully through the forest, its fur flowing in the breeze."*

Prompt C: “*The tiger, which has horse legs and no black strips on its fur, moves gracefully through the forest, its fur flowing in the breeze.*”

Figure 5: Mixture of [“Tiger”] and [“Horse”]. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 17-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

D MORE EXAMPLES

In this section, we will show more experimental results that the video generated directly from the guidance prompt does not exhibit the desired mixed features from the prompts.

E FULL ALGORITHM

In this section, we provide the algorithm for 3D attention in Algorithm 4.

972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025

Prompt A: “*The eggplant, freshly washed, served in a dish, is placed on the wooden table.*”

Step:

$i = 1$



$i = 8$

$i = 9$

$i = 10$

$i = 30$

Prompt B: “*The orange, freshly washed, served in a dish, is placed on the wooden table.*”



Prompt C: “*The eggplant, with the color of yellow, freshly washed, served in a dish, is placed on the wooden table.*”

Figure 6: **Mixture of [“Eggplant”] and [“Orange”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 9-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1026
 1027
 1028
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038 **Prompt A:** “*The airplane landed gently on the runway, its*
 1039 *wheels touching the ground with precision.*”
 1040 Step:
 1041 $i = 1$ 
 1042 $i = 15$ 
 1043 $i = 16$ 
 1044 $i = 17$ 
 1045 $i = 30$ 
 1046
 1047
 1048
 1049
 1050 **Prompt B:** “*The horse stopped gracefully at the water's edge,*
 1051 *its reflection shimmering in the pond.*”
 1052 
 1053
 1054
 1055 **Prompt C:** “*The robot horse stopped gracefully at the water's*
 1056 *edge, its reflection shimmering in the pond.*”
 1057
 1058
 1059
 1060
 1061
 1062
 1063

1064 Figure 7: **Mixture of [“Airplane”] and [“Horse”].** Our objective is to mix the features described
 1065 in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
 1066 steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal em-
 1067 bedding and generate the corresponding video. The video generated directly from Prompt C does
 1068 not exhibit the desired mixed features from Prompts A and B.
 1069
 1070
 1071
 1072
 1073
 1074
 1075
 1076
 1077
 1078
 1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

Prompt A: “*An airplane soared above the clouds, its engines humming as it crossed the horizon.*”

Step:

 $i = 1$  $i = 14$ $i = 15$ $i = 16$ $i = 30$

Prompt B: “*An automobile drove along the winding mountain road, its engine purring smoothly.*”



Prompt C: “*An automobile with airplane wings soared above the clouds, its engines humming as it crossed the horizon.*”

Figure 8: **Mixture of [“Airplane”] and [“Automobile”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

Prompt A: “*An airplane glided smoothly in the sky.*”

1201

Step:

1202

 $i = 1$ 

1203

1204

 $i = 14$

1205

 $i = 15$

1206

 $i = 16$

1207

 $i = 30$

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

Prompt B: “*A deer moved quietly in the woods.*”Prompt C: “*An airplane with antlers, glided smoothly in the sky.*”

1226

Figure 10: **Mixture of [“Airplane”] and [“Deer”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

Step:

 $i = 1$ Prompt A: “*The airplane rested on the runway.*” $i = 16$ $i = 17$ $i = 18$ $i = 30$ Prompt B: “*The ship anchored at the harbor.*”Prompt C: “*The ship with the shape of airplane, anchored at the harbor.*”

Figure 11: **Mixture of [“Airplane”] and [“Ship”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 17-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

Prompt A: “*The airplane rested on the runway.*”

1309

Step:

 $i = 1$  $i = 16$ $i = 17$ $i = 18$ $i = 30$ Prompt B: “*The truck parked at the station.*”

1319

1320

1321

1322

1323

1324

1325

1326

1327

Prompt C: “*The airplane with the shape of truck, rested on the runway.*”

1331

1332

1333

Figure 12: **Mixture of [“Airplane”] and [“Truck”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 17-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362

Prompt A: “*The automobile parked by the forest edge.*”

1363 Step:

1364 $i = 1$



1365 $i = 14$

1366 $i = 15$

1367 $i = 16$

1368 $i = 30$

Prompt B: “*The deer stood quietly at the forest edge.*”



Prompt C: “*The deer with the body of a car, stood quietly at the forest edge.*”

1388 Figure 13: **Mixture of [“Automobile”] and [“Deer”].** Our objective is to mix the features described
1389 in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
1390 steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal em-
1391 bedding and generate the corresponding video. The video generated directly from Prompt C does
1392 not exhibit the desired mixed features from Prompts A and B.

1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

Prompt A: “*The automobile parked by the barn.*”1417
Step: $i = 1$  $i = 15$ $i = 16$ $i = 17$ $i = 30$ Prompt B: “*The horse stood quietly in the stable.*”Prompt C: “*The automobile with four horse legs, parked by the barn.*”

Figure 14: **Mixture of [“Automobile”] and [“Horse”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1447

1448

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458

1459

1460

1461

1462

1463

1464

1465

1466

1467

1468

1469

1470

1471

Step:

 $i = 1$ Prompt A: “*The automobile parked by the beach.*” $i = 14$ $i = 15$ $i = 16$ $i = 30$ Prompt B: “*The ship anchored at the harbor.*”Prompt C: “*The ship with the shape of the automobile, anchored at the harbor.*”

Figure 15: **Mixture of [“Automobile”] and [“Ship”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1501

1502

1503

1504

1505

1506

1507

1508

1509

1510

1511

1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525

Prompt A: “*A bird perched on a tree branch.*”

1526 Step:

1527 $i = 1$



1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565

Prompt C: “*A bird with a cat head, perched on a tree branch.*”

Figure 16: **Mixture of [“Bird”] and [“Cat”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 17-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1566

1567

1568

1569

1570

1571

1572

1573

1574

1575

1576

1577

1578

1579

1580 Step:

1581 $i = 1$ Prompt A: “*The bird perched on a branch.*”

1595

1596

1597

1598

1599

1600

1601

1602

1603

Figure 17: **Mixture of [“Bird”] and [“Dog”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 12-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1608

1609

1610

1611

1612

1613

1614

1615

1616

1617

1618

1619

Prompt B: “*The dog lay under the tree.*”Prompt C: “*A bird with four legs, perched on a tree branch.*”

1620

1621

1622

1623

1624

1625

1626

1627

1628

1629

1630

1631

1632

Prompt A: “*The bird perched on a branch.*”

1633

Step:

 $i = 1$ Prompt B: “*The deer stood under the tree.*” $i = 15$ $i = 16$ $i = 17$ $i = 30$ Prompt C: “*A bird with a deer head, perched on a tree branch.*”

Figure 18: **Mixture of [“Bird”] and [“Deer”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1663

1664

1665

1666

1667

1668

1669

1670

1671

1672

1673

1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740

Prompt A: “*An orchid bloomed gracefully in the greenhouse.*”

1741 Step:

1742 $i = 1$



1743 $i = 14$

1744 $i = 15$

1745 $i = 16$

1746 $i = 30$

Prompt B: “*A starfish rested quietly on the ocean floor.*”



Prompt C: “*A starfish with the shape of orchid, rested quietly on the ocean floor.*”

1766 Figure 20: **Mixture of [“Orchid”] and [“Starfish”].** Our objective is to mix the features described
1767 in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
1768 steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal em-
1769 bedding and generate the corresponding video. The video generated directly from Prompt C does
1770 not exhibit the desired mixed features from Prompts A and B.

1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781

1782

1783

1784

1785

1786

1787

1788

1789

1790

1791

1792

1793

1794

Step:

 $i = 1$ Prompt A: “*A sunflower bloomed brightly in the summer field.*” $i = 19$ Prompt A: “*A sunflower bloomed brightly in the summer field.*” $i = 20$ Prompt A: “*A sunflower bloomed brightly in the summer field.*” $i = 21$ Prompt A: “*A sunflower bloomed brightly in the summer field.*” $i = 30$ Prompt A: “*A sunflower bloomed brightly in the summer field.*”Prompt B: “*A starfish rested quietly on the ocean floor.*”Prompt C: “*A starfish with the shape and color of sunflower, rested quietly on the ocean floor.*”Prompt C: “*A starfish with the shape and color of sunflower, rested quietly on the ocean floor.*”

Figure 21: **Mixture of [“Sunflower”] and [“Starfish”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 20-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1825

1826

1827

1828

1829

1830

1831

1832

1833

1834

1835

1836

1837

1838

1839

1840

1841

1842

1843

1844

1845

1846

1847

1848

Step:

 $i = 1$ Prompt A: “*A sunflower stood tall in the garden.*”

1865

1866

1867

1868

1869

1870

1871

1872

1873

Prompt C: “*A snail with the shape and color of sunflower, crawled slowly across the ground.*”

1874

Figure 22: **Mixture of [“Sunflower”] and [“Snail”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 19-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1879

1880

1881

1882

1883

1884

1885

1886

1887

1888

1889

1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901

1902 **Prompt A: “A sunflower stood tall in the bright field.”**

1903 Step:

1904 $i = 1$



1905 $i = 16$

1906 $i = 17$

1907 $i = 18$

1908 $i = 30$

1909 **Prompt B: “A crab laid on the sandy beach.”**



1910 **Prompt C: “A crab with the shape of sunflower,
1911 laid on the sandy beach.”**

1912 Figure 23: **Mixture of [“Sunflower”] and [“Crab”].** Our objective is to mix the features described
1913 in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
1914 steps to 30. Using Algorithm 1, we identify the 17-th interpolation embedding as the optimal em-
1915 bedding and generate the corresponding video. The video generated directly from Prompt C does
1916 not exhibit the desired mixed features from Prompts A and B.

1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943

1944

1945

1946

1947

1948

1949

1950

1951

1952

1953

1954

1955

1956

Prompt A: “A **bird** perched on a tree branch.”

1957

Step:

 $i = 1$ Prompt B: “A **horse** stood under the tree.”

1973

1974

1975

1976

1977

1978

Prompt C: “A **bird** with a **horse** head, perched
on a tree branch.”

1979

1980

1981

Figure 24: **Mixture of [“Bird”] and [“Horse”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

1987

1988

1989

1990

1991

1992

1993

1994

1995

1996

1997

1998

1999

2000

2001

2002

2003

2004

2005

2006

2007

2008

2009

2010

Step:

 $i = 1$ Prompt A: “A **butterfly** landed delicately on a vibrant petal.”

2011

2012

2013

2014

2015

2016

2017

2018

2019

2020

2021

2022

2023

2024

2025

2026

2027

2028

2029

2030

2031

2032

2033

2034

2035

Prompt B: “A **starfish** clung to a coral reef beneath the waves.”Prompt C: “A **starfish with the shape of butterfly**, clung to a coral reef beneath the waves.”

Figure 25: **Mixture of [“Butterfly”] and [“Starfish”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2041

2042

2043

2044

2045

2046

2047

2048

2049

2050

2051

2052

2053

2054

2055

2056

2057

2058

2059

2060

2061

2062

2063

2064 **Prompt A: “A *cat* stretched lazily under the sun.”**

2065

Step:

2066

 $i = 1$ 

2067

2068

2069

2070

2071

2072

2073

2074

2075

2076

2077

2078

2079

2080

2081

2082

2083

2084

2085

2086

2087

2088

2089

Prompt B: “A *deer* rested peacefully under the tree.”Prompt C: “A *deer with a cat face*, rested peacefully under the tree.”

2090

Figure 26: **Mixture of [“Cat”] and [“Deer”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2095

2096

2097

2098

2099

2100

2101

2102

2103

2104

2105

2106
 2107
 2108
 2109
 2110
 2111
 2112
 2113
 2114
 2115
 2116
 2117
 2118
 2119
 2120 Step:
 2121 $i = 1$ 
 2122 $i = 15$ 
 2123 $i = 16$ 
 2124 $i = 17$ 
 2125 $i = 30$ 
 2126
 2127
 2128
 2129
 2130
 2131
 2132
 2133
 2134
 2135 Prompt B: “A **dog** lay quietly on the porch.”
 2136
 2137
 2138
 2139
 2140
 2141 Prompt C: “A **cat** with a **dog** face, sat quietly
 2142 on the windowsill.”
 2143
 2144 Figure 27: **Mixture of [“Cat”] and [“Dog”]**. Our objective is to mix the features described in
 2145 Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
 2146 steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal em-
 2147 bedding and generate the corresponding video. The video generated directly from Prompt C does
 2148 not exhibit the desired mixed features from Prompts A and B.
 2149
 2150
 2151
 2152
 2153
 2154
 2155
 2156
 2157
 2158
 2159

2160

2161

2162

2163

2164

2165

2166

2167

2168

2169

2170

2171

2172

Prompt A: “*The cat stretched lazily under the sun.*”

2173

Step:

 $i = 1$  $i = 12$  $i = 13$  $i = 14$  $i = 30$ Prompt B: “*The frog basked quietly under the sun.*”

2189

2190

2191

2192

2193

2194

Prompt C: “*The frog with a cat face, basked quietly under the sun.*”

2195

2196

2197

Figure 28: **Mixture of [“Cat”] and [“Frog”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 13-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2203

2204

2205

2206

2207

2208

2209

2210

2211

2212

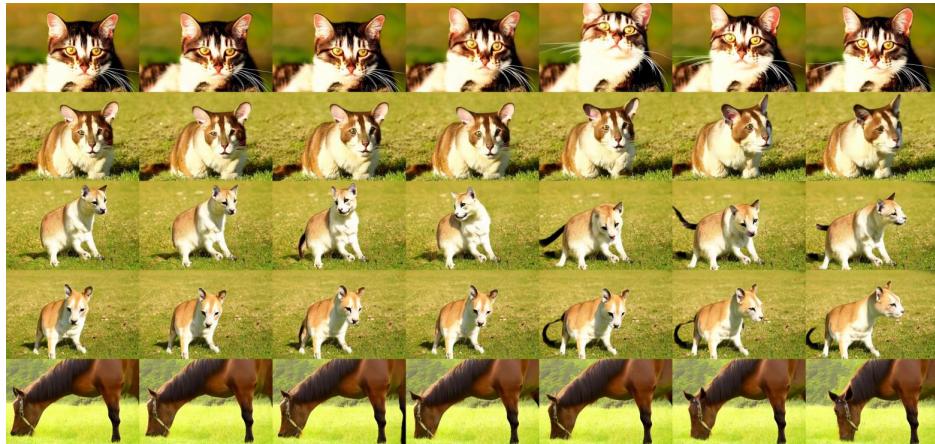
2213

2214
 2215
 2216
 2217
 2218
 2219
 2220
 2221
 2222
 2223
 2224
 2225
 2226

Prompt A: “*The cat stretched lazily in the sun.*”

2227
 Step:

2228
 2229
 $i = 1$



2230
 2231
 2232
 2233
 2234
 2235
 2236
 2237
 2238
 2239
 2240
 2241
 2242

Prompt B: “*The horse grazed peacefully in the meadow.*”



2243
 2244
 2245
 2246
 2247
 2248
 2249
 2250
 2251

Prompt C: “*The horse with a cat face, grazed peacefully in the meadow.*”

2252
 2253
 2254
 2255
 2256
 2257
 2258
 2259
 2260
 2261
 2262
 2263
 2264
 2265
 2266
 2267

Figure 29: **Mixture of [“Cat”] and [“Horse”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 11-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2268
 2269
 2270
 2271
 2272
 2273
 2274
 2275
 2276
 2277
 2278
 2279

2280 Prompt A: “*The otter rested on a smooth stone by the water.*”

2281 Step:

2282 $i = 1$



2283 $i = 13$

2284 $i = 14$

2285 $i = 15$

2286 $i = 30$

2287 Prompt B: “*The lizard basked on the tree bark under the sun.*”



2288 Prompt C: “*The otter with lizard skin, rested on a smooth*
 2289 *stone by the water.*”

2290
 2291
 2292
 2293
 2294
 2295
 2296
 2297
 2298
 2299
 2300
 2301
 2302
 2303
 2304
 2305
 2306 Figure 30: **Mixture of [“Otter”] and [“Lizard”].** Our objective is to mix the features described in
 2307 Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation
 2308 steps to 30. Using Algorithm 1, we identify the 14-th interpolation embedding as the optimal em-
 2309 bedding and generate the corresponding video. The video generated directly from Prompt C does
 2310 not exhibit the desired mixed features from Prompts A and B.

2311
 2312
 2313
 2314
 2315
 2316
 2317
 2318
 2319
 2320
 2321

2322
 2323
 2324
 2325
 2326
 2327
 2328
 2329
 2330
 2331
 2332
 2333
 2334

Prompt A: “A *kangaroo* rested in the shade of a tall tree.”

2335
 Step:

2336
 2337
 $i = 1$



2338
 2339
 $i = 15$



2340
 2341
 $i = 16$



2342
 2343
 $i = 17$



2344
 2345
 $i = 30$



Prompt B: “A *lizard* basked in the sunlight on a flat rock.”

2351
 2352
 2353
 2354
 2355
 2356
 2357
 2358
 2359



Prompt C: “A *lizard with kangaroo legs*, basked in the sunlight on a flat rock.”

2360
 2361
 2362
 2363
 2364
 Figure 31: **Mixture of [“Kangaroo”] and [“Lizard”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2365
 2366
 2367
 2368
 2369
 2370
 2371
 2372
 2373
 2374
 2375

2430

2431

2432

2433

2434

2435

2436

2437

2438

2439

2440

2441

2442

Prompt A: “A **dog** lay lazily in the sun.”

2443

Step:

 $i = 1$  $i = 15$  $i = 16$  $i = 17$  $i = 30$ Prompt B: “A **frog** sat still on a rock in the sun.”

2459

2460

2461

2462

2463

2464

Prompt C: “A **frog with a dog head**, sat still on a rock in the sun.”

2465

2466

2467

Figure 33: **Mixture of [“Dog”] and [“Frog”]**. Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 16-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2473

2474

2475

2476

2477

2478

2479

2480

2481

2482

2483

2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496

Prompt A: “*The red panda rested on a sturdy branch.*”

2497 Step:

2498 $i = 1$



2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511

Prompt B: “*The rabbit sat still under a tree.*”

2512
2513
2514
2515
2516
2517
2518
2519
2520
2521



Prompt C: “*The red panda with a rabbit face, rested on a sturdy branch.*”

2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537

Figure 34: **Mixture of [“Red Panda”] and [“Rabbit”].** Our objective is to mix the features described in Prompt A and Prompt B with the guidance of Prompt C. We set the total number of interpolation steps to 30. Using Algorithm 1, we identify the 15-th interpolation embedding as the optimal embedding and generate the corresponding video. The video generated directly from Prompt C does not exhibit the desired mixed features from Prompts A and B.

2538
 2539
 2540
 2541
 2542
 2543
 2544
 2545
 2546
 2547
 2548

Algorithm 4 3D Attention

2549 1: **datastructure** 3D ATTENTION ▷ Definition 4.7
 2550 2: **members**
 2551 3: $n \in \mathcal{N}$: the length of input sequence
 2552 4: $n_f \in \mathcal{N}$: the number of frames
 2553 5: $h \in \mathcal{N}$: the height of video
 2554 6: $w \in \mathcal{N}$: the width of video
 2555 7: $d \in \mathcal{N}$: the hidden dimension
 2556 8: $c \in \mathcal{N}$: the channel of video
 2557 9: $c_{\text{patch}} \in \mathbb{R}^{n \times d}$: the channel of patch embedding.
 2558 10: $E_t \in \mathbb{R}^{n \times d}$: the text embedding.
 2559 11: $E_{\text{video}} \in \mathbb{R}^{n_f \times h \times w \times c}$: the video embedding.
 2560 12: $E_{\text{patch}} \in \mathbb{R}^{n_f \times h' \times w' \times c_{\text{patch}}}$: the patch embedding.
 2561 13: $\phi_{\text{conv}}(X, c_{\text{in}}, c_{\text{out}}, p, s)$: the convolution layer. ▷ Definition 4.5
 2562 14: $\text{Attn}(X)$: the attention block. ▷ Definition 4.4
 2563 15: $\phi_{\text{linear}}(X)$: the linear projection. ▷ Definition 4.6
 2564 16: **end members**
 2565 17:
 2566 18: **procedure** 3D ATTENTION($E_t \in \mathbb{R}^{n \times d}, E_v \in \mathbb{R}^{n_f \times h \times w \times c}$)
 2567 19: /* E_{patch} dimension: $[n_f, h, w, c_v] \rightarrow [n_f, h', w', c_{\text{patch}}]$ */
 2568 20: $E_{\text{patch}} \leftarrow \phi_{\text{conv}}(E_v, c_v, c_{\text{patch}}, p = 2, s = 2)$
 2569 21: /* E_{patch} dimension: $[n_f, h', w', c_{\text{patch}}] \rightarrow [n_f \times h' \times w', c_{\text{patch}}]$ */
 2570 22: $E_{\text{patch}} \leftarrow \text{reshape}(E_{\text{patch}})$
 2571 23: /* E_{hidden} dimension: $[n + n_f \times h' \times w', c_{\text{patch}}]$ */
 2572 24: $E_{\text{hidden}} \leftarrow \text{concat}(E_t, E_{\text{patch}})$
 2573 25: /* E_{hidden} dimension: $[n + n_f \times h' \times w', c_{\text{patch}}]$ */
 2574 26: $E_{\text{hidden}} \leftarrow \text{Attn}(E_{\text{hidden}})$
 2575 27: /* E_t dimension: $[n, d]$ */
 2576 28: /* E_{patch} dimension: $[n_f \times h' \times w', c_{\text{patch}}]$ */
 2577 29: $E_t, E_{\text{patch}} \leftarrow \text{split}(E_{\text{hidden}})$
 2578 30: /* E_v dimension: $[n_f \times h' \times w', c_{\text{patch}}] \rightarrow [n_f \times h \times w, c_v]$ */
 2579 31: $E_v \leftarrow \phi_{\text{linear}}(E_{\text{patch}})$
 2580 32: /* E_v dimension: $[n_f \times h \times w, c_v] \rightarrow [n_f, h, w, c_v]$ */
 2581 33: $E_v \leftarrow \text{reshape}(E_v)$
 34: **Return** E_v
 35: **end procedure**

2582
 2583
 2584
 2585
 2586
 2587
 2588
 2589
 2590
 2591

2592 **LLM USAGE DISCLOSURE**

2593

2594 LLMs were used only to polish language, such as grammar and wording. These models did not
2595 contribute to idea creation or writing, and the authors take full responsibility for this paper's content.

2596

2597

2598

2599

2600

2601

2602

2603

2604

2605

2606

2607

2608

2609

2610

2611

2612

2613

2614

2615

2616

2617

2618

2619

2620

2621

2622

2623

2624

2625

2626

2627

2628

2629

2630

2631

2632

2633

2634

2635

2636

2637

2638

2639

2640

2641

2642

2643

2644

2645