arXiv:2602.09484v1 [eess.IV] 10 Feb 2026

越小越好：生成模型可为短视频预加载提供动力

Liming Liu、Jiangkai Wu、Xinggong Zhang 北京大学，中国

摘要—**短视频平台广泛采用预加载，通过提前下载后续内容来尽可能减少播放卡顿。然而，现有策略面临取舍：激进的预加载策略可以减少卡顿，却会浪费带宽；保守的策略可以节省流量，却会增加播放卡顿的风险。本文提出 PromptPream，这是一种计算驱动的预加载范式，通过利用本地计算降低带宽需求，打破这一取舍。PromptPream 不传输像素级视频块，而是发送紧凑的语义提示，再利用 Stable Diffusion 等生成模型将其解码为高质量帧。我们提出了三项核心技术来实现这一范式：（1）一种基于梯度的提示反演 方法，可将帧压缩为少量紧凑的词元嵌入；（2）一种计算感知调度策略，可联合优化网络和计算资源的使用；（3）一种可扩展的搜索算法，用于应对调度器引入的更大调度空间。评估结果表明，与传统策略相比，PromptStream 可将卡顿和带宽浪费均减少 31% 以上，并将体验质量（QoE）提升 45%。**

一、引言

TikTok 和 Instagram Reels 等短视频平台是移动生态系统中最耗带宽的应用之一 [1]。为了降低感知延迟并改善用户体验，这些平台严重依赖在用户实际观看视频之前预加载视频块 [2], [3]。预加载可以掩盖网络卡顿并使播放更加流畅，但也会带来一个长期存在且广为人知的两难问题：当带宽或用户行为预测不准确时，系统要么预加载过少，导致播放卡顿 [4], ，要么预加载过多，将带宽浪费在用户未观看的内容上 [5]。这种卡顿–浪费权衡并非只是参数调优问题，而是现有预加载系统工作方式的根本局限：大多数系统只会对是否预加载特定视频块做出二元决策，无法权衡分块保真度、大小或解码成本。

本文探索一种新范式：利用本地计算降低带宽需求，从而缓解卡顿和浪费。我们的核心想法是利用普遍可用的 GPU 硬件；在视频解码期间，GPU 几乎处于闲置状态。超分辨率[6] 或更先进的神经编解码器[7]也可以用本地计算换取更高质量，而我们则将这一思路进一步推进。我们不再传输像素级编码，而是发送紧凑的语义提示 ，由现代生成模型 [8]对其进行解码，以重建高质量帧，从而在相同带宽约束下提供更大的画质提升潜力。

但要将这一新想法转化为实用的系统，还面临诸多挑战。首先，我们需要一种方法，将视频帧转换为能够保留视觉保真度的最简表示。其次，计算资源限制使传统的预加载策略失效，因此需要新的机制，在带宽、解码延迟和计算资源限制等因素下评估并安排数据块。最后，我们必须在一个庞大且相互依赖的空间中做出预加载决策，其中涉及混合编解码器、不同的解码时间以及乱序执行；选项数量之多，使得传统的决策算法（如模型预测控制（MPC）） [9] 在计算上难以实现。

为应对这些挑战，我们提出了一个由三个关键模块组成的系统。首先，我们开发了一个**基于梯度的提示反演模块(§IV)**，将视频帧转换为紧凑的语义表示。通过反向传播优化文本嵌入，我们生成少量学习得到的词元；这些词元与简短的文本提示一起传输到客户端，并使用 Stable Diffusion 解码为高质量的视频帧。其次，我们设计了一种**计算感知调度策略(§V)**，同时考虑带宽和解码延迟。我们扩展了卡顿时间的定义，将解码延迟也纳入其中，并引入一种乱序下载机制，通过在不同计算单元上并行执行提示解码、H.265 解码和网络传输，提高硬件利用率。最后，我们提出了一种**基于树的搜索算法(§VI)**，以降低规划开销。通过结合蒙特卡洛树搜索与剪枝技术，我们能够高效探索由多种编解码器、比特率、解码延迟和下载顺序所形成的指数级巨大决策空间。

结果令人鼓舞。三个模块协同工作时，我们的系统显著提升了短视频预加载的性能。与现有策略相比，该系统将卡顿和带宽浪费减少了 31% 以上，同时将用户平均体验质量提升了 45%。这些提升展现了利用本地计算提高短视频预加载效率和质量的实际潜力。

II. 动机与 C挑战

A. 卡顿和浪费仍不可避免

短视频平台广泛采用视频预加载来提升播放体验 [2], [3]。系统通过预测用户接下来可能观看的内容，并提前下载部分视频，旨在减少卡顿并确保观看流畅。然而，在实际应用中，预加载仍面临两个问题：播放卡顿和带宽浪费 [2]。

卡顿—浪费困境的根本原因在于预测误差：很难预判用户会继续观看视频多长时间

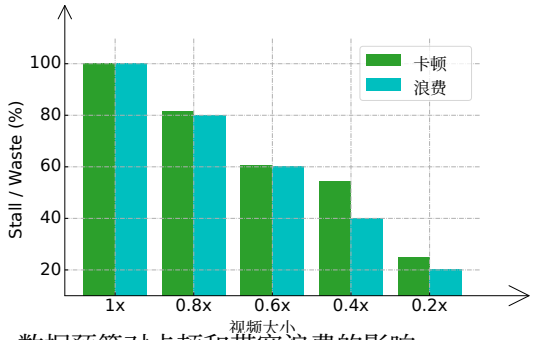


图 1：数据预算对卡顿和带宽浪费的影响。

，[10]也很难预判未来可用的带宽有多少 [11]。因此，预加载决策往往涉及二元权衡。预加载过少，若后续数据块未能及时送达，就可能导致播放卡顿 [4],；预加载过多，若用户提前滑走，则会浪费带宽 [5]。系统必须在相互竞争的目标之间做出选择，往往要牺牲其中一个来保障另一个。

以往的方法 [2], [3] 尝试通过设计更好的预测器或评分规则来改进预加载决策。但只要决策空间仍然是二元的，这种权衡就无法从根本上得到解决。

B. 我们能否同时减少卡顿和浪费？

但如果我们能在不降低质量的情况下减少这种成本呢？我们的关键洞见是，在不损害感知质量的前提下减小数据块大小，可以同时减少播放卡顿和带宽浪费。

为了探索这一想法，我们在 PDAS [12]模拟器上进行了一项实验。该模拟器是广泛用于评估短视频预加载系统的框架。我们唯一做的改动是减小数据集中每个视频块的大小，同时保持其质量指标、用户滑动轨迹、带宽轨迹和预加载算法不变。这模拟了这样一种系统：视频所需的带宽更少，却不会牺牲视觉质量。

结果令人瞩目。如图 1 所示，减小数据块大小可使卡顿时间和带宽浪费均减少 70% 以上。此外，由于每个数据块消耗的带宽更少，系统便有了更大的余量，可以在不超出带宽预算的情况下选择更高的比特率。

这表明，如果我们能找到一种在不损害感知质量的前提下缩小数据块的方法，就能获得一个新的机会：同时提升性能和用户体验。

C. 利用计算打破权衡

但如何实现呢？虽然超分辨率[6] 或更先进的神经编解码器[7] 也能以本地计算换取更高画质，但我们将这一理念更进一步。我们不再传输像素级编码，而是发送简洁的语义提示，由现代生成模型 [8] 解码并重建高质量画面；在相同带宽约束下，这种方式能为画质提升带来更大的潜力。更重要的是，其中许多模型都是完全开源的，并且可以优化为在现代移动硬件上运行，包括消费级智能手机中已有的 GPU 和 NPU [13]。

通过将大多数用户设备上已有的本地计算能力引入预加载流程，我们如今可以用计算换取更少的传输，而不再仅仅依赖带宽和预测准确率。

D. 挑战

尽管利用计算来减少带宽的想法颇具吸引力，但要在实际的短视频预加载系统中实现这一点，仍面临若干重大挑战。

a) 挑战 *1*：构建紧凑且有效的语义表示： 用语义输入替代传统视频数据，需要一种方法将每一帧编码为最精简的表示，同时仍支持高保真重建。我们通过设计一个基于梯度的提示反演模块来解决这一问题，详见第 IV 节。

b) 挑战 *2*：在计算与带宽约束下进行调度： 与传统编解码器不同，使用扩散模型解码基于提示的表示会引入显著延迟，并从根本上改变预加载决策的制定方式。当解码成为瓶颈时，评分函数还必须纳入计算成本。为应对这一挑战，我们开发了一个计算感知调度器，根据带宽和计算资源的可用情况重新安排下载顺序并调整数据块优先级排序，详见第 V 节。

c) 挑战 *3*：在庞大且灵活的决策空间中进行搜索： 将计算纳入预加载流程后，我们必须在一个时间窗口内进行推理，以了解一系列下载与解码决策将如何相互影响。此外，我们新的数据块评分机制还带来了额外复杂性：每种比特率都有提示增强选项，而且这些选项可以乱序下载和解码。为解决这一问题，我们设计了一种基于蒙特卡洛树搜索的结构化决策搜索过程，并结合剪枝策略，及早舍弃前景不佳的分支，详见第 VI 节。

III. 系统概述

如图2所示，我们的系统通过引入两个新组件扩展了传统的视频传输流水线：一个基于提示的编码器-解码器对，以及一个感知计算状况的决策模块。这些新增组件使系统能够利用闲置计算资源来优化预加载性能。系统围绕三个角色构建：编码器 负责为每个视频块准备多种表示形式，包括传统的 H.265 编码和基于提示的编码；服务器 存储所有候选版本及其元数据；客户端 则根据网络和计算状况决定下载哪些内容、按何种顺序下载，以及何时解码。下文将详细介绍编码器和观看端流水线。

a) 编码流水线：给定一段视频，编码器会为每个数据块生成多个版本。第一种是由标准视频编码器生成的、具有不同比特率的常规 H.265 比特流集合。第二种是通过我们的基于梯度的提示反演模块获得的基于提示的表示，该模块会生成紧凑的文本嵌入

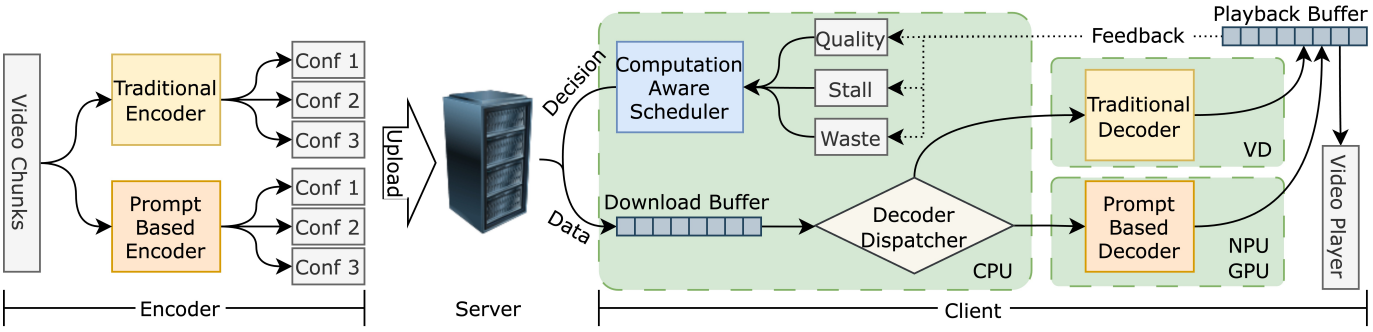


图 2：**系统概述**。 我们的系统通过引入基于提示的编码和计算感知调度，扩展了传统的短视频预加载。该系统包含三个角色：编码器，用于生成传统（H.265）和基于提示的表示；服务器，用于存储每个数据块的多个选项；以及观看端，利用计算感知调度器决定下载和解码哪些数据块。观看端利用 CPU、GPU、NPU 和视频解码器（VD）之间的并行性，并通过解码器分发器将每个数据块路由到适当的后端。

能够通过扩散模型重建视频帧。所有表示都会上传到服务器，并与解码延迟、比特率和估计质量等元数据一起存储。

b) 客户端调度器与解码器： 观看视频时，客户端负责决定下载哪些数据块、以何种顺序下载，以及采用何种格式。这些决策由我们的计算感知调度器作出，它会考虑网络带宽、计算能力、解码延迟和播放截止时间。使用 H.265 编码的数据块由 CPU 和专用视频解码（VD）单元解码，而基于提示的数据块则通过扩散模型在 GPU 或 NPU 上重建。由于这些硬件资源彼此独立运行，系统利用这种并行性来提高利用率并减少卡顿。调度器可以按任意顺序下载数据块，也可以在下载传统数据块的同时并行预解码基于提示的关键帧，从而灵活规划播放。

IV. 提示反演

本节详细介绍我们如何利用基于梯度的提示反演框架生成视频帧的紧凑语义表示，该框架构建于 Stable Diffusion 之上。随后，我们解释为实现最小带宽使用量和高感知质量而选择反演哪些组件。最后，我们介绍如何将提示编码与传统的 H.265 压缩相结合，以加速解码并降低计算开销。

A. 基于梯度的提示反演

为实现语义视频压缩，我们采用 Stable Diffusion 生成模型，该模型在文本条件下通过对随机噪声进行去噪来合成图像。文本条件的构造方式是：使用分词器 [14] 模块将词汇表中的词元嵌入为词向量。随后，CLIP [15]模型处理这些词嵌入，生成与图像空间对齐的句子级嵌入。包括 CLIP 和扩散去噪模块在内的整个流水线都是完全可微的。

如图3所示，我们的编码流水线首先使用

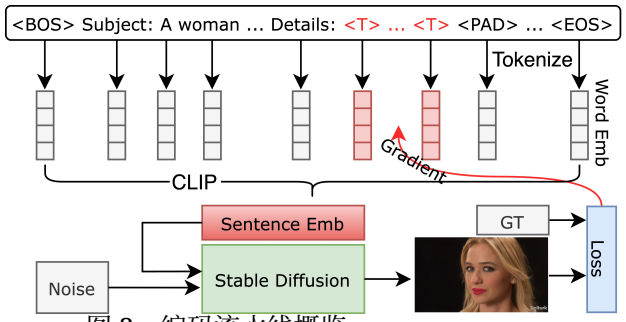


图 3：编码流水线概览。

图像转文本模块 [16]生成输入帧的文本描述。接着，我们在该描述后追加一些可学习标记（记作 <T>），构建完整的 Stable Diffusion 提示词。这些可学习标记的嵌入是我们优化过程的目标。

该提示词会输入标准的 Stable Diffusion 流程。首先由分词器进行分词，然后由 CLIP 模块编码为句子级表示。该句子级嵌入作为扩散模型的文本条件，扩散模型对固定的随机噪声向量进行去噪，从而生成重建图像。使用固定的噪声种子可确保梯度在各次迭代中保持稳定一致。随后，使用逐像素均方误差损失和感知 LPIPS 损失，将重建图像与原始输入帧进行比较。梯度会通过扩散模型和 CLIP 编码器反向传播，但只更新可学习 <T> 词元的嵌入。经过足够多次迭代后，这些词元便能重建出与原始帧高度相似的近似图像。

B. 应该反演什么？句子还是词元？

一个关键问题是，应该反演文本条件的哪个层级，才能获得最佳的压缩效果和重建质量。先前的工作 [17] 提出，可以反演句子级嵌入，因为它具有低秩结构，可通过矩阵分解进行降维。

我们开展了实验，对这一低秩假设提出了质疑。首先，如图 4 所示，我们发现，尝试使用低秩张量近似句子嵌入，会产生抽象的重建结果，并且

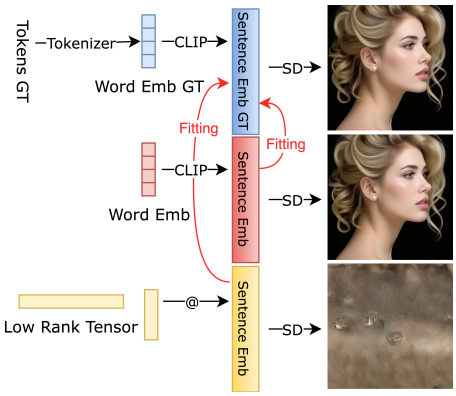


图 4：句子嵌入的低秩近似重建无法保留细节，表明低秩假设并不成立。

与原始图像存在显著偏差，这表明该嵌入缺乏低秩结构。其次，我们证明，仅使用一小组词元级嵌入——只有四个词元，数据量远少于完整的句子嵌入——经过优化后，便能以几乎与原始句子嵌入同样出色的效果重建图像。这些发现表明，相较于反演句子嵌入，反演词元级嵌入能够实现更好的压缩和更高的重建保真度，因为词元嵌入更符合扩散模型所预期的分布。

优化词级嵌入还有一个优点：它们与文本提示处于同一空间。这使系统能够利用 CLIP 和底层语言模型强大的语义先验与组合能力，从而实现表现力更强、可解释性更好的重建。通过联合传输一条简短的文本提示和一小组优化后的词元嵌入，我们可以用更少的数据获得更高的重建质量。

C. 通过混合编码加速解码

虽然对每一帧都使用基于提示的反演可以实现高压缩率，但其计算量很大，因此不适合在移动电话等资源受限设备上进行实时解码 [18]。然而，如表 I 所示，在视觉质量（例如 LPIPS [19]）相当时，关键帧的提示嵌入所占比特数明显少于采用 H. 编码的传统 I 帧 265 [20]。这一发现促使我们采用混合策略：仅对 I 帧使用基于提示的反演，其余帧则采用传统的低比特率 B/P 帧压缩。

我们的实验表明，与纯 H.265 编码相比，该混合编码方案在感知质量相同的情况下，可将比特率降低 50% 以上，在计算成本与带宽效率之间实现了实用的权衡。

V. 计算感知策略

A. 数据块评分

基于提示的解码压缩效果出色，但延迟很高。解码一个 1 秒的 1080p 数据块

表 I：不同目标比特率下 I 帧的质量（LPIPS ↓）和帧大小（KB）比较。我们的关键帧始终能实现更优的质量与大小权衡。

比特率（kbps）	Ours		H.265 I帧	
	LPIPS ↓	大小（KB）	LPIPS ↓	大小（KB）
200	0.459	8.8	0.543	7.4
400	0.459	8.8	0.510	11.2
600	0.459	8.8	0.494	14.8
1200	0.459	8.8	0.454	27.7

在移动端 GPU 或 NPU 上可能需要超过 1400 毫秒，而 H.265 解码耗时不到 1 毫秒。因此，不能简单地将所有数据块都替换为基于提示的格式。相反，系统必须谨慎混合基于提示的数据块和 H.265 块，并对其进行调度，以满足播放截止时间。

因此，我们定义了一种计算感知评分函数，综合考虑带宽、计算量和播放截止时间，以指导预加载决策。每个数据块 i 都关联四项指标：视觉质量 q_i、质量变化 $v_i = |q_i - q_{i-1}|$、卡顿时长 σ_i和带宽成本 b_i。这些指标反映了用户感知保真度、流畅度和带宽消耗。卡顿时长 σ_i 根据下载、解码和播放调度计算。设 δ_i 为数据块 i 的下载时长，χ_i 为解码时间，π_i 为播放时长。设 Δ_i、Γ_i、β_i和 Φ_i 分别表示下载结束时间、解码结束时间、缓冲区可用时间和播放开始时间。这些值通过递归计算得出：

$$\Delta_i = \Delta_{i-1} + \delta_i, \quad \Delta_0 \triangleq 0 \tag{1}$$

$$\Gamma_i = \max(\Delta_i, \Gamma_{i-1}) + \chi_i, \quad \Gamma_0 \triangleq 0 \tag{2}$$

接着，我们将卡顿时长定义为超过缓冲区可用时间的延迟：

$$\sigma_i = \max(0, \Gamma_i - \beta_i) \tag{3}$$

根据这些定义，我们将块级评分f_i 计算为加权和：

$$f_i = w_1 q_i - w_2 v_i - w_3 \sigma_i - w_4 b_i \tag{4}$$

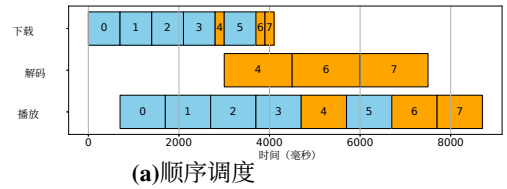
其中，w_1–w_4 是可调权重，用于反映系统在质量、流畅度、延迟和效率方面的偏好。

最后，我们的目标是选择一个调度计划，使所有视频块的总效用最大化。形式上，我们要寻找满足以下条件的最优计划 $\mathcal{P}^*$：

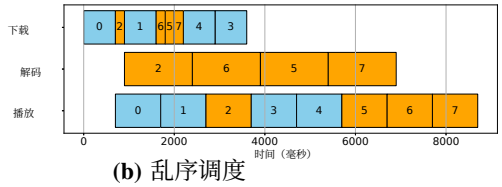
$$\mathcal{P}^* = \arg\max_{\mathcal{P}} \sum_{i=1}^{N} f_i \tag{5}$$

B. 乱序，性能更优

然而，一旦引入较大的解码延迟，传统的顺序下载策略就会变得低效。我们的关键洞见是，现代移动硬件支持多个并发计算单元：CPU 与视频解码器负责 H.265 解码，GPU/NPU 负责提示词解码，而下载则由 CPU 管理。因此，下载与解码可以并行进行，从而更高效地调度块。



(a)顺序调度



(b) 乱序调度

图5：顺序与乱序分块加载策略的对比。乱序调度能够更早、更高效地进行提示词分块解码，从而以提示格式提供更多内容，减少带宽使用量并提高计算资源利用率。

考虑一个简单示例：每个视频块代表 1000ms 的播放时长。基于提示词的分块需要 200ms 下载和 1500ms 解码，而 H.265 块需要 700ms 下载，解码延迟可忽略不计。我们比较两种调度策略：一种严格按播放顺序下载块，另一种允许乱序选择，以最大化计算与带宽利用率。

如图5所示，在顺序策略下，系统会下载5个 H.265 块，并且在该时间窗口内只能下载3个提示块。只有播放缓冲区足够长后，才能开始提示解码，这会导致 GPU/NPU 资源利用不足，并延迟提示格式的使用。相比之下，乱序调度器会更早开始下载和解码提示块。这样一来，系统可以在相同时间预算内解码4个提示块，从而有效地将更多内容转换为紧凑格式。

VI. 使用蒙特卡洛树搜索与剪枝进行规划

我们的系统必须针对多个未来数据块的规划时域做出预加载决策。对于每个数据块 c_i，都有一组候选配置 $R_i = \{r_1^i, r_2^i, \dots\}$，对应不同的编码方式（例如 H.265 与提示格式）、比特率和解码延迟。可能方案的总数会呈指数级增长，即 $\prod_{i=1}^{H} |R_i|$，因此无法进行穷举。传统的模型预测控制（MPC）方法只探索数量有限的候选轨迹，无法满足这一规模的需求。

为应对这一挑战，我们设计了一种基于蒙特卡洛树搜索（MCTS）的规划算法 [21]。我们将规划时域定义为 H 个数据块。在每一步中，规划器都会依据 UCT 规则采样有潜力的决策路径 [22]:

$$\mathrm{UCT}(s) = \frac{V(s)}{n(s)} + \alpha \cdot \sqrt{\frac{\log N}{n(s)}} \tag{6}$$

其中，$n(s)$ 是 s 被访问的次数，N 是从根节点开始的模拟总次数，α 是可调节的探索参数，而其奖励 $V(s)$ 则是从根节点到该节点 s 的分块评分之和:

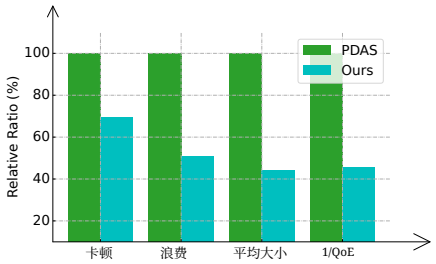


图 6：PDAS 与我们的方法的归一化指标对比。所有 PDAS 数值均归一化为 100%。与 PDAS 相比，我们的方法在卡顿、浪费、视频大小和 QoE 倒数方面的数值都显著更低，表明整体性能更佳。

$$V(s) = \sum_{j=1}^{i} f_j \tag{7}$$

为进一步缩小搜索空间，我们采用两种关键技术：激进剪枝和提前停止。

首先，在节点扩展期间，我们会剪除任何因解码延迟而导致播放卡顿的配置。具体而言，如果满足以下条件，则丢弃该子节点:

$$\mathrm{ComputeStall}_j > 0 \quad \text{for any } j \le i \tag{8}$$

这种严格的剪枝可确保树中的任何部分计划都不会因计算延迟而引入卡顿，从而保证只探索可行路径。

其次，我们引入提前停止机制，以确保在运行时约束下及时做出决策。如果规划周期的剩余时间不足以继续探索，规划器可以立即选择迄今为止找到的最佳完整计划或部分计划。这样，系统便能在严格的运行时预算内生成高质量的调度。

VII. 性能评估*A.* 视频质量

我们比较我们的方法、神经编解码器NVC [23]，以及基于超分辨率的编解码器NAS [6]。所有方法均在280 kbps的相同平均比特率约束下进行评估。如图7中LPIPS [24]分数的累积分布函数（CDF）所示，我们的方法显著提升了感知质量。具体而言，LPIPS大于0.37的帧所占比例相较H.265降低了40%，表明视觉保真度有大幅提升。

B. 预加载性能

我们使用 Grand Challenge[25]提供的轨迹来评估调度器，该调度器在各项关键指标上均实现了显著的端到端改进。如图 6 所示，与 PDAS 基线相比，我们的方法减少了卡顿时间、浪费数据和整体视频大小，同时提升了体验质量。这些提升体现了对带宽和设备端计算资源的更充分利用，也证明了联合优化解码成本与调度顺序的有效性。

C. 模块消融

词级嵌入优化。这种改进提升了感知质量：平均 LPIPS 从 0.372（句子级）降至 0.356（词级），这证实了

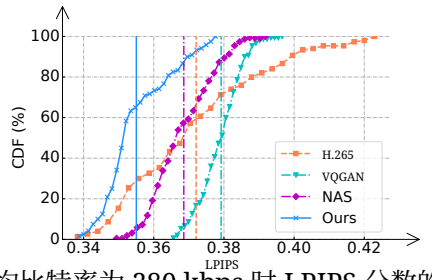


图 7：平均比特率为 280 kbps 时 LPIPS 分数的 CDF。与 H.265 相比，我们的方法将低质量帧的比例降低了 40%。

更精准的语义对齐能够有效提高帧重建的保真度。

蒙特卡洛树搜索。 当预测窗口设置为 7 时，暴力枚举需要探索全部 $7! \times 6^7 \approx 1.4 \times 10^9$ 种顺序—配置组合。相比之下，我们的蒙特卡洛树方法仅需经过 1.2×10^5 次评估即可收敛到相同的最优解，实现超过 $10^4\times$的加速。这一搜索步骤的减少直接转化为更低的计算成本。

VIII. 结论

本文提出了一种短视频预加载新范式，利用本地计算降低带宽需求，并缓解根本性的卡顿—浪费困境。通过利用用户设备上大量闲置的计算资源，我们的系统传输紧凑的语义提示，而非像素级编码，使生成模型能够在严格的码率约束下重建高质量帧。为使这一设想切实可行，我们设计了三个关键模块：用于语义压缩的基于梯度的提示反演模块、用于协同实现带宽—时延优化的计算感知调度器，以及一种基于树的搜索算法，可高效探索编解码器和解码顺序所构成的广阔决策空间。实验结果表明，与现有预加载策略相比，我们的方法将卡顿和带宽浪费减少了 31% 以上，并将用户体验质量提升了 45%。我们相信，这项工作为计算辅助视频传输开辟了一个前景广阔的方向。本工作得到国家自然科学基金委员会项目（62431017）和字节跳动项目（CT20241126107484）的资助。我们衷心感谢媒体融合生产技术与系统国家重点实验室、智能出版媒体技术重点实验室提供的支持。通讯作者为 Xinggong Zhang（zhangxg@pku.edu.cn）。

参考文献

[1] Sandvine，《2024年全球互联网现象报告》，2024，https://www.sandvine.com/global-internet-phenomena-report-2024。[2] N. T. Phong、T. T. Huong、P. N. Nam、T. C. Thang 和 D. Nguyen，《短视频流媒体的联合预加载与码率自适应》，*IEEE Access*，第11卷，第121 064–121 076页，2023。[3] H. Zhang、Y. Ban、X. Zhang、Z. Guo、Z. Xu、S. Meng、J. Li 和 Y. Wang，《APL：基于李雅普诺夫优化的短视频自适应预加载》，载于*2020 IEEE*国际视觉通信与图像处理会议（*VCIP*）。IEEE，2020，第13–16页。[4] H. T. Le、D. V. Nguyen、N. P. Ngoc、A. T. Pham 和 T. C. Thang，《用于自适应HTTP流媒体的基于缓冲区的码率自适应》，载于*2013*年先进通信技术国际会议（*ATC 2013*）。IEEE，2013，第33–38页。[5] Y. Zhang、Y. Liu、L. Guo 和 J. Y. Lee，《通过移动和无线网络对大型短视频服务进行测量》，IEEE移动计算汇刊，第22卷，第6期，第3472–3488页，2022。[6] H. Yeo、Y. Jung、J. Kim、J. Shin 和 D. Han，《神经自适应内容感知互联网视频传输》，载于第13届USENIX操作系统设计与实现研讨会（OSDI 18），2018，第645–661页。[7] Z. Jia、B. Li、J. Li、W. Xie、L. Qi、H. Li 和 Y. Lu，《迈向实用的实时神经视频压缩》，载于计算机视觉与模式识别会议论文集，2025，第12 543–12 552页。[8] R. Rombach、A. Blattmann、D. Lorenz、P. Esser 和 B. Ommer，《利用潜在扩散模型进行高分辨率图像合成》，载于IEEE/CVF计算机视觉与模式识别会议论文集（CVPR），2022，第10 684–10 695页。[9] B. Kouvaritakis 和 M. Cannon，《模型预测控制》，瑞士：施普林格国际出版公司，第38卷，第13-56期，第7页，2016。[10] L. Zhang、Y. Zhang 和 J. Yan，《基于用户偏好的短视频流媒体预加载与自适应码率算法》，载于数字电视与无线多媒体通信国际论坛。施普林格，2023，第401–412页。[11] E. Kurdoglu、Y. Liu、Y. Wang、Y. Shi、C. Gu 和 J. Lyu，《蜂窝网络视频通话的实时带宽预测与码率自适应》，载于第7届多媒体系统国际会议论文集，2016，第1–11页。[12] C. Zhou、Y. Ban、Y. Zhao、L. Guo 和 B. Yu，《PDAS：面向短视频的概率驱动自适应流媒体》，载于第30届ACM多媒体国际会议论文集，2022，第7021–7025页。[13] A. Orhon、M. Siracusa 和 A. Wadhwa，《在Apple芯片上使用Core ML运行Stable Diffusion》，2022。 [在线]。可用地址：null [14] T. Wolf、L. Debut、V. Sanh、J. Chaumond、C. Delangue、A. Moi、P. Cistac、T. Rault、R. Louf、M. Funtowicz 和 J. Brew，《Transformers：最先进的自然语言处理》，载于2020年自然语言处理实证方法会议：系统演示论文集。计算语言学协会，2020，第38–45页。[在线]。可用地址：https://www.aclweb.org/anthology/2020.emnlp-demos.6。[15] A. Radford、J. W. Kim、C. Hallacy、A. Ramesh、G. Goh、S. Agarwal、G. Sastry、A. Askell、P. Mishkin、J. Clark、G. Krueger 和 I. Sutskever，《从自然语言监督中学习可迁移视觉模型》，载于第38届国际机器学习会议（ICML）论文集，2021。 [在线]。可用地址：https://arxiv.org/abs/2103.00020。[16] J. Li、D. Li、C. Xiong 和 S. C. Hoi，《BLIP：通过自举式语言-图像预训练实现统一的视觉语言理解与生成》，载于第39届国际机器学习会议（ICML）论文集，2022。 [在线]。可用地址：https://arxiv.org/abs/2201.12086。[17] J. Wu、L. Liu、Y. Tan、J. Hao 和 X. Zhang，《Promptus：提示词流媒体能否借助Stable Diffusion取代视频流媒体》，arXiv预印本 arXiv:2405.20032，2024。[18] L. Liu、J. Wu、H. Wang、P. Wang、Z. Guo 和 X. Zhang，《Promptmobile：面向低带宽移动视频流媒体的高效Promptus》，arXiv预印本 arXiv:2503.16112，2025。[19] J. Johnson、A. Alahi 和 L. Fei-Fei，《用于实时风格迁移和超分辨率的感知损失》，载于《计算机视觉——ECCV 2016：第14届欧洲会议，荷兰阿姆斯特丹，2016年10月11日至14日，论文集，第二部分14》。施普林格，2016，第694–711页。[20] 《H.265》，2024，https://www.itu.int/rec/T-REC-H.265。[21] C. B. Browne、E. Powley、D. Whitehouse、S. M. Lucas、P. I. Cowling、P. Rohlfshagen、S. Tavener、D. Perez、S. Samothrakis 和 S. Colton，《蒙特卡洛树搜索方法综述》，IEEE计算智能与人工智能游戏汇刊，2012。[22] L. Kocsis 和 C. Szepesv´ari，《基于赌博机的蒙特卡洛规划》，载于欧洲机器学习会议。施普林格，2006，第282–293页。[23] P. Esser、R. Rombach 和 B. Ommer，《驯服Transformers以实现高分辨率图像合成》，载于IEEE/CVF计算机视觉与模式识别会议论文集，2021，第12 873–12 883页。[24] R. Zhang、P. Isola、A. A. Efros、E. Shechtman 和 O. Wang，《深层特征作为感知度量的惊人效用》，载于IEEE计算机视觉与模式识别会议论文集，2018，第586–595页。[25]《ACM多媒体2022大挑战赛：短视频流媒体》，https://www.aitrans.online/MMGC2022/，2022，[在线；访问日期：6月8日-2022]。