arXiv:2602.12641v1 [cs.NI] 13 Feb 2026

# Artic：面向人工智能的实时通信与多模态大语言模型视频助手

Jiangkai Wu、Zhiyuan Ren、Junquan Zhong、Liming Liu、Xinggong Zhang北京大学

## 摘要

人工智能视频助手作为实时通信（RTC）的一种新范式应运而生，其中一方是部署在云端的多模态大语言模型（MLLM）。这使人类与人工智能之间的交互更加直观，类似于与真人聊天。然而，当前实时通信框架与人工智能视频助手之间存在根本性的不匹配，这源于体验质量（QoE）的巨大转变以及更具挑战性的网络环境。对我们的生产原型进行的测量也证实，当前的RTC存在缺陷，会导致延迟峰值和准确率下降。

为应对这些挑战，我们提出了***Artic***，一种面向多模态大语言模型视频助手的人工智能实时通信框架，探索从“人类观看视频”到“人工智能理解视频”的转变。具体而言，Artic提出了：（1）响应能力感知自适应比特率，利用多模态大语言模型准确率的饱和特性主动限制比特率，预留带宽余量以吸收未来波动，从而降低延迟；（2）零开销上下文感知流传输，将有限的比特率分配给对响应最重要的区域，即使在超低码率下也能保持准确率；（3）退化视频理解基准，这是首个评估实时通信导致的视频退化如何影响多模态大语言模型准确率的基准。使用真实世界上行链路轨迹进行的原型实验表明，与现有方法相比，Artic将准确率显著提高了15.12%，并将延迟降低了135.31毫秒。我们将在https://github.com/pku-netvideo/DeViBench发布该基准和代码。

## 1 引言

人工智能视频助手 [1–5、7、19–21] 是实时通信（RTC）的一种新范式。借助多模态大语言模型（MLLMs）[6, 35, 52, 53, 60, 61, 70]，这些系统可直接接收视频和音频输入，实现类似于与真人助手聊天的直观交互 [31]。除了基本的问答之外，现代助手正逐渐发展出复杂的智能体能力（§2.1），例如主动响应 [3, 11, 13]和视觉记忆 [1, 3, 5–7, 16]。然而，多模态大语言模型需要高性能计算资源（例如，8×A100 GPU [48]）来支持实时推理。由于人工智能视频助手越来越多地通过轻量便携设备（例如智能眼镜 [8, 10, 14, 17, 18]）访问，而这些设备无法满足这些需求，多模态大语言模型不可避免地部署在服务器上，并依靠实时通信连接客户端。如图1所示，客户端向服务器传输高比特率的视频和音频，同时接收低比特率的音频和文本响应。

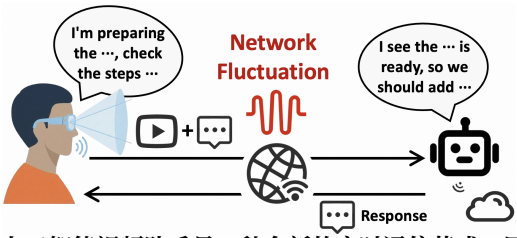



**图1：人工智能视频助手是一种全新的实时通信范式。用户将视频和音频发送给人工智能进行思考，人工智能则反馈音频。**

然而，当前实时通信框架 [30] 与人工智能视频助手之间存在根本性的不匹配，严重降低了用户的体验质量（QoE）。首先，体验质量目标已从人类感知质量（例如VMAF、卡顿率）转变为多模态大语言模型的响应准确率和更严格的时延约束（§2.2）。其次，使用场景已从久坐应用（例如视频会议）演变为“随时随地”的辅助（例如户外导航 [3, 63]），其中移动性和不受限制的用户行为会导致带宽剧烈波动 [33, 40, 47]。第三，流量模式变为上行主导，与传统的下行链路密集型（例如云游戏、远程桌面）或对称型（例如视频通话）服务相比，系统明显更容易受到上行链路容量有限的影响 [33, 36, 37, 39, 42, 51]。至关重要的是，**对我们的生产原型进行的测量证实，传统的面向人类的实时通信在真实环境中表现不佳，导致延迟峰值和准确率下降**（§2.3），打破了用户与“真人”互动的错觉。

能否通过实时通信设计**为人工智能视频助手**始终保证低延迟和稳健的高准确率？实现这一愿景，有助于我们从网络系统的角度探索人工智能研究的“圣杯” *[31]*：让人工智能像真正的人类一样进行交互。

为实现这一目标，我们确定了三个关键挑战：

**首先，当准确率达到饱和后，拥塞控制（CC）反而会适得其反。**在传统实时通信中，只要有可用带宽，拥塞控制算法就会持续提高视频比特率（图2左侧），因为这些算法针对的是

人眼：更高的码率始终能提升感知质量。相比之下，多模态大语言模型的表现截然不同：视频质量达到足够水平（例如图 3 中的 968 千比特每秒）后，进一步提高比特率几乎无法提升准确率。达到饱和后，传统的最大化比特率以填满带宽的策略不仅浪费资源，还存在风险；它会占用带宽余量，显著增加带宽波动时出现高延迟的风险（例如图2中的第500–600视频帧）。

**其次，上下文无关编码效率低下**。带宽不足迫使实时通信以低码率编码视频，这不可避免地导致视频质量下降并引发回答错误（例如图 3(c)）。我们发现，要回答特定问题，多模态大语言模型通常只需要局部区域具有高保真度。例如，在图 3 中，当请求是“请帮我检查一下正在补货的商品”时，只有用户手部附近的商品需要清晰可辨；其他区域的清晰度不会影响准确率。然而，现有的上下文无关编码会不加区分地分配比特，导致关键细节变得模糊。虽然传统的感兴趣区域编码可以优先处理特定区域（例如视频会议中的人脸），但它依赖于预定义的目标。相比之下，多模态大语言模型关注的内容取决于当前的对话上下文，而这具有高度动态性，无法预先确定。

**第三，目前没有基准能够评估实时通信中的视频退化对多模态大语言模型准确率的影响**。传统的视频流基准测试关注人类感知质量（例如 VMAF、卡顿率），而非准确率。尽管一些多模态大语言模型基准会评估流式视频场景中的回答质量（例如 StreamingBench [44]），但其主要目的是测试多模态大语言模型的智能水平；因此，所有输入视频都采用理想的高码率（例如 4000 千比特每秒）。这使得现有问答样本过于简单且偏向高层次，只需粗略理解视频内容便能正确作答。例如，我们在 StreamingBench 上的测试表明，只有 8% 的问答样本对退化敏感（即在高码率下回答正确、低码率下回答错误）。这无法体现现实场景中的情况：涉及丰富细节的查询极易受到质量退化的影响。如图 3 所示，对于“商品上的文字是什么？”这一问题，即使轻微的模糊也会导致回答错误。

为应对这些挑战，我们提出了***Artic***，一个面向人工智能的实时通信系统，服务于多模态大语言模型视频助手，并探索网络需求如何从“人类观看视频”转向“人工智能理解视频”。Artic 从根本上重新设计了实时通信流程，涵盖比特率自适应、视频编码、反馈控制、基准构建和体验质量评估，使网络传输与多模态大语言模型的认知状态相匹配。总的来说，Artic 的系统设计包含三项核心贡献：

• **响应能力感知自适应比特率**。Artic 不再盲目遵循拥塞控制，而是将多模态大语言模型的响应能力纳入比特率自适应。具体而言，Artic 利用多模态大语言模型的置信度反馈，判断当前响应能力是否已达饱和。如果已达饱和，即使带宽充足，Artic 也会主动限制比特率。该策略预留带宽余量以应对网络波动，从而尽可能减少延迟峰值。

• **零开销上下文感知流传输**。不同于上下文无关编码，Artic 提出将有限的比特率分配给当前对话上下文中最重要的区域。具体而言，Artic 指示多模态大语言模型实时反馈对生成响应至关重要的区域。客户端接收到这些区域后，会自适应地调整编码量化参数（QP），从而确保多模态大语言模型即使在超低带宽条件下也能捕捉关键的视觉语义。这种由多模态大语言模型驱动的反馈机制避免了客户端侧重要性识别的开销，同时提供了更原生且准确的上下文感知能力。

• **退化视频理解基准**。Artic 引入了首个评估 RTC 引发的视频退化如何影响多模态大语言模型准确率的基准。Artic 不采用人工标注，而是构建了一个自动化流程，用于生成“退化敏感样本”。借助该流程，Artic 生成了 1,968 个样本，涵盖 6*2 个场景类别，共计 88,680 秒视频。该基准包含测试集和验证集划分，不仅能够评估人工智能视频助手的体验质量，还可用于系统超参数调优以及多模态大语言模型的上下文学习。**我们将开源该基准**。

我们使用真实世界的移动上行链路轨迹对 Artic 进行了原型开发和评估。与现有框架相比，Artic 将准确率显著提高了 15.12%，并将延迟降低了 135.31 毫秒。我们还验证了 Artic 的泛化能力和鲁棒性。此外，我们证明了 Artic 带来的计算、带宽和费用开销极小。

## 2 动机

### 2.1 人工智能视频助手背景

人工智能视频助手迅速兴起（例如，Grok [7],Gemini Live [5], ChatGPT [1], Copilot [2] 和豆包 [3]）。这些系统由多模态大语言模型（多模态大语言模型） [6,35, 52, 53, 60, 61, 70], 提供支持，能够直接接收视频和音频输入，而非文本提示。多模态大语言模型处理这些输入并生成回复，为用户提供反馈。这使得用户能够进行自然交互，其体验可与人类助理相媲美。除了基本的视觉问答外，它们还在发展更复杂的智能体能力：

**主动响应**。不同于早期仅对明确提问作出回应的轮次式交互，现代助手（如 Google Project Astra [11], 豆包 [3], raven-0 [13]）支持

**表 1：服务质量体验目标对比**

| 场景 | 传统实时通信 | 人工智能视频助手 |
|---|---|---|
| QoE 因素 | **保真度**（例如，峰值信噪比、结构相似性、VMAF） | **准确率**（多模态大语言模型回复） |
| | **流畅度**（例如，帧率、卡顿率） | |
| | **一致性**（例如，质量波动） | |
| | **延迟**（允许更多预算） | **延迟**（允许更少预算） |

全双工交互。它们通过分析环境变化和用户意图，主动决定何时回应。例如，如果用户指示“陪我看比赛，进球时喊一声‘进球’”，助手会持续监控视频流，并在事件发生时及时回应。

**视觉记忆**。与早期无状态的交互不同，如今领先的助手（例如 Gemini Live [5], ChatGPT [1],Replika [16]）都具备记忆能力，从而确保多轮对话的一致性。此外，助手还能回忆起先前视频帧中的视觉信息。例如，用户询问“我把手机弄丢了，请告诉我把它落在哪里了”，助手便可以在历史视频帧中找到手机，并推断出手机遗失的位置。**轻量便携客户端**。人工智能助手正从固定式桌面平台转向移动应用（例如 Grok [7], 豆包 [3]）和可穿戴设备（例如智能眼镜 [8, 10, 14, 17, 18]）。这一转变推动部署方向转向轻量便携终端。

这些能力使实时通信成为人工智能视频助手的必需。首先，多模态大语言模型需要高性能计算资源（例如 8× A100 GPU [48]）来支持实时推理。然而，轻量客户端设备无法满足这些计算和功耗需求。因此，多模态大语言模型不可避免地需要部署在云端，由客户端传输用户的视频和音频以供推理。此外，为了支持主动响应能力，系统必须维持持续数分钟至数小时的实时通信会话，而不是传输零散图像或较短片段。**因此，人工智能视频助手为实时通信开辟了一个全新的场景**。

## 2.2 人工智能视频助手与传统实时通信的差异

人工智能视频助手的独特特性为实时通信带来了新的网络需求，与以人类为中心的通信有所不同。下面我们将分析这些变化。

**体验质量从人类感知质量转变为多模态大语言模型的响应准确率和延迟**。在传统实时通信中，体验质量以人类视觉系统为中心。为了满足人眼的需求，实时通信系统会优化保真度（采用诸如峰值信噪比 [68, 69], 结构相似性 [32, 34, 62] 或 VMAF [22] 等指标来确保清晰度）、流畅度（保持较高的帧率 [46, 50, 68, 69] 并尽量减少卡顿 [32, 50, 68, 69]）以及一致性（避免频繁的质量波动 [69]）。然而，在人工智能视频助手中，视频的使用者从人类转变为多模态大语言模型。因此，不再需要严格遵循感知相关指标；主要优化目标转为提高多模态大语言模型的响应准确率。这使实时通信的容忍度提升至语义层面。例如，只要准确率得到保证，模糊等视觉劣化便可以接受（例如，图 3(b)）。

尽管上述方面的体验质量要求有所放宽，但延迟仍然至关重要。由于现代助手 [3, 11,13] 旨在目标事件一发生便主动做出响应，因此帧传输的任何延迟都会直接转化为助手响应的延迟。此外，人工智能助手对延迟的约束条件甚至更加严格。为确保流畅的交互体验，响应时延需要保持在 300 ms 以下 [43]。在传统实时通信中，另一端的人类同伴可以即时响应，这意味着实时通信可使用的延迟预算相对充足。然而，多模态大语言模型以耗时的自回归方式生成响应。即使仅使用音频输入，计算延迟也至少达到 232 ms [41]。若要将总响应时延控制在 300 ms 以内，留给实时通信的预算最多只有 68 ms，这很难得到保证。因此，除了准确率之外，尽可能降低帧延迟也是另一项主要目标。我们在表 1 中总结了这些服务质量体验因素。

**高移动性和用户行为会加剧带宽波动**。人工智能视频助手（例如智能眼镜 [8, 10, 14, 17, 18]）的便携性要求其能够随时随地使用，这给实时通信带来了显著的稳定性挑战。物理移动本身就会降低链路质量。无论使用 Wi-Fi 还是 5G，持续移动都会导致频繁的信号衰减和切换 [33, 40, 47]，从而引发剧烈的带宽波动，远超静态连接中的波动幅度。

更关键的是，用户行为不再受限。例如，在传统视频会议中，用户通常会配合，主动寻找连接稳定的地点，并保持静止，以确保网络质量。相比之下，人工智能视频助手是在“真实环境”中使用的。用户期望在穿行于复杂环境（例如进入电梯或地铁）时也能实现无缝交互，这会使实时通信会话遭遇剧烈的带宽骤降和中断，而这类情况在静态环境中极少发生。

**上行链路比下行链路更为关键**。传统实时通信场景通常呈现下行链路流量较大或上下行对称的流量模式。在一对一视频通话中，由于每位用户既是视频发送方也是接收方，因此流量是对称的。在视频会议中，大多数参与者（发言者除外）面临更大的下行链路压力：他们需要接收多路视频流，但只发送一路。类似地，在云游戏或远程桌面等交互式视频场景中，

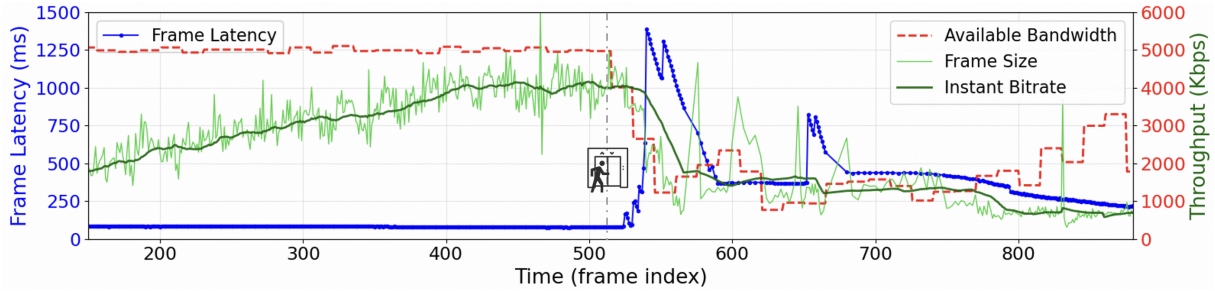



**图2：时延测量研究。当带宽充足时，在拥塞控制的驱动下，视频比特率会持续上升。然而，受移动性和用户行为的影响，带宽会骤然下降（例如视频帧 525 处）。比特率降低滞后导致网络拥塞，进而引发严重的延迟峰值（例如 1,389 毫秒）。**

客户端发送低比特率的控制信号，同时接收高质量视频，因此流量明显以下行链路为主。这些流量模式与当前的网络基础设施（例如 5G）十分契合，因为其下行链路带宽的设计值远高于上行链路 [33, 36, 37, 39, 42, 51]，例如高出 4.8 至 6.5 倍 [33]。

相比之下，人工智能视频助手依赖单向上行链路传输。客户端仅作为视频发送方，持续向多模态大语言模型传输高比特率视频，而多模态大语言模型仅通过低比特率音频或文本返回反馈。这种流量模式与现代网络的非对称设计背道而驰。因此，瓶颈完全转移至上行链路，与传统实时通信场景相比，这给体验质量带来了更大的挑战。

## 2.3 测量研究：真实网络状况对助手体验质量的影响

为实证展示真实网络限制如何损害 AI助手体验，我们在本节开展了一项测量研究。接下来，我们将先介绍测量设置，再呈现分析结果。

为捕捉第 §2.2 节中的网络挑战，我们收集了反映真实流量模式的真实网络跟踪数据。我们在商用 5G 网络上使用一部 iPhone 17 搭建了测试平台，并通过 iperf [25] 向一台生产环境 Linux服务器生成上行链路用户数据报协议流量。我们记录接收吞吐量，以估算可用带宽。为严格隔离移动性与其他因素的影响，我们采用了两种具体策略。首先，我们在凌晨 2:45 进行测试，以尽量减少交叉流量竞争。其次，我们将发送速率限制为 5 Mbps——这是链路在静止状态下可以轻松维持的速率——从而消除互联网侧瓶颈的影响。这确保了所观察到的任何波动都归因于物理环境变化，而非外部网络拥塞。至于用户行为，我们选择了乘坐电梯这一日常生活中十分常见的场景。

我们按照 OpenAI 的 AI视频通话标准 [26]实现了一个人工智能视频助手原型。前端是一个基于 Web 的 AI助手，作为实时通信发送端，用于采集用户的音频和视频。后端运行在生产环境中的 Linux服务器上，作为实时通信接收端，并调用多模态大语言模型生成响应。具体而言，我们使用商用 Agora SDK [27] 实现实时通信，并使用 Google Gemini Live API [5] 调用多模态大语言模型。为控制网络条件，我们将发送端的流量经由一条自定义管道传输，该管道会动态限制带宽，以重放采集到的轨迹。

我们在图2中展示了带宽轨迹、视频比特率、视频帧大小和帧延迟，并在图 3中展示了准确率示例。基于这些结果，我们得出以下观察结论：

**严峻的网络状况是常态。** 如图2所示，当用户处于静止状态时，可用带宽持续达到5 Mbps上限，证实在理想条件下链路稳定且充足。然而，在第525视频帧，当用户进入电梯时，可用带宽迅速降至1.23 Mbps。这表明，对于“移动中”的*AI*视频助手而言，网络质量由日常用户行为和物理环境决定，并在1.5秒内从优良变为较差。

**剧烈的带宽波动会导致延迟峰值。**在实时通信系统中，比特率通常由拥塞控制算法（例如，GCC [30], BBR [29]）进行自适应调节。当延迟或丢包没有增加时，拥塞控制算法会提高比特率以探测可用带宽。然而，当上述带宽骤降发生时，拥塞控制算法会出现自适应滞后（约*1.17*秒）。因此，比特率超过带宽会导致严重的数据包积压，从而使延迟高达1,389 毫秒。

**低带宽会降低准确率。** 面对高延迟和丢包，拥塞控制算法最终会大幅降低比特率。然而，如图 3 所示，低比特率会导致严重模糊，直接造成多模态大语言模型给出错误回答。为验证这些错误源于视频退化，而非模型智能有限，我们将 Gemini Live 模型替换为 Gemini 3 Pro [24] 模型，并让其处理相同的退化帧。




**图 3：准确率测量研究。(a)(b) 准确率呈现饱和。随着比特率增加，多模态大语言模型始终保持准确。(c) 低码率会导致多模态大语言模型出错。(d) 即使使用更强大的 Pro模型，错误仍然存在，**这表明错误源于视频退化，而非模型能力。

Pro模型仍然出现相同的错误，这证实了 准确率下降是由实时通信导致的视觉伪影造成的，而非模型能力所致。

## 2.4 关键洞见与潜在收益

基于上述测量，我们的目标是从实时通信的角度优化人工智能助手体验质量，尤其是在具有挑战性的网络条件下。本节提出源自人类与多模态大语言模型感知差异的三项关键洞见，并随后讨论潜在收益：

**多模态大语言模型准确率的饱和使得在带宽波动期间能够降低延迟。**传统实时通信算法在带宽充足时会激进地提高视频比特率，这是针对人眼优化的，因为更高的码率会持续提升感知质量（例如，支持 8K@120FPS）。然而，正如我们的测量（图 3）所示，多模态大语言模型表现出根本不同的行为：一旦视频质量达到足够水平（例如 968 千比特每秒），模型的回答始终保持正确。在这一饱和区间内，进一步提高比特率不会带来准确率收益。相反，它会占用带宽余量，从而在带宽波动时显著增加高延迟风险。

我们不再盲目提高比特率，而是将模型的响应能力纳入比特率决策环。如果多模态大语言模型在当前比特率下能够正确响应，即使带宽充足，我们也会停止提高比特率（例如图2中的第0—500帧）。这一策略有意保持比特率的“最大余量”，预留余量以应对未来的带宽波动（如第525帧所示）。**上下文感知编码可在低带宽下提升准确率。**在视频编码中，比特率与失真之间始终存在权衡。低比特率会导致模糊和块效应等画质退化。在传统实时通信中，一种可行的解决方案是为人眼关注区域分配更多比特率，例如视频会议中的人脸，同时牺牲其他区域，以保留关键区域。

我们发现，这一概念对于人工智能视频助手同样至关重要，但不同区域的重要性取决于当前的对话上下文，而非人类的视线。例如，在图 3 中，针对“请帮我检查一下要补货的商品”这一提问，多模态大语言模型只需关注用户手边的商品区域，其余部分无关紧要。因此，视频编码不应与上下文无关，而应具备上下文感知能力。应为对响应重要的区域分配更多比特率，为无关区域分配更少比特率。

**第一个基准评估视频流媒体质量对多模态大语言模型准确率的影响。**根据 §2.2，人工智能视频助手的体验质量指标已从感知转向准确率。现有的视频流基准测试并不适用，因为它们侧重于感知质量，且不涉及响应准确率。在多模态大语言模型领域，有一些针对流式视频理解任务的基准 [56, 64]，例如 StreamingBench [44]。在这些基准中，每个视频都包含若干问答（QA）样本，用于评估响应准确率。然而，这些基准旨在测试多模态大语言模型的智能水平，因此所有输入视频都采用理想的高比特率（例如 4000 千比特每秒）。

为了评估视频流媒体质量如何影响准确率，我们将 StreamingBench 中的视频转码为 200 千比特每秒。我们使用原始问答样本测试这些低码率视频。结果表明，在低码率下回答错误、而在高码率下回答正确的问答样本仅占 **8%**。这是因为 StreamingBench 中的问答样本过于简单且偏向高层次，只需粗略了解视频内容便能正确作答。然而，在现实场景中，往往有许多细节丰富的问题，对视频质量非常敏感。例如，在图 3 中，当问题是“产品上的文字是什么？”时，即使轻微模糊也会导致无法正确作答。因此，有必要建立一个更具挑战性的基准，以反映视频退化对多模态大语言模型准确率的现实影响。

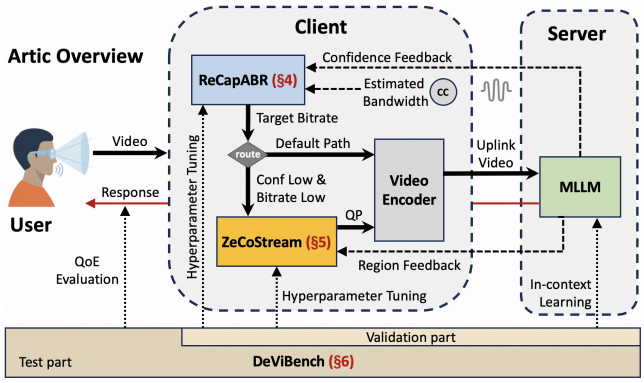



**图 4：Artic 概览**

## 3 概述

基于上述见解，我们提出了***Artic***，一个面向视频助手的人工智能实时通信系统。如图 4 所示，Artic 包含三个核心模块：响应能力感知的自适应比特率（§4）、零开销上下文感知流传输（§5）以及退化视频理解基准（§6）。其工作流程如下：

***ReCapABR***（***Re***sponse ***Cap***ability-aware ***A***daptive ***B***it***r***ate，响应能力感知自适应比特率）通过衔接网络拥塞控制与多模态大语言模型的认知状态来确定视频比特率。它利用多模态大语言模型反馈的置信度分数，评估模型当前的响应能力是否已达到饱和。不同于传统的 ABR，ReCapABR 引入了一种保持机制：如果拥塞控制建议提高比特率（由于可用带宽增加），但多模态大语言模型表明其在当前视频质量下已经表现良好，ReCapABR 将覆盖拥塞控制的决策，维持当前比特率。这一策略预留了带宽余量，以吸收未来的波动。在其他情况下——例如多模态大语言模型难以准确回答问题，或带宽下降时——ReCapABR 遵循拥塞控制设定的目标比特率，在避免拥塞的同时最大化视频质量。

***ZeCoStream***（零开销的上下文感知流式传输）可优化空间比特率分配。在网络状况正常时，ReCapABR 确定的比特率会输入标准视频编码器。然而，当比特率降至多模态大语言模型难以准确作答的临界水平，且带宽不足以支持更高比特率时，便会触发 ZeCoStream。根据多模态大语言模型的反馈，ZeCoStream 能够识别“对响应至关重要”的区域，且不会产生额外开销。随后，它会指示编码器执行 QP自适应编码，使有限的比特率几乎全部分配给这些区域。这样，即使带宽约束极低，多模态大语言模型仍能捕捉关键视觉语义。ZeCoStream 默认处于禁用状态，因为降低与响应无关区域的质量会导致记忆不完整，可能损害后续对话上下文。因此，只有在上述必要情况下才会触发该功能。

***DeViBench***（***De***graded ***Vi***deo Understanding ***Bench***mark，分级视频理解基准）作为评估基础。我们构建 DeViBench，以定量评估实时通信引起的性能下降对多模态大语言模型准确率的影响。它包括测试集和验证集。测试集用于评估 Artic 系统的体验质量，而验证集则用于调优 ReCapABR 和 ZeCoStream 的超参数，以及进行多模态大语言模型的上下文学习。

## 4 响应能力感知自适应比特率

本节提出 **ReCapABR**。首先，我们分析影响不同比特率下多模态大语言模型准确率的因素。其次，我们探讨如何实现响应能力感知。第三，我们介绍基于置信度的比特率控制。

### 4.1 不同码率下哪些因素会影响多模态大语言模型准确率?

与人类感知不同，多模态大语言模型的理解能力不会随着比特率降低而逐渐下降，而是表现出“饱和”现象。我们确定了低码率下决定多模态大语言模型准确率的三个关键因素：

**先验知识与分布外（OOD）内容**。多模态大语言模型经过海量数据集的预训练，内化了丰富的知识。当视觉输入与这些先验知识相符时，即使输入质量下降，模型仍能推断出正确结果。例如，观看世界杯比赛时，如果多模态大语言模型识别出特定的球衣颜色和号码，即使由于低码率造成的模糊导致面部完全无法辨认，也能根据其内部知识推断出球员身份（例如“梅西”）。相反，对于分布外内容，模型无法依赖先验知识，必须具备较高的视觉保真度才能提取特征。

**已见内容与未见内容的记忆**。 如§2.1所述，当前多模态大语言模型会在一段时间内保持视觉记忆。如果内容此前已被清晰捕捉，模型便能从记忆中“回忆”其属性，从而降低对当前视频帧质量的依赖。例如，若用户举起一个药瓶，且摄像头捕捉到标签的清晰视频帧，多模态大语言模型便会存储该信息。数分钟后，即使视频因网络中断而变得模糊，多模态大语言模型仍可通过从记忆中检索来回答有关剂量的问题；而新出现的内容则需要即时的高质量画面才能被识别。

**低信息密度与高信息密度**。 视频帧中的语义信息密度决定了压缩伪影的影响。语义信息稀疏的场景具有较强的韧性，而信息密集的场景则很脆弱。例如，

即使画面模糊，展示户外草坪和天空的场景在语义上仍保持不变，因为信息损失并不会改变对场景的基本理解。相比之下，文本密集型文档或充满代码的屏幕依赖高频细节。低码率下，视频编解码器会最先平滑掉这些细节，直接导致识别错误或产生幻觉。

## 4.2 响应能力感知

要实现响应能力感知，最直接的方法是构建一个数学模型，将比特率及上述影响因素映射到多模态大语言模型的准确率，这与传统的体验质量建模类似 [38, 57]。然而，在人工智能视频助手的场景中，显式建立这种映射模型面临着重大挑战：

**泛化能力**。 多模态大语言模型之间存在极大的异质性。不同的模型系列（例如 ChatGPT 与 Gemini）和参数规模（例如 7B 与 70B）对视频退化的敏感度迥异。在一个多模态大语言模型上训练的回归模型不太可能有效迁移到另一个模型。此外，商业多模态大语言模型频繁更新，会迅速使离线训练的预测器失效。

**复杂性**。 如上所述，视觉质量与响应准确率之间的关系高度非线性且依赖于内容（例如，取决于先验知识或记忆），因此轻量级拟合函数难以奏效。

为应对这些挑战，我们不依赖外部建模，而是利用多模态大语言模型的内在能力。由于云端多模态大语言模型已经具备强大的视频理解与推理能力，一种可行的方法是让它直接评估并反馈其在当前码率下的响应能力。具体而言，我们采用上下文学习建立实时反馈闭环。通过设计专门的系统提示词，我们指示多模态大语言模型评估当前接收的视频是否信息充分。随后，多模态大语言模型会针对当前时间戳输出一个响应置信度分数 $C_t \in [0,1]\, t$，并将其反馈给客户端。

## 4.3 基于置信度的比特率控制

基于实时置信度反馈 $C_t$，我们设计了一种比特率控制算法，使比特率维持在刚好足以让多模态大语言模型作出响应的水平，从而预留带宽余量以应对带宽波动。

我们首先量化当前性能与目标之间的差距。设 $\tau$ 为置信度阈值（我们根据验证集 §6.2 设置 $\tau = 0.8$ ），它表示多模态大语言模型正确响应所需的置信度水平。我们将归一化置信度差定义为 $\delta_t = (\tau - C_t)/\tau$。正的 $\delta_t$ 表示多模态大语言模型难以识别视频细节，因此需要更高质量；而负的 $\delta_t$ 则表明多模态大语言模型在当前比特率下已经能够很好地响应

为了将这一差距映射到控制逻辑，我们使用敏感度参数 $\gamma$ 计算权重 $w_t$：

$$w_t = \delta_t \cdot |\delta_t|^{\gamma-1} \tag{1}$$

我们根据验证集 §6.2 设置 $\gamma = 2$ 。这种二次缩放使算法能够在置信度显著偏低时迅速采取强力调整，以快速恢复准确率；当 $C_t$ 接近 $\tau$时，则进行细粒度调整。

最后，我们使用 $w_t$ 更新比特率。令 $\hat{Bt}$ 表示拥塞控制算法得出的估计带宽。下一步的比特率 $R_{t+1}$ 计算如下：

$$R_{t+1} = \min\left(\hat{B}_t, R_t + w_t \cdot \left(\hat{B}_t - R_t\right)\right) \tag{2}$$

ˆ 若发生拥塞（$B_t < R_t$），比特率会立即被 $B_t$ˆ ˆ限制。相反，当带宽充足（$B_t \geq R_t$）时，算法会根据 $w_t$进行调整。若 $C_t > \tau$， $w_t$会变为负值，导致比特率主动降低以节省带宽。当 $C_t \approx \tau$时， $w_t$ 会趋近于零，从而使比特率保持在稳定的平衡状态。这一机制形成了一种“按需”传输模式：仅在理解内容有必要时才消耗带宽，并为带宽波动留出安全裕度。

# 5 零开销上下文感知流传输

本节提出 **ZeCoStream**。首先，我们介绍如何以零开销识别对响应重要的区域。其次，我们解决反馈延迟导致的区域过期问题。最后，我们提出一种上下文感知的码率分配策略。

## 5.1 如何以零开销实现上下文感知?

要识别对响应重要的区域，一种直接的方法是在客户端部署轻量级模型（例如 CLIP [54]），以关联对话上下文与不同的视频区域。然而，如 §2.1 所述，人工智能视频助手的客户端通常面临计算能力和电池续航时间有限的问题（例如智能眼镜 [8, 10, 14, 18]），因此难以支持本地模型推理。尽管现有方案提出将计算任务从可穿戴设备卸载到配对智能手机上，但在手机上进行设备端推理仍会带来不可忽视的计算延迟 [54]。

如何在零开销的情况下实现上下文感知？我们观察到，大多数最先进的多模态大语言模型都天然具备实时定位能力 [3, 5, 52, 65,66]。如图5所示，多模态大语言模型可以根据输入提示准确定位相关区域。例如，当对话上下文为“留意右侧。那辆卡车要左转了吗？”时，右侧的卡车就能被精准定位。除了单个物体外，这种

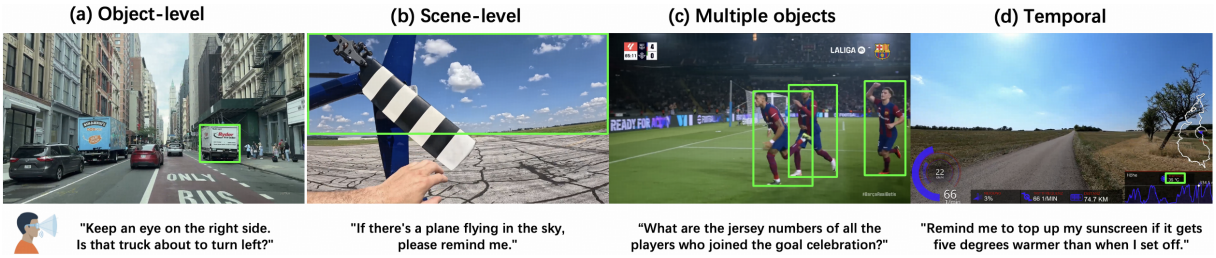



**图5：如何识别对响应重要的区域？现代多模态大语言模型天然具备实时定位能力，能够根据对话上下文准确定位相关区域（绿色框）。除单个物体外，这一能力还可扩展至多个物体、场景级上下文和时间上下文。**

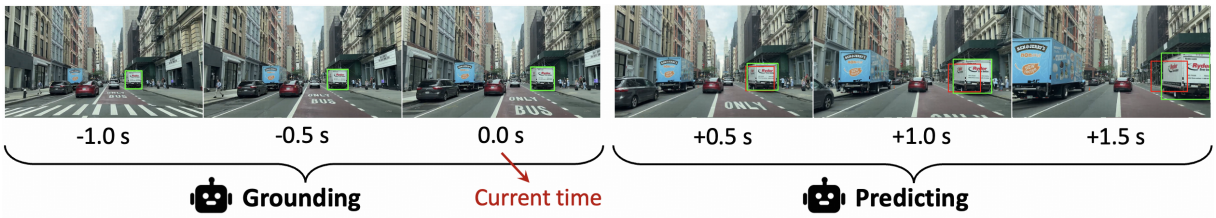



**图6：为抵消反馈延迟，我们要求多模态大语言模型不仅定位当前的重要区域（绿色框），还要预测其在短时间范围内的未来轨迹（红色框）。详情请参见§5.2。**

能力还适用于多个物体、场景级上下文和时间上下文。因此，与 ReCapABR 类似，我们利用服务器端多模态大语言模型，实时向客户端反馈对响应重要的区域的边界框。

## 5.2 通过先定位后预测处理反馈延迟

然而，反馈区域无法直接用于视频编码。这是因为这些区域本质上已经过时，对应的是至少在一个往返时间加上 MLLM推理时间之前捕获的视频帧。虽然对于静态物体（例如停放的汽车）而言，这种延迟可以忽略不计，但对于动态物体而言却至关重要。例如，移动中的车辆可能已经驶出过时的边界框，从而使反馈对于当前视频帧失去作用。

为了解决这一问题，我们要求多模态大语言模型不仅要定位当前的重要区域，还要预测其短时间范围内的未来轨迹。如图 6 所示，仅依据过去 1.5 秒的视频帧，多模态大语言模型就能准确预测接下来 1.5 秒内该区域的边界框。因此，我们将多个未来时间戳对应的一系列预测边界框发送回客户端。随后，客户端将当前时间戳与相应的预测结果相匹配，以指导视频编码。一个关键问题仍待解答：预测需要覆盖多长的延迟？我们在一款商用人工智能视频助手（豆包应用 [3]）上进行了测量，发现从捕获视频帧到收到预测结果的总延迟为 1.20 秒至 1.52 秒。由于图 6 展示了多模态大语言模型在 1.5 秒的时间窗口内仍能保持较高的预测准确率，因此我们的方法能够有效补偿反馈延迟。

## 5.3 上下文感知的码率分配

根据多模态大语言模型反馈的预测边界框，客户端会根据不同视频区域对响应的重要性，为其分配有限的比特率。我们将每个视频帧（$W \times H$）划分为由不重叠图像块组成的网格（例如，64×64 像素）。令$\mathrm{K}_t$表示视频帧 $t$的预测边界框集合。我们计算位置（$i, j$）处图像块的上下文重要性分数$\rho_{i,j} \in [0,1]$：

$$\rho_{i,j} = \max\left(0, 1 - \frac{d_{i,j}}{\mu \cdot \sqrt{W^2 + H^2}}\right) \tag{3}$$

其中，$d_{i,j}$ 表示图像块中心到$\mathrm{K}_t$中的任意框最近边界的欧几里得距离（如果图像块位于框内，则为$d_{i,j} = 0$）。我们将该距离除以视频帧对角线长度，并乘以系数 $\mu = 0.5$进行归一化。这表示当距离超过对角线长度的一半时，重要性会衰减至零。

随后，我们将此分数映射为视频编码的量化参数（QP）。QP 控制压缩级别：较大的值会降低比特率，但会引入模糊；较小的值则能保留细节。令 $Q_{min}$ 和 $Q_{max}$分别表示 QP 范围的下限和上限（例如，$Q_{min} = 20, Q_{max} = 51$）。我们采用非线性映射：

$$Q_{i,j} = Q_{\min} + (Q_{\max} - Q_{\min}) \cdot (1 - \rho_{i,j})^2 \tag{4}$$

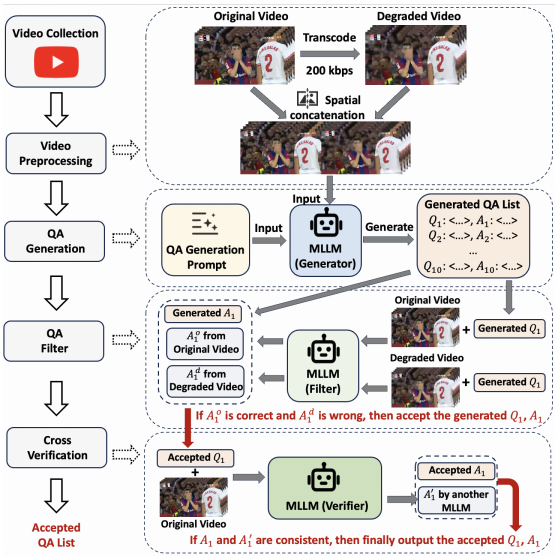



**图 7：DeViBench 自动问答样本构建流程。详情请参见 §6.1。**

在此公式中，重要区域 ($\rho_{i,j} = 1$) 内的图像块会被赋予 $Q_{min}$，以确保最高保真度。随着距离增加，QP 会以二次方式逐渐升至$Q_{max}$。这会促使编码器对不相关区域进行激进压缩，确保有限的码率预算集中用于对多模态大语言模型推理至关重要的内容。

## 6 退化视频理解基准

本节提出**DeViBench**。我们首先详细介绍自动问答构建流程，然后总结该基准。

### 6.1 自动问答构建流程

如§2.4所述，我们需要构建对视频质量敏感的问答样本。为此，最直接的方法是聘请志愿者针对退化视频提出棘手的问题。然而，这种方法成本过高且效率低下，阻碍了数据集的扩展。**因此，我们提出了一个问题：能否自动且低成本地构建问答样本?**

重新审视人工智能视频助手的背景，多模态大语言模型已经能够理解视频并给出回答。因此，我们利用多模态大语言模型取代人类志愿者，开发自动问答样本构建流程。如图7所示，该流程包括5个步骤：

**视频收集**。 我们首先收集用于提问的视频。为与现有多模态大语言模型基准的领域和规模保持一致 [44], , 我们直接使用其中的视频（舍弃问答内容）。

**视频预处理**。为了让多模态大语言模型理解低码率造成的质量下降，我们使用 x265 将原始视频转码为低码率版本（200 千比特每秒）使用 ffmpeg 版本 N-118035-gc1e3d55f99。将低码率视频与原始视频水平拼接成一个视频。随后，将拼接后的视频输入多模态大语言模型，以进行理解和问答生成。

**问答生成**。 为了让多模态大语言模型基于拼接后的视频生成问答样本，我们精心设计了提示词，其中包含角色设定、上下文、核心任务、执行步骤、约束条件和输出格式方面的指导，详见附录（图16）。该提示词确保多模态大语言模型能够识别质量差异，并生成质量敏感型问答样本。最初，我们尝试生成选择题，但发现这类题目本质上挑战性较低。一方面，提示词中包含的选项为多模态大语言模型提供了隐含提示；另一方面，即使视频质量严重下降，多模态大语言模型仍然可以从四个候选选项中猜出答案，从而保证至少25%的基线准确率。因此，我们转而生成自由作答题。我们采用通义千问3-VL-plus [12] 作为生成器。

**问答筛选**。 生成的问答对将经过筛选。我们分别将原始视频和低码率视频输入多模态大语言模型，并使用生成的问答对进行提问。如果原始视频的回答正确，而低码率视频的回答错误，我们就接受该问答对。请注意，由于问题采用自由回答形式，因此不适合通过精确字符串匹配来判断回答是否正确。我们改用另一个多模态大语言模型作为评判器，评估回答与答案之间的语义一致性。实际操作中，使用 Qwen2.5 Omni [60] 作为筛选器，使用 Qwen-Flash [49] 作为评判器。在这一步中，**25.25%** 的问答对可以被接受。

**交叉验证**。 由于生成器生成的答案也可能有误，因此无法通过上述测试将这类答案筛除。因此，我们使用另一个多模态大语言模型进行交叉验证。我们将上述已接受的问题输入另一个多模态大语言模型，如果新答案与上述已接受的答案一致，就最终批准该问答对。在我们的实验中，采用 GLM-4.6V [65] 作为验证器，**89.37%** 的已接受问答对通过了交叉验证。综合以上所有验证，最终生成的问答对中有**22.57%**有效。

### 6.2 基准测试摘要

最后，我们生成了1,968个问答样本，详情汇总于表2，包括问答类型、总时长、总花费和总耗时。我们还分析了不同问答类型的分布，如图8所示。从类别来看，文本丰富型理解（81.86%）、属性感知（11.53%）、物体感知（3.56%）、计数（2.54%）、动作感知（0.36%）和空间理解（0.15%）各占一定比例。从时间依赖性来看，

**表2：基准测试摘要**

| | |
|---|---|
| 问答样本数量 | 1,968 |
| 问答样本类型 | 6*2 |
| 总时长（秒） | 88,680 |
| 总花费（美元） | 48.06 |
| 总耗时（秒） | 102,704 |

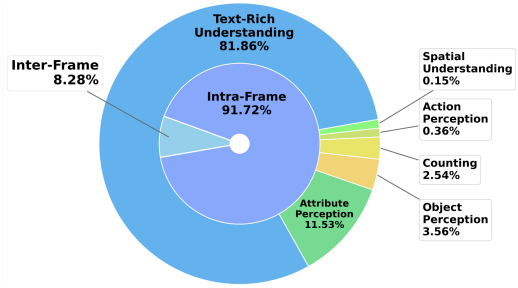



**图8：我们生成的问答样本的分布。外环：问答类别。内环：回答该问题是否需要跨视频帧信息。**

8.28%的问题需要帧间信息（多个视频帧）才能作答，而91.72%的问题仅凭帧内信息（单个视频帧）即可作答。这种不均衡的数据分布是合理的，因为不同问题对视频退化的敏感程度各不相同。这种差异不仅受问题类型影响，也与其对时间或空间信息的依赖有关。

为确认这些由多模态大语言模型生成的问答样本是否可用，我们随机抽取了100个问答样本进行人工回答检查。其中，96%的生成问题人类可以回答，93%的生成答案是正确的。这也证明了我们的自动问答构建流程的有效性。这100个抽查的问答样本被指定为验证集，其余样本构成测试集。验证集用于ReCapABR的超参数调优（例如，$\tau$，$\gamma$）和ZeCoStream的超参数调优（例如，$\mu$），以及多模态大语言模型反馈上下文学习。测试集用于评估（§7.1）。

# 7 评估

## 7.1 实验设置

我们使用 C++ 和 Python 构建了 Artic 原型。本文将介绍实现细节和实验设置。

**基线。** Artic 主要与传统的以人类为中心的 WebRTC 进行比较，后者采用标准视频编解码器（例如 H.265 [59]）和传统的拥塞控制算法（例如 GCC [30], BBR [29]）。我们还设置了几个简单基线，以分离各组件的贡献，包括 WebRTC+ZeCoStream、WebRTC+ReCapABR。

**指标。** 人工智能视频助手的服务质量体验目标包括响应准确率和延迟（表 1，§2.2）。对于准确率，我们计算在 DeViBench（§6）上正确回答的样本数占样本总数的比例。请注意，用于超参数调优的验证样本不纳入计算。延迟指从客户端编码器到多模态大语言模型端解码器所需的时间，称为帧延迟。帧延迟直接影响多模态大语言模型的端到端响应时延。我们不采用完整的端到端延迟，因为它主要由多模态大语言模型推理时间决定，而这与实时通信设计无关。

**测试视频。** 我们使用 DeViBench 中的视频进行评估，其规模汇总于表2。为实现细粒度的量化参数控制，我们采用 Kvazaar [55] 实现的 H.265 编码器，用于我们的方法（Ze-CoStream）和基线（标准编码）。我们将比特率、分辨率和帧率配置与行业设置 [23]保持一致。解码时，我们的方法和基线均通过 decord 0.6.0 使用 x265，并采用相同的解码参数以确保一致性。

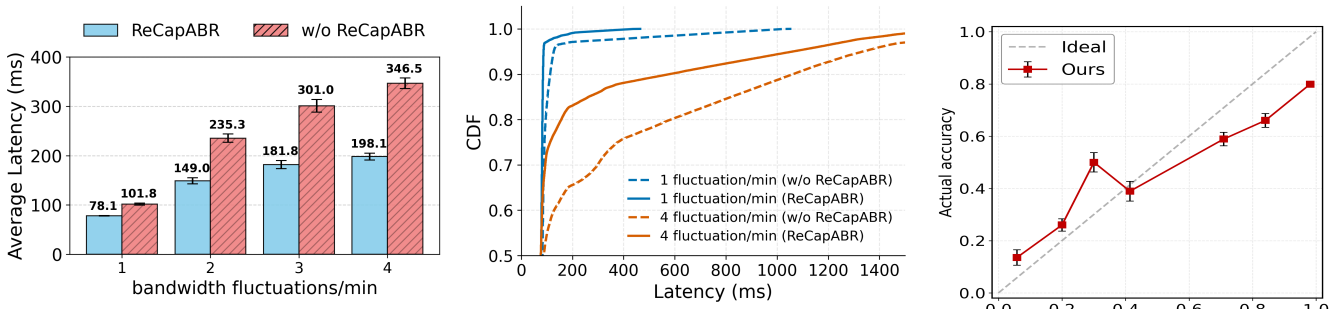
**测试多模态大语言模型。** 我们使用 GLM-4.6V-Flash（9B） [66]评估准确率。为避免循环论证，该模型不同于用于构建 DeViBench（§6）的筛选模型（Qwen2.5-Omni [60]）。测试前，代码和模型均已冻结。我们遵循默认设置（包括相同的随机种子、系统提示词和官方推荐配置），未针对问答测试集进行任何调优（§6.2）。**网络轨迹。** 为匹配人工智能视频助手的流量模式，我们使用一个 5G上行带宽轨迹 [37],数据集以及自行采集的轨迹（§2.3）。为重现用户移动性和行为导致的带宽波动，我们选取驾驶和步行情景中的轨迹，并筛除带宽波动显著的片段。我们循环播放这些轨迹，使其时长与测试视频相匹配。

**网络模拟。** 基于 Razor [15], ，我们开发了一种基于 WebRTC 的单向视频传输系统，同时支持 GCC 和 BBR。我们使用 Mahimahi [9]，通过重放上述轨迹来动态限制发送端的上行链路带宽。我们将队列长度设为 60 个数据包，并采用尾丢弃策略。

## 7.2 ReCapABR 带来的延迟收益

本节展示 ReCapABR 带来的延迟收益。我们在不同的带宽波动频率下，将 ReCapABR 与 WebRTC（GCC）进行比较。我们在若干行业标准带宽等级之间随机切换 [23]，并将波动频率定义为每分钟切换（波动）的次数。图 9a 和图 9b 分别展示了平均延迟和累积分布函数（CDF）。

**ReCapABR 能在带宽波动时有效降低延迟。** 首先，ReCapABR 始终能降低




**(a) 平均延迟** **(b) 延迟分布**

**图9：ReCapABR 显著降低了带宽波动情况下的延迟，尤其是在波动变得更加频繁时。详情见 §7.2。图10：ReCapABR 对置信度误差的鲁棒性。**

各种带宽波动频率下的平均延迟。其次，带宽波动越频繁，降低延迟的效果越显著。图9a显示，每分钟波动1次时，与基线相比，ReCapABR 可将延迟降低23.7毫秒；每分钟波动4次时，降幅扩大至148.4毫秒。这是因为在剧烈波动下，遵循拥塞控制的基线更容易出现过度传输。相比之下，ReCapABR 通过对拥塞控制估算的带宽进行限幅，预留了更多余量来应对带宽骤降。第三，ReCapABR 显著提高了低时延帧的占比。例如，如图9b所示，在每分钟波动4次时，基线中延迟低于200毫秒的视频帧仅占65.5%，而 ReCapABR 达到了82.5%。这一提升同样归功于 ReCapABR 的余量机制。**对置信度误差的鲁棒性。** ReCapABR 依赖多模态大语言模型反馈中的置信度分数。理想情况下，这些分数能够表明模型在当前比特率下能否给出正确回答。图10展示了置信度分数与实际准确率之间的关系。结果表明，置信度分数总体上与准确率相符。因此，采用保守的限幅策略后，ReCapABR 对置信度误差仍具有鲁棒性。

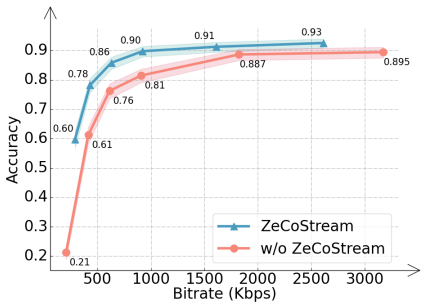



**图11：ZeCoStream 在带宽受限条件下显著提高准确率，并降低实现高准确率所需的带宽。**

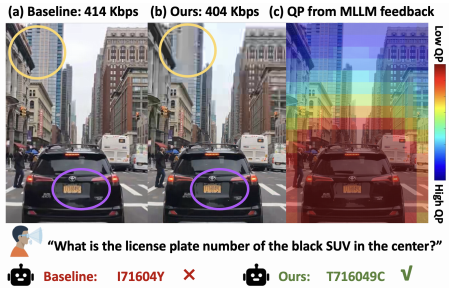
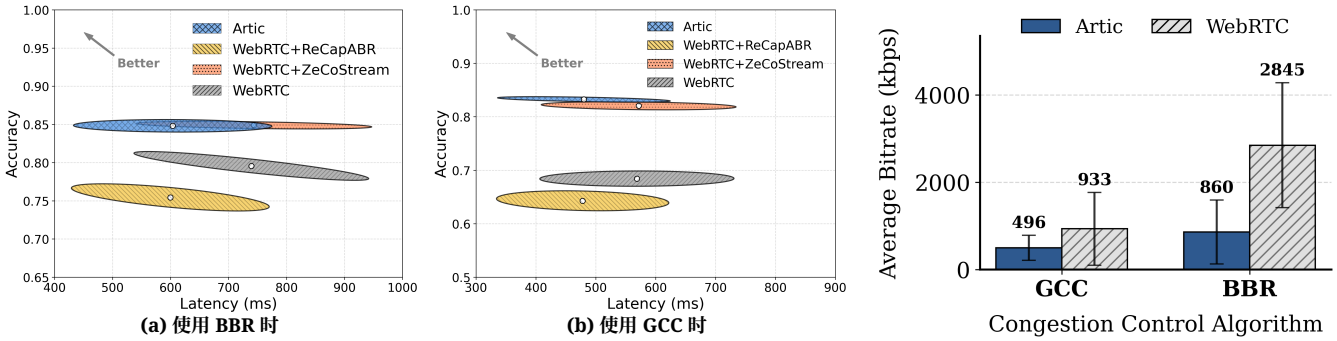



**图12：(a) 使用默认设置编码。(b) 使用 ZeCoStream 编码。(c) 多模态大语言模型反馈的量化参数。结果显示，ZeCoStream 会为重要区域（例如紫色圆圈）分配更多比特，为无关区域（例如黄色圆圈）分配更少比特。**

## 7.3 ZeCoStream 带来的准确率提升

本节证明 ZeCoStream 带来的准确率提升。具体而言，我们比较 ZeCoStream 与不考虑上下文的标准编码在各种码率（行业标准设置 [23]）下的准确率。结果如图11所示。**ZeCoStream 大幅提升了带宽受限条件下的准确率。**首先，在低带宽场景中，ZeCoStream 能有效保持准确率。例如，在 290 千比特每秒的低码率下，基线准确率仅为 0.39，而 ZeCoStream 可将其维持在 0.60。其次，ZeCoStream 显著降低了实现高准确率所需的带宽。例如，要达到 0.9 的准确率，基线需要超过3171.37千比特每秒的比特率，而ZeCoStream仅以907.60千比特每秒即可实现。这些增益归功于ZeCoStream能够将有限的码率分配给对响应最为关键的区域，确保每一比特都直接贡献于准确率。图12可视化了输入多模态大语言模型的两个采样视频帧。即使在相近的码率下（414千比特每秒 vs. 404千比特每秒），ZeCoStream仍在对响应重要的区域（例如紫色圆圈）保持更高的保真度，同时允许对响应无关的区域（例如黄色圆圈）更加模糊，从而提高多模态大语言模型准确率。




(a) 使用 BBR 时 (b) 使用 GCC 时

**图13：在真实网络跟踪数据下的整体性能提升。图14：Artic 将带宽开销有效降低了 69.77%。Artic 将准确率提升了 15.12%，并将延迟降低了 135.31 毫秒。**

## 7.4 基于轨迹的模拟

本节展示了在真实网络跟踪数据下的端到端性能增益。我们将Artic与三个基线进行比较，并分别以GCC和BBR作为底层拥塞控制算法进行测试。结果如图13所示。

**整体性能提升。** 首先，Artic 同时提升了准确率并降低了延迟。例如，使用 BBR 时（图13a），与 WebRTC 相比，Artic 将平均准确率从 79.62% 提高到 84.80%，同时将平均延迟降低了 135.31 毫秒。其次，ZeCoStream 有效弥补了 ReCapABR 导致的准确率下降。例如，使用 GCC 时，尽管 WebRTC+ReCapABR 与 WebRTC 相比将延迟从 568.13 毫秒降低到 477.49 毫秒，但准确率却损失了 4.27%（图13b）。值得注意的是，通过集成 ZeCoStream，Artic 将准确率提升了 19.39%，比 WebRTC 高出 15.12%。

**泛化。** 首先，Artic 在 GCC 和 BBR 下均提升了延迟和准确率。其次，我们发现，不同的底层拥塞控制算法会影响 Artic 在延迟和准确率之间的权衡。例如，与 BBR 相比，Artic 在 GCC 下的准确率提升幅度高出 9.94%，但延迟收益低 46.82 毫秒。这是因为 GCC 往往较为保守，在网络波动时会低估带宽。在带宽估计较低时，ReCapABR 更可能采用基于拥塞控制的适配方式，导致延迟收益减少。相反，在较低码率下，ZeCoStream 会将有限的码率分配给对响应重要的区域，从而带来更显著的准确率提升。

## 7.5 开销

本节将分析人工智能视频助手中 Artic 相较于 WebRTC 的开销。

**带宽开销。** 如§2.2所述，人工智能视频助手的带宽压力完全落在上行链路上。Artic不仅避免加重这一负担，还通过ReCapABR实际降低了上行带宽开销。图14显示，Artic将带宽使用量分别降低了46.84%和69.77%

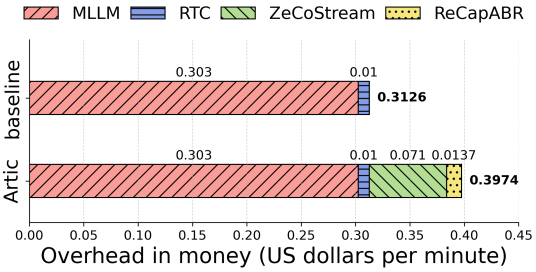



**图15：Artic 将货币成本提高了 27.13%。**

（GCC/BBR）对比WebRTC。这一降幅归因于ReCapABR的比特率封顶。在下行链路上，由于数据量极小（例如置信度分数和区域坐标），反馈开销可以忽略不计。此外，由于下行链路没有视频流传输，带宽充裕，因此反馈开销可以忽略不计。

**计算开销。** Artic 在客户端引入的计算开销可以忽略不计，因为其算法主要依赖服务器端多模态大语言模型的反馈（例如 ReCapABR 中的置信度分数和 ZeCoStream 中的重要区域）。本地仅执行一些简单的数值计算，例如将区域映射为量化参数，以及将置信度分数映射为目标比特率；这些操作的成本可以忽略不计。至于服务器端开销，由于现代多模态大语言模型通常通过 API 访问，并根据令牌用量计费，因此我们使用货币成本对该开销进行量化。**货币成本。** Artic 会给人工智能视频助手带来多少额外成本？根据实际实验支出，我们估算了各组件每分钟的成本，如图15所示。对于标准的人工智能视频助手，主要成本来自多模态大语言模型 API（$0.303/分钟） [67] 和实时通信 API（$0.01/分钟） [27, 45]。Artic 的 ZeCoStream 和 ReCapABR 分别引入了 $0.071/分钟和 $0.0137/分钟的额外成本 [28], 。因此，Artic 将总成本从基线的

$0.3126/分钟提高至 $0.3974/分钟，增幅为 27.13%。鉴于延迟降低了 135.31 毫秒，准确率提高了 15.12%（§7.4），这一成本是合理的。

## 8 结论

本文提出了 Artic，一种面向视频助手的人工智能实时通信框架。我们设计了：（1）ReCapABR，用于吸收带宽波动，从而降低延迟；（2）ZeCoStream，用于在低带宽条件下维持准确率；以及（3）DeViBench，用于评估实时通信导致的性能退化对多模态大语言模型准确率的影响。原型实验表明，与现有方法相比，Artic 将准确率提高了 15.12%，并将延迟降低了 135.31 毫秒。

本研究不会引发任何伦理问题。

**致谢**。本文是我们此前发表于 HotNets 2025 [58] 的工作的扩展版本。与研讨会版本相比，本版本新增了：（a）一项测量研究（§2.3）。（b）全新的核心贡献 ReCapABR（§4）。（c）重新设计的 ZeCoStream，具备零开销能力（§5）。（d）对 DeViBench 的重构与扩展（§6）。（e）全新的评估章节（§7）。

## 参考文献

[1] 2025. ChatGPT。（2025年）。https://chatgpt.com/。[2] 2025。Copilot Vision。（2025年）。https://support.microsoft.com/en-us/topic/using-copilot-vision-with-microsoft-copilot-3c67686f-fa97-40f6-8a3e-0e45265d425f。[3] 2025。豆包。（2025年）。https://www.doubao.com/。[4] 2025。Duolingo Max。（2025年）。https://www.duolingo.com/help/what-i s-duolingo-max/。[5] 2025。Gemini Live。（2025年）。https://gemini.google/overview/gemini-live/。[6] 2025。GLM-RealTime。（2025年）。https://zhipu-ef7018ed.mintlify.app/cn/guide/models/sound-and-video/glm-realtime。[7] 2025。Grok。（2025年）。https://grok.com/。[8] 2025。INMO。（2025年）。https://inmolens.com/。[9] 2025。Mahimahi。（2025年）。http://mahimahi.mit.edu。[10] 2025。Omi Glass 开发套件。（2025年）。https://www.omi.me/products/om i-glass-dev-kit。[11] 2025。Project-Astra。（2025年）。https://deepmind.google/models/project -astra/。[12] 2025。Qwen3-VL-Plus。（2025年）。https://www.alibabacloud.com/help/en /model-studio/models#11b9e47081cwz。[13] 2025。Raven-0。（2025年）。https://docs.tavus.io/sections/models#raven-p erception-model。[14] 2025。Ray-Ban Meta AI 眼镜。（2025年）。https://www.meta.com/ai-gla sses/social/。[15] 2025。Razor。（2025年）。https://github.com/yuanrongxi/razor。[16] 2025。Replika。（2025年）。https://replika.com/。[17] 2025。Samsung Galaxy XR。（2025年）。https://www.samsung.com/us/xr/ galaxy-xr/galaxy-xr/。[18] 2025。SOLO-Airgo。（2025年）。https://solosglasses.com/blogs/news/solos-airgo-v2-modular-smart-glasses-with-camera-and-accessible-ai。[19] 2025。Soulfan AI。（2025年）。https://soulfunai.com/。[20] 2025。Tavus。（2025年）。https://www.tavus.io/。[21] 2025。腾讯元宝。（2025年）。https://yuanbao.tencent.com/。[22] 2025. VMAF. (2025). https://github.com/Netflix/vmaf. [23] 2026. Agora 视频 SDK 3.7.0 API 参考：视频编码器配置。(2026). https://api-ref.agora.io/en/video-sdk/electron/3.x/interfaces/videoencoderconfiguration.html. [24] 2026. Google Gemini 3 Pro. (2026). https://deepmind.google/models /gemini/pro/. [25] 2026. Hurricane Electric 网络工具。(2026). https://networktools .he.net/. [26] 2026. 通过 WebRTC 使用实时 API。(2026). https://platform.openai.co m/docs/guides/realtime-webrtc. [27] Agora. 2026. Agora 的灵活定价模式。(2026). https://www.agora.io/en/pricing/ [访问日期：2月 6, 2026]. [28] byteplus. 2026. ModelArk API 文档。(2026). https://docs.byteplus.com/en/docs/ModelArk/ [访问日期：2月 6, 2026].[29] Neal Cardwell、Yuchung Cheng、C Stephen Gunn、Soheil Hassas Yeganeh 和 Van Jacobson。2017。BBR：基于拥塞的拥塞控制。《计算机协会通讯》60, 2 (2017), 58–66.[30] Gaetano Carlucci、Luca De Cicco、Stefan Holmer 和 Saverio Mascolo。2016。WebRTC（网络实时通信）Google 拥塞控制的分析与设计。载于第七届多媒体系统国际会议论文集。1–12.[31] Joya Chen、Zhaoyang Lv、Shiwei Wu、Kevin Qinghong Lin、Chenan Song、Difei Gao、Jia-Wei Liu、Ziteng Gao、Dongxing Mao 和 Mike Zheng Shou。2024。VideoLLM-Online：面向流式视频的在线视频大语言模型。载于*IEEE/CVF*计算机视觉与模式识别会议论文集。18407–18418.[32] Yihua Cheng、Ziyi Zhang、Hanchen Li、Anton Arapin、Yue Zhang、Qizheng Zhang、Yuhan Liu、Kuntai Du、Xu Zhang、Francis Y Yan 等。2024。GRACE：通过神经编解码器实现抗丢包的实时视频。载于第*21*届*USENIX*网络系统设计与实现研讨会（*NSDI 24*）。509–531.[33] Yufei Feng、Phuc Dinh、Moinak Ghoshal、Eduardo Baena、Hamza Bouchebbah 和 Dimitrios Koutsonikolas。2025。剖析运营中的5G毫米波网络波束管理。《*ACM*网络学报》3, CoNEXT2 (2025), 1–26.[34] Sadjad Fouladi、John Emmons、Emre Orbay、Catherine Wu、Riad S Wahby 和 Keith Winstein。2018。Salsify：通过视频编解码器与传输协议的紧密集成实现低延迟网络视频。载于第*15*届*USENIX*网络系统设计与实现研讨会（*NSDI 18*）。267–282。[35] Chaoyou Fu、Haojia Lin、Xiong Wang、Yi-Fan Zhang、Yunhang Shen、Xiaoyu Liu、Haoyu Cao、Zuwei Long、Heting Gao、Ke Li 等。2025。Vita-1.5：迈向 GPT-4o 级别的实时视觉与语音交互。*arXiv* 预印本 *arXiv:2501.01957* (2025).[36] Moinak Ghoshal、Omar Basit、Sizhe Wang、Phuc Dinh、Imran Khan、Yufei Feng、Zhekun Yu、Y Charlie Hu 和 Dimitrios Koutsonikolas。2025。首个大规模运营中的5G独立组网研究。《*ACM*网络学报》3, CoNEXT4 (2025), 1–21.[37] Moinak Ghoshal、Z. Jonny Kong、Qiang Xu、Zixiao Lu、Shivang Aggarwal、Imran Khan、Yuanjie Li、Y. Charlie Hu 和 Dimitrios Koutsonikolas。2022。5G毫米波网络上行链路性能深度研究。载于第二届*ACM SIGCOMM 5G*及后*5G*网络测量、建模与用例研讨会论文集（*5G-MeMU*）。[38] Yu Guan、Chengyuan Zheng、Xinggong Zhang、Zongming Guo 和 Junchen Jiang。2019。Pano：通过更好地理解质量感知来优化360 度视频流。载于*ACM*数据通信特别兴趣组会议论文集。394–407.[39] Yuting Guo、Sizhe Wang、Moinak Ghoshal、Y Charlie Hu 和 Dimitrios Koutsonikolas。2023。蜂窝网络中多用户增强现实的异步 SLAM 优势：一项测量研究。载于*2023*年新兴多媒体系统研讨会论文集。34–40。

[40] Jiahai Hu、Lin Wang、Jing Wu、Qiangyu Pei、Fangming Liu 和 Bo Li。2025。不同移动场景下跨层5G性能的比较测量研究。《计算机网络》257 (2025), 110952.[41] Aaron Hurst、Adam Lerer、Adam P Goucher、Adam Perelman、Aditya Ramesh、Aidan Clark、AJ Ostrow、Akila Welihinda、Alan Hayes、Alec Radford 等。2024。GPT-4o 系统卡。*arXiv* 预印本 *arXiv:2410.21276*(2024).[42] Imran Khan、Moinak Ghoshal、Joana Angjo、Sigrid Dimce、Mushahid Hussain、Paniz Parastar、Yenchia Yu、Xueting Deng、Sumit Hawal、Shirui Huang 等。2025。5G部署的成熟度如何? 一项为期一年的5G上行性能横截面研究。《计算机通信》237 (2025), 108153.[43] Zeqi Lai、Weisen Liu、Qian Wu、Hewu Li、Jingxi Xu 和 Jianping Wu。2022。SpaceRTC：释放巨型星座在实时通信中的低延迟潜力。载于*IEEE INFOCOM 2022——IEEE*计算机通信会议。IEEE，1339–1348.[44] Junming Lin、Zheng Fang、Chi Chen、Zihao Wan、Fuwen Luo、Peng Li、Yang Liu 和 Maosong Sun。2024。Strea mingBench：评估多模态大语言模型实现流式视频理解的差距。*arXiv* 预印本 *arXiv:2411.03628* (2024).[45] LiveKit。2026。LiveKit订阅页面。(2026). https://livekit.io/pri cing [访问日期：2月 6, 2026].[46] Zili Meng、Yaning Guo、Chen Sun、Bo Wang、Justine Sherry、Hongqiang Harry Liu 和 Mingwei Xu。2022。通过最短控制环路实现无线实时通信的稳定低延迟。载于*ACM SIGCOMM 2022*会议论文集。193–206.[47] Arvind Narayanan、Eman Ramadan、Rishabh Mehta、Xinyue Hu、Qingxu Liu、Rostand AK Fezeu、Udhaya Kumar Dayalan、Saurabh Verma、Peiqi Ji、Tao Li 等。2020。Lumos5G：绘制并预测商用毫米波5G吞吐量。载于*ACM互*联网测量会议论文集。176–193.[48] Haoran Qiu、Anish Biswas、Zihan Zhao、Jayashree Mohan、Alind Khare、Esha Choukse、Íñigo Goiri、Zeyu Zhang、Haiying Shen、Chetan Bansal、Ramachandran Ramjee 和 Rodrigo Fonseca。2025。Mod-Serve：用于可扩展多模态模型服务的模态与阶段感知资源解聚。(2025). arXiv:cs.DC/2502.00937 https://arxiv.org/abs/ 2502 00937 。通义千问。通义千问-Flash。( ). [49] 2026 2026 https://www.alibabacloud.com/help/en/model-studio/models#qwenflashintl [ 6, 访问日期：2月 .[50] Michael Rudow、Francis Y Yan、Abhishek Kumar、Ganesh Anantha-narayanan、Martin Ellis 和 KV Rashmi。2023。Tambur：通过流式编码实现高效视频会议丢包恢复。载于第*20*届*USENIX*网络系统设计与实现研讨会（*NSDI 23*）。953–971。[51] Hendrik Schippers、Melina Geis、Stefan Böocker 和 Christian Wietfeld。2025。DoNext：用于机器学习驱动的5G移动网络分析的开放获取测量数据集。*IEEE*通信与网络机器学习汇刊(2025).[52] Gemini团队、Petko Georgiev、Ving Ian Lei、Ryan Burnell、Libin Bai、Anmol Gulati、Garrett Tanzer、Damien Vincent、Zhufeng Pan、Shibo Wang 等。2024。Gemini 1.5：解锁跨数百万个上下文 token 的多模态理解。*arXiv* 预印本 *arXiv:2403.05530* (2024).[53] Wenwen Tong、Hewei Guo、Dongchuan Ran、Jiangnan Chen、Jiefan Lu、Kaibin Wang、Keqiang Li、Xiaoxu Zhu、Jiakui Li、Kehan Li 等。2025。Interactiveomni：用于视听多轮对话的统一全模态模型。*arXiv* 预印本 *arXiv:2510.13747* (2025).[54] Pavan Kumar Anasosalu Vasu、Hadi Pouransari、Fartash Faghri、Raviteja Vemulapalli 和 Oncel Tuzel。2024年。Mobileclip：通过多模态强化训练实现快速图像-文本模型。载于*IEEE/CVF*计算机视觉与模式识别会议论文集。15963–15974。[55] Marko Viitanen、Ari Koivula、Ari Lemmetti、Arttu Ylä-Outinen、Jarno Vanne 和 Timo D. Hämäläinen。2016年。Kvazaar：开源 HEVC/H.265 编码器。载于第*24*届*ACM*国际多媒体会议论文集。http://doi.acm.org/10.1145/2964284.2973796[56] Yuxuan Wang、Yueqian Wang、Bo Chen、Tong Wu、Dongyan Zhao 和 Zilong Zheng。2025年。OmniMMI：流式视频场景中的综合多模态交互基准。载于计算机视觉与模式识别会议论文集。18925–18935。[57] Jiangkai Wu、Yu Guan、Qi Mao、Yong Cui、Zongming Guo 和 Xinggong Zhang。2023年。ZGaming：通过图像预测实现零延迟3D云游戏。载于*ACM SIGCOMM 2023*会议论文集。710–723。[58] Jiangkai Wu、Zhiyuan Ren、Liming Liu 和 Xinggong Zhang。2025年。与人工智能聊天：实时视频通信从人类转向人工智能的意外转变。载于第*24*届*ACM*网络热点问题研讨会论文集。402–410。[59] x 265 2024 265 2024 265 。 。H. 。 [ 1, 2025] （ ）。https://www.itu.int/rec/T-REC-H. 访问日期：12月。[60] Jin Xu、Zhifang Guo、Jinzheng He、Hangrui Hu、Ting He、Shuai Bai、Keqin Chen、Jialin Wang、Yang Fan、Kai Dang 等。2025年。通义千问2.5-Omni技术报告。*arXiv*预印本 *arXiv:2503.20215* （2025年）。[61] Jin Xu、Zhifang Guo、Hangrui Hu、Yunfei Chu、Xiong Wang、Jinzheng He、Yuxuan Wang、Xian Shi、Ting He、Xinfa Zhu、Yuanjun Lv、Yongqi Wang、Dake Guo、He Wang、Linhan Ma、Pei Zhang、Xinyu Zhang、Hongkun Hao、Zishan Guo、Baosong Yang、Bin Zhang、Ziyang Ma、Xipin Wei、Shuai Bai、Keqin Chen、Xuejing Liu、Peng Wang、Mingkun Yang、Dayiheng Liu、Xingzhang Ren、Bo Zheng、Rui Men、Fan Zhou、Bowen Yu、Jianxin Yang、Le Yu、Jingren Z hou 和 Junyang Lin。2025年。通义千问3-Omni技术报告。（2025年）。arXiv:cs.CL/2509.17765 https: //arxiv.org/abs/2509.17765[62] Francis Y Yan、Hudson Ayers、Chenzhi Zhu、Sadjad Fouladi、James Hong、Keyi Zhang、Philip Levis 和 Keith Winstein。2020年。现场学习：一项关于视频流传输的随机实验。载于第*17*届*USENIX*网络系统设计与实现研讨会（*NSDI 20*）。495–511。[63] Jingkang Yang、Shuai Liu、Hongming Guo、Yuhao Dong、Xiamengwei Zhang、Sicheng Zhang、Pengyun Wang、Zitang Zhou、Binzhu Xie、Ziyue Wang、Bei Ouyang、Zhengyu Lin、Marco Cominelli、Zhongang Cai、Yuanhan Zhang、Peiyuan Zhang、Fangzhou Hong、Joerg Widmer、Francesco Gringoli、Lei Yang、Bo Li 和 Ziwei Liu。2026年。EgoLife：迈向自我中心生活助理。（2026年）。arXiv:cs.CV/2503.03803 https://arxiv.org/abs/2503.03803[64] Zhenyu Yang、Yuhang Hu、Zemin Du、Dizhan Xue、Shengsheng Qian、Jiahong Wu、Fan Yang、Weiming Dong 和 Changsheng Xu。 [n. d.]。SVBench：面向流式视频理解的时间性多轮对话基准。载于第十三届国际学习表征会议。[65] Z.ai。2026年。GLM-4.6V。（2026年）。 https://docs.z.ai/guides/vlm/glm-4.6v [访问日期：2月 6, 2026]。[66] Z.ai。2026年。GLM-4.6V-Flash。（2026年）。 https://huggingface.co/zai-org/G LM-4.6V-Flash [访问日期：2月 6, 2026]。[67] Z.ai。2026年。GLM-RealTime API文档。（2026年）。 https://docs.bigmodel.cn/cn/guide/models/sound-and-video/glm-realtime [访问日期：2月 6, 2026]。[68] Huanhuan Zhang、Anfu Zhou、Yuhan Hu、Chaoyue Li、Guangping Wang、Xinyu Zhang、Huadong Ma、Leilei Wu、Aiyun Chen 和

Changhui Wu。2021年。Loki：通过融合基于规则的模型，提升基于学习的实时视频适配的长尾性能。载于第*27*届移动计算与网络年度国际会议论文集。775–788。

[69] Huanhuan Zhang、Anfu Zhou、卢佳敏、马若轩、Yuhan Hu、李聪、Xinyu Zhang、Huadong Ma 和陈晓江。2020。OnRL：通过在线强化学习改进移动视频通话。载于第 *26* 届移动计算与网络年度国际会议论文集。1–14。

[70] 张攀、董晓怡、曹宇航、臧宇航、钱睿、魏希林、陈琳、李一飞、牛俊博、丁双瑞等。2024。InternLM-XComposer2.5-OmniLive：用于长期流式视频与音频交互的综合多模态系统。*arXiv* 预印本 *arXiv:2412.09596*（2024）。

# A 附录

## A.1 使用的提示词

图16展示了用于问答生成的提示词。

Prompt used for **QA Sample Generation**

**Persona**: You are an expert Visual Question Answering (VQA) system designer, specializing in perceiving fine-grained details.
**Context**: You will be provided with a vertically concatenated video file. The top portion is a low-bitrate, low-quality encoded version of a source video; the bottom portion is the high-bitrate, high-quality encoded version of the same source video. Both parts are perfectly synchronized in content.
**Core Task**: Your objective is to generate a ***quality-sensitive open-ended question-answer pair***. This involves creating a specific question and identifying the precise correct answer. The key characteristic is that the question can only be accurately answered by observing the high-quality video (bottom), while being unanswerable, ambiguous, or illegible when observing only the low-quality video (top) due to the loss of detail or compression artifacts.
**Execution Steps**:
1. ***Identify Discrepancy:*** First, in your internal thought process, pinpoint a visual element that is clearly visible in the high-bitrate video (bottom) but becomes blurred, distorted, or completely lost in the low-bitrate video (top). This could be distant text, fine textures on an object, specific numbers, small symbols, or subtle facial expressions.
2. ***Formulate Question:*** Design a question that strictly revolves around this identified "point of difference". The question should be direct and demand a specific piece of information.
3. ***Determine Answer:*** Extract the single, correct answer based on the high-bitrate video (bottom). The answer should be concise (e.g., a specific word, number, color, or short phrase) and objective.
**Constraints**:
***Wording:*** The question must be phrased neutrally and universally, as if you have only seen a single video source. **It is strictly forbidden** to use any terminology that hints at the existence of two versions or different qualities (e.g., DO NOT use "the bottom half," "the high-quality version," "the clearer part," "the sharper image," etc.).
***Focus:*** The question should focus on an objective, concrete fact (e.g., "What is the number plate?", "What does the small sign say?"), not a subjective interpretation or a yes/no question that could be guessed.
***Uniqueness:*** The answer must be impossible to determine from the low-quality top section alone.

**Output Format**: Please provide the final result strictly in the following format:
***Question:*** [Your formulated question here]
***Answer:*** [The concise correct answer]

**图16：我们用于问答样本生成的提示词。**