StreamDuo: A Live Commentator and a Research Agent for Unseen Games

Zixian Gao[1,2], **Penghao Luo**[3], **Xiangcheng Cao**[1], **Lixin Xiu**[2],
Ziqi Cai[1,4,5], **Zhen Li**[1,2], **Erwin Wu**[5], **Kaipeng Zhang**[1,†]

[1]Alaya Lab [2]The University of Tokyo [3]Georgia Institute of Technology
[4]Peking University [5]Institute of Science Tokyo

Streaming video LLMs can now commentate live gameplay, but they are trained and tested on the same games. We study cross-game streaming commentary, in which a model trained on one game commentates games it has never seen, and find that current commentators transfer their timing but not their knowledge of the game world. We propose StreamDuo, which pairs a trainable commentator with a frozen research agent that reads the wiki of the game being played. When the commentator sees something it cannot name, it sends the current frames, and the research agent returns a few wiki entries two seconds later while the stream continues. We train only the commentator, on a single game, in two stages. Information-gain labels teach it to ask only where an answer would help it say what the human caster said, and on-policy self-distillation from a copy of itself that sees the ideal answer teaches it to use noisy answers. Trained on Baldur's Gate 3 and tested on ten unseen games, StreamDuo raises entity recall from 0.082 to 0.116 over Proact-VL trained on the same data, cuts source-game leakage from 9.9% to 1.0%, and achieves the best timing and the highest judged liveliness and overall quality among streaming models.

Project Page: https://zixiangao.github.io/StreamDuo/
Code: https://github.com/ZixianGao/StreamDuo-code
Correspondence: Kaipeng Zhang
Date: November 2, 2026

ALAYA LAB



Figure 1 StreamDuo commentates Tears of the Kingdom, a game it has never seen, after training on Baldur's Gate 3 alone. When the commentator cannot name a character, it asks the research agent with the frames, and the wiki entry that arrives two seconds later lets it say who she is. Proact-VL, retrained on the same data, says something unrelated.

1 Introduction

Live game streaming has become one of the largest forms of online video, and commentary is what turns raw gameplay into a show. A good caster reacts the moment something happens and tells the audience who just appeared, what item was picked up and where the fight is taking place. Automating this requires a video model that watches an unbounded stream, decides every second whether to speak, and says something both timely and correct about the game world. Video LLMs [3, 19, 21, 40] understand short clips well, and streaming video LLMs [6, 7, 30, 42, 43, 46, 52] decide frame by frame whether to respond. Automatic commentary was first studied for sports and esports [26, 31, 37, 51], where the teams and rules are known in advance. Proact-VL [47] recently brought streaming video LLMs to game commentary and reports strong timing and quality on a multi-game benchmark.

These systems are trained and evaluated on the same games. In practice a commentator constantly meets titles, characters and items it has never seen, and the entities that make commentary informative are exactly the ones that change from one game to the next. We therefore study cross-game streaming commentary, in which a model is trained on a single game and commentates games that never appear in training. Under this setting the standard recipe breaks down in a revealing way (Fig. 1). Retrained on Baldur's Gate 3 alone, a Proact-VL commentator rarely names what is on screen in ten unseen games, and when it does, it often reaches for its training game, for example calling a character in Tears of the Kingdom "Karlach". Timing transfers across games, but knowledge does not.

A natural remedy is to give the model the missing knowledge. Retrieval-augmented generation [2, 15, 18, 24], tool-using agents [16, 34, 50] and multi-agent systems [44] all supply external information to a model. Live commentary, however, imposes constraints these methods do not address. Knowledge retrieved before the stream starts is mostly irrelevant to the current second, and in our experiments a game-level dossier barely changes what the commentator says. Retrieving at every second floods the context. Text queries are a further bottleneck, since the commentator would have to describe in words something it cannot recognize. Most importantly, the commentator must decide when an answer is worth asking for, which depends on what this particular model does not know. Language models can partly tell what they do not know [13, 17, 41], but streaming video models have no such mechanism.

We propose StreamDuo, a two-agent streaming commentator. A commentator agent watches the stream and talks. When it sees something it cannot name, it asks a research agent by sending the current frames. The research agent, a frozen vision-language model with access to the game's wiki, returns a few entries two seconds later while the stream continues. Perception and language, which transfer across games, stay in the commentator, and game-specific knowledge is looked up at test time in whatever game is being played. We train only the commentator, and only on the source game, in two stages. The first teaches it when to ask, with ask labels placed where an answer would actually help the model reproduce what the human caster said. The second teaches it how to use noisy answers, through on-policy self-distillation [54] from a privileged copy of itself that sees an ideal answer containing the right entry. On ten unseen games, StreamDuo reduces source-game leakage from 9.9% to 1.0% and raises entity recall from 0.082 to 0.116 compared with Proact-VL trained on the same data, while also improving timing and the commentary quality judged by an LLM on all ten test games.

In summary, our contributions are as follows.

- We formulate cross-game streaming commentary, together with a clean evaluation protocol and a source-game leakage metric, and show that current streaming commentators transfer their timing but not their knowledge of the game world.
- We propose StreamDuo, a two-agent commentator that asks a frozen research agent with frames and receives its answers while the stream continues, trained with information-gain ask labels and on-policy self-distillation from a privileged copy of itself.
- On ten unseen games, StreamDuo outperforms zero-shot streaming models, Proact-VL retrained on the same data, and the same models given the research agent's knowledge without interaction.

2 Related Work

Streaming video LLMs and live commentary. Video LLMs understand complete clips [3, 19–21, 40, 45], and streaming variants process frames as they arrive, either by managing memory over long streams [8, 10, 29, 36, 46, 52] or by deciding when to respond without being prompted [6, 30, 39, 42, 43, 49], as measured by online benchmarks [22, 27]. LiveCC [7] and Proact-VL [47] bring this to live commentary. Earlier commentary systems targeted a single domain such as chess [14], soccer [26, 28, 31], racing [12] or esports [37, 51, 53], often with structured game data. All of these models speak from what is in their weights and context, and are evaluated on the domains they are trained on. We test on games whose entities are never seen in training and let the commentator obtain the missing knowledge while the stream runs.

Retrieval, tools and multiple agents. Retrieval-augmented generation supplies external text [18], and adaptive variants retrieve only when the model is uncertain [2, 13, 15, 17, 41]. Knowledge-based visual question answering retrieves encyclopedic entries for an image [5, 25, 48], and video RAG retrieves auxiliary text for video understanding [24]. Tool-using agents call search while reasoning [16, 34, 50], and multi-agent frameworks let models with different roles cooperate [44]. In StreamDuo the query is the frames themselves, the answer arrives after a delay while the commentator keeps talking, and the decision to ask is learned from how much an answer would change this particular model's output.

Distillation with privileged information. A teacher with access to information unavailable at test time can guide a student [23, 38], and on-policy distillation trains the student on its own samples [1, 9] to avoid the mismatch between training on reference sequences and generating at test time [4, 33]. On-policy self-distillation [54] uses the same model, conditioned on privileged information, as its own teacher. We apply it to a two-agent setting in which the privileged information is the answer the research agent should have returned, and sharpen the teacher's use of that answer with context-aware decoding [35].

3 Method

Our goal is a commentator that keeps talking at the right moments in a game it has never seen, and that says the right names when it does. We first describe the streaming setting and why it fails across games (Sec. 3.1). We then present StreamDuo, which pairs the commentator with a research agent (Sec. 3.2), and the two training stages that teach the commentator when to ask (Sec. 3.3) and how to use what it is told (Sec. 3.4). Fig. 2 illustrates the whole pipeline.

3.1 Streaming commentary across games

A live stream arrives one second at a time. At second t the commentator receives the frames v_t of that second and a short history h_t made of the co-commentator's last line and its own recent lines. Following Proact-VL [47], each second is one turn of a video LLM π_θ. The turn ends with a special flag token whose hidden state z_t feeds a response head, which outputs the probability $p_t = \sigma(w_s^\top z_t)$ that the commentator should speak now. If it speaks, the language model generates a short line $y_t \sim \pi_\theta(\cdot \mid h_{\le t}, v_{\le t})$, and otherwise it stays silent. Both parts are trained on human commentary with a language-modeling loss on the spoken lines and a binary loss on the speak decisions,

$$\mathcal{L}_{\text{base}} = \mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{spk}}. \tag{1}$$

In cross-game commentary, π_θ is trained on a single source game G_s and evaluated on unseen games $G_t \neq G_s$. The two parts of the model behave very differently under this shift. When to speak depends on what happens on screen, such as a fight starting or a menu opening, and transfers well. What to say about the entities on screen depends on knowledge of the game world, and a model trained on one game only knows that one. Faced with an unfamiliar character, it either describes it vaguely or reaches for a name it learned from the source game. To keep the model from tying names to what it memorized, we rename the entities of the source game consistently and at random during training, so that a name in the training data is evidence only within its own clip. This removes the shortcut but cannot supply the names of a new game. That knowledge has to come from somewhere else at test time.

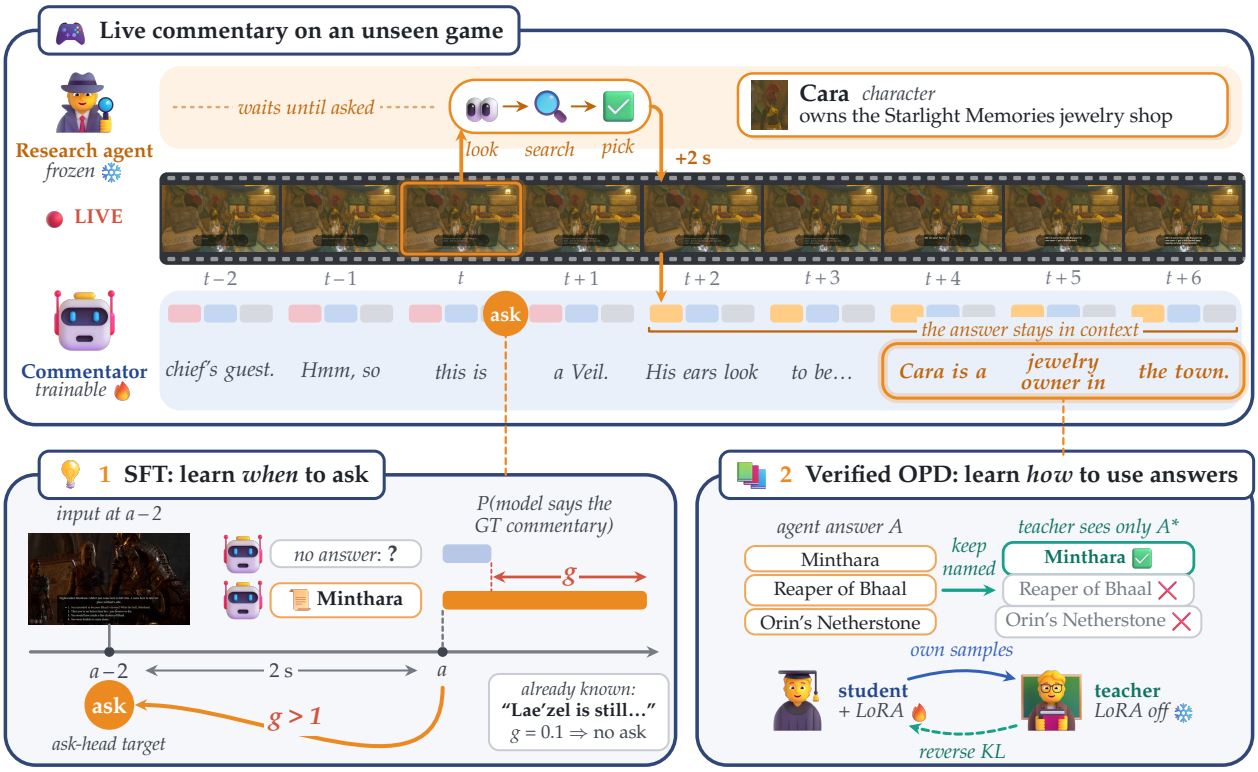


Figure 2 Overview of StreamDuo. Top: on an unseen game, the commentator asks the frozen research agent with the current frames, and the wiki entry it returns two seconds later lets the commentator name what is on screen. Bottom left: Stage 1 labels an ask only where an answer would make the human line more likely. Bottom right: Stage 2 distills a frozen copy of the commentator that sees the ideal answer into the student that sees the noisy one.

3.2 StreamDuo

StreamDuo splits the job between two agents with complementary strengths. The commentator agent π_θ watches every second of the stream and decides when to talk, but only knows the game it was trained on. The research agent $\mathcal{R}$ sees nothing of the stream unless asked, but can read the wiki of whatever game is being played. Commentary emerges from their exchange. The commentator notices that it does not recognize something, the research agent identifies it, and the commentator turns the answer into a line.

The commentator asks with frames. A natural interface would be a text query, but a model that cannot recognize a character also cannot describe it well enough to retrieve it. The commentator therefore sends the frames themselves. A second head on the same flag state estimates whether an answer is needed,

$$q_t = \sigma(w_a^\top z_t), \tag{2}$$

and the commentator sends v_t to the research agent when q_t exceeds a threshold.

The research agent looks and searches. The research agent is a frozen vision-language model with a search tool over the game's wiki. Before the stream starts, it writes a short dossier of the game, which the commentator receives as general background. The dossier cannot say which of its entities is on screen at a given second, so the real work happens when a question arrives. The research agent looks at the frames it was sent, proposes a few search keys from what it sees, including any names readable on screen, searches the wiki, looks at the frames again to check the candidates, and returns up to three entries, each with a name, a type and a one-line description. We write the returned answer as A_t. Because the research agent reads the frames rather than a description of them, it can identify a character the commentator could never have described in a text query.

Answers arrive while the stream continues. Looking things up takes time, and the stream does not wait. The answer to a question asked at second t arrives δ seconds later and is placed in the history $h_{t+\delta}$, where it stays available for the following seconds. Meanwhile the commentator keeps deciding every second whether to speak. This asynchrony is part of the problem the commentator has to learn to live with. It must ask before it needs the answer, and connect an answer to what it is seeing when the answer arrives.

Speaking and stopping. An answer often gives the commentator something to say, and a model that has started an explanation tends to keep going after the moment passes. We therefore use a higher bar for continuing than for starting. The commentator starts speaking when p_t exceeds τ, and after a run of spoken seconds continues only when p_t exceeds a larger threshold τ_c. It can thus react immediately to new events and stop once the event ends.

Only the commentator is trained, and only on the source game. The research agent stays frozen and brings its knowledge from the wiki of the game at hand, so the same pair can be pointed at any game for which a wiki exists. The commentator has to learn two things that standard training does not teach, namely when an answer is worth asking for and how to turn a noisy answer into the right line.

3.3 Learning when to ask

The obvious supervision for the ask head is to ask shortly before the human caster names an entity. This ignores the one thing that makes asking useful, which is whether the model already knows the answer. A model that reliably recognizes the protagonist gains nothing from asking about her, while a model that cannot tell two bosses apart gains a lot. The label should depend on the model.

We define the value of asking through the model itself. Let y_a be the line the human says at second a and let $A_a^\star$ be an ideal answer that contains the entity named in y_a. A frozen copy of the commentator, π_{θ_0}, scores this line twice, once with the answer in its history and once without it. The information gain

$$g_a = \log \pi_{\theta_0}(y_a \mid h_{\le a}, A_a^\star, v_{\le a}) - \log \pi_{\theta_0}(y_a \mid h_{\le a}, v_{\le a}) \tag{3}$$

measures how much the answer helps the model say what the caster said. We label second $a - \delta$ as an ask moment when $g_a > \gamma$, so that the answer would arrive in time. Lines the model can already produce receive no label because their gain is small, and so do lines that no answer could help with, such as tactical remarks. What remains are the moments when the model does not know and an answer would fix it.

The ask head is trained with a binary loss on these labels together with the base objective,

$$\mathcal{L}_1 = \mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{spk}} + \lambda\, \mathcal{L}_{\text{ask}}. \tag{4}$$

In the same stage the commentator learns to read answers. During training, each answer placed in the history contains the entity the caster names next together with distractor entries from the same game, and some answers miss the entity altogether, mimicking what a real research agent returns. Because entity names are randomly renamed in each clip, the model cannot have memorized them and must take them from the answer that it receives.

3.4 Learning to use the answer

Supervised training on answers teaches the format of an answer but not how to use it. The answers the commentator receives at test time come from a real research agent. They are often plausible but wrong, and the right entry, when present, is mixed with others. A model fine-tuned to copy entries from its history will repeat whatever arrives, and a model that ignores its history will never say the new names. What we want is a model that uses an entry when it matches the frames and otherwise does not.

We obtain this behavior with on-policy self-distillation [54]. The student is the commentator π_θ, which sees a noisy answer A exactly as it would at test time. The teacher is a frozen copy of the same model, π_{θ_0}, which sees the ideal answer $A^\star$ for the same moment. The teacher is not a larger model. It knows more only because it is told which entry is right, and its behavior shows what using that entry looks like.

The effect of an answer on a fine-tuned model is often small, because the model has learned to rely on the frames and its own memory. We make the teacher's use of the answer explicit by contrasting its predictions with and without the answer [35],

$$\log p_T(\cdot) = \log \mathrm{softmax}\Big((1+\alpha)\, \ell_{\theta_0}(\cdot \mid A^\star) - \alpha\, \ell_{\theta_0}(\cdot \mid \varnothing) \Big), \tag{5}$$

where ℓ_{θ_0} denotes the teacher's next-token logits given the same frames and history and $\alpha \geq 0$ controls the amplification. Tokens that become likely because of the answer, such as the name of the entity on screen, are pushed up, and tokens the model would say anyway are left unchanged.

Training is on-policy. For a moment right after an answer arrives, the student samples its own lines $\hat{y} \sim \pi_\theta(\cdot \mid h, A, v)$, and the teacher scores each of their tokens. We minimize the reverse Kullback-Leibler divergence over the student's samples,

$$\mathcal{L}_2 = \mathbb{E}_{\hat{y} \sim \pi_\theta} \Big[\sum_k \mathrm{KL}\big(\pi_\theta(\cdot \mid \hat{y}_{<k}, h, A, v) \,\big\|\, p_T(\cdot \mid \hat{y}_{<k}) \big) \Big]. \tag{6}$$

Sampling from the student matters here. The student is corrected on the lines it would actually say, including the ones where it repeats a wrong entry, rather than on human lines it would never produce. The reverse divergence makes the student commit to what the teacher considers likely instead of spreading its probability over every name in the answer. Only a small set of low-rank adapter weights is updated in this stage, so the speaking behavior learned before is preserved, and the teacher is simply the model with the LoRA adapter switched off during its forward pass.

At test time neither the teacher nor the ideal answer is used. The commentator watches the stream, asks with its frames when its ask head fires, and speaks with whatever the research agent returns.

4 Experiments

4.1 Setup

Data. We use the game commentary benchmark of Proact-VL [47], training on 10,286 Baldur's Gate 3 clips and testing on 2,470 clips from ten other games, from open-world adventures such as Tears of the Kingdom to esports such as League of Legends. The original inputs leak game knowledge through the caster persona, the earlier transcript and the other caster's lines. We therefore keep only the persona's speaking style, remove earlier transcripts and replace game-specific names in the other caster's lines with typed placeholders such as [PLAYER]. Game knowledge can then only come from the wiki of 190,722 pages.

Metrics. For timing we report TimeDiff and F1 of Proact-VL. For knowledge we match wiki entities in the two commentaries. Ent-R is the share of the human's entities that the model also names, Ent-F1 adds precision, and K-strict counts only lines that need game knowledge. These count the first 126 words of each clip, twice the human average, so that talking more cannot raise recall. Qwen3-30B [3] scores PAUC, CC, LiveU and FinalQ of Proact-VL on all test clips.

Baselines. We compare with streaming video LLMs that decide when to respond, VideoLLM-online [6], LiveCC [7], Dispider [30], MMDuet [42], StreamingVLM [46], MMDuet2 [43] and Proact-VL [47], each with its official code and the game title as context. We test them zero-shot and after one epoch on the same Baldur's Gate 3 clips with the official recipe. Proact-VL has seen the test games, so we retrain it from LiveCC-7B-Base. Dispider releases no training code, and for MMDuet2 we rerun only the supervised stage. Each trained baseline is also given the research agent's dossier, to separate the knowledge itself from the way in which StreamDuo obtains it during the stream.

Implementation. The commentator starts from LiveCC-7B-Base [7], as Proact-VL does. Stage 1 fine-tunes all parameters for one epoch at a learning rate of 10^{-5} with $\gamma = 1$ nat. Stage 2 trains a rank-32 LoRA [11] for 300 steps with $\alpha = 1$. The research agent is Qwen3-VL-8B [3] with BM25 search [32] and returns up to three entries after $\delta = 2$ s. At test time it speaks when $p_t > 0.27$ and asks when $q_t > 0.9$. All runs use eight NVIDIA H100 GPUs for training and evaluation.

Table 1 Cross-game streaming commentary. Models are trained on Baldur's Gate 3 only (or not trained) and evaluated on 2,470 clips from ten unseen games. Knowledge metrics use the first 126 words of each clip (twice the human average), so a model cannot gain recall by talking more. Judge metrics are scored by Qwen3-30B. ‡Supervised stage of the official recipe only, without RL. **Bold**: best; underline: second best.

		Timing		Knowledge			LLM judge			
Method	Venue	TimeDiff↓	F1↑	Ent-R↑	Ent-F1↑	K-strict↑	PAUC↑	CC↑	LiveU↑	FinalQ↑
Zero-shot										
VideoLLM-online [6]	CVPR'24	11.63	0.063	0.002	0.003	0.001	0.000	0.134	3.49	2.39
LiveCC-7B-Instruct [7]	CVPR'25	3.03	0.470	0.099	0.139	0.115	0.031	0.127	3.56	2.59
Dispider [30]	CVPR'25	6.10	0.156	0.043	0.042	0.055	0.006	0.083	3.61	2.78
MMDuet [42]	EMNLP'25	4.01	0.297	0.017	0.023	0.020	0.002	0.049	3.55	2.50
StreamingVLM [46]	ICLR'26	1.83	0.509	0.076	0.081	0.086	0.025	0.097	4.58	3.25
MMDuet2 [43]	ICLR'26	3.52	0.330	0.073	0.091	0.083	0.026	0.107	5.05	3.47
Trained on Baldur's Gate 3										
VideoLLM-online [6]	CVPR'24	5.41	0.228	0.000	0.000	0.000	0.001	0.124	3.71	2.27
+ agent dossier		4.25	0.231	0.001	0.001	0.001	0.001	0.129	3.63	1.98
LiveCC-7B-Instruct [7]	CVPR'25	24.08	0.036	0.028	0.053	0.034	0.031	0.306	1.73	1.55
+ agent dossier		19.89	0.072	0.031	0.050	0.034	0.030	0.252	2.35	1.81
MMDuet [42]	EMNLP'25	3.62	0.256	0.008	0.015	0.009	0.001	0.094	4.30	2.39
+ agent dossier		16.46	0.072	0.004	0.009	0.010	0.001	0.166	4.17	2.57
StreamingVLM [46]	ICLR'26	1.95	0.478	0.076	0.111	0.083	0.039	0.090	4.77	3.20
+ agent dossier		2.07	0.467	0.066	0.097	0.071	0.036	0.087	4.73	3.25
MMDuet2‡ [43]	ICLR'26	24.07	0.016	0.021	0.039	0.018	0.029	0.312	1.75	1.52
+ agent dossier		24.33	0.015	0.020	0.038	0.018	0.027	**0.313**	1.71	1.50
Proact-VL [47]	ICML'26	1.84	0.536	0.082	0.118	0.085	0.050	0.095	4.91	3.32
+ agent dossier		1.88	0.548	0.083	0.111	0.088	0.048	0.089	4.80	3.30
StreamDuo (ours)	–	**1.76**	**0.552**	**0.116**	**0.141**	**0.127**	**0.055**	0.120	**5.14**	**3.53**

4.2 Main results

Tab. 1 shows that existing streaming models fail on unseen games in two ways. Zero-shot, they rarely speak at the right moments. LiveCC names the most entities among them but has a TimeDiff of 3.03 s, and the others either stay silent or describe the scene. After training on Baldur's Gate 3, Proact-VL and StreamingVLM learn when to speak, but their entity recall stays below that of zero-shot LiveCC, and VideoLLM-online speaks more often without naming a single correct entity. Models that decide silence through their language output, such as LiveCC and MMDuet2, collapse to a few words per clip and mostly read the dialogue shown on screen. Timing transfers across games, knowledge does not. Adding the research agent's game dossier to their context barely helps, for example raising the K-strict of Proact-VL by only 0.003, since a dossier covers the whole game but cannot tell which entity is on screen. StreamDuo is best on all metrics except CC. Compared with Proact-VL, trained from the same base on the same clips, it raises entity recall from 0.082 to 0.116 and K-strict from 0.085 to 0.127, while responding faster and receiving higher judge scores. It also names more entities than zero-shot LiveCC without giving up timing. CC favors near-silent models in this comparison, such as MMDuet2 with a few words per clip, and does not reflect the quality of the commentary itself.

4.3 Contribution of each component

Tab. 2 adds the components of StreamDuo one at a time, all with the same inference settings. The first row follows the recipe of Proact-VL. Renaming the entities of the source game and adding the dossier do not raise entity recall (0.082 to 0.083), which agrees with the dossier results in Tab. 1, but they make the commentary more natural, raising LiveU from 4.91 to 5.03. Asking the research agent with the frames adds what the dossier cannot provide, the identity of what is on screen, and lifts Ent-R to 0.092 while also improving timing. OPSD gives the largest gain of the three components. It raises Ent-R from 0.092 to

Table 2 Ablation of StreamDuo components. Rename & dossier: random entity renaming during training plus the game dossier in context. Asking: Stage 1. OPSD: Stage 2. The first row corresponds to the Proact-VL baseline trained on Baldur's Gate 3. Best results are in bold.

Components			Timing		Knowledge			LLM judge			
Rename & dossier	Asking	OPSD	TimeDiff↓	F1↑	Ent-R↑	Ent-F1↑	K-strict↑	PAUC↑	CC↑	LiveU↑	FinalQ↑
×	×	×	1.84	0.536	0.082	0.118	0.085	0.050	0.095	4.91	3.32
✓	×	×	1.88	0.523	0.083	0.116	0.090	0.045	0.093	5.03	3.38
✓	✓	×	1.78	0.544	0.092	0.123	0.094	0.052	0.104	5.10	3.48
✓	✓	✓	**1.76**	**0.552**	**0.116**	**0.141**	**0.127**	**0.055**	**0.120**	**5.14**	**3.53**

0.116 and K-strict from 0.094 to 0.127, as the commentator learns to use the answers it receives instead of only asking for them. Together the three components raise Ent-R by 0.034 and K-strict by 0.042 over the Proact-VL recipe, and the full model is best on every metric.

4.4 When to ask

Table 3 When to ask. *Top:* Stage 1 ask labels, placed at every human entity mention (naive) or only where the answer yields an information gain of $\gamma = 1$ nat (info. gain). *Bottom:* test-time asking policy of the full model. Shaded rows are the settings used by StreamDuo; best results within each block are in bold.

	Timing		Knowledge			LLM judge			
Setting	TimeDiff↓	F1↑	Ent-R↑	Ent-F1↑	K-strict↑	PAUC↑	CC↑	LiveU↑	FinalQ↑
Stage 1 ask labels									
Naive	1.903	0.534	0.079	0.107	0.087	0.046	0.098	5.01	3.38
Info. gain	**1.784**	**0.544**	**0.092**	**0.123**	**0.094**	**0.052**	**0.104**	**5.10**	**3.48**
Test-time asking									
Never	1.764	0.553	0.104	0.127	0.114	0.051	0.117	**5.14**	**3.54**
Every 3 s	1.800	**0.563**	**0.199**	0.105	**0.208**	0.043	0.086	4.76	3.36
Learned	**1.763**	0.552	0.116	**0.141**	0.127	**0.055**	**0.120**	**5.14**	3.53

The upper part of Tab. 3 changes the labels that train the ask head. Labeling every second in which the human names an entity also marks names on screen or already known, and every metric drops, Ent-R from 0.092 to 0.079 and FinalQ from 3.48 to 3.38. The lower part changes the asking policy of the full model at test time. Without asking, timing and judge scores stay the same but knowledge falls, to an Ent-R of 0.104 and a K-strict of 0.114. Asking every 3 s nearly doubles recall, but many of the added names do not fit the moment, so Ent-F1 falls to 0.105 and all judge scores drop. The learned policy asks 264 times over 2,470 clips and gives the best balance, with the highest Ent-F1 of 0.141.

4.5 When does the commentator ask?

Fig. 3 aligns the commentary of StreamDuo with its own questions on the test games. Questions are rare, 264 over 2,470 clips. Before a question, the commentator names an entity in only 1 to 4% of the seconds, below the 7% of a random second, so it asks when it has no name to offer. When the answer arrives two seconds later, the share jumps to 32%, 4.6 times that of a random second. It stays above the random rate for five more seconds, at 22% one and two seconds after the answer and 12% five seconds after it, and returns to the base rate within about eight seconds. The jump comes from the answer and not from the scene. With asking disabled, the same seconds name an entity only 5% of the time. The commentator thus uses an answer at once.

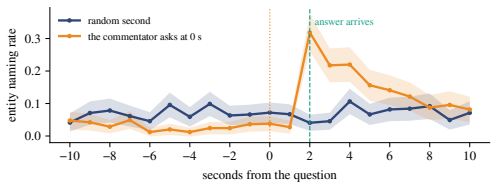


Figure 3 Entity naming around questions (orange) and random seconds (blue). The answer arrives at 2 s.

4.6 What Stage 2 needs

Fig. 4 varies the loss and the teacher of Stage 2. Fine-tuning on the real answers of the research agent gives the lowest TimeDiff and a higher LiveU, but entity recall falls below that of Stage 1 (0.086 against 0.092). Without amplification ($\alpha = 0$) the gain over Stage 1 is small. The forward divergence spreads the student over all the tokens the teacher allows, so the student names an entity wherever the amplified teacher gives it some probability. It reaches the highest Ent-R (0.131) and K-strict (0.145), but one in five of its entity mentions is a name from Baldur's Gate 3 (0.193, against 0.010 for the full model), and its LiveU and FinalQ fall to 4.82 and 3.31, the lowest of all variants. All three loss variants also lower FinalQ. Weaker teachers keep the quality of the commentary but add less knowledge. A teacher that only knows which received entries the human names, or one that checks them on screen with a frozen VLM, reaches an Ent-R of 0.093 or 0.102. Only the full recipe, the reverse divergence with amplification and the ideal answer, improves all knowledge and judge metrics over Stage 1, raising Ent-R to 0.116 and K-strict to 0.127 without leaking names from the source game.

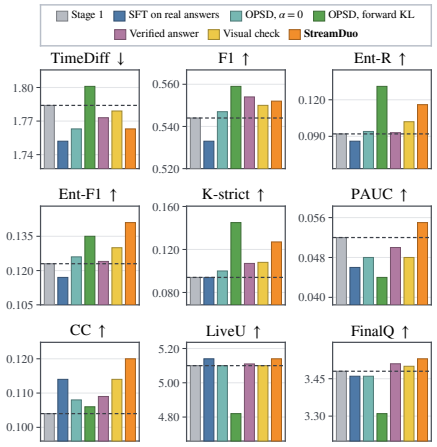


Figure 4 Stage 2 variants. Dashed line: Stage 1.

4.7 Swapping the research agent

Table 4 Swapping the research agent at test time without retraining the commentator. All agents answer the same moments, every second of the 2,470 test clips (80,529 in total). Answers: number of these moments for which the agent returns an answer. The shaded row is the default agent used by StreamDuo. Best results are in bold.

		Knowledge			LLM judge	
Research agent	Answers	Ent-R↑	Ent-F1↑	K-strict↑	LiveU↑	FinalQ↑
Qwen3-VL-2B	18,103	0.101	0.125	0.107	5.14	3.53
Qwen3-VL-4B	50,445	0.105	0.129	0.111	5.13	3.52
Qwen3-VL-8B	52,548	**0.116**	**0.141**	**0.127**	5.14	3.53
Qwen3-VL-30B-A3B	54,888	0.107	0.129	0.117	**5.15**	**3.55**

Tab. 4 replaces the research agent at test time with other Qwen3-VL models, from 2B to 30B-A3B, keeping the commentator and the wiki search unchanged. All agents answer every second of the test set, so the 8B row equals the main result. The 8B agent gives the most knowledge, but the choice of agent matters little. Ent-R stays between 0.101 and 0.116 and the judge scores within 0.03 of each other, even though the 2B agent answers only 18,103 of the 80,529 moments against about 50,000 for the larger ones, and 30B-A3B does not improve on 8B. Within this family and search, the size of the research agent is not the bottleneck. What matters more is how the commentator uses the entries it receives. With the same 8B agent, Stage 2 alone raises Ent-R from 0.092 to 0.116 (Sec. 4.6), far more than any change of agent.

4.8 Qualitative examples

Fig. 5 shows StreamDuo and Proact-VL at the same seconds of seven unseen games. In the first five examples the commentator asks and, two seconds later, names what the answer contains. In StarCraft II it names Byun's Ghost Academy, which the human caster also names, and the player GuMiho in another match. In Elden Ring and Minecraft the entry is also on screen, in the equipment menu, the chest and the skill slot of Thops's Barrier, so the research agent finds it from the frames although the commentator cannot read it. The answers are not clean. Each holds up to three entries, and the commentator mostly keeps the ones that match the frames, Bloodhound's Fang but not Bloodhound's Finesse or Slash, and the golden apple but not the enchanted one. It does not always filter well, and in StarCraft II it also repeats the Warden Support Alpha, an entry unrelated to the match. Proact-VL speaks at the same seconds but stays vague, as in "Wait, this is a weapon", names the wrong potion, or calls an Elden Ring enemy group the Sharran, a

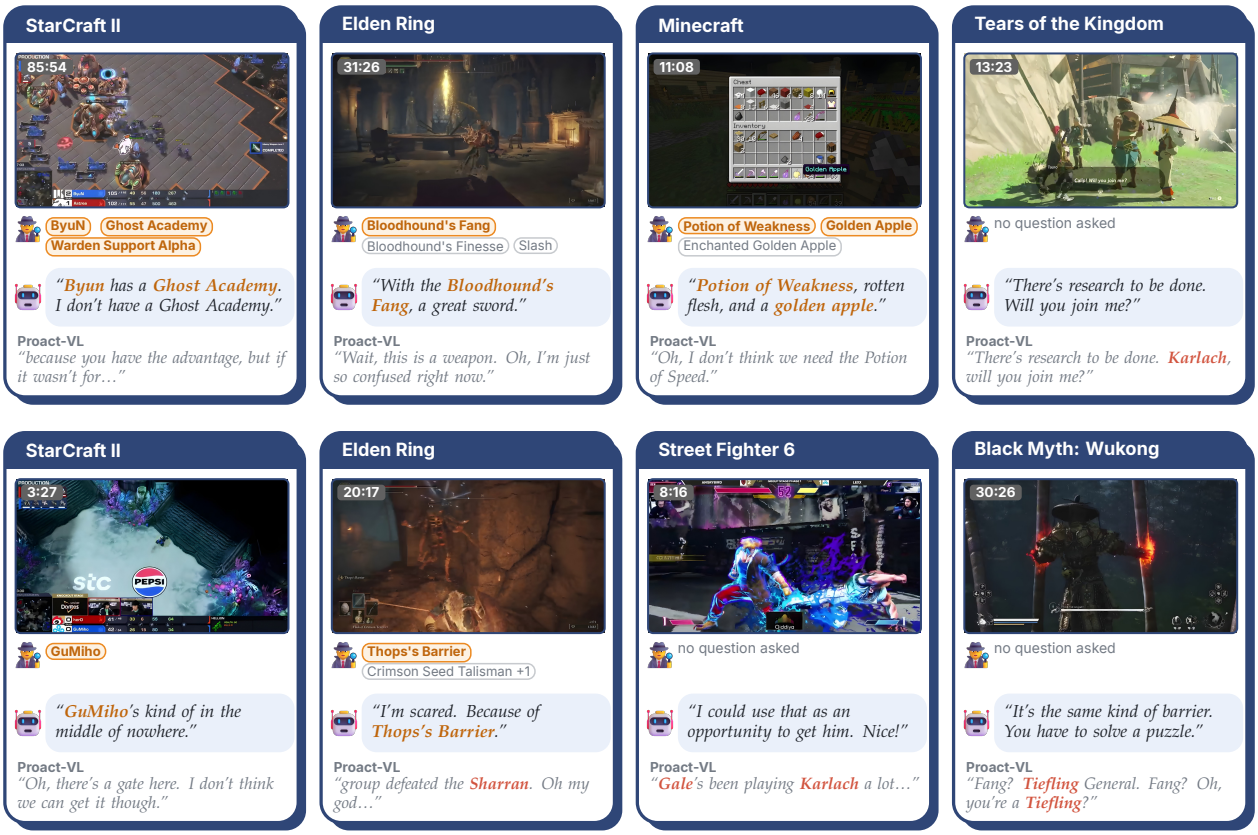


Figure 5 Examples on unseen games, with the frame at the second the research agent's answer arrives. Orange entries are used by the commentator, and gray ones are not. Proact-VL is retrained on the same data and shown at the same seconds, and names from Baldur's Gate 3 are in red.

faction of Baldur's Gate 3. The last three examples show what happens without a question. In Tears of the Kingdom, Street Fighter 6 and Black Myth: Wukong the commentator does not ask, since its ask score stays below the threshold. StreamDuo describes the scene without a name, while Proact-VL fills in names from its training game, addressing a character as Karlach, saying a fighter has been playing Karlach, and calling a boss a Tiefling. Such names are not rare. Over all test clips, names from the source game make up 9.9% of the entity mentions of Proact-VL, 200 of its 2,011, and 1.0% of ours, 34 of 3,363, although StreamDuo names almost twice as many entities. Without an answer, the commentator leaves a name out instead of borrowing one from the game it was trained on.

5 Conclusion

We studied cross-game streaming commentary, in which a model trained on one game commentates games it has never seen, and found that current streaming commentators transfer their timing but not their knowledge. To address this, StreamDuo pairs a trainable commentator with a frozen research agent that it asks with its frames and that answers from the game's wiki while the stream continues. Information-gain labels teach the commentator when an answer is worth asking for, and on-policy self-distillation from a privileged copy of itself teaches it to use noisy answers. Trained on Baldur's Gate 3 alone, StreamDuo names more of what is on screen in ten unseen games, rarely falls back on names from its training game, and keeps the timing and quality of the strongest streaming commentators, without any training on the ten games it is tested on.

Acknowledgments

This study was carried out using the TSUBAME4.0 supercomputer at Institute of Science Tokyo.

References

[1] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263, 2024.

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pages 9112–9141, 2024.

[3] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[4] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[5] Davide Caffagni, Federico Cocchi, Nicholas Moratelli, Sara Sarto, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Wiki-llava: Hierarchical retrieval-augmented generation for multimodal llms. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1818–1826. IEEE, 2024.

[6] Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18407–18418. IEEE, 2024.

[7] Joya Chen, Ziyun Zeng, Yiqi Lin, Wei Li, Zejun Ma, and Mike Zheng Shou. Live: Learning video llm with streaming speech transcription at scale. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29083–29095. IEEE, 2025.

[8] Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. Streaming video question-answering with in-context video kv-cache retrieval. In *International Conference on Learning Representations*, volume 2025, pages 42115–42127, 2025.

[9] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pages 32694–32717, 2024.

[10] Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. Ma-lmm: Memory-augmented large multimodal model for long-term video understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13504–13514. IEEE, 2024.

[11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[12] Tatsuya Ishigaki, Goran Topić, Yumi Hamazono, Hiroshi Noji, Ichiro Kobayashi, Yusuke Miyao, and Hiroya Takamura. Generating racing game commentary from vision, language, and structured data. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 103–113, 2021.

[13] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 conference of the north american chapter of the association for computational linguistics: Human language technologies (volume 1: Long papers)*, pages 7036–7050, 2024.

[14] Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1671, 2018.

[15] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 7969–7992, 2023.

[16] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[17] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[18] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[19] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[20] Yanwei Li, Chengyao Wang, and Jiaya Jia. Llama-vid: An image is worth 2 tokens in large language models. In *European conference on computer vision*, pages 323–340. Springer, 2024.

[21] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 5971–5984, 2024.

[22] Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12147–12151. IEEE, 2026.

[23] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*, 2015.

[24] Yongdong Luo, Xiawu Zheng, Guilin Li, Shukang Yin, Haojia Lin, Chaoyou Fu, Jinfa Huang, Jiayi Ji, Fei Chao, Jiebo Luo, et al. Video-rag: Visually-aligned retrieval-augmented long video comprehension. *Advances in Neural Information Processing Systems*, 38:168008–168033, 2026.

[25] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 3190–3199. IEEE, 2019.

[26] Hassan Mkhallati, Anthony Cioppa, Silvio Giancola, Bernard Ghanem, and Marc Van Droogenbroeck. Soccernet-caption: Dense video captioning for soccer broadcasts commentaries. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5074–5085. IEEE, 2023.

[27] Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. Ovo-bench: How far is your video-llms from real-world online video understanding? In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18902–18913. IEEE, 2025.

[28] Ji Qi, Jifan Yu, Teng Tu, Kunyu Gao, Yifan Xu, Xinyu Guan, Xiaozhi Wang, Bin Xu, Lei Hou, Juanzi Li, et al. Goal: A challenging knowledge-grounded video captioning benchmark for real-time soccer commentary generation. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 5391–5395, 2023.

[29] Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Shuangrui Ding, Dahua Lin, and Jiaqi Wang. Streaming long video understanding with large language models. *Advances in Neural Information Processing Systems*, 37:119336–119360, 2024.

[30] Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24045–24055. IEEE, 2025.

[31] Jiayuan Rao, Haoning Wu, Chang Liu, Yanfeng Wang, and Weidi Xie. Matchtime: Towards automatic soccer game commentary generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1671–1685, 2024.

[32] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 4(1-2):1–174, 2009.

[33] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[34] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

[35] Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 783–791, 2024.

[36] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18221–18232. IEEE, 2024.

[37] Tsunehiko Tanaka and Edgar Simo-Serra. Lol-v2t: Large-scale esports video description dataset. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4552–4561. IEEE, 2021.

[38] Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 22(5-6):544–557, 2009.

[39] Haibo Wang, Bo Feng, Zhengfeng Lai, Mingze Xu, Shiyu Li, Weifeng Ge, Afshin Dehghan, Meng Cao, and Ping Huang. Streambridge: Turning your offline video large language model into a proactive streaming assistant. *Advances in Neural Information Processing Systems*, 38:132332–132359, 2026.

[40] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[41] Yile Wang, Peng Li, Maosong Sun, and Yang Liu. Self-knowledge guided retrieval augmentation for large language models. In *Findings of the association for computational linguistics: EMNLP 2023*, pages 10303–10315, 2023.

[42] Yueqian Wang, Xiaojun Meng, Yuxuan Wang, Jianxin Liang, Jiansheng Wei, Huishuai Zhang, and Dongyan Zhao. Videollm knows when to speak: Enhancing time-sensitive video comprehension with video-text duet interaction format. In *EMNLP (Findings)*, pages 6338–6359, 2025.

[43] Yueqian Wang, Songxiang Liu, Disong Wang, Nuo Xu, Guanglu Wan, Huishuai Zhang, and Dongyan Zhao. Mmduet2: Enhancing proactive interaction of video mllms with multi-turn reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pages 23257–23270, 2026.

[44] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

[45] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models. *arXiv preprint arXiv:2407.15841*, 2024.

[46] Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. Streamingvlm: Real-time understanding for infinite video streams. In *International Conference on Learning Representations*, volume 2026, pages 61463–61475, 2026.

[47] Weicai Yan, Yuhong Dai, Qi Ran, Haodong Li, Wang Lin, Tao Jin, Xing Xie, Hao Liao, and Jianxun Lian. Proact-vl: A proactive videollm for real-time ai companions. *arXiv preprint arXiv:2603.03447*, 2026.

[48] Yibin Yan and Weidi Xie. Echosight: Advancing visual-language models with wiki knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1538–1551, 2024.

[49] Zhenyu Yang, Kairui Zhang, Yuhang Hu, Bing Wang, Shengsheng Qian, Bin Wen, Fan Yang, Tingting Gao, Weiming Dong, and Changsheng Xu. Livestar: Live streaming assistant for real-world online video understanding. *Advances in Neural Information Processing Systems*, 38:31266–31304, 2026.

[50] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[51] Dawei Zhang, Sixing Wu, Yao Guo, and Xiangqun Chen. Moba-e2c: Generating moba game commentaries via capturing highlight events from the meta-data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4545–4556, 2022.

[52] Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams. *arXiv preprint arXiv:2406.08085*, 2024.

[53] Zhihao Zhang, Feiqi Cao, Yingbin Mo, Yiran Zhang, Josiah Poon, and Caren Han. Game-mug: Multimodal oriented game situation understanding and commentary generation dataset. *arXiv preprint arXiv:2404.19175*, 2024.

[54] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Appendix

A Source-game leakage

A commentator trained on one game can fill an entity it does not know with a name it memorized from that game. We call this source-game leakage and measure it as the share of Baldur's Gate 3 names among all entity names a model says in the unseen games, counting only names that exist neither in the test game nor in its wiki. Tab. 5 reports it for all models trained on Baldur's Gate 3, on all 2,470 clips and without a word cap.

The two models that speak the most entity names before StreamDuo leak the most. 17.3% of the names said by StreamingVLM and 9.9% of those said by Proact-VL come from Baldur's Gate 3, such as Karlach in Street Fighter 6 or Lae'zel in League of Legends. Leakage is worst in esports, where the commentary is full of player and champion names the model has never seen. In League of Legends, 54.6% of the names said by Proact-VL come from Baldur's Gate 3, against 7.8% for StreamDuo. The agent dossier lowers the overall share only a little, to 14.7% and 6.3%, since it lists the entities of the game but cannot tell which one is on screen. The other baselines leak less because they name few entities at all, with fewer than 900 test-game

Table 5 Source-game leakage on all 2,470 test clips, without a word cap. Mentions: names of the test game. Leaked: Baldur's Gate 3 names that exist neither in the test game nor in its wiki. Leakage: share of leaked names among all names said.

Method	Mentions↑	Leaked↓	Leakage↓
VideoLLM-online [6]	372	14	3.6%
+ *agent dossier*	676	1	0.1%
LiveCC-7B-Instruct [7]	217	2	0.9%
+ *agent dossier*	867	4	0.5%
MMDuet [42]	298	0	0.0%
+ *agent dossier*	99	1	1.0%
StreamingVLM [46]	1,271	265	17.3%
+ *agent dossier*	1,188	205	14.7%
MMDuet2‡ [43]	147	4	2.6%
+ *agent dossier*	135	2	1.5%
Proact-VL [47]	1,811	200	9.9%
+ *agent dossier*	2,533	170	6.3%
StreamDuo (ours)	**3,329**	34	1.0%

names over all 2,470 clips. A low share then says little about their knowledge. StreamDuo names the most entities of the test games, 3,329, and keeps leakage at 1.0%.

Each part of StreamDuo contributes to this. Run with the same inference settings as StreamDuo, the recipe of Proact-VL leaks 5.8% of its names. Random renaming with the dossier in context (Stage 0) lowers this to 2.1%, since a name from the source game is no longer reliable evidence during training. Learning when to ask (Stage 1) lowers it further to 0.8%. With asking disabled at test time, the full model leaks 2.9%, three times as much as with asking. When the commentator does not know an entity, it asks the research agent instead of reaching for a name from the source game. The loss of Stage 2 matters as well. With the forward instead of the reverse divergence, leakage rises to 19.3%, the highest of all models, because the student learns to name an entity wherever the teacher gives it some probability (Sec. 4.6).

A.1 Efficiency

Table 6 Latency on one NVIDIA H100 GPU, one request at a time. Research agent: seconds per answer over 200 test moments. Commentator: seconds of compute per second of video over 20 test clips, including frame decoding.

Stage	Mean	Median	P90	Max
Research agent (s per answer)				
Propose search keys	0.322	0.272	0.594	0.628
BM25 wiki search	0.002	0.001	0.005	0.011
Pick entries	0.082	0.091	0.114	0.188
Total	0.405	0.363	0.643	0.713
Commentator (s per video second)				
Total	0.256	0.242	0.306	0.332

A live commentator has to keep up with the stream, and an answer is only useful if it arrives while the entity is still on screen. Tab. 6 measures both agents on one H100 GPU with one request at a time, as in a live stream, using the same models, prompts and settings as in all other experiments.

The research agent answers in 0.41 s on average and in at most 0.71 s over 200 test moments, so every answer is ready before the 2 s at which StreamDuo places it in the history. The delay of 2 s therefore leaves a margin of more than a second, and a slower or larger agent would still fit. Most of this time goes to proposing search keys from the frames (0.32 s). Picking up to three entries from the retrieved candidates

takes 0.08 s, and the BM25 search over the wiki takes about 2 ms. Building the search index is done once before the stream and takes 1.5 s.

The commentator needs 0.26 s of compute for each second of video, about four times faster than real time, and at most 0.33 s on the slowest of 20 test clips. This includes decoding the frames, the speak and ask heads and the generation of each line. All 20 clips are mostly spoken, so the measurement covers the most expensive case.

The research agent runs in parallel and is called only when the commentator asks. In one test run, the commentator asked 264 times over the 26 hours of test video, about once every six minutes, and in 243 of the 2,470 clips. The research agent is therefore idle most of the time, and its cost adds little to that of the commentator. StreamDuo can commentate a live stream on two GPUs, one for each agent.

B More qualitative examples

Fig. 6 and Fig. 7 show 32 more examples from all ten unseen games, none of them in Fig. 5. Each card shows the frame at the given time, the entries that the research agent returns, StreamDuo's commentary and that of Proact-VL, retrained on the same data, at the same seconds. All text is the models' own output; "…" marks a cut.

In 21 examples the commentator asks and names what the answer contains within a few seconds. The entries cover the main kinds of game knowledge: players and units in StarCraft II (ByuN, Mothership, Liberator), items and equipment in Elden Ring and Minecraft (Radahn's Greaves, Goldmask's Set, the enchantments of a diamond chestplate), places in Tears of the Kingdom (Rito Village, Tabantha Tundra) and skills in Cyberpunk 2077 (Short Circuit, Breach Protocol). The commentator does not repeat every entry. It keeps the Dusk Claymore but not the Dusk Bow, and the Zonaite Sword but not the Strong Zonaite Sword. It sometimes reads the wiki entry instead of commenting on the game, as in "Rosalind Myers is the third and current president…" or "Brightcap, a mushroom that grows in…".

In the other 11 examples no question is asked. StreamDuo then describes the action or reads a name off the screen, such as Elphael, Brace of the Haligtree in Elden Ring. At these seconds Proact-VL names characters, creatures and places from Baldur's Gate 3: Lae'zel and the Mind Flayers in League of Legends, Karlach in Counter-Strike 2, Street Fighter 6, Yu-Gi-Oh! and Tears of the Kingdom, and the Underdark in Black Myth: Wukong. In the examples where StreamDuo asks, Proact-VL at the same seconds names none of the entries and stays vague, as in "Okay, I'll try this one." or "How do we get back down?".

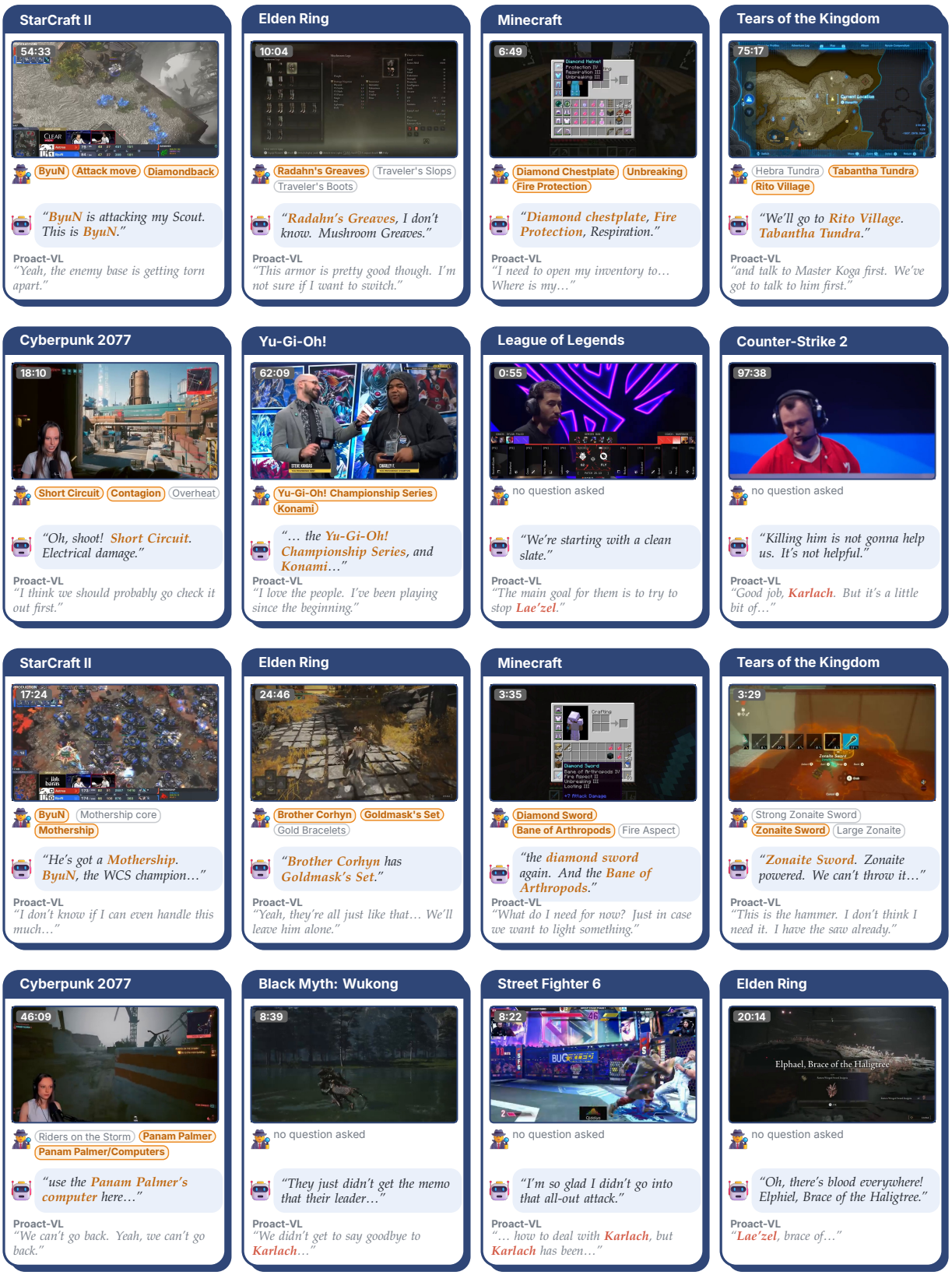


Figure 6 More examples from the ten unseen games, with StreamDuo and Proact-VL at the same seconds (1/2). Chips: the entries in the research agent's answer, in orange those the commentator names. Red: names from Baldur's Gate 3.

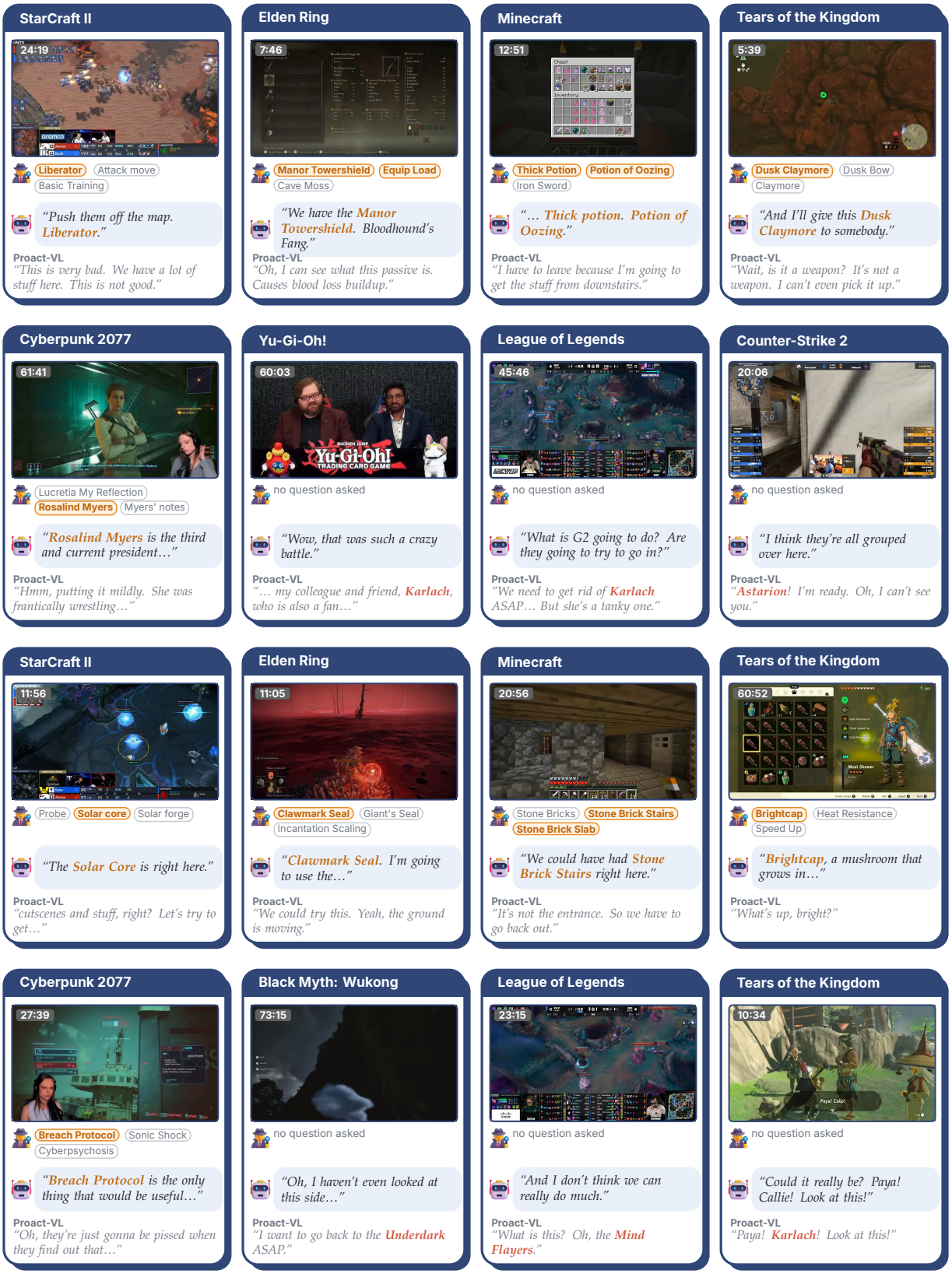


Figure 7 More examples from the ten unseen games (2/2).