BusterX++: Complete Implementation Guide

I'll walk you through this paper section by section, providing all implementation details.

?
--

Paper Overview

Title: BusterX++: Towards Unified Cross-Modal AI-Generated Content Detection and Explanation with MLLM

Authors: Haiquan Wen, Tianxiao Li, Zhenglin Huang, Yiwei He, Guangliang Cheng (University of Liverpool, UK)

Project Page: https://github.com/l8cv/BusterX

arXiv: 2507.14632v2 (July 31, 2025)

1. Abstract

What BusterX++ Does:

Unified cross-modal detection framework for both AI-generated images AND videos

Provides explainable rationales (not just binary classification)

Uses MLLM + Reinforcement Learning (RL) without cold-start

Key Innovations:

Multi-stage Training - 3-stage progressive training

Thinking Reward - External model evaluates reasoning quality

Hybrid Reasoning - Can switch between thinking/non-thinking modes

GenBuster++ Benchmark - 4,000 human-curated images/videos

2. Section 1: Introduction

Problem Statement

Traditional non-MLLM methods (CNNs, etc.) lack interpretability and generalization

Existing MLLM-based methods are single-modality (image OR video only)

Cold-start + RL paradigm limits performance improvement in RL stage

BusterX++ Solution (Figure 1 Framework Comparison)

Method Type

Input

Output

Explanation

Non-MLLM

Image OR Video

REAL/FAKE

? No

MLLM

Image OR Video

REAL/FAKE

? Yes

BusterX++

Image AND Video

REAL/FAKE + Reasoning

? Optional

Three Main Contributions

Unified cross-modal framework achieving SOTA on single-modality benchmarks

Novel RL post-training WITHOUT cold-start

GenBuster++ benchmark (4,000 samples)

3. Section 2: Related Work

2.1 AI-Generated Content Detection Benchmarks

Image Benchmarks Referenced:

GenImage [71]

DMImage [11]

TrueFake [14]

MMTD-SET [64] (from FakeShield)

SID-Set [23] (from SIDA)

So-Fake-Set [24]

Video Benchmarks Referenced:

GVF [32]

GenVideo [8] (DeMamba paper)

GenVidBench [34]

GenBuster-200K [59] (original BusterX paper)

2.2 AI-Generated Content Detection Methods

Non-MLLM Methods: MesoNet, EfficientNet, CNN-based detectors

MLLM-Based Methods:

AntifakePrompt [7]

FakeShield [64]

SIDA [23]

X?-DFD [9]

FakeVLM [60]

LEGION [26]

FakeScope [30]

VLF-FFD [38]

AIGI-Holme [69]

So-Fake-R1 [24]

MM-Det [44]

BusterX [59]

2.3 Multimodal Large Language Models Referenced

GPT-4o [36]

Gemini 2 [47]

Claude 4 [2]

Kimi-k1.5 [48]

Qwen-VL [4] <- Base model used

InternVL [70]

2.4 Post-Training in LLM

Key RL Algorithms:

GRPO (Group Relative Policy Optimization) from DeepSeek-R1 [13]

REINFORCE++ [22]

DAPO (Dynamic sAmpling Policy Optimization) [65] <- Used in BusterX++

GVPO [67]

Key Insight from Literature:

"SFT Memorizes, RL Generalizes" - SFT tends to memorize training data, whereas RL enhances visual reasoning abilities [10, 61]

4. Section 3: Benchmark (GenBuster++)

3.1 Motivation

Limitations of Existing Benchmarks:

Single-modality focus (image OR video)

Limited human curation (contain unrealistic content)

3.2 Benchmark Construction

Total Size: 4,000 samples

1,000 real images

1,000 fake images

1,000 real videos

1,000 fake videos

Data Sources

Real Images/Videos:

Source: OpenVid-1M HD [33]

Pre-filtered for diverse real-world scenarios

Fake Images/Videos - Multiple Sources:

MagicArena (high-rated samples)

Custom Generation Pipeline:

Get social media images via Reddit API

Use Qwen-2.5-VL to generate detailed captions

Use captions as prompts for diffusion models

Image Generators Used (7 total):

Generator	Approximate Count
FLUX 1.1	~400+
SDXL	~200
Recraft	~150
GPT-4o	~100
Imagen 3	~50
Ideogram 3.0	~50

Ideogram 2.0

~50

Video Generators Used (13 total):

Generator

Approximate Count

Wan2.1

~250

HunyuanVideo

~200

Hailuo Video 01

~150

Seedance 1.0

~100

Kling 2.0

~80

Runway Gen-3

~50

Luma Ray2

~50

Pika

~30

PixVerse V4

~30

SkyReels V1

~30

Sora

~20

Veo 2

~5

Veo 3

~5

Scenario Categories (9 total):

Category

Count

Human

1,900

Animal

460

Landscape

300

Food

200

Vehicle

150

Building

150

Object

130

Plant

40

Artwork

20

Data Filtering Process

For Real Images/Videos:

Filter by resolution, frame rate, bitrate

Remove duplicates from same origin

Manual review: remove watermarks, anime, synthetic backgrounds
-> Results in 1,000 real images + 1,000 real videos

For Fake Images/Videos (Novel Two-Stage Pipeline):

Stage 1: Mix real and fake samples -> Experts identify samples
that "appear real"

Stage 2: Re-examine to separate actually synthetic ones ->
Results in 1,000 fake images + 1,000 fake videos

Post-Processing Standardization

Images:

Resolution: 1024?1024

Videos:

Resolution: 1920?1080

Duration: 5 seconds

Frame rate: 24 FPS

Encoding: HEVC with x265

Why Standardize:

Eliminates encoding biases

Ensures consistency across different generators

5. Section 4: Method

4.1 Challenges of Cold-Start

Problem with Traditional Cold-Start + RL:

Most MLLM + RL methods use SFT cold-start before RL

Cold-start accelerates training BUT restricts RL stage improvement

CoT data quality from prompt engineering is unreliable

Human "fakeness" judgments are based on subtle, non-linguistic cues

BusterX++ Solution: Abandon cold-start entirely, start directly with RL

4.2 Multi-Stage Training

Base RL Algorithm: DAPO (Dynamic sAmpling Policy Optimization)

DAPO Objective Function:

J_DAPO(?) = E[1/?|o?| ? ?? ?? min(r?,?(?)???,?, clip(r?,?(?), 1−?_low, 1+?_high)???,?)]

Subject to: 0 < |{o? | is_equivalent(a, o?)}| < G

Where:

G = group size of sampled outputs

r?,?(?) = ??(o?,? | q, o?,<t) / ??_old(o?,? | q, o?,<t) (policy ratio)

??,? = (r? − mean({r?})) / std({r?}) (normalized advantage)

?_low, ?_high = clipping hyperparameters

Stage-1: Foundation RL

Purpose: Build fundamental classification capabilities

What Happens:

Train model on basic classification tasks (real vs fake)

Model receives rewards based on classification accuracy

Learns basic characteristics of real/fake content

Reward Function for Stage-1:

R_stage-1 = r_fmt + r_overlong + r_acc

Stage-2: Thinking Mode Fusion

Purpose: Enable switching between reasoning and non-reasoning modes

Mechanism: SFT with special chat templates (inspired by Qwen3)

SFT Data: ~Several hundred samples collected from Stage-1 model

Chat Templates (Table 1):

Thinking Mode:

<|im_start|>user

{query} /think<|im_end|>

<|im_start|>assistant

<think>

{thinking_content}

</think>

<answer>

{response}

</answer><|im_end|>

Non-Thinking Mode:

<|im_start|>user

{query} /no_think<|im_end|>

<|im_start|>assistant

<think>

</think>

<answer>

{response}

</answer><|im_end|>

Behavior:

Default or /think -> Generate detailed CoT reasoning

/no_think -> Skip reasoning, output answer directly

Note: Ablation shows Stage-2 has minimal impact on final performance but adds flexibility.

Stage-3: Advanced RL

Purpose: Enhance response quality with advanced RL techniques

Key Addition: Thinking Reward from external model

Reward Function for Stage-3:

R_stage-3 = r_fmt + r_overlong + r_acc + r_hybrid + r_think

Training Mix: Uses both /think and /no_think prompts

Observation (Figure 5): CoT length naturally increases with training steps (from ~200 to ~600 tokens over 2k steps)

Important: Applying Thinking Reward too early (Stage-1) destabilizes training!

4.3 Reward Functions

Five Reward Components:

1. Format Reward (r_fmt)

python

if response matches "<think>...</think><answer>...</answer>":

r_fmt = 0

else:

r_fmt = -1

2. Soft Overlong Reward (r_overlong)

python

if L_gen <= L_max - L_cache:

r_overlong = 0

elif L_max - L_cache < L_gen <= L_max:

r_overlong = ((L_max - L_cache) - L_gen) / L_cache # Linear penalty

else: # L_gen > L_max

r_overlong = -1

3. Accuracy Reward (r_acc)

python

if classification_correct:

r_acc = 1

else:

r_acc = 0

4. Hybrid Thinking Reward (r_hybrid)

python

if response follows Table 1 template correctly:

r_hybrid = 0

else: # Skips thinking in /think mode OR thinks in /no_think mode

r_hybrid = -1

5. Thinking Reward (r_think)
---

python

def thinking_reward(response, mode, external_model):

if mode == "/no_think":

return 0

else:

model_score = external_model.evaluate(response) # 0 to 1

return min(r_acc, model_score)

```

Where `0 ? r_think ? 1`

---

## 6. Section 5: Experiments

### Experimental Setup

**Base Model:** `Qwen2.5-VL-7B-Instruct`

**External Thinking Reward Model:**
`SophiaVL-R1-Thinking-Reward-Model-3B`

**Video Sampling:** 16 frames at 4 FPS

**Training Configuration:**

| Parameter | Value |
|---|---|
| Fine-tuning Method | LoRA |
| LoRA Rank | 16 |
| LoRA Alpha | 32 |
| Learning Rate | 1?10?? |
| Precision | bfloat16 |

**Evaluation Metric:** Accuracy (ACC) for each subcategory

---

### 5.1 Performance on Single-Modality Benchmarks

#### Table 2: So-Fake-Set Results (Image Detection)

| Method | Acc | F1 |
|---|---|---|
| CnnSpott | 91.2 | 90.8 |
| DiffForensics | 91.7 | 91.4 |
| DIRE | 91.9 | 91.7 |
```

| HIFI-Net | 88.4 | 83.9 |

| SIDA | 91.9 | 91.5 |

| So-Fake-R1 | 93.2 | 92.9 |

| **BusterX++** | **93.9** | **93.7** |

| BusterX++(/no_think) | 92.3 | 92.1 |

Table 3: GenBuster-200K Results (Video Detection)

| Method | Test ACC | Test F1 | OOD ACC | OOD F1 |

|--------|----------|---------|---------|---------|

| 3D ResNet | 70.6 | 73.5 | 65.6 | 70.6 |

| 3D ResNeXt | 72.6 | 75.5 | 65.1 | 71.0 |

| ViViT | 78.5 | 81.2 | 76.2 | 79.4 |

| VideoMAE | 79.1 | 81.7 | 76.9 | 80.3 |

| DeMamba | 82.0 | 83.9 | 79.3 | 82.0 |

| BusterX | 85.5 | 85.5 | 84.8 | 85.1 |

| **BusterX++** | **88.3** | **88.3** | **92.4** | **92.3** |

| BusterX++(/no_think) | 87.5 | 87.4 | 91.5 | 91.5 |

Key Improvements:

- +0.7% on So-Fake-Set (vs So-Fake-R1)

- +2.8% on GenBuster-200K Test Set (vs BusterX)

- +7.6% on GenBuster-200K OOD Benchmark (vs BusterX)

5.2 Performance on GenBuster++ (Cross-Modal)

Training: Re-train on mixture of image and video data

Table 4: Comparison with Existing MLLMs

| Method | Image Real | Image Fake | Video Real | Video Fake |
Overall |

|--------|------------|------------|------------|------------|---------|

| MiMo-VL-7B-RL | 91.9 | 12.8 | 86.1 | 42.7 | 58.4 |

| InternVL3-8B | 82.1 | 18.4 | 80.1 | 41.3 | 55.5 |

| Keye-VL-8B | 98.3 | 1.8 | 95.7 | 11.8 | 51.9 |

| MiniCPM-o 2.6 | 27.1 | 78.8 | 78.4 | 29.8 | 53.3 |

| Qwen2.5-Omni-7B | 78.7 | 28.6 | 81.4 | 31.4 | 55.0 |

| Qwen2.5-VL-7B | 92.4 | 8.9 | 92.6 | 27.7 | 55.4 |

| BusterX | 79.2 | 54.3 | 86.4 | 53.1 | 68.3 |

| **BusterX++** | **80.4** | **76.2** | **95.3** | **57.9** |
77.5 |

| BusterX++(/no_think) | 80.5 | 74.4 | 96.4 | 55.9 | 76.8 |

Key Insight: Non-thinking mode only drops ~0.7% while saving computation.

5.3 Cold Start vs. Non-Cold Start

Table 5: Training Strategy Comparison

Cold-Start	Stage-1	Stage-3	Img Real	Img Fake	Vid Real	Vid Fake	Overall
?	-	-	72.4	64.7	80.5	51.9	67.4
?	?	-	77.3	67.6	88.1	53.7	71.7
?	?	?	81.0	65.9	91.4	53.2	72.9
-	?	-	78.6	63.4	86.8	48.6	69.4
-	**?**	**?**	**81.2**	**76.7**	**94.1**	**57.5**	**77.4**

Cold-Start Setup: ~1k Chain-of-Thought samples collected

Key Finding:

- Non-cold start initially underperforms (69.4% vs 71.7% after Stage-1)
- Non-cold start significantly outperforms after Stage-3 (77.4% vs 72.9%)
- **Conclusion:** Skip cold-start, go directly to RL!

5.4 Ablation Study

Table 6: Data Modality Ablation

Image Data	Video Data	Img Real	Img Fake	Vid Real	Vid Fake	Overall
?	-	79.0	72.3	81.2	52.9	71.4
-	?	78.7	65.4	92.4	55.0	72.9
?	**?**	**80.4**	**76.2**	**95.3**	**57.9**	**77.5**

Key Finding: Cross-modal training benefits BOTH modalities!

Table 7: Training Strategy Ablation

Stage-1	Stage-2	Stage-3	Img Real	Img Fake	Vid Real	Vid Fake	Overall
?	-	-	78.6	63.4	86.8	48.6	69.4

| ? | ? | - | 78.5 | 63.0 | 87.2 | 48.4 | 69.3 |

| ? | - | ? | 81.2 | 76.7 | 94.1 | 57.5 | 77.4 |

| ? | ? | ? | 80.4 | 76.2 | 95.3 | 57.9 | 77.5 |

Key Finding: Stage-2 has minimal performance impact (˜0.1%) but adds mode-switching flexibility.

5.5 Robustness Study

Table 8: Perturbation Robustness

| JPEG | Noise | Blur | Img Real | Img Fake | Vid Real | Vid Fake | Overall |

|------|-------|------|----------|----------|----------|----------|---------|

| - | - | - | 80.4 | 76.2 | 95.3 | 57.9 | 77.5 |

| ? | - | - | 82.1 | 67.2 | 94.5 | 55.6 | 74.9 |

| - | ? | - | 76.4 | 66.7 | 95.1 | 49.2 | 71.9 |

| - | - | ? | 91.6 | 66.4 | 93.9 | 57.6 | 77.4 |

| ? | ? | ? | 90.8 | 53.5 | 97.0 | 40.8 | 70.5 |

Perturbation Settings:

- JPEG Compression: quality=70

- Gaussian Noise: ?=5

- Gaussian Blur: standard

- Degradation Cascade: inspired by Real-ESRGAN [56]

Key Finding: Model is robust to low-level distortions even without training on degraded data.

5.6 Case Study (Figure 6)

Three Key Observations from BusterX++ Explanations:

1. **Stable Reasoning:** Systematic step-by-step analysis approach

2. **Attention to Low-Level Details:** Spots subtle anomalies (motion blur, lighting inconsistencies)

3. **Advanced Reasoning:** Uses pre-training knowledge (e.g., recognizes Jaguar I-PACE model)

---

7. Section 6: Conclusion and Limitations

Limitations Identified

1. **Generative Technology Adaptation:** Latest generators in GenBuster++ are more challenging than GenBuster-200K generators

2. **Potential Post-training Bottleneck:** May be approaching performance ceiling; future work should explore other training

stages

---

8. Supplementary Material

Section A: Design of Prompts

System Prompt:

```
A conversation between User and Assistant. The user asks a
question,

and the Assistant solves it. The assistant first thinks about
the

reasoning process in the mind and then provides the user with
the answer.

The reasoning process and answer are enclosed within <think>
</think>

and <answer> </answer> tags, respectively, i.e., <think>
reasoning

process here </think><answer> answer here </answer>
```

User Prompt for IMAGES:

```
Please analyze whether there are any inconsistencies or obvious
signs

of forgery in the image, and finally come to a conclusion: Is
this

image real or fake?

Please think about this question as if you were a human
pondering deeply.

Engage in an internal dialogue using expressions such as 'let me
think',

'wait', 'Hmm', 'oh, I see', 'let's break it down', etc, or other
natural

language thought expressions. It's encouraged to include
self-reflection

or verification in the reasoning process.

Then, just answer this MCQ with a single letter:

Q: Is this image real or fake?

Options:

A) real

B) fake
```

User Prompt for VIDEOS:

```

Please analyze whether there are any inconsistencies or obvious
signs

of forgery in the video, and finally come to a conclusion: Is
this

video real or fake?

Please think about this question as if you were a human
pondering deeply.

Engage in an internal dialogue using expressions such as 'let me
think',

'wait', 'Hmm', 'oh, I see', 'let's break it down', etc, or other
natural

language thought expressions. It's encouraged to include
self-reflection

or verification in the reasoning process.

Then, just answer this MCQ with a single letter:

Q: Is this video real or fake?

Options:

A) real

B) fake

```

Section B: Non-Thinking Mode Response

```

<think>

</think>

<answer>

A

</answer>

9. ? Available Resources for Implementation
-------------------------------------------

Models on HuggingFace
---------------------

Model

HuggingFace Link

Purpose

Qwen2.5-VL-7B-Instruct

Qwen/Qwen2.5-VL-7B-Instruct

Base MLLM for fine-tuning
```

SophiaVL-R1-Thinking-Reward-Model-3B

deepseek-ai/SophiaVL-R1-Thinking-Reward-Model-3B

External model for Thinking Reward

Qwen2.5-VL (for captioning)

Qwen/Qwen2.5-VL-72B-Instruct

Caption generation for data pipeline

Datasets

Dataset

Source

Purpose

OpenVid-1M HD

GitHub/HuggingFace

Source for real videos

So-Fake-Set

From So-Fake-R1 paper

Image detection benchmark

GenBuster-200K

From BusterX paper

Video detection benchmark

GenBuster++

https://github.com/l8cv/BusterX

Cross-modal benchmark (to be released)

Code Repositories

Repository

Link

Purpose

BusterX/BusterX++

https://github.com/l8cv/BusterX

Main implementation

ms-swift

https://github.com/modelscope/swift

Training framework used

DAPO

From ByteDance

RL algorithm implementation

Image/Video Generators (for data creation)

Generator

Access

FLUX 1.1

Black Forest Labs API

SDXL

HuggingFace stabilityai/stable-diffusion-xl-base-1.0

GPT-4o

OpenAI API

Seedance 1.0

ByteDance (research access)

SkyReels V1

https://github.com/SkyworkAI/SkyReels-V1

HunyuanVideo

Tencent (open source)

Wan2.1

Alibaba

10. ? Implementation ChecklistStep 1: Environment Setup
---

bash

Clone repository

git clone https://github.com/l8cv/BusterX

cd BusterX

Install dependencies (based on ms-swift)

pip install ms-swift transformers accelerate peft

Step 2: Prepare Base Model
---

python

from transformers import Qwen2VLForConditionalGeneration

model = Qwen2VLForConditionalGeneration.from_pretrained(

"Qwen/Qwen2.5-VL-7B-Instruct",

torch_dtype=torch.bfloat16

)

Step 3: Configure LoRA
---

python

from peft import LoraConfig

lora_config = LoraConfig(

r=16,

lora_alpha=32,

target_modules=["q_proj", "v_proj"], # Adjust based on
architecture

)

Step 4: Prepare Datasets

Download So-Fake-Set, GenBuster-200K for training

Post-process videos: 1920?1080, 5s, 24FPS, HEVC

Post-process images: 1024?1024

Step 5: Implement Reward Functions

python

def compute_reward(response, ground_truth, mode,
thinking_model):

r_fmt = 0 if valid_format(response) else -1

r_overlong = compute_overlong_penalty(response)

r_acc = 1 if extract_answer(response) == ground_truth else 0

if stage == 3:

r_hybrid = compute_hybrid_reward(response, mode)

r_think = compute_thinking_reward(response, mode,
thinking_model)

return r_fmt + r_overlong + r_acc + r_hybrid + r_think

else:

return r_fmt + r_overlong + r_acc

Step 6: Multi-Stage Training

Stage-1: Run DAPO with basic rewards (~X steps)

Stage-2: Collect samples from Stage-1, run SFT for mode fusion

Stage-3: Run DAPO with full rewards including Thinking Reward

Step 7: Evaluation

Test on So-Fake-Set (image)

Test on GenBuster-200K (video)

Test on GenBuster++ (cross-modal)

This comprehensive guide covers everything in the paper. The key innovation is the cold-start-free multi-stage RL training combined with cross-modal data mixing to achieve unified image+video detection with explainable reasoning.

PAPER EXPLAINED

BusterX++: Towards Unified Cross-Modal AI-Generated Content Detection and Explanation with MLLMComplete Detailed Implementation Guide

Paper Metadata and Access Information

This paper is titled "BusterX++: Towards Unified Cross-Modal AI-Generated Content Detection and Explanation with MLLM" and was authored by Haiquan Wen, Tianxiao Li, Zhenglin Huang, Yiwei He, and Guangliang Cheng, all affiliated with the University of Liverpool in the United Kingdom. Guangliang Cheng serves as the corresponding author and can be reached at guangliang.cheng@liverpool.ac.uk. The paper is available on arXiv with identifier 2507.14632v2, with version 2 dated July 31, 2025. The official project page and code repository is hosted at https://github.com/l8cv/BusterX, which contains implementations for both the original BusterX and the newer BusterX++ frameworks.

Abstract: Understanding the Core Problem and Solution

The abstract establishes the context and motivation for this work. Recent advances in generative AI have dramatically improved both image and video synthesis capabilities, which has significantly increased the risk of misinformation through sophisticated fake content being spread across digital platforms. In response to this threat, detection methods have evolved from traditional approaches based on convolutional neural networks and other classical machine learning techniques to more modern approaches based on multimodal large language models, commonly abbreviated as MLLMs. These MLLM-based approaches offer enhanced transparency and interpretability in identifying synthetic media because they can provide textual explanations for their decisions rather than simply outputting a binary classification.

However, the authors identify a fundamental limitation in current detection systems: they remain limited by their single-modality design. This means that existing approaches analyze images or videos separately, which makes them ineffective against synthetic content that might combine multiple media formats or when a unified detection system is needed. To address these challenges, the authors introduce BusterX++, which they describe as a novel framework designed specifically for cross-modal detection and explanation of synthetic media. The term "cross-modal" here means that the same model can process and detect both AI-generated images and AI-generated videos within a unified framework.

The approach incorporates an advanced reinforcement learning post-training strategy that eliminates the need for a cold-start phase. The cold-start phase refers to a common practice in training reasoning models where supervised fine-tuning is performed on chain-of-thought data before beginning reinforcement learning. The authors argue that this cold-start phase can actually limit the performance improvements achievable during the reinforcement learning

stage.

Through three key innovations--Multi-stage Training, Thinking Reward, and Hybrid Reasoning--BusterX++ achieves stable and substantial performance improvements. Multi-stage training refers to dividing the training process into distinct phases with different objectives. Thinking Reward involves using an external model to evaluate the quality of the reasoning process generated by the model. Hybrid Reasoning refers to the model's ability to switch between a thinking mode where it generates detailed chain-of-thought reasoning and a non-thinking mode where it directly outputs classification results without engaging in detailed reasoning.

To enable comprehensive evaluation of cross-modal detection capabilities, the authors also present GenBuster++, which is a cross-modal benchmark that leverages state-of-the-art image and video generation techniques. This benchmark comprises 4,000 images and video clips that have been meticulously curated by human experts using a novel filtering methodology to ensure high quality, diversity, and real-world applicability. The extensive experiments demonstrate the effectiveness and generalizability of the approach.

Section 1: Introduction

The Rise of Generative AI and Its Dual Nature

The introduction begins by discussing recent advancements in generative AI, citing specific examples such as Seedance 1.0, which is referenced as citation [17] in the paper, and GPT-4o, referenced as citation [36]. These technologies have unlocked tremendous potential across various sectors including advertising, education, and entertainment. The authors note that these technologies have revolutionized content creation by enabling the production of highly realistic images and videos.

However, the authors emphasize that this power has a darker side as well. The increasing prevalence of AI-generated content on social media platforms is blurring the lines between authentic and synthetic media. This phenomenon not only challenges our ability to verify content credibility but also raises profound concerns about the integrity of information in the digital age. Because of these concerns, researchers have prioritized AI-Generated content detection, commonly abbreviated as AIGC detection, as a critical area of study. Consequently, the development of robust detection frameworks has become increasingly urgent.

Evolution of Detection Methods: From Non-MLLM to MLLM Approaches

The authors provide a historical perspective on the evolution of detection methods. Initially, non-MLLM methods dominated the field of AI-generated content detection. These methods are referenced through citations [1, 6, 8, 15, 35, 40, 45, 46, 54, 68] and include approaches based on convolutional neural networks, EfficientNet architectures, and other classical deep learning techniques. These methods primarily focused on single-modality detection of either images or videos, but not both within the same framework. While these methods achieved acceptable accuracy within their specific domains, they exhibited significant limitations in generalizing to unseen generation techniques and cross-modal scenarios. Additionally, the lack of interpretability in their decision-making process raised concerns about transparency and trustworthiness in practical applications.

Recently, the focus has shifted to MLLM-based approaches, which are referenced through citations [7, 9, 23, 24, 26, 30, 38, 59, 60, 64, 69]. These approaches have considerably enhanced the transparency and interpretability of detection outcomes in both AI-generated image and video detection fields. The authors cite several notable works in this area: FakeShield [64], SIDA [23], and X?-DFD [9] for image detection, and MM-Det [44] and BusterX [59] for video detection. FakeShield is particularly notable as it was published at ICLR 2025 and introduced the concept of explainable image forgery detection and localization using multimodal large language models.

Limitations of Current MLLM-Based Methods

Despite the progress made by MLLM-based methods, the authors identify two key limitations that their work aims to address.

The first limitation is that current methods are restricted to single-modality image or video detection and have not sufficiently explored the cross-modal capabilities of MLLMs. This means that if you want to detect both fake images and fake videos, you would need to train and deploy two separate models, which is inefficient and fails to leverage the potential for knowledge transfer between these related tasks.

The second limitation relates to the training methodology. Most MLLM combined with reinforcement learning methods, as referenced in citations [24, 59], rely on a resource-intensive cold-start before the reinforcement learning stage. During this cold-start phase, the model is trained using supervised fine-tuning on chain-of-thought data that has been generated through various means such as prompting larger models or human annotation. However, the performance improvement during the subsequent reinforcement learning stage is often restricted, suggesting that the cold-start approach may actually limit the model’s ability to learn better reasoning patterns through reinforcement learning.

The BusterX++ Solution

To overcome these limitations, the authors introduce BusterX++, a novel framework designed for unified cross-modal AI-generated content detection and interpretation. The framework comparison is illustrated in Figure 1 of the paper, which shows three different types of approaches.

Non-MLLM methods can only take either an image or a video as input (indicated by "or" in the figure) and produce only a REAL/FAKE classification without any explanation. MLLM-based methods also take either an image or video as input and produce a REAL/FAKE classification along with an explanation, but they still cannot handle both modalities in a unified way. In contrast, BusterX++ can take both images and videos as input (indicated by "and" in the figure) and produces a reasoning chain followed by a response that includes the REAL/FAKE classification. Importantly, BusterX++ also supports a "skip reasoning" path where it can directly output the classification without generating the full reasoning chain, and the explanation is marked as optional.

The authors leverage the cross-modal capabilities of MLLMs by employing a unified training strategy that incorporates both images and videos during the post-training phase. Their experiments reveal an important finding: joint training across modalities leads to mutually beneficial performance gains, with each modality enhancing the other’s detection accuracy. This

means that training on both images and videos simultaneously actually improves performance on both tasks compared to training on each modality separately.

Additionally, the authors propose a novel cold-start-free post-training strategy that integrates three key components: Multi-stage Training, Thinking Reward, and Hybrid Reasoning. By directly starting with reinforcement learning without the cold-start supervised fine-tuning phase, their model demonstrates superior generalization and adaptability compared to cold-start-dependent methods.

Furthermore, the model offers an optional non-thinking mode, which allows it to directly output classification results without engaging in detailed reasoning. This flexibility not only reduces computational overhead but also enhances the model's practicality in scenarios where rapid classification is prioritized, while still maintaining its advanced reasoning capabilities for cases requiring detailed explanations.

The Need for a New Benchmark

While BusterX++ advances AI-generated content detection technology, the authors note that the lack of a suitable benchmark to comprehensively evaluate its cross-modal capabilities posed a substantial challenge. Although there are some existing datasets and benchmarks in the field, such as So-Fake-Set [24] and GenBuster-200K [59], they still have issues.

The first issue is that existing benchmarks focus on single modalities, such as image or video, but not both together. This makes it impossible to properly evaluate cross-modal detection capabilities. The second issue is that existing benchmarks lack fine-grained human curation and contain some content that is not realistic enough, which can lead to inflated performance metrics that don't reflect real-world performance.

Recognizing this critical gap, the authors created GenBuster++, a cross-modal benchmark designed to meet the demands of modern AI-generated content detection. It leverages state-of-the-art image and video generation techniques and consists of 4,000 images and video clips. Each fake sample undergoes a rigorous two-stage filtering process. In the first stage, experts identify samples that are perceived as real from a mixed set containing both real and fake content. In the second stage, they isolate those samples that are actually synthetic from the ones identified as appearing real in the first stage. This ensures that the benchmark contains only high-quality fake samples that are genuinely difficult to distinguish from real content.

Summary of Contributions

The authors summarize their contributions as three main points.

First, they introduce BusterX++, a unified cross-modal detection and explanation framework designed to identify AI-generated content across both images and videos. In addition to its multimodal capabilities, BusterX++ achieves state-of-the-art performance on single-modality benchmarks, meaning it outperforms existing methods even when evaluated only on images or only on videos.

Second, they adopt a novel reinforcement learning post-training strategy without cold-start, which effectively improves the model's final performance compared to the cold-start-dependent

strategy.

Third, they introduce GenBuster++, a cross-modal high-quality benchmark consisting of 4,000 images and video clips. It provides a reliable standard for cross-modal AI-generated content detection and strongly supports evaluation and application expansion.

Section 2: Related Work

2.1 AI-Generated Content Detection Benchmark

The authors provide a comprehensive review of the evolution of benchmarks in the AI-generated content detection field. They note that the landscape of AI-generated content detection benchmarks has evolved to reflect the growing complexity of synthetic media.

The early benchmarks, referenced through citations [12, 20, 27, 28, 40, 62, 72], focused mainly on GAN-generated facial forgeries. This focus was consistent with the initial emphasis of deepfake technology on the modification of human identities, where techniques like face swapping and face reenactment were the primary concerns. These early datasets typically contained videos of faces that had been manipulated using generative adversarial networks.

With the development of more sophisticated generative models, particularly diffusion models, the focus gradually shifted toward generating diverse and realistic content beyond just faces. In the image domain, benchmarks such as GenImage [71], DMImage [11], TrueFake [14], MMTD-SET [64], SID-Set [23], and So-Fake-Set [24] have significantly expanded the scope of research. GenImage is described as a million-scale benchmark for detecting AI-generated images. DMImage focuses on images generated by diffusion models. TrueFake is described as a real-world case dataset of last generation fake images that are also shared on social networks. MMTD-SET comes from the FakeShield paper and contains multi-modal tamper description data. SID-Set comes from the SIDA paper and focuses on social media image deepfake detection. So-Fake-Set is the largest social media image forgery benchmark available.

Similarly, the video domain has witnessed substantial progress with benchmarks like GVF [32], GenVideo [8], GenVidBench [34], and GenBuster-200K [59]. GVF stands for Generated Video Forensics and focuses on detecting generated videos via frame consistency. GenVideo is described as a million-scale benchmark associated with the DeMamba detector. GenVidBench is described as a challenging benchmark for detecting AI-generated video. GenBuster-200K comes from the original BusterX paper and is the most recent high-resolution video forgery dataset, containing video clips generated by both open-source and commercial models. These benchmarks leverage state-of-the-art video generators to produce high-quality synthetic content that closely mimics real-world scenarios. They have paved the way for more robust and versatile detection frameworks capable of addressing the challenges posed by modern synthetic content.

Despite these significant advancements, the authors note that existing benchmarks have notable limitations. Most are confined to single-modality data, restricting their utility in assessing cross-modal model capabilities. Additionally, most of them lack fine-grained human curation, resulting in inconsistent quality where some samples are obviously fake and provide little challenge for detection systems. GenBuster++ addresses these challenges by integrating both image and video modalities and

leveraging two-stage filtering to ensure that each sample is high-quality, diverse, and relevant to real-world scenarios.

2.2 AI-Generated Content Detection Method

The authors then review the evolution of detection methods themselves. Traditional AI-generated content detection methods are primarily non-MLLM, as referenced through citations [1, 6, 8, 15, 35, 40, 45, 46, 54, 68], and focus on binary classification tasks. These methods typically extract features from images or videos using convolutional neural networks or other architectures and then train a classifier to distinguish between real and fake content. While these methods achieved reasonable accuracy on in-domain data (meaning data generated using the same techniques as the training data), they exhibited limited generalization to unseen generative techniques and lacked interpretability. When a traditional detector says an image is fake, it provides no explanation of why it made that determination.

Recent advancements introduced MLLM-based methods in both AI-generated image and video detection fields. The authors cite several notable examples: AntifakePrompt [7] which uses prompt-tuning for fake image detection, FakeShield [64] which provides explainable image forgery detection and localization, SIDA [23] which focuses on social media image deepfake detection with localization and explanation, X?-DFD [9] which provides explainable and extendable deepfake detection, FakeVLM [60] which spots fake content using large multimodal models, LEGION [26] which learns to ground and explain for synthetic image detection, FakeScope [30] which is a large multimodal expert model for transparent AI-generated image forensics, VLF-FFD [38] which is an MLLM-enhanced face forgery detection solution, AIGI-Holmes [69] which works towards explainable and generalizable AI-generated image detection via multimodal large language models, So-Fake-R1 [24] which benchmarks and explains social media image forgery detection, MM-Det [44] which works on learning multi-modal forgery representation for diffusion generated video detection, and BusterX [59] which is the predecessor to BusterX++ focusing on MLLM-powered AI-generated video forgery detection and explanation. These methods enhanced detection transparency by providing natural language explanations alongside their classifications.

However, the authors emphasize that these approaches remained confined to single-modality inputs, failing to fully leverage the cross-modal capabilities of MLLMs. A multimodal large language model is inherently designed to process multiple types of input, so restricting it to only images or only videos does not take full advantage of its capabilities. To address these limitations, the authors propose BusterX++, which demonstrates superior generalization and adaptability in cross-modal detection and explanation.

2.3 Multimodal Large Language Model

The authors provide context on the state of multimodal large language models. Recent advancements in MLLMs have emphasized enhancing cross-modal reasoning capabilities, meaning the ability to reason about content that involves multiple modalities such as text and images or text and videos.

On the commercial side, models like GPT-4o [36] from OpenAI, Gemini 2 [47] from Google, Claude 4 [2] from Anthropic, and Kimi-k1.5 [48] from Moonshot AI have demonstrated remarkable capabilities in understanding and reasoning about multimodal content. These commercial models represent the current state-of-the-art in terms of general-purpose multimodal

understanding.

On the open-source side, models such as Qwen-VL [4] from Alibaba and InternVL [70] from Shanghai AI Lab stand out as particularly capable. These models allocate additional computational resources to complex reasoning tasks, pushing the limits of existing benchmarks. The community has even introduced extremely challenging benchmarks like Humanity's Last Exam [39] to benchmark the models' limitations, suggesting that standard benchmarks are becoming too easy for the most capable models.

The authors note that BusterX++ further explores the reasoning capabilities of MLLMs in the domain of AI-generated content detection. Rather than using these models for general-purpose tasks, BusterX++ fine-tunes them specifically for the task of detecting and explaining AI-generated content.

2.4 Post-Training in Large Language Model

The authors discuss the post-training phase of large language models, which has advanced rapidly in recent years. The post-training phase refers to additional training that is performed after the initial pretraining, and it typically includes supervised fine-tuning (SFT) and reinforcement learning (RL).

Models such as DeepSeek-R1 [13] and Kimi-k1.5 [48] have demonstrated remarkable reasoning abilities thanks to reinforcement learning. In particular, DeepSeek-R1 introduced Group Relative Policy Optimization (GRPO), which is a variant of the Proximal Policy Optimization (PPO) algorithm referenced as [41]. GRPO effectively enhances the model's reasoning capacity by training it to generate better reasoning chains through a process of sampling multiple outputs and learning from their relative quality.

The community has also made numerous attempts to improve upon GRPO. REINFORCE++ [22] provides a simple and efficient approach for aligning large language models. DAPO [65], which stands for Dynamic sAmpling Policy Optimization, is the algorithm that BusterX++ adopts as its reinforcement learning strategy. GVPO [67] stands for Group Variance Policy Optimization and provides another alternative approach.

Building on this foundation, models like Qwen3 [51] and Seed1.6-Thinking [42] introduced hybrid thinking and multi-stage RL post-training. Hybrid thinking refers to the ability to switch between different modes of reasoning, such as detailed step-by-step thinking versus quick direct answers. Multi-stage post-training refers to dividing the training process into distinct phases with different objectives.

Reward Models, referenced through citations [31, 37, 55], still play a crucial role in guiding and shaping the behavior of reasoning. A reward model is a separate model that is trained to evaluate the quality of outputs generated by the main model. SophiaVL-R1 [16] proposed a thinking reward model that supervises the overall quality of the reasoning content, which is the external model that BusterX++ uses to provide thinking rewards during training.

The authors highlight an important finding from the literature, specifically from citations [10, 61], which indicates that "SFT Memorizes, RL Generalizes." This means that supervised fine-tuning tends to cause the model to memorize the specific patterns in the training data, whereas reinforcement learning enhances the model's visual reasoning abilities in a more general way that transfers better to new situations. While the

cold-start plus RL paradigm has been widely adopted in modern LLM post-training, some research suggests that the cold-start phase may not be essential [61]. The authors therefore explore a novel post-training strategy without cold-start and achieved encouraging results.

Section 3: Benchmark (GenBuster++)3.1 Motivation

The authors explain their motivation for creating a new benchmark. In the field of deepfake detection, research has primarily focused on facial data due to its significant societal impact. Manipulated videos of people's faces can be used for various harmful purposes including fraud, defamation, and political manipulation, which makes face-based deepfake detection a critical research area.

However, with the rapid advancement of generative models, attention has increasingly shifted toward wider AI-generated video content, which poses broader challenges and risks. Modern generative models can create entirely synthetic scenes, objects, animals, and landscapes, not just manipulated faces. This broader scope of synthetic content requires detection systems that can identify AI-generated content regardless of what the content depicts.

Although several datasets and benchmarks have been developed for AI-generated content detection, as referenced through citations [8, 11, 14, 23, 24, 32, 34, 59, 64, 71], they still have limitations. The first limitation is Single-Modality Focus, meaning that existing work predominantly concentrates on images or videos but not both together. This makes it difficult to evaluate detection systems that aim to handle both modalities. The second limitation is Limited Human Curation, meaning that most datasets lack fine-grained human curation and contain some content that is not realistic enough. When a benchmark contains obviously fake samples, detection systems can achieve high accuracy by learning to detect these obvious artifacts, but they may fail on more realistic fake content encountered in real-world scenarios.

To address these limitations, the authors introduce GenBuster++, a cross-modal benchmark for MLLM evaluation. The benchmark is visualized in Figure 2 of the paper, which shows three panels. Panel (a) shows the distribution of video samples across 13 different video generators, with Wan2.1 having the most samples followed by HunyuanVideo and Hailuo Video 01. Panel (b) shows the distribution of image samples across 7 different image generators, with FLUX 1.1 having the most samples. Panel (c) shows a pie chart of the scenario categories, with Human being the largest category at 1,900 samples, followed by Animal at 460 samples, Landscape at 300 samples, Food at 200 samples, Vehicle at 150 samples, Building at 150 samples, Object at 130 samples, Plant at 40 samples, and Artwork at 20 samples.

3.2 Benchmark Construction

The benchmark leverages state-of-the-art image and video generation techniques and consists of 4,000 images and video clips in total. It has two parts: real images and videos from real-world scenarios, and synthetic images and videos that simulate real-world conditions.

Data Sources

The authors first describe how they constructed a data pool for subsequent filtering.

For real images and videos, they sourced a large number of samples from OpenVid-1M HD [33], which is a dataset that covers a diverse range of real-world scenarios. OpenVid-1M HD is described as a large-scale high-quality dataset for text-to-video generation, but it contains real videos that can be used as the real class in a detection benchmark. These samples were carefully pre-filtered to ensure they come from a wide variety of scenes.

For fake images and videos, they sourced samples from multiple sources. The first source was MagicArena, specifically using high-rated samples from that platform. MagicArena appears to be a platform where AI-generated content is rated by users, so high-rated samples would be those that appear most realistic.

Additionally, the authors constructed a custom pipeline for generating fake content. They used Reddit's official API to obtain social media images that cover a wide range of real-world scenarios. These images serve as references for the types of content that should be generated. Then they employed Qwen-2.5-VL [5] to generate detailed captions that describe the content of these source images. Qwen-2.5-VL is a multimodal large language model that can generate textual descriptions of images. These captions then serve as prompts for various diffusion models to generate synthetic content. This approach ensures that the generated fake content covers similar topics and scenarios as real social media content.

The image generators used include FLUX [29], which is a flow-matching based image generation model from Black Forest Labs, and GPT-4o [36] from OpenAI, among others. The full list of 7 image generators shown in Figure 2(b) includes: FLUX 1.1, SDXL (Stable Diffusion XL), Recraft, GPT-4o, Imagen 3 (from Google), Ideogram 3.0, and Ideogram 2.0.

The video generators used include Seedance 1.0 [17], which is a video generation model from ByteDance, and SkyReels V1 [43] from Skywork AI, among others. The full list of 13 video generators shown in Figure 2(a) includes: Runway Gen-3, Hailuo Video 01, HunyuanVideo (from Tencent), Kling 2.0, Luma Ray2, Pika, PixVerse V4, Seedance 1.0, SkyReels V1, Sora (from OpenAI), Veo 2 (from Google), Veo 3, and Wan2.1 (from Alibaba).

Data Filtering

At this stage, the authors introduce a rigorous filtering strategy to ensure that the final samples are of high quality, diverse in content, and closely aligned with real-world scenarios.

For real images and videos, the filtering process involves three steps. First, they filter out a large number of low-quality samples based on objective criteria including resolution, video frame rate, and video bitrate. Samples that fall below certain thresholds for these metrics are removed. Second, they eliminate duplicate content from the same origin clip. This is important because the source dataset might contain multiple clips extracted from the same original video, which would introduce redundancy. Third, they manually check each sample one by one, removing those with extensive watermarks, anime content, or an obviously synthetic background. This manual review ensures that the "real" class truly contains only authentic real-world content. This process results in a final set of 1,000 real images and 1,000 real videos.

For fake images and videos, the authors implement a novel two-stage filtering pipeline that is designed to ensure the

high quality and realism of fake samples. In the first stage, they create a mixed pool that contains both real and fake samples from the previous data pool. Experts carefully review this pool to identify samples that appeared real to them. The experts are not told which samples are real and which are fake; they simply identify samples that look authentic. In the second stage, these identified samples (the ones that appeared real) are re-examined to separate those that were actually synthetic from those that were actually real. The synthetic ones that passed the first stage as appearing real are kept, because these are the challenging fake samples that can fool human experts. This process results in a final set of 1,000 fake images and 1,000 fake videos.

This two-stage filtering approach is crucial for creating a challenging benchmark. By selecting only the fake samples that human experts initially perceived as real, the benchmark ensures that all fake samples are high-quality and realistic. Detection systems that perform well on this benchmark are genuinely capable of detecting realistic fake content, not just obvious fakes.

Post-processing

The authors employ the same post-processing approach as the original BusterX paper to standardize the samples.

For images, they standardize the resolution to 1024?1024 pixels. This ensures that all images have the same dimensions, eliminating any potential shortcuts that a detection system might learn based on resolution differences between real and fake images.

For video clips, they standardize multiple parameters: the resolution is set to 1920?1080 (Full HD), the duration is set to 5 seconds, and the frame rate is set to 24 frames per second. All videos are encoded using HEVC (H.265) encoding with the x265 encoder.

Figure 3 in the paper shows visual examples from GenBuster++, displaying a grid of 8 images showing diverse content including a cat on a rooftop with cherry blossoms, an elderly couple, a woman smiling, and various other scenes.

The authors explain that this unified post-processing approach offers several benefits. First, it eliminates encoding biases. By using HEVC encoding with x265 for all content, they eliminate potential biases that could arise from different encoding preferences used by different generators. Some generators might use specific codecs or compression settings that could serve as shortcuts for detection. Second, it ensures consistency across sources. Standardization ensures consistency across videos or images generated by different models, which may have varying original resolutions and frame rates. Without this standardization, a detection system might learn to identify fake content based on these superficial characteristics rather than the actual visual artifacts of AI generation.

Section 4: Method

This section introduces BusterX++, which the authors describe as a novel framework for detecting AI-generated images and videos with detailed explanations. The framework is visualized in Figure 4 of the paper, which shows the overall architecture.

The figure shows that the framework takes visual input (either

an image or video) along with a text prompt and feeds them to a base model. The training proceeds through three stages: Stage-1 Foundation RL, Stage-2 Thinking Mode Fusion, and Stage-3 Advanced RL. During training, a policy model generates multiple outputs with thinking tags, which are then evaluated by reward functions (R1, R2, R3, ... Rn) and a Thinking Reward Model. These rewards are combined to update the policy model.

The final BusterX++ model can analyze both videos and images. For a video, an example output shows: "<think>\nLet's analyze this video step by step to determine if it is real or fake: 1. Setting and Environment: The setting appears to be a traditional East Asian architectural style, possibly Chinese or Japanese, given the tiled roof and the style of the roof tiles. However, ..." and continues with analysis of the subject (a cat) noting that "The cat itself looks very realistic at first glance, but..." For an image, the output follows a similar pattern with step-by-step analysis.

The authors train a reasoning MLLM with Chain-of-Thought (CoT) [58] and reinforcement learning (RL). Chain-of-Thought refers to the practice of having the model generate its reasoning process before producing a final answer, which has been shown to improve performance on complex reasoning tasks. They expand the training data modality by mixing images and videos during post-training and refine the post-training strategy to boost the model's final performance.

4.1 Challenges of Cold-Start

The authors begin by discussing the challenges associated with the traditional cold-start approach to training reasoning models. Existing MLLM combined with RL methods, as exemplified by citations [24, 59] which correspond to So-Fake-R1 and the original BusterX, mostly depend on a supervised fine-tuning cold-start stage before reinforcement learning. During this cold-start phase, chain-of-thought data is collected, either by prompting larger models, having humans write reasoning chains, or using other methods, and the model is trained to imitate these reasoning chains using standard supervised learning.

However, despite accelerating training, the performance improvement in the RL stage is often restricted when using cold-start. The authors conjecture that this limitation may result from the reasoning quality of CoT data used in the cold-start phase, which could undermine the model's reasoning ability. If the cold-start data contains suboptimal reasoning patterns, the model learns to produce similar suboptimal patterns, and subsequent RL training may not be able to fully correct these learned patterns.

The authors provide a deeper explanation for why generating high-quality CoT data is difficult for this specific task. Human judgments about the "fakeness" of images or videos are often based on subtle, intuitive, and multi-dimensional cues such as unnatural reflections, inconsistent lighting, slight motion artifacts, or uncanny valley effects. These judgments involve complex perceptual processes that are difficult to articulate in language.

Given the non-linear and non-linguistic nature of cross-modal preferences in this task, it is extremely challenging to precisely elaborate a linear thinking chain for "why it is fake." This point is supported by citation [25], which discusses the challenges of chain-of-thought reasoning for improving multimodal large language models. When a human expert looks at a fake image and determines it is fake, they may not be able to fully explain all the subtle cues that contributed to

their judgment. Attempting to capture these judgments in explicit reasoning chains inevitably loses some of the nuance.

Thus, generating CoT data from MLLMs with prompt engineering is not only difficult to guarantee quality but also potentially degrades the model's reasoning performance. The model may learn to produce verbose but ultimately unhelpful reasoning, or it may learn to focus on the wrong features based on what was salient in the generated CoT data. Consequently, the authors abandon the cold start entirely and achieve promising results by starting directly with reinforcement learning.

4.2 Multi-Stage Training

The authors employ a multi-stage training strategy, setting different learning objectives for each stage to stabilize the training process. Rather than trying to achieve all objectives in a single training phase, they break down the training into stages that progressively build up the model's capabilities.

As with the original BusterX, they adopt Dynamic sAmpling Policy Optimization (DAPO) [65] as their reinforcement learning strategy. DAPO is a variant of policy gradient methods that has been shown to be effective for training language models.

The authors provide the mathematical formulation of DAPO. DAPO samples a group of outputs {o?}???? for each question q from the old policy ?????. In other words, for each input question, the model generates G different candidate outputs by sampling from its current policy. The reward model is then used to score the outputs, yielding {r?}???? correspondingly. Each output receives a scalar reward indicating its quality. Then DAPO optimizes the policy model ?? by maximizing the following objective:

The DAPO objective J_DAPO(?) is defined as the expected value over the data distribution D and the sampled outputs, of a normalized sum over all outputs and all tokens. For each token, the objective is the minimum of two terms: the first term is the product of the importance ratio r?,?(?) and the advantage ??,?, and the second term is the clipped version where the importance ratio is clipped to be between 1 - ???w and 1 + ?????, multiplied by the advantage. This clipping mechanism, inherited from PPO, prevents the policy from changing too dramatically in a single update.

There is also a constraint that ensures meaningful learning: 0 < |{o? | is_equivalent(a, o?)}| < G. This constraint requires that among the G sampled outputs, at least one should be equivalent to the correct answer (so there is something positive to learn from) but not all of them should be correct (so there is room for improvement).

The hyperparameters ???w and ????? control the clipping bounds. The importance ratio r?,?(?) is defined as the ratio of the probability of generating token o?,? under the current policy ?? to the probability under the old policy ?????, conditioned on the question q and the previous tokens o?,<?.

The advantage ??,? is computed as the normalized reward: the reward r? minus the mean of all rewards in the group, divided by the standard deviation of the rewards in the group. This normalization ensures that the advantage has zero mean and unit variance within each group, which helps stabilize training.

Stage-1: Foundation RL

In this stage, the focus is on building the fundamental

capabilities of the model. The authors use RL to train the model on basic classification tasks. The model learns to distinguish between real and AI-generated content by receiving rewards based on the accuracy of its classifications. This helps the model quickly grasp the basic characteristics and features of different types of data, laying a solid foundation for subsequent training.

At this stage, the model is not yet expected to produce high-quality reasoning or to switch between thinking modes. The goal is simply to learn to make accurate classifications. The reward function used in Stage-1 is:

R_stage-1 = r_fmt + r_overlong + r_acc

This combines rewards for correct formatting, penalties for overly long responses, and rewards for correct classification accuracy.

Stage-2: Thinking Mode Fusion

This stage involves supervised fine-tuning to enable switching between reasoning and direct answering modes. The authors introduce a Thinking Mode Fusion mechanism inspired by Qwen3 [51], which introduced the concept of hybrid thinking modes in language models.

For the SFT data, the authors collect several hundred samples from the Stage-1 model. These samples are formatted according to the templates shown in Table 1 of the paper.

In Thinking Mode, the chat template is structured as follows: The user message starts with <|im_start|>user, followed by the query and the /think instruction, then <|im_end|>. The assistant response starts with <|im_start|>assistant, followed by <think>, then the thinking content, then </think>, then <answer>, then the response, then </answer>, and finally <|im_end|>.

In Non-Thinking Mode, the structure is similar but with the /no_think instruction instead of /think, and crucially, the <think> and </think> tags are present but empty (no thinking content between them). The response goes directly in the <answer> tags.

The model is trained to generate detailed CoT explanations when there is no specific instruction or when a /think prompt is provided, and to directly output answers when a /no_think prompt is encountered. This flexibility allows the model to adapt to different scenarios. In some cases, users may want detailed explanations to understand why content was classified as fake. In other cases, users may need rapid classification without the overhead of generating reasoning.

Compared to thinking mode which often uses hundreds of tokens for the reasoning chain, non-thinking mode can produce answers with minimal tokens, saving computational resources while maintaining classification accuracy.

The ablation study presented later in the paper shows that this stage has minimal impact on final performance in terms of accuracy. However, it provides the important capability of mode switching, which is valuable for practical deployment.

Stage-3: Advanced RL

In the final stage, the authors further enhance the model’s

response quality using advanced RL techniques. The model continues to receive rewards for accurate classifications and well-formatted responses, but additional reward components are introduced.

They employ a mix of /think and /no_think prompts during training, which reinforces the model's ability to switch between the two modes appropriately. The model learns to generate detailed reasoning when prompted to think and to skip reasoning when prompted not to think.

Additionally, the authors introduce a Thinking Reward provided by an external model. This reward evaluates the quality of the model's thinking process, replacing the Length Reward used in the original BusterX. The Length Reward in BusterX simply encouraged longer reasoning chains, based on the assumption that longer reasoning is better. However, this can lead to verbose but unhelpful reasoning. The Thinking Reward instead uses a separate model that has been trained to evaluate reasoning quality, providing a more nuanced signal about whether the reasoning is actually good.

Figure 5 in the paper shows a graph of CoT Length (measured in tokens) plotted against training steps. The graph shows that CoT length increases naturally with training, starting from around 200 tokens at step 500 and increasing to around 600 tokens by step 2000, with considerable variance. This demonstrates that even without explicitly rewarding length, the model naturally learns to produce longer and more detailed reasoning as training progresses. The Thinking Reward encourages quality reasoning, and quality reasoning tends to be more detailed.

The authors note an important finding: applying the Thinking Reward too early, such as in Stage-1, can destabilize training. This underscores the importance of their multi-stage approach. The model first needs to learn basic classification (Stage-1), then learn mode switching (Stage-2), and only then can it benefit from the more nuanced Thinking Reward signal (Stage-3).

The reward function for Stage-3 is:

R_stage-3 = R_stage-1 + r_hybrid + r_think

This adds the hybrid thinking reward and the thinking reward to the Stage-1 rewards.

4.3 Reward Functions

The authors design several reward functions and employ different combinations of them in each RL stage. They describe five reward components in detail.

Format Reward (r_fmt)

The Format reward ensures that the model's response adheres to the specified format. If the model response contains the proper structure with <think>...</think> and <answer>...</answer> tags, it receives a reward of r_fmt = 0. If the response does not follow this format, it receives a penalty of r_fmt = -1. This reward is necessary because the downstream parsing of the model's output depends on these tags being present. If the model outputs a response without the proper tags, it cannot be properly processed.

Soft Overlong Reward (r_overlong)

The Soft Overlong reward penalizes responses that exceed the maximum allowed length, but does so in a gradual manner rather than a hard cutoff. The formula is:

r_overlong = 0, if L_gen ? L_max − L_cache r_overlong = ((L_max − L_cache) − L_gen) / L_cache, if L_max − L_cache < L_gen ? L_max r_overlong = −1, if L_max < L_gen

Here, L_gen is the length of the generated response, L_max is the maximum allowed length, and L_cache is a buffer parameter that defines a transition zone. If the generated length is well below the maximum (within the cache buffer), there is no penalty. If the generated length is in the transition zone (between L_max − L_cache and L_max), there is a linear penalty that increases as the length approaches the maximum. If the generated length exceeds the maximum, the full penalty of −1 is applied. This soft penalty allows the model to learn to control its output length without being too harshly penalized for occasionally going slightly over.

Accuracy Reward (r_acc)

The Accuracy reward is the most straightforward: if the model classifies correctly (predicting real for real content or fake for fake content), it receives a reward of r_acc = 1. If the classification is incorrect, the reward is r_acc = 0. This directly incentivizes the model to make correct predictions.

Hybrid Thinking Reward (r_hybrid)

The Hybrid Thinking reward ensures that the model correctly follows the mode indicated by the prompt. If the response adheres to the template shown in Table 1, meaning the model thinks when asked to think and doesn’t think when asked not to think, it receives r_hybrid = 0. If the model skips thinking when in Thinking Mode (when the prompt is /think or no specific instruction) or thinks when in Non-Thinking Mode (when the prompt is /no_think), it receives a penalty of r_hybrid = −1. This reward trains the model to respect the mode switching instructions.

Thinking Reward (r_think)

The Thinking Reward evaluates the quality of the reasoning content using an external model M. The formula is:

r_think = 0, if in /no_think mode r_think = min(r_acc, M(y_res)), otherwise

Where M(y_res) is the score assigned by the external model to the response y_res, and this score is between 0 and 1.

This formulation has several important properties. First, in non-thinking mode, the thinking reward is always 0 because there is no thinking content to evaluate. Second, the thinking reward is bounded by the accuracy reward. If the model gets the classification wrong (r_acc = 0), then the thinking reward is also 0 regardless of how good the reasoning appears. This prevents the model from being rewarded for producing plausible-sounding but ultimately incorrect reasoning. Only when the model makes a correct prediction can it receive a positive thinking reward, and even then, the reward is limited by the external model’s assessment of the reasoning quality.

Total Reward by Stage

The total reward for each RL stage is computed as follows:

For Stage-1: R_stage-1 = r_fmt + r_overlong + r_acc

This includes only the basic rewards for formatting, length control, and accuracy.

For Stage-3: R_stage-3 = R_stage-1 + r_hybrid + r_think

This adds the hybrid thinking reward and the thinking reward to the Stage-1 rewards.

Note that Stage-2 uses supervised fine-tuning rather than reinforcement learning, so it does not use the reward functions.

Section 5: Experiments

Experimental Setup

The authors provide detailed information about their experimental setup.

They adopt Qwen2.5-VL-7B-Instruct [4] as their base model. This is a 7 billion parameter multimodal large language model from Alibaba's Qwen team that has been instruction-tuned for following user instructions. The model is available on HuggingFace at Qwen/Qwen2.5-VL-7B-Instruct.

For the external model used to compute the Thinking Reward, they use SophiaVL-R1-Thinking-Reward-Model-3B [16]. This is a 3 billion parameter model that has been trained to evaluate the quality of reasoning content in multimodal contexts.

For video-level detection, they sample 16 frames at a rate of 4 FPS from each video. This means they extract frames at 4 frames per second and take 16 of them, covering 4 seconds of video content. These frames are then processed by the multimodal model as a sequence of images.

For model training, they employ LoRA [21] Parameter-Efficient Fine-Tuning. LoRA (Low-Rank Adaptation) is a technique that adds small trainable matrices to the frozen pretrained weights, allowing efficient fine-tuning without updating all model parameters. They use a rank of 16 and an alpha of 32 for the LoRA configuration. The rank determines the dimensionality of the low-rank matrices, and alpha is a scaling factor.

The model is trained using a learning rate of 1 ? 10?? with bfloat16 precision to optimize computational efficiency. bfloat16 is a 16-bit floating point format that maintains the same exponent range as 32-bit floats while reducing precision, allowing for faster computation and reduced memory usage.

They evaluate using accuracy (ACC) for each subcategory in GenBuster++, reporting separate accuracies for real images, fake images, real videos, and fake videos, as well as overall accuracy.

5.1 Performance on Single-Modality Benchmarks

The authors begin by validating the effectiveness of BusterX++ on two of the latest high-quality single-modality benchmarks. This demonstrates that the cross-modal approach does not

sacrifice performance on individual modalities.

Results on So-Fake-Set (Image Detection)

So-Fake-Set [24] is described as the largest social media image forgery benchmark. It contains images from social media that have been either real or generated/manipulated using various AI tools.

Table 2 in the paper shows the performance comparison. The baseline methods include CnnSpott [54] which achieves 91.2% accuracy and 90.8% F1 score, DiffForensics [66] which achieves 91.7% accuracy and 91.4% F1 score, DIRE [57] which achieves 91.9% accuracy and 91.7% F1 score, HIFI-Net [18] which achieves 88.4% accuracy and 83.9% F1 score, SIDA [23] which achieves 91.9% accuracy and 91.5% F1 score, and So-Fake-R1 [24] which achieves 93.2% accuracy and 92.9% F1 score.

BusterX++ achieves 93.9% accuracy and 93.7% F1 score, outperforming all baselines. In non-thinking mode, BusterX++ achieves 92.3% accuracy and 92.1% F1 score, which is lower than thinking mode but still competitive with the baselines.

The improvement of 0.7 percentage points over So-Fake-R1 (93.9% vs 93.2%) demonstrates that the cold-start-free approach can achieve better results than the cold-start approach used by So-Fake-R1.

Results on GenBuster-200K (Video Detection)

GenBuster-200K [59] is described as the most recent high-resolution video forgery dataset, containing video clips generated by both open-source and commercial models. It includes a standard test set and an out-of-distribution (OOD) benchmark that tests generalization to unseen generators.

Table 3 in the paper shows the performance comparison. The baseline methods include 3D ResNet [19] which achieves 70.6% accuracy and 73.5% F1 on the test set and 65.6% accuracy and 70.6% F1 on the OOD benchmark, 3D ResNeXt [19] which achieves 72.6% and 75.5% on the test set and 65.1% and 71.0% on OOD, ViViT [3] which achieves 78.5% and 81.2% on the test set and 76.2% and 79.4% on OOD, VideoMAE [53] which achieves 79.1% and 81.7% on the test set and 76.9% and 80.3% on OOD, DeMamba [8] which achieves 82.0% and 83.9% on the test set and 79.3% and 82.0% on OOD, and the original BusterX [59] which achieves 85.5% and 85.5% on the test set and 84.8% and 85.1% on OOD.

BusterX++ achieves 88.3% accuracy and 88.3% F1 on the test set and 92.4% accuracy and 92.3% F1 on the OOD benchmark. In non-thinking mode, it achieves 87.5% and 87.4% on the test set and 91.5% and 91.5% on OOD.

The improvement is substantial: 2.8 percentage points over BusterX on the test set (88.3% vs 85.5%) and a remarkable 7.6 percentage points on the OOD benchmark (92.4% vs 84.8%). The large improvement on the OOD benchmark demonstrates that the cold-start-free approach with thinking rewards leads to better generalization to unseen generators.

The authors note that their approach surpasses the MLLM baselines BusterX and So-Fake-R1, which both employ a standard cold-start plus RL method. BusterX++ starts directly with reinforcement learning, achieving 0.6% improvement in accuracy on So-Fake-Set and 2.8% on GenBuster-200K Test Set, while retaining stronger zero-shot robustness with a 7.6% improvement on the GenBuster-200K out-of-domain benchmark.

5.2 Performance on GenBuster++ (Cross-Modal)

To assess cross-modal capabilities, the authors re-train the model on a mixture of image and video data from the aforementioned datasets. This unified training allows the model to learn from both modalities simultaneously.

Table 4 in the paper shows the comparison with existing MLLMs on the GenBuster++ benchmark. The results are broken down by image real, image fake, video real, video fake, and overall accuracy.

MiMo-VL-7B-RL [52] achieves 91.9% on image real, 12.8% on image fake, 86.1% on video real, 42.7% on video fake, and 58.4% overall. This model performs well on real content but poorly on fake content, suggesting it has a strong bias toward predicting "real."

InternVL3-8B [70] achieves 82.1% on image real, 18.4% on image fake, 80.1% on video real, 41.3% on video fake, and 55.5% overall. Similar pattern of poor performance on fake content.

Keye-VL-8B [49] achieves 98.3% on image real, 1.8% on image fake, 95.7% on video real, 11.8% on video fake, and 51.9% overall. This model has an extreme bias toward predicting real, achieving near-perfect scores on real content but essentially failing completely on fake content.

MiniCPM-o 2.6 [50] achieves 27.1% on image real, 78.8% on image fake, 78.4% on video real, 29.8% on video fake, and 53.3% overall. This model shows the opposite pattern with relatively better fake detection but poor real detection.

Qwen2.5-Omni-7B [63] achieves 78.7% on image real, 28.6% on image fake, 81.4% on video real, 31.4% on video fake, and 55.0% overall.

The base model Qwen2.5-VL-7B [4] achieves 92.4% on image real, 8.9% on image fake, 92.6% on video real, 27.7% on video fake, and 55.4% overall. Without fine-tuning for this task, the model struggles significantly with fake content detection.

The original BusterX [59] achieves 79.2% on image real, 54.3% on image fake, 86.4% on video real, 53.1% on video fake, and 68.3% overall. This shows more balanced performance but still has room for improvement.

BusterX++ achieves 80.4% on image real, 76.2% on image fake, 95.3% on video real, 57.9% on video fake, and 77.5% overall. This represents a significant improvement of 9.2 percentage points over BusterX (77.5% vs 68.3%).

In non-thinking mode, BusterX++ achieves 80.5% on image real, 74.4% on image fake, 96.4% on video real, 55.9% on video fake, and 76.8% overall.

The authors highlight several key findings. Compared to BusterX, BusterX++ shows enhanced adaptability in handling diverse data modalities. This indicates the model's strong cross-modal generalization ability. Moreover, in the Non-Thinking Mode, BusterX++ retains virtually the same performance as the Thinking Mode with only approximately a 0.7% drop (77.5% vs 76.8%), underscoring its flexibility and practical utility.

5.3 Cold Start vs. Non-Cold Start

The authors compare the traditional cold-start training strategy with their proposed non-cold start approach on GenBuster++. This

ablation study directly demonstrates the benefit of removing the cold-start phase.

To establish the cold-start baseline, they collect about 1,000 Chain-of-Thought samples using prompt engineering with the base model. These samples are used for supervised fine-tuning in the cold-start phase. They skip Stage-2 in this comparison as it does not significantly affect performance.

Table 5 in the paper shows the results. With cold-start and no subsequent stages, the model achieves 72.4% on image real, 64.7% on image fake, 80.5% on video real, 51.9% on video fake, and 67.4% overall. With cold-start plus Stage-1 (Foundation RL), the model improves to 77.3%, 67.6%, 88.1%, 53.7%, and 71.7% overall. With cold-start plus Stage-1 plus Stage-3 (Advanced RL), the model achieves 81.0%, 65.9%, 91.4%, 53.2%, and 72.9% overall.

Without cold-start and with only Stage-1, the model achieves 78.6%, 63.4%, 86.8%, 48.6%, and 69.4% overall. Without cold-start and with Stage-1 plus Stage-3, the model achieves 81.2%, 76.7%, 94.1%, 57.5%, and 77.4% overall.

The authors note an important pattern: although the non-cold start strategy initially underperforms (69.4% vs 71.7%) compared to the cold-start method after Stage-1 alone, it demonstrates significantly superior performance (77.4% vs 72.9%) after Stage-3 training. The difference of 4.5 percentage points is substantial and shows that the RL stage is able to achieve much larger improvements when not constrained by patterns learned during cold-start.

By directly starting with RL and incorporating multi-stage training, the model achieves better generalization and adaptability. The results highlight the effectiveness of the training strategy in enhancing the model's final performance. The key insight is that cold-start may provide a faster initial improvement but limits the ceiling that can be reached through RL, whereas starting directly with RL has a higher ceiling even if initial progress is slower.

5.4 Ablation Study

The authors conduct ablation studies to understand the contribution of different components of their approach.

Data Modality Ablation

Table 6 in the paper compares the performance of models trained on single-modality data with those trained on cross-modal data.

Training on image data only achieves 79.0% on image real, 72.3% on image fake, 81.2% on video real, 52.9% on video fake, and 71.4% overall. Training on video data only achieves 78.7% on image real, 65.4% on image fake, 92.4% on video real, 55.0% on video fake, and 72.9% overall. Training on both image and video data achieves 80.4% on image real, 76.2% on image fake, 95.3% on video real, 57.9% on video fake, and 77.5% overall.

The key finding is that single-modality trained models still underperform the cross-modal approach even within their own modality. For example, the image-only model achieves 79.0% on image real and 72.3% on image fake, but the cross-modal model achieves 80.4% on image real and 76.2% on image fake. Similarly, the video-only model achieves 92.4% on video real and 55.0% on video fake, but the cross-modal model achieves 95.3% on video real and 57.9% on video fake.

This demonstrates that training on both modalities simultaneously provides a synergistic benefit where each modality helps improve performance on the other. The model likely learns more robust features that transfer across modalities.

Training Strategy Ablation

Table 7 in the paper shows the ablation study for the training strategy.

With only Stage-1, the model achieves 78.6% on image real, 63.4% on image fake, 86.8% on video real, 48.6% on video fake, and 69.4% overall. With Stage-1 plus Stage-2 (Thinking Mode Fusion), the model achieves 78.5%, 63.0%, 87.2%, 48.4%, and 69.3% overall. With Stage-1 plus Stage-3 (Advanced RL, skipping Stage-2), the model achieves 81.2%, 76.7%, 94.1%, 57.5%, and 77.4% overall. With all three stages, the model achieves 80.4%, 76.2%, 95.3%, 57.9%, and 77.5% overall.

Each training stage demonstrates stable performance improvement, with the full pipeline ultimately achieving superior results. Importantly, Stage-2 has minimal impact on the final performance (69.3% vs 69.4% when added after Stage-1, and 77.5% vs 77.4% when added between Stage-1 and Stage-3). This is as expected since Stage-2 is primarily for adding the mode-switching capability rather than improving detection accuracy.

These results validate the effectiveness of the comprehensive training approach in enhancing the model's cross-modal capabilities and overall performance.

5.5 Robustness Study

The authors further assess the robustness of BusterX++ against common visual perturbations to simulate real-world scenarios. In real-world deployment, images and videos often undergo various transformations such as compression, noise addition, and blurring, either intentionally or as a side effect of transmission and storage. A robust detector should maintain performance under these conditions.

They test four types of perturbations: JPEG Compression with quality parameter set to 70 (moderate compression), Gaussian Noise with standard deviation ?=5, Gaussian Blur, and a Degradation Cascade inspired by Real-ESRGAN [56] that combines multiple degradation types.

Table 8 in the paper shows the robustness evaluation results.

Without any perturbation, the model achieves 80.4% on image real, 76.2% on image fake, 95.3% on video real, 57.9% on video fake, and 77.5% overall.

With JPEG compression only, the model achieves 82.1%, 67.2%, 94.5%, 55.6%, and 74.9% overall. The drop in fake image detection (76.2% to 67.2%) is notable, possibly because compression removes some of the subtle artifacts that indicate AI generation.

With Gaussian noise only, the model achieves 76.4%, 66.7%, 95.1%, 49.2%, and 71.9% overall. Noise affects all categories, with the largest impact on video fake detection (57.9% to 49.2%).

With Gaussian blur only, the model achieves 91.6%, 66.4%, 93.9%, 57.6%, and 77.4% overall. Interestingly, blur increases performance on image real (80.4% to 91.6%) while decreasing fake detection. The overall accuracy is nearly unchanged (77.5% to 77.4%), suggesting that blur has only a marginal effect on overall performance even though it produces a mild decline within sub-categories.

With all three perturbations combined (the degradation cascade), the model achieves 90.8%, 53.5%, 97.0%, 40.8%, and 70.5% overall. This is the harshest condition, and the model shows the largest performance drop, particularly on fake content detection.

The authors emphasize that BusterX++ shows remarkable stability against these low-level distortions, even without specific training on degraded data. The model was trained only on clean data but still maintains reasonable performance under various degradations. BusterX++ still delivers solid performance under the harshest degradation cascade, which highlights its robustness and practical value in diverse and unpredictable real-world settings.

5.6 Case Study

The authors provide a detailed case study in Figure 6 of the paper to showcase the model's performance in distinguishing between real and fake content. The figure shows four example cases with the model's explanations.

Through their analysis, the authors identify three interesting findings from BusterX++'s explanations:

First, Stable Reasoning: The model demonstrates a systematic approach to analyzing visual content. It follows a consistent pattern of examining different aspects of the content such as lighting and shadows, perspective and depth, vegetation and texture, structure details, and overall composition. This step-by-step reasoning allows for a thorough understanding of the content's authenticity.

Second, Attention to Low-Level Details: The model pays close attention to details that are easily overlooked by humans. It can spot subtle anomalies like unnatural motion blur, irregularities in lighting and shadows, overly uniform textures, and edges that are too sharp or too clean. These low-level details are crucial for determining authenticity because they often reveal the artifacts of AI generation.

Third, Advanced Reasoning: The model begins to demonstrate advanced reasoning by activating the pre-training knowledge stored in the base MLLM. For example, in one case study involving a car video, the model recognizes that "The design of the car matches known models of Jaguar, specifically the I-PACE, which is a modern electric vehicle. The features such as the LED headlights, the grille shape, and the overall body style are consistent with actual production models." This use of external knowledge helps the model determine whether objects in the content are consistent with the real world.

Section 6: Conclusion and Limitations

In this paper, the authors present BusterX++, a novel framework for unified cross-modal AI-generated content detection and explanation. To support evaluation and application expansion, they introduce GenBuster++, a cross-modal benchmark that leverages state-of-the-art image and video generation

techniques. By leveraging Multi-stage Training, Thinking Reward, and Hybrid Reasoning, BusterX++ sets new state-of-the-art results in GenBuster++ and other latest benchmarks.

Despite promising progress, the authors recognize certain limitations and aim to address them through future optimization.

The first limitation is Generative Technology Adaptation. While their framework demonstrates effectiveness in detecting fake samples from GenBuster-200K (the older dataset), they observe that samples generated with the latest technologies in GenBuster++ present a more significant challenge. The performance gap indicates that the rapid evolution of generative methods necessitates continuous adaptation and enhancement of detection frameworks to maintain efficacy. As new generators become available, detection systems must be updated to handle their outputs.

The second limitation is Potential Bottleneck in Post-training. The post-training phase may be approaching a performance bottleneck in this task. The improvements from Stage-1 to Stage-3 are substantial, but there may be limits to how much further improvement can be achieved through post-training alone. To further advance the capabilities of MLLM for this task, future work should explore other training stages, possibly including improvements to the base model pretraining or novel training paradigms.

Supplementary Material

The supplementary material provides additional details that are essential for reproducibility.

Section A: Design of Prompts

The authors provide the exact prompts used in their system.

The System Prompt is: "A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think><answer> answer here </answer>"

This system prompt establishes the format that the model should follow, with explicit thinking enclosed in tags followed by the answer.

The User Prompt for images is: "Please analyze whether there are any inconsistencies or obvious signs of forgery in the image, and finally come to a conclusion: Is this image real or fake? Please think about this question as if you were a human pondering deeply. Engage in an internal dialogue using expressions such as 'let me think', 'wait', 'Hmm', 'oh, I see', 'let's break it down', etc, or other natural language thought expressions. It's encouraged to include self-reflection or verification in the reasoning process. Then, just answer this MCQ with a single letter: Q: Is this image real or fake? Options: A) real B) fake"

The User Prompt for videos follows the same structure but replaces "image" with "video" throughout.

These prompts are carefully designed to elicit chain-of-thought

reasoning. The instruction to engage in internal dialogue with expressions like "let me think" and "wait" encourages the model to produce more natural and thorough reasoning rather than jumping directly to a conclusion.

Section B: Non-Thinking Mode Response

The authors show the format of a response in non-thinking mode: "<think></think><answer>A</answer>"

In this mode, the thinking tags are present but empty, and the answer is provided directly. This maintains the structural format while minimizing output length.

Sections C and D: Complete Responses and Additional Examples

The supplementary material includes complete response examples for four cases: a fake image, a real image, a fake video, and a real video. Additionally, Section D provides more examples showing the model's reasoning for various types of content.

For example, the complete response for a fake image (a lighthouse on a boardwalk) includes detailed analysis across seven points: Lighting and Shadows (noting that shadows are too uniform), Perspective and Depth (noting the perfectly straight boardwalk is unlikely in reality), Vegetation and Texture (noting reeds are too uniform), Lighthouse Structure (noting edges are too sharp and clean), Boardwalk and Railings (noting missing natural wear and tear), Sky and Clouds (noting clouds have unrealistic uniformity), and Overall Composition (noting the image is too symmetrical).

The response for a real image (a person with beard and sweater) similarly analyzes Physical Details, Clothing and Accessories, Background and Environment, Human Features, and Technical Quality, concluding that there are no inconsistencies.

These examples demonstrate the depth and quality of reasoning that BusterX++ can produce when explaining its detection decisions.

Available Resources for Implementation

Based on the paper and related work, here are the resources you would need to implement BusterX++:

Base Model

The base model is Qwen2.5-VL-7B-Instruct from Alibaba's Qwen team. This model is available on HuggingFace at the repository Qwen/Qwen2.5-VL-7B-Instruct. This is a 7 billion parameter multimodal large language model that can process both images and videos along with text. It has been instruction-tuned to follow user instructions.

External Thinking Reward Model

The external model used for computing the Thinking Reward is SophiaVL-R1-Thinking-Reward-Model-3B, which is referenced as citation [16] in the paper. This model evaluates the quality of reasoning content and provides a score between 0 and 1. You would need to find this model, which may be available through the authors' institutional resources or through related research groups.

Training Framework

The paper mentions that the work is built upon ms-swift, which is a training framework from ModelScope (Alibaba's AI platform). The ms-swift framework is available on GitHub at modelscope/swift and provides tools for efficient fine-tuning of large language models including LoRA support.

Datasets for Training

For training, you would need access to So-Fake-Set [24] for social media image forgery data and GenBuster-200K [59] for video forgery data. The So-Fake-Set is associated with the So-Fake-R1 paper and benchmarks social media image forgery. The GenBuster-200K dataset comes from the original BusterX paper.

Datasets for Real Content

For sourcing real content, the paper mentions OpenVid-1M HD [33], which is a large-scale high-quality dataset available for text-to-video generation research but contains real videos that can be used as the real class.

Image Generators

If you want to create your own fake images for training or evaluation, the paper uses FLUX 1.1 from Black Forest Labs (available through their API), SDXL which is available on HuggingFace at stabilityai/stable-diffusion-xl-base-1.0, GPT-4o through OpenAI's API, Recraft, Imagen 3 through Google's API, and Ideogram through their API.

Video Generators

For video generation, many of the generators mentioned are commercial services. HunyuanVideo from Tencent has been open-sourced. SkyReels V1 is available on GitHub at SkyworkAI/SkyReels-V1. Other generators like Sora (OpenAI), Seedance (ByteDance), Wan2.1 (Alibaba), and others are either commercial services or have limited availability.

Caption Generation Model

For generating captions to use as prompts for content generation, the paper uses Qwen-2.5-VL, which is available in various sizes on HuggingFace under the Qwen organization.

Project Repository

The official implementation is available at https://github.com/l8cv/BusterX, which contains code for both the original BusterX and BusterX++.

Implementation Workflow Summary

To implement BusterX++ based on this paper, you would follow these steps:

First, set up the environment by installing the ms-swift training framework, PyTorch with CUDA support, the transformers

library, the peft library for LoRA fine-tuning, and any other dependencies specified in the project repository.

Second, prepare the base model by downloading Qwen2.5-VL-7B-Instruct and setting up the LoRA configuration with rank 16 and alpha 32.

Third, prepare the training data by obtaining or creating datasets of real and fake images and videos. Process images to 1024?1024 resolution. Process videos to 1920?1080 resolution, 5 second duration, 24 FPS frame rate, and HEVC encoding.

Fourth, set up the reward functions by implementing the format reward, soft overlong reward, accuracy reward, hybrid thinking reward, and thinking reward using the external model.

Fifth, conduct Stage-1 Foundation RL training using DAPO with the R_stage-1 reward function until the model learns basic classification.

Sixth, conduct Stage-2 Thinking Mode Fusion by collecting samples from the Stage-1 model and performing supervised fine-tuning with the thinking and non-thinking mode templates.

Seventh, conduct Stage-3 Advanced RL using DAPO with the full R_stage-3 reward function, mixing both /think and /no_think prompts.

Finally, evaluate on GenBuster++ and other benchmarks to verify performance.

Opus 4.5

Claude is AI and can make mistakes. Please double-check responses.