AutoAD III: The Prequel – Back to the Pixels

Tengda Han[1] Max Bain[1] Arsha Nagrani[1†] Gül Varol[1,2] Weidi Xie[1,3] Andrew Zisserman[1]
[1]Visual Geometry Group, University of Oxford [2]LIGM, École des Ponts ParisTech [3]CMIC, Shanghai Jiao Tong University
https://www.robots.ox.ac.uk/vgg/research/autoad/

Abstract

Generating Audio Description (AD) for movies is a challenging task that requires fine-grained visual understanding and an awareness of the characters and their names. Currently, visual language models for AD generation are limited by a lack of suitable training data, and also their evaluation is hampered by using performance measures not specialized to the AD domain. In this paper, we make three contributions: (i) We propose two approaches for constructing AD datasets with aligned video data, and build training and evaluation datasets using these. These datasets will be publicly released; (ii) We develop a Q-former-based architecture which ingests raw video and generates AD, using frozen pre-trained visual encoders and large language models; and (iii) We provide new evaluation metrics to benchmark AD quality that are well matched to human performance. Taken together, we improve the state of the art on AD generation.

1. Introduction

Cinema is a matter of what's in the frame and what's out.
Martin Scorsese

Audio description (AD) is an accessibility tool for the blind and visually impaired that describes visual content which is essential for following video programs[1]. Automatically generating AD text is a challenging task as the information must be accurate, character-aware, story-aware, complementary to the soundtrack, and distilled into the gaps between speech. For TV broadcasters in the US and UK, providing AD for a certain percentage of video content has become a legal requirement.

With the current power of visual-to-text generative models, generating AD automatically is now becoming possible [66], and there has been a recent flurry of interest in this goal [20, 21], kick-started by the availability of films and AD provided in the MAD dataset [54]. Key innovations have included partial training of AD generative models using available large-scale datasets [20], and the introduction of a *character bank* to provide hints (as prompts) for the language model for the crucial objective of naming the characters in the generated text descriptions [21]. However, MAD only provides frame-level CLIP features (and only at 5 Hz) and this has limited the ability of generative models to provide fine-grained spatial details. Recent Visual Language Models (VLMs) [1, 31–33, 71, 74] have accessed the spatial feature map of the image (or video) in order to obtain fuller descriptions or answer more detailed questions.

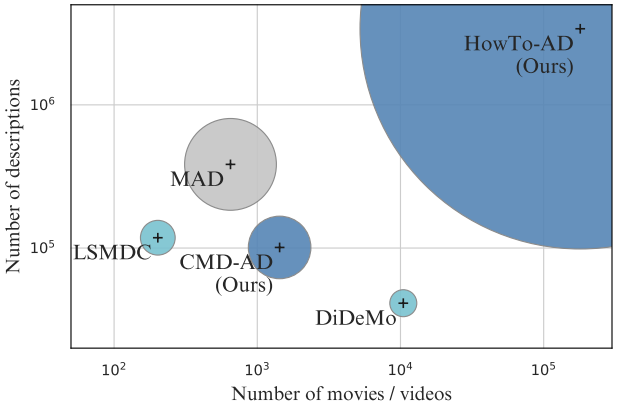


Figure 1. We propose two new movie Audio Description (AD) datasets *with pixels* – **CMD-AD** and **HowTo-AD** by temporally aligning or textually transforming existing pixel video datasets. The marker size is proportional to the total video durations and grey color indicates datasets with features instead of raw pixels.

We make the following contributions: First, we provide *two new datasets* that can be used to train an AD generation model end-to-end. The datasets go beyond MAD [54] in that they provide video as input, rather than only a CLIP frame feature, i.e., they go *back to the pixels*. The first dataset, *CMD-AD*, is constructed from two publicly available sources – the AD descriptions for films available from AudioVault[2] and the movie clips available from CMD [4]. The challenge in this case is how to determine the temporal alignment of these two sources given that one has only audio with AD, and the other (CMD) is non-contiguous with timings unknown with respect to the original movies. The second dataset, *HowTo-AD*, is constructed from the large-scale HowTo100M video dataset [40] that originally consists of YouTube videos with narrated instructions. Inspired by the use of Language Models (LMs) to rephrase the instructions as video captions in HowtoCaption [53], we use LMs to repurpose HowTo100M as an AD dataset containing

[1]https://www.3playmedia.com/learn/popular-topics/audio-description/

[2]https://audiovault.net

Dataset	with pixels	# movies	# AD	total duration
MovieNet [24]	✓	1100	–	–
LSMDC [46]	✓	202	118k	147h
MAD [54]	✗	650	384k	1027h
CMD [4]	✓	3605	–	1270h
CMD [4] ∩ AudioVault-8K [20]	✓	1803	–	647h
CMD-AD (ours)	✓	1432	101k	477h
HowTo-AD (ours)	✓	180,034*	3.4M	23652h

Table 1. **Statistics of Movie AD datasets.** Only a small number of movie datasets with AD are available, and they have different limitations: MovieNet only provides keyframes, LSMDC is short in duration, MAD only provides frame-level visual features, and CMD does not have corresponding ADs. We propose two new datasets for AD generation task: CMD-AD and HowTo-AD. *: strictly they are long videos rather than movies.

videos with an associated character bank, and text descriptions of the visual content that also names the person performing the actions. While this dataset is not a ground-truth AD dataset, we show that the pseudo ground-truth annotations are a valuable source of training data for AD. The statistics of these two new datasets are given in Table 1, and visually illustrated in Figure 1.

Our second contribution is a new architecture for AD generation that directly inputs a video clip and character bank proposals, and outputs a character-aware description. The model is based on the Q-former architecture of BLIP-2 [32] that bridges the visual space with the language space, then generates textual outputs with a large language model [72, 77, 84]. Our architecture is different from BLIP-2 [32] in that (i) it takes multi-frame movie clips as visual inputs, and (ii) it incorporates character bank information both from the face exemplars and the character names.

Our third contribution is on *evaluation*. Previous methods use a small test set of only 10 movies for evaluation. We introduce an evaluation dataset of 100 movies, based on our aligned movie clips with AD from AudioVault. This has far more *diversity* than the previous test sets used, covering, e.g. science fiction, westerns, action, horror, cartoon, and romance. As well as introducing a new test set, we also adopt two new evaluation methods. For AD, the gold standard evaluation is to compare the generated AD with that provided by humans. For model development, however, an automatic scalable evaluation is required. Previous works have used captioning metrics such as CIDEr [61] but these have severe limitations since they essentially measure n-gram accuracy, and the same semantic AD can be presented in multiple equivalent ways. To deal with this problem, [21] introduced a retrieval-based assessment, evaluating how often we can pick out the correct AD out of multiple neighboring ADs by comparing them to the generated AD using BertScore [79] semantic text similarity. In this work we adopt two new measures. The first called CRITIC, addresses one essential element of AD that distinguishes it from standard video captioning – that it must name the characters involved. The second measure follows the recent trend in using LLMs to assess the veracity of captioning [8, 11, 39, 55, 81] As an exemplar of the usefulness of these new measures we also use them to assess inter-rater consistency where the same film has AD provided by several human annotators. On these and traditional metrics, we show that our new architecture trained on raw pixels directly achieves impressive results for the task of Movie AD, outperforming previous works on both the standard MAD [54] eval set, and our new proposed test set.

2. Related Work

Dense video captioning. With the availability of large-scale data, the field has made significant progress in captioning images [31, 32, 41, 71], and trimmed short video segments [35, 38, 48, 49]. Movie AD generation is more related to the task of *dense* video captioning, where the goal is to concurrently address temporal localization and to describe each identified interval in an untrimmed video [30]. The approach to dense captioning has been explored either in two stages [25, 26, 30, 62, 64] or a single stage [7, 9, 14, 34, 42, 45, 50, 51, 62, 65, 68, 83], depending on whether localization and captioning are jointly addressed. Standard evaluation benchmarks for dense captioning consist of web videos such as YouCook2 [82], ActivityNet Captions [30], and ViTT [23]. Recently, Vid2Seq [68] repurposed the narrated web videos YT-Temporal-1B [75], using transcribed speech as the supervision source. In a similar spirit, HowToCaption [53] was collected by transforming the narrations of HowTo100M [40] into caption-like descriptions using LLMs, and LaVila [80] captioned long videos to enable large-scale video-text pretraining, also leveraging LLMs. The distinction between AD generation and dense captioning lies in the former's focus on character names, story relevance, and avoidance of interference with important audio content (*e.g.* character speech).

Movie understanding datasets. For movie understanding, current datasets facilitate a range of computer vision tasks including metadata classification [43], VQA [58], and visual character grounding [47]. These tasks often rely on auxiliary data such as movie plots [67], book adaptations [57], or AD [54] for dense annotations. However, due to copyright constraints, many datasets are limited to offering visual features (MAD [54]) or sparse keyframes (MovieNet [24]). CMD [4] circumvents this by providing urls to licensed YouTube clips. AutoAD [20] improves on automatic AD collection, and provides large-scale audio AD data.

Improvements in VLMs for images and videos. The recent success of LLMs [12, 59, 60, 78] and vision encoders [15, 44] has led to an explosion of multimodal (vision and language) models that can jointly understand both vision and text data. These methods largely work by mapping *frozen* image encoders (e.g. CLIP [44], EVA-CLIP [16]) to the textual embedding space of *frozen* LLMs [59, 60, 78], for example Flamingo [1], which does so via a Perceiver resampler [27], or BLIP2 [32], which uses a Q-former to achieve a similar mapping. Video-LLama [77] extends this idea to the audiovisual domain, by using the multimodal ImageBind [18] encoder in conjunction with video and audio Q-formers. While MV-GPT [48] finetunes a native video backbone [3] for the task of video captioning, most

works adapt image encoders to the costly video domain via temporally sampling a few frames with large strides [10, 63], or by representing each frame by a single token [65, 68, 83]. Given the impressive generalisation capabilities of these works made of up strong frozen components [73], we also adopt a similar approach, leveraging Video-LLama [77] and BLIP-2 [32] models as our backbone, with the key addition that we also integrate character information. More recent works such as MiniGPT-4 [84], MovieChat [55] and VideoBLIP [72] use stronger instruction-tuned LLMs, enabling further zero-shot capabilities.

Captioning evaluation. Human evaluation is the gold standard for judging caption quality, however it requires multiple annotators for consistency, is expensive and exceptionally slow. Existing automatic metrics, such as BLEU, ROUGE and CIDEr [61], all primarily measure n-gram overlap (however have different weighting schemes between n-grams, and across precision/recall), and do not capture the inherent subjectivity of the task, where different phrasing is often equally valid. Other metrics include SPICE [2] (adds action and object relationships), while model-based metrics using earlier language models or image-language models include BERT-Score [79], BERT-Score++ [70] (fine-tunes BERT for image captioning), LEIC [13] and NUBIA [29] (custom trained models for image caption evaluation), TIGEr [28], CLIPScore [22], and EMScore [52]. Given the explosion of LLMs, however, recent works explore the use of state-of-the-art LLMs, such as GPT-4, as a surrogate for humans. Because these models are often trained with RLHF, they already exhibit strong human alignment [6], and can be used to assess text quality well (LLM-as-a-judge). [11, 81] show that using strong LLMs as judges (such as GPT-4) aligns highly with human preferences on a range of standard language-based tasks, such as conversational instruction following. CLAIR [8] extends this idea to image captioning, showing similar strong correlations to human preferences on visual-language datasets such as MS-COCO and Flickr8K, while VideoChatGPT [39] and MovieChat [55] use LLM-assisted evaluation for video tasks such as videoQA as well.

3. New Datasets for Pixels to AD

In this section, we describe our two new datasets that contain raw video pixels mapped to AD annotation: CMD-AD (Sec. 3.1) which is based on CMD [4], and HowTo-AD (Sec. 3.2) based on HowTo100M [40].

3.1. CMD-AD – Pixels from Aligned CMD

The AudioVault website provides human annotated Audio Descriptions in the form of audio files with the spoken AD added to the original movie soundtrack (no video). The CMD dataset [4] consists of short (about 2 minutes long) non-contiguous movie clips in the form of video files on YouTube (around 10 clips per movie). Although there are about 2000 movies overlapping between these two data sources, temporally aligning the AD with the movie clips from CMD is a non-trivial task due to several challenges: First, the movie soundtracks from AudioVault audio files have been modified and re-encoded to add the AD, therefore the audio signals from AudioVault files and CMD movie clips are not identical; second, AudioVault audio files cover the full movie duration (e.g. around 90 minutes), whilst a CMD clip covers only 2 minutes, and performing precise alignment over the extent of the movie has the potential for many erroneous matches across the search space; third, the same movie published in different locations might have been recorded at different speeds (e.g. NTSC 29.97 fps vs. PAL 25 fps[3]), introducing another unknown into the alignment.

Split	# movies	# AD
CMD-AD-Train	1332	93,952
CMD-AD-Eval	100	7,316
total	1432	101,268

Table 2. **Statistics of the CMD-AD dataset.**

We propose a two-stage alignment pipeline to overcome these challenges and get precise temporal alignment between hour-long AudioVault audio files and non-contiguous short CMD movie clips from the same movie. To achieve this, we use two quasi-independent modalities: (i) the transcribed spoken text from the characters (not the AD), and (ii) the raw audio signal containing both non-speech sounds (music, sound effects) and the speech.

Stage1: Text-text alignment. The aim in this stage is to first roughly localize the CMD movie clip with the AudioVault audio to reduce the search space. In detail, we use WhisperX [5] with the diarization module to separate the AD narration from the character speech, and obtain movie 'subtitles' with timestamps for both AudioVault audio and CMD movie clips. These are denoted as $\mathcal{S}_{\text{AV}} = \{(s_1,t_1),...,(s_m,t_m)\}$ and $\mathcal{S}_{\text{CMD}} = \{(s'_1,t'_1),...,(s'_n,t'_n)\}$, where each s_i denotes subtitle strings and t_i denotes the temporal extent of this subtitle. Note that $n \ll m$ because CMD movie clips are much shorter than the entire movie, also the subtitles from the two sources are different because of arbitrary sentence partitioning by WhisperX or possible diarization errors. To localize the CMD clip on the AudioVault movie time axis, we compute a simple word-error-rate (WER) using a sliding window approach as follows: we combine the CMD subtitles into a paragraph $\mathcal{P}_{\text{CMD}} = [s'_1;...;s'_n]$, then compute WER with AudioVault subtitles within a chunk size of n. Formally, let $\mathcal{P}^{(i)}_{\text{AV}} = [s_i;...;s_{i+n}]$ denote the AudioVault subtitle paragraph consisting of n continuous subtitle entries starting from i-th subtitles. For a particular CMD clip, the objective of text-text alignment is

$$T_{\text{tt-align}} = \underset{t_i}{\text{argmin}} \left\{ \text{WER}(\mathcal{P}_{\text{CMD}}, \mathcal{P}^{(i)}_{\text{AV}}) \right\}. \quad (1)$$

The text-text alignment is not accurate when the CMD movie clip does not have many dialogues, *e.g.* in action movies. In practice, we find it gives reliable rough time points for more than 90% of CMD clips by randomly checking 10+ movies manually.

Stage2: Audio-audio alignment. Given the rough alignment

[3] https://en.wikipedia.org/wiki/576i#PAL_speed-up

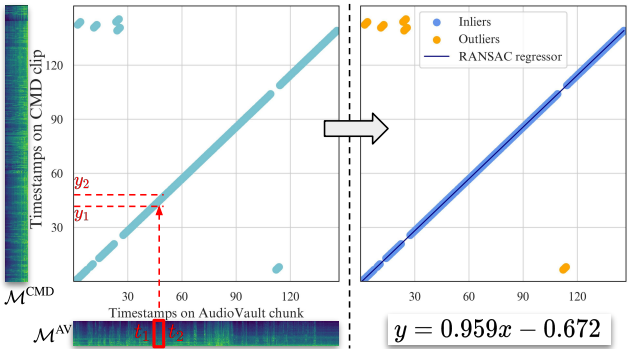


Figure 2. **Audio-audio alignment between two sources. (left)**: For each small audio segment on AudioVault, we find the best-matching audio segment on CMD clip, and plot two timestamps as scatters; **(right)**: Fitting a straight line with RANSAC we can get the precise mapping function between two sources. The slope of the fitted line $0.959 < 1$ indicates this CMD clip plays slightly faster than the corresponding AudioVault chunk.

(which may be noisy) provided by Stage 1, this stage aims to verify the match, and obtain a *precise* temporal alignment by comparing audio signals from the two sources. The objective is to get a precise linear mapping for each CMD movie clip:

$$f: \{T_{\text{AV}} \rightarrow T_{\text{CMD}}\} = W \cdot t_{\text{AV}} + B, \tag{2}$$

where W is the speed rate between the AudioVault and CMD movie sources which might not be 1.0 due to different movie fps, and B is the time shift. The key idea here is that even though the individual CMD clips are matched locally, the parameters W and B can be assumed to be global (i.e. constant) across the movie. Hence, matches can be verified as they will lie on a line specified by W and B, and this line can be obtained by a standard robust fitting method. Here we use RANSAC [17].

To obtain precise audio alignment, we perform alignment on low-level audio representation mel-spectrogram. First, we compute mel-spectrogram for both AudioVault audio and the CMD movie clip, denoted as $\mathcal{M}^{\text{AV}}$ and $\mathcal{M}^{\text{CMD}}$. We only take a short AudioVault audio chunk based on the previous text-text alignment result. Second, we mask out mel-spectrogram regions of Audio Descriptions based on the timestamps obtained from WhisperX, as the AD signal only exists in AudioVault and not in the CMD movie clip. Next, we perform sliding window matching with a window size $w = 1.6s$. For each 1.6-second audio chunk on AudioVault starting from t_1 to t_2, we find the corresponding timestamps on CMD audio which has a maximum correlation:

$$y_1, y_2 = \underset{t_i, t_i+w}{\text{argmax}} \left\{ \text{cor}\left(\mathcal{M}^{\text{AV}}_{[t_1,t_2]}, \mathcal{M}^{\text{CMD}}_{[t_i,t_i+w]} \right) \right\}. \tag{3}$$

These matches can be thought of as points on a scatter plot from (t_1, y_1) to (t_2, y_2) for a series of small windows from AudioVault, as shown in Figure 2. Finally, we use a RANSAC algorithm to fit a line through these match points over all clips to obtain the mapping in Equation (2). Based on the ratio between common movie fps, we filter RANSAC output by $0.8 < W' < 1.25$ and empirically choose mean-square-error MSE < 100. We find these two conditions give very decisive boundaries for confident RANSAC output. For instance, the successful RANSAC fitting at Figure 2 has an MSE of 0.68, whereas failed fittings typically have an MSE > 500.

Summary. With this two-stage method, we obtain accurate temporal alignment between AudioVault audio and CMD movie clips, therefore we can map the AudioVault AD annotations onto the CMD time axis to get video-text annotations. This gives us the dataset CMD-AD (statistics are provided in Table 2), consisting of 101k AD segments spanning 1,432 movies. Note that the total number of overlapping movies between the two datasets is 1,803, which means an 80% success rate of precise alignment. A higher success rate can be achieved by using a larger search window or an iterative alignment pipeline, which we leave as future work.

We use 1332 movies for training and 100 movies for evaluation, naming the splits CMD-AD-Train and CMD-AD-Eval sets, respectively.

3.2. HowTo-AD – Pixels from HowTo100M

Our second dataset is based on the large-scale instructional video dataset HowTo100M [40], that contains over 1.2M videos with ASR subtitles from YouTube. At first glance, the ASR transcripts of these videos may look drastically different from that of AD in movies, since the spoken words are primarily aimed to instruct the viewer on how to carry out various daily tasks.

However, we can *transform* the instruction ASR into pseudo-AD in two steps. The first step is to adopt the captions generated from HowToCaption [53], where the ASR transcripts have been transformed into concise and *descriptive captions* with large language models (LLMs). To improve caption temporal alignment with the corresponding video timestamps, the authors employ an off-the-shelf Temporal Alignment Network [19], while also discarding non-alignable subtitles (such as "Hello, welcome to my channel!"). The second step addresses the key difference between descriptive captions and audio descriptions, that is, character names do not appear in the captions. For this transformation, we detect the subjects of description sentences and uniformly replace them with a randomly chosen character name, *e.g.* transforming '*a man* is pouring wine' into '*John* is pouring wine'. This completes the transformation from HowTo100M captions to the HowTo-AD.

Additionally, to mimic having a *character bank* as external knowledge as in [21, 36, 69], we also provide each instructional video with a pseudo-character bank that includes: the chosen character name and the character portrait face extracted from the instructional video, and a few face exemplars sampled from other videos to mimic off-screen characters. An overview of the pipeline with an example is shown in Figure 3.

Because of the noisy nature of YouTube videos and the abundance of data in the HowTo100M dataset, we filter out less preferable videos by the quality of subject detection in HowToCaption, the frequency of names in ASR, and the quality of

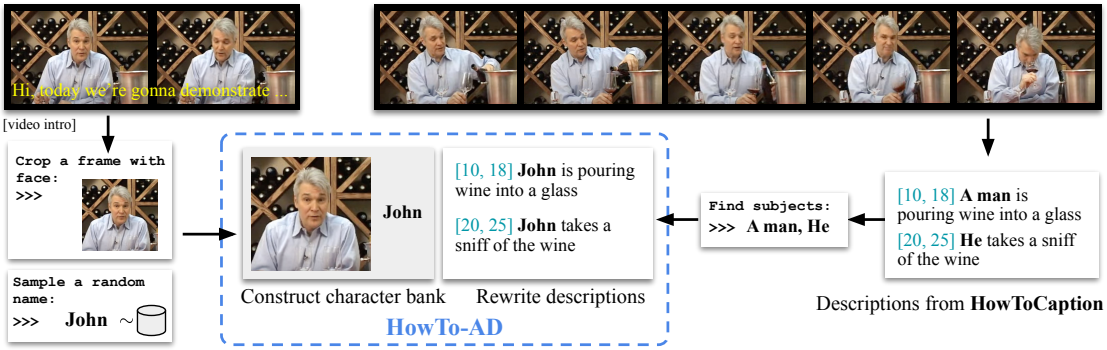


Figure 3. **HowTo-AD dataset**. We convert the LLM rewritten video descriptions (from HowToCaption) to fit movie audio descriptions by (i) uniformly replacing the subjects in descriptions with a randomly sampled name, *i.e.* **John**, and (2) constructing a character bank by providing a frame with the instructor and the randomly sampled name. The video sample is from https://youtu.be/aRbQb19v2JI.

character portrait faces; details are in the Appendix. As shown in Table 1, the HowTo-AD dataset ends up with a subset of 180k YouTube videos from the original HowTo100M dataset – which is about 20% of the full HowTo100M – and 3.4M transformed AD segments with timestamps from HowToCaption dataset.

4. Model Architecture

With pixel data available, we propose two visual captioning models based on BLIP2 [32] and Llama2 [60] for movie AD generation. Specifically, we propose two new architectures called Movie-BLIP2 and Movie-Llama2. Both of them take 8 video frames, resized at 224×224 pixels as inputs, then use EVA-CLIP [56] to extract dense visual features. Next, we use a Q-former to attend to spatial-temporal feature grids to extract visual descriptors represented by 32 vectors. Both models also processes image inputs from character face exemplars. In this case, they take a single image resized at 224×224 pixels, and then use the same EVA-CLIP to extract visual features in spatial grid, and the same Q-former to attend to this spatial feature grid and extract 32 vectors as image descriptors. The video and image descriptors are passed to two shallow projection heads respectively, to project them on the language embedding space. Finally, the projected visual outputs together with language prompts are passed to a large language model (OPT for Movie-BLIP2 and Llama2 for Movie-Llama2) to generate movie AD in text form. An overview of architecture is shown in Figure 4.

The Movie-BLIP2 architecture inherits from the original Image-based BLIP2 architecture, and it uses OPT [78] as the language model. The Movie-Llama2 architecture inherits from the image-based MiniGPT-4 [84] which connects BLIP2's visual embedding with Llama2 language embedding [60]. Our Movie-Llama2 follows the same setup and uses Llama2 as the language model. We take pre-trained checkpoints from open-sourced projects [72] and [77]. Details of these architectures are in the Appendix. Following previous works [84], by default, all the visual backbone, language model, and the Q-former are frozen, and we only train the projection heads.

Training details. By default, we adopt a two-stage training strategy. The model is firstly pretrained on HowTo-AD and then finetuned on CMD-AD-Train. We pretrain on HowTo-AD for 1 epoch and finetune on CMD-AD-Train for 2 epochs. We find finetuning beyond 2 epochs leads to overfitting. We use a batch size of 8 AD samples, an AdamW optimizer [37] with 3×10^{-5} learning rate and a cosine decay schedule. For both the pretraining and finetuning stages, the training pipeline fits in a single A40 GPU with 48GB GPU memory. More training details are provided in the Appendix.

5. Evaluation Methods

We propose two new methods for evaluating movie AD generation: CRITIC for identifying correct characters, and an LLM-based AD evaluation for assessing holistic semantics of AD.

CRITIC (Co-Referencing In Text for Identifying Characters). The CRITIC metric assesses the accuracy of character naming in predicted AD against human-generated reference AD. The metric is designed to be robust to (i) co-referencing complexities (ii) pronoun usage, and (iii) orthographic variation in character names. The objective is to measure the quality of character reference in the generated AD compared to the ground-truth AD. For example, the model might generate AD with the text 'Jack' or pronouns like 'he', the CRITIC metric aims to evaluate the accuracy of these references.

To achieve this, a co-referencing model[4] is applied to both predicted and reference AD passages. Specifically, let $\mathcal{C}=$ "$c_1,c_2,...,c_n$." denote the set of official character names from the cast list of a movie, combined into a single sentence. For a specific movie, we group the predicted and reference audio descriptions into long paragraphs, denoted as AD_{pred} and AD_{ref} respectively. In order to guide the co-referencing model to detect character names, both AD_{pred} and AD_{ref} are prefixed with the character list sentence $\mathcal{C}$, as shown in Figure 5 (**a** and **c**).

Next, the co-referencing model is applied to both paragraphs from prediction and reference, yielding sets of identities E_{pred} and E_{ref}. Each identity E includes references and pronouns linked to a single entity. We only keep identities containing exactly one character name from C, ensuring distinct association

[4] https://github.com/shon-otmazgin/fastcoref

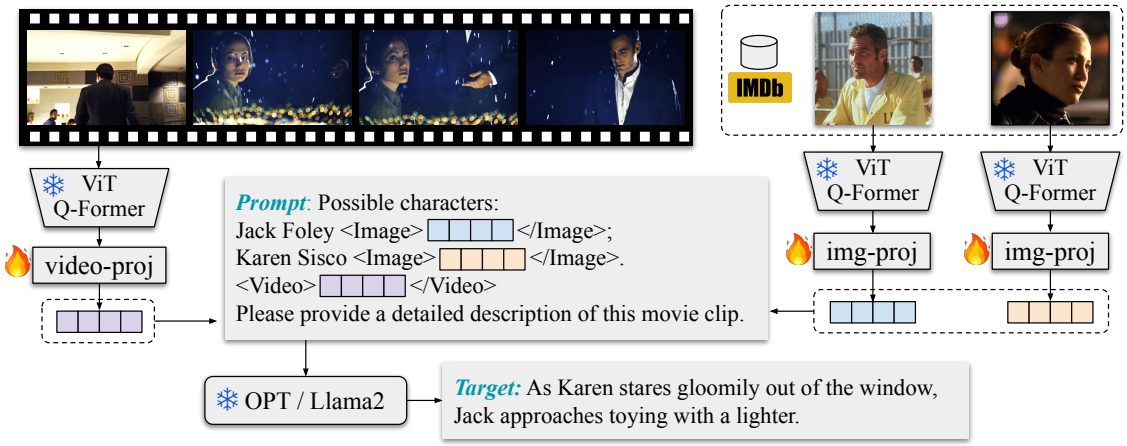


Figure 4. **Architecture overview.** Our model takes as input movie frames and movie character bank from IMDb including face exemplars and character names, and produces character-aware audio descriptions. The input images/videos are first fed to a frozen visual feature extractor to obtain spatial or spatial-temporal visual features. Then it uses a shared Q-former to process the visual information and project them to the language embedding space, to leverage frozen large language models(LLM) like OPT and Llama2 for text generation.

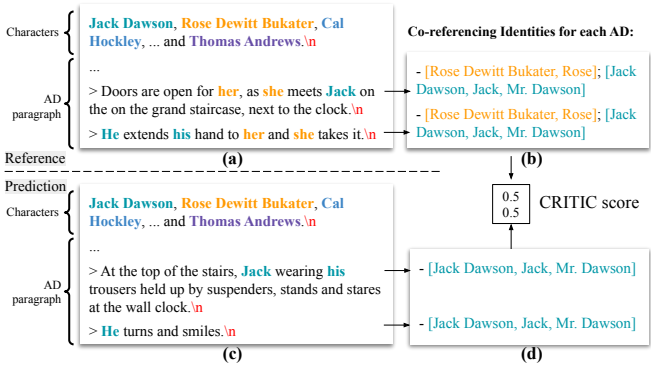


Figure 5. **Illustration of the CRITIC metric.** The paragraphs consisting character list and AD (a,c) are fed into a co-referencing model to get co-referencing identities (b,d). The CRITIC metric computes an IoU between the identities in the prediction vs. the identities from the reference.

with individual characters. Importantly, we remove pronouns like 'he', 'she', and 'they' in each co-referencing identity to exclude ambiguous pronoun matching. Next, we map each sentence to its corresponding set of co-referencing identities, which may be empty (when no names are recognized), singular, or multiple, as depicted in Figure 5 (**b** and **d**).

The CRITIC metric M_{CRITIC} is then calculated as an IoU, for i-th AD reference (with valid character identities):

$$M_{\text{CRITIC}} = \frac{|E_{\text{pred}} \cap E_{\text{ref}}|}{|E_{\text{pred}} \cup E_{\text{ref}}|} \tag{4}$$

where $|E_{\text{pred}} \cap E_{\text{ref}}|$ is the count of matching identities between predicted and reference ADs, and $|E_{\text{pred}} \cup E_{\text{ref}}|$ is the total count of unique identities in both the prediction and the reference. The CRITIC score is averaged across all the AD references. Intuitively, if the predicted AD includes a name, the CRITIC score verifies whether the name refers to the correct identity; if the prediction includes a pronoun like 'he', the CRITIC score first resolves the identity by co-referencing, then verifies whether the name is correct. The CRITIC score has a range between 0 and 1, with 1 being perfect.

LLM-AD-eval. We also adopt the LLM as a judge [11, 81] procedure for AD quality assessment. Following previous works [39], we use a 'gpt-3.5-turbo' API from OpenAI and an open-sourced 'llama-2-7b-chat' model [60]. prompting the model to assess the matching quality between a pair of predicted AD and ground-truth AD by a score from 1 to 5, where 5 indicates the best matching and 1 indicates the worst matching. To be complementary to the CRITIC metric, for LLM-AD-eval we instruct the LLM to (1) consider pronouns as valid matches and ignore character names, and (2) focus on human actions, objects and interactions. The customized prompts are provided in the Appendix.

6. Experiments

We outline the datasets (Sec. 6.1) and evaluation measures (Sec. 6.2) employed in our experiments, and provide an analysis on inter-rater agreement between AD annotation versions (Sec. 6.3). We report quantitative results, ablating the architectural design and the effect of HowToAD pretraining (Sec. 6.4), followed by qualitative results thanks to our pixel movie data (Sec. 6.5).

6.1. Datasets

AudioVault-8k is the dataset collected from https://audiovault.net/ by [21] that contains full-movie AD and subtitles transcribed from user-uploaded audio description files covering 7800 movies. **CMD (Condensed Movie Dataset)** [4] contains movie clips collected from YouTube for more than 3k movies. On average each movie has 10 non-contiguous clips and each clip spans for a few minutes. **HowTo100M** [40] contains 1M YouTube long videos with more than 100M ASR segments. It is typically used for video pretraining. **CMD-AD** is the new movie AD dataset introduced

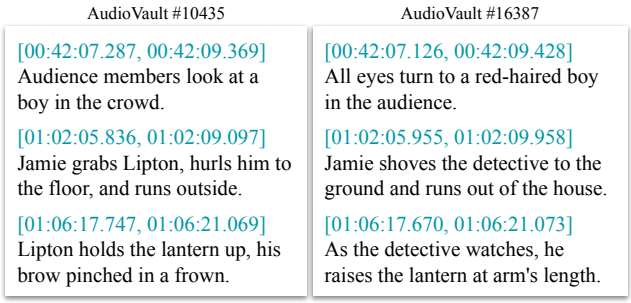


Figure 6. **An example of inter-rater evaluation.** Some movies on AudioVault have multiple available ADs. Two versions of human-annotated ADs for the same visual scene are shown here. These ADs are filtered with a tIoU threshold of 0.9, and from the movie 'Dead Silence'(`tt0455760`).

tIoU	#movies	#AD pairs	CIDEr	R@1/5	CRITIC	LLM-AD-eval†
0.8	315	4447	61.5	71.2	42.0	2.56 / 3.04
0.9	267	999	69.8	80.4	47.6	3.06 / 3.53

Table 3. **Inter-rater agreement on AudioVault AD annotations.** AD from different annotators do not usually synchronize. A higher tIoU threshold filters out fewer AD pairs but they are more likely to describe the exact same visual event. †: LLM-AD-Eval scores are computed from 'gpt-3.5-turbo' / 'llama-2-7b-chat' respectively. For all the metrics, a higher number indicates better quality. R@1/5 and CRITIC are upperbounded at 100 and LLM-AD-eval is between 1 to 5.

in Section 3.1, by aligning AD data with CMD clips [4]. It contains 101k ADs for more than 1432 movies. We split a 100-movie evaluation set named CMD-AD-Eval and use the rest for training. **HowTo-AD** is the new AD dataset introduced in Section 3.2, transformed from HowTo100M [40]. It contains 180k YouTube videos with augmented descriptions and character exemplars. We mainly use it for AD generation pertaining.

6.2. Evaluation Measures

In addition to the two new evaluation measures introduced in Section 5, **CRITIC** and **LLM-AD-Eval**, we also monitor Recall@k/N, CIDEr, and perplexity. **Recall@k/N** [21] is a retrieval metric that distinguishes the predicted text among a set of temporal neighbours. The parameters k and N mean within a temporal window of N neighbouring reference ADs, whether the predicted AD can retrieve the corresponding reference AD at top-k position. We use Recall@1/5 on CMD-AD-Eval and Recall@5/16 on MAD-Eval to compare with previous works. We use the official implementation provided by [21]. **CIDEr** [61] is a popular text similarity metric that is based on word matching rate. We include Recall@k/N and CIDEr here as they have been used in recent work on AD [20, 21] and we also compare on the test datasets of those works.

6.3. Inter-rater Evaluations

Many of the films in Audiovault have multiple ADs available. Typically these are UK and US versions. In this section, we use the agreement between the human-provided AD versions to assess the usefulness of the four evaluation metrics – CIDEr, Recall@1/5, CRITIC, and LLM-AD-eval. Note, these evaluations are carried out directly on the text version of the AD, so no pixels are involved.

In the AudioVault-8K dataset, there are 402 movies with more than one version of AD annotations. We conduct inter-rater experiments on this subset of AD annotations. Three challenges emerge when conducting inter-rater comparisons for the same visual scene: (i) Different versions of AD annotations might correspond to different versions of the same movie which do not naturally synchronize as introduced in Section 3.1. We apply the audio-audio alignment pipeline in 3.1 to synchronize both AD annotations. (ii) The timing of providing AD is subjective and arbitrary within a short time interval [21], to obtain different ADs for the exact same visual moment, we have to filter the time segments of two AD versions with a temporal Intersection-over-Union (tIoU). (iii) For about 20% of movies, the multiple AD versions from AudioVault are simply narrating the same scripts again with minor modifications, which does not reflect independent inter-rater comparisons. We filter out those movies by checking the exact sentence-matching rate.

The inter-rater evaluation is shown in Table 3. With a higher tIoU threshold, we get fewer AD annotation pairs covering fewer movies, but the AD annotation pairs are more likely to describe the same movie scene. The numbers can be regarded as human-level upperbound. Note that with a lower tIoU (from 0.9 to 0.8), all the metrics drop significantly, highlighting the temporal sensitivity of AD tasks and the importance of precise data alignment. A few pairs of human-annotated AD from AudioVault are shown in Figure 6, where the left and right panels are from two annotators for the same visual scene.

Method	V-model	L-model	CIDEr	R@1/5	CRITIC	LLM-AD-eval†
AutoAD-II	CLIP-B-32	GPT2	13.5	26.1	8.2	1.53 / 2.08
Movie-BLIP2	Eva-CLIP	OPT-2.7B	21.2	29.3	24.5	**2.13** / 2.66
Movie-Llama2	Eva-CLIP	LLama2-7B	**21.7**	**30.0**	**25.2**	2.05 / **2.85**

Table 4. **Architecture experiments** on CMD-AD-Eval. We compare the proposed two architectures with AutoAD-II. All of them take character bank inputs. †: from 'gpt-3.5-turbo' / 'llama-2-7b-chat' respectively. Note, AutoAD-II is trained with averaged frame features to mimic its original setting on the feature-only MAD dataset.

Method	pretrain	CIDEr	R@1/5	CRITIC	LLM-AD-eval†
Movie-BLIP2	✗	21.2	29.3	24.5	2.13 / 2.66
Movie-BLIP2	HowTo-AD	**22.3**	**29.8**	**30.2**	**2.25 / 2.78**
Movie-Llama2	✗	21.7	30.0	25.2	2.05 / 2.85
Movie-Llama2	HowToCaption‡	20.8	29.4	25.6	2.07 / 2.85
Movie-Llama2	HowTo-AD	**25.0**	**31.2**	**32.7**	**2.29 / 2.92**

Table 5. **Effect of HowTo-AD pretraining** on CMD-AD-Eval. †: from 'gpt-3.5-turbo' / 'llama-2-7b-chat', respectively. ‡ uses the same 180k-video subset as HowTo-AD, but without constructing character banks or rewriting captions.

6.4. Quantitative Results

Architecture Comparisons on Aligned-CMD. In Table 4, we compare the proposed Movie-BLIP2 and Movie-Llama2 architectures with previous methods on the CMD-AD dataset. All of these models are trained on the CMD-AD-Train set and

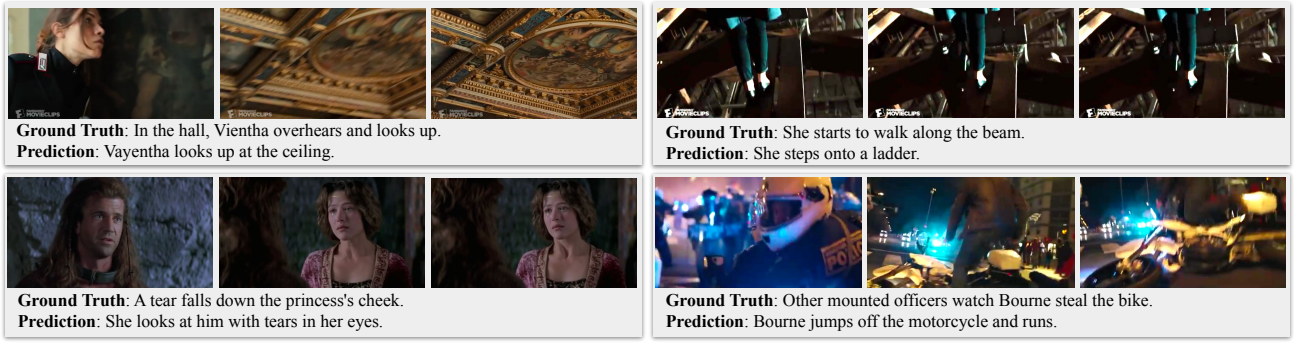


Figure 7. **Qualitative results.** AutoAD-III predictions correctly identify the semantics of the scene, by referring to the characters ('Vayentha', 'Bourne'), their relations ('looks at him'), actions ('steps onto', 'jumps off'), emotions ('tears'), objects ('ladder', 'motorcycle'). The comparison with the ground truth further highlights the limitations of the n-gram based metrics since the same meaning can be conveyed with different wordings.

Method	CMD-AD-Eval				MAD-Eval	
	CIDEr	R@1/5	CRITIC	LLM-AD-Eval†	CIDEr	R@5/16
Video-BLIP2 [72] (no ft)	4.8	22.0	0.0	1.40 / 1.89	5.0	35.2
Video-Llama2 [77] (no ft)	5.2	23.6	0.0	1.43 / 1.91	4.8	33.8
MM-Narrator-GPT4 [76]	-	-	-	-	13.9	-
AutoAD-I [20]	-	-	-	-	14.3	42.1
AutoAD-II [21]	13.5	26.1	8.2	1.53 / 2.08	19.2	51.3
Movie-BLIP2 **(ours)**	22.3	29.8	30.2	2.25 / 2.78	22.8*	52.0*
Movie-Llama2 **(ours)**	25.0	31.2	32.7	2.29 / 2.92	24.0*	52.8*

Table 6. **Comparison with other methods on CMD-AD-Eval and MAD-Eval.** Additionally, we evaluate the out-of-the-box Video-BLIP2 and Video-Llama2 video captioning models (without any AD finetuning) directly on both datasets. †: from 'gpt-3.5-turbo' / 'llama-2-7b-chat' respectively. *: these results are obtained on MAD-Eval *without* any training on MAD-Train.

evaluated on the CMD-AD-Eval set. We implement and train AutoAD-II on CMD-AD-Train based on the public codebase. The results show that both Movie-BLIP2 and Movie-Llama2 perform much better than AutoAD-II architecture. The stronger performance is attributed to multiple factors – stronger visual backbone, stronger language model, and taking visual *grid* feature as input instead of a single vector as in AutoAD-II. Movie-Llama2 has a much stronger language model than Movie-BLIP2 (Llama2-7B vs OPT-2.7B), but it achieves a similar performance wrt Movie-BLIP2. Note that all these models are not pretrained on HowTo-AD yet.

Effect of HowTo-AD Pretraining. Taking the Movie-BLIP2 and Movie-Llama2 settings from Table 4, we compare the effect of HowTo-AD by pretraining the same architecture on the HowTo-AD dataset then finetuning on CMD-AD-Train set. We also pretrain with the same subset from HowToCaption without using the character bank or rewriting captions. The results in Table 5 show that large-scale pretraining on our HowTo-AD dataset substantially boosts the performance on all four metrics for both models. *e.g.* improving CRITIC from 25.2 to 32.7 and CIDEr from 21.7 to 25.0 for Movie-Llama2. But pretraining on HowToCaption does not help much on the finetuned movie AD task, possibly because of the domain gap from the data and task.

Comparison with Other Methods. We compare with other methods on two datasets: CMD-AD-Eval introduced in this work, and MAD-Eval proposed in [20] (Table 6). Note that MAD-Eval is a 10-movie subset from LSMDC, where we can get short movie clips for evaluation. However, we can not perform any training on MAD-Train since no pixels are available. The proposed method Movie-BLIP2 and Movie-Llama2 perform much better than the previous methods including MM-Narrator with GPT4 as the language model. We also evaluate video captioning models like Video-BLIP2 and Video-Llama2 but neither of them performs well on AD, highlighting the challenges of Movie AD task.

6.5. Qualitative Analysis

Figure 7 illustrates several random examples from the CMD-AD-Eval set. For each sample, we display the predictions of our Movie-Llama2 model, as well as the ground truth AD. We observe that, while different wording than the ground truth, the semantics of the AD content remain largely similar for Our method. Interestingly, in the first example, the ASR pipeline [5] transcribed the name incorrectly as 'Vientha' but our model fixed the name through the character bank. More examples can be found in the Appendix.

7. Conclusion

This work advances automatic AD generation for movies by: (i) collecting AD for pixel data through audio-audio alignment between full movies (without pixels) and public movie snippets, and pseudo-labelling instruction videos; (ii) showing that recent video-language architectures provide a significant performance boost, bringing AD generation systems closer to real-world applications; and (iii) proposing new evaluation methods tailored for AD. One of the limitations that necessitates future work is the coherence across AD narrations throughout the movie: AD should not repeat the same information, or provide story-irrelevant details. To this end, external knowledge such as plot summaries may be utilized to incorporate story-centric elements. Future directions could also explore the harmony between the narration tone and the movie content for an engaging experience.

Acknowledgements. This research is funded by EPSRC PG VisualAI EP/T028572/1, and ANR-21-CE23-0003-01 CorVis.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022. 1, 2

[2] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *Proc. ECCV*, pages 382–398. Springer, 2016. 3

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViViT: A Video Vision Transformer. In *ICCV*, 2021. 2

[4] Max Bain, Arsha Nagrani, Andrew Brown, and Andrew Zisserman. Condensed movies: Story based retrieval with contextual embeddings. In *Proc. ACCV*, 2020. 1, 2, 3, 6, 7

[5] Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. Whisperx: Time-accurate speech transcription of long-form audio. In *INTERSPEECH*, 2023. 3, 8

[6] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023. 3

[7] Aman Chadha, Gurneet Arora, and Navpreet Kaloty. iPerceive: Applying common-sense reasoning to multi-modal dense video captioning and video question answering. In *Proc. WACV*, 2021. 2

[8] David Chan, Suzanne Petryk, Joseph E Gonzalez, Trevor Darrell, and John Canny. CLAIR: Evaluating image captions with large language models. *arXiv preprint arXiv:2310.12971*, 2023. 2, 3

[9] Shaoxiang Chen and Yu-Gang Jiang. Towards bridging event captioner and sentence localizer for weakly supervised dense event captioning. In *Proc. CVPR*, 2021. 2

[10] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023. 3

[11] Cheng-Han Chiang and Hung-yi Lee. Can large language models be an alternative to human evaluations? *arXiv preprint arXiv:2305.01937*, 2023. 2, 3, 6

[12] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 2

[13] Yin Cui, Guandao Yang, Andreas Veit, Xun Huang, and Serge Belongie. Learning to evaluate image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5804–5812, 2018. 3

[14] Chaorui Deng, Shizhe Chen, Da Chen, Yuan He, and Qi Wu. Sketch, ground, and refine: Top-down dense video captioning. In *CVPR*, 2021. 2

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[16] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. EVA: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023. 2

[17] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981. 4

[18] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. ImageBind: One embedding space to bind them all. In *CVPR*, 2023. 2

[19] Tengda Han, Weidi Xie, and Andrew Zisserman. Temporal alignment networks for long-term video. In *Proc. CVPR*, 2022. 4

[20] Tengda Han, Max Bain, Arsha Nagrani, Gül Varol, Weidi Xie, and Andrew Zisserman. AutoAD: Movie description in context. In *Proc. CVPR*, 2023. 1, 2, 7, 8

[21] Tengda Han, Max Bain, Arsha Nagrani, Gül Varol, Weidi Xie, and Andrew Zisserman. AutoAD II: The sequel – who, when, and what in movie audio description. In *Proc. ICCV*, 2023. 1, 2, 4, 6, 7, 8, 14

[22] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *EMNLP*, 2021. 3

[23] Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera, and Radu Soricut. Multimodal pretraining for dense video captioning. *arXiv preprint arXiv:2011.11760*, 2020. 2

[24] Qingqiu Huang, Yu Xiong, Anyi Rao, Jiaze Wang, and Dahua Lin. MovieNet: A holistic dataset for movie understanding. In *ECCV*, 2020. 2

[25] Vladimir Iashin and Esa Rahtu. A better use of audio-visual cues: Dense video captioning with bi-modal transformer. In *BMVC*, 2020. 2

[26] Vladimir Iashin and Esa Rahtu. Multi-modal dense video captioning. In *CVPR Workshops on Multimodal Learning*, 2020. 2

[27] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021. 2

[28] Ming Jiang, Qiuyuan Huang, Lei Zhang, Xin Wang, Pengchuan Zhang, Zhe Gan, Jana Diesner, and Jianfeng Gao. Tiger: Text-to-image grounding for image caption evaluation. *arXiv preprint arXiv:1909.02050*, 2019. 3

[29] Hassan Kane, Muhammed Yusuf Kocyigit, Ali Abdalla, Pelkins Ajanoh, and Mohamed Coulibali. Nubia: Neural based interchangeability assessor for text generation. *arXiv preprint arXiv:2004.14667*, 2020. 3

[30] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proc. ICCV*, pages 706–715, 2017. 2

[31] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022. 1, 2

[32] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv:2301.12597*, 2023. 2, 3, 5

[33] Linjie Li, Zhe Gan, Kevin Lin, Chung-Ching Lin, Zicheng Liu, Ce Liu, and Lijuan Wang. LAVENDER: Unifying video-language understanding as masked language modeling. In *CVPR*, 2023. 1

[34] Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. Jointly localizing and describing events for dense video captioning. In *Proc. CVPR*, 2018. 2

[35] Kevin Lin, Linjie Li, Chung-Ching Lin, Faisal Ahmed, Zhe Gan, Zicheng Liu, Yumao Lu, and Lijuan Wang. SwinBERT: End-to-end transformers with sparse attention for video captioning. In *Proc. CVPR*, 2022. 2

[36] Kevin Lin, Faisal Ahmed, Linjie Li, Chung-Ching Lin, Ehsan Azarnasab, Zhengyuan Yang, Jianfeng Wang, Lin Liang, Zicheng Liu, Yumao Lu, Ce Liu, and Lijuan Wang. Mm-vid: Advancing video understanding with gpt-4v(ision). *arXiv preprint arXiv:2310.19773*, 2023. 4

[37] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[38] Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. UniViLM: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020. 2

[39] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2023. 2, 3, 6

[40] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proc. ICCV*, pages 2630–2640, 2019. 1, 2, 3, 4, 6, 7

[41] Ron Mokady, Amir Hertz, and Amit H Bermano. ClipCap: CLIP prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021. 2

[42] Jonghwan Mun, Linjie Yang, Zhou Ren, Ning Xu, and Bohyung Han. Streamlined dense video captioning. In *Proc. CVPR*, 2019. 2

[43] Alejandro Pardo, Fabian Caba Heilbron, Juan León Alcázar, Ali Thabet, and Bernard Ghanem. Moviecuts: A new dataset and benchmark for cut type recognition. In *European Conference on Computer Vision*, pages 668–685. Springer, 2022. 2

[44] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proc. ICML*, 2021. 2

[45] Tanzila Rahman, Bicheng Xu, and Leonid Sigal. Watch, listen and tell: Multi-modal weakly supervised dense event captioning. In *Proc. ICCV*, 2019. 2

[46] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *Proc. CVPR*, 2015. 2

[47] Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher Pal, Hugo Larochelle, Aaron Courville, and Bernt Schiele. Movie description. *IJCV*, 123(1):94–120, 2017. 2

[48] Paul Hongsuck Seo, Arsha Nagrani, Anurag Arnab, and Cordelia Schmid. End-to-end generative pretraining for multimodal video captioning. In *Proc. CVPR*, pages 17959–17968, 2022. 2

[49] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Association for Computational Linguistics*, 2018. 2

[50] Zhiqiang Shen, Jianguo Li, Zhou Su, Minjun Li, Yurong Chen, Yu-Gang Jiang, and Xiangyang Xue. Weakly supervised dense video captioning. In *Proc. CVPR*, 2017. 2

[51] Botian Shi, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and Ming Zhou. Dense procedure captioning in narrated instructional videos. In *Association for Computational Linguistics*, 2019. 2

[52] Yaya Shi, Xu Yang, Haiyang Xu, Chunfeng Yuan, Bing Li, Weiming Hu, and Zheng-Jun Zha. Emscore: Evaluating video captioning via coarse-grained and fine-grained embedding matching. In *Proc. CVPR*, pages 17929–17938, 2022. 3

[53] Nina Shvetsova, Anna Kukleva, Xudong Hong, Christian Rupprecht, Bernt Schiele, and Hilde Kuehne. HowToCaption: Prompting LLMs to transform video annotations at scale. *arXiv:2310.04900*, 2023. 1, 2, 4

[54] Mattia Soldan, Alejandro Pardo, Juan León Alcázar, Fabian Caba, Chen Zhao, Silvio Giancola, and Bernard Ghanem. MAD: A scalable dataset for language grounding in videos from movie audio descriptions. In *Proc. CVPR*, 2022. 1, 2

[55] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Xun Guo, Tian Ye, Yan Lu, Jenq-Neng Hwang, and Gaoang Wang. MovieChat: From dense token to sparse memory for long video understanding. *arXiv:2307.16449*, 2023. 2, 3

[56] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023. 5

[57] Makarand Tapaswi, Martin Bauml, and Rainer Stiefelhagen. Book2movie: Aligning video scenes with book chapters. In *Proc. CVPR*, pages 1827–1835, 2015. 2

[58] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4631–4640, 2016. 2

[59] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv:2302.13971*, 2023. 2

[60] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2, 5, 6

[61] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proc. CVPR*, pages 4566–4575, 2015. 2, 3, 7

[62] Jingwen Wang, Wenhao Jiang, Lin Ma, Wei Liu, and Yong Xu. Bidirectional attentive fusion with context gating for dense video captioning. In *Proc. CVPR*, 2018. 2

[63] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022. 3

[64] Teng Wang, Huicheng Zheng, Mingjing Yu, Qian Tian, and Haifeng Hu. Event-centric hierarchical representation for dense video captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020. 2

[65] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. In *Proc. ICCV*, 2021. 2, 3

[66] Yujia Wang, Wei Liang, Haikun Huang, Yongqi Zhang, Dingzeyu Li, and Lap-Fai Yu. Toward automatic audio description generation for accessible videos. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2021. 1

[67] Yu Xiong, Qingqiu Huang, Lingfeng Guo, Hang Zhou, Bolei Zhou, and Dahua Lin. A graph-based framework to bridge movies and synopses. In *Proc. ICCV*, pages 4592–4601, 2019. 2

[68] Antoine Yang, Arsha Nagrani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. Vid2seq: Large-scale pretraining of a visual language model for dense video captioning. *arXiv preprint arXiv:2302.14115*, 2023. 2, 3

[69] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of LMMs: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9, 2023. 4

[70] Yanzhi Yi, Hangyu Deng, and Jinglu Hu. Improving image captioning evaluation by considering inter references variance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 985–994, 2020. 3

[71] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive captioners are image-text foundation models. *Transactions on Machine Learning Research*, 2022. 1, 2

[72] Keunwoo Peter Yu. VideoBLIP, 2023. 2, 3, 5, 8, 13

[73] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. In *NeurIPS*, 2023. 3

[74] Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. MERLOT: Multimodal neural script knowledge models. In *NeurIPS*, 2021. 1

[75] Rowan Zellers, Jiasen Lu, Ximing Lu, Youngjae Yu, Yanpeng Zhao, Mohammadreza Salehi, Aditya Kusupati, Jack Hessel, Ali Farhadi, and Yejin Choi. MERLOT reserve: Multimodal neural script knowledge through vision and language and sound. In *CVPR*, 2022. 2

[76] Chaoyi Zhang, Kevin Lin, Zhengyuan Yang, Jianfeng Wang, Linjie Li, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. Mm-narrator: Narrating long-form videos with multimodal in-context learning. *arXiv preprint arXiv:2311.17435*, 2023. 8

[77] Hang Zhang, Xin Li, and Lidong Bing. Video-LLaMA: An instruction-tuned audio-visual language model for video understanding. In *EMNLP 2023 Demo*, 2023. 2, 3, 5, 8, 13

[78] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022. 2, 5

[79] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with bert. In *Proc. ICLR*, 2020. 2, 3

[80] Yue Zhao, Ishan Misra, Philipp Krähenbühl, and Rohit Girdhar. Learning video representations from large language models. In *CVPR*, 2023. 2

[81] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. 2, 3, 6

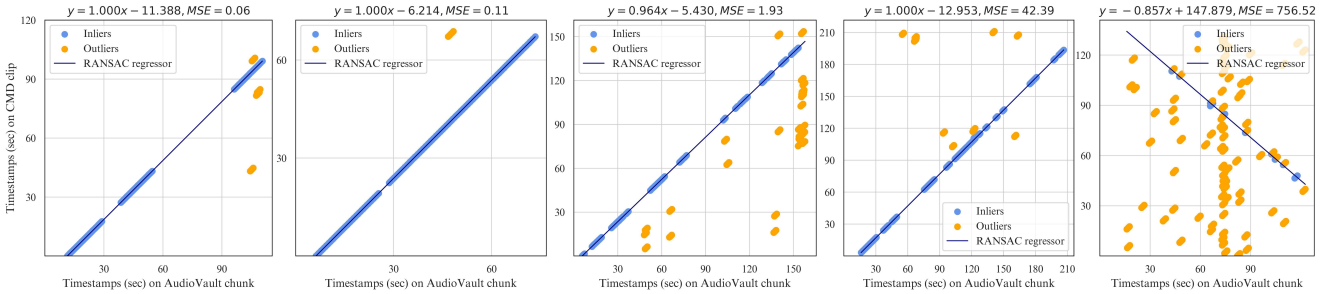
[82] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI*, 2018. 2

[83] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proc. CVPR*, 2018. 2, 3

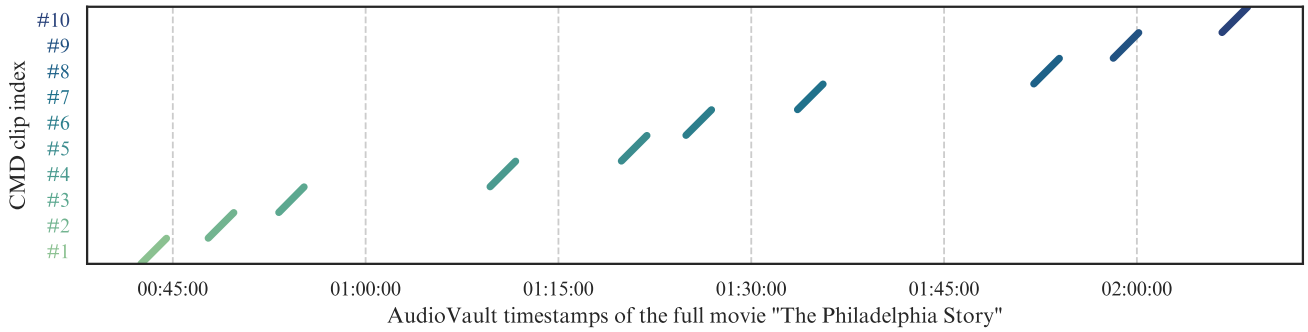
[84] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023. 2, 3, 5

AutoAD III: The Prequel – Back to the Pixels

Appendix



(a)



(b)

Figure A.1. **(a)** More examples of RANSAC for audio-audio alignment. Five samples are chosen from five different movies. The four samples on the left are successful RANSAC results that have low MSE and reasonable slopes. The sample on the right is filtered out since the fitted result from RANSAC has a high MSE and the slope does match our criteria. Note that the inliers (blue dots) are not continuous because those time segments correspond to AD audio and have been masked out before the alignment process. **(b)** Audio-audio alignment results of a full movie. We localize ten CMD clips from the corresponding *full-movie* AudioVault audio file by visualizing their aligned timestamps from the RANSAC algorithm. We can clearly see that CMD dataset contains multiple discontinuous movie clips, but the precise timestamps of each clip are obtained from our alignment algorithm, which are not provided by CMD. The example movie is 'The Philadelphia Story' (1940, `tt0032904`).

A. Dataset Details

A.1. CMD-AD – Pixels from Aligned CMD

Following the main paper Figure 2, a few more examples of RANSAC output are shown in Figure A.1-(a). Note that we filter valid RANSAC output by two thresholds: (1) the slope $0.8 < W' < 1.25$, and (2) MSE< 100. Additionally, we visualize the audio-audio alignment results of a *full movie* example in Figure A.1-(b).

The script for conducting audio-audio alignment between CMD movie clip and AudioVault audio chunk is shown in Algorithm 1.

A.2. HowTo-AD – Pixels from HowTo100M

The original HowTo100M and HowToCaption datasets have more than 1.2M videos. We apply a sequence of filtering processes to get high-quality videos for our purpose. (i) we use the `spacy` package in Python to detect unique *names* in the YouTube ASR. We discard videos with more than 5 unique names, because we find those videos typically correspond to News or Sports programmes and are not suitable to be transformed to movie AD style. (ii) we use the `spacy` package to detect the *subject* of each sentence in the HowToCaption dataset. We discard those videos which do not contain a subject in any of the captions, *e.g.* the captions like 'cut carrots', 'chunk onion', *etc.* Because our caption transforming stage in the main paper Section 3.2 requires the detected subjects. (iii) we extract a few video frames at the introduction of the video – which is inferred from the first few ASR timestamps – then we use a face detector[5] to verify whether a face exists in these frames. We discard videos that do not have frames with faces because our character bank requires face exemplars.

A combination of these filters reduces the number of videos

[5] https://github.com/ageitgey/face_recognition

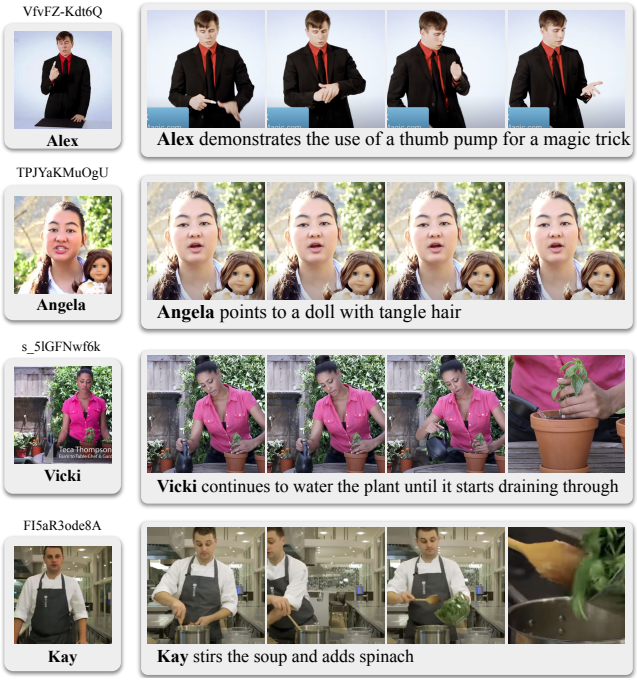


Figure A.2. More samples from HowTo-AD dataset. Four examples are shown in this figure with the 'character bank' shown on the left and the re-written video caption with randomly sampled character names shown on the right. The YouTube IDs are displayed on top of the character banks.

from 1.2M to 180k, from which we build the HowTo-AD dataset as described in the main paper Section 3.2.

In Figure A.2, we show a few more examples from HowTo-AD dataset. We notice a few limitations of the HowTo-AD: (1) inevitably, the video categories are dominated by 'Food and Entertaining', in other words, cooking activities. Since the dataset originates from HowTo100M where 50% of the categories belongs to cooking activities. A breakdown of category statistics is shown in Figure A.3. (2) our current pipeline that transforms HowToCaption into HowTo-AD assumes the video only has a single character. However, it's common that in instructional videos, two or three instructors appear in the scene. The single-character assumption may introduce noisy AD in the HowTo-AD dataset, and we consider fixing this in future work.

B. Implementation Details

B.1. Architectures

Movie-Llama2 is based on the visual branch of the Video-Llama [77] repository[6]. The input to the model is 8 movie frames uniformly sampled from a given AD time segment. The model contains five components, as detailed below. (i) The visual encoder is from EVA-CLIP which takes input from $224 \times 224 \times 3$ pixels and uses 14×14 patch size. The

[6] https://github.com/DAMO-NLP-SG/Video-LLaMA

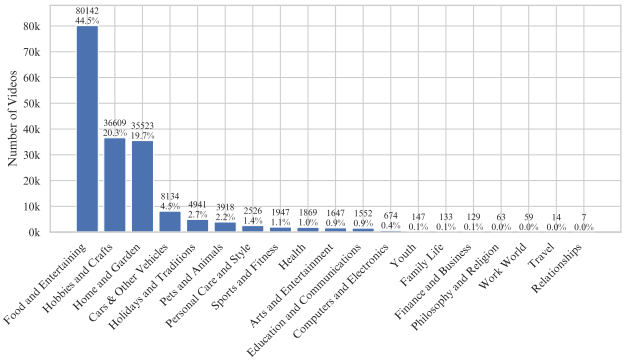


Figure A.3. Category distribution of HowTo-AD dataset. Similar to the original HowTo100M dataset, the HowTo-AD dataset has a long-tailed distribution over action categories, dominated by 'Food and Entertaining', 'Hobbies and Crafts' and 'Home and Garden'.

architecture contains 39 transformer blocks with a consistent hidden dimension of 1408 channels and 16 heads. (ii) The Q-former is from BLIP-2, which contains 12 transformer blocks with 32 learnable queries and cross-attends visual features with a dimension of $(1+16^2) \times 768$. For each image input, it produces features with a dimension of 32×768. (iii) The video Q-Former is from Video-Llama, which also contains 12 transformer blocks with 32 learnable queries. It cross-attends video features with a dimension of $8 \times 32 \times 768$ and produces 32×768 video features. (iv) The projector is a linear layer that projects 32×768 visual features into 32×4096 dimension to fit the language model. We have two projectors for video and image inputs respectively. (v) The language model is Llama2-7B, which contains 32 transformer blocks with 4096 hidden dimensions.

By default, we only train the parameters in the video Q-Former described in (iii) and the projector described in (iv). The rest of the components are initialized from Video-Llama pretrained weights. We also experimented with a setting that only trains the linear projector (iv), it produces a lower performance (CIDEr drops by 1.5).

Movie-BLIP2 is based on the Video-BLIP [72] repository[7]. With an input of 8 movie frames, the architecture is mostly the same as Video-Llama, except for two differences. First, the functions of the Q-former and video Q-Former are merged. In Movie-BLIP2, the Q-former cross-attends to *all* the visual features at once with a dimension of $8 \times (1+16^2) \times 1408$, then produces video features 32×768. There is no *video* Q-Former in Movie-BLIP2. Second, the language model is OPT-2.7B, which contains 32 transformer blocks with 2560 hidden dimensions.

Similarly, we train the parameters in the Q-former and the projector. The rest of the components are initialized from Movie-BLIP pretrained weights. But only training the projector produces a similar performance.

[7] https://github.com/yukw777/VideoBLIP

AudioVault **version 1**	AudioVault **version 2**
• Donny turns to glare at the back of his mother's chair. • He paces the room. • Donnie marches purposefully out of the room.	• ~~Donny~~Donnie turns to glare at the back of his mother's chair. • He paces the room. • Donnie marches purposefully out of the room.
• He picks up the receiver and looks at his New York calling card. • He pulls the grouping of four chairs closer and props his feet on them. • The officer checks the form and sets it aside.	• He picks up ~~the~~a receiver and looks at his New York calling card. • He sits up and pulls the ~~grouping of four chairs closer and props his feet on them~~entire seat unit closer to himself. • The officer ~~checks~~glances at the form and sets it aside.

Figure A.4. Special cases of inter-rater analysis: Two versions of AD from AudioVault might not be produced independently. In these two examples, annotators edit a few words from the earlier version, to produce a new version of AD. We *discard* these samples when conducting inter-rater analysis since these examples do not reflect real human-human agreement on AD.

	AudioVault annotations	Movie-Llama2 predictions	Metrics per sentence	
Sherlock Holmes	1. Holmes follows her around a corner. 2. Mary is in her late thirties with a mop of curly red hair under her black veil. 3. Dazed, he lifts up his head. 4. His vision is blurred, but he sees a man scoop Mary up off the road and drag her towards the carriage.	1. Holmes follows her. 2. The carriage pulls up outside a narrow alleyway. 3. He lies on the ground, his head twisted at an odd angle. 4. He falls to the ground.	CRITIC (0-1)	LLM-AD-Eval (0-5)
			1.0	5
			0.0	0
			1.0	4
			0.33	4
Back to the Future	1. Marty's speedometer tops at a stop sign, he turns hard right. 2. Marty's radiation helmet falls over his head as the car zooms in the barn door. 3. George rides through the square on a bicycle, and Marty bursts out of the cafe after him.	1. Marty's car speeds down the road. 2. The car speeds through the barn and out the other side. 3. Marty rides his bike out of the diner.	CRITIC (0-1)	LLM-AD-Eval (0-5)
			1.0	2
			0.0	2
			0.5	4

Figure A.5. Examples of CRITIC and LLM-AD-Eval metrics. We show seven AD annotation-prediction pairs from two movies. For both CRITIC and LLM-AD-Eval metrics, higher values indicate higher quality. For details of metrics please refer to the main paper Section 5.

B.2. LLM-AD-Eval

We use the `gpt-3.5-turbo` model from OpenAI to construct the LLM-AD-Eval metric. Our customized prompt is shown in Algorithm 2. The LLM is instructed to evaluate the level of match between two sentences and return a score between 0 to 5.

C. Additional Results

C.1. Details of Inter-rater Analysis on AudioVault

As described in the main paper Section 6.3, there are 402 movies with multiple versions of AD annotations. However, we found in some cases two versions of AD annotations are not produced *independently*, as shown in Figure A.4 – annotators slightly modify an older version of AD to produce a new version, therefore these samples do not reflect the real human-human agreement on AD annotations. To discard these samples in our inter-rater analysis, we apply a filter to detect the identical AD annotations for each movie and discard those movies with more than one identical annotations. It ends up filtering out 18% from 402 movies, and the remaining trustworthy independent AD annotations are used for inter-rater analysis.

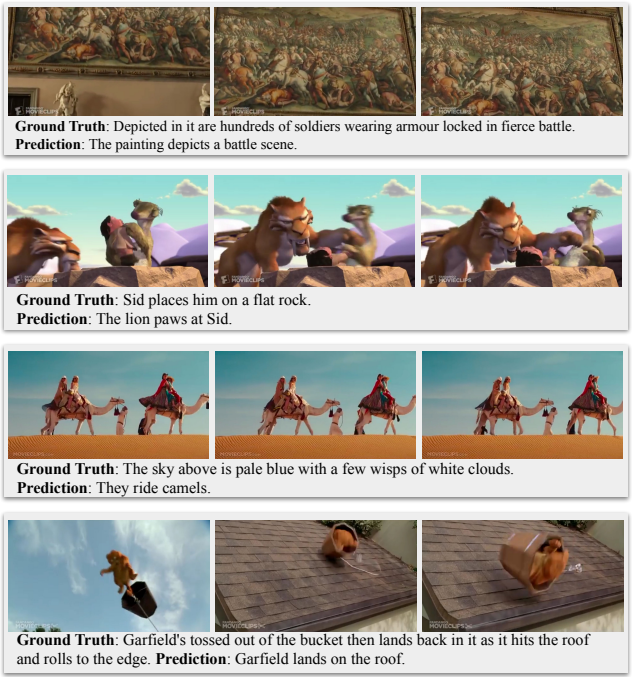


Figure A.6. Additional qualitative results. For each movie clip, three frames are uniformly extracted and visualized. The movie clips are from Inferno (2016, `tt3062096`), Ice Age (2002, `tt0268380`), Sex and the City 2 (2010, `tt1261945`) and Garfield (2004, `tt0356634`). For qualitative results in video format, please refer to the `mp4` file in our supplementary material.

C.2. Analysis of the Evaluation Metrics

A few examples of AD annotations and (both good and bad) model predictions are shown in Figure A.5, we show both CRITIC and LLM-AD-Eval metrics for each pair. Qualitatively, we can feel that LLM-AD-Eval correlates to human judgement of semantic matching between two sentences. The CRITIC metric gives zeros when the predictions do not contain the characters in the annotation.

C.3. Better Character Exemplars for CMD-AD

In AutoAD-II [21], the character exemplars are constructed from the in-context nearest neighbours of actors' IMDb profile pictures. However, we qualitatively find this method does not work well for old movies where the IMDb profile pictures are of low quality, or cartoon movies where the IMDb profile pictures are the voice actors instead of the characters.

We investigated an additional source of character exemplars, which are from the 'photo' pages on IMDb, like `https://www.imdb.com/title/tt2294629/characters/nm0068338` or `https://www.imdb.com/title/`

Method	Setting	CIDEr	CRITIC
Movie-Llama2	new exemplars	**21.7**	**25.2**
Movie-Llama2	old exemplars	20.1	23.3

Table A.1. **Effect of new character exemplars** on CMD-AD-Eval. Both experiments are conducted without HowTo-AD pretraining.

tt0046912/characters/nm0001537. These photos are the movie frames uploaded by users, corresponding to specific characters. However, this source has two major issues: first, the photos are biased towards famous characters, and less-known characters usually do not have photos uploaded; second, photos under one character may contain other characters. We find only 50% characters have valid photo pages, and we use face detector[8] to select photos with only one face; For those without photo pages, we fall back to the original AutoAD-II strategy to find in-context exemplars. The main paper uses this new strategy for character exemplars for all the experiments. We ablate the effect of this update on the character exemplar in Table A.1.

D. Additional Qualitative Results

More qualitative results are shown in Figure A.6. Additionally, on the project page https://www.robots.ox.ac.uk/vgg/research/autoad/, we display a few visualization of our model predictions as MP4 files. The AD ground-truth and our model predictions are provided as subtitles with the format '[title] ground-truth ‖ prediction'. The voice of AD is generated from OpenAI text-to-speech API[9] and fused with the original movie soundtrack.

[8] https://github.com/ageitgey/face_recognition

[9] https://platform.openai.com/docs/guides/text-to-speech

Algorithm 1: Python script for audio-audio alignment.

```
import torch
import torchaudio
import numpy as np
from sklearn.linear_model import RANSACRegressor
from sklearn.metrics import mean_squared_error

def aa_match(cmd_wav, audiovault_wav, audiovault_anno):
    """
    cmd_wav: audio path of CMD
    audiovault_wav: audio path of AudioVault chunk,
    audiovault_anno: AD annotations (with timestamps) of the AudioVault chunk

    Return: mse, slope and intercept of RANSAC
    """
    resolution_per_mel = 512/16000

    cmd_waveform, cmd_sr = torchaudio.load(cmd_wav)
    cmd_melspec = mel_spectrogram(cmd_waveform)
    cmd_melspec = cmd_melspec / cmd_melspec.norm(dim=-2, keepdim=True)

    av_waveform, av_sr = torchaudio.load(audiovault_wav)
    av_melspec = mel_spectrogram(av_waveform)
    av_melspec = av_melspec / av_melspec.norm(dim=-2, keepdim=True)

    # mask melspec as zeros if AD appears. Only compute non-AD part
    av_melspec_masked = torch.clone(av_melspec)
    start_end_mel_idx = []
    for _, row in audiovault_anno.iterrows():
        start_mel_idx = int(row['start'] / resolution_per_mel) # second to mel idx
        end_mel_idx = int(row['end'] / resolution_per_mel)
        av_melspec_masked[...,start_mel_idx:end_mel_idx] = 0
        start_end_mel_idx.append([start_mel_idx, end_mel_idx])

    window_size = 50 # equal to 3.2 second

    # use 1D conv to compute correlation in parallel
    # use unfold() to get chunks by moving windows
    melspec_av_all_cuts = av_melspec_masked.unfold(dimension=-1, size=window_size, step=window_size)
    melspec_unfold = cmd_melspec.unfold(dimension=-1, size=window_size, step=1)
    conv_output = torch.einsum('bctw,bcaw->bta', melspec_unfold, melspec_av_all_cuts)
    ad_mask = torch.any(melspec_av_all_cuts.mean(1)==0, dim=-1)
    conv_output = conv_output[0, :, ~ad_mask[0].bool()] # only keep non-AD chunks
    max_cor, max_pos = conv_output.max(0)
    plot_outputs = torch.stack(
        [torch.arange(0, av_melspec_masked.shape[-1]-window_size+1, window_size)[~ad_mask[0].bool()],
         max_pos, max_cor/window_size], -1).numpy()

    # prepare scatters for RANSAC
    peak_cor = np.max([i[-1] for i in plot_outputs])
    samples_X = (plot_outputs[:, 1:2] + np.linspace(0, window_size, 5)[None,:]).flatten()
    samples_y = (plot_outputs[:, 0:1] + np.linspace(0, window_size, 5)[None,:]).flatten()
    samples_weight = plot_outputs[:, 2:3].repeat(5, axis=1).flatten()

    ransac = RANSACRegressor(random_state=0, residual_threshold=50).fit(
        samples_X[:,None], samples_y[:,None], samples_weight)

    inlier_mask = ransac.inlier_mask_
    outlier_mask = np.logical_not(inlier_mask)
    slope = ransac.estimator_.coef_[0,0]
    intercept = ransac.estimator_.intercept_[0]

    # get mse
    y_pred = ransac.predict(samples_X[inlier_mask][:,None])
    mse = mean_squared_error(samples_y[inlier_mask][:,None], y_pred)

    print('MSE=', mse)
    print('slope=', slope)
    print('intercept=', intercept)
    return mse, slope, intercept
```

Algorithm 2: Python script for LLM-AD-Eval, with OpenAI gpt-3.5-turbo API.

```
import ast
import openai

def eval_each(text_gt, text_pred, client):
    # Compute the LLM-AD-Eval score
    completion = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {
                "role": "system",
                "content":
                    "You are an intelligent chatbot designed
                         for evaluating the quality of generative outputs for movie audio descriptions. "
                    "Your task is to compare the predicted audio descriptions with the correct audio
                        descriptions and determine its level of match, considering mainly the visual elements
                         like actions, objects and interactions. Here's how you can accomplish the task:"
                    "------"
                    "##INSTRUCTIONS: "
                    "- Check if the predicted audio description covers
                         the main visual events from the movie, especially focusing on the verbs and nouns.\n"
                    "- Evaluate whether the predicted audio
                         description includes specific details rather than just generic points. It should
                         provide comprehensive information that is tied to specific elements of the video.\n"
                    "- Consider synonyms or
                         paraphrases as valid matches. Consider pronouns like 'he' or 'she' as valid matches
                         with character names. Consider different character names as valid matches. \n"
                    "- Provide a single evaluation score that reflects the level of match of the
                         prediction, considering the visual elements like actions, objects and interactions."
            },
            {
                "role": "user",
                "content":
                    "Please evaluate the following movie audio description pair:\n\n"
                    f"Correct Audio Description: {text_gt}\n"
                    f"Predicted Audio Description: {text_pred}\n\n"
                    "Provide your evaluation only as a matching score where the matching score
                         is an integer value between 0 and 5, with 5 indicating the highest level of match. "
                    "Please generate the response in the form of a Python dictionary string
                         with keys 'score', where its value is the matching score in INTEGER, not STRING."
                    "DO NOT PROVIDE
                         ANY OTHER OUTPUT TEXT OR EXPLANATION. Only provide the Python dictionary string. "
                    "For example, your response should look like this: {'score': }."
            }
        ]
    )
    # Convert response to a Python dictionary.
    response_message = completion.choices[0].message.content
    response_dict = ast.literal_eval(response_message)
    return response_dict

client = OpenAI()
eval_each(text_gt, text_pred, client)
```