Open-Domain Video Commentary Generation

Edison Marrese-Taylor[1*], Yumi Hamazono[1,2*], Tatsuya Ishigaki[1], Goran Topic[1]
Yusuke Miyao[1,3] Ichiro Kobayashi[1,2], Hiroya Takamura[1]
National Institute of Advanced Industrial Science and Technology[1]
Ochanomizu University[2], The University of Tokyo[3]
{edison.marrese,ishigaki.tatsuya}@aist.go.jp
{goran.topic,takamura.hiroya}@aist.go.jp
{hamazono.yumi,koba}@is.ocha.ac.jp, yusuke@is.s.u-tokyo.ac.jp
[*]Authors contributed equally to this work.

Abstract

Live commentary plays an important role in sports broadcasts and video games, making spectators more excited and immersed. In this context, though approaches for automatically generating such commentary have been proposed in the past, they have been generally concerned with specific fields, where it is possible to leverage domain-specific information. In light of this, we propose the task of generating video commentary in an open-domain fashion. We detail the construction of a new large-scale dataset of transcribed commentary aligned with videos containing various human actions in a variety of domains, and propose approaches based on well-known neural architectures to tackle the task. To understand the strengths and limitations of current approaches, we present an in-depth empirical study based on our data. Our results suggest clear trade-offs between textual and visual inputs for the models and highlight the importance of relying on external knowledge in this open-domain setting, resulting in a set of robust baselines for our task.

1 Introduction

When watching TV programs or online live streams we find that the visual content is often accompanied by objective statements or subjective remarks about the events in the video given by a commentator in real time, aiming to help the spectators understand events in the videos. It has been shown that such live commentaries play an important role in sports broadcasts and video game matches or streams, for example, making spectators more excited, more immersed, and better informed about the content they are viewing (Schaffrath, 2003), thus enhancing the value of online and home videos (Ishigaki et al., 2021). In view of this, the task of automatically generating such live commentaries on specific domains, such as sports (Kim and Choi, 2020) or video games (Ishigaki et al., 2021) has been proposed recently, with models often relying on field-specific information to aid the generation. Live commentary, in this paper, refers to commentary that can be listened to as audio or read as subtitle together with the original visual content.

In light of this, in this paper we depart from previous work on automatic live commentary generation by proposing an open-domain setting, where the goal is to enable models to generate live commentaries for videos containing actions in a variety of situations. Compared with related work (Taniguchi et al., 2019; Kim and Choi, 2020) and in-domain versions of our task (Ishigaki et al., 2021), ours is particularly challenging as it keeps us from using domain-specific features, which have proven essential for attaining good performance.

Our task can be seen as a combination of fundamental tasks in Computer Vision (CV) and Natural Language Processing (NLP), like video captioning (Venugopalan et al., 2015; Zhou et al., 2018a), dense video captioning (DVC) (Krishna et al., 2017), and data-to-text (Puduppully and Lapata, 2021). The task is similar to video captioning, for example in the context of instructional "cooking" videos as there are no overlaps between the relevant video sections with key events occurring sequentially. However, it is critically different from previous work because each utterance is only partially aligned with the events in the video, both in terms of their temporal location and of the correlation between their content and the objects in the video. This partial alignment, which is due to multiple factors such as the presence of speech in the original video, attempts of the commentator to make the content more amenable, among others, is the key factor defining our task.

To benchmark progress on this newly-introduced task, we construct a new large-scale dataset of transcribed live commentaries aligned with videos containing various human actions in a variety of domains. Our dataset contains a total of 6,771 videos sampled from ActivityNet (Caba Heilbron et al.,

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 7326 - 7339
December 7-11, 2022

2015; Krishna et al., 2017), and annotated for our purposes using crowdsourcing, resulting in more than 25k timestamped commentary utterances containing information to help the spectators understand events in the videos.

Finally, we present a multi-modal approach to tackle our proposed task and accompanied by in-depth empirical study based on our data. In the absence of field-specific information due to the open-domain nature of our task, we show how existing models can be successfully leveraged, highlighting the importance of access to external knowledge, in our case in the form of pre-trained models, in order to attain better performance. Furthermore, our study sheds light on the temporal alignment issue that is fundamental to our task, suggesting clear trade-offs between the role of the textual and visual modalities, as well as limitations between on-line and off-line settings.

We hope our work helps provide a concrete direction for further research on commentary generation tasks, both in the in-domain and our newly-proposed open-domain settings. Our dataset and models are available at `github.com/epochx/live-commentary`.

2 Related Work

Developing machine learning techniques that can utilize language to describe what happens in a video remains an open challenge, which lies at the core of both CV and NLP. In this context, seminal work by Venugopalan et al. (2015) was, to the best of our knowledge, the first to tackle the task of describing videos in an open-domain setting. In their proposed video captioning task (VC), a system had to generate a natural language description of the main activity on a given short video (<30 s).

This video captioning task was later extended by Krishna et al. (2017) who proposed the task of dense video captioning, where a model is required to detect multiple events that occur in a video, and then describe each one using natural language. Current approaches generally focus on two separate sub-tasks. First, there is a temporal event proposal step, where relevant segments in the input video are first identified, after which a captioning model generates the natural language descriptions of the identified zones (Wang et al., 2020; Deng et al., 2021). However, end-to-end approaches based on the Transformer (Vaswani et al., 2017) have been also recently proposed (Zhou et al., 2018b; Wang et al., 2021).

In the context of automatically generating commentary for sports matches, Taniguchi et al. (2019) used play event data (i.e., structured data about the involved player, the location in the pitch, the type of play, etc.) to generate live soccer-match commentaries to be displayed on a web page, in a data-to-text generation setting. Also, Kim and Choi (2020) recently proposed a system to automatically generate summarized commentary for baseball games, directly from match videos, which are to be displayed during short breaks between plays. Our approach is therefore different from these systems as they aim to generate commentaries that are not supposed to be displayed alongside the video content. Crucially, we also note they rely on domain-specific data for solving their respective tasks. While this is obviously the case for the former, we observe that the latter is based on several sub-components that aim at modeling specifics of the game from the input video —including a player detector and a pitching result recognizer— in addition to relying on a domain ontology for the generation step.

More recently, Ishigaki et al. (2021) proposed the task of automatically generating commentaries for racing car videogame streams, releasing the first annotated dataset of gameplay videos aligned with transcribed spoken commentaries. In this context, their main contribution is the release of structured data, that is, telemetry data, including the positions and the speeds of cars, and the steering wheel angles as extracted directly from the video game engine, which they leverage on their proposed approach. Their results show that utilizing these features, it leads to substantial performance improvements over models that only receive visual and textual inputs, again showing the importance of in-domain information for the task.

Finally, our work is also related to simultaneous translation (Cho and Esipova, 2016). Specifically, our task is similar to this in the real-time scenario (please see Section 4 for details), as it also requires the model to account for the trade-off between waiting in order to have more information to caption, and providing a caption with a shorter delay.

3 Dataset

To benchmark progress on our proposed task, we present a new large-scale dataset of transcribed live commentaries aligned with videos. Since our

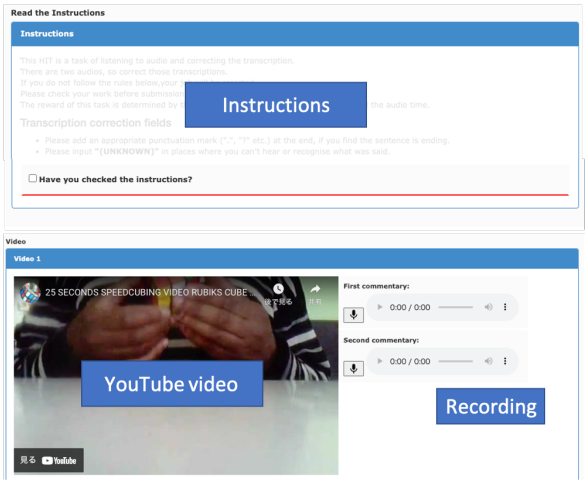

Figure 1: Annotation interface where each worker can record their commentary for a given video, also showing the set of initial instructions we provided.

aim is to cover a large variety of actions, we rely on data from the popular ActivityNet benchmark (Caba Heilbron et al., 2015), which contains videos showing various human actions in a variety of domains. We sample videos from both the training and validation splits, limiting their duration to be between thirty seconds and four minutes, and making sure we only use videos that were currently available on YouTube as of 2021.

3.1 Collecting Commentary Annotations

To collect video commentary annotations, we resorted to crowdsourcing using Amazon Mechanical Turk (AMT). The overall process can be divided into three sub-tasks, which we detail below.

Recording commentary Each crowdworker was presented with the interface shown in Figure 1. Before the data collection process began, we showed each crowdworker an example video with the kind of commentary we require, ensuring that all workers saw concrete evidence of what kind of work we expected. The workers were then specifically instructed to comment on what was happening in the video, including their subjective impressions, while watching the video.

The audio recording started together with the playback of the video, and workers were instructed to press the stop button after the video ended and they had nothing else to add. In case a worker did not stop the recording, it was designed so that the recording would be forcibly stopped one minute after the end of the video. Each worker was allowed to work on a maximum of two videos, and for each case we asked them to record their commentary twice. The first recording aimed to collect the data giving the annotator no previous knowledge about the video, which we argue is closer to what actual commentators experience on live TV or streams. However, since commentators are generally knowledgeable on the topics/events in the video, we also collected data after the annotators were already familiar with the content of the video, thus approximately simulating this scenario.

To improve the quality of the recordings collected, we adjusted their volume and removed background noise using Pydub[1]. We also filtered out cases of extreme discrepancies between the video and recording lengths, limiting the audio recording time to be only between 90% and 200% of the corresponding video time.

Transcribing and labeling timestamps Once we obtained a clean set of commentary audios, we proceeded to transcribe them to text using Amazon Transcribe[2], which offers the speech-to-text functionality we need in the same platform, while attaining excellent performance with a WER of 11.76% (Xu et al., 2021) on the Microsoft Scalable Noisy Speech Dataset (Reddy et al., 2019). Amazon Transcribe automatically splits each input speech into separate utterances and provides us with token-level start and end timestamps for each. We label the start and end time of each utterance using the corresponding token timestamps.

To exclude low-quality recordings with extremely long periods of silence, we compute the total commentary duration based on these time labels, and only keep examples whose duration was between 30% and 110% of the video length.

Overall, from a total of 46,588 recordings we collected, we removed 165 that were much shorter or longer than the input video duration, 112 recordings where the background noise was too loud, or nothing was recorded, and 1,137 recordings because they contained the extremely long periods of silence.

Correcting the transcripts Since the results of the automatic speech-to-text may contain errors, and the utterances may contain fillers and rephrasing, we again resorted to AMT to ensure the quality of our obtained annotations by asking workers to review and correct the automatic transcriptions.

[1] https://github.com/jiaaro/pydub
[2] https://aws.amazon.com/transcribe

To this end, we presented each annotator with the processed commentaries allowing them to listen to the audio sections for each utterance and correct the transcripts by removing fillers, rephrasing or correcting ungrammatical passages. Workers for this phase were assigned randomly and did not necessarily match assignments for the first round. During this second phase, we also asked the annotators to use a special marker whenever they found unintelligible passages. We later showed these to different annotators, requesting them to replace the markers with the appropriate content.

To adequately compensate our annotators, we first studied how long they would require to perform our tasks. For the first round, our preliminary experiments showed that each worker would need approximately 16 minutes to annotate 2 videos. Based on this, we decided to pay our workers \$3/task. For the second step, we used a selection of workers with different qualifications depending on the task. We compensated these workers accordingly in the range between \$0.5 to \$5, leading to an average compensation of \$1.1 per task. These compensations were obtained by means of annotation trials using shorter speech inputs, where we found that correcting the transcripts took roughly 2 to 2.5 longer than the input recording. Finally, we also observed that the crowdworkers had a tendency to produce short recordings, and to generate transcripts with few words, so we offered them bonuses for longer recordings and/or more wordy commentaries.

Overall, the crowdsourcing processes took approximately 10 days and 1.5 months respectively. Including the preparation time, it took almost five months in total. Excluding taxes, user fees and the cost of testing, the recording and transcription annotation cost approximately USD 34,000 each.

3.2 Dataset Overview

Inter-annotator agreement To verify that our workers have provided semantically-meaningful and relevant commentary, we follow previous work by Krishna et al. (2017) and compare the annotated timestamps of each utterance from a given worker against the maximal overlapping combination of utterances from other annotators. Our analysis is consistent with previous findings (Baldassano et al., 2017) showing that workers generally agree with each other when annotating temporal boundaries, with a temporal intersection over union (tIoU) of 45.5%. Though this value is slightly lower than the 59.5% tIoU of ActivityNet Captions[3], we believe this result is normal due to the nature of our task, where each commentator may decide to talk at a slightly different time.

These results are also supported by the inter-annotator agreement between our data and ActivityNet Captions, where we obtained an average tIoU of 0.187, which we think again suggests a lack of alignment between the actions appearing in the video and the utterances because of the commentator delaying or rushing his statements to make the content more clear.

Finally, we also study inter-annotator agreement in terms of commentary content. For this, we rely on BLEU-4 and SPICE (Anderson et al., 2016), an automated caption evaluation metric defined over scene graphs, commonly used when evaluating tasks like DVC. We specifically use these metrics to compare the set of utterances for a pair of annotators in a given video, sorted by time. We perform this study on the validation split our data and of ActivityNet Captions, which is also contains two different sets of annotations. For ActivityNet Captions, we obtain mean values of 4.86 and 0.12 for BLEU-4 and SPICE respectively, while for our dataset we obtained 2.59 and 0.034 respectively, which we again think reflect the nature of our proposed task. Please see Section A.2 in our supplementary material for additional details of this study, including a qualitative analysis.

Statistics As Table 1 shows, annotation efforts allowed us to collect more than 25k commentaries from 715 workers, covering a total of 6,771 videos. Figure 2 shows an example of the utterances provided by two annotators for a video, also compared to the dense captions from Krishna et al. (2017). As the example suggests, our annotations are very densely distributed on the videos, covering 78 $\pm$ 21% of the video duration on average, which compares to 55 $\pm$ 17% for the in-domain data released by Ishigaki et al. (2021), which we regard as the closest to our setting. We also note that the average number of utterances per video in our data (17.64) is much smaller than in the in-domain setting (52.25), also with shorter silence gaps between them, which we surmise is due to the shorter nature of our open-domain videos.

[3]We note that we could not independently obtain their reported value of 70.2%.

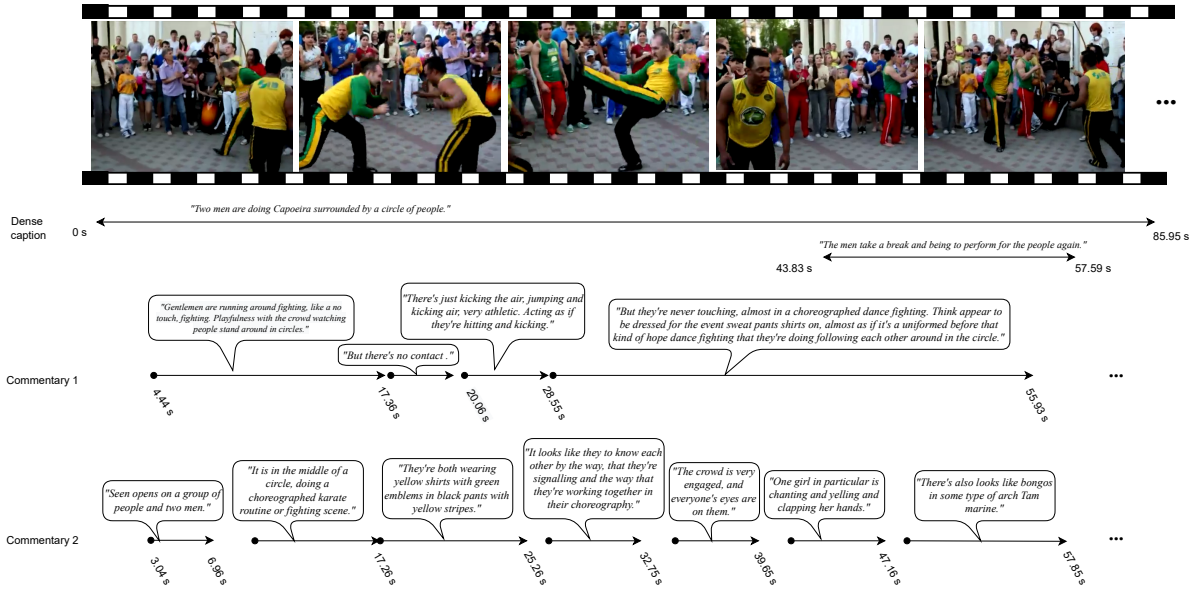

Figure 2: Example of an annotated video in our dataset, showing the input provided by two different crowdworkers, and compared to the dense annotations from Krishna et al. (2017).

Task	**DVC**	**Commentary Generation**	
Domain	**Open Domain**		**In Domain**
Data	Krishna et al.	Ours	Ishigaki et al.
Videos	14,926	6,771	2,473
- Total duration	487:35:43	239:58:03	226:37:53
- Max duration	755 s	238 s	452 s
- Min duration	1 s	30 s	246 s
Annotations	19,811	25,817	2,473
- Total # annotators	-	715	5
- Total ann. duration	742:23:29	904:01:24	226:37:53
- Total # of annotations	71,957	455,485	129,226
- Avg. # of sent. per ann.	3.63	17.64	52.25
- % of ann. with overlap	10%	0 %	0%
- Avg. length of silence	-	0.84 s	3.46 s

Table 1: Statistics of our open-domain video commentary dataset, compared to ActivityNet Captions (Krishna et al., 2017) for DVC, and the in-domain dataset by Ishigaki et al. (2021).

4 Proposed Approach

In this section, we discuss our proposed models to tackle the task of open-domain commentary generation. In the following, we assume that a given video $V \in \mathcal{V}$ can be characterized as a sequence of frames such that $V = \{v_f\}_{f=1}^{F}$. Each video in $\mathcal{V}$ is annotated with a sequence of natural language utterances $U = \{u_t\}_{t=1}^{T}$ containing live commentary of what is happening in a certain period of time defined by non-overlapping intervals with start and end timestamps t_s and t_e respectively.

We propose two different settings: (1) *Off-line Live Commentary Generation*: automatic commentary generation of pre-recorded videos, where the model has access to the whole video during both training and inference, and (2) *Real-time Live Commentary Generation*: automatic commentary generation for a live video, where the models need to balance the trade-off between waiting for additional data and generating an utterance without a too long delay. This is a more challenging setting that can be regarded as multi-modal analogous to simultaneous translation. Though in this paper we focus on the off-line setting, our empirical study is also concerned with challenges that we foresee will be critical for the on-line version, as we will show in Section 5.

Previous work on related tasks (Taniguchi et al., 2019; Kim and Choi, 2020) and on in-domain versions of our task (Ishigaki et al., 2021) have traditionally divided the efforts into simpler sub-tasks that can be combined into a pipeline approach. We follow a similar approach, also presenting two sub-tasks.

- **Timing Identification:** The task concerned with identifying suitable start and end timestamps in the video for each utterance. Given the open-domain nature of our task, we believe in order to solve it correctly, models should be able to recognize relevant events in the video at each time-point for a given granularity, which can act as anchors for the commentaries.

- **Utterance Generation:** The task of generat-

ing the appropriate utterance for a given input segment of the video.

In the following sub-sections, we propose models to tackle these sub-tasks using the first round of annotations we obtained, leaving further analysis on the second round for future work. In the rest of the paper, for clarity we focus on a given input example V. In all cases, we assume the existence of video encoder that maps the F input video frames into a sequence of video features $G = \{\boldsymbol{g}_n \in \mathbb{R}^{d_v}\}$, $n = 1, \ldots, N$, accompanied by a mapping function[4] that allows us to transform timestamps t into feature indices $\tau \in \{1, \ldots, N\}$.

4.1 Timing Identification

Previous work has extensively used in-domain information to perform tasks similar to timing identification. Due to the open-domain nature of our task, we believe that a key element in proposing such a model is access to external knowledge, either in the form of a formal knowledge base, or by utilizing a pre-trained model that has already captured information that can be useful for our task.

In light of this, here we propose to rely on Temporal Action Localization (TAL), a fundamental task in CV with aims at temporally localizing human activities in untrimmed video sequences. This task has traditionally been an essential component in DVC, where models often leverage ActivityNet (Caba Heilbron et al., 2015) for learning, a dataset containing videos labeled with more than 200 different activities. Models generally work by generating proposals, that is, relevant zones where the target activities may be located. We denote this set of proposals as $\{p_m\}_{m=1}^{M}$, with $p_m = (t_m^s, t_m^e, c_m)$ where t_m^s, t_m^e are predicted start and end times respectively, and c_m captures model certainty.

Video sections annotated on datasets such as ActivityNet may overlap substantially due to the semantics of the activity labels. As a result, the proposals will also overlap, which is not directly suitable for our task. We are then interested in selecting a series of segments proposed by a given TAL model which are highly correlated to our annotated segments, therefore also reducing the overlap between them. To that end, we adapt the approach by Mun et al. (2019) to our setting. Our model receives an input set of already-extracted proposals $\mathbb{P} = \{p_m\}_{m=1}^{M}$, encodes them and iteratively selects the most relevant subset by using a Pointer Network (Vinyals et al., 2015).

For a given proposal p_m, we map the timestamps to feature indices τ_m^s and τ_m^e with $1 \leq \tau_m^s < \tau_m^e \leq N$, and use G to represent p_m as a sequence of $K = \tau_m^e - \tau_m^s + 1$ features $\boldsymbol{g}_{\tau_m^s}, \ldots, \boldsymbol{g}_{\tau_m^e}$. We begin by obtaining a fixed-length representation for each proposal, denoted as $\boldsymbol{p}_m$. We do so as Equation (1) shows, below.

$$\boldsymbol{x}_k = \text{GRU}_E(\boldsymbol{g}_{\tau_m^s+k-1}, \boldsymbol{x}_{k-1}) \quad k = 1, \ldots, K, \quad (1)$$

where $\boldsymbol{x}_k$ is the hidden state of GRU_E at iteration k. We initialize GRU_E by setting $\boldsymbol{x}_0 = 0$, and define $\boldsymbol{p}_m = [\boldsymbol{x}_1; \boldsymbol{x}_K]$ as our segment representation, where ; denotes concatenation. We sort our segment representations following the starting times of each proposal and feed them through another GRU which is in charge of generating an initial contextualized representation for our pointer net, as shown in Equation (2), below:

$$\boldsymbol{h}_m = \text{GRU}_C(\boldsymbol{p}_m, \boldsymbol{h}_{m-1}) \quad m = 1, \ldots, M. \quad (2)$$

Again, the initial hidden state of GRU_C is set to zero ($\boldsymbol{h}_0 = 0$), and we use its last hidden state to initialize our Pointer Net, setting $\boldsymbol{s}_0 = \boldsymbol{h}_M$. We then iterate over the set of sorted proposals, one by one, performing the following operations:

$$score(\boldsymbol{s}_i, \boldsymbol{x}_m) = \boldsymbol{v}^T \tanh([\boldsymbol{s}_i; \boldsymbol{x}_m]), \quad (3)$$

$$a_{i,m} = \frac{\exp(score(\boldsymbol{s}_i, \boldsymbol{x}_m))}{\sum_{m=1}^{M} \exp(score(\boldsymbol{s}_i, \boldsymbol{x}_m))}, \quad (4)$$

$$\boldsymbol{s}_i = \text{GRU}_P([\boldsymbol{x}_{m^\star}; pos(m^\star)], \boldsymbol{s}_{i-1}), \quad (5)$$

where $\boldsymbol{v}$ is a vector of trainable parameters, $\boldsymbol{a}_i$ is an M-sized vector, $pos(m)$ is a function returning a binary vector of size d_p that captures the normalized location of proposal p_m, and $m^\star = \arg\max_m(a_{i,m})$ with $m^\star \in \{1, \ldots, M\}$. As shown in Equation (5), at each iteration we compute a probability distribution over the proposal set and greedily select the most likely proposal. To compute the hidden state of the Pointer Net for the next step, the representation of the selected proposal $\boldsymbol{x}_{m^\star}$ is then concatenated with a representation of its location and fed to GRU_P. We incorporate a special EOS proposal to the input set as p_0, and the model iterates until all the proposals have been selected or until the Pointer Net selects p_0.

Our model is trained to select proposals that have a high overlap with the ground truth annotations,

[4] We apply the mapping $\tau = (t \cdot n \cdot \text{fps})/N$ to transform frame/feature index to time.

minimizing the loss defined below:

$$\mathcal{L} = -\sum_{r=1}^{R}\sum_{m=1}^{M} \text{tIoU}(s_r, p_m)\log a_{r,m} + (1-\text{tIoU}(s_r, p_m))\log(1-a_{r,m}), \quad (6)$$

where $\{s_r\}_{r=1}^{R}$ is the sorted ground truth segment sequence extracted from our annotations, $a_{r,m}$ is the likelihood that the r^{th} proposal is selected at the m^{th} iteration, and $\text{tIoU}(\cdot,\cdot)$ is the temporal intersection over union.

4.2 Utterance Generation Model

Our utterance generation model is a Transformer model (Vaswani et al., 2017) which has been proven effective in several multi-modal tasks (Chen et al., 2020; Hong et al., 2021). We follow previous work and feed both visual and textual context into our model. Concretely, our model operates on a sequence of tokens $\{s_j\}$, which comes from the previous utterance, and a video segment $\{v_t\}$ which spans from 3 seconds before each gold-standard annotation, until its end (t_s-3, t_e). We regarded this it was a reasonable amount of context compared to Ishigaki et al. (2021) given that the average time gap between utterances in our data was approximately 1 second. The model is trained to minimize the cross entropy between the generated and gold-standard token sequences, and has the following main components.

Video Encoder As done for the Timing Identification task, a given video segment with timestamps (t_s, t_e) is mapped to τ_m^s, τ_m^e, and we use G to represent it as a sequence of $K = \tau_m^e - \tau_m^s + 1$ features $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_K$.

Text Encoder The module in charge of mapping the input text into a sequence of vectors. Concretely, sentences are split using the BERT-base tokenizer, which also prepends the special `CLS` token, and adds the `SEP` marker at the end. Each token is mapped to a learned embedding of dimension d_m, in which learned positional encodings are incorporated. Assuming a tokenized input length of L, this results in a sequence of $L+1$ vectors $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_L$.

Multi-modal Encoder A Transformer that receives both textual and video features, previously obtained by the respective encoders. For the former, we directly input the embedded text token sequence, while for the latter, we first project $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_K$ into the hidden dimension using a trainable linear layer and further combine this with a set of learned positional encodings, resulting in $\tilde{\boldsymbol{g}}_1, \ldots, \tilde{\boldsymbol{g}}_K$. These two encoded vector sequences are concatenated lengthwise and passed through 12 encoder blocks with 12 attention heads, to produce $\boldsymbol{h}_0, \ldots, \boldsymbol{h}_{K+L}$.

Decoder A Transformer decoder with 6 layers and 8 heads, with cross attention. The decoder receives the multi-modal contextualized features $\boldsymbol{h}_0, \ldots, \boldsymbol{h}_{K+L}$ and is required to generate captions in an auto-regressive manner.

5 Empirical Study

5.1 Timing Identification

Performance Evaluation To determine if a predicted time interval is a true positive, we follow previous work (Krishna et al., 2017) and inspect the temporal intersection over union (tIoU) with each ground truth segment, and check whether it is greater than or equal to a given threshold (0.3, 0.5, 0.7 and 0.9). We measure precision and recall for all thresholds, and report the averaged F1-scores which we compute based on those values. To study the impact of our model in the aforementioned proposal overlap issue, we compare all possible pairs from a set of proposals and measure their overlap in seconds and report the mean value obtained.

Implementation Details We use an off-line video encoding function based on C3D features (Tran et al., 2015) extracted every 8 frames with $d_v = 500$. As an activity recognition backend, we utilize DBG (Lin et al., 2020) pre-trained on ActivityNet, and as input use the top 100 proposals selected with the soft-NMS algorithm proposed by the authors. The GRU and LSTM networks have a hidden size of 512, we set $d_p = 100$, and the model is trained for a maximum of 25 epochs with a learning rate of 4×10^{-4} using Adam (Kingma and Ba, 2015).

Baselines We test a model variation that uses I3D features (Carreira and Zisserman, 2017; please see Section 5.2 for details about how these features are extracted), and also test our approach on the proposal phase of the DVC task on ActivityNet Captions (ANet Cap), comparing with the original performance as reported by Mun et al. (2019).

Results As can be seen in Table 2, our model is able to obtain a mean F1-score of 28.91, which degrades slightly when using I3D instead of C3D.

Though C3D outperforms I3D in terms of F1-score, we can observe that the proposal overlap was higher in that case, suggesting there is a trade-off between these metrics. We think this is partially explained by the difference in sampling rate across these features, which gives the model with I3D a finer time granularity.

We also see that the performance of the model on our task is substantially lower compared to ANet Cap, where it reaches a mean F1-score of 55.62. We surmise this performance reduction is due to the additional difficulty of our task, which requires the model to select a substantially larger set of segments compared to ANet Cap. However, we note that our approach is able to output a set of proposals with relative low overlap, which is not only substantially lower than in the case of ANet Cap, but also lower than 7.34 seconds, the average proposal overlap of our backend model (DGB).

We believe this suggests that our model is an effective way to adapt TAL to our setting. However, we point out that improvements in this direction will be ultimately limited by the nature of the annotations used to train the TAL model.

5.2 Utterance Generation

Implementation Details We again utilize an offline video encoding function, using the features released by Rodriguez-Opazo et al. (2021) who extracted features of size $d_v = 1024$ using I3D (Carreira and Zisserman, 2017) with average pooling, taking as input the raw frames of dimension 256×256, at 25fps. These features provided better results than C3D features on our early experiments. Our Transformer-based model uses $d_m = 512$ and is trained with a maximum learning rate of 10^{-4} with Adam and a linear annealing for 5% of the epochs, with a batch size of 8 using 4 NVIDIA V100 GPUs. During inference, we utilize beam search with a beam size of 5. For evaluation, we resort to BLEU-4 (Papineni et al., 2002) separately for segments for different annotators.

Dataset	Model	Feat.	Backend	Props.	F1	Overlap (s)
ANet Cap.	Original	C3D	SST	2.85	56.66	-
ANet Cap.	Our Impl.	C3D	DBG	2.73	55.62	10.5
Ours	Our Impl.	I3D	DBG	12.57	28.03	4.5
Ours	Our Impl.	C3D	DBG	12.67	**28.91**	5.6

Table 2: Results of our timing identification model. In the table, ANet Cap indicates results of the model on the Activity Net Captions dataset, Props. indicates the per-video average number of proposals selected, F1 is the average F1-score for tIoU thresholds 0.3, 0.5, 0.7 and 0.9, and Overlap denotes the average overlap of the selected proposals.

Encoder	Decoder	BLEU-4
Random	Random	2.11
Random	BART-base Dec.	1.66
BERT-base	Random	2.07
BERT-base	BART-base Dec.	**2.26**

Table 3: Impact of model initialization.

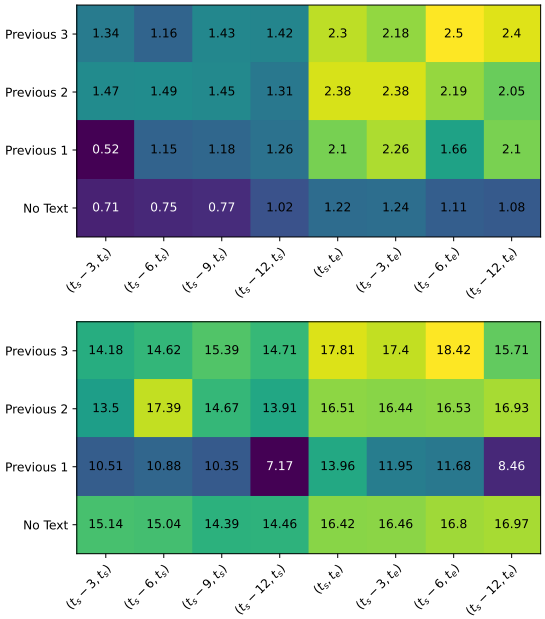

Figure 3: Results of our experiments controlling the amount of visual/textual fed to our model in our open-domain setting (top) and in-domain (bottom) setting by (Ishigaki et al., 2021). In the charts, the y-axis denotes the amount of textual content fed, while values on the x-axis denote the time interval of video features fed (order is not strict but correlates positively with total input length). For example, (t_s, t_e) means we use video features extracted from the gold standard annotated segments.

We begin by studying the impact of initializing our encoder and/or decoder with existing pre-trained models. In particular, we initialize our encoder with BERT (Devlin et al., 2019) and our decoder with the BART decoder (Lewis et al., 2020). As shown in Table 3, utilizing pre-trained encoders and decoders simultaneously to initialize model components leads to significant performance improvements. Based on these results, all the models in the rest of this paper are initialized following this approach.

Given the nature of our task, where each commentary is only partially aligned with the events

in the video, we are interested in studying the role of the visual and textual contexts in the ability of the model to generate the correct utterances. To that end, we propose several experiments where we control the amount of context fed to the model in both modalities. Additionally, to study how our proposed utterance generation model would work on an in-domain setting, we test it on the data by Ishigaki et al. (2021). In order to do so, we obtain their raw videos, resize them to a dimension 256×256, and extract I3D features as indicated above, using the pre-trained model trained on the Kinetics dataset, released by Carreira and Zisserman (2017). We initialize our encoder with Japanese-BERT (Suzuki and Takahashi, 2021) and use the corresponding tokenizer. We also study the role of visual/textual features on this data.

Figure 3 summarizes the results of our experiments, controlling the amount of visual/textual context fed to the model. We can clearly observe performance sweet spots in both datasets, where more context generally leading to better performance. We also note that access to more context on one modality can, to some extent, compensate for access to the other, which we surmise is due to the limited amount of input length the model can handle (512). We also see that the model generally struggles to attain good performance when only receiving features that precede the start timestamp annotation of a given utterance, which we think suggests the importance of the wait/generate trade-off in the real-time scenario.

Finally, we compare the performance of our proposed utterance generation model with that of Ishigaki et al. (2021). Their approach is also a multimodal model which as input receives the sequence of the previous 10 video frames, captured every second, 10 sets of structured telemetry data extracted from the gameplay, and the previous gold standard utterance. Input images are encoded using ViT (Dosovitskiy et al., 2020), input/output text is character-level tokenized, and the overall model is based on an seq2seq with LSTMs (Hochreiter and Schmidhuber, 1997).

As Table 4 shows, our model is able to attain substantially better performance when using a similar amount of only video context (9 v/s 10 seconds before t_s, for 10.35 vs 7.46 BLEU-4), which we believe validates our choice of model and input video representation. We also observe that performance increases to 13.96 BLEU-4 when feeding video features for the gold standard interval (t_s, t_e) but does not improve further with additional video context before t_s. It is only when also having access to textual context that the model is able to leverage the additional video context, ultimately leading to a maximum BLEU-4 score of 18.42. We think this is because the textual context helps the model decide what to say when the model needs to generate something that is not well aligned with the content of the input video segment, as well as to keep track of what has already been said previously. Finally, comparing this value to 24.01, the top performance reported by Ishigaki et al. (2021), we see the importance of access to in-domain information in this scenario, which contributes the most increasing the performance in their case.

Data	Model	Modal.	BLEU-4
Open Domain	Ours	V	1.24
	Ours	V + T	2.38
In Domain	Ours	V	13.96
	Ours	V + T	18.42
	Ishigaki et al.	V	7.46
		S	23.39
		S + T	23.86
		S + T + V	24.01

Table 4: Performance comparison across domains, where S, T and V stand for structured, textual and visual context, respectively.

6 Conclusions and Future Work

In this paper, we have proposed the task of generating live commentaries in an open-domain fashion. This is a substantial extension to previous work, which so far as only focused on single domains, where it is possible to use specific information.

To benchmark progress on our newly-introduced task, we also present a dataset of transcribed live commentaries aligned with more than 6K videos containing various human actions in a variety of domains. We present models to tackle our task alongside an in-depth experimental study, showing the strengths and limitations of current approaches.

Based on our results, for the future we are interested in developing in-domain multi-modal Transformer models can also incorporate in-domain specific information such as the structured data by (Ishigaki et al., 2021). For the open-domain task, we are interested in exploring end-to-end models recently proposed for DVC (Wang et al., 2021), as well as in tackling the on-line version of our task.

Limitations

In this work, we have introduced the task of generating live commentaries in an open-domain, including a dataset to use as a benchmark for progress. Our data contains videos from YouTube with annotations in English, and we focus mainly on this language. However, in this paper we also work with in-domain data for a related task which is annotated in Japanese. Our results show that our model is able to perform well on such dataset, with superior performance to the original results by Ishigaki et al. (2021) when controlling for input data. While we believe this shows the overall effectiveness of our approach in the task of commentary generation in general, we have no evidence to suggest how well these capabilities could generalize to other languages in our open-domain setting, where the performance is already low for English. This may prove specially important in low-resource languages, where access to pre-trained models is limited.

In terms of hardware requirements, all our experiments were performed on a large cluster, where we usually rely on a node with 4 NVIDIA V100 GPUs. We spent a total of approximately 1,000 USD in our experimental setup, most of which is due to our study of the captioning model under different settings. Despite training/inference being distributed across our 4 GPUs, our model can still run on single GPUs.

Ethical considerations

In terms of our audio collection efforts, although unlike conversation data, live commentary in our work is supposed to objectively describe the events in the video and contain no information about the speakers, we took extra care to protect their privacy; each commentary has been anonymized, and each annotator working on transcription has agreed to use the audio data solely for the given task.

Acknowledgements

This paper is based on results obtained from a project JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). For experiments, computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) was used.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: Semantic Propositional Image Caption Evaluation. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 382–398, Cham. Springer International Publishing.

Christopher Baldassano, Janice Chen, Asieh Zadbood, Jonathan W. Pillow, Uri Hasson, and Kenneth A. Norman. 2017. Discovering Event Structure in Continuous Narrative Perception and Memory. *Neuron*, 95(3):709–721.e5.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. 2015. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970.

Joao Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: UNiversal Image-TExt Representation Learning. In *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 104–120, Cham. Springer International Publishing.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv:1606.02012 [cs]*.

Chaorui Deng, Shizhe Chen, Da Chen, Yuan He, and Qi Wu. 2021. Sketch, Ground, and Refine: Top-Down Dense Video Captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 234–243.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. VLN BERT: A Recurrent Vision-and-Language BERT for Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Tatsuya Ishigaki, Goran Topic, Yumi Hamazono, Hiroshi Noji, Ichiro Kobayashi, Yusuke Miyao, and Hiroya Takamura. 2021. Generating Racing Game Commentary from Vision, Language, and Structured Data. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 103–113, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Byeong Jo Kim and Yong Suk Choi. 2020. Automatic baseball commentary generation using deep learning. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 1056–1065. Association for Computing Machinery, New York, NY, USA.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-Captioning Events in Videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 706–715.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chuming Lin, Jian Li, Yabiao Wang, Ying Tai, Donghao Luo, Zhipeng Cui, Chengjie Wang, Jilin Li, Feiyue Huang, and Rongrong Ji. 2020. Fast Learning of Temporal Action Proposal via Dense Boundary Generator. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):11499–11506.

Jonghwan Mun, Linjie Yang, Zhou Ren, Ning Xu, and Bohyung Han. 2019. Streamlined Dense Video Captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6588–6597.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ratish Puduppully and Mirella Lapata. 2021. Data-to-text Generation with Macro Planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

Chandan K.A. Reddy, Ebrahim Beyrami, Jamie Pool, Ross Cutler, Sriram Srinivasan, and Johannes Gehrke. 2019. A Scalable Noisy Speech Dataset and Online Subjective Test Framework. In *Interspeech 2019*, pages 1816–1820. ISCA.

Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Basura Fernando, Hongdong Li, and Stephen Gould. 2021. Dori: Discovering object relationships for moment localization of a natural language query in a video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1079–1088.

Michael Schaffrath. 2003. Mehr als 1:0! Bedeutung des Live-Kommentars bei Fußballübertragungen– eine explorative Fallstudie [more than 1:0! the importance of live commentary on football matches – an exploratory case study]. *Medien und Kommunikationswissenschaft*, 51.

Masatoshi Suzuki and Ryo Takahashi. 2021. Japanese bert. `https://github.com/cl-tohoku/bert-japanese`.

Yasufumi Taniguchi, Yukun Feng, Hiroya Takamura, and Manabu Okumura. 2019. Generating Live Soccer-Match Commentary from Play Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7096–7103.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning Spatiotemporal Features with 3D Convolutional Networks. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 4489–4497, USA. IEEE Computer Society.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. 2015. Translating Videos to Natural Language Using Deep Recurrent Neural Networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1494–1504, Denver, Colorado. Association for Computational Linguistics.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. 2021. End-to-End Dense Video Captioning With Parallel Decoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6847–6857.

Teng Wang, Huicheng Zheng, and Mingjing Yu. 2020. Dense-Captioning Events in Videos: SYSU Submission to ActivityNet Challenge 2020. *arXiv:2006.11693 [cs]*.

Binbin Xu, Chongyang Tao, Zidu Feng, Youssef Raqui, and Sylvie Ranwez. 2021. A Benchmarking on Cloud based Speech-To-Text Services for French Speech and Background Noise Effect.

Luowei Zhou, Chenliang Xu, and Jason J. Corso. 2018a. Towards Automatic Learning of Procedures From Web Instructional Videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Luowei Zhou, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong. 2018b. End-to-End Dense Video Captioning With Masked Transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8739–8748.

A Appendix

A.1 Experiment Details

- Training Duration:
  - Time Identification: Models generally take 1 to 2 hours to train, on a single GPU.
  - Utterance Generation: Models take approximately 4 hours to train for 20 epochs on the data by Ishigaki et al. (2021), reaching their best performance after around 13 epochs. On our dataset the behavior is similar, but the model takes approximately 8 hours in total.
- Total Parameters:
  - Time Identification: 10,487,297 parameters.
  - Utterance Generation: 225,750,540 parameters.
- Hyper-parameters: Besides what explained in the paper, we did not specifically run hyper-parameter exploration. However, on our early experiments both models we tried several learning rate variations around the reported 10^{-4} and manually picked the best for our main experimental setup.
- Dataset splits: for ActivityNet Captions, we use the official train/valid splits. For our data, we use the videos corresponding to the same train/validation splits from ActivityNet Captions. In both cases, we report results on the validation split, as the test-portion of Activity Net is held-out and used for annual challenges. For Ishigaki et al. (2021) data, use the provided train/valid/test splits, and report performance on the test set.
- Generation Evaluation: to evaluate our generative models we use the sacrebleu (Post, 2018) implementation of BLEU-4, tokenizing English text with the "13a" tokenizer and Japanese text with the "ja-mecab" tokenizer.

A.2 Inter-annotator agreement details

To more comprehensibly verify that the workers have provided semantically meaningful commentary, we also study inter-annotator agreement in terms of commentary content. In order to do so, we use BLEU-4 and SPICE (Anderson et al., 2016) to compare the set of utterances for a pair of annotators in a given video, sorted by time. As a result of such comparison, we obtain a matrix-like arrangement of values which can be used as a proxy the degree of content agreement over time for a pair of annotators. We perform this study on the validation split our data and of ActivityNet Captions, which is also contains two different sets of annotations.

To obtain an aggregated value for each dataset, we compute the mean value of each matrix-like result, which can be seen as a lower bound for the overall content agreement. More precise agreement values could be obtained by first aligning or matching utterances, or by penalizing scores by time distance, but in practice we found that our lower bound score plus the visual inspections detailed below were sufficient. We average across all the validation videos in both datasets and report. For ActivityNet Captions, we obtain mean values of 4.86 and 0.12 for BLEU-4 and SPICE respectively, while for our dataset we obtained 2.59 and 0.034 respectively. While the values for our data are lower than ANet Cap, we think they are still reasonable and that they reflect the nature of the task where we know the alignment between video content and utterance is lower.

Finally, Figure 4 and Figure show examples of how our matrices can be used to visually inspect the agreement, in this case on two randomly-sampled

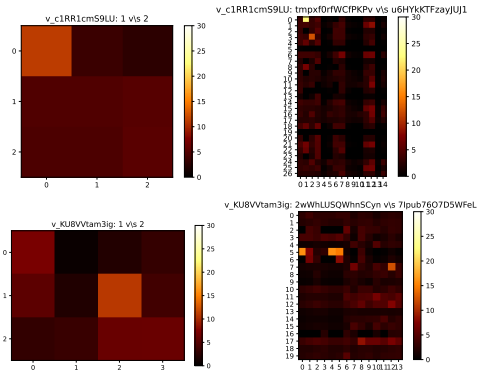

Figure 4: Inter-annotator agreement in terms of content based on BLEU-4, for videos "c1RR1cmS9LU" and "KU8VVtam3ig", for ActivityNet Captions (left) and our dataset (right).

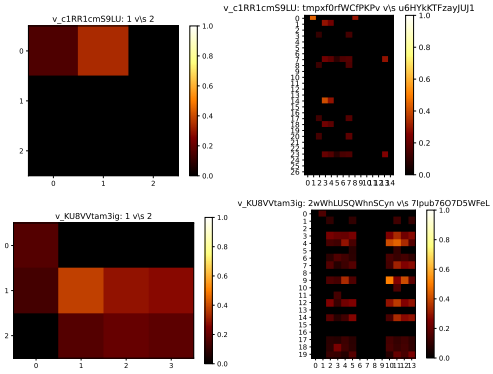

Figure 5: Inter-annotator agreement in terms of content based on SPICE, for videos "c1RR1cmS9LU" and "KU8VVtam3ig", for ActivityNet Captions (left) and our dataset (right).

videos. It is possible to observe that both datasets present a similar amount of agreement in terms of content, despite our examples having substantially longer utterance sequences, which we believe helps validate the quality of our annotations.