Memory-efficient Temporal Moment Localization in Long Videos

Cristian Rodriguez-Opazo[1*], Edison Marrese-Taylor[2,3*]
Basura Fernando[4], Hiroya Takamura[2], Qi Wu[1]
[1] Australian Institute for Machine Learning
[2] National Institute of Advanced Industrial Science and Technology
[3] The University of Tokyo, [4] A* STAR, Singapore
{edison.marrese, takamura.hiroya}@aist.go.jp
{cristian.rodriguezopazo, qi.wu01}@adelaide.edu.au
Fernando_Basura@ihpc.a-star.edu.sg

Abstract

Temporal Moment Localization is a challenging multimodal task which aims to identify the start and end timestamps of a moment of interest in an input untrimmed video, given a query in natural language. Solving this task correctly requires understanding the temporal relationships in the entire input video, but processing such long inputs and reasoning about them is memory and computationally expensive. In light of this issue, we propose Stochastic Bucket-wise Feature Sampling (SBFS), a stochastic sampling module that allows methods to process long videos at a constant memory footprint. We further combine SBFS with a new consistency loss to propose LOCFORMER, a Transformer-based model that can process videos as long as 18 minutes. We test our proposals on relevant benchmark datasets, showing that not only can LOCFORMER achieve excellent results, but also that our sampling is more effective than competing counterparts. Concretely, SBFS consistently improves the performance of prior work, by up to 3.13% in the mean temporal IoU, leading to a new state-of-the-art performance on Charades-STA and YouCookII, while also obtaining up to 12.8x speed-up at testing time and reducing memory requirements by up to 5x.

1 Introduction

Processing long untrimmed videos for understanding and reasoning is a computationally expensive task that not only demands a smart approach that can capture global and local interaction, but also requires allocating thousands of frames in memory (Wu et al., 2022). Previous work so far has mostly focused on parsing the visual content in independent snapshots of a video, which limits the approaches in terms of modeling long-term dependency between events in the input.

[*]These authors contributed equally to this work

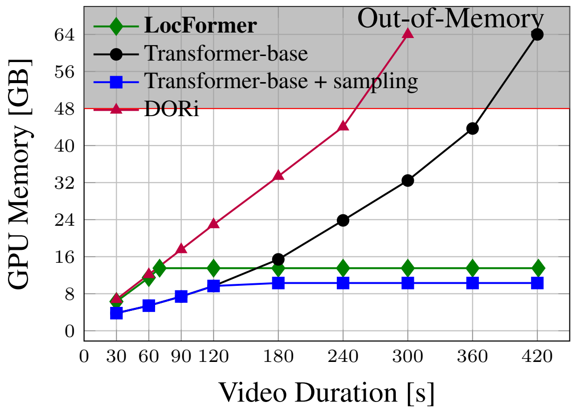

Figure 1: Empirical upper-bound memory consumption of LOCFORMER, Transformer-base, with and without enabling our proposed sampling technique (SBFS), and DORi (Rodriguez-Opazo et al., 2021). Memory usage (y-axis) is computed on an NVIDIA RTX-8000 GPU, using a batch size of 32 and assuming all sequences have maximum length for the given video duration.

In this paper, we particularly look at the task of temporal moment localization, which aims to identify the start and end timestamps of *a moment of interest* in an input untrimmed video given the query in natural language (Richard et al., 2018; Lin et al., 2017; Escorcia et al., 2016; Chao et al., 2018; Gao et al., 2017b; Xu et al., 2019). Recent approaches have aimed at directly predicting the starting and ending temporal locations, or regressing them from the input video, moving away from the propose-and-rank based approaches (Yuan et al., 2019c; Ghosh et al., 2019; Rodriguez-Opazo et al., 2020). Although these models are more efficient and can potentially capture the influence between different events in a video, they still require a considerable amount of computation and/or memory since they need to process the whole input video at once.

Recent improvements in both Natural Language Processing (NLP) and Computer Vision (CV) tasks can be attributed to the Transformer (Vaswani et al., 2017) model. Despite their success, one main drawback of these models is their computational cost, with memory usage ballooning as model sizes increase to attain better performance. This issue is exemplified by many existing developments in Transformer models for video understanding only being capable of processing short inputs at a time due to memory constraints. For example, TimeSformer-large (Bertasius et al., 2021) can only process inputs that are 24 seconds long[1], ViVIT (Arnab et al., 2021) and X-ViT (Bulat et al., 2021) can receive inputs that are up to 32 frames long, while other approaches like MERLOT (Zellers et al., 2021) and ClipBERT (Lei et al., 2021) specifically sample a single or a few frames from the whole video.

In light of this issue, we present Stochastic Bucket-wise Feature Sampling (SBFS). During training, SBFS first splits the input feature sequence into a fixed number of sections (buckets) and then selects a single feature per bucket using a stochastic approach. At testing time, our module selects the sufficient statistic of each bucket, which allows us to obtain a feature sample-set that is representative of the video content for the task at hand while keeping the memory footprint constant

To show the effectiveness of our sampling technique, we combine SBFS with a broad selection of models from previous work, and also use it to propose LOCFORMER, a Transformer-based model that operates at a constant memory footprint regardless of the input length, as shown in Figure 1. We conduct experiments on three challenging datasets, Charades-STA (Gao et al., 2017a), ActivityNet Captions (Caba Heilbron et al., 2015; Krishna et al., 2017) and YouCookII (Zhou et al., 2018b,a). We show that by applying SBFS in previous work we improve their performance in all considered datasets, leading to a new state-of-the-art on Charades-STA and YouCookII, while reducing the memory requirements by up to 5x. Furthermore, LOCFORMER is able to obtain state-of-the-art performance in the latter, and competitive results elsewhere.

We believe our results highlight the importance of sampling techniques as a valid mechanism to obtain better coverage of long input videos while keeping memory usage under budget. This ultimately provides a concrete direction for further research on tasks where it is necessary to cover long untrimmed videos, which include but are not limited to video grounding. We release our code and data[2] to encourage future research in this area.

[1]TimeSformer-base/large reads 8 frames at 32/4 fps respectively.

2 Related Work

Sampling To the best of our knowledge, the earliest example of a sampling technique that is similar to ours is the work of Nakagawa and Nakanishi (1988), who proposed a stochastic version of dynamic time warping for speech recognition of the Japanese language in the late 80s. This idea was further extended by Suryanto et al. (2016) in the context of motion recognition, where a randomized version of dynamic time warping for this task was introduced. We also find several models that utilize sampling techniques for action recognition in videos, in this case to specifically select salient clips, such as SCSampler (Korbar et al., 2019) and MGSampler (Zhi et al., 2021), which were later adopted by models like MVFNet (Wu et al., 2021). Also, Adaframe (Wu et al., 2019b) recently proposed a framework that adaptively selects relevant frames on a per-input basis for fast video recognition. Fayyaz et al. (2021) introduced differentiable parameter-free Adaptive Token Sampling (ATS), which uses a scoring to adaptively sample significant tokens and can be plugged into any existing vision transformer architecture. Finally, Wu et al. (2019a), recently proposed to process short video segments at a time also augmenting the models with a long-term bank from where global features can be sampled. This ultimately enables them to process longer overall inputs without losing global information. While their motivation is similar to ours, we note that their proposed sampling technique is not stochastic and requires learning. Moreover, our results show that performance improvements are possible without access to global context, making our contributions fundamentally different in this sense.

Temporal Action and Moment Localization The goal of *Temporal Action Localization* is to solve the problem of recognizing and determining temporal boundaries of action instances in videos, with extensive previous work devoted to it (Shou et al., 2016; Gu et al., 2018; Girdhar et al., 2019). Given its limitations —restricted to a pre-defined list of labels— the task of language-driven *temporal moment localization* was introduced as a generalization (Gao et al., 2017a; Hendricks et al., 2017). In this task, the goal is to determine the start and

[2]https://github.com/crodriguezo/locformer

end times of the video segment that best corresponds to a given natural language query. As this requires the model to extract useful information from the textual semantics in the query in order to identify the moment, this task is also usually regarded as video grounding. Early approaches that tackled temporal moment localization, including (Liu et al., 2018) and Ge et al. (2019), were mainly based on the generation of candidate clips which could later be ranked. Soon after, Chen et al. (2018), Chen and Jiang (2019), and Xu et al. (2019) focused mainly on reducing the number of proposals by producing query-guided or query-dependent approaches. Recently, Zhang et al. (2021a) also adopted a Transformer-based model for this setting, being the most relevant to our work.

The extensive computation of enumerating candidates in the above-mentioned proposal-based methods led to the development of methods that can directly output the temporal coordinates of the segment, namely, proposal-free approaches. In this context, Ghosh et al. (2019) first focused directly on predicting the start and end frames using regressions, and soon after Rodriguez-Opazo et al. (2020) improved results by modelling label uncertainty. While Mun Mun et al. (2020) and Zeng et al. (2020) later proposed more sophisticated modality matching strategies, some more recent approaches have focused on better contextualizing I3D (Carreira and Zisserman, 2017) video features by proposing other model variations (Li et al., 2021). More recently, Liu et al. (2021), CPNet (Li et al., 2021), VSLNet (Zhang et al., 2020a) have pushed performance further up. Finally, models like DORi (Rodriguez-Opazo et al., 2021), which also incorporates spatial features and CPN (Zhao et al., 2021) have proposed ad-hoc graph-based approaches. Compared to these models, our contributions go in a different direction which can be complementary because of their large memory consumption, as shown in Section 5.4.

3 Stochastic Bucket-wise Feature Sampling

Let G be a sequence of input video features extracted by a video encoding function $F_V(V)$. We are interested in developing a module to limit the memory budget of a given model when dealing with long video inputs. In order to do this, we limit the overall memory budget of the model by shortening the sequence of video features fed into the localization module. We do this in practice by proposing a technique that we call Stochastic Bucket-wise Feature Sampling (SBFS), which returns a sequence of length at most b derived from G, as follows.

$$\text{SBFS}(G; b) = \begin{cases} \{\boldsymbol{g}_i\}_{i=1}^{n} & \text{if } n \leq b \\ \{\boldsymbol{g}_{f(k)}\}_{k=1}^{m(n,b)} & \text{if } n > b \end{cases} \tag{1}$$

In Equation 1, $m(n, b)$ characterizes the number of buckets to be allocated to host video features, and is defined as $m(n, b) = \lfloor \frac{n}{\lceil n/b \rceil} \rfloor \leq b$, where $\lfloor\rfloor$, $\lceil\rceil$ are the floor and ceiling operators, respectively. The index $f(k)$ is sampled according to a uniform distribution over the indices of the features in the bucket, as Equation 2 shows, below.

$$f(k) \sim \mathcal{U}_{\lceil n/b \rceil \cdot (k-1)}^{\lceil n/b \rceil \cdot k} \tag{2}$$

More intuitively, we create a fixed number of buckets and allocate features to each by equally distributing them into the buckets. During training, we randomly sample a single feature for each bucket, following a uniform distribution, effectively reducing the input sequence length to at most b, the number of buckets. When doing this, we also accordingly convert the original set of labels $\tau^s, \tau^e \in [1, \dots, n]$ into $\bar{\tau}^s, \bar{\tau}^e \in [1, \dots, b]$. For simplicity, without loss of generality, for the rest of the paper we will assume the sampling module always returns total of b video features.

It is possible to prove that any sampled video feature sequence $\tilde{G}$ obtained using SBFS contains sufficient statistics of G, which allows us to train our models on very long videos with adequate guarantees. From this, it also follows that although SBFS can also be applied during inference, it is better to decouple the model from this stochastic component and instead utilize the max pooling operator over the features of the bucket at inference time. This gives models increased stability when predicting, without sacrificing performance, as we will show in Section 5.3. We refer the reader to Section A of our supplementary material for the full details on our theoretical analysis of SBFS.

4 LocFormer

In what follows, we assume that a given video $V \in \mathcal{V}$ can be characterized as a sequence of frames such that $V = \{v_t\}$ with $t = 1, \dots, l$. Each video in $\mathcal{V}$ is annotated with a natural language passage $S \in \mathcal{S}$ where S is a sequence of

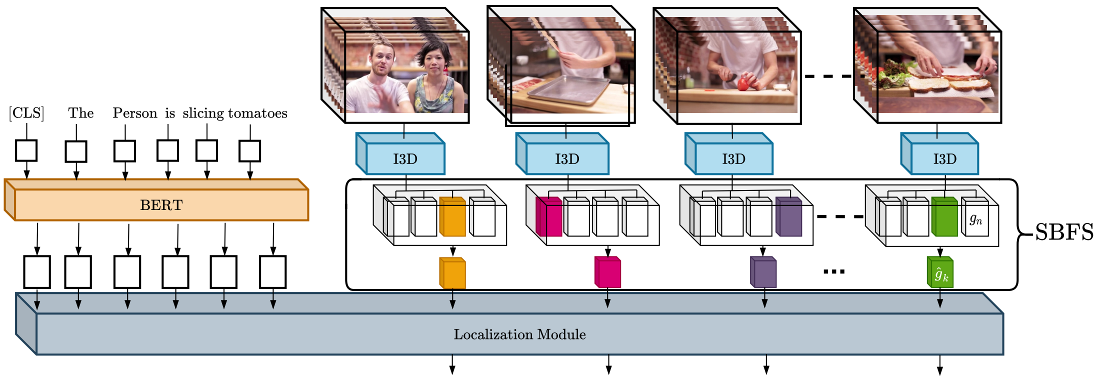

Figure 2: Our method uses a sampling technique that divides the video into a fixed number of buckets. Then, we uniformly sample a single I3D feature from each bucket, which is then fed into the localization module, a multi-modal Transformer model that receives video features and language features obtained from BERT.

tokens $S = \{s_j\}$ with $j = 1, \ldots, m$, which describes what is happening in a certain period of time. Formally, this interval is defined by t^s and t^e, the starting and ending points of the annotations in time, respectively. Although in the data a given video may be annotated with more than one single moment, and one natural language description may be associated to multiple moments, in this work we assume each derived case as an independent, separate training example.

Our model is trained to predict the most likely temporal localization of the contents of a given input query S in terms of its start and end positions $t^{s\star}$ and $t^{e\star}$ in the video. We apply the mapping $\tau = (t \cdot n \cdot \text{fps})/l$ to transform frame/feature index to time, converting t^s and t^e into τ^s and τ^e, which correspond to specific integer feature positions such that $\tau^s, \tau^e \in [1, \ldots, n]$.

LOCFORMER follows the Transformer architecture (Vaswani et al., 2017), which has been recently extended to multi-modal scenarios as in UNITER (Chen et al., 2020) in the context of vision-and-language, and Recurrent VLN-BERT (Hong et al., 2021) for vision-and-language navigation. Our model operates on sequences of tokens $\{s_j\}$ and a video $\{v_t\}$ characterized as a sequence of frames, as specified earlier. The overall architecture is composed by three main modules, as shown in Figure 2. (1) The Video Encoding Module, which is in charge of mapping video frames to vectors, and obtaining a sample that is representative of the video contents using SBFS, (2) the Text Encoding Module, a Transformer model with dimension d_m in charge of extracting useful representations from the natural language query, and (3) the Localization Module, a multi-modal Transformer, also with hidden dimension d_m, which receives both textual and video features from the previous modules and is in charge of estimating τ^s and τ^e. In the following subsections, we give details about each component and how they interact.

Video Encoding Module with SBFS Our video encoding module is in charge of mapping the l input video frames into a sequence of video features $G = \{\boldsymbol{g}_i \in \mathbb{R}^{d_v}\}, i = 1, \ldots, n$. At the core of this module lies SBFS, which we use to select a subset of the features that are representative of the contents of the video. Thus, the sampled video feature sequence $\tilde{G}$ will later be fed into the localization module.

Text encoding module Sentences are processed using the BERT tokenizer, which also prepends the special `CLS` token, and adds the `SEP` marker at the end. Each token is mapped to learned embeddings of dimension d_m and summed with learned positional encodings of the same size. These vectors are passed through $\bar{L}$ encoder transformer blocks with $\bar{M}$ attention heads, to produce final text representations $[\bar{\boldsymbol{h}}_0, \ldots, \bar{\boldsymbol{h}}_m]$.

Localization module A Transformer model that receives both textual and video features, previously obtained by the respective modules. For the former, we directly input $\bar{\boldsymbol{h}}_0, \ldots, \bar{\boldsymbol{h}}_m$, while for the latter we first project $\tilde{G} = [\tilde{\boldsymbol{g}}_1, \ldots, \tilde{\boldsymbol{g}}_b]$ into the hidden dimension using a trainable linear layer and further combine this with a set of learned positional encodings. These two encoded vector sequences are concatenated lengthwise and passed through L encoder blocks with M attention heads, to produce $[\boldsymbol{h}_0, \ldots, \boldsymbol{h}_{m+b}]$.

From these vectors, we select $[\boldsymbol{h}_{m+1}, \ldots, \boldsymbol{h}_{m+b}]$ and utilize the same localization function proposed by Rodriguez-Opazo et al. (2020) as the main training signal, namely, feature-level soft classification task on the time dimension. Concretely, two different MLP layers produce scores of each position being the start/end of the location, which are passed through a softmax activation to obtain $\hat{\boldsymbol{\tau}}^s, \hat{\boldsymbol{\tau}}^e \in \mathbb{R}^b$, which are compared to soft-labels using the Kullback-Leibler divergence ($\mathcal{L}_{KL}$).

In order to guide the model to utilize the information in the relevant section of the video, we encourage the attention heads of this module to put more weight into the target video portions during training, adapting the approach proposed by Rodriguez-Opazo et al. (2020) as shown below.

$$\mathcal{L}_{att} = -\sum_{l=1}^{L}\sum_{m=1}^{M}(\mathbf{1} - \boldsymbol{x} \otimes \boldsymbol{x}) * \log(\mathbf{1} - \boldsymbol{A}^{l,m}) \quad (3)$$

In Equation 3, $\boldsymbol{A}^{(l,m)}$ is the attention matrix of the l-th layer and m-th attention head of the localization module and $\boldsymbol{x} \in \mathbb{R}^{m+b}$ is a vector that denotes which areas of the output sequence will be subject to our guiding signal, and is defined as $\boldsymbol{x} = [\mathbf{1}_m; \boldsymbol{\delta}_{\tau^s \leq i \leq \tau^e}]$, where ; denotes concatenation, δ is the Kronecker delta returning 1 when i is inside the range of τ, and $\mathbf{1}_k$ denotes a vector of ones of size k.

Since our proposed localization loss is not sensitive to the order of the predictions of the starting and ending locations because there is no conditioning on the time in the model portions that generate them, in this paper we additionally propose to induce the model to respect the start-end order, taking a probabilistic approach. Concretely, we push the expected location of the start of the segment (S) to be before the expected location of the ending (E) location, which is equivalent to requiring $\mathbb{E}(E) - \mathbb{E}(S) > 0$. Replacing the values of the expectations, we obtain the following.

$$\mathbb{E}(E) - \mathbb{E}(S) = \sum_{i=1}^{b}\hat{\tau}_i^e i - \sum_{i=1}^{b}\hat{\tau}_i^s i = \sum_{i=1}^{b} i(\hat{\tau}_i^e - \hat{\tau}_i^s) \quad (4)$$

In Equation 4 above, $\hat{\tau}_i^s$ and $\hat{\tau}_i^e$ are integers that denote the predicted probability value of the starting and ending localizations at position i. Based on this derivation, we formally implement our loss by minimizing the negative difference of the expected values as shown in Equation 5, below.

$$\mathcal{L}_{se} = \min(0, \sum_{i=1}^{b} i(\hat{\tau}_i^s - \hat{\tau}_i^e)) \quad (5)$$

Finally, our model is trained with the direct summation of the three losses introduced earlier, such that $\mathcal{L} = \mathcal{L}_{KL} + \mathcal{L}_{att} + \mathcal{L}_{se}$.

5 Experiments

5.1 Datasets

To evaluate our proposed approach, we work with three widely-utilized and challenging datasets.

Charades-STA Built upon the Charades dataset (Sigurdsson et al., 2016), which provides time-based annotations using a pre-defined set of activity classes, and general video descriptions. We use the predefined train and test sets, containing 12,408 and 3,720 moment-query pairs respectively. Videos are 31 seconds long on average and a maximum duration of 194 seconds, with 2.4 moments on average, each being 8.2 seconds long on average.

ActivityNet Captions Introduced by Krishna et al. (2017), this dataset originally constructed for dense video captioning, consists of 20k YouTube videos with an average length of 120 seconds and a maximum duration of 755 seconds. The videos contain 3.65 temporally localized time intervals and sentence descriptions on average, where the average length of the descriptions is 13.48 words. Following the previous methods, we report the performance on the combined validation sets.

YouCookII Consists of 2,000 long untrimmed videos from 89 cooking recipes obtained from YouTube by Zhou et al. (2018b). Each step for cooking these dishes was annotated with temporal boundaries and aligned with the corresponding section of the recipe. The average video length is 316 seconds and a maximum duration of 755 seconds. In terms of relevant moment segments, each video has 7.73 moments on average, with each segment being 19.63 seconds long on average.

5.2 Implementation Details

For our experiments, we consider an off-line video encoding function $F_V(V)$, following previous work (Ghosh et al., 2019; Rodriguez-Opazo et al., 2020; Rodriguez-Opazo et al., 2021; Wang et al., 2020; Yuan et al., 2019a,b; Zhang et al., 2019). Concretely, we first pre-process the videos by extracting features of size 1024 using I3D with average pooling, taking as input the raw frames of dimension 256×256, at 25fps. We use the pre-trained model trained on Kinetics for ActivityNet and YouCookII released by Carreira and Zisserman (2017). For Charades-STA, we use the pre-trained model trained on Charades. For the natural language input, we use the BERT-base-uncased tokenizer and keep the parameters of the Text Encoder fixed. Models are trained using Adam (Kingma and Ba, 2014). Evaluation is based on two widely used metrics proposed by Gao et al. (2017a), namely the Recall at various thresholds of the temporal Intersection over Union (tIoU or $R@\alpha$) measuring the percentage of predictions that have tIoU with ground truth larger than certain α, and the mean averaged tIoU (mIoU). We use three α thresholds 0.3, 0.5 and 0.7.

5.3 Ablation Studies

We begin by performing an empirical study of our proposed stochastic sampling technique, comparing it to several alternatives. We specifically consider the following sampling approaches.

Random: As a naive baseline, we randomly sample features maintaining the order.

Fixed-rate video down-sampling (FRVS): We experiment with I3D features extracted at a lower frame-rate of 5fps (Ghosh et al., 2019), which can be regarded as a form of low-level down-sampling.

Fixed-rate feature down-sampling (FRFS): We experiment with two fixed-rate down-sampling techniques at the feature level, bucket-level mean-pooling and max-poling.

Dynamic Time Warping (DTW): We perform dynamic time warping between the non-structured video features and the fixed size temporal sequence created using our stochastic sampling technique and max-pooling applied inside each bucket. In this way, we assign features to each bucket that will later be randomly sampled.

Dynamic-rate feature down-sampling (DRFS): We utilize the similarity across features to dynamically create each bucket. While many variations are possible here, we decided to utilize a cosine distance-based heuristic to create the buckets. Please see Section B in the supplementary material for all the details.

SBFS Variations: We experiment with different alternatives for inference. Concretely, we always apply our stochastic sampling during training, and either use it for inference as well (SBFS-all), or replace it with bucket-wise mean pooling (SBFS-mean) or max pooling (SBFS).

Sampling	Performance			
	R@0.3	R@0.6	R@0.7	mIoU
Random	09.42	03.35	00.74	0 9.24
FRVS-5fps	37.54	22.68	10.62	23.90
FRFS-mean	45.99	31.01	15.46	29.98
FRFS-max	45.53	30.61	15.55	30.11
DTW	27.58	13.52	04.47	18.22
DRFS	33.48	17.84	06.56	21.57
SBFS-all	46.28	30.04	15.32	30.34
SBFS-mean	46.68	30.61	15.23	30.53
SBFS	**46.76**	**31.33**	**15.81**	**30.92**

Table 1: Performance of LOCFORMER on YoucookII replacing SBFS with alternative sampling techniques.

For the experiments, we combine each of these sampling approaches with the rest of the LOCFORMER architecture, and always use a bucket size of 200. Regarding the data, we use the YouCookII dataset, as it contains videos that can be as long as 18 minutes with queries that use rich language, which should help illustrate the importance of the sampling.

As the results in Table 1 show, the effectiveness of our sampling technique is clear, specially when compared with more naive alternatives like random sampling, or simple mean pooling. We see that low-level down-sampling techniques, that extract fewer frames from the original video, are not effective either. In contrast, the naive version of the max-pooling-based sampling stands out, performing similarly but still below SBFS. To confirm the effectiveness of our approach against FRFS-max and FRFS-mean, we further compared average results of five runs using different random seeds. We obtained baseline performances of $0.3006 \pm 1.6 \times 10^{-3}$ for FRFS-max and $0.3010 \pm 2.9 \times 10^{-3}$ for FRFS-mean. In contrast, SBFS obtained a performance of $0.3060 \pm 2.9 \times 10^{-3}$, which we find is statistically superior to both baselines at 99% confidence. These results help illustrate the importance of the stochastic approach we have taken, which enables us to limit the input to the model while still exposing it to all the training data in the long run, significantly improving its generalization capabilities. Finally, we also note that all the tested sampling alternatives except DTW and

Bucket Size	Performance			
	R@0.3	R@0.6	R@0.7	mIoU
100	35.74	27.41	11.88	29.03
200	**46.76**	**31.33**	15.81	**30.92**
300	46.36	31.07	**15.95**	30.63
400	45.25	30.30	15.58	30.23
500	44.33	28.69	14.18	29.14

Table 2: Impact on performance of LOCFORMER on the YoucookII dataset as parameter b, the bucket size, changes.

DRFS do not utilize information about the features when generating the buckets. It is interesting to see that many of these arguably simpler sampling techniques, including SBFS, outperform data-informed approaches.

We also study the impact of SBFS at different bucket sizes. For these experiments we use the YouCookII dataset, which contains the longest videos on average, and test bucket sizes ranging from 100 to 500. As we can see on Table 2, variations on parameter b have an impact consistent with its expected behavior, with diminishing results as b increases, and a clear performance sweet-spot at $b = 200$ which we adopt for the main experiments in our paper.

Finally, we ablate LOCFORMER component-by-component, to test the effects of each one of our introduced losses, and of SBFS. We also compare LOCFORMER with two highly-competitive Transformer-based model variations: (1) **Transformer-base** a randomly-initialized multi-modal Transformer-base[3], into which we directly feed the text input and the sampled video features, previously embedding them using a learned embedding matrix and a linear projection layer, respectively. Each encoder uses a separate set of positional embeddings, and we also add a type embedding to indicate the model the nature of each vector. After this, the embedded sequences are concatenated lengthwise and passed through the transformer blocks; (2) **BERT-base**, where we initialize our Transformer-base variation with the weights of BERT. In this case, the projection linear layer of the video features and the respective positional encodings are randomly initialized.

Both model variations apply our attention guiding loss $\mathcal{L}_{att}$ to all the attention heads in all the layers. This effectively means that there is no functionality separation inside these models. We note that these baselines are comparable to existing multi-modal transformer models such as VilBERT (Lu et al., 2019) and UNITER (Chen et al., 2020). Experiments are again performed in YouCookII, which contains the longest videos.

As seen in Table 3, our results first highlight the importance of SBFS, enabling all kinds of Transformer-based models we tested to process long untrimmed videos, which would otherwise lead to out-of-memory errors. Regarding the interaction of the attention loss with different model variations, we see that this additional training signal leads to consistent gains for all Transformer-based variations, but that these are larger in the case of LOCFORMER. We surmise this is due to the attention loss potentially interfering with the inductive bias that the baselines require to process the multi-modal inputs, as well as with the already acquired bias in the case of BERT, reflected in certain attention patterns for each head which have been studied and documented by Rogers et al. (2020) among others. This ultimately highlights the importance of separating functionality inside Transformer models for our task, which allows our model to perform better overall.

On top of SBFS, our results also show how each of our proposed component clearly helps increase performance. In particular, we observe that the addition of our attention loss ($\mathcal{L}_{att}$) results in the largest performance improvements, which is consistent with the findings of Rodriguez-Opazo et al. (2020). While the improvements due to $\mathcal{L}_{se}$ are comparatively less substantial, with differences across α bands, we see that combining both losses leads to the best results, showing that the losses complement each other well, as expected.

5.4 Combining SBFS with previous work

We now focus on studying the ability of our sampling technique to be combined with existing models. We do this by incorporating SBFS into three proposal-free models selected from the literature, and testing them on our datasets. We consider ExCL (Ghosh et al., 2019) and TMLGA (Rodriguez-Opazo et al., 2020), which have been extensively studied in the past years, as well as DORi. We utilized our own implementation of ExCL with our I3D features extracted at 25 fps[4],

[3]We follow the original notation (Vaswani et al., 2017) using 12 layers and 12 attention heads.

[4]The original implementation used a frame-rate of 5 fps.

Component			Transformer-base				BERT-base				Locformer			
SBFS	$\mathcal{L}_{att}$	$\mathcal{L}_{se}$	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU
			OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
✓			33.65	20.50	9.05	22.23	37.29	22.51	10.74	24.61	42.33	28.26	12.83	27.49
✓		✓	33.71	19.90	9.08	21.89	39.46	25.60	12.57	26.41	42.84	28.01	13.80	28.04
✓	✓		40.86	25.09	11.51	26.16	41.32	27.38	14.15	27.88	46.39	30.58	15.35	30.57
✓	✓	✓	**41.75**	**26.52**	**13.34**	**27.83**	**42.18**	**28.01**	**14.63**	**28.24**	**46.76**	**31.33**	**15.81**	**30.92**

Table 3: Results of our Transformer ablation study, performed on the YouCookII dataset, where OOM indicates we obtained an out-of-memory error even when using the largest GPU at our disposal.

Method	Charades-STA					ActivityNet					YouCookII				
	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU	Mem.[GB]	R@0.3	R@0.5	R@0.7	mIoU
ExCL	2.8	62.28	39.74	22.53	42.28	6.4	55.49	39.33	23.04	40.32	6.9	26.58	15.72	8.19	18.99
+ SBFS	**1.6**	**62.74**	**42.04**	**24.57**	**43.05**	**1.8**	**56.42**	**40.37**	**24.70**	**41.13**	**1.8**	**30.96**	**18.64**	**10.05**	**21.76**
TMLGA	2.3	67.53	52.02	33.74	48.22	7.3	51.28	33.04	19.26	37.78	9.5	33.48	20.65	10.94	23.07
+ SBFS	**1.5**	**70.67**	**52.20**	**33.90**	**49.18**	**1.7**	**53.00**	**35.10**	**19.83**	**37.85**	**1.8**	**39.25**	**25.40**	**12.80**	**26.20**
DORi ♣	32.8	72.72	59.65	40.56	53.28	34.7	57.89	41.49	**26.41**	42.78	46.4	43.36	30.47	18.24	30.46
+ SBFS	**23.1**	**72.90**	**59.67**	**40.94**	**53.44**	**24.0**	**58.89**	**42.21**	26.36	**43.02**	**24.2**	**46.74**	**32.19**	**18.33**	**31.69**

Table 4: Results of our experiments combining SBFS with existing work. Except where indicated, experiments were performed using a batch size 32. ♣ indicates experiments performed using a batch size of 4 due to memory constraints.

Method	Training		Inference	
	TMLGA	DORi	TMLGA	DORi
Baseline	3.18	72	0.36	32
+ SBFS	0.72	10	0.08	2.5
Speed-Up	**4.41**	**7.2**	**4.5**	**12.8**

Table 5: Time in minutes for processing 10,337 and 3,492 queries from the YouCookII dataset in training and inference (test split) respectively.

and directly integrated the original implementations of the latter into our code.

As Table 4 shows, SBFS is able to consistently provide performance improvements in all cases, with gains of up to 3.13% in terms of the mean temporal IoU. These improvements lead to new state-of-the-art results on both the Charades-STA and YouCookII datasets. We note that despite not having access to the spatial information that DORi incorporates, the performance of LOCFORMER is very competitive to that of DORi+SBFS in YouCookII, which contains the longest videos.

We further study the impact on training and inference time when adding our sampling module. For this study consider TMLGA and DORi, and again use YoucookII since it contains the longest videos. As shown in Table 5, SBFS consistently leads to shorter training and inference times, with the degree of impact depending on dataset and model. Concretely, on DORi we obtain a speed-up of 7.2x in training and 12.8x for inference, which again illustrates the effectiveness of our approach, especially in the context of larger models. We also notice that the methods converge in a similar number of epochs with or without adding SBFS, meaning that we can significantly reduce the convergence time, as our sampling enables models to go through each example faster.

5.5 Comparison to state-of-the-art models

Finally, we compare the performance of our proposals on the datasets considered against several prior works. We consider models based on different approaches, specifically proposal-based models including CBP (Wang et al., 2020), MS-TAN-2D (Zhang et al., 2021b) and MNM (Wang et al., 2022), as well as TripNet (Hahn et al., 2020), based on reinforcement learning.

In addition to that, we also compare our approach to more recent methods that do not rely on proposals, including ABLR (Yuan et al., 2019c), ExCL, TMLGA and LGVTI (Mun et al., 2020), CPNet (Li et al., 2021), as well as VSLNet (Zhang et al., 2020b) and BCPN (Nawaz et al., 2022), which cast our task as visual question answering, and CPL (Zheng et al., 2022) which also incorporate Transformer-based components. Finally, we also consider CPN (Zhao et al., 2021) and DORi,

Method	Charades-STA				ActivityNet				YouCookII			
	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU	R@0.3	R@0.5	R@0.7	mIoU
ABLR †	-	24.36	9.00	-	55.67	36.79	-	36.99	-	-	-	-
TripNet	51.33	36.61	14.50	-	48.42	32.19	13.93	-	-	-	-	-
CBP	50.19	36.80	18.87	35.74	54.30	35.76	17.80	36.85	-	-	-	-
ExCL ‡	65.10	44.10	22.60	-	62.10	41.60	23.90	-	26.58	15.72	8.19	18.99
TMLGA	67.53	52.02	33.74	48.22	51.28	33.04	19.26	37.78	33.48	20.65	10.94	23.07
LGVTI	72.96	59.46	35.48	51.38	58.52	41.51	23.07	41.13	-	-	-	-
DORi	72.72	59.65	40.56	53.28	57.89	41.49	26.41	42.78	43.36	30.47	18.24	30.46
VSLNet	70.46	54.19	35.22	50.02	63.16	43.22	26.16	43.19	-	-	-	-
CPNet	-	60.27	38.74	52.00	-	40.56	21.63	40.65	-	-	-	-
CPN	**75.53**	59.77	36.67	53.14	62.81	45.10	28.10	**45.70**	-	-	-	-
BCPN	73.42	**61.77**	**43.91**	-	66.87	44.53	30.11	-	-	-	-	-
MS-2D-TAN	-	60.08	37.39	-	62.09	45.50	28.28	-	-	-	-	-
CPL	66.40	49.24	22.39	-	**82.55**	**55.73**	**31.37**	-	-	-	-	-
MNM	-	47.31	27.28	-	65.05	48.59	29.26	-	-	-	-	-
DORi + SBFS	72.90	59.67	40.94	**53.44**	58.89	42.21	26.36	43.02	46.74	**32.19**	**18.33**	**31.69**
LOCFORMER	71.88	58.52	38.51	51.76	60.61	43.74	27.04	44.05	**46.76**	31.33	15.81	30.92

Table 6: Performance comparison of our approach with existing methods for different tIoU α levels. Underlined and bold results indicate second-best and best performance for each dataset. Values are reported on the validation split of Charades-STA and the ActivityNet Captions. † Results for ABLR are as reported by Chen and Jiang (2019). ‡ The results reported by ExCL for ActivityNet have 3,370 missing videos, and the results on YoucookII were obtained using our own implementation.

two models that contain specific graph-based approaches for the task, with DORi also incorporating spatio-temporal features.

Table 6 summarizes our best results on Charades-STA, ActivityNet Captions and YouCookII datasets, while also comparing the obtained performance to relevant prior work. We can see that overall LOCFORMER is able to offer excellent performance, closing the gap with sophisticated graph-based models like CPN and DORi as well as Transformer-based models like CPL, with new state-of-the-art performance on the Charades-STA dataset. The results also show the effectiveness of our sampling approach when dealing with long untrimmed videos, as we observe that by combining DORi with SBFS we are able to also attain state-of-the art results on the YoucookII dataset.

6 Conclusion

In this paper we have presented LOCFORMER, a Transformer-based model for the task of temporal moment localization which operates at a constant maximum memory footprint regardless of the input length. The success of our model fundamentally relies on our modular design, which allows us to separate functionality, and SBFS, where we split the sequence of input video features into a fixed number of buckets and select a single feature per bucket using a stochastic approach.

Experiments conducted on three challenging datasets show that LOCFORMER obtains competitive results, and show that our sampling technique can improve the performance of prior work on all considered datasets, leading to a new state-of-the-art on YoucookII and Charades-STA. We think these results highlight the importance of sampling techniques as a valid mechanism to obtain better coverage of long input videos while keeping memory usage low.

For future work, we are interested in testing our sampling approach in other relevant tasks in the context of video-and-language, for example video retrieval. We are also interested in extending our approach to address its limitations, for example, using adaptive or iterative sampling to treat different areas of the video with different granularity.

Acknowledgements

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

Limitations

We believe our results show that sampling is a valid mechanism to obtain better coverage of long input videos, while keeping memory usage under budget. However, it is important to stress that in this work we primarily focused on proposal-free models for temporal moment localization and thus have no evidence to suggest such improvements would be observed in other models.

There is also a specific scenario where our sampling could degrade the performance of a given model. This scenario occurs when the span of the query, *a.k.a.* moment, is located completely inside a given bucket, with many additional frames/features in the same bucket. In this case, the best that a model can do is to predict a moment covering the whole bucket, this losing granularity. In practice, this occurs when the ratio between the duration of a given moment and the duration of the video is vanishingly small. We believe this issue can be alleviated in future work by recursively generating and exploring buckets, an issue that we would like to tackle in the future.

Our results are also limited to the features utilized by the models we considered, and we offer no evidence that our technique will generalize to those cases as well. In terms of data, we have only experimented with queries in English. While it would be interesting to experiment with different languages, this is so far a limitation of the datasets that exist.

References

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. 2021. ViViT: A Video Vision Transformer. *arXiv:2103.15691 null.*

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. 2021. Is Space-Time Attention All You Need for Video Understanding? *arXiv:2102.05095 [cs].*

Adrian Bulat, Juan-Manuel Perez-Rua, Swathikiran Sudhakaran, Brais Martinez, and Georgios Tzimiropoulos. 2021. Space-time Mixing Attention for Video Transformer. *arXiv:2106.05968 [cs].*

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.*

Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A. Ross, Jia Deng, and Rahul Sukthankar. 2018. Rethinking the faster R-CNN architecture for temporal action localization. *CVPR.*

Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. 2018. Temporally grounding natural sentence in video. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 162–171, Brussels, Belgium. Association for Computational Linguistics.

Shaoxiang Chen and Yu-Gang Jiang. 2019. Semantic proposal for activity localizaiton in videos via sentence query. *AAAI.*

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 *ECCV 2020*, Lecture Notes in Computer Science, pages 104–120, Cham. Springer International Publishing.

Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. 2016. DAPs: Deep Action Proposals for Action Understanding. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 768–784. Springer International Publishing.

Mohsen Fayyaz, Soroush Abbasi Kouhpayegani, Farnoush Rezaei Jafari, Eric Sommerlade, Hamid Reza Vaezi Joze, Hamed Pirsiavash, and Juergen Gall. 2021. Ats: Adaptive token sampling for efficient vision transformers. *arXiv preprint arXiv:2111.15667.*

Ronald A Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222(594-604):309–368.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. 2017a. Tall: Temporal activity localization via language query. In *ICCV.*

Jiyang Gao, Zhenheng Yang, Chen Sun, Kan Chen, and Ram Nevatia. 2017b. TURN TAP: temporal unit regression network for temporal action proposals. *ICCV.*

Runzhou Ge, Jiyang Gao, Kan Chen, and Ram Nevatia. 2019. Mac: Mining activity concepts for language-based temporal localization. In *WACV.*

S. Ghosh, A. Agarwal, Zarana Parekh, and A. Hauptmann. 2019. Excl: Extractive clip localization using natural language descriptions. In *NAACL-HLT.*

Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. 2019. Video Action Transformer Network. In *CVPR*, pages 244–253.

Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul

Sukthankar, Cordelia Schmid, and Jitendra Malik. 2018. AVA: A Video Dataset of Spatio-Temporally Localized Atomic Visual Actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056.

Meera Hahn, Asim Kadav, James M Rehg, and Hans Peter Graf. 2020. Tripping through time: Efficient localization of activities in videos. *BMVC*.

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *ICCV*.

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.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*.

Bruno Korbar, Du Tran, and Lorenzo Torresani. 2019. Scsampler: Sampling salient clips from video for efficient action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *ICCV*.

Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L. Berg, Mohit Bansal, and Jingjing Liu. 2021. Less Is More: ClipBERT for Video-and-Language Learning via Sparse Sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7331–7341.

Kun Li, Dan Guo, and Meng Wang. 2021. Proposal-Free Video Grounding with Contextual Pyramid Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(3):1902–1910.

Tianwei Lin, Xu Zhao, and Zheng Shou. 2017. Single Shot Temporal Action Detection. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 988–996, New York, NY, USA. ACM. Event-place: Mountain View, California, USA.

Daizong Liu, Xiaoye Qu, Jianfeng Dong, and Pan Zhou. 2021. Adaptive Proposal Generation Network for Temporal Sentence Localization in Videos. *arXiv:2109.06398 [cs]*.

Meng Liu, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. 2018. Attentive moment retrieval in videos. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 15–24. ACM.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *Advances in Neural Information Processing Systems 32*, pages 13–23. Curran Associates, Inc.

Jonghwan Mun, Minsu Cho, and Bohyung Han. 2020. Local-Global Video-Text Interactions for Temporal Grounding. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10807–10816, Seattle, WA, USA. IEEE.

Seiichi Nakagawa and Hirobumi Nakanishi. 1988. Speaker-independent english consonant and japanese word recognition by a stochastic dynamic time warping method. *IETE Journal of Research*, 34(1):87–95.

Hafiza Sadia Nawaz, Zhensheng Shi, Yanhai Gan, Amanuel Hirpa, Junyu Dong, and Haiyong Zheng. 2022. Temporal Moment Localization via Natural Language by Utilizing Video Question Answers as a Special Variant and Bypassing NLP for Corpora. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1.

Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. 2018. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2.

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.

Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Fatemeh Sadat Saleh, Hongdong Li, and Stephen Gould. 2020. Proposal-free temporal moment localization of a natural-language query in video using guided attention. *WACV*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. 2016. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*.

Chendra Hadi Suryanto, Jing-Hao Xue, and Kazuhiro Fukui. 2016. Randomized time warping for motion recognition. *Image and Vision Computing*, 54:1–11.

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.

Jingwen Wang, Lin Ma, and Wenhao Jiang. 2020. Temporally grounding language queries in videos by contextual boundary-aware prediction. *AAAI*.

Zhenzhi Wang, Limin Wang, Tao Wu, Tianhao Li, and Gangshan Wu. 2022. Negative Sample Matters: A Renaissance of Metric Learning for Temporal Grounding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(3):2613–2623.

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. 2019a. Long-Term Feature Banks for Detailed Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 284–293.

Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. 2022. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. *arXiv preprint arXiv:2201.08383*.

Wenhao Wu, Dongliang He, Tianwei Lin, Fu Li, Chuang Gan, and Errui Ding. 2021. Mvfnet: Multi-view fusion network for efficient video recognition. In *AAAI*.

Zuxuan Wu, Caiming Xiong, Chih-Yao Ma, Richard Socher, and Larry S. Davis. 2019b. Adaframe: Adaptive frame selection for fast video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huijuan Xu, Kun He, L Sigal, S Sclaroff, and K Saenko. 2019. Multilevel language and vision integration for text-to-clip retrieval. In *AAAI*.

Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. 2019a. Semantic Conditioned Dynamic Modulation for Temporal Sentence Grounding in Videos. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 536–546. Curran Associates, Inc.

Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. 2019b. Semantic conditioned dynamic modulation for temporal sentence grounding in videos. In *Advances in Neural Information Processing Systems*, pages 534–544.

Yitian Yuan, Tao Mei, and Wenwu Zhu. 2019c. To find where you talk: Temporal sentence localization in video with attention based location regression. *AAAI*.

Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. 2021. MERLOT: Multimodal Neural Script Knowledge Models. *arXiv:2106.02636 [cs]*.

Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. 2020. Dense Regression Network for Video Grounding. *arXiv:2004.03545 [cs]*.

Da Zhang, Xiyang Dai, Xin Wang, Yuan-Fang Wang, and Larry S Davis. 2019. Man: Moment alignment network for natural language moment retrieval via iterative graph adjustment. *CVPR*.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. 2020a. Span-based Localizing Network for Natural Language Video Localization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6543–6554. Association for Computational Linguistics.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. 2020b. Span-based Localizing Network for Natural Language Video Localization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6543–6554, Online. Association for Computational Linguistics.

Mingxing Zhang, Yang Yang, Xinghan Chen, Yanli Ji, Xing Xu, Jingjing Li, and Heng Tao Shen. 2021a. Multi-Stage Aggregated Transformer Network for Temporal Language Localization in Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12669–12678.

Songyang Zhang, Houwen Peng, Jianlong Fu, Yijuan Lu, and Jiebo Luo. 2021b. Multi-Scale 2D Temporal Adjacency Networks for Moment Localization with Natural Language. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Yang Zhao, Zhou Zhao, Zhu Zhang, and Zhijie Lin. 2021. Cascaded Prediction Network via Segment Tree for Temporal Video Grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4197–4206.

Minghang Zheng, Yanjie Huang, Qingchao Chen, Yuxin Peng, and Yang Liu. 2022. Weakly Supervised Temporal Sentence Grounding With Gaussian-Based Contrastive Proposal Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15555–15564.

Yuan Zhi, Zhan Tong, Limin Wang, and Gangshan Wu. 2021. Mgsampler: An explainable sampling strategy for video action recognition. *CoRR*, abs/2104.09952.

Luowei Zhou, Nathan Louis, and Jason J Corso. 2018a. Weakly-supervised video object grounding from text by loss weighting and object interaction. In *British Machine Vision Conference*.

Luowei Zhou, Chenliang Xu, and Jason J Corso. 2018b. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence*, pages 7590–7598.

A Theoretical Analysis of SBFS

Let us begin by providing some key definitions that we will be requiring to perform our analysis.

A *statistic* is a function $T = r(X_1, \ldots, X_n)$ of the random sample $X_1, \ldots, X_n$, which carries information of the sampled data, such as the sample mean and sample variance. We say that a statistic satisfies the criterion of sufficiency when no other statistic which can be calculated from the same sample provides any additional information as to the value, of the parameter to be estimated. We can easily find a sufficient statistics by using the *Fisher–Neyman Factorization Theorem.*

Factorization theorem: *given a random sample $X_1, \ldots, X_n$ with joint density $f(x_1, \ldots, x_n|\theta)$ a statistic $T = r(X_1, \ldots, X_n)$ is sufficient if and only if the joint density can be factored as follows:*

$$f(x_1, \ldots, x_n|\theta) = u(x_1, \ldots, x_n)v(r(x_1, \ldots, x_n), \theta)$$

where u and v are non-negative functions. The function u can depend on the full random sample $x_1, \ldots, x_n$ but not on the unknown parameter θ. The function v can depend on θ, but can depend on the random sample only through the value of $r(x_1, \ldots, x_n)$.

In our case, let us assume that our bucket contains features X_i that are independent and uniformly distributed on $[0, \theta]$ where θ is unknown. Then, the probability dense function can be written as a product of individual densities since the observations are independent,

$$f(x_1, \ldots, x_n|\theta) = \frac{1}{\theta}\mathbf{1}_{\{0 \leq x_1 \leq \theta\}} \cdots \frac{1}{\theta}\mathbf{1}_{\{0 \leq x_n \leq \theta\}}$$

Here $\mathbf{1}(E)$ is an indicator function. It is 1 if the event E holds, and 0 if it does not. Now $x_i \leq \theta$ for $i = 1, \ldots, n$ if and only if $\max\{x_1, \ldots, x_n\} \leq \theta$. Therefore,

$$f(x_1, \ldots, x_n|\theta) = \frac{1}{\theta^n}\mathbf{1}_{\{0 \leq \min\{x_i\}\}}\mathbf{1}_{\{\max\{x_i\} \leq \theta\}}$$

Thus, the *factorization theorem* shows that $T = \max\{X_1, \ldots, X_n\}$ is a sufficient statistic since the density function takes the required form, where $u = \mathbf{1}_{\{0 \leq \min\{x_i\}\}}$ and $v = \frac{1}{\theta^n}\mathbf{1}_{\{\max\{x_i\} \leq \theta\}}$, which is a function that only depends on θ and $T = \max\{x_i\}$.

With this in mind, we can now move on to our analysis. We first note that nature of SBFS is that a single feature is sampled from each bucket with equal probability each time. If we assume features are i.i.d., this implies that the probability of getting a sampled video feature sequence is $P(\tilde{G}) = (b/n)^b$. As this is a very small probability, the number of potential distinct sampled video feature sets ($\tilde{G}$) is exceptionally large.

Fortunately, video features (frames) within a bucket are highly correlated as they originate from neighboring video frames which are generally very similar, and one may make a weak assumption that bucket population in bucket k may not contain sufficiently more information than any sampled feature $\boldsymbol{g}_{f(k)}$ from the bucket population.

In other words, if $\boldsymbol{g}_{f(k)}$ is sufficient statistic of the bucket k, then any sampled video feature sequence $\tilde{G}$ using SBFS contains sufficient statistics of G, and SBFS(G) is the sufficient statistic of video feature population G. Therefore, we can make the following proposition.

Proposition 1 *Any sampled video feature sequence $\tilde{G}$ from* SBFS *method is a sufficient statistic of video feature population G.*

The above proposition is very important as it allows us to train any complex model, such as a Transformer, on very long videos using SBFS with adequate guarantees. Next, we also present an interesting insight on how to pool features within a bucket. To do this, let us denote the j-th dimension of the feature vector $\boldsymbol{g}_i$ by g_i^j.

Proposition 2 *If the j-th dimension of vectors within the bucket k has a uniform population for all j, i.e. $g_{f(k)}^j$ is uniformly distributed on $[1, \mu_j]$ where μ_j is unknown, by the Fisher–Neyman factorization theorem (Fisher, 1922), the sufficient statistics of the population within the bucket k is given by the max-pooling operator over the bucket features* [5].

Although our SBFS can also be applied during inference, founded by Proposition 2, it is better to decouple the model from this stochastic component and instead utilize max pooling operator over the features of the bucket at inference time. This gives models increased stability when predicting, without sacrificing performance, as we showed in Section 5.3.

B Dynamic-Rate Feature Down-Sampling

In this section, we present details of our Dynamic-Rate Feature Down-Sampling ablation experiments. With this sampling heuristic, our intention was to

[5]Please check the supplementary material for details.

create buckets that satisfy the two following conditions. First, we would like each bucket to hold semantically similar features, using similarity on the embedding space as a proxy. Second, we aim to group features in a way such that the number of buckets l that hold the totality of the features in the video is smaller than the desired number of buckets b.

The bucket construction procedure works as follows. For an input feature sequence of length n, we use the cosine distance to compute the semantic similarity between each of the features in the video, and construct the pairwise distance matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$. We then start the process with a single bucket that contains only the first feature x_1, and add features to this bucket starting from x_2. Feature x_2 will be added to the bucket if and only if the cosine-distance $D_{1,2} < th$, where th is a threshold parameter, and otherwise a new bucket is started and the process is repeated until all the features have been processed. Once this is done, we evaluate the number of buckets l that were created, and if $l > b$, then we reduce the threshold th by a small margin (0.01) and generate all the buckets again. This process is repeated until the $l \leq b$ condition is satisfied. Please see Algorithm 1 below, for additional details.

Algorithm 1 Dynamic-rate Feature down-sampling using cosine similarity Algorithm

```
$\boldsymbol{D} = 1 -$ pairwise_distances$(V)$
$th = 1.0$
$flag = True$
index_sample = { }
while $flag$ do
    indx $= 0$
    $st = 0$
    $ed = 1$
    for $ed \leftarrow st$ to len$(V)$ do
        $s = \boldsymbol{D}_{st,ed}$
        if $s < th$ then
            samples_in_bucket = []
            for $i \leftarrow st$ to $ed$ do
                samples_in_bucket.append(i)
            end for
            index_sample[indx] = samples_in_bucket
            $st = ed$
            indx = indx + 1
        end if
        if indx <= bucket_size then:
            flag = False
        else
            $th = th - 0.01$
            index_sample = { }
        end if
    end for
end while
```

C Qualitative Results

In this section, we present qualitative results of our method for each one of datasets we use for evaluation. Ground truth (GT) and predictions in Figures 3, 4 and 5 are in seconds.

As seen on Figure 3, in the case of ActivityNet Caption, our method is able to localize the query *The man continue to rub the board using his polishing tools* with a high temporal intersection over union (IoU) of 80.41%. Though not visible in the figure, we also note that the end of this video is full of black frames and information about the creator, for example, webpage and logos. This exemplifies how ground truth annotations can be inaccurate, and how our model can adequately deal with these issues.

Figure 4 presents qualitative results for YookCookII dataset. In this case, we specifically present our predictions on one of the longest videos in the dataset, with a duration of 11 minutes and 46 seconds, and where the natural language query is *slice up the ginger finely*. As seen, our method obtains an impressive performance considering that the moment of interest lasts only 20 seconds. This figure also serves to exemplify one of the limitations of the bucketing approach we take. It is possible to see that the predictions of our model, though precise overall, add 0.96 seconds to both the start and end locations. This is a result of the maximum granularity given by the buckets and features in our system.

Finally, Figure 5 shows an example of the predictions of our model on the Charades-STA dataset. In this case, we also choose one of the longest videos in the data, with a duration of approximately 1 minute. For the query *person walks into room holding a bag*, our method obtains a good performance of 95.70% of temporal IoU.

D Experiment Details

- Total Parameters:
  - ExCL: 6.9M parameters.

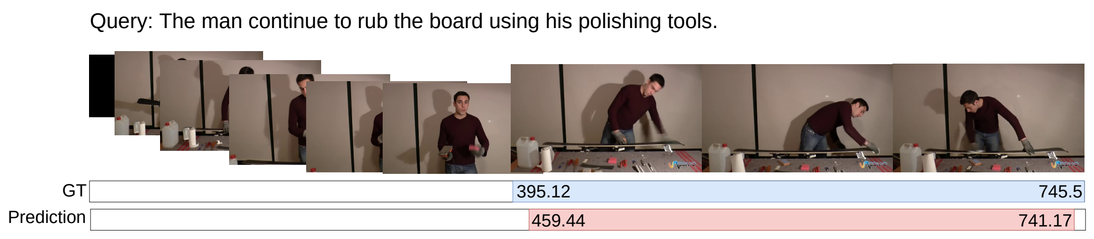
Query: The man continue to rub the board using his polishing tools.
GT
395.12
745.5
Prediction
459.44
741.17

Figure 3: Results of our method on the ActivityNet dataset in a very long video of 12 minutes and 25 seconds (745.5 seconds). Our method can localize the query *The man continue to rub the board using his polishing tools* with a temporal IoU of 80.41%

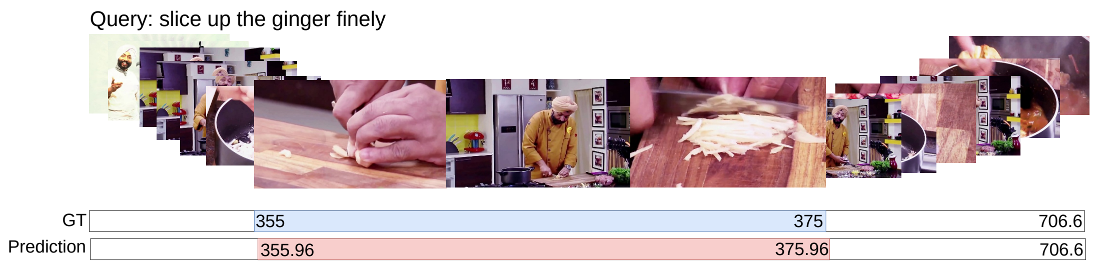

Figure 4: Qualitative result of our method on the YouCooKII dataset in one of the longest videos in the dataset (11 minutes and 46 seconds.) Our method obtains a temporal IoU of 90.83% for the query *slice up the ginger finely*.

  - TMLGA: 4.7M parameters.
  - DORi: 10.4M parameters.
  - LOCFORMER: 86.1M parameters.

- Hyper-parameters: Besides what is explained in the paper, we did not specifically run hyper-parameter exploration. On our early experiments, we tried some learning rate variations, but all of our reported results use 10^{-3} or respect the original setting of the model.

- Hardware requirements: Our experiments were performed on two kinds of GPUs, 16-GB NVIDIA V100 and 48-GB Quadro RTX 8000. The former we access by means of nodes on a large cluster, where each node has four such GPUs. Some experiments, especially on the ActivityNet dataset with DORi or LOCFORMER, were performed using data parallelism to speed up training time, but all of our proposed models can still run on single GPUs. For the experiments with the original DORi model or the baseline Transformer models without SBFS, we recommend using a GPU with at least 32GB of memory.

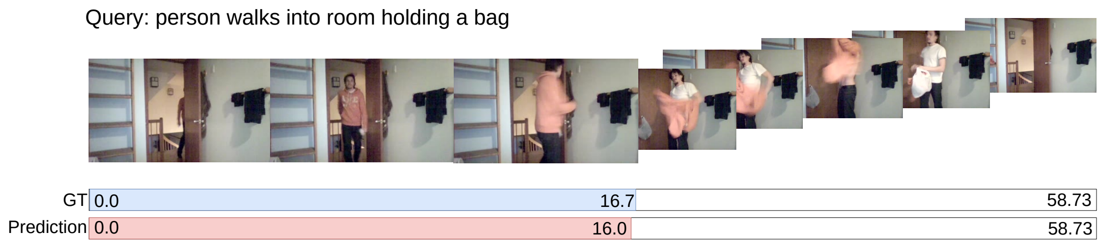

Figure 5: Charades-STA qualitative results on one of the longest videos in the dataset. For the query *person walks into room holding a bag*, our method obtains 95.70% IoU with respect to the ground truth annotations.