This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

End-to-End Dense Video Captioning with Masked Transformer

Luowei Zhou*
University of Michigan
luozhou@umich.edu

Yingbo Zhou*
Salesforce Research
yingbo.zhou@salesforce.com

Jason J. Corso
University of Michigan
jjcorso@eecs.umich.edu

Richard Socher
Salesforce Research
richard@socher.org

Caiming Xiong†
Salesforce Research
cxiong@salesforce.com

Abstract

Dense video captioning aims to generate text descriptions for all events in an untrimmed video. This involves both detecting and describing events. Therefore, all previous methods on dense video captioning tackle this problem by building two models, i.e. *an event proposal and a captioning model, for these two sub-problems. The models are either trained separately or in alternation. This prevents direct influence of the language description to the event proposal, which is important for generating accurate descriptions. To address this problem, we propose an end-to-end transformer model for dense video captioning. The encoder encodes the video into appropriate representations. The proposal decoder decodes from the encoding with different anchors to form video event proposals. The captioning decoder employs a masking network to restrict its attention to the proposal event over the encoding feature. This masking network converts the event proposal to a differentiable mask, which ensures the consistency between the proposal and captioning during training. In addition, our model employs a* self-attention *mechanism, which enables the use of efficient non-recurrent structure during encoding and leads to performance improvements. We demonstrate the effectiveness of this end-to-end model on ActivityNet Captions and YouCookII datasets, where we achieved 10.12 and 6.58 METEOR score, respectively.*

1. Introduction

Video has become an important source for humans to learn and acquire knowledge (*e.g.* video lectures, making sandwiches [20], changing tires [1]). Video content consumes high cognitive bandwidth, and thus is slow for humans to digest. Although the visual signal itself can sometimes disambiguate certain semantics, one way to make video content more easily and rapidly understood by humans is to compress it in a way that retains the semantics. This is particularly important given the massive amount of video being produced everyday. Video summarization [41] is one way of doing this, but it loses the language components of the video, which are particularly important in instructional videos. Dense video captioning [19]—describing events in the video with descriptive natural language—is another way of achieving this compression while retaining the language components.

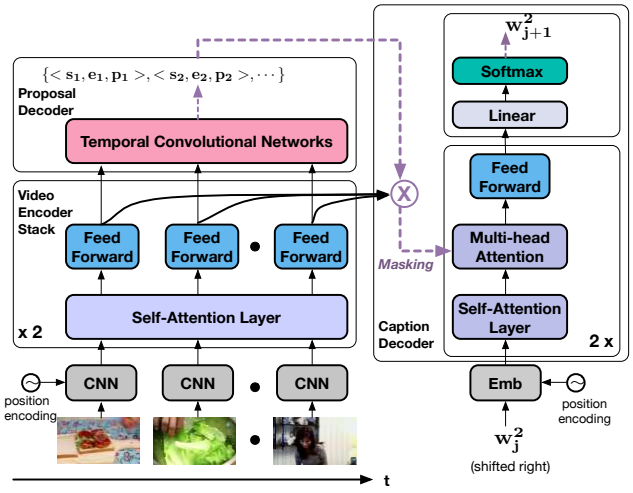

Figure 1. Dense video captioning is to localize (temporal) events from a video, which are then described with natural language sentences. We leverage temporal convolutional networks and self-attention mechanisms for precise event proposal generation and captioning.

Dense video captioning can be decomposed into two parts: event detection and event description. Existing methods tackle these two sub-problems using event proposal and captioning modules, and exploit two ways to combine them for dense video captioning. One way is to train the two

*Equal contribution
†Corresponding author

modules independently and generate descriptions for the best event proposals with the best captioning model [12]. The other way is to alternate training [19] between the two modules, *i.e*., alternate between i) training the proposal module only and ii) training the captioning module on the positive event proposals while fine-tuning the proposal module. However, in either case, the language information cannot have direct impacts on the event proposal.

Intuitively, the video event segments and language are closely related and the language information should be able to help localize events in the video. To this end, we propose an encoder-decoder based end-to-end model for doing dense video captioning (see Fig. 1). The encoder encodes the video frames (features) into the proper representation. The proposal decoder then decodes this representation with different anchors to form event proposals, *i.e*., start and end time of the event, and a confidence score. The captioning decoder then decodes the proposal specific representation using a masking network, which converts the event proposal into a differentiable mask. This continuous mask enables both the proposal and captioning decoder to be trained consistently, *i.e*. the proposal module now learns to adjust its prediction based on the quality of the generated caption. In other words, the language information from caption now is able to guide the visual model to generate more plausible proposals. In contrast to the existing methods where the proposal module solves a class-agnostic binary classification problem regardless the details in the video content, our model enforces the consistency between the content in the proposed video segment and the semantic information in the language description.

Another challenge for dense video captioning, and more broadly for sequence modeling tasks, is the need to learn a representation that is capable of capturing long term dependencies. Recurrent Neural Networks (RNN) are possible solutions to this problem, however, learning such representation is still difficult [23]. *Self-attention* [21, 24, 29] allows for an attention mechanism within a module and is a potential way to learn this long-range dependence. In self-attention the higher layer in the same module is able to attend to all states below it. This made the length of the paths of states from the higher layer to all states in the lower layer to be one, and thus facilitates more effective learning. The shorter path length facilitates learning these dependencies because larger gradients can now pass to all states. Transformer [29] implements a fast self-attention mechanism and has demonstrated its effectiveness in machine translation. Unlike traditional sequential models, transformer does not require unrolling across time, and therefore trains and tests much faster as compared to RNN based models. We employ transformer in both the encoder and decoder of our model.

Our main contributions are twofold. First, we propose an end-to-end model for doing dense video captioning. A differentiable masking scheme is proposed to ensure the consistency between proposal and captioning module during training. Second, we employ self-attention: a scheme that facilitates the learning of long-range dependencies to do dense video captioning. To the best of our knowledge, our model is the first one that does not use a RNN-based model for doing dense video captioning. In addition, we achieve competitive results on ActivityNet Captions [19] and YouCookII [42] datasets.

2. Related Work

Image and Video Captioning. In contrast to earlier video captioning papers, which are based on models like hidden Markov models and ontologies [39, 6], recent work on captioning is dominated by deep neural network-based methods [32, 34, 37, 43, 36, 26]. Generally, they use Convolutional Neural Networks (CNNs) [28, 15] for encoding video frames, followed by a recurrent language decoder, e.g., Long Short-Term Memory [17]. They vary mainly based on frame encoding, e.g., via mean-pooling [31, 10], recurrent nets [7, 30], and attention mechanisms [35, 22, 10]. The attention mechanism was initially proposed for machine translation [3] and has achieved top performance in various language generation tasks, either as temporal attention [35], semantic attention [10] or both [22]. Our work falls into the first of the three types. In addition to using cross-module attention, we apply self-attention [29] within each module.

Temporal Action Proposals. Temporal action proposals (TAP) aim to temporally localize action-agnostic proposals in a long untrimmed video. Existing methods formulate TAP as a binary classification problem and differ in how the proposals are proposed and discriminated from the background. Shuo et al. [27] propose and classify proposal candidates directly over video frames in a sliding window fashion, which is computationally expensive. More recently, inspired by the anchoring mechanism from object detection [25], two types of methods have been proposed—explicit anchoring [11, 42] and implicit anchoring [8, 4]. In the former case, each anchor is an encoding of the visual features between the anchor temporal boundaries and is classfied as action or background. In implicit anchoring, recurrent networks encode the video sequence and, at each anchor center, multiple anchors with various sizes are proposed based on the same visual feature. So far, explicit anchoring methods accompanied with location regression yield better performance [11]. Our proposal module is based upon Zhou et al. [42], which is designed to detect long complicated events rather than actions. We further improve the framework with a temporal convolutional proposal network and self-attention based context encoding.

Dense Video Captioning. The video paragraph captioning method proposed by Yu *et al*. [40] generates sentence

descriptions for temporally localized video events. However, the temporal locations of each event are provided beforehand. Das et al. [6] generates dense captions over the entire video using sparse object stitching, but their work relies on a top-down ontology for the actual description and is not data-driven like the recent captioning methods. The most similar work to ours is Krishna et al. [19] who introduce a dense video captioning model that learns to propose the event locations and caption each event with a sentence. However, they combine the proposal and the captioning modules through co-training and are not able to take advantage of language to benefit the event proposal [16]. To this end, we propose an end-to-end framework for doing dense video captioning that is able to produce proposal and description simultaneously. Also, our work directly incorporates the semantics from captions to the proposal module.

3. Preliminary

In this section we introduce some background on Transformer [29], which is the building block for our model. We start by introducing the *scaled dot-product attention*, which is the foundation of transformer. Given a query $q_i \in \mathbb{R}^d$ from all T' queries, a set of keys $k_t \in \mathbb{R}^d$ and values $v_t \in \mathbb{R}^d$ where $t = 1, 2, ..., T$, the scaled dot-product attention outputs a weighted sum of values v_t, where the weights are determined by the dot-products of query q and keys k_t. In practice, we pack k_t and v_t into matricies $K = (k_1, ..., k_T)$ and $V = (v_1, ..., v_T)$, respectively. The attention output on query q is:

$$A(q_i, K, V) = V \frac{\exp\left\{K^T q_i / \sqrt{d}\right\}}{\sum_{t=1}^{T} \exp\{k_t^T q_i / \sqrt{d}\}} \tag{1}$$

The *multi-head attention* consists of H paralleled scaled dot-product attention layers called "head", where each "head" is an independent dot-product attention. The attention output from multi-head attention is as below:

$$\text{MA}(q_i, K, V) = W^O \begin{pmatrix} \text{head}_1 \\ \cdots \\ \text{head}_H \end{pmatrix} \tag{2}$$

$$\text{head}_j = A(W_j^q q_i, W_j^K K, W_j^V V) \tag{3}$$

where $W_j^q, W_j^K, W_j^V \in \mathbb{R}^{\frac{d}{H} \times d}$ are the independent head projection matrices, $j = 1, 2, ..., H$, and $W^O \in \mathbb{R}^{d \times d}$.

This formulation of attention is quite general, for example when the query is the hidden states from the decoder, and both the keys and values are all the encoder hidden states, it represents the common cross-module attention. *Self-attention* [29] is another case of multi-head attention where the queries, keys and values are all from the same hidden layer (see also in Fig. 2).

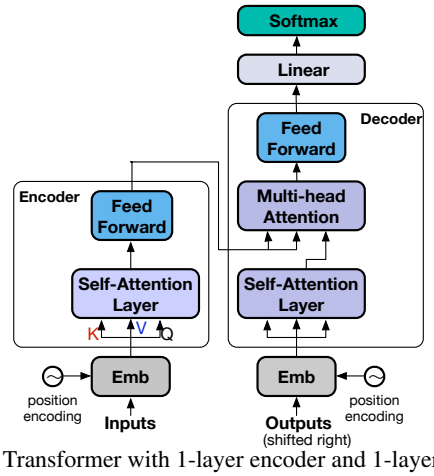

Figure 2. Transformer with 1-layer encoder and 1-layer decoder.

Now we are ready to introduce Transformer model, which is an encoder-decoder based model that is originally proposed for machine translation [29]. The building block for Transformer is multi-head attention and a pointwise feed-forward layer. The pointwise feed-forward layer takes the input from multi-head attention layer, and further transforms it through two linear projections with ReLU activation. The feed-forward layer can also be viewed as two convolution layers with kernel size one. The encoder and decoder of Transformer is composed by multiple such building blocks, and they have the same number of layers. The decoder from each layer takes input from the encoder of the same layer as well as the lower layer decoder output. Self-attention is applied to both encoder and decoder. Cross-module attention between encoder and decoder is also applied. Note that the self-attention layer in the decoder can only attend to the current and previous positions to preserve the auto-regressive property. Residual connection [15] is applied to all input and output layers. Additionally, layer normalization [2] (LayerNorm) is applied to all layers. Fig. 2 shows a one layered transformer.

4. End-to-End Dense Video Captioning

Our end-to-end model is composed of three parts: a video encoder, a proposal decoder, and a captioning decoder that contains a mask prediction network to generate text description from a given proposal. The video encoder is composed of multiple self-attention layers. The proposal decoder takes the visual features from the encoder and outputs event proposals. The mask prediction network takes the proposal output and generates a differentiable mask for a certain event proposal. To make the decoder caption the current proposal, we then apply this mask by element-wise multiplication between it, the input visual embedding and all outputs from proposal encoder. In the following sections, we illustrate each component of our model in detail.

4.1. Video Encoder

Each frame x_t of the video $X = \{x_1, \ldots, x_T\}$ is first encoded to a continuous representation $F^0 = \{f_1^0, \ldots, f_T^0\}$. It is then fed forward to L encoding layers, where each layer learns a representation $F^{l+1} = V(F^l)$ by taking input from previous layer l,

$$\mathrm{V}(F^l) = \Psi(\mathrm{PF}(\Gamma(F^l)), \Gamma(F^l)) \tag{4}$$

$$\Gamma(F^l) = \begin{pmatrix} \Psi(\mathrm{MA}(f_1^l, F^l, F^l), f_1^l)^\top \\ \cdots \\ \Psi(\mathrm{MA}(f_T^l, F^l, F^l), f_T^l)^\top \end{pmatrix}^\top \tag{5}$$

$$\Psi(\alpha, \beta) = \mathrm{LayerNorm}(\alpha + \beta) \tag{6}$$

$$\mathrm{PF}(\gamma) = M_2^l \max(0, M_1^l \gamma + b_1^l) + b_2^l \tag{7}$$

where $\Psi(\cdot)$ represents the function that performs layer normalization on the residual output, $\mathrm{PF}(\cdot)$ denotes the 2-layered feed-forward neural network with ReLU nonlinearity for the first layer, M_1^l, M_2^l are the weights for the feed-forward layers, and b_1^l, b_2^l are the biases. Notice the self-attention used in eq. 5. At each time step t, f_t^l is given as the query to the attention layer and the output is the weight sum of f_t^l, $t = 1, 2, ..., T$, which encodes not only the information regarding the current time step, but also all other time steps. Therefore, each time step of the output from the self-attention is able to encode all context information. In addition, it is easy to see that the length of the path between time steps is only one. In contrast to recurrent models, this makes the gradient update independent with respect to their position in time, and thus makes learning potential dependencies amongst distant frames easier.

4.2. Proposal Decoder

Our event proposal decoder is based on ProcNets [42], for its state-of-the-art performance on long dense event proposals. We adopt the same anchor-offset mechanism as in ProcNets and design a set of N explicit anchors for event proposals. Each anchor-based proposal is represented by an event proposal score $P_e \in [0, 1]$ and two offsets: center θ_c and length θ_l. The associated anchor has length l_a and center c_a. The proposal boundaries (S_p, E_p) are determined by the anchor locations and offsets:

$$\begin{aligned} c_p &= c_a + \theta_c l_a \quad l_p = l_a \exp\{\theta_l\}, \\ S_p &= c_p - l_p/2 \quad E_p = c_p + l_p/2. \end{aligned} \tag{8}$$

These proposal outputs are obtained from temporal convolution (*i.e.* 1-D convolutions) applied on the last layer output of the visual encoder. The score indicates the likelihood for a proposal to be an event. The offsets are used to adjust the proposed segment boundaries from the associated anchor locations. We made following changes to ProcNets:

- The sequential prediction module in ProcNets is removed, as the event segments in a video are not closely coupled and the number of events is small in general.
- Use input from a multi-head self-attention layer instead of a bidirectional LSTM (Bi-LSTM) layer [14].
- Use multi-layer temporal convolutions to generate the proposal score and offsets. The temporal convolutional network contain three 1-D conv. layers, with batch normalization [18]. We use ReLU activation for hidden layers.
- In our model, the conv. stride depends on kernel size ($\lceil \frac{kernel\ size}{s} \rceil$) versus always 1 in ProcNets[1].

We encode the video context by a self-attention layer as it has potential to learn better context representation. Changing stride size based on kernel size reduces the number of longer proposals so that the training samples is more balanced, because a larger kernel size makes it easier to get good overlap with ground truth. It also speeds up training as the number of long proposals is reduced.

4.3. Captioning Decoder

Masked Transformer. The captioning decoder takes input from both the visual encoder and the proposal decoder. Given a proposal tuple (P_e, S_p, E_p) and visual representations $\{F^1, \ldots, F^L\}$, the L-layered captioning decoder generates the t-th word by doing the following

$$Y_{\leq t}^{l+1} = \mathrm{C}(Y_{\leq t}^l) = \Psi(\mathrm{PF}(\Phi(Y_{\leq t}^l)), \Phi(Y_{\leq t}^l)) \tag{9}$$

$$\Phi(Y_{\leq t}^l) = \begin{pmatrix} \Psi(\mathrm{MA}(\Omega(Y_{\leq t}^l)_1, \hat{F}^l, \hat{F}^l), \Omega(Y_{\leq t}^l)_1) \\ \cdots \\ \Psi(\mathrm{MA}(\Omega(Y_{\leq t}^l)_t, \hat{F}^l, \hat{F}^l), \Omega(Y_{\leq t}^l)_t) \end{pmatrix} \tag{10}$$

$$\Omega(Y_{\leq t}^l) = \begin{pmatrix} \Psi(\mathrm{MA}(y_1^l, Y^l, Y^l), y_1^l)^\top \\ \cdots \\ \Psi(\mathrm{MA}(y_t^l, Y^l, Y^l), y_t^l)^\top \end{pmatrix} \tag{11}$$

$$\hat{F}^l = f_M(S_p, E_p) \odot F^l \tag{12}$$

$$p(w_{t+1}|X, Y_{\leq t}^L) = \mathrm{softmax}(W^V y_{t+1}^L) \tag{13}$$

where y_i^0 represents word vector, $Y_{\leq t}^l = \{y_1^l, \ldots, y_t^l\}$, w_{t+1} denotes the probability of each word in the vocabulary for time $t+1$, $W^V \in \mathbb{R}^{\nu \times d}$ denotes the word embedding matrix with vocabulary size ν, and $\odot$ indicates elementwise multiplication. $\mathrm{C}(\cdot)$ denotes the decoder representation, *i.e.* the output from feed-forward layer in Fig. 1. $\Phi(\cdot)$ denotes the cross module attention that use the current decoder states to attend to encoder states (*i.e.* multi-head attention in Fig. 1). $\Omega(\cdot)$ represents the self-attention in decoder. Notice that the subscript $\leq t$ restricts the attention only on the already generated words. $f_M : \mathbb{R}^2 \mapsto [0, 1]^T$ is a masking function that output values (near) zero when outside the predicted starting and ending locations, and (near) one otherwise. With this

[1] s is a scalar that affects the convolution stride for different kernel size

function, the receptive region of the model is restricted to the current segment so that the visual representation focuses on describing the current event. Note that during decoding, the encoder performs the forward propagation again so that the representation of each encoder layer contains only the information for the current proposal (see eq. 12). This is different from simply multiplying the mask with the existing representation from the encoder during proposal prediction, since the representation of the latter still contains information that is outside the proposal region. The representation from the L-th layer of captioning decoder is then used for predicting the next word for the current proposal using a linear layer with softmax activation (see eq. 13).

Differentiable Proposal Mask. We cannot choose any arbitrary function for f_M as a discrete one would prevent us from doing end-to-end training. We therefore propose to use a fully differentiable function to obtain the mask for visual events. This function f_M maps the predicted proposal location to a differentiable mask $M \in \mathbb{R}^T$ for each time step $i \in \{1, \ldots, T\}$.

$$f_M(S_p, E_p, S_a, E_a, i) = \sigma(g(\\ [\rho(S_p, :), \rho(E_p, :), \rho(S_a, :), \rho(E_e, :), \text{Bin}(S_a, E_a, :)])) \quad (14)$$

$$\rho(pos, i) = \begin{cases} \sin(pos/10000^{i/d}) & i \textit{ is even} \\ \cos(pos/10000^{(i-1)/d}) & \textit{otherwise} \end{cases} \quad (15)$$

$$\text{Bin}(S_a, E_a, i) = \begin{cases} 1 & \textit{if } i \in [S_a, E_a] \\ 0 & \textit{otherwise} \end{cases} \quad (16)$$

where S_a and E_a are the start and end position of anchor, $[\cdot]$ denotes concatenation, $g(\cdot)$ is a continuous function, and $\sigma(\cdot)$ is the logistic sigmoid function. We choose to use a multilayer perceptron to parameterize g. In other words, we have a feed-forward neural network that takes the positional encoding from the anchor and predicted boundary positions and the corresponding binary mask to predict the continuous mask. We use the same positional encoding strategy as in [29].

Directly learning the mask would be difficult and unnecessary, since we would already have a reasonable boundary prediction from the proposal module. Therefore, we use a gated formulation that lets the model choose between the learned continuous mask and the discrete mask obtained from the proposal module. More precisely, the gated masking function f_{GM} is

$$f_{GM}(S_p, E_p, S_a, E_a, i) = \\ P_e \text{Bin}(S_p, E_p, i) + (1 - P_e) f_M(S_p, E_p, S_a, E_a, i) \quad (17)$$

Since the proposal score $P_e \in [0, 1]$, it now acts as a gating mechanism. This can also be viewed as a modulation between the continuous and proposal masks, the continuous mask is used as a supplement for the proposal mask in case the confidence is low from the proposal module.

4.4. Model Learning

Our model is fully differentiable and can be trained consistently from end-to-end The event proposal anchors are sampled as follows. Anchors that have overlap greater than 70% with any ground-truth segments are regarded as positive samples and ones that have less than 30% overlap with all ground-truth segments are negative. The proposal boundaries for positive samples are regressed to the ground-truth boundaries (offsets). We randomly sample $U = 10$ anchors from positive and negative anchor pools that correspond to one ground-truth segment for each mini-batch.

The loss for training our model has four parts: the regression loss $\mathcal{L}_r$ for event boundary prediction, the binary cross entropy mask prediction loss $\mathcal{L}_m$, the event classification loss $\mathcal{L}_e$ (*i.e.* prediction P_e), and the captioning model loss $\mathcal{L}_c$. The final loss $\mathcal{L}$ is a combination of these four losses,

$$\begin{aligned} \mathcal{L}_r &= \text{Smooth}_{\ell 1}(\hat{\theta}_c, \theta_c) + \text{Smooth}_{\ell 1}(\hat{\theta}_l, \theta_l) \\ \mathcal{L}_m^i &= \text{BCE}(Bin(S_p, E_p, i), f_M(S_p, E_p, S_a, E_a, i)) \\ \mathcal{L}_e &= \text{BCE}(\hat{P}_e, P_e) \\ \mathcal{L}_c^t &= \text{CE}(\hat{w}_t, p(w_t | X, Y_{\leq t-1}^L)) \\ \mathcal{L} &= \lambda_1 \mathcal{L}_r + \lambda_2 \sum_i \mathcal{L}_m^i + \lambda_3 \mathcal{L}_e + \lambda_4 \sum_t \mathcal{L}_c^t \end{aligned}$$

where $\text{Smooth}_{\ell 1}$ is the smooth ℓ_1 loss defined in [13], BCE denotes binary cross entropy, CE represents cross entropy loss, $\hat{\theta}_c$ and $\hat{\theta}_l$ represent the ground-truth center and length offset with respect to the current anchor, $\hat{P}_e$ is the ground-truth label for the proposed event, $\hat{w}_t$ denotes the ground-truth word at time step t, and $\lambda_{1\ldots4} \in \mathbb{R}^+$ are the coefficients that balance the contribution from each loss.

Simple Single Stage Models. The key for our proposed model to work is not the single stage learning of a compositional loss, but the ability to keep the consistency between the proposal and captioning. For example, we could make a single-stage trainable model by simply sticking them together with multi-task learning. More precisely, we can have the same model but choose a non-differentiable masking function f_M in eq. 12. The same training procedure can be applied for this model (see the following section). Since the masking function would then be non-differentiable, error from the captioning model cannot be back propagated to modify the proposal predictions. However, the captioning decoder is still able to influence the visual representation that is learned from the visual encoder. This may be undesirable, as the updates the visual representation may lead to worse performance for the proposal decoder. As a baseline, we also test this single-stage model in our experiments.

5. Implementation Details

For the proposal decoder, the temporal convolutional networks take the last encoding output from video encoder

as the input. The sizes of the temporal convolution kernels vary from 1 to 251 and we set the stride factor s to 50. For our Transformer model, we set the model dimension $d = 1024$ (same as the Bi-LSTM hidden size) and set the hidden size of feed-forward layer to 2048. We set number of heads (H) to 8. In addition to the residual dropout and attention dropout layers in Transformer, we add a 1-D dropout layer at the visual input embedding to avoid overfitting. We use recurrent dropout proposed in [9] for this 1-D dropout. Due to space limits, more details are included in the supplementary material.

6. Experiments

6.1. Datasets

ActivityNet Captions [19] and YouCookII [42] are the two largest datasets with temporal event segments annotated and described by natural language sentences. ActivityNet Captions contains 20k videos, and on average each video has 3.65 events annotated. YouCookII has 2k videos and the average number of segments per video is 7.70. The train/val/test splits for ActivityNet Captions are 0.5:0.25:0.25 while for YouCookII are 0.66:0.23:0.1. We report our results from both datasets on the validation sets. For ActivityNet Captions, we also show the testing results on the evaluation server while the testing set for YouCookII is not available.

Data Preprocessing. We down-sample the video every 0.5s and extract the 1-D appearance and optical flow features per frame, as suggested by Xiong et al. [33]. For appearance features, we take the output of the "Flatten-673" layer in ResNet-200 [15]; for optical flow features, we extract the optical flow from 5 contiguous frames, encode with BN-Inception [18] and take output of the "global-pool" layer. Both networks are pre-trained on the ActivityNet dataset [5] for the action recognition task. We then concatenate the two feature vector and further encode with a linear layer. We set the window size T to 480. The input is zero padded in case the number of sampled frames is smaller than the size of the window. Otherwise, the video is truncated to fit the window. Note that we do not fine-tune the visual features for efficiency considerations, however, allowing fine-tuning may lead to better performance.

6.2. Baseline and Metrics

Baselines. Most of the existing methods can only caption an entire video or specified video clip. For example, LSTM-YT [31], S2YT [30], TempoAttn [35], H-RNN [40] and DEM [19]. The most relevant baseline is TempoAttn, where the model temporally attends on visual sequence inputs as the input of LSTM language encoder. For a fair comparison, we made the following changes to the original TempoAttn. First, all the methods take the same visual feature input. Second, we add a Bi-LSTM context encoder to TempoAttn while our method use self-attention context encoder. Third, we apply temporal attention on Bi-LSTM output for all the language decoder layers in TempoAttn since our decoder has attention each layer. We name this baseline Bi-LSTM+TempoAttn. Since zero inputs deteriorates Bi-LSTM encoding, we only apply the masking on the output of the LSTM encoder when it is passed to the decoder. We also compare with a a simple single-stage Masked Transformer baseline as mentioned in section 4.4, where the model employs a discrete binary mask.

For event proposals, we compare our self-attention transformer-based model with ProcNets and our own baseline with Bi-LSTM. For captioning-only models, we use the same baseline as the full dense video captioning but instead, replace the learned proposals with ground-truth proposals. Results for other dense captioning methods (*e.g.* the best published method DEM [19]) are not available on the validation set nor is the source code released. So, we compare our methods against those methods that participated in CVPR 2017 ActivityNet Video Dense-captioning Challenge [12] for test set performance on ActivityNet.

Evaluation Metrics. For ground-truth segment captioning, we measure the captioning performance with most commonly-used evaluation metrics: BLEU{3,4} and METEOR. For dense captioning, the evaluate metric takes both proposal accuracy and captioning accuracy into account. Given a tIoU threshold, if the proposal has an overlapping larger than the threshold with any ground-truth segments, the metric score is computed for the generated sentence and the corresponding ground-truth sentence. Otherwise, the metric score is set to 0. The scores are then averaged across all the proposals and finally averaged across all the tIoU thresholds–0.3, 0.5, 0.7, 0.9 in this case.

6.3. Comparison with State-of-the-Art Methods

We compare our proposed method with baselines on the ActivityNet Caption dataset. The validation and testing set results are shown in Tab. 1 and 2, respectively. All our models outperform the LSTM-based models by a large margin, which may be attributed to their better ability of modeling long-range dependencies.

We also test the performance of our model on the YouCookII dataset, and the result is shown in Tab. 3. Here, we see similar trend on performance. Our transformer based model outperforms the LSTM baseline by a significant amount. However, the results on learned proposals are much worse as compared to the ActivityNet dataset. This is possibly because of small objects, such as utensils and ingredients, are hard to detect using global visual features but are crucial for describing a recipe. Hence, one future extension for our work is to incorporate object detectors/trackers [38, 39] into the current captioning system.

Table 1. Captioning results from ActivityNet Caption Dataset with learned event proposals. All results are on the validation set and all our models are based on 2-layer Transformer. We report BLEU (B) and METEOR (M). All results are on the validation set. Top scores are highlighted.

Method	B@3	B@4	M
Bi-LSTM +TempoAttn	2.43	1.01	7.49
Masked Transformer	4.47	2.14	9.43
End-to-end Masked Transformer	**4.76**	**2.23**	**9.56**

Table 2. Dense video captioning challenge leader board results. For results from the same team, we keep the highest one.

Method	METEOR
DEM [19]	4.82
Wang et al.	9.12
Jin et al.	9.62
Guo et al.	9.87
Yao et al.[2](Ensemble)	12.84
Our Method	**10.12**

Table 3. Recipe generation benchmark on YouCookII validation set. GT proposals indicate the ground-truth segments are given during inference.

Method	**GT Proposals**		**Learned Proposals**	
	B@4	M	B@4	M
Bi-LSTM +TempoAttn	0.87	8.15	0.08	4.62
Our Method	**1.42**	**11.20**	**0.30**	**6.58**

We show qualitative results in Fig. 3 where the proposed method generates captions with more relevant semantic information. More visualizations are in the supplementary.

6.4. Model Analysis

In this section we perform experiments to analyze the effectiveness of our model on different sub-tasks of dense video captioning.

Video Event Proposal. We first evaluate the effect of self-attention on event proposal, and the results are shown in Tab. 4. We use standard average recall (AR) metric [8, 12] given 100 proposals. Bi-LSTM indicates our improved ProcNets-prop model by using temporal convolutional and large kernel strides. We use our full model here, where the context encoder is replaced by our video encoder. We have noticed that the anchor sizes have a large impact on the results. So, for fair comparison, we maintain the same anchor sizes across all three methods. Our proposed Bi-LSTM model gains a 7% relative improvement from the baseline results from the deeper proposal network and more balanced anchor candidates. Our video encoder further yields a 4.5% improvement from our recurrent nets-based model. We show the recall curve under high tIoU threshold (0.8) in Fig. 4 follow the convention [19]. DAPs [8], is initially proposed for short action proposals and adapted later for long event proposal [19]. The proposed models outperforms DAPs-event and ProcNets-prop by significant margins. Transformer based and Bi-LSTM based models yield similar recall results given sufficient number of proposals (100), while our self-attention encoding model is more accurate when the allowed number of proposals is small.

[2]This work is unpublished. It employs external data for model training and the final prediction is obtained from an ensemble of models.

Table 4. Event proposal results from ActivityNet Captions dataset. We compare our proposed methods with our baseline method ProcNets-prop on the validation set.

Method	Average Recall (%)
ProcNets-prop [42]	47.01
Bi-LSTM (ours)	50.65
Self-Attn (our)	**52.95**

Table 5. Captioning results from ActivityNet Caption Dataset with ground-truth proposals. All results are on the validation set. Top two scores are highlighted.

Method	B@3	B@4	M
Bi-LSTM +TempoAttn	4.8	2.1	10.02
Our Method			
1-layer	**5.80**	2.66	10.92
2-layer	5.69	2.67	11.06
4-layer	**5.70**	**2.77**	**11.11**
6-layer	5.66	**2.71**	**11.10**

Dense Video Captioning. Next, we look at the dense video captioning results in an ideal setting: doing the captioning based on the ground-truth event segments. This will give us an ideal captioning performance since all event proposals are accurate. Because we need access to ground-truth event proposal during test time, we report the results on validation set[3] (see Tab. 5). The proposed Masked Transformer (section 4.3) outperforms the baseline by a large margin (by more than 1 METEOR point). This directly substantiates the effectiveness of the transformer on both visual and language encoding and multi-head temporal attention. We notice that as the number of encoder and decoder layers increases, the performance gets further boosts by 1.3%-1.7%. As can be noted here, the 2-layer transformer strikes a good balance point between performance and computation, and thus we use 2-layer transformer for all our experiments.

Analysis on Long Events. As mentioned in section 4.1, learning long-range dependencies should be easier with self-attention, since the next layer observes information from all time steps of the previous layer. To validate this hypothesis directly, we test our model against the LSTM

[3]The results are overly optimistic, however, it is fine here since we are interested in the best situation performance. The comparison is also fair, since all methods are tuned to optimize the validation set performance.

Ground-truth
Event 0: Two teams are playing volleyball in a indoor court.
Event 1: Two teams wearing dark uniforms are doing a volleyball competition, then appears a team with yellow t-shirts.
Event 2: Then, a boy with a red t-shirt serves the ball and the teams start to hit and running to pass the ball, then another team wearing green shorts enters the court.
Event 3: After, team wearing blue uniform competes with teams wearing white and red uniforms.

Masked Trans. (ours)
Event 0: a large group of people are seen standing around a gymnasium playing a game of volleyball
Event 1: the people in black and yellow team scores a goal
Event 2: the people continue playing the game back and fourth while the people watch on the sidelines
Event 3: the people continue playing the game back and fourth while the camera captures their movements

Bi-LSTM+TempoAttn
Event 0: a large group of people are seen standing around a field playing a game of soccer
Event 1: the players are playing the game of tug of war
Event 2: the people continue playing with one another and end by walking away
Event 3: the people continue playing and ends with one another and the other

Ground-truth
Event 0: A man is writing something on a clipboard.
Event 1: A man holds a ball behind his head and spins around several times and throws the ball.
Event 2: People use measuring tape to measure the distance.

Masked Trans. (ours)
Event 0: a man is seen standing in a large circle and leads into a man holding a ball and
Event 1: the man spins the ball around and throws the ball
Event 2: the man throws the ball and his throw the distance

Bi-LSTM+TempoAttn
Event 0: a man is seen standing on a field with a man standing on a field
Event 1: he throws the ball and throws it back and forth
Event 2: he throws the ball and throws it back and forth

Figure 3. Qualitative results on ActivityNet Captions. The color bars represent different events. Colored text highlight relevant content to the event. Our model generates more relevant attributes as compared to the baseline.

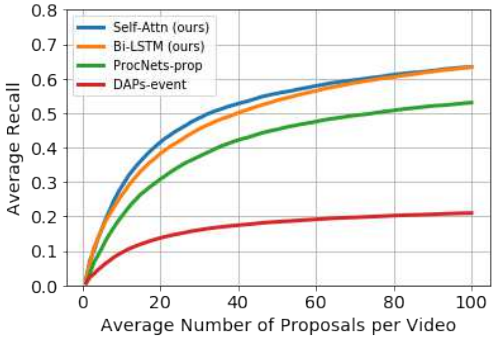

Figure 4. Event proposal recall curve under tIoU threshold 0.8 with average 100 proposals per video.

Table 6. Evaluating only long events from ActivityNet Caption Dataset. GT proposals indicate the ground-truth segments are given during inference.

	GT Proposals		**Learned Proposals**	
Method	B@4	M	B@4	M
Bi-LSTM +TempoAttn	0.84	5.39	0.42	3.99
Our Method	**1.13**	**5.90**	**1.04**	**5.93**

baseline on longer event segments (where the events are at least 50s long) from the ActivityNet Caption dataset, where learning the long-range dependencies are crucial for achieving good performance. It is clear from the result (see Tab. 6) that our transformer based model performs significantly better than the LSTM baseline. The discrepancy is even larger when the model needs to learn both the proposal and captioning, which demonstrate the effectiveness of self-attention in facilitate learning long range dependencies.

7. Conclusion

We propose an end-to-end model for dense video captioning. The model is composed of an encoder and two decoders. The encoder encodes the input video to proper visual representations. The proposal decoder then decodes from this representation with different anchors to form video event proposals. The captioning decoder employs a differentiable masking network to restrict its attention to the proposal event, ensures the consistency between the proposal and captioning during training. In addition, we propose to use self-attention for dense video captioning. We achieved significant performance improvement on both event proposal and captioning tasks as compared to RNN-based models. We demonstrate the effectiveness of our models on ActivityNet Captions and YouCookII dataset.

Acknowledgement. The technical work was performed while Luowei was an intern at Salesforce Research. This work is also partly supported by ARO W911NF-15-1-0354 and DARPA FA8750-17-2-0112. This article solely reflects the opinions and conclusions of its authors but not the funding agents.

References

[1] J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, pages 4575–4583, 2016. 1

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 2

[4] S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. C. Niebles. Sst: Single-stream temporal action proposals. In *CVPR*, pages 2911–2920, 2017. 2

[5] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, pages 961–970, 2015. 6

[6] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2, 3

[7] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, pages 2625–2634, 2015. 2

[8] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem. Daps: Deep action proposals for action understanding. In *ECCV*, pages 768–784, 2016. 2, 7

[9] Y. Gal and Z. Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*, pages 1019–1027, 2016. 6

[10] Z. Gan, C. Gan, X. He, Y. Pu, K. Tran, J. Gao, L. Carin, and L. Deng. Semantic compositional networks for visual captioning. *CVPR*, 2017. 2

[11] J. Gao, Z. Yang, C. Sun, K. Chen, and R. Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. *ICCV*, 2017. 2

[12] B. Ghanem, J. C. Niebles, C. Snoek, F. Caba Heilbron, H. Alwassel, R. Khrisna, V. Escorcia, K. Hata, and S. Buch. Activitynet challenge 2017 summary. *arXiv preprint arXiv:1710.08011*, 2017. 2, 6, 7

[13] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 5

[14] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. 4

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2, 3, 6

[16] F. C. Heilbron, W. Barrios, V. Escorcia, and B. Ghanem. Scc: Semantic context cascade for efficient action detection. 3

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[18] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 4, 6

[19] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles. Dense-captioning events in videos. *ICCV*, 2017. 1, 2, 3, 6, 7

[20] H. Kuehne, A. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, pages 780–787, 2014. 1

[21] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017. 2

[22] Y. Pan, T. Yao, H. Li, and T. Mei. Video captioning with transferred semantic attributes. *CVPR*, 2017. 2

[23] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *ICML*, pages 1310–1318, 2013. 2

[24] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017. 2

[25] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 2

[26] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. *arXiv preprint arXiv:1612.00563*, 2016. 2

[27] Z. Shou, D. Wang, and S.-F. Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, pages 1049–1058, 2016. 2

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *NIPS*, 2017. 2, 3, 5

[30] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *ICCV*, pages 4534–4542, 2015. 2, 6

[31] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014. 2, 6

[32] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015. 2

[33] Y. Xiong, L. Wang, Z. Wang, B. Zhang, H. Song, W. Li, D. Lin, Y. Qiao, L. Van Gool, and X. Tang. Cuhk & ethz & siat submission to activitynet challenge 2016. *arXiv preprint arXiv:1608.00797*, 2016. 6

[34] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015. 2

[35] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *CVPR*, 2015. 2, 6

[36] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei. Boosting image captioning with attributes. *ICCV*, 2017. 2

[37] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *CVPR*, pages 4651–4659, 2016. 2

[38] H. Yu, N. Siddharth, A. Barbu, and J. M. Siskind. A compositional framework for grounding language inference, generation, and acquisition in video. *J. Artif. Intell. Res.(JAIR)*, 52:601–713, 2015. 6

[39] H. Yu and J. M. Siskind. Grounded language learning from video described with sentences. In *ACL (1)*, pages 53–63, 2013. 2, 6

[40] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video paragraph captioning using hierarchical recurrent neural networks. *CVPR*, 2016. 2, 6

[41] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman. Video summarization with long short-term memory. In *ECCV*, pages 766–782. Springer, 2016. 1

[42] L. Zhou, C. Xu, and J. J. Corso. Towards automatic learning of procedures from web instructional videos. *AAAI*, 2018. 2, 4, 6, 7

[43] L. Zhou, C. Xu, P. Koch, and J. J. Corso. Watch what you just said: Image captioning with text-conditional attention. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 305–313. ACM, 2017. 2