Salient Developments in Video Captioning

Ruchika Chavhan
ruchikachavhan@iitb.ac.in

Indian Institute of Technology Bombay

September 9, 2021

Overview

Problem Statment

Input: A sequence of video frames $V = (v_1, v_2, ..., v_n)$.
Output: A sentence $S = (w_1, w_2, ..., w_m)$

Conditional probability of output sequence S given an input sequence V is given by $p(w_1, w_2, ..., w_m | v_1, v_2, ..., v_n)$

Therefore, we would like to maximise the log likelihood of sentence S given video frames V, and captioning model parameters θ

$$\theta^* = \text{argmax} \sum_{V,S} \log p(S|V) \tag{1}$$

If we assume a model that generates a the word sequence in order, then

$$\log p(S|V) = \sum_{t=0}^{N} \log p(w_t | V, w_1, w_2, ..., w_{t-1}) \tag{2}$$

Problem Statment (cont.)

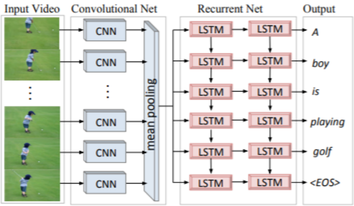
Common Datasets:

- MSR-VTT: Microsoft Research Video to Text
- MSVD: YouTube clips with captions
- ActivityNet Captions, YouCook: Captions available for temporal segments of each video. Mostly used for dense captioning

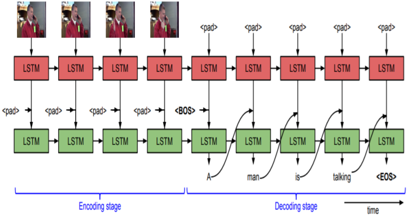
Evaluation Metrics:

- ROUGE-L: Relative length of Longest Common Subsequence
- BLEU-n: Percentage of similar n-grams
- METEOR: Harmonic mean of unigram precision and recall
- CIDEr: Cosine similarities between Term Frequency Inverse Document Frequency (TF-IDF)

Sequence to sequence models



(a) Translating Videos to Natural Language Using Deep Recurrent Neural Networks, NAACL-HLT 2015



(b) Sequence to Sequence – Video to Text, ICCV 2015

$$
\begin{aligned}
z_t &= W_{zh} h_t; \\
p(w \mid z_t) &= \frac{\exp(W_w z_t)}{\sum_{w' \in D} \exp(W_{w'} z_t)}
\end{aligned} \tag{3}
$$

Sequence to sequence models (cont.)

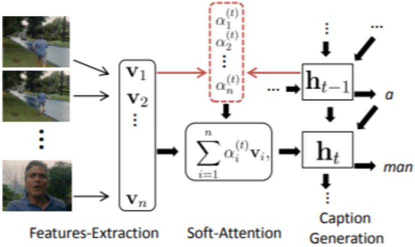


Figure 2: Describing Videos by Exploiting Temporal Structure, ICCV 2015.

$$e_i^{(t)} = \mathrm{w}^\top \tanh\left(\mathrm{W}_a \mathrm{h}_{t-1} + \mathrm{U}_a \mathrm{v}_i + \mathrm{b}_a\right)$$

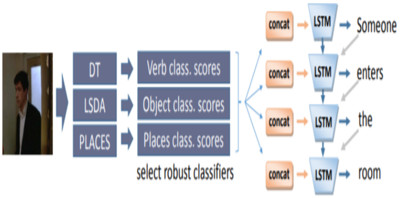
$$\alpha_i^{(t)} = \exp\left\{e_i^{(t)}\right\} / \sum_{j=1}^{n} \exp\left\{e_j^{(t)}\right\}$$

$$\varphi_t(V) = \sum_{i=1}^{n} \alpha_i^{(t)} \mathrm{v}_i$$

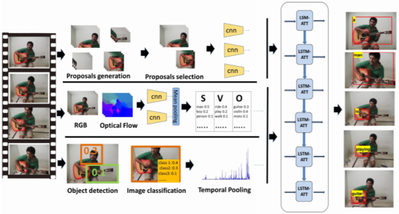
$$\mathrm{LSTM}(\varphi_t(V), h_{t-1}) = p(w_t)$$

Sequence to sequence models (cont.)

Using features from other auxiliary tasks:



(a) The Long-Short Story of Movie Description



(b) Spatio-Temporal Attention Models for Grounded Video Captioning, ACCV 2016

Sequence to sequence models (cont.)

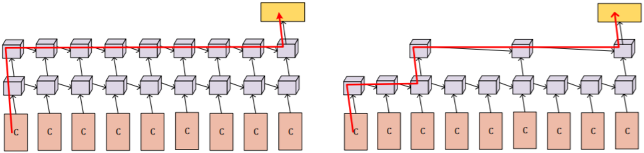


Figure 4: Hierarchical Recurrent Neural Encoder for Video Representation with Application to Captioning, 2015

The input sequence $(x_1, x_2, ..., x_T)$ into several chunks $(x_1, x_2, ..., x_n)$, $(x_{1+s}, x_{2+s}, ..., x_{n+s}), ..., (x_{T-n+1}, x_{T-n+2}, ..., x_T)$, where s is stride and it denotes the number of temporal units two adjacent chunks are apart. After inputting these subsequences into the LSTM filter, we will get a sequence of feature vectors $h_1, h_2, .., h_{\frac{T}{n}}$.

Sequence to sequence models (cont.)

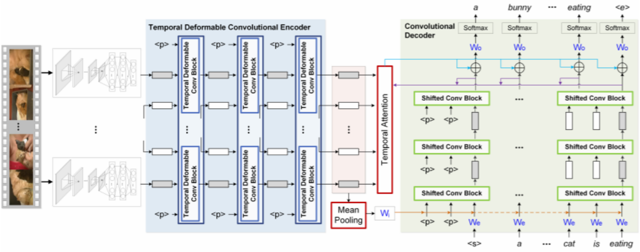


Figure 5: Temporal Deformable Convolutional Encoder-Decoder Networks for Video Captioning, AAAI 2019

Consider the output of the $l-1$ temporal convolutional block to be $\mathrm{p}^{l-1} = (p_1^{l-1}, \ldots, p_{N_v}^{l-1})$, where N_v is the number of frames in the video. Each output intermediate state p_i^l is achieved by feeding the subsequence $X = (p_{i+r_1}^{l-1}, p_{i+r_2}^{l-1}, \ldots, p_{i+r_k}^{l-1})$ into a temporal deformable convolution. Here, $r_n \in \{-k/2, ..., k/2\}$.

Temporal Deformable Convolutional Encoder:

$$\begin{aligned}
\Delta r^i &= W_f^l \left[p_{i+r_1}^{l-1}, p_{i+r_2}^{l-1}, \ldots, p_{i+r_k}^{l-1}\right] + b_f^l \\
o_i^l &= W_d^l \left[p_{i+r_1+\Delta r_1^i}^{l-1}, p_{i+r_2+\Delta r_2^i}^{l-1}, \ldots, p_{i+r_k+\Delta r_k^i}^{l-1}\right] + b_d^l \\
p_{i+r_n+\Delta r_n^i}^{l-1} &= \sum_s B\left(s, i + r_n + \Delta r_n^i\right) p_s^{l-1} \\
p_i^l &= g\left(o_i^l\right) + p_i^{l-1}
\end{aligned} \tag{4}$$

B is a function defined by $B(a, b) = \max(0, 1 - |a - b|)$ and $g(A, B) = g\left(o_i^l\right) = A \otimes \sigma(B)$ is the gated linear unit (GLU) activation function (Note: o_i^l is twice the dimension).

Temporal Deformable Convolutional Encoder:

$$q_t^l = g\left(W_l^q \left[q_{t-k+1}^{l-1}, q_{t-k+2}^{l-1}, \ldots, q_\iota^{l-1}\right] + b_l^q\right) + q_\iota^{l-1} \tag{5}$$

Sequence to sequence models (cont.)

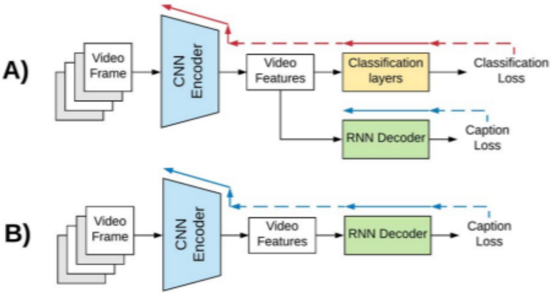


Figure 6: End-to-End Video Captioning, ICCVw 2019

- In all previous papers, CNNs are pretrained on object and/or action recognition tasks and used to encode video-level features.
- The decoder is then optimised on such static features to generate the video's description.
- This is a sub-optimal disjoint setup!

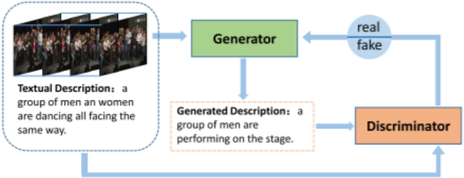
Two step training process:

- In Stage 1, the weights of the pre-trained encoder are frozen to train the decoder. Decoder is trained with respect to pre-computer encoder features.
- In Stage 2, The whole network is trained end-to-end while freezing the batch normalisation layer.

Decoder: Let E be the work embedding function, y_t be a word of the original caption, and $\varphi_t(V)$ be the hidden state of the LSTM attended with the visual features. The input to the LSTM is given by $z_t = [\varphi_t(V), E[y_{t-1}]]$

$$\begin{aligned} u_t &= W_u [\varphi_t(V), h_t] + E[y_{t-1}] + b_u \\ p_t &= \operatorname{softmax}(W_p \tanh(u_t) + b_p) \end{aligned} \tag{6}$$

Video Captioning by Adversarial LSTM



$$\mathcal{L}_D(\boldsymbol{Y}, D(\boldsymbol{S})) = -\frac{1}{m}\sum_{i=1}^{m}[(Y_i)\log(D(S_i)) + (1-(Y_i)(\log(1-D(\boldsymbol{S}_i)))]$$

$$\begin{aligned} \text{minimizing}: \ \mathcal{L}(\boldsymbol{S} \mid \boldsymbol{V}) = \ & \mathbf{E}_{s\sim P(s), v\sim P(v)}[\log P(\boldsymbol{S} \mid \boldsymbol{V})] + \\ & \mathbf{E}_{s\sim P(s)}[\log(1-D(G(\boldsymbol{S})))] \end{aligned} \tag{7}$$

Adversarial methods (cont.)

Caveat: The LSTM outputs $p(w_t | V, w_1, ..., w_{t-1})$, from which a word w_t is sampled. Hence, Words should be in an one-hot format which not only are high dimensionality but also are discrete. Makes it difficult for gradients to propoagate!

One solution is to use a soft-`argmax` function instead of the conventional `argmax`

$$w_{t-1} = \varepsilon_{w_e} \left(\text{softmax} \left\langle V h_{t-1} \odot L \right\rangle, W_e\right) \tag{8}$$

Another thing to notice! In the paper, the discriminator is implemented in the form of a convolutional network. Word embeddings of a sentence of length T are concatenated, and are represented as a matrix $X_d \in \mathrm{R}^{C \times T} = (x_{d_1}, ..., x_{d_T})$.

Why not use LSTM model for the discriminator?

Adversarial methods (cont.)

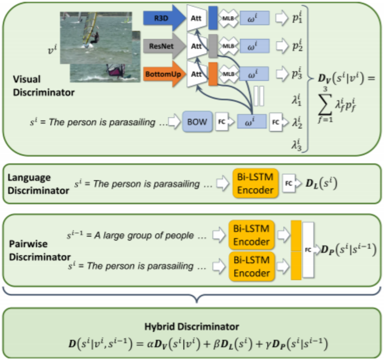


Figure 7: Adversarial Inference for Multi-Sentence Video Description, CVPR 2019

Adversarial methods (cont.)

1. **Visual Discriminator** Visual relevance

$$p_f^i = \sigma\left(\tanh\left(U^T \hat{v}_f^i\right) \odot \tanh\left(V^T \omega^i\right)\right)$$
$$\lambda_f^i = \frac{e^{a_f^T \omega^i}}{\sum_j e^{a_j^T \omega^i}}$$
$$D_V\left(s^i \mid v^i\right) = \sum_f \lambda_f^i p_f^i \tag{9}$$

2. **Language /Pairwise Discriminator** To promote fluency and grammatical correctness. Negative pairs are created by shuffling random words/sentences and repeating some phrases/sentences.

$$h_t^1, h_t^2 = \text{Bi-LSTM}(S) \tag{10}$$
$$D_L\left(s^i\right) = \sigma\left(W_L[h_t^1, h_t^2] + b_L\right) \tag{11}$$

Reinforcement Learning based methods

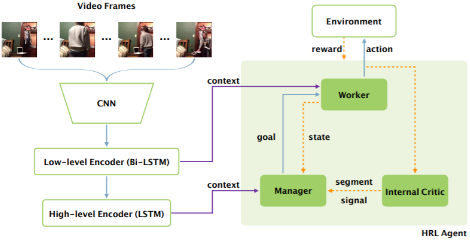


Figure 8: Video Captioning via Hierarchical Reinforcement Learning, CVPR'18

- **Worker:** Generates a word for each time step by following the goal proposed by the manager
- **Manager:** Operates at a lower temporal resolution and emits a goal when needed for the worker to accomplish
- **Internal Critic:** Determines if the worker has accomplished the goal

Reinforcement Learning based methods (cont.)

Manager and Worker: Quantities with sub/super-script W/M belong to the Worker/Manager. All the hidden and cell states are obtained from respective LSTMs. g_t denotes the latent continuous goal vector.

$$\begin{aligned} h_t^M &= S^M \left(h_{t-1}^M, \left[c_t^M, h_{t-1}^W\right]\right) \\ g_t &= u_M \left(h_t^M\right) \\ h_t^W &= S^W \left(h_{t-1}^W, \left[c_t^W, g_t, a_{t-1}\right]\right) \\ x_t &= u_W \left(h_t^W\right) \\ \pi_t &= \text{Soft Max}\left(x_t\right) \end{aligned} \tag{12}$$

Internal Critic: The critic is pre-trained to maximize the likelihood of $\sum_t \log p\left(z_t^* \mid a_1, \cdots, a_{t-1}\right)$ given ground truth signal. Once it is trained, it predicts the probability that actions are in accordance with g_t

$$\begin{aligned} h_t^I &= RNN\left(h_{t-1}^I, a_t\right) \\ p\left(z_t\right) &= \text{sigmoid}\left(W_z h_t^I + b_z\right) \end{aligned} \tag{13}$$

Reinforcement Learning based methods (cont.)

Worker π_{θ_w} (stochastic policy) : Using the REINFORCE Algorithm, we get $\nabla_{\theta_w} L(\theta_w) \approx -\left(R(a_t) - b_t^w\right) \nabla_{\theta_w} \log \pi_{\theta_w}(a_t)$. Here, $b_t = f(h_t^W)$ is the baseline that reduces the variance without changing the expected gradient.

Manager μ_{θ_m} (deterministic policy) : Let $e_{t,c}$ and $R(e_{t,c})$ denote the expected action of length c performed by following goal g_t and reward accumulated till time t respectively.

$$L(\theta_m) = -\mathbb{E}_{g_t}\left[R(e_t)\,\pi\left(e_{t,c}; s_t, g_t = \mu_{\theta_m}(s_t)\right)\right]$$

$$\nabla_{\theta_m} L(\theta_m) = -\mathbb{E}_{g_t}\left[R(e_{t,c})\,\nabla_{g_t}\pi(e_{t,c}; s_t, g_t)\,\nabla_{\theta_m}\mu_{\theta_m}(s_t)\right]$$

$$\nabla_{\theta_m} L(\theta_m) = -R(e_{t,c})\,\nabla_{g_t}\log\pi(e_{t,c})\,\nabla_{\theta_m}\mu_{\theta_m}(s_t)$$

$$\nabla_{\theta_m} L(\theta_m) = -R(e_{t,c})\left[\sum_{i=t}^{t+c-1}\nabla_{g_t}\log\pi(a_i)\right]\nabla_{\theta_m}\mu_{\theta_m}(s_t) \quad (14)$$

Reinforcement Learning based methods (cont.)

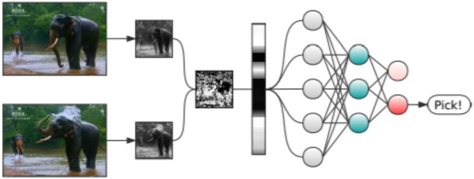


Figure 9: Less Is More: Picking Informative Frames for Video Captioning, 2018

Informative frame picking: Selecting a subset of frames from the video such that they convey relevant visual and temporally consistent information.

Reinforcement Learning based methods (cont.)

Let the last picked frame be $\tilde{g}$ and the frame in consideration at time t be g_t.

$$
\begin{aligned}
d_t &= \tilde{g} - g_t \\
\mathrm{s}_t &= W_2 \cdot (\max(W_1 \cdot \mathrm{vec}(\mathrm{d}_t) + \mathrm{b}_1, 0)) + \mathrm{b}_2 \\
p_\theta(a_t \mid \mathrm{z}_t, \tilde{\mathrm{g}}) &\sim \mathrm{softmax}(\mathrm{s}_t)
\end{aligned}
\tag{15}
$$

Rewards:

- Language rewards: Accuracy of generated sentence with respect to predicted sentence $r_l(c_i, S_i) = \mathrm{CIDEr}(c_i, S_i)$
- Visual Diversity reward: Variance of selected frames $r_v(v_i) = \sum_{j=1}^{D} \sqrt{\frac{1}{N_p} \sum_{i=1}^{N_p} \left(\mathrm{x}_i^{(j)} - \mu^{(j)}\right)^2}$

Reinforcement Learning based methods (cont.)

Therefore, the final reward is given by

$$r\left(v_i\right)=\begin{cases}\lambda_l r_l\left(v_i, S_i\right)+\lambda_v r_v\left(v_i\right) & \text{if} \quad N_{\min} \leq N_p \leq N_{\max} \\ R^{-} & \text{otherwise}\end{cases} \tag{16}$$

Training:

- Policy: The encoder-decoder sentence generator is trained on the cross entropy of generated sentences and ground truth captions.

$$L_{\mathrm{X}}(\omega)=-\sum_{t=1}^{m} \log \left(p_{\omega}\left(\mathrm{y}_t \mid \mathrm{y}_{t-1}, \mathrm{y}_{t-2}, \ldots \mathrm{y}_1, \mathrm{v}\right)\right) \tag{17}$$

- PickNet: Using the REINFORCE Algorithm,

$$\begin{aligned} L_R(\theta) &= -\mathbb{E}_{\mathrm{a}^s \sim p_\theta}\left[r\left(\mathrm{a}^s\right)\right] \\ \nabla_\theta L_R(\theta) &= -\mathbb{E}_{\mathrm{a}^s \sim p_\theta}\left[r\left(\mathrm{a}^s\right) \nabla_\theta \log p_\theta\left(\mathrm{a}^s\right)\right] \end{aligned} \tag{18}$$

Semi-supervised learning based methods

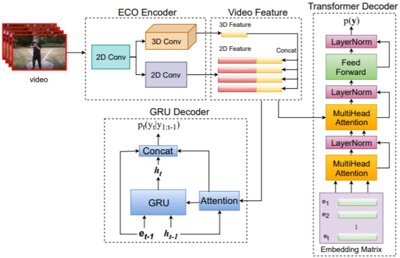


Figure 10: Semi-Supervised Learning for Video Captioning, ACL 2020

- For labeled data, models are trained with the traditional cross-entropy loss.
- For unlabeled data, a self-critical policy gradient method is utilised.

Semi-supervised learning based methods (cont.)

Let u and u^* denote the output class distribution of the original video and augmented video. Considering an on-policy method, let $y_t = \text{Sample}(p_\theta(\hat{y}_{1:t-1}, \boldsymbol{u}_b))$

$$\hat{r} = \sum_{t=1}^{T} \hat{d}_t = \sum_{t=1}^{T} D_{KL}\left(p_\theta\left(\hat{y}_t \mid \hat{y}_{1:t-1}, \boldsymbol{u}\right) \| p_\theta\left(\hat{y}_t \mid \hat{y}_{1:t-1}, \boldsymbol{u}^*\right)\right) \tag{19}$$

Baselines for the self-critical training sequences are found using greedy policy, where $\tilde{y}_t = \underset{\tilde{y}_t}{\arg\max} p_\theta\left(\tilde{y}_{1:t-1}, \boldsymbol{u}\right)$

$$\tilde{r}_t = D_{KL}\left(p_\theta\left(\tilde{y}_t \mid \tilde{y}_{1:t-1}, \boldsymbol{u}\right) \| p_\theta\left(\tilde{y}_t \mid \tilde{y}_{1:t-1}, \boldsymbol{u}^*\right)\right) \tag{20}$$

The policy gradient update step is given by

$$\nabla_\theta L_u(\theta) = -\sum_{t=1}^{T} (\hat{r} - \tilde{r}) \nabla_\theta \log p_\theta\left(\hat{y}_t \mid \hat{y}_{1:t-1}, \boldsymbol{u}\right) \tag{21}$$

Semi-supervised learning based methods (cont.)

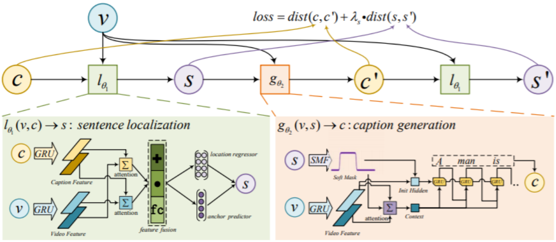


Figure 11: Weakly Supervised Dense Event Captioning in Videos, NIPS 2018

During training, only a few sentences are available for each video. It is assumed that each caption describes one temporal segment, and each temporal segment has one caption. The training is carried out in a cycle of dual problems.

- Sentence localiztaion
- Caption generation

Semi-supervised learning based methods (cont.)

Video $V = (v_1, v_2, ..., v_n)$.

Temporally continuous segments of $V \rightarrow$ temporally coordinates $\{S = (m_i, v_i)\}_i^N$ where m_i and v_i denote center and width respectively.

Let C_i be the caption for each temporal segment.

The dual tasks are defined as:

- **Sentence localization:** To localize segment S_i corresponded to the given caption C_i by learning the mapping $l_{\theta_1} : (V, C_i) \rightarrow S_i$
- **Event Captioning:** Inversely generate caption C_i for the given segment S_i by learning the function $g_{\theta_2} : (V, S_i) \rightarrow C_i$

Semi-supervised learning based methods (cont.)

When we nest the two functions together, we obtain:

$$C_i = g_{\theta_2}(V, l_{\theta_1}(V, C_i)) \tag{22}$$

$$S_i = l_{\theta_1}(V, g_{\theta_2}(V, S_i)) \tag{23}$$

The dual problems exist simultaneously once the correspondence between S_i and C_i is one-to-one. We train the parameters θ_1 and θ_2 to minimise the loss function given by:

$$\mathcal{L}_c = \operatorname{dist}\left(C_i, g_{\theta_2}\left(V, l_{\theta_1}\left(V, C_i\right)\right)\right) \tag{24}$$

Testing procedure:

- Cannot apply the cycle process as caption is unknown
- Perform caption generation on a bunch of randomly initialized segments and then map the resulting captions back to the segment space using l_{θ_1}

Semi-supervised learning based methods (cont.)

Fixed Point Iteration: An iteration is defined as:

$$S(t+1) = l_{\theta_1}(V, g_{\theta_2}(V, S(t))) \tag{25}$$

where $S(t)$ will converge to the fixed-point solution i.e. $S^\star = l_{\theta_1}(V, g_{\theta_2}(V, S^\star))$, if there exists a sufficiently small $\epsilon > 0$ satisfying $\|S(0) - S^\star\| < \epsilon$ and the function $l_{\theta_1}(V, g_{\theta_2}(V, S))$ is locally Lipschitz continuous around $S^\star$ with Lipschitz constant $L < 1$.

- Sample a batch of random candidate segments $\{S_i^{(r)}\}_i^{N_r}$ for the target video as initial guesses, and then perform the fixed point iteration to obtain S_i'.
- S_i' is then used to generate captions using the caption generator.
- With only one iteration, the method delivers promising results.

Semi-supervised learning based methods (cont.)

How to enforce that temporal segments of true data converge to fixed-point solutions by one-round iteration?

Recall: We have no supervision for temporal segments.

$$\mathcal{L}_s = \text{dist}\left(l_{\theta_1}\left(\boldsymbol{V}, \boldsymbol{C}_i\right), l_{\theta_1}\left(\boldsymbol{V}, g_{\theta_2}\left(\boldsymbol{V}, \varepsilon_i + l_{\theta_1}\left(\boldsymbol{V}, \boldsymbol{C}_i\right)\right)\right)\right), \tag{26}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma)$ is a Gaussian noise.
The total loss function is:

$$\mathcal{L} = \mathcal{L}_c + \lambda_s \mathcal{L}_s \tag{27}$$

Zero-shot video captioning

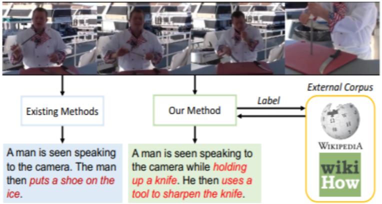


Figure 12: Learning to Compose Topic-Aware Mixture of Experts for Zero-Shot Video Captioning, AAAI 2019

Problems: "Sharpening Knives" is a novel activity unseen in training, and existing methods fail to generate a pertinent caption because it is aware of neither the action "sharpening" nor the object "knife".

Goal: A model is required to accurately describe novel activities in videos without any explicit paired training data.

Zero-shot video captioning (cont.)

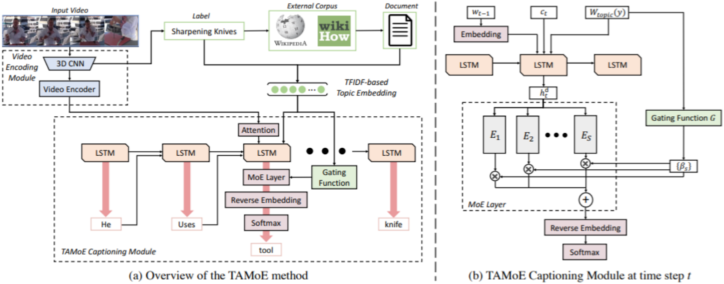


Figure 13: Framework: Learning to Compose Topic-Aware Mixture of Experts for Zero-Shot Video Captioning, AAAI 2019

Zero-shot video captioning (cont.)

Video Encoding Module: Given an input video $v_1, v_2, .., v_n$, a pretrained 3D convolutional neural networks is employed to extract the segment-level features $\{f_j\}$, which are further sent to a bidirectional LSTM.

Term Frequency-Inverse Document Frequency (TFIDF) -based Topic Embedding: The segment level features are assigned a topic and further topic-related documents from various data sources (Wikihow etc) are fetched.

Given an activity label y and related documents D_y, topic-specific knowledge representations is given by TF-IDF.

$$g_k(y) = \frac{z_k(y)}{\sum_{x_l \in D_y} z_l(y)} \log \left(\frac{|Y|}{\sum_{y' \in Y} \min\left(1, z_k\left(y'\right)\right)} \right) \tag{28}$$

Weights $g_k(y)$ demonstrate the relevance of each unigram x_k to the topic-related documents D_y, where $z_k(y)$ is the number of times the unigram x_k occurs in the documents

Zero-shot video captioning (cont.)

The TF-IDF embeddings and topic-aware embeddings are given by are given by $W_{\text{topic}}(y) = \sum_{x_k \in D_y} g_k(y) W_{\text{fasttext}}\ (x_k)$.

Attention Based LSTM: Given a contect vector c_t, calculated from the weighted sum of encoded video features, the hidden state of LSTM is given by $h_t^d = LSTM\left(\left[w_{t-1}, c_t, W_{\text{topic}}\ (y)\right], h_{t-1}^d\right)$

Mixture-of-Expert Layer and Topic-Aware Gating Function: All throughout the framework, it is assumed that the basics of captioning are shared among topics. A mixture of S experts is considered, which consist of mapping function from the latent representation h_t^d to the vocabulary.

$$o_t = \sum_{s=1}^{S} \beta_s E_s\left(h_t^d\right) \tag{29}$$

$$\beta_s = \frac{\exp\left(G\left(W_{topic}(y)\right)_s / \tau\right)}{\sum_{i=1}^{S} \exp\left(G\left(W_{\text{topic}}\ (y)\right)_i / \tau\right)} \tag{30}$$

Transformer-based methods

Transformer: A model that relies entirely on an attention mechanism to draw global dependencies between input and output, thus forfeiting any recurrence.

Basic components of a transformer network:
Scaled Dot-product Attention: Given a matrix of queries $Q \in \mathrm{R}^{d_k}$, keys $K \in \mathrm{R}^{d_k}$ and values $V \in \mathrm{R}^{d_v}$, the matrix of outputs is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{31}$$

Multi-head Attention: This function consists of h different heads, where each head performs the scaled dot-product attention.

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{ Concat } (\text{ head }_1, \ldots, \text{ head }_\mathrm{h}) W^O \\ \text{where head }_\mathrm{i} &= \text{ Attention } \left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{32}$$

Positional encoding:

- To add position related information to the input embedding, which was forfeited due to the lack of convolution or recurrent function.
- Implemented in the form of sine and cosine function of their relative positions with respect to dimension i (Eq. 33).

$$
\begin{aligned}
PE_{(pos,2i)} &= \sin\left(pos/10000^{2i/d_{\text{model}}}\right) \\
PE_{(pos,2i+1)} &= \cos\left(pos/10000^{2i/d_{\text{model}}}\right)
\end{aligned}
\tag{33}
$$

Leftward information flow: Without any recurrence, information is free to flow in the left (i.e. from w_{t+1} to w_t). This implies that autoregressive property is broken!
Solution? Masked Multi-Head attention: Set value $-\infty$ for all illegal connections.

Transformer-based methods (cont.)

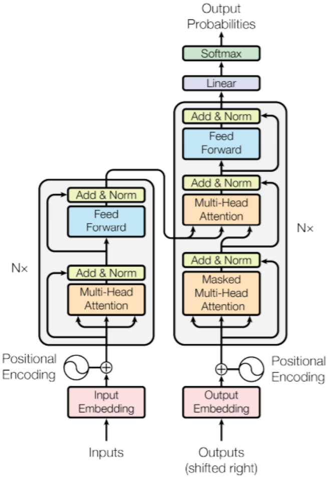


Figure 14: Attention is all you Need, NIPS 2017

- For "encoder-decoder" attention layers:
  - Q: previous decoder layer
  - K, V: output of the encoder
- For encoder self-attention: Q, K, V come from output of the previous layer in the encoder
- For decoder self-attention, each position is allowed to attend to all positions in the decoder up to and including that position.

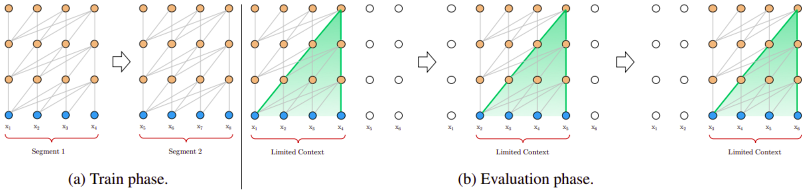


Figure 15: Character-Level Language Modeling with Deeper Self-Attention faces **context fragmentation**

- Model cannot capture any longer-term dependency beyond the predefined context length
- How to train a Transformer to effectively encode an arbitrarily long context into a fixed size representation?
- How can we improve evaluation procedure?

Transformer - XL

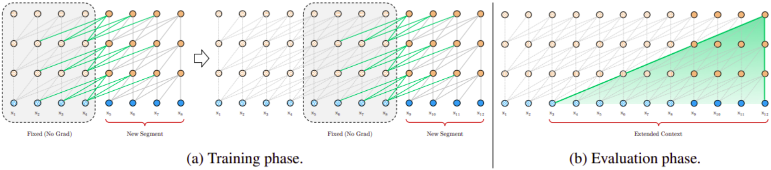


Figure 16: Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

$$
\begin{aligned}
(\mathrm{h}_{\tau+1}^{n-1} &= \left[\mathrm{SG}\left(\mathrm{h}_{\tau}^{n-1}\right) \circ \mathrm{h}_{\tau+1}^{n-1}\right]) \\
(\mathrm{q}_{\tau+1}^{n}, \mathrm{k}_{\tau+1}^{n}, \mathrm{v}_{\tau+1}^{n} &= \mathrm{h}_{\tau+1}^{n-1}\mathrm{W}_{q}^{\top}, \mathrm{h}_{\tau+1}^{n-1}\mathrm{W}_{k}^{\top}, \mathrm{h}_{\tau+1}^{n-1}\mathrm{W}_{v}^{\top}) \\
\mathrm{h}_{\tau+1}^{n} &= Transformer - Layer((\mathrm{q}_{\tau+1}^{n}, \mathrm{k}_{\tau+1}^{n}, \mathrm{v}_{\tau+1}^{n}))
\end{aligned} \tag{34}
$$

Transformer-based methods (cont.)

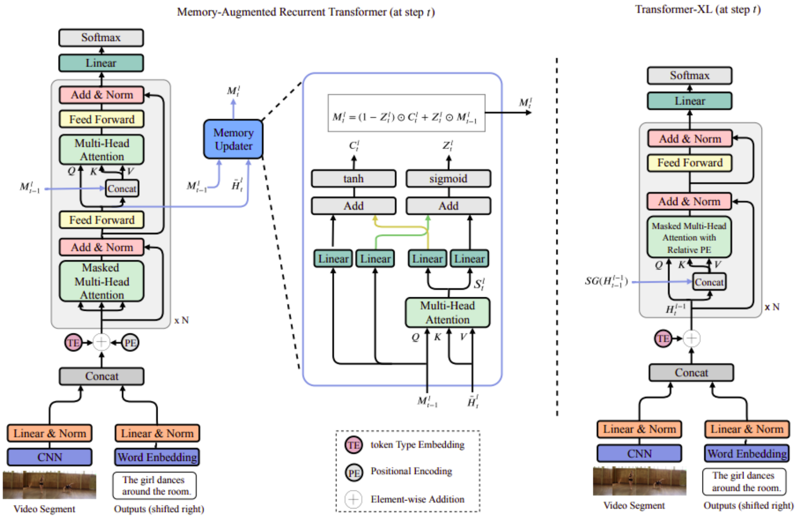


Figure 17: Using Transformers in video captioning

Transformer-based methods (cont.)

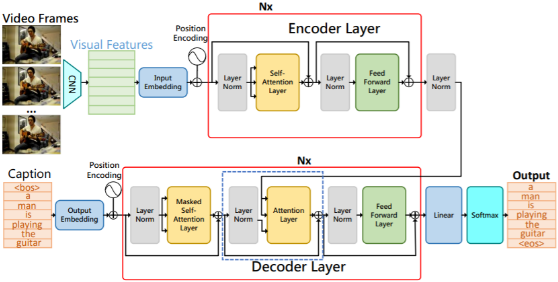


Figure 18: TVT: Two-View Transformer Network for Video Captioning, ACML'18. The visual feature is a attended function of features extracted from RGB images by 2D CNN and those obtained by a 3D CNN for motion recognition.

Transformer-based methods (cont.)

Two views?: Because two transformers have been used for visual features and motion features.
Let E_f be the matrix of frame representations obtained by 2D CNN on each frame, and E_m be obtained by a 3-D CNN on consecutive frames. The corresponding sentence is denoted by D_s

$$\begin{aligned} Q_f &= \text{ LayerNorm } (D_s) W_f^Q \\ K_f &= E_f W_f^K \\ V_f &= E_f W_f^V \end{aligned} \tag{35}$$

$$\begin{aligned} C_f &= \text{MultiHead}(Q_f, K_f, V_f) \\ C_m &= \text{MultiHead}(Q_m, K_m, V_m) \end{aligned} \tag{36}$$

Further, columns from the two features C_f and C_m are concatenated along with D_s, and passed through another transformer as K, V.

Transformer-based methods (cont.)

BERT in video captioning

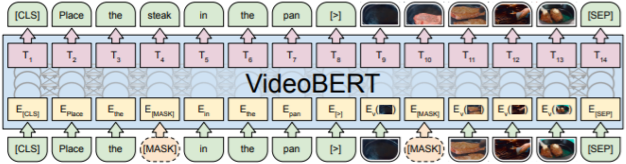
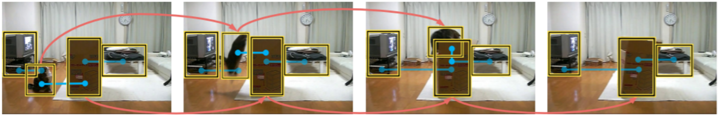


Figure 19: VideoBERT: A Joint Model for Video and Language Representation Learning

- BERT proposes to learn language representations by using a "masked language model" training objective $L(\theta) = E_{x \sim D} \sum_{l=1}^{L} \log p\left(x_l \mid x_{\backslash l}; \theta\right)$
- For video captioning, the model predicts the masked video features and words.

Graph-based methods



"A cat jumps into a box."

Figure 20: Problem Statement: Spatio-Temporal Graph for Video Captioning with Knowledge Distillation, CVPR 2020

- Objects in a video interact with each other spatially and transform their location, pose, etc temporally. To capture these two correlations, the graph has been split into two components: spatial and temporal.

Graph-based methods (cont.)

Visual Features: Scene features $\{f_1, f_2, \ldots, f_T\}$ obtained from 2D CNN, 3D CNN features $\{v_1, v_2, \ldots, v_L\}$, and Faster R-CNN object features $F_o = \left\{o_1^1, o_1^2, \ldots, o_t^j, \ldots, o_T^{N_T}\right\}$.

Adjacency matrices for graph:

$$G_{tij}^{\text{space}} = \frac{\exp \sigma_{tij}}{\sum_{j=1}^{N_t} \exp \sigma_{tij}} \tag{37}$$

$$G_{tij}^{\text{time}} = \frac{\exp \cos\left(o_t^i, o_{t+1}^j\right)}{\sum_{j=1}^{N_{t+1}} \exp \cos\left(o_t^i, o_{t+1}^j\right)} \tag{38}$$

$$G^{st} = \begin{bmatrix} G_1^{\text{space}} & G_1^{\text{time}} & 0 & \ldots & 0 \\ 0 & G_2^{\text{space}} & G_2^{\text{time}} & \ldots & 0 \\ 0 & 0 & G_3^{\text{space}} & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & G_T^{\text{space}} \end{bmatrix} \tag{39}$$

Graph-based methods (cont.)

Graph convolution network: For a GCN with N_l layers stacked on top of each other, the graph is updated via

$$H^{(l+1)} = \mathrm{ReLU}\left(H^{(l)} + \Lambda^{-\frac{1}{2}} G^{st} \Lambda^{-\frac{1}{2}} H^{(l)} W^{(l)}\right) \tag{40}$$

$H^{(0)}$ is the stack of object features while the final object representing features obtained from the GCN is denoted by $F\prime_o$.

Incorporating both scene and object features: Two separate language decoders are used for both scene and object features, trained on the cross entropy loss denoted by $L_{\text{s-lang}}$ and $L_{\text{o-lang}}$ respectively.
The distillation loss is obtained by minimising cross entropy between the probability distributions produced by the two decoders.

$$\begin{aligned} L_{\text{distill}} &= -\sum_{x \in V} P_s(x) \log\left(\frac{P_o(x)}{P_s(x)}\right) \\ L &= L_{o\_lang} + \lambda_{sl} L_{s\_lang} + \lambda_d L_{\text{distill}}\ , \end{aligned} \tag{41}$$

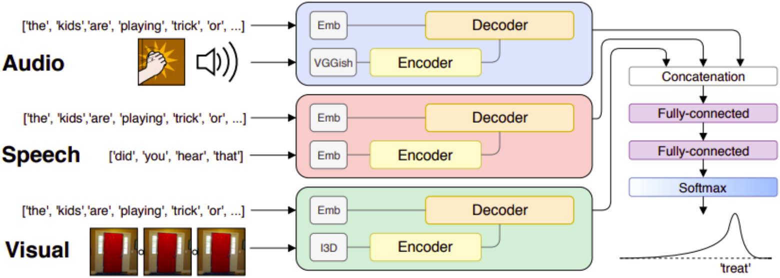


Figure 21: Multi-modal Dense Video Captioning (MDVC) framework, CVPRw 2020

Audio-visual captioning (cont.)

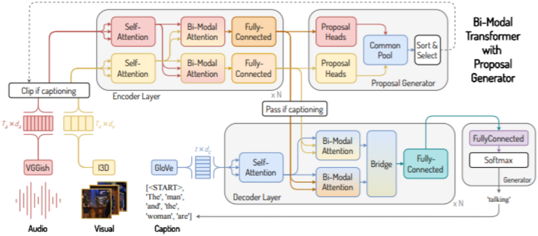


Figure 22: A Better Use of Audio-Visual Cues: Dense Video Captioning with Bi-modal Transformer, BMVC 2020

- Let A and V denote audio features and video features respectively. The ground truth captions are denoted by C.
- The encoder and decoder consist of three different layers: self-attention, bi-modal attention, and position-wise fully-connected layers.

Other training paradigms:

Some Novel Research in Video Captioning

- Active Learning for Video Description With Cluster-Regularized Ensemble Ranking, ACCV 2020
  - Study of different active learning methods for sequence-to-sequence video captioning
- Open Book Video Captioning, CVPR 2021
  - By using actions and objects in the video, the model generates textual descriptions that are not limited to the video or vocabulary

Possible research direction:

- **Cross-domain captioning:** This has been extensively studied for image captioning, but its extension to video captioning seems non-trivial.
- **Crafting adversarial examples:** Some work has been done for image captioning and video recognition, but to the best of my knowledge, none for captioning.