PicSOM and EURECOM Experiments in TRECVID 2019

Workshop notebook paper

Héctor Laria Mantecón[+], Jorma Laaksonen[+], Danny Francis[*], Benoit Huet[*]

[+]Department of Computer Science
Aalto University School of Science
P.O.Box 15400, FI-00076 Aalto, Finland
firstname.lastname@aalto.fi

[*]Department of Data Science
EURECOM, Campus SophiaTech
450 route des Chappes
06410 Biot, France
firstname.lastname@eurecom.fr

Abstract

This year, the PicSOM and EURECOM teams participated only in the Video to Text Description (VTT), Description Generation subtask. Both groups submitted one or two runs labeled as a "MeMAD" submission, stemming from a joint EU H2020 research project with that name. In total, the PicSOM team submitted four runs and EURECOM one run. The goal of the PicSOM submissions was to study the effect of using either image or video features or both. The goal of the EURECOM submission was to experiment with the use of Curriculum Learning in video captioning. The submitted five runs are as follows:

- PICSOM.1-MEMAD.PRIMARY: uses ResNet and I3D features for initialising the LSTM generator, and is trained on MS COCO + TGIF using self-critical loss,
- PICSOM.2-MEMAD: uses I3D features as initialisation, and is trained on TGIF using self-critical loss,
- PICSOM.3: uses ResNet features as initialisation, and is trained on MS COCO + TGIF using self-critical loss,
- PICSOM.4: is the same as PICSOM.1-MEMAD.PRIMARY except that the loss function used is cross-entropy,
- EURECOM.MEMAD.PRIMARY: uses I3D features to initialize a GRU generator, and is trained on TGIF + MSR-VTT + MSVD with cross-entropy and curriculum learning.

The runs aim at comparing the use of cross-entropy and self-critical training loss functions and to showing whether one can successfully use both still image and video features even when the COCO dataset does not allow the extractions of I3D video features. Based on the results of the runs, it seems that using both video and still image features, one can obtain better captioning results than with either one of the single modalities alone. The Curriculum Learning process proposed does not seem to be beneficial.

I. INTRODUCTION

In this notebook paper, we describe the PicSOM and EURECOM teams' experiments for the TRECVID 2019 evaluation [1]. We participated only in the Video to Text Description (VTT) subtask Description Generation. Our approaches are variations of the "Show and tell" model [2], augmented with a richer set of contextual features [3], self-critical training [4] and Curriculum Learning [5]. Both teams' systems have been used to produce the runs presented in this paper. The captioning models are described in more detail in Section II and their used training loss functions in Section III. Then, we describe the features in Section IV and the datasets used for training in Section V. Our experiments, submitted runs and results are discussed in Section VI and conclusions are drawn in Section VII.

II. NEURAL CAPTIONING MODELS

In our experiments we have used two different Python-based software projects for caption generation. The PicSOM team's *DeepCaption*, uses the PyTorch library, whereas EURECOM's *CLCaption* approach is based on using the TensorFlow library.

A. DeepCaption

The PicSOM team's LSTM [6] model has been implemented in PyTorch and is available as open source.[1] The features are translated to the hidden size of the LSTM by using a fully connected layer. We apply dropout and batch normalization [7] at this layer. As the loss function, we similarly use cross entropy, in addition to Reinforcement Learning with self-critical loss function [4] in order to fine-tune a well-performing model. The fine-tuning is implemented either by switching to the self-critical loss in training time or by specifying a pre-trained model to load and fine-tune.

B. CLCaption

For EURECOM's first participation in the TRECVID VTT captioning task, we submitted a run based on a model trained

[1]https://github.com/aalto-cbir/DeepCaption

by Curriculum Learning [8]. We implemented our model using the TensorFlow framework for Python [9].

The idea behind Curriculum Learning is to present data during training in an ascending order of difficulty: first epochs are based on easy samples, and after each epoch, more difficult samples are added to training data. We computed a difficulty score for a given sample composed of a video and a corresponding caption as follows: the caption is translated into a list of indices (the bigger the indices the less frequent the corresponding word), the score of the sample is then the maximum index of its caption. Once samples have been scored, we trained the model starting with an easy subset of the training set, and adding after each epoch more complex samples.

Video features have been extracted with an I3D neural network [10], input to a fully connected layer and then processed by a GRU [11] to generate captions. Cross-entropy loss has been used for training the model.

III. Training loss functions

In order to train the architecture so that its output distribution approximates the target distribution at each decoding step t, several optimisation objectives are used. Recent progress on sequence training enables new optimisation paradigms, which are applied and compared in this work.

A. Cross-entropy

Traditionally, the teacher forcing algorithm [5] is the most common method to maximise the log-likelihood of a model output X to match the ground truth $y = \{y_1, y_2, \cdots, y_T\}$. It minimises the cross-entropy objective

$$\mathcal{L}_{CE} = -\sum_{t=1}^{T} \log p_\theta \left(y_t \mid y_{t-1}, \mathbf{h}_{t-1}, X\right), \qquad (1)$$

where $\mathbf{h}_{t-1}$ is the hidden state of the RNN from the previous step and p_θ the probability of an output parametrized by θ. In the inference time, the output can be produced simply by greedy sampling of the sequence being generated.

B. Self-critical

Lately, Reinforcement Learning ideas have been used to optimise a captioning system based on recurrent neural network language models. Such a system can be seen as an agent taking actions according to a policy π_θ and outputting a word $\hat{y}_t$ as an action.

One proposed approach is the self-critical algorithm [4], where the output at inference time of the model $\hat{y}^g_{i,t}$ is used, normally applying greedy search. The sequences are scored using a reward function r. Thanks to the properties of this optimisation, NLP metrics can be used as reward to affect the actual loss. In our case, CIDErD [12] is used. The final objective reads

$$\begin{aligned}\mathcal{L}_\theta = \tfrac{1}{N} \textstyle\sum_{i=1}^{N} \sum_t \log \pi_\theta \left(\hat{y}_{i,t} \mid \hat{y}_{i,t-1}, \mathbf{s}_{i,t}, \mathbf{h}_{i,t-1}\right) \\ \cdot \left(r(\hat{y}_{i,1}, \cdots, \hat{y}_{i,T}) - r(\hat{y}^g_{i,1}, \cdots, \hat{y}^g_{i,T})\right)\end{aligned} . \qquad (2)$$

IV. Features

Table I summarizes the features used in our experiments and their dimensionalities.

TABLE I
SUMMARY OF THE FEATURES USED IN OUR EXPERIMENTS.

abbr.	feature	dim.	modality
rn	CNN ResNet	4096	image
fr	Faster R-CNN	80	image
i3d	I3D	2048	video

A. CNN

We are using pre-trained CNN features from ResNet 101 and 152. The 2048-dimensional features from the pool5 layer average to five crops from the original and horizontally flipped images. These features have then been concatenated together and are referred to as "rn" in Table I and later in this paper. When applied to a video object, we have used the middlemost frame of the video.

B. FasterRCNN

The existence of certain objects in the visual scene has an effect on sentence formation and influences the adjectives used in human sentences. To extract this information, we use an object detector, specifically the Faster Region-based Convolutional Neural Network (R-CNN) [13]. This network predicts the object locations as bounding boxes and object detection scores of the 80 object categories of Microsoft Common Objects in Common Context (MS-COCO) database.[2] In our current approach we, however, ignore the location information and encode the object detection scores on the image level. We obtain thus an 80-dimensional feature vector using the detection score for each category, and refer to it as "fr". When applied to a video object, we have used the middlemost frame of the video.

C. I3D

To encode video features, the PicSOM team adopted Inflated 3D Convolutional Network (I3D) [10]. It builds upon already competent image recognition models (2D) and inflates the filters and kernels to 3D, thus creating an additional temporal dimension. Concretely, the base network used is ImageNet-pretrained Inception-V1 [14] using two streams [15]. The videos were first resampled to 25 frames per second as in the original I3D paper and 128 frames were taken from the center. For DeepCaption, the extractor is applied convolutionally over the whole video and the output is average-pooled in order to produce a 2048-dimensional feature vector.

Regarding CLCaption, features have been extracted before the softmax layer, thus obtaining a 600-dimensional features vector. These features have then been input to the CLCaption model.

[2] http://cocodataset.org/

V. Training data

Table II gives a summary of the databases and the features we have extracted for them. In Tables II and III, we have shortened the dataset names with one letter abbreviations.

TABLE II
SUMMARY OF THE TRAINING DATASETS USED IN OUR EXPERIMENTS.

	dataset	items	captions	features
C	COCO	82,783 img	414,113	rn fr
M	MSR-VTT	6,513 vid	130,260	rn i3d
T	TGIF	125,713 vid	125,713	rn fr i3d
V	MSVD	1,969 vid	80,800	rn i3d

A. COCO

The *Microsoft Common Objects in COntext (MS COCO)* dataset [16] has 2,500,000 labeled instances in 328,000 images, consisting on 80 object categories. COCO is focused on non-iconic views (or non-canonical perspectives) of objects, contextual reasoning between objects, and precise 2D localization of objects.

B. MSR-VTT

The *MSR-Video to Text (MSR-VTT)* dataset [17] provides 10,000 web video clips with 41.2 hours and 200,000 clip-sentence pairs in total, covering a comprehensive list of 20 categories and a wide variety of video content. Each clip was annotated with about 20 natural sentences. Additionally, the audio channel is provided too.

C. TGIF

The *Tumblr GIF (TGIF)* dataset [18] contains 100,000 animated GIFs and 120,000 natural language sentences. This dataset aims to provide motion information involved between image sequences (or frames).

D. MSVD

The *Microsoft Research Video Description Corpus (MSVD)* [19] consists of 85,000 English video description sentences and more than 1,000 for a dozen more languages. It contains a set of 2,089 videos, showing a single, unambiguous action or event.

VI. Experiments and results

During the development stage, the PicSOM team ran a number of experiments to select the best combinations of features and training datasets. We evaluated our results using the previously released ground truth of TRECVID VTT 2018 test set. The four runs submitted are identified as "p-19-s1" to "p-19-s4" in Table III. The runs "p-18-s2" and "p-18-a3" we created using our best model in the last year's submissions and the best model we experimented with after the last year's workshop, respectively.

Runs identified as "p-19-s1" and "p-19-s4" use I3D video features extracted from the TGIF dataset. We used also the COCO dataset for training the models for those runs, but because we could not extract I3D video features from the still images of that dataset, we used the average value of the I3D feature vectors of the TGIF dataset for each COCO image.

Based on evaluation on the TRECVID VTT 2018 test set, we ended up using a 2-layer LSTM for DeepCaption with an embedding vector size of 512, and 1024 for the hidden state dimensionality in all PicSOM team's runs. Both in the input translation layer and in the LSTM we applied a dropout of 0.5. We used Adam optimiser [20] for the self-critical stage with a learning rate of 5×10^{-5} and no weight decay. Additionally, gradient clipping is performed when a range $[-0.1, 0.1]$ is exceeded. The models were pretrained using centered RMSprop [21] with a learning rate of 0.001 and weight decay (L2 penalty) of 10^{-6}.

EURECOM's CLCaption is based on a GRU with 1024-dimensional hidden states. The size of the input I3D vectors is 600. The fully-connected layer output is of dimension 1024. No dropout nor batch normalization were used. The training algorithm we used was RMSProp with a learning rate of 0.0001 and mini-batches of size 64. The CLCaption run is identified as "e-19-e1" in Table III where all experiments are briefly summarized and their results presented.

The four "setup" columns in Table III specify the submission type (I=image, V=video, B=both), the loss function (ce=cross-entropy, sc=self-critical), the features, and the datasets used in the RNN model training.

The features are concatenations of the following:

rn = CNN ResNet, see IV-A
fr = Faster R-CNN, see IV-B
i3d = I3D, see IV-C

The used datasets are combinations of the following:

C = COCO, see V-A
M = MSR-VTT, see V-B
T = TGIF, see V-C
V = MSVD, see V-D

Our results compared to those of the other submitted runs are visualized with bar charts for each automatic performance measure in Figures 1–5.

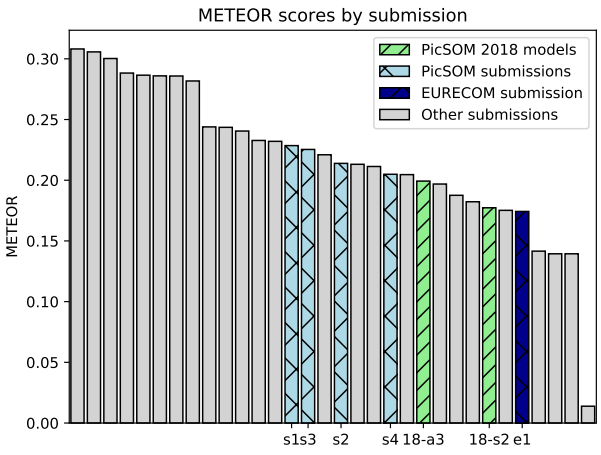

Fig. 1. METEOR results of our teams and others.

TABLE III
RESULTS OF OUR SUBMISSIONS (P-19-S1,. . . ,4, E-19-E1) AND SOME NOTEWORTHY EARLIER MODELS (P-18-S2, P-18-A3). THE P-* RUNS ARE BY THE PICSOM TEAM AND THE E-* RUN BY THE EURECOM TEAM.

	setup				2018				2019				
id	**t**	**loss**	**feat**	**data**	**METEOR**	**CIDEr**	**CIDErD**	**BLEU**	**METEOR**	**CIDEr**	**CIDErD**	**BLEU**	**STS**
p-18-s2	I	ce	rn+fr	C+M	0.1541	0.1657	0.0476	0.0091	0.1773	0.1858	0.0722	0.0207	–
p-18-a3	I	ce	rn	C+T	0.1776	0.1948	0.0700	0.0197	0.1993	0.2174	0.1004	0.0288	–
p-19-s1	B	sc	rn+i3d	C+T	**0.2055**	**0.3025**	**0.1157**	0.0294	**0.2285**	**0.3277**	**0.1615**	**0.0385**	0.4168
p-19-s2	V	sc	i3d	T	0.1958	0.2718	0.0949	**0.0348**	0.2139	0.2773	0.1245	*0.0379*	*0.4169*
p-19-s3	I	sc	rn	C+T	*0.2007*	*0.2777*	*0.1074*	*0.0301*	*0.2254*	*0.3130*	*0.1569*	0.0345	**0.4282**
p-19-s4	B	ce	rn+i3d	C+T	0.1850	0.2190	0.0822	0.0213	0.2049	0.2348	0.1147	0.0319	0.4057
e-19-e1	V	ce	i3d	M+T+V	–	–	–	–	0.1743	0.2340	0.0710	0.0068	0.4214

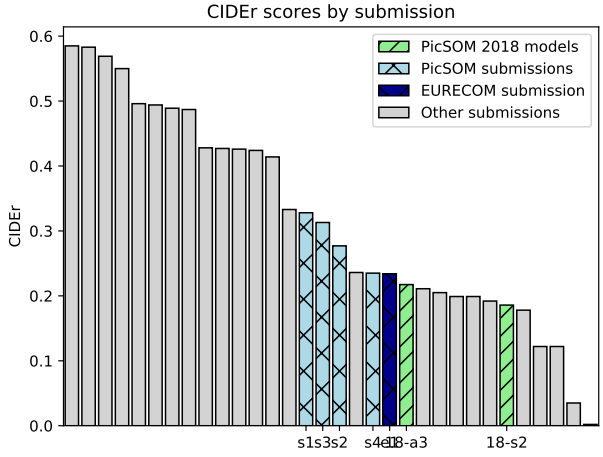

Fig. 2. CIDEr results of our teams and others.

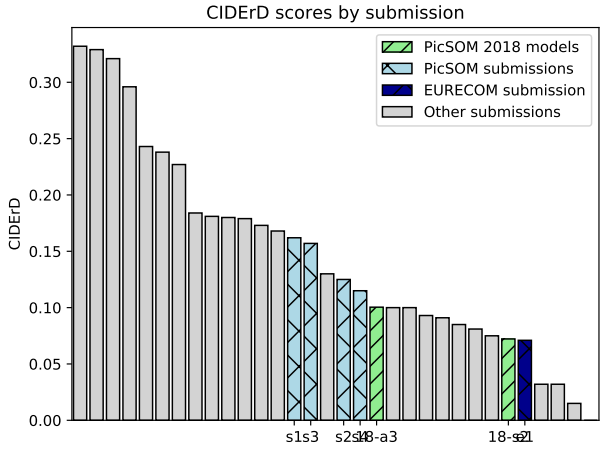

Fig. 3. CIDErD results of our teams and others.

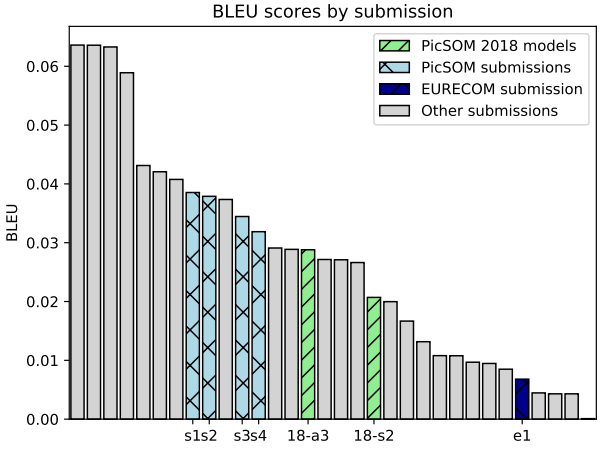

Fig. 4. BLEU results of our teams and others.

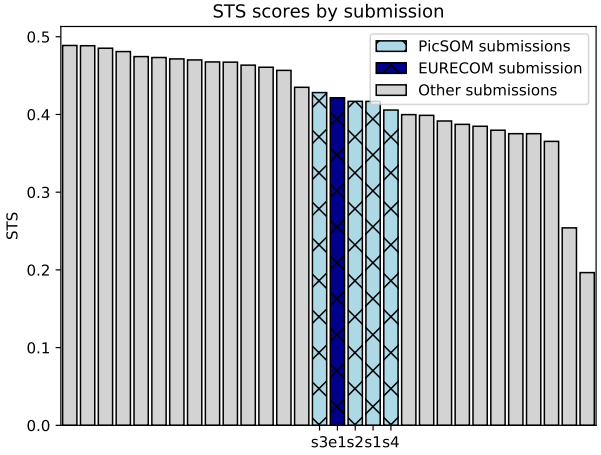

Fig. 5. STS results of our teams and others.

VII. CONCLUSIONS

There were two main research question in the PicSOM team's set of four submissions. First, we wanted to compare the implementations of cross-entropy and self-critical training loss functions in our DeepCaption code. The results with self-critical training were better in all measures, but this could of course be expected based on our and other teams' earlier experiments. Based on our observations, however, the use of this loss alone does not imply a straightforward jump in caption quality as much as the score increment suggests.

Second, we aimed to know whether we could successfully use both still image and video features even when the COCO dataset does not allow the extractions of I3D video features. The trick we applied was to use the average of the I3D video features extracted from the TGIF dataset for all images in the COCO dataset. For the COCO images the video features were thus non-informative, but still allowed us to use two datasets

and two different feature extraction schemes together. The results of this approach were encouraging as they were better than those with either dataset or either feature used alone.

Additionally, we could now compared the current performance of the PicSOM team's DeepCaption model to its performance in the last year's evaluation. We have clearly made substantial progress compared to both the last year's submission and to the post-workshop experiments reported in our previous notebook paper. However, compared to the level of performance reached by some of the other research groups, we are still clearly behind as all the groups seem to have improved from the previous year.

The results obtained by CLCaption are far from standing comparison with the best runs of TRECVID VTT 2019. However, multiple ways to improve them can be explored, such as different scoring methods or finer curriculum learning algorithms. We will explore these directions to boost the results of CLCaption.

ACKNOWLEDGMENTS

This work has been funded by the grant 313988 *Deep neural networks in scene graph generation for perception of visual multimedia semantics* (DeepGraph) of the Academy of Finland, the ANR (the French National Research Agency) via the ANTRACT project, and the European Union's Horizon 2020 research and innovation programme under grant agreement No 780069 *Methods for Managing Audiovisual Data: Combining Automatic Efficiency with Human Accuracy* (MeMAD). This work was supported by the Academy of Finland Flagship programme: Finnish Center for Artificial Intelligence, FCAI. The calculations were performed using computer resources provided by the Aalto University's *Aalto Science IT* project and CSC – IT Center for Science Ltd. We also acknowledge the support of NVIDIA Corporation with the donation of TITAN X and Quadro P6000 GPUs used for parts of this research.

REFERENCES

[1] George Awad, Asad Butt, Keith Curtis, Yooyoung Lee, Jonathan Fiscus, Afzal Godil, Andrew Delgado, Alan F. Smeaton, Yvette Graham, Wessel Kraaij, and Georges Quénot. Trecvid 2019: An evaluation campaign to benchmark video activity detection, video captioning and matching, and video search & retrieval. In *Proceedings of TRECVID 2019*. NIST, USA, 2019.

[2] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[3] Rakshith Shetty, Hamed R.-Tavakoli, and Jorma Laaksonen. Image and video captioning with augmented neural architectures. *IEEE MultiMedia*, 25(2):34–46, April 2018.

[4] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. *CoRR*, abs/1612.00563, 2016.

[5] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *CoRR*, abs/1506.03099, 2015.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[8] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, pages 41–48, 2009.

[9] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016.*, pages 265–283, 2016.

[10] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[11] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734, 2014.

[12] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[15] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. *CoRR*, abs/1604.06573, 2016.

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[17] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5288–5296, 2016.

[18] Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. TGIF: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4641–4650, 2016.

[19] David L. Chen and William B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 190–200, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[21] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.