Sequence to Sequence Model for Video Captioning

Yu Guo
Department of Electrical Engineering, Stanford
yuguo68@stanford.edu

Bowen Yao
Department of Statistics, Stanford
boweny@stanford.edu

Yue Liu
Department of Statistics, Stanford
yuel3@stanford.edu

Abstract

Automatically generating video captions with natural language remains a challenge for both the field of nature language processing and computer vision. Recurrent Neural Networks (RNNs), which models sequence dynamics, has proved to be effective in visual interpretation. Based on a recent sequence to sequence model for video captioning, which is designed to learn the temporal structure of the sequence of frames and the sequence model of the generated sentences with RNNs, we investigate how pretrained language model and attentional mechanism can aid the generation of natural language descriptions of videos. We evaluate our improvements on the Microsoft Video Description Corpus (MSVD) dataset, which is a standard dataset for this task. The results demonstrate that our approach outperforms original sequence to sequence model and achieves state-of-art baselines. We further run our model one a much harder Montreal Video Annotation Dataset (M-VAD), where the model also shows promising results.

1 Introduction

Video has been ubiquitous on the Internet, broadcasting channels, as well as personal devices nowadays, and massive new video data is being created from entertainment, sports and everyday life. There arises great interest to describe videos with natural language text for a wide variety of applications in human-robot interaction, semantic analysis of video contents, and describing videos for the visually impaired. However, this has been a challenging task for decades due to the complex temporal dynamics of videos.

Inspired by the recent advances in machine translation using recurrent neural networks (RNNs) [1, 2, 3], most recent attempts on video captioning use RNNs to 'translate' videos to natural language texts [4, 5, 6, 7, 8, 9]. Long Short Term Memory (LSTM)[10], a specific type of RNN, which can effectively capture long term temporal dependence, has been widely investigated for video captioning due to the intrinsic temporal nature of videos. A recent sequence to sequence, video to text model (S2VT) [5], which reads videos frames sequentially and output words sequentially using LSTMs, has drawn our attention due to its ability to learn the temporal information of videos, and flexibility to handle a variable number of video frames. While our model is based on this sequence to sequence model, we have made two major modifications aiming to improve it.

First, deep learning methods such as RNN typically require a large set of data to train. However, video data with high quality, paired description texts is currently limited [11]. On the other hand, raw text corpora is widely available and is able to provide rich linguistic information that could aid video captioning. In S2VT, the word embedding matrix are model variables, which can only be trained with a small corpus associated with the videos. Hence it shall be useful to use a pretrained

language model which is trained with a much larger corpus to generate word vectors. In our model, we use Glove [12] to generate such input word vectors.

Second, attention mechanisms have been incorporated into a variety of neural networks, since they allow salient features to dynamically come to the forefront as needed [13]. Several attention models have been reported to be useful in various tasks including machine translation[14, 2], image captioning[13, 15], speech recognition[16, 17], and video captioning[7, 18]. In our model, we implement an attention layer at the decoder stage which can draw useful information from previous word input.

We evaluate our model using METEOR[19], which takes into account both phrase matches as well as semantic alignment using WordNet synonyms and is proved to be the most relevant metric in video captioning task, where the number of reference is relatively small[20]. We run our experiment on the standard Microsoft Video Description Corpus (MSVD) and another much harder Montreal Video Annotation Dataset (M-VAD).

2 Related work

The sequence to sequence model, which our work is based on, has attracted considerable interests. Here we briefly introduce a few works that set this model as benchmark and aim to improve it or beat it with different architectures. In [11], the authors also investigate pretrained language models to improve S2VT. There also exist a few works using attention mechanisms similar to ours in spirit [18, 21].

An interesting work in [9] introduces two loss functions, the relevance loss and coherence loss, which measure the relevance degree of the video content and sentence semantics, and the contextual relationships among the generated words in the sentence, respectively. By jointly minimizing these two loss function, the authors achieve performance gain compared to [5].

A hierarchical recurrent neural encoder is proposed in [8], where in the encoding stage, an additional temporal filter layer composing of LSTM cells is introduced to better uncover the temporal dependency of the input frames. To our knowledge, this work has reported the highest METEOR score on the MSVD data so far, which can be attributed to the more nonlinearity added by the temporal filter layer.

A bidirectional multirate video reconstruction model is reported in [22]. By 'multirate', the frame sampling rate is varied in accordance with different motion speed; higher motion speed obtains higher sampling rate. By 'bidirectional', after sampling the frames, the authors construct both a forward and a backward temporal transitions of the frames, reflecting the temporal information in different views. Also, gated recurrent unit, rather than LSTM, is adopted in the model.

3 Approach

We propose a sequence to sequence model with global attention to tackle this video description task. Our model takes as input a sequence of video frames $(x_1, x_2, \cdots, x_m)$ and our output is a sequence of words $(y_1, y_2, \cdots, y_n)$ that describes the corresponding video. Both the video frames and the descriptions are allowed to be of varying length.

3.1 Input Feature Extraction

3.1.1 Video Frames

We resort to a similar approach as in image captioning[6][23] to extract video frame feature representations, where images are passed through a Convolutional Neural Network and the near top layer feature map gets extracted. For our work, we use a pre-trained Google Inception-v3[24][1] CNN, which achieves 3.58% top 1 error rate on *ILSVRC 2012*. We take the next-to-top layer named *pool3:0* as our video frame features, which lie in a 2048 dimensions space.

[1] https://storage.googleapis.com/download.tensorflow.org/models/inception_dec_2015.zip

3.1.2 Word Embeddings

All the words appeared in the data are tokenized and represented using one-hot vector encoding, which is a vector of length vocabulary-size and has value 1 only at one position and 0 otherwise. Note that we need to add to the vocabulary 2 special tokens [BOS] and [EOS] to represent the beginning and end of a sentence.

We also need an embedding representation of all words as input to the Neural Network. While previous sequence to sequence models[5][4] train their own word embeddings through back-propagation, we choose to use pre-trained Glove word embeddings[12], which makes use of global co-occurrence statistics of words and is proved to be effective in many downstream NLP tasks. We use 300-dimension Glove word embeddings trained on 840 Billion tokens[2]. Using these word embeddings trained on such enormous corpus as opposed to training the word embeddings on our relatively small dataset enables us to harness the information digested from huge amount of data and boost our model's performance. Our experiments in 4 prove such boost.

3.2 Model Architecture

Figure 1 is an illustration of our model. Similar double-layer LSTM structures can be found in [5][4] whereas in our model we introduce the extra attention layer and our experiments in 4 show that this attention mechanism is essential in boosting the model to give nearly state-of-art performance.

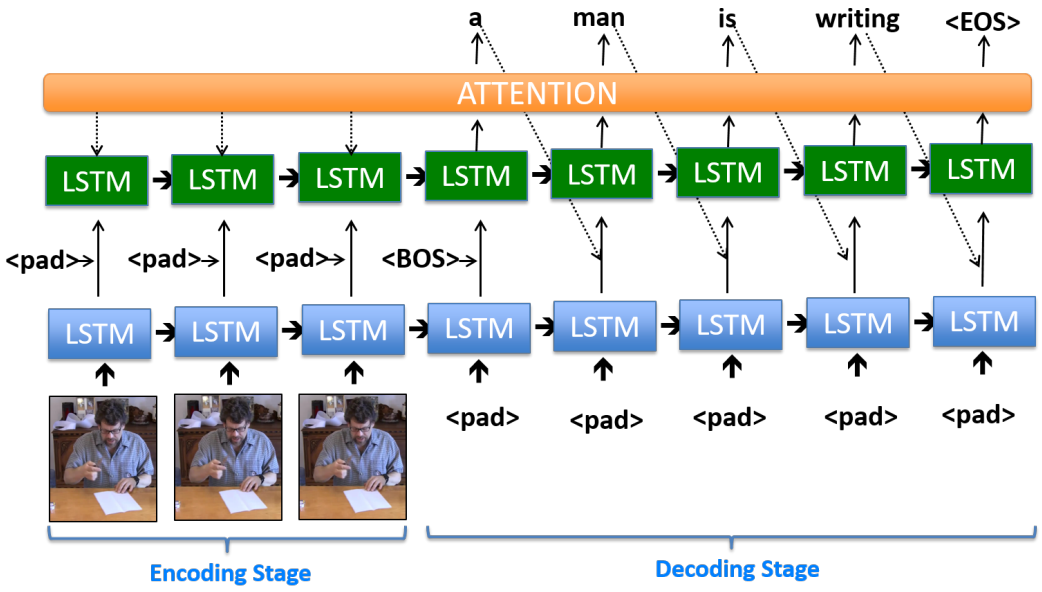

Figure 1: Sequence to sequence model for generating video description

For this double-layer network, the second layer (colored green) takes the first layer's (colored blue) output(h_t) as input(x_t). This kind of design allows shared parameters between encoder and decoder and enables the model to be trained end-to-end. We use 1000 hidden units for both layers of LSTM. For each layer of LSTM, we unroll it for 80 time steps, which is found to be a good balance between memory consumption and the ability to feed more frames to the model[5].

3.2.1 Training Time Behavior

During the encoding stage, the first layer LSTM takes a sequence of video frame features as input and decodes them as hidden representations (h_t). These encoded features then get concatenated with null padded input words (all zeros) and sent to the second LSTM layer for further encoding. Note that no loss or attention is calculated during the encoding stage.

[2]https://nlp.stanford.edu/projects/glove/

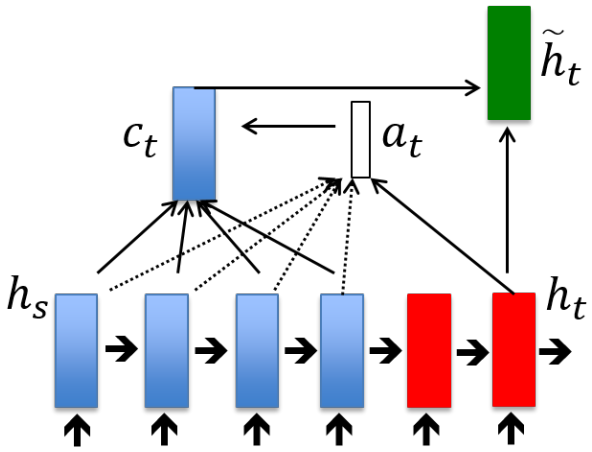

Figure 2: The effective global attention model

After exhausting all frames of a given video clip, the model enters the decoding stage, signified by a beginning-of-sentence ([BOS]) tag replacing the previous null padded input to wake the network up for decoding its current hidden representation into a sequence of words. During the decoding stage, the first layer LSTM takes null padded input and the outputs (h_t) get concatenated with the vector embeddings of the previous ground truth word. Then same as the encoding stage, these concatenated features get fed into the second layer, whose outputs then get passed into the attention layer.

The attention layer enables the decoding stage to look directly back into the source and learn to draw information from there if necessary. We use an attention model similar as [14], which does not interfere the calculation of the last layer LSTM and is proved to be more computational effective. Figure 2 illustrates the attention model in detail. At each time step of the decoding stage(colored red in Figure 2), we calculate the pair wise similarity between the decoding output (h_t) and any encoding(source) output (h_s) through a bilinear operation:

$$a_{s,t} = h_t^T W_{simi} h_s \tag{1}$$

where W_{simi} contains parameters to be learned.
Then we form a weighted sum of all source outputs with weights being the calculated similarity measure:

$$c_t = \sum_{s=1}^{S} a_{s,t} h_s \tag{2}$$

a_t can be viewed as a mixture of information drawn by h_t from all the source to facility the prediction. Then we concatenate h_t and a_t to get the final hidden representation $\tilde{h}_t$. At last, we finish the model by piping $\tilde{h}_t$ into a soft-max classifier with class size being the vocabulary size and a cross-entropy loss at the top.

3.2.2 Test Time Behavior

During test time, most part of the model remains unchanged, except that now we do not have the ground truth label. Hence at decoding stage, the output of the first layer LSTM gets padded with the embedding vector of the predicted word of the previous time step. We take all words emitted by the model before the first emitted [EOS] tag as the final predicted sentence.

3.2.3 Parameter Reduction

One important difference between video captioning task and other common sequence to sequence model application such as machine translation is that video captioning tends to have smaller dataset due to limited human resource to watch and label each video. Hence a model with too much parameters may be subjected to under-fit. Here we propose a model that allows us to further use information

embedded in the pre-trained Glove word vectors and reduce free parameters:
After calculating the final hidden representation $\tilde{h_t}$, instead of using a matrix of size $2h \times V$ where h is hidden representation size and V is vocabulary size, to directly transform $\tilde{h_t}$ into scores for each word, we use a matrix of size $2h \times d$ to map $\tilde{h_t}$ into word embeddings v, where d is the dimension of the Glove word embedding space. Then we predict the output word to be the word whose embedding vector has the largest inner product with v. Such inner product similarity measure is reasonable in that Glove is trained exactly the same way, and the parameter size is actually reduced to a great extent in that $d \ll V$. In 4, we show this technique boosts the model's performance.

4 Experiments

In this section, we describe experiments in the following order: data sets, experiment settings, evaluation metrics and experiment results.

4.1 Data sets

We used two video description data sets to train and evaluate our model: Microsoft Video Description Corpus (MSVD)[25] and Montreal Video Annotation Dataset (M-VAD)[26].

4.1.1 Microsoft Video Description Corpus (MSVD)

The Microsoft Video description corpus[25] was formed from a list of Youtube video segments, each of which records a single action done by one or several subjects. Then for each video segment, there is one sentence describing it. For the raw text descriptions, we did similar preprocessing as in [5]: First we filtered out all the non-English descriptions and those with mis-spelling words. Then we converted all the invalid descriptions to lower case and tokenized them. For raw videos, we sample images by every ten frames[4].

4.1.2 Montreal Video Annotation Dataset (M-VAD)

The M-VAD data set[26] consists of in total 41872 video description corpus. Compared with MSVD corpus, they are annotated scripts for movies, and does not have a consistent and simple sentence structure as MSVD does. We utilized similar methods[5] to clean, tokenize and reorganize the corpus data.

4.2 Evaluation metrics

We evaluated our model performance by several quantitative metrics including BLEU[27], ROUGE-L[28], CIDEr[20], and METEOR[19]. METEOR takes both syntactical and semantic similarity into account by matching the output with reference at token, paraphrase and WordNet synonyms.[5]. Since it was reported to be better[19] than BLEU[27] and ROUGE-L[28], in our project we decided to use METEOR as our main metric. We built our evaluation system based on the code[5] released in 2015.

4.3 Experiment settings

4.3.1 Train-Test split on MSVD

In MSVD, each video was cut into several pieces, each of which corresponds to one or more descriptions. In order to prevent pieces from the same video(although they may record different actions) appearing in both train and test sets, and thus bias up our evaluation scores, we divided the data into 3 chunks by their video names, forming Train, Development and Test sets with the ratio of 6:2:2. In addition, because METEOR[19] requires that predictions to be evaluated have a unique id on each line, for those video segments with multiple descriptions, we computed METEOR by matching the model output with each of the video segment descriptions. This procedure ended up a Train set of 48868 examples, and multiple Development and Test sets of around 350 examples each.

4.3.2 Train-Test split on M-VAD

The M-VAD data[26] has already been split into 3 parts the same way as discussed in section 4.3.1. After preprocessing as mentioned in section 4.1.2, we obtained a Train set of 36214 samples and a Test set of 4881 samples. Here due to time and computing power limitation, we only used Train and Test set and did not do hyper-parameter tuning on Development set for both the two data sets.

4.4 Experiment results

Based on the data described in the last two sections, we train several models with different word vectors and architectures, and do evaluation in the following order:
Let us denote the dimension of Glove vectors as d, the corpus size on which the vector were trained as s, our basic model as M_1, basic model with attention as M_2, and basic model with attention+Parameter Reduction as M_3. We will report evaluation results of the models M_1, M_2, M_3 with increasing d and s on the two data sets.

4.4.1 MSVD results

Test results on MSVD results can be seen in table1, along with several results from related work and similar models[5][7]. The first column is the model used, and the second column is additional settings for models, such as what image features to use, the dimension of word embeddings(denoted as d) and corpus size on which the embeddings were trained(denoted as s).Our basic model with 50-dimensional word embeddings was able to achieve comparable results (27.6%) as previous works (27.9%).It can be seen that as the model gets trained with longer word embeddings on more diverse corpus get used, it performs better and better ($27.6\% \rightarrow 28.9\%$).Moreover, as attention and parameter reduction are introduced to our model, the METEOR score also gets boosted ($28.9\% \rightarrow 30.4\%$),which outperformed the previous best results in [5] on this data set.

MODEL	MODEL SETTING	METEOR
S2VT	Flow (AlexNet)[5]	24.3
S2VT	RGB (AlexNet)[5]	27.9
S2VT	RGB (VGG) random frame order[5]	28.2
Temporal attention	GoogleNet[7]	29.0
S2VT	RGB (VGG)[5]	29.2
Temporal attention	GoogleNet + 3D-CNN[7]	29.6
S2VT	RGB (VGG) + Flow (AlexNet)[5]	29.8
S2S(ours)		
Basic	$d = 50$, $s = 6 \times 10^9$	27.6
Basic	$d = 300$, $s = 6 \times 10^9$	28.9
Basic	$d = 300$, $s = 840 \times 10^9$	28.9
Basic +Parameter reduction	$d = 300$, $s = 840 \times 10^9$	28.5
Basic +Attention	$d = 300$, $s = 840 \times 10^9$	29.4
Basic +Attention+Parameter reduction	$d = 300$, $s = 840 \times 10^9$	$\mathbf{30.4}$

Table 1: MSVD results(METEOR in %, higher is better)

4.4.2 M-VAD results

Test results on M-VAD results can be seen in table2, along with several results from related work and similar models[29][5][7]. Due to time and computation limitation, our model is just trained for 1 or 2 epochs and neither did we do hyper-parameter tuning on this data set. Nevertheless we achieved comparable results with some of the previous work. Also, we do observe that our model is getting better as we increase training epochs so we expect our model to give as satisfying or even better results compared to previous word given enough time and computing resources. One interesting note is that our model performs better without parameter reduction on this M-VAD data. This may be due to that M-VAD is a larger and more complicated data set, and as a result, our model needs more parameters to fit the data.

MODEL	MODEL SETTING	METEOR
Visual-Labels[29]		6.3
Temporal attention[7]	GoogleNet+3D-CNN	4.3
Mean pool	VGG	6.1
S2VT[5]	RGB (VGG)	6.7
S2S(ours)		
Basic +Attention	$d = 300$, $s = 840 \times 10^9$	**5.1**
Basic +Attention+Embedding Matrix	$d = 300$, $s = 840 \times 10^9$	**4.9**

Table 2: M-VAD results(METEOR in %, higher is better)

4.4.3 Qualitative results

In addition to quantitative evaluation, we also did qualitative evaluation by manually examining the model outputs and comparing them with corresponding ground truth and videos. We sample part of them and present them here 3.

We consider outputs from our model as 3 different types: correct prediction, relevant prediction and irrelevant prediction. All of them are listed by columns. From the results we can see, the model did a fairly good job in recognizing and describing unambiguous actions and relatively simple scenes, such as cooking and dancing. This results in the correct predictions. For the cases in the middle of 3, our model managed to recognize some of the objects and the their associated actions in the video but failed to extract deeper semantic information from the input sequence. This might be due to lack of prior knowledge in training data (for example "a man is serving a man" was described as "a man is talking with a man", and "rhinoceros" was described as "large animal"). This results in relevant predictions. For even more complicated actions and scenes, the model failed to recognize objects and understand the videos, which results in irrelevant predictions.

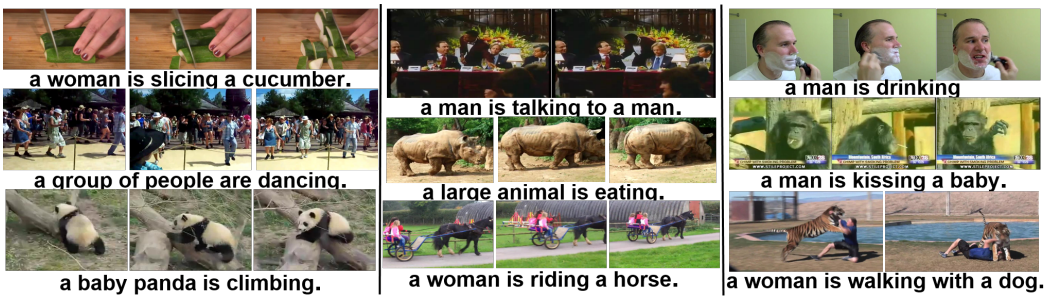

Figure 3: Qualitative results sampled from model prediction outputs: From left to right are correct descriptions, relevant descriptions and irrelevant predictions, separated by black vertical line

5 Conclusion

In this project we explored several model architectures for video captioning. Based on the sequence-to-sequence model originally proposed in [5], we introduced multiple modifications including Glove vectors, attention and parameter reduction. We show through experiments that Glove vectors and attention, by helping the model captures more semantic information from input sequences, boosts the model's performance. Additionally, parameter reduction via utilizing word embedding matrix also improves model performance and reduces number of trainable parameters.

Due to limitation of time and computing power, we did not do exhaustive hyper-parameter tuning. However, our model performance is very close to the state of art and outperforms the previous work results[5] on MSVD.

In the future we would like to do more exhaustive hyper-parameter tuning to fully investigate our model's potentials. Also, other techniques that have shown promising results in other sequence to sequence model, such as beam search[30] and local attention[14] should also be studied in this video to description context.

6 Author Contributions

- Yu Guo: Implement the main sequence to sequence architecture; Prepare video frames for the M-VAD dataset; Propose the parameter reduction with smaller word output matrix.
- Bowen Yao: Prepossess image and extract image features; Write runnable training and test architecture; Tune the model; Add attention mechanism.
- Yue Liu: Description corpus preprocessing for MSVD and M-VAD; Model text input construction; Model evaluation.

References

[1] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[4] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[5] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4534–4542, 2015.

[6] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[7] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015.

[8] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1029–1038, 2016.

[9] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4594–4602, 2016.

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

[11] Subhashini Venugopalan, Lisa Anne Hendricks, Raymond Mooney, and Kate Saenko. Improving lstm-based video description with linguistic knowledge mined from text. *arXiv preprint arXiv:1604.01729*, 2016.

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[13] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, volume 14, pages 77–81, 2015.

[14] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[15] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4651–4659, 2016.

[16] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, pages 577–585, 2015.

[17] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4945–4949. IEEE, 2016.

[18] Mihai Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Spatio-temporal attention models for grounded video captioning. *arXiv preprint arXiv:1610.04997*, 2016.

[19] Michael Denkowski Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. *ACL 2014*, page 376, 2014.

[20] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575, 2015.

[21] Zhao Guo, Lianli Gao, Jingkuan Song, Xing Xu, Jie Shao, and Heng Tao Shen. Attention-based lstm with semantic consistency for videos captioning. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 357–361. ACM, 2016.

[22] Linchao Zhu, Zhongwen Xu, and Yi Yang. Bidirectional multirate reconstruction for temporal modeling in videos. *arXiv preprint arXiv:1611.09053*, 2016.

[23] 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*, pages 3156–3164, 2015.

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[25] 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 (ACL-2011)*, Portland, OR, June 2011.

[26] Atousa Torabi, Christopher Pal, Hugo Larochelle, and Aaron Courville. Using descriptive video services to create a large data source for video annotation research. *arXiv preprint arXiv:1503.01070*, 2015.

[27] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[28] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.

[29] Anna Rohrbach, Marcus Rohrbach, and Bernt Schiele. The long-short story of movie description. In *German Conference on Pattern Recognition*, pages 209–221. Springer, 2015.

[30] Sam Wiseman and Alexander M Rush. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*, 2016.