Frame- and Segment-Level Features and Candidate Pool Evaluation for Video Caption Generation

Rakshith Shetty
Dept. of Computer Science
Aalto University, Espoo, Finland
rakshith.shetty@aalto.fi

Jorma Laaksonen
Dept. of Computer Science
Aalto University, Espoo, Finland
jorma.laaksonen@aalto.fi

ABSTRACT

We present our submission to the Microsoft Video to Language Challenge of generating short captions describing videos in the challenge dataset. Our model is based on the encoder–decoder pipeline, popular in image and video captioning systems. We propose to utilize two different kinds of video features, one to capture the video content in terms of objects and attributes, and the other to capture the motion and action information. Using these diverse features we train models specializing in two separate input sub-domains. We then train an evaluator model which is used to pick the best caption from the pool of candidates generated by these domain expert models. We argue that this approach is better suited for the current video captioning task, compared to using a single model, due to the diversity in the dataset.

Efficacy of our method is proven by the fact that it was rated best in MSR Video to Language Challenge, as per human evaluation. Additionally, we were ranked second in the automatic evaluation metrics based table.

1. INTRODUCTION

The problem of describing videos using natural language has garnered a lot of interest after the great progress recently made in image captioning. This development has been partly driven also by the availability of large datasets of images and videos with human-annotated captions describing them. The M-VAD [5] and MPII-MD [10] datasets used in the first Large Scale Movie Description Challenge suffered from having only one reference caption for each video. Multiple reference captions improve the language models learned and also lead to better automatic evaluation as shown in [12]. The Microsoft Video to Language dataset (MSR-VTT) [17] addresses this by providing 20 captions for each video.

A popular recipe for solving the visual captioning problem has been to use an encoder–decoder model [14, 13]. The encoder produces a feature vector representation of the input, which the decoder, usually a Long-Short Term Memory (LSTM) network, takes as input and generates a caption.

MM '16, October 15-19, 2016, Amsterdam, Netherlands

DOI: http://dx.doi.org/10.1145/2964284.2984062

Unlike in the case of images, where the convolutional neural network (CNN) image features have become the *de facto* standard features for many image understanding related tasks, no single video feature extraction method has achieved the best performance across tasks and datasets. Dense trajectories [15] and Improved dense trajectories [16] have been popular hand-crafted video feature extractors in action recognition. Following the success of deep CNNs on static images, there have been attempts with 3-D CNNs which operate directly on video segments [4, 11]. However, these models need a lot of training data and are usually pre-trained on some large dataset, e.g. the Sports-1M data.

The task of caption generation also requires us to describe the objects seen in the video and their attributes, in addition to recognizing actions. This can be solved by extracting features from individual frames [13] or keyframes [8] using CNNs pre-trained on the ImageNet dataset.

In this work we explore both problems, video feature extraction and visual content description, for the task of automatic video captioning. We experiment with methods which treat videos as bags of images and with methods mainly focusing on capturing the motion information present in videos. We also study how to combine these features and present an effective method to ensemble multiple caption generators trained on different video features.

2. DATASET

The MSR-VTT dataset consists of 10000 video clips with 20 human-annotated captions for each of them. Each video belongs to one of 20 categories including *music*, *gaming*, *sports*, *news*, etc. The dataset is split into training (6513 clips), validation (497 clips) and test (2990 clips) sets. The training set videos are between 10 to 30 seconds long, with most of them less than 20 seconds in duration.

The reference captions come from a vocabulary of ∼30k words, which we have filtered down to ∼8k after removing words occurring fewer than 5 times. Although the number of video clips in the dataset is relatively small compared to M-VAD [5] and MPII-MD [10], the dataset is attractive due to the diversity of its videos and due to the larger number of reference captions per video.

Any video captioning algorithm applied to this dataset needs to handle very diverse video styles, eg. video game play footage, news and interviews. Action recognition based features will suffer with videos with lots of cuts and scene changes. Conversely, with frame-based features, the system will fail to identify fine-grained differences in diverse action-oriented categories like sports or cooking. Thus our

approach is to use multiple captioning models, each trained on different types of features, and an evaluator network to pick the final caption among the candidates.

3. OUR METHOD

3.1 Overview

We use the standard encoder–decoder framework for our video caption generator. This framework has been widely used in the literature for machine translation and image [14] and video [8, 6] captioning. The encoder stage consists of various feature extraction modules which encode the input video into a vectorial form. The decoder is the language model that generates a caption using these feature vectors.

We also utilize an additional model, an evaluator network which takes as its inputs a generated caption and a video feature, and computes a score measuring how well these two match. This evaluator is used to build an ensemble of multiple caption generator models trained on different features.

3.2 Feature Extraction

We use two different paradigms for video feature extraction. The first one is to treat the video as just a sequence of 2-D static images and use CNNs trained on ImageNet [1] to extract static image features from these frames. The second approach is to treat the video as 3-D data, consisting of a sequence of video segments, and use methods which also consider the variations along the time dimension for feature extraction. In both the above methods, pooling methods are used to combine multiple frame/segment-level features into one video-level feature vector.

3.2.1 Frame-Level Feature Extraction

We use the GoogLeNet [9] model trained on ImageNet for extracting frame-level features. For efficiency, we only extract these features from one frame every second.

In the GoogLeNet we have used the *5th Inception module*, having the dimensionality of 1024. We augment these features with the reverse spatial pyramid pooling proposed in [2] with two scale levels. The second level consists of a 3×3 grid with overlaps and horizontal flipping, resulting in a total of 26 regions, on the scale of two. The activations of the regions are then pooled using diverse combinations of average and maximum pooling. Finally, the activations of the different scales are concatenated resulting in 2048-dimensional features. By using the varying combinations of pooling techniques, we have obtained three somewhat different feature sets from the same network.

We use the mean pooling to fuse the multiple frame-level features extracted from one video into a single vector.

3.2.2 Video Segment Feature Extraction

For extracting segment-based features we have used two different algorithms, namely dense trajectories [15, 16] and 3-D CNN network based C3D features [11].

We have used both the standard [15] and the improved versions [16] of the dense trajectory features. For both versions, the trajectories and their descriptors (HOG, HOF, MBHx and MBHy as described in [15]) are first extracted for the entire video. All of these five features are separately encoded into a fixed-size vector using bag-of-features encoding with a codebook of 1000 vectors. Each codebook is obtained using k-means clustering on 250k random trajectory samples from the training set. Finally, concatenating the vector encodings of each of the descriptors we get a video feature vector of 5000 dimensions.

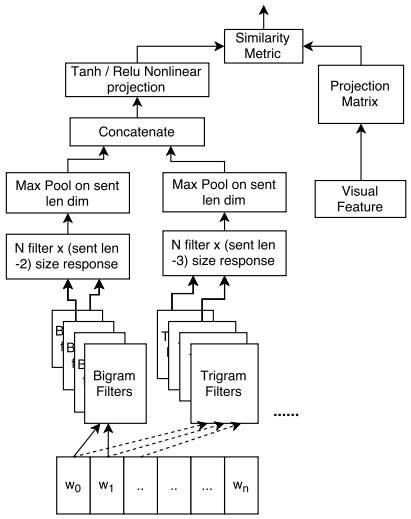


Figure 1: CNN-based evaluator network to compute the similarity between a video and a caption.

As an alternative, we also extract video-segment features with the deep 3-D convolutional model, C3D [11], pre-trained on the Sports-1M dataset. The features are extracted for every 16 frames from the *fc7* layer of the network and mean pooled to get a 4096-dimensional feature vector representation of the video.

We also utilize the video category information available for all videos in all splits of the dataset. This is input to the language model as a one-hot vector of 20 dimensions.

3.3 Language Model

We use the LSTM network based language model and training procedures widely used in visual captioning [14] with some tweaks. Firstly, we train deeper LSTM networks with residual connections between the layers. This is similar to the residual connections introduced in [3] for CNNs. In our case, the output of the lower LSTM layer is added to the output of the above layer with residual connections.

We also allow the language model to use two separate input features, one to initialize the network and the other one to persist throughout the caption generation process as shown in [8]. Thus our language model can be trained using two different feature inputs, *init* and *persist*, fed through separate channels. We utilize this architecture to train many different language models by using the diverse feature pairs from the ones described in Section 3.2.

3.4 Evaluator Network

Using the different video feature combinations we have, we can train diverse caption generator models. Examining the captions generated by them for a set of videos, we find that each model tends to generate the best captions for different videos. If we can reliably evaluate the suitability of a caption for a given video, we can pick out the best candidate and achieve better results than with any single model.

We implement this by training a new *evaluator network* whose task is to pick out the best candidate from the can-

Table 1: Performance of various features and network depths on the validation set of MSR-VTT

#	init	persist	depth	perplex	BLEU-4	METEOR	CIDEr	ROUGE-L
1	dt	gCNN+20Categ	1	27.31	0.396	0.268	0.438	0.588
2	dt	gCNN+20Categ	2	27.73	0.409	0.268	0.433	0.598
3	dt	gCNN+20Categ	3	28.44	0.370	0.262	0.397	0.575
4	idt	gCNN+20Categ	2	28.13	0.398	0.268	0.432	0.587
5	dt	c3dfc7	2	29.58	0.369	0.268	0.413	0.577
6	*CNN evaluator based ensemble of best 4 models*				**0.411**	**0.277**	**0.464**	**0.596**

didate set, given an input video. The model takes as input one video feature and one input sentence and computes a similarity metric between them. The evaluator is trained discriminatively, in contrast to the language models which have been trained generatively. Thus, it learns to focus on semantically interesting phrases without having to learn the details needed to generate a syntactically correct sentence.

Figure 1 shows the block diagram of our evaluator. It consists of a CNN sentence encoder network which takes a sequence of word vectors as input and learns a fixed-size vector embedding of the sentence. The model also has a projection matrix which maps the video feature to the same space as the sentence embedding. We then use the cosine distance to score the similarity between the two embeddings.

The evaluator model is trained to maximize the score assigned to matching caption–video pairs, while minimizing the score assigned to random negative caption–video pairs. We sample up to 50 negative captions per video from the ground truth captions of other videos. The trained evaluator is used to rerank captions from multiple generator models, trained on different feature combinations and architectures, to pick the best matching candidate.

4. EXPERIMENTS AND RESULTS

All LSTM layers in the language models we train have 512 hidden units, but we vary the number of layers as shown in Table 1. We train the language models on the entire training set of the MSR-VTT dataset using stochastic gradient descent with the RMSProp algorithm and dropout regularization. The errors are backpropagated to all the language model parameters and word embedding matrices, but the feature extraction models are kept constant.

In all our experiments we use beam search to generate sentences from a trained model. After experimenting with different beam sizes, we found that the beam size of $b = 5$ works adequately across all our models.

4.1 Validation Set Results

In order to measure the performance differences due to the different feature combinations and architectural changes, we use the validation set of the MSR-VTT dataset which contains 497 videos. Performance is measured quantitatively using the standard automatic evaluation metrics, namely METEOR, CIDEr, ROUGE-L and BLEU.

Table 1 shows the results on the validation set. The columns *init* and *persist* indicate the features used for those input channels, respectively, in the language model. The column *depth* is the number of layers in the language model and *perplex* is the perplexity measure on the validation set.

Models #1, #2 and #3 all use the dense trajectory (dt) features as *init* input and the mean pooled frame-level GoogLeNet features concatenated with the video category vector (gCNN+20Categ) as the *persist* input. They vary in the number of layers in the language model. Comparing their performances we see that the 2-layer model outperforms the single layered model by a small margin, while the 3-layer version is the inferior one.

Model #4 is similar to #2, but uses the improved dense trajectories (idt) as the *init* input instead. Model #5 differs from #2 by the fact that it uses mean pooled 3-D convolutional features as the *persist* input. We see that both #4 and #5 are competitive, but slightly worse than our best single model, #2. Upon qualitatively analyzing the model outputs, we see that each of them performs well on different kinds of videos. For example, model #5, which only uses input features trained for action recognition, does well in videos involving a lot of motion, but suffers in recognizing the overall scenery of the video. Conversely, model #2 trained on frame-level features does better in recognizing objects and scenes, but makes mistakes with the sequence of their appearance, possibly due to the pooling operation. This fact can be observed in examples in Figure 2. Model #5 produces a better caption on the video in the first column, while #2 does better on the video in the second column.

To get maximum utility out of these diverse models, we use the CNN evaluator network to pick the best candidate from the pool of captions generated by our top four models, #1, #2, #4 and #5. The evaluator is trained using the gCNN+20Categ as the video feature. It is shown as model #6 in Table 1. We can see that the CNN evaluator significantly outperforms, in all the four metrics, every model it picks its candidates from. Figure 2 shows an example in the third column where the CNN evaluator picks a better caption than the one generated by our best single model.

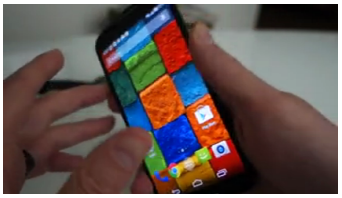
4.2 Challenge Results

Since the CNN evaluator model performed the best on the validation set, we submitted that result in the MSR-VTT Challenge. Our submission appears on the leaderboard as *Aalto*. The submissions were evaluated on the blind test set using the above automatic metrics. The results are shown in Table 2a. Our submission was ranked overall second considering the average ranking across the metrics.

The submissions were also subject to human evaluation as the automatic metrics are known to deviate from human judgements in case of both image [7] and video [6] data. The human evaluation was based on three criteria: Coherence (C1), Relevance (C2) and Helpfulness for the blind (C3). Table 2b presents the results of the human evaluation. As per human judgement, our submission was ranked the first among the 22 entries in the challenge.

Comparing the automatic metric and human evaluation based leaderboards we see that the disagreement between the two is relatively minor, with most teams in the top 10

#6: a man is running in a gym
#2: a man is running
#5: a man is playing basketball



#6: a person is playing with a rubix cube
#2: a man is holding a phone
#5: a person is playing with a rubix cube

#6: cartoon characters are interacting
#2: a person is playing a video game
#5: a group of people are talking

Figure 2: Sample captions generated for some test set videos by our models.

Table 2: Top 5 teams as per metrics and human evaluations on the test set

Team	BLEU-4	METEOR	CIDEr	ROUGE-L
v2t_navigator	**0.408**	**0.282**	0.448	**0.609**
Aalto	0.398	0.269	0.457	0.598
VideoLAB	0.391	0.277	0.441	0.606
ruc-uva	0.387	0.269	**0.459**	0.587
Fudan-ILC	0.387	0.268	0.419	0.595

(a) Automatic metric-based evaluation

Team	C1	C2	C3
Aalto	**3.263**	3.104	**3.244**
v2t_navigator	3.261	3.091	3.154
VideoLAB	3.237	**3.109**	3.143
Fudan-ILC	3.185	2.999	2.979
ruc-uva	3.225	2.997	2.933

(b) Human evaluation

changing their ranking by only one position. This can most likely be attributed to having a large number of 20 reference captions per video for the automatic evaluation.

5. CONCLUSIONS

We have presented our video captioning model which shared the first position in the MSR Video to Language Challenge. Our architecture consists of multiple caption generator models, based on the encoder–decoder framework and trained on different video features, and an evaluator model trained to pick the best candidate from the ensemble. We argue that this model is well-suited for the MSR-VTT dataset which contains diverse types of videos. Corroborating our claims, our model topped the human evaluation based leaderboard and finished as second in the automatic metric based leaderboard in the challenge.

6. ACKNOWLEDGMENTS

This work was funded by the grant 251170 of the Academy of Finland and calculations performed with resources of the Aalto University School of Science “Science-IT” project.

7. REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[2] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. arXiv.org:1403.1840, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv*, abs/1512.03385, 2015.

[4] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[5] A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele. A dataset for movie description. In *CVPR*, 2015.

[6] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. J. Pal, H. Larochelle, A. C. Courville, and B. Schiele. Movie description. *arXiv*, abs/1605.03705, 2016.

[7] Y. C. M. Ruggero, Ronchi, and T.-Y. Lin. Microsoft COCO 1st captioning challenge. http://lsun.cs.princeton.edu/slides/caption_open.pdf, 2016 (accessed July 1, 2016).

[8] R. Shetty and J. Laaksonen. Video captioning with recurrent networks based on frame-and video-level features and visual content classification. *ICCV Workshop on LSMDC*, arXiv abs/1512.02949, 2015.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv*, abs/1409.4842, 2014.

[10] A. Torabi, P. Chris, L. Hugo, and C. Aaron. Using descriptive video services to create a large data source for video annotation research. *arXiv*, abs/1503.01070, 2015.

[11] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. C3D: generic features for video analysis. *arXiv*, abs/1412.0767, 2014.

[12] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, June 2015.

[13] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *CVPR*, 2015.

[14] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[15] H. Wang, A. Kläser, C. Schmid, and C. Liu. Action recognition by dense trajectories. In *CVPR*, 2011.

[16] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.

[17] J. Xu, T. Mei, T. Yao, and Y. Rui. MSR-VTT: A large video description dataset for bridging video and language. In *CVPR*, 2016.