Sjöberg, Mats; Tavakoli, Hamed R.; Xu, Zhicun; Mantecón, Héctor Laria; Laaksonen, Jorma

PicSOM Experiments in TRECVID 2018

Published in:
Proceedings of the TRECVID 2018 Workshop

Published: 01/11/2018

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Sjöberg, M., Tavakoli, H. R., Xu, Z., Mantecón, H. L., & Laaksonen, J. (2018). PicSOM Experiments in TRECVID 2018. In *Proceedings of the TRECVID 2018 Workshop* TRECVID. https://www-nlpir.nist.gov/projects/tvpubs/tv18.papers/picsom.pdf

PicSOM Experiments in TRECVID 2018

Workshop notebook paper

Mats Sjöberg[+], Hamed R. Tavakoli[+], Zhicun Xu[*], Héctor Laria Mantecón[+], Jorma Laaksonen[+]
[+]Department of Computer Science
Aalto University School of Science
P.O.Box 15400, FI-00076 Aalto, Finland

[*]Department of Signal Processing and Acoustics
Aalto University School of Electrical Engineering
P.O.Box 12200, FI-00076 Aalto, Finland

firstname.lastname@aalto.fi

Abstract

This year, the PicSOM group participated only in the Video to Text (VTT), Description Generation subtask. For our submitted runs we used either the MSR-VTT dataset only, or MS COCO and MSR-VTT jointly for training. We used LSTM recurrent neural networks to generate descriptions based on multi-modal features extracted from the videos. We submitted four runs:

- PICSOM_1: uses ResNet features for initialising the LSTM generator, and object and scene-type detection features as persistent input to the generator which is trained on MS COCO + MSR-VTT,
- PICSOM_2: uses ResNet and object detection features for initialisation, and is trained on MS COCO + MSR-VTT, this is the only run based on our new PyTorch codebase,
- PICSOM_3: uses ResNet and video category features for initialisation, and trajectory and audio-visual embedding features for persistent features, trained on MSR-VTT only,
- PICSOM_4: is the same as PICSOM_3 except that the audio-visual embedding feature has been replaced with audio class detection outputs.

The most significant difference between our runs came from expanding the original MSR-VTT training dataset by including MS COCO, which contains images annotated with captions. Having a larger and more diverse training set seems to bring larger improvements to the performance measures than using more advanced features. This finding has been confirmed also by our post-submission experiments that we are still continuing.

I. Introduction

In this notebook paper, we describe the PicSOM group's experiments for the TRECVID 2018 evaluation [1]. We participated only in the Video to Text Description (VTT) subtask for Description Generation. Our approach is a variation of "Show and tell" model [2], augmented with a richer set of contextual features [3]. This year we are transitioning from an old Theano-based neural captioning system to a new PyTorch-based one, mainly because the new code base will make future development and experimentation easier from a practical standpoint. Both systems have been used to produce the runs presented in this paper, and they are described in more detail in Section II. Next, we describe the features (Section III) and datasets used for training (Section IV). Our experiments, submitted runs and results are discussed in Section V and conclusions are drawn in Section VI.

II. Neural captioning models

In our experiments we have used two different Python-based software projects for caption generation. The first and older one, *NeuraltalkTheano*, uses the Theano library whereas the second and newer one, *DeepCaption*, uses the PyTorch library.

A. NeuraltalkTheano

The Theano-based neural captioning system is described in our recent paper [3], and the source code of the implementation is freely available.[1] The neural architecture is similar to the one proposed in [2], but adds several novel properties including residual connections between the LSTM layers, and the effective usage of persistent features. Persistent features are given as an additional input to the recurrent model at each step of the caption generation, while in the typical setup features are used only for initializing the LSTM generator. The loss function is Cross Entropy between the real caption and the probabilities of the caption produced by the model. A full description of the method can be found in [3].

B. DeepCaption

This year we have started to develop a new PyTorch code base, also available as open source.[2] The goal is to re-implement NeuraltalkTheano in a PyTorch architecture that is more maintainable in the long run, and thus facilitate faster development and experimentation. So far we have not yet implemented all the features of NeuraltalkTheano, in particular beam search and residual connections are still missing and have thus not been used in the DeepCaption-based result presented here.

The features are translated to the hidden size of the LSTM by using a fully connected layer. We apply dropout and batch normalization [4] at this layer. As the loss function, we

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

similarly use Cross Entropy, in addition to the Reinforcement Learning Self-critical loss function [5] in order to fine-tune a well-performing model. The fine-tuning is implemented either by switching to this loss in training time or by specifying a pre-trained model to load.

III. Features

In this section we describe the visual and auditory features used in our experiments. Table I summarizes the features and their dimensionalities. In the cases when an image modality feature extraction method has been applied to a video object, we have used the middlemost frame of the video.

TABLE I
SUMMARY OF THE FEATURES USED IN OUR EXPERIMENTS.

abbr.	feature	dim.	modality
rn	ResNet	4096	image
frA	Faster R-CNN	480	image
frB	Faster R-CNN	80	image
s	SUN397	397	image
c	category	20	image
t	trajectory	5000	video
as	audioset	527	audio
mm	multimodal	2048	multimodal

A. CNN

We are using two types of CNN features, one representation for still images, including single video frames, and one for the sequence of images, i.e. videos. Both architectures are based on the ResNet [6] model.

Image features, 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 article.

Video features, on the video level, when there is a sequence of 16 images, we are using a 3D CNN ResNet architecture with 152 layers [7], where final fully connected layers are removed to produce a 2048-dimensional feature vector. These features are fed to the language model.

B. FasterRCNN

The types of objects and their locations in the visual scene have 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) [8]. 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.[3] We use these object proposals to create object location maps. We divide the image into independent m horizontal and n vertical strips. The vertical and horizontal cells will thus overlap and the total number of cells is $m + n$.

We first define a grid on the image where each of the cells, $F_c(i)$, accumulates the integral of Gaussian distributions fit to the object proposals of the class category c as

$$F_c(i) = \sum_{b_k \in BB(c)} \iint_{b_k \cap G(i)} p(b_k) N(\text{center}(b_k), \text{diag}(b_k)) \ , \quad (1)$$

where $BB(c)$ is the set containing bounding box object proposals for category c, $p(b_k)$ is the confidence assigned by the detector to proposal b_k, $G(i)$ is the grid cell at position i and $N(\mu, \sigma)$ are Gaussians of given mean μ and standard deviation σ. In our experiments we used $m = n = 3$ and abbreviate these $(3 + 3) \times 80 = 480$ -dimensional features as "frA".

We can further reduce the feature size by discarding the location information completely and just encoding the object detection scores on the image level. We obtain such an 80-dimensional feature vector using the detection score for each category, and refer to it as "frB". For brevity of notation, concatenation of "frA" and "frB" to 560-dimensional features will be abbreviated as "frAB" in the tables below.

C. Semantic concept and category features

For still images, we use a scene-type cue as a feature to the language model. We detect the scene-type using a bank of specialized visual scene detectors trained on CNN features extracted for the SUN Scene Categorization Benchmark database.[4] Scene-type classifiers are designed using Radial Basis Function Support Vector Machine (RBF-SVM) [9]. CNN features extracted from GoogLeNet [10] pre-trained on the MIT Places dataset (from the *3rd classification branch*) is used to train a separate classifier for each variant. Each classifier determines the degree of association of a given image to the 397 scene types of the SUN database. Thus, for an input image, we form a 397-dimensional feature vector consisting of these raw scene type scores in the range of $[0, 1]$. This feature vector is referred to as "s" in Table I and below.

We also utilize the video category information, available in the MSR-VTT, as a one-hot feature vector of 20 dimensions. For the test set we have generated the corresponding feature by training a set of 20 category detectors and using the MSR-VTT category information as training data. For this purpose, we again used RBF-SVM classifiers and GoogLeNet features extracted using an ImageNet pre-trained model. This feature vector is referred to as "c".

D. Trajectory features

For encoding genuinely video content, we use trajectory features. Dense trajectories [11] and their histogram of oriented gradients (HOG), histogram of optical flow (HOF), and motion boundary histogram x and y directions (MBHx and MBHy) descriptors are first extracted for the entire video. These five features are separately encoded into a fixed-size vector using a bag-of-features encoding with a codebook of 1,000 vectors. Each codebook was obtained using k-means

[3] http://cocodataset.org/

[4] https://groups.csail.mit.edu/vision/SUN/

clustering on 250,000 random trajectory samples from the training set. Finally, concatenating the vector encodings of each of the descriptors results in a video feature vector of 5000 dimensions. This feature vector is referred to as "t" in our tables.

E. Audio features

The provided audio features are the occurrence probabilities for the 527 classes of the Google AudioSet Ontology[12]. AudioSet contains over 2 million 10-second human-labeled soundtracks segmented from YouTube videos. Each soundtrack can have multiple labels such as "acoustics guitar" or "door bell". Instead of providing the original audio files, AudioSet gives compact 128-dimensional embeddings which are the output of a modified VGG model, namely VGGish, for the log-mel spectrogram of audio with around a length of one second. Thus the dimension of the training data is 10×128 after being fed into the VGGish.

Inspired by the work [13] and [14], we built a similar multi-level attention model for the 10-second audio classification and achieved a mean average precision of 0.344. Since the length of the audio files in TRECVID are around 6 seconds, we decided to concatenate the same audio twice or more times and then truncated the extra parts to match the 10-second requirements. Finally, the modified 10-second audios are fed into a multi-level attention model to get the probabilities. This feature vector is referred to as "as" in Table I and below.

F. Multimodal features

To encode audiovisual information, we adopted a joint embedding space based on the ResNet [6] deep neural architecture. The architecture has two branches, one for audio and one for video. The audio branch consists of 7 residual convolution blocks and accepts a log spectrogram as input. The log spectrogram, corresponding to one second of audio data, is processed into a vector of 2048 prior to the combination with the video data by going through two fully-connected layers. On the video branch, we use the video features described in Section III-A. The output of the audio branch is used to re-weight the video branch weights. These combined responses are then fed into another fully connected layer, producing a vector of size 2048.

For training we borrow the weights for the video branch from pre-trained models of [7], which are trained on the kinetics database [15]. For the audio part, we train our network on the speech commands database [16] and then, borrow the weights from this model. The audio-visual embedding is then trained on an auxiliary task, namely action recognition. Afterwards, the obtained feature representations from this new architecture, referred to as "mm" in Table I, are used for multimodal caption generation.

IV. Training data

Here we describe the datasets used for training our captioning models. 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 frAB s
M	MSR-VTT	6,513 vid	130,260	rn frAB s c t as mm
T	TGIF	125,713 vid	125,713	rn frAB s
V	MSVD	1,969 vid	80,800	rn frAB s

A. COCO

The *Microsoft Common Objects in COntext (MS COCO)* dataset [17] 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 [18] 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. It is intended to foster spatio-temporal information modelling and pooling strategies in video data, as well as make a broader range of domains available as opposed to previous datasets.

C. TGIF

The *Tumblr GIF (TGIF)* dataset [19] contains 100,000 animated GIFs and 120,000 natural language sentences. This dataset aims to provide motion information involved between image sequences (or frames). Authors explain that focusing on a limited series of still frames, often without narrative or need for context, and always without audio is an easier step towards full video understanding.

D. MSVD

The *Microsoft Research Video Description Corpus (MSVD)* [20] consists of 85,000 English video description sentences and more than 1,000 for a dozen more languages. Gathering efforts for this dataset presented early crowd-sourcing methodologies for video annotation. It contains a set of 2,089 videos, showing a single, unambiguous action or event. Additionally, descriptions of the same video segment can then be used as translation data if they are in different languages.

V. Experiments and results

During the development stage, we ran a number of experiments to select the best combinations of features and training data. We evaluated our results using the previously released TRECVID VTT 2016 test set. The four runs that we finally submitted are identified as "s1" to "s4" in Table III. Runs "b1" to "b4" we had created also before the submission deadline and we evaluated them locally when the 2018 ground truth was released. Then, experiments "a1" and "a2" were

TABLE III
RESULTS OF OUR SUBMISSIONS (S1,. . . , S4) AND SOME NOTEWORTHY PRE (B1,. . . , B4) AND POST (A1,. . . , A4) EXPERIMENTS.

	setup					2016		2018				
id	**mod**	**loss**	**init**	**pers**	**data**	**METEOR**	**CIDEr**	**METEOR**	**CIDEr**	**CIDErD**	**BLEU**	**STS**
b1	dc	ce	rn	–	C+M	0.2135	0.2620	0.1513	0.1584	0.0471	**0.0110**	
b2	dc	ce	rn+frAB	–	C+M	**0.2186**	**0.2872**	0.1515	0.1714	0.0475	0.0082	
b3	nt	ce	rn+c	t	M	0.2005	0.2379	0.1415	0.1495	0.0364	0.0051	
b4	nt	ce	rn	frAB+s	C	0.1890	0.1907	**0.1675**	**0.1808**	**0.0641**	0.0091	
s1	nt	ce	rn	frAB+s	C+M	0.2147	**0.2886**	0.1488	**0.1720**	0.0450	0.0053	**0.3806**
s2	dc	ce	rn+frB	–	C+M	**0.2214**	0.2750	**0.1540**	0.1660	**0.0480**	**0.0091**	0.3739
s3	nt	ce	rn+c	t+mm	M	0.2039	0.2437	0.1468	0.1520	0.0390	0.0055	0.3676
s4	nt	ce	rn+c	t+as	M	0.2021	0.2413	0.1464	0.1590	0.0400	0.0048	0.3713
a1	dc	ce	rn+frAB+s	–	C+M+T	0.2238	0.3158	0.1562	0.1910	0.0525	0.0122	
a2	dc	ce	rn+frAB+s	–	C+M+T+V	0.2300	0.3080	0.1654	0.1984	0.0653	0.0166	
a3	dc	ce	rn	–	C+T	0.2343	0.2997	0.1776	0.1948	0.0700	0.0197	
a4	dc	ce+sc	rn	–	C+T	**0.2562**	**0.4827**	**0.2058**	**0.2843**	**0.1028**	**0.0298**	

performed before the workshop and reported in the notebook draft, whereas "a3" and "a4" were run after the workshop.

For the NeuraltalkTheano system (abbreviated as "nt" in Table III), the beam size in the caption generation stage was varied and it was found out that the models consistently performed best with beam size equal to one. For this reason we did not yet implement beam search for the DeepCaption system (abbreviated as "dc" in Table III), instead we used a simple greedy selection approach equivalent to having beam size one.

With the NeuraltalkTheano system we first tried using either only the COCO training data or only the MSR-VTT training data. These are seen as runs "b3" and "b4" in Table III, respectively. With the MSR-VTT training data we were also able to experiment with the multimodal "mm" and audioset "as" features, which was not possible with the image-only COCO data. These experiments are seen as submissions "s3" and "s4" in the result table, respectively. The best results with the 2016 testing data were obtained when the COCO and MSR-VTT data were used together by using only the visual modality and the middlemost video frames of the latter dataset. We regarded this as our overall best result when evaluated as the CIDEr score on the 2016 testing data and submitted it as "s1". For all our NeuraltalkTheano experiments we varied the combinations of the used features for the LSTM model initialization and for the persistent features. The feature combinations shown in Table III provided the best results on the 2016 testing data. We also varied the beam search size, but without an exception, the best results were always obtained with the beam size equal to one.

Based on evaluation on the TRECVID 2016 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. Exceptionally the runs "a1" and "a2" use 1024 for the embedding vector, which seems motivated as well by the larger dimensionality of the input features. Both in the input translation layer and in the LSTM we applied a dropout of 0.5. We used centered RMSprop [21] with a learning rate of 0.001 and weight decay (L2 penalty) of 10^{-6}.

The last experiment "a4" starts from the "a3" pre-trained model, minimizing Self-critical loss "sc" instead of the previous Cross Entropy "ce". To compute the rewards for the loss, CIDEr metric is used by providing all the reference captions for an image. This is the same way as it would be done in the test time, following the indications of the paper [5].

All experiments are briefly summarized and their results presented in Table III. The five "setup" columns specify the captioning model (nt=NeuralTalkTheano, dc=DeepCaption), the loss function (ce=Cross Entropy, sc=Self-critical), the initializing and persistent features, and the datasets used in the LSTM model training.

The features are concatenations of the following:

rn = ResNet, see III-A
frA = Faster R-CNN 480-dim, see III-B
frB = Faster R-CNN 80-dim, see III-B
s = SUN397, see III-C
c = Category of 20 video genres, see III-C
t = Trajectory, see III-D
as = Audioset, see III-E
mm = MultiModal, see III-F

The used datasets are concatenations of the following datasets, each described in one of the subsections of the previous section:

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

VI. CONCLUSIONS

The most practical conclusion for our group's internal use is that we have been successful in replacing our old Theano-based captioning system with a new PyTorch-based one that has overtaken the old system in performance.

Compared to the level of performance reached by some of the other research groups we are, however, still clearly

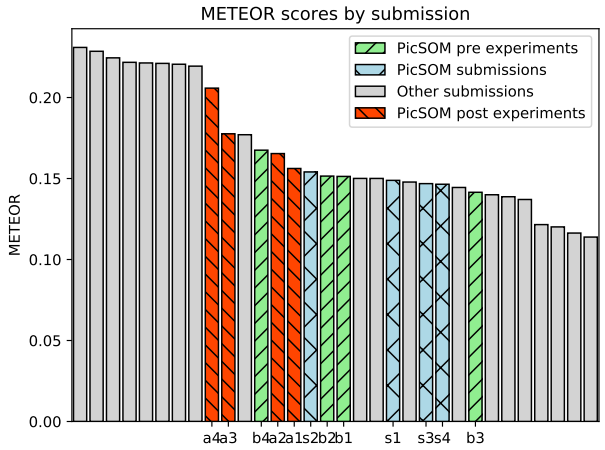

Fig. 1. METEOR results of our group and others.

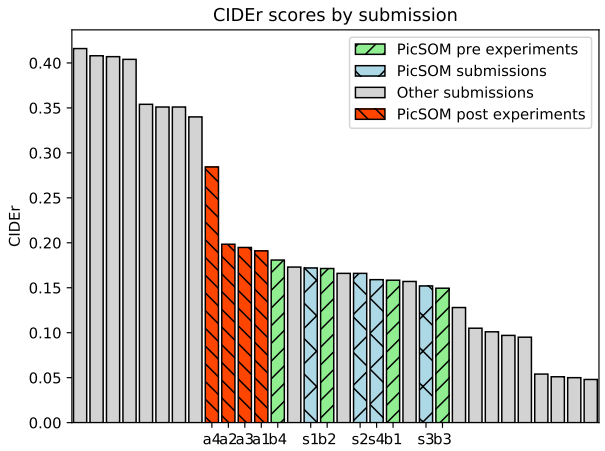

Fig. 2. CIDEr results of our group and others.

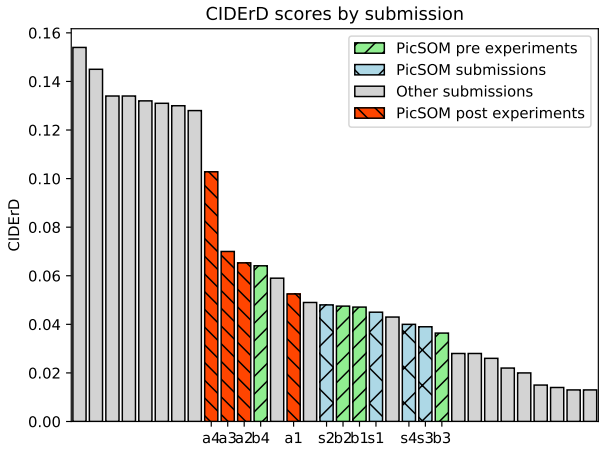

Fig. 3. CIDErD results of our group and others.

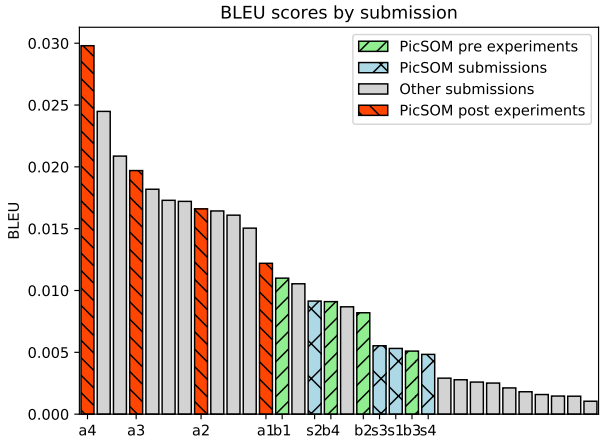

Fig. 4. BLEU results of our group and others.

behind. We will still need to continue our efforts to improve the evaluation scores obtained by our DeepCaption system.

Comparing the results we obtained by using the 2016 VTT test data and those obtained with this year's test data, we can draw two conclusions. First, this year's results seem to be clearly worse than those of 2016. This will need further studies as the number of reference captions has increased from two to five, which in general should have had the opposite effect on the results. Second, it seems that we were not able to choose the best-performing models among the variants we experimented with, based on the evaluation scores with the 2016 data.

The pre-submission results obtained by using only the COCO training data now seem to be very competitive. In general, however, our results indicate that using more training data, even with only one image frame for each training set video, is more beneficial than using genuinely video, audio or multimodal features.

Our final post-workshop experiments evince the promise of Self-critical learning as the final tuning method.

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 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). 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, Jonathan Fiscus, Afzal Godil, Alan F. Smeaton, Yvette Graham, Wessel Kraaij, Georges Quénot, Joao Magalhaes, David Semedo, and Saverio Blasi. Trecvid 2018: Benchmarking video activity detection, video captioning and matching, video storytelling linking and video search. In *Proceedings of TRECVID 2018*. NIST, USA, 2018.

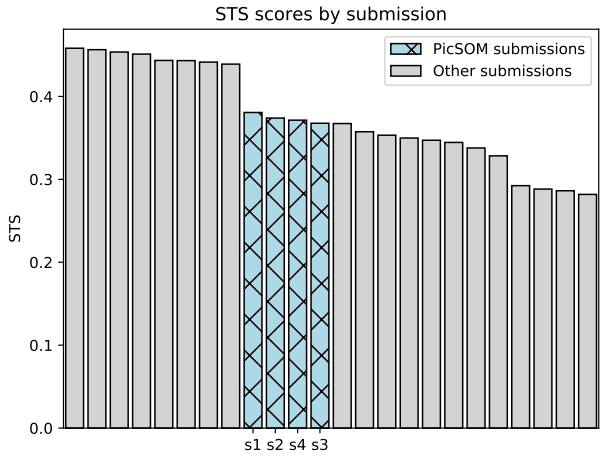

Fig. 5. STS results of our group and others.

[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] 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.

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

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. Technical report, Microsoft Research, 2015.

[7] Kensho Hara, Hirokatsu Kataoka Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

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

[9] Markus Koskela and Jorma Laaksonen. Convolutional network features for scene recognition. In *Proceedings of the 22nd ACM International Conference on Multimedia*, Orlando, FL, USA, November 2014.

[10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv*, abs/1409.4842, 2014.

[11] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3169–3176. IEEE Computer Society, 2011.

[12] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 776–780. IEEE, 2017.

[13] Qiuqiang Kong, Yong Xu, Wenwu Wang, and Mark D Plumbley. Audio set classification with attention model: A probabilistic perspective. *arXiv preprint arXiv:1711.00927*, 2017.

[14] Changsong Yu, Karim Said Barsim, Qiuqiang Kong, and Bin Yang. Multi-level attention model for weakly supervised audio classification. *arXiv preprint arXiv:1803.02353*, 2018.

[15] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv:1705.06950*, 2017.

[16] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv:1804.03209*, 2018.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

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