Extended Self-Critical Pipeline for Transforming Videos to Text (VTT Task 2021) – Team: MMCUniAugsburg

Philipp Harzig
Multimedia Computing and Computer Vision Lab
University of Augsburg
Augsburg, Germany
philipp.harzig@uni-a.de

Moritz Einfalt
Multimedia Computing and Computer Vision Lab
University of Augsburg
Augsburg, Germany
moritz.einfalt@uni-a.de

Katja Ludwig
Multimedia Computing and Computer Vision Lab
University of Augsburg
Augsburg, Germany
katja.ludwig@uni-a.de

Rainer Lienhart
Multimedia Computing and Computer Vision Lab
University of Augsburg
Augsburg, Germany
rainer.lienhart@uni-a.de

Abstract—The Multimedia and computer Vision Lab of the University of Augsburg participated in the VTT task only. We use the VATEx [1] and TRECVID-VTT [2] datasets for training our VTT models. We base our model on the Transformer [3] approach for both of our submitted runs, i.e., for run 2021-01. For our second model (2021-02), we adapt the X-Linear Attention Networks for Image Captioning [4] which does not yield the desired bump in scores. For both models, we train on the complete VATEx dataset and 90% of the TRECVID-VTT dataset for pretraining while using the remaining 10% for validation.

We finetune both models with self-critical sequence training [5], which boosts the validation performance significantly. Overall, we find that training a Video-to-Text system on traditional Image Captioning pipelines [6] delivers very poor performance. When switching to a Transformer-based architecture our results greatly improve and the generated captions match better with the corresponding video (see Figure 3).

I. INTRODUCTION

In this notebook paper, we present our Video-to-Text model, which allows to create descriptions for arbitrary videos. Our model is inspired by the classical Transformer [3] approach.

II. MODEL

A. Preprocessing of Videos

**Single Images.** In order to process the videos in our model, we first need to extract single frames. We use ffmpeg for extracting every frame of each video of the respective dataset. We use ResNet-101 [7] to compute features for the extracted frames. More specifically, we resize the input images to $224 \times 224$ and use the average pooled features with dimension $R^{2048}$.

**3D Features.** We additionally extract features with the Inflated 3D ConvNet (3D) [8] similar to frame-level features. Instead of forwarding frame images through the ResNet-101 V2 network, we extract video clip features with the RGB-I3D pretrained on the Kinetics Human Action Video dataset [9].

**Audio features.** We take the audio of the video, resample it to 16 kHz and extract features with the VGGish [10] network. If no audio stream for a video is existent, we create a dummy feature vector with all zeros.

B. Preprocessing of Tokens

In contrast to our 2020 submission, we do not employ a default tokenizer, but we use the WordPiece Tokenizer [11] to generate the tokens. We load pretrained embedding weights [1] from the BERT\textsubscript{SMALL} model.

C. Model

An overview of our model architecture is depicted in Figure 1. In comparison with the original Transformer [3] architecture, we changed the encoder part to accept image features instead of embedded words. That is, we exchanged the sentence encoder with a video encoder. More specifically, we replaced the input embedding with an image embedding, which is standard practice in common image captioning models [6]. An image embedding layer embeds the image features into the desired embedding space. In our model, we use ResNet-101 features $\in R^{2048}$ and embed them into the encoder space with dimension $d_{\text{model}} = 512$. Additionally, we concatenate audio features extracted by the VGGish [10] network. We use a separate embedding layer for the audio features.

https://tfhub.dev/google/small_bert/bert_uncased_L-8_H-512_A-8/1
Fig. 1: Our model architecture was slightly modified from the original Transformer [3] to allow vision and audio frames as input to the encoder blocks. Model image inspired by [3] and modified to match our architecture.

We also use positional encoding to encode the order of every single frames in the video. As the Transformer architecture does not care about the order of the input, i.e., every frame can influence every other frame in the same way, we need to explicitly tell the encoder the frame number. Similar to the original paper, we use a positional encoding to encode the frame number, which we add on top of the embedded image features. The sequence length of image features or I3D features is varying. Therefore, we cannot simply concatenate vision and audio features as the added positional encoding may signal the encoder that it receives a vision feature as input when in reality it is an audio feature. We assume a fixed starting position for all audio features which we set to 300 (i.e., there are no more than 300 image frames for any video in the dataset). Finally, we add positional encodings for indexes [300, 301, ...] on top of the embedded audio features.

In the encoder, we make use of the memory-augmented encoding [12], which encodes multi-level visual relationships with a priori knowledge. In the original work, Cornia et al. use a persistent, learnable memory vector which is concatenated to the key and value of the self-attention blocks of the Transformer. These memory vectors allow to encode persistent a-priori knowledge about relationships between image regions. In contrast to the original work, we work with video sequences instead of still images with regions. Adapted to our architecture, the memory vector encodes a-priori knowledge about relationships between frames in a given video. We did not change the architecture of the decoder block (see Section III).

### III. DATASOURCES

We use two datasets for training our models, which are described below. Additionally, we show some dataset statistics in Table I. Note that we also trained on AC-GIF [13] and MSR-VTT [14] in last year’s challenge. However, we found that VATEX delivers far better and more consistent results.

#### A. TRECVID-VTT

We use the official TRECVID-VTT dataset [15] which contains videos from the TRECVID VTT from 2016-2019. We only use the Twitter Vine subset of videos. In total, this subset contains 6,475 videos from which we use 5,971 available videos with 22,547 captions. In all our experiments we train on 90% and validate the model on 10% of the videos.

#### B. VATEX

We additionally train on the VATEX Dataset [1] to boost the performance of our final models. We trained our last-year models on MSR-VTT [14], but found that the MSR-VTT dataset is less representative than the VATEX dataset and results in lower scores. The VATEX dataset is split into 4 sets, i.e., the training set, the validation set, the public test set and the private test set. The VATEX dataset comes with 10 English and 10 Chinese captions per video clip. Most video clips have a length of 10 s.

### IV. MODEL CONFIGURATIONS

We submitted two models for the Video-to-Text (VTT) task. Both of our models are pretrained on a merged dataset and then finetuned on the merged dataset as well.
For our primary model (cf. 2021-01), we first train a base model on the full MSR-VTT dataset and 90% of the TRECVID-VTT dataset. We select the model by employing an early-stopping strategy on the CIDEr score of the remaining 10% of the TRECVID-VTT dataset. In contrast to last year’s primary model (2020-01), we train on 13D features instead of ResNet features. Furthermore, we add audio features for the VATEX part of our training set (the TRECVID dataset does not come with audio) and train the model with a modified learning rate schedule. For finetuning, we use the base model and train it on the same dataset, but enable self-critical sequence learning \cite{DBLP:journals/corr/abs-1909-05858} with a constant learning rate $\eta = 5 \cdot 10^{-6}$. Here, we calculate the CIDEr scores for a baseline caption $\hat{w}$ and sample 5 additional captions $w^s$, respectively. Subsequently, we can baseline the reward of the sampled captions by subtracting the CIDEr score for the baseline caption. As a consequence, sampled captions with a higher CIDEr score than the baseline caption get a positive reward and vice versa. The gradient of the loss function can be approximated as follows:

$$\nabla_{\theta} L(\theta) \approx -(r(w^s) - r(\hat{w})) \nabla_{\theta} \log p_\theta(w^s).$$

Each word will be weighted according to its log probability and $r(\cdot)$ is the reward function. $\theta$ are the parameters of the network and define a policy $p_\theta$. For our final models, we additionally optimize the BLEU-4 metric. Therefore, our reward function becomes

$$r(\cdot) = \lambda_{\text{CIDEr}} \cdot r_{\text{CIDEr}}(\cdot) + \lambda_{\text{BLEU-4}} \cdot r_{\text{BLEU-4}}(\cdot),$$

where $\lambda$ is a weight for the corresponding metric.

Our second model (cf. 2021-02) is trained similarly, except we implemented X-Linear Attention \cite{DBLP:journals/corr/abs-1703-01460}. We fine-tune this model in the same way as model 2021-01. In Table III, we present the number of training samples used for training the base and finetuned models.

Our models use 8 encoder and 8 decoder blocks. We use 8 attention heads and a model dimension of $d_{\text{model}} = 512$. For the position-wise feed-forward networks, we set $d_{\text{ff}} = 2048$ as the inner-layer dimensionality. We use a memory-vector size of $d_{\text{memory}} = 64$. The primary model use the default BERT subtoken vocabulary with 30522 subword tokens. It does not use complete words for the vocabulary, but tries to build words from subwords, i.e., it splits words into subwords if a word is not in the initial dictionary.

### Table II: Submitted models (in bold) and their respective validation scores.

| Model | epochs | ft | Features | [mv] | Vocabulary | lr Schedule | B-4 | C | M |
|-------|--------|----|----------|-----|------------|-------------|-----|---|---|
| 2020-01-ft | 25 | ✓ | CNN | 64 | Default | Default | 0.076 | 0.176 | 0.116 |
| 2020-02-ft | 1 | ✓ | CNN | 64 | Default | Default | 0.061 | 0.151 | 0.110 |
| 2021-01 | 43 | — | 13D | 64 | WP-BERT | sgd | 0.101 | 0.249 | 0.249 |
| 2021-01-ft | 3 | ✓ | 13D | 64 | WP-BERT | $5 \cdot 10^{-6}$ | 0.142 | 0.308 | 0.160 |
| 2021-02-ft | 15 | — | 13D | 64 | WP-BERT | sgd | 0.109 | 0.226 | 0.137 |
| 2021-02-ft | 0.33 | ✓ | 13D | 64 | WP-BERT | $5 \cdot 10^{-6}$ | 0.115 | 0.244 | 0.142 |

### Table III: Data source used for training our models. We also depict the total number of training and validation samples used.

| Model: Data sources | # train samples | # val samples |
|---------------------|-----------------|---------------|
| 1: MSR-VTT + 90% VATEX | 273,314 | 3602 |

### V. Training

We train our models in a multi GPU setting, i.e., we train the model on 4 NVIDIA Tesla A100 GPUs simultaneously. We use a batch size of 128 per GPU, resulting in an effective batch size of 512. We use the Adam \cite{DBLP:journals/corr/KingmaB14} optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. Similar to \cite{DBLP:journals/corr/abs-1909-05858}, we train with a variable learning rate $\eta$ over the course of the training (schedule-default). However, we combine the original learning rate with SGDR (Stochastic Gradient Descent with Warm Restarts, schedule-sgdr) \cite{DBLP:journals/corr/abs-2103-05495} learning rate schedule. We plot the SGD learning rate schedule combined with a warm-up phase in Figure 2. In contrast to the original Transformer architecture, we used $w = 10,000$ for the number of warm-up steps.

For the base model of our primary model (2021-01), we observed the best validation performance on TRECVID-VTT after 43 epochs with a CIDEr score of 0.249. We used this model to finetune with self-critical sequence learning (2021-01-ft). In doing so, we significantly improved the scores as
TABLE IV: Submitted models and their respective performance on the unseen test dataset. Models with (1977) denote performance on the extended unreleased test set from 2021. The other models are comparable to last year’s results (seen in the first two rows).

| Model                  | BLEU  | CIDEr | CIDEr-D | METEOR |
|-----------------------|-------|-------|---------|--------|
| 2020-01-ft            | 0.018 | 0.140 | 0.064   | 0.202  |
| 2020-02-ft            | 0.011 | 0.136 | 0.060   | 0.204  |
| 2021-01-ft            | 0.022 | 0.315 | 0.180   | 0.292  |
| 2021-02-ft            | 0.015 | 0.247 | 0.137   | 0.260  |
| 2021-01-ft (1977)     | 0.022 | 0.313 | 0.178   | 0.292  |
| 2021-02-ft (1977)     | 0.015 | 0.246 | 0.137   | 0.260  |

For the TRECVID 2020 workshop [18], we submitted captions generated on the provided test videos (1,700) for basic transformer models (see last year’s notebook paper for details [19]). In a nutshell, we implemented a vanilla Transformer that accepts only image features from a ResNet with support for memory-augmented vectors [12]. For this year’s workshop [15], we extended our model to support audio frames, features from the Inflated 3D ConvNet (I3D) [8] and self-critical sequence training. We submitted captions generated by our two finetunes models (2021-01-ft and 2021-02-ft).

These captions were evaluated by the workshops organizers. Compared to our validation set scores, the evaluation on the test set yields worse results as can be seen in Table IV. Especially, the BLEU score is much lower on the test data than on the evaluation data.

We depict five videos and their generated caption in Figure 3. We see that for the first three videos their generated captions from the model 2021-01-ft match the video content quite good. The first video description is correct. Only if we look closer, we see that one person is giving the other person a massage.

In the second video, our model detects correctly that we see a football field and a group of people which are indeed playing football. In the third video, our model detects a young woman who looks into a camera. However, it fails to detect that the woman is cheerleading in the background of some apples. For the fourth video the model correctly detects a man. But the man is not reading a book, rather he is showing an ad in front of his notebook. In the fifth video, the model detects that there is a basketball game going on. However, it shows the audience rather than basketball players sitting on a bench. But in the first frame, we see a basketball play, hence, the model may take this as a hint for generating the sentence.

VII. Conclusion

In this notebook paper, we presented our VTT model based on a Transformer [3] architecture. By extracting features for every frame of the videos, we were able to adapt the Transformer architecture to use videos in the encoder block. Furthermore, we extracted features with the 3D network that is may extract contextual information related to the time-axis of the video. In addition, we modified the Multi-Head Attention of the encoder to use memory vectors similar to [12] which allow to memorize a priori knowledge about relationships between video frames. Finally, we finetune our models with self-critical sequence learning that directly optimizes the CIDEr and BLEU-4 metrics. Thus, we generate captions that describe video contents (see Figure 3). However, as not all objects and circumstances of the videos are detected and described correctly, we want to address object and relationship detection in future work.
[12] M. Cornia, M. Stefanini, L. Baraldi, and R. Cucchiara, “Meshed-memory
transformer for image captioning,” in Proceedings of the IEEE/CVF
Conference on Computer Vision and Pattern Recognition, pp. 10578–
10587, 2020.
[13] Y. Pan, Y. Li, J. Luo, J. Xu, T. Yao, and T. Mei, “Auto-captions on gif:
A large-scale video-sentence dataset for vision-language pre-training,”
arXiv preprint arXiv:2007.02375, 2020.
[14] J. Xu, T. Mei, T. Yao, and Y. Rui, “Msv-vtt: A large video description
dataset for bridging video and language,” in Proceedings of the IEEE
conference on computer vision and pattern recognition, pp. 5288–5296,
2016.
[15] G. Awad, A. A. Butt, K. Curtis, J. Fiscus, A. Godil, Y. Lee, A. Delgado,
J. Zhang, E. Godard, B. Chocot, L. Diduch, J. Liu, Y. Graham, G. J. F.
Jones, and G. Quénot, “Evaluating multiple video understanding and
retrieval tasks at trecvid 2021,” in Proceedings of TRECVID 2021; NIST,
USA, 2021.
[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,”
arXiv preprint arXiv:1412.6980, 2014.
[17] I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with
warm restarts,” arXiv preprint arXiv:1608.03983, 2016.
[18] G. Awad, A. A. Butt, K. Curtis, Y. Lee, J. Fiscus, A. Godil, A. Delgado,
J. Zhang, E. Godard, L. Diduch, J. Liu, A. F. Smeaton, Y. Graham,
G. J. F. Jones, W. Kraaij, and G. Quénot, “Trecvid 2020: comprehensive
campaign for evaluating video retrieval tasks across multiple application
domains,” in Proceedings of TRECVID 2020, NIST, USA, 2020.
[19] P. Harzig, M. Einfalt, K. Ludwig, and R. Lienhart, “Transforming
videos to text (VTT task) team: Mmcuniaugsburg,” in 2020 TREC
Video Retrieval Evaluation, TRECVID 2020, Gaithersburg, MD, USA,
December 8-11, 2020 (G. Awad, A. A. Butt, K. Curtis, J. G. Fiscus,
A. Godil, Y. Lee, A. Delgado, J. Zhang, E. Godard, B. Chocot, L. L.
Diduch, J. Liu, A. F. Smeaton, Y. Graham, G. J. F. Jones, W. Kraaij,
and G. Quénot, eds.), National Institute of Standards and Technology
(NIST), 2020.
Fig. 3: Five videos from the validation dataset and the corresponding captions generated by our models.