M.S. THESIS

Transitional Adaptation of Pretrained Models for Visual Storytelling

시각적 스토리텔링을 위한 사전 훈련된 언어 모델의 전이 적용

BY

정지완

FEBRUARY 2023

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

Transitional Adaptation of Pretrained Models for Visual Storytelling

시각적 스토리텔링을 위한 사전 훈련된 언어 모델의 전이 적용

BY

정지완

FEBRUARY 2023

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Transitional Adaptation of Pretrained Models for Visual Storytelling

시각적 스토리텔링을 위한 사전 훈련된 언어 모델의 전이 적용

지도교수 김 건 희

이 논문을 공학석사 학위논문으로 제출함

2022 년 11 월

서울대학교 대학원

컴퓨터 공학부

정 지 완

정지완의 공학석사 학위논문을 인준함

2022 년 12 월

위 원 장	문병로
부위원장	이재욱
위 원	김건희

Abstract

Previous models for vision-to-language generation tasks usually pretrain a visual encoder and a language generator in the respective domains and jointly finetune them with the target task. However, this direct transfer practice may suffer from the discord between visual specificity and language fluency since they are often separately trained from large corpora of visual and text data with no common ground. In this work, we claim that a transitional adaptation task is required between pretraining and finetuning to harmonize the visual encoder and the language model for challenging downstream target tasks like visual storytelling. We propose a novel approach named *Transitional Adaptation of Pretrained Model (TAPM)* that adapts the multi-modal modules to each other with a simpler alignment task between visual inputs only with no need for text labels. Through extensive experiments, we show that the adaptation step significantly improves the performance of multiple language models for sequential video and image captioning tasks. We achieve new state-of-the-art performance on both language metrics and human evaluation in the multi-sentence description task of LSMDC 2019 [1] and the image storytelling task of VIST [2]. Our experiments reveal that this improvement in caption quality does not depend on the specific choice of language models.

Keywords: artificial intelligence, multomodal learning, visual storytelling
Student Number: 2019-29077

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Most models for vision-to-language generation tasks consist of a visual encoder to extract visual information from input images or videos, a language model to generate text sentences, and a mechanism to weld the two modules into one harmonized architecture. For example, recent models for visual captioning [3, 4] adopt a pretrained visual encoder and a pretrained language generator and then optimize the target cross-modal generation objective with the downstream datasets [5, 6, 7, 8, 9, 10]. In this process, however, no transitional adaptation step has proposed to match the potentially substantial differences between the information stored in the visual encoder and the language generator, as they are separately trained from large sets of visual and text data with no common ground (*e.g.*images from ImageNet and text from Wikipedia).

This work is motivated by that this direct transfer of pretrained models to a downstream task may suffer from the dissonance between visual specificity and language fluency. For example, finetuning pretrained language models on another target task may result in catastrophic forgetting of the language gener-

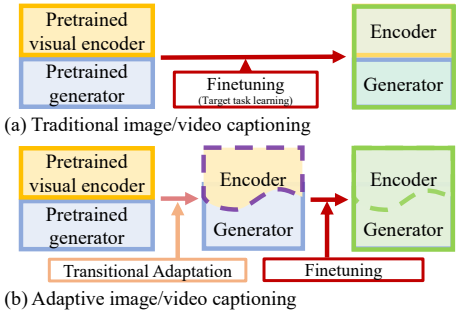


Figure 1.1 Comparison between existing captioning models and our Transitional Adaptation of Pretrained Model (TAPM). (a) Previous captioning models start from a pretrained visual encoder and a language generator and then directly finetune with the downstream datasets. (b) TAPM includes a simple pretext task as an adaptation process that harmonizes the generator with the visual encoder before optimizing the target objective.

ation capability [11, 12]. Moreover, existing captioning models have often been criticized for not sufficiently conditioning on the visual context and thus lack visual discriminability [13, 14].

Considering the potentially vast gap between the nature of the information stored in the visual encoder and the language decoder, it would be difficult for them to work in harmony at once for another challenging objective of vision-to-language generation. In this light, we believe a simpler objective dedicated to improving coordination between the two separately pretrained models could help the model get prepared for the target objective eventually better and faster.

Therefore, we present *Transitional Adaptation of Pretrained Model* (TAPM) for visual storytelling as the first approach that proposes an explicit visual adaptation step to harmonize the visual encoder with the pretrained language models as depicted in Fig. 1.1. Our adaptation step can be trained with only visual in-

puts, such as images or videos with no text label. We outline the contributions of this work as follows:

1. Our work is the first attempt to demonstrate an auxiliary adaptation loss's effectiveness in welding a visual encoder with a pretrained language model. By extensive experiments, we show that this additional adaptation between pretraining and finetuning consistently improves the captioning quality of various language models such as GPT-2 [15], XLM [16], and QRNN [17].

2. We present the sequential coherence loss that can adapt the language generator using only sequential video/image inputs with no text label. We also introduce two recipes critical to TAPM's success: (i) using the language model outputs for adaptation training and (ii) using the split-training process.

3. We evaluate TAPM in two storytelling tasks: sequential video captioning in the LSMDC 2019 [1] and sequential image captioning in VIST [2]. TAPM achieves new state-of-the-art performance in both tasks in terms of automatic language metrics and human evaluation.

Chapter 2

Related Work

2.1 Visual Storytelling

Unlike direct and literal descriptions, visual storytelling aims to generate a more figurative and consistent narrative for consecutive images or videos [2]. Some earlier works [18, 19] explore the summarization of long videos into the storyline representation. Park *et al.* [20, 21] integrate an entity-based local coherence model to generate a coherent flow of multiple sentences for a photo album. Fan *et al.* [22] use a shorter prompt as the intermediate representation. Jain *et al.* [23] combine SMTs and RNNs to merge independent descriptions into a coherent story. Huang *et al.* [24] propose a two-level hierarchical RL-based decoder to plan a semantic topic first and then generate consistent sentences. Tang *et al.* [25] employ an attribute-based hierarchical decoder to create paragraphs using policy gradient with word-level rewards and adversarial training. Fan *et al.* [26] exploit a predicate-argument structure of the text to build coherent stories. Gella *et al.* [27] introduce the VideoStory dataset for generating stories

from social media videos. AdvInf [28] uses adversarial inference and MART [29] memory augmented transformer to generate paragraph-level captions.

Most previous works on visual storytelling require both visual encoder and language generator. Our work is orthogonally applicable to these approaches to better adapt the language decoder for visual context before training the models with the main vision-to-language objective, including Reinforcement and adversarial learning.

2.2 Auxiliary Losses for Captioning

Autoregressive language models trained with cross-entropy often suffer from exposure bias [30]. Several works on captioning have leveraged reinforcement learning by using rewards as auxiliary loss signals to ameliorate this bias. Zhang *et al.*[31] directly optimize language quality metrics with an actor-critic framework. Liu *et al.*[32] optimize a linear combination of language metrics using Monte Carlo rollouts. SCST [33] improves the REINFORCE algorithm to correctly normalize external rewards using the test-time inference algorithm's output. Ren *et al.*[34] use the embedding similarity between generated sentences and image features as the reward. These reinforcement learning approaches have been extended to the video captioning problem [35, 36]. While reinforcement learning can help training non-differentiable objectives, it is known to be unstable [37]. Other types of auxiliary losses have also been adopted for captioning problems. Ma *et al.*[38] employ the cyclic reconstruction to enforce the localization of each word in an image. Zhou *et al.*[39] add visual grounding supervision to enhance the sentence generation quality. HINT [40] learns to match the attention map to human attention for grounded image captioning. VideoBERT [41] extends the text-based BERT to build bidirectional modeling

between videos and captions. Compared to previous work, our work does not require additional visual caption data since it takes self-supervision losses with only sequential visual inputs.

2.3 Pretrained Models for Vision-to-Language Tasks

Recently, many works have demonstrated the power of self-supervision based representation learning in cross-modal settings. LXMERT [42] and ViLBERT [43] pretrain two-stream transformers on various tasks including masked cross-modal language model (LM) objectives. LXMERT is extended later with adaptive sparse attention [44]. VisualBERT [45] and VL-BERT [46] uses single-stream transformers. CMR [47] models the relevance between the textual entities and visual entities. UNITER [48] and Unicoder-VL [49] use object detection based objectives in addition to the masked LM loss. VideoBERT [41] trains a transformer for video-language tasks using vector quantization to categorize videos into discrete tokens. CBT [50] replaces the softmax loss of BERT with noise contrastive estimation.

These approaches aim to learn general representations, and our method adapts the trained representations to the target cross-modal generation tasks. Thus, our model is orthogonal to the aforementioned self-supervised representations and consistently improves the final performance even with the self-supervised representation. Furthermore, they often use the masked cross-modal objectives that require both visual data and associated sentences (with blanks); contrarily, our method does not require text data at all for self-supervision.

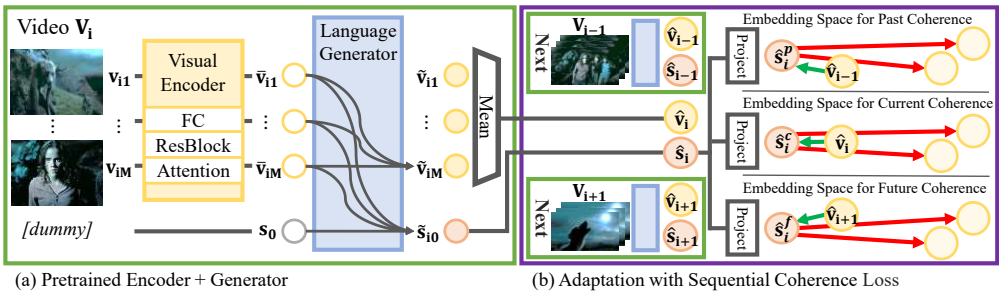
Chapter 3

Approach

We demonstrate our TAPM approach in visual storytelling tasks, which are a sequential extension of visual captioning. Its goal is to generate coherent C sentences for C visual inputs of video clips or images. We henceforth explain our model in the context of sequential video captioning because it subsumes sequential image captioning.

Fig. 3.1 illustrates the overall architecture, which consists of the visual encoder (section 3.1) and the language generator (section 3.2). We train the visual encoder and the language generator with the adaptation loss before finetuning them with the downstream captioning tasks (section 3.3). We employ the sequential coherence loss as the adaptation loss to encourage both distinctiveness and coherence in sequential captions. These losses are applied to the language model outputs in order to update the visual encoder in accordance with the language model (section 3.5). Finally, the encoder and the generator are trained with the target objective of visual storytelling.

For overall training, we use a split-training approach (section 3.5) that helps



(a) Pretrained Encoder + Generator (b) Adaptation with Sequential Coherence Loss

Figure 3.1 Illustration of the proposed TAPM framework. (a) TAPM harmonizes a pretrained visual encoder (section 3.1) with a pretrained language generator (section 3.2) to improve a target captioning task. In the adaptation phase, the model takes only videos (or images) as the input. Given a video, the language generator builds the corresponding video embedding ($\widehat{\mathbf{v}}_i$) and text embedding ($\widehat{\mathbf{s}}_i$) per each video. (b) We introduce sequential coherence loss to improve temporal coherence in visual storytelling tasks. We first use the respective FC layers (f^p, f^c and f^f) to project the text embedding ($\widehat{\mathbf{s}}_i$) into the past, current, and future visual space. We then encourage the respective past, current, and future text embedding to be closer to their corresponding visual representations (**Pull (Green arrows)**) than the other visual representations (**Push (Red arrows)**).

the decoder retain language generation capability. Since the adaptation loss is not a generation loss, it may degrade the language understanding of the pretrained language model. Hence, split-training fixes the language generator weights during the adaptation phase.

3.1 The Visual Encoder

Given a video clip, we utilize pretrained feature extractors to extract vector feature v_{ij} for each frame j. The set of pretrained feature extractors varies depending on datasets and will be covered in section 4.1. We then reduce the vectors to M segments by mean-pooling them over temporal dimension.

With the extracted features of a video clip $\mathbf{V}_i = \{\mathbf{v}_{i1}, \ldots, \mathbf{v}_{iM}\}$ as inputs, the visual encoder builds task-specific representations $\overline{\mathbf{V}}_i = \{\overline{\mathbf{v}}_{i1}, \ldots, \overline{\mathbf{v}}_{iM}\}$. Our visual encoder consists of two fully connected (FC) layers followed by Leaky ReLU [51], three layers of residual blocks, and a final self-attention layer [52]. A residual block consists of two FC layers and a ReLU activation [53]. After processing the visual inputs, we mean-pool the previous and next frame representation and concatenate them to the current representation to encode the context information.

3.2 The Language Generator

For the language generator, one can use any language model. In our experiments, it is implemented by (but not confined to) GPT-2 [15], GPT [54], XLM [16], QRNN [17], and LSTM [55]. We use GPT-2-small [15] pretrained on a corpus dataset of 8 million web pages as the default generator due to its best performance among other language model s. We will report the results of other language models in section 4.3.

3.3 Adaptation training

We train the visual encoder with a simple auxiliary objective to harmonize it with the language generator in the adaptation phase. Here, we describe how to encode the visual and text representations for calculating the adaptation

loss given the video inputs. The adaptation loss for visual storytelling will be discussed in the next section.

The language generator takes the task-specific representation $\overline{\mathbf{V}}_i$ from the visual encoder as inputs and generates the contextualized representation for visual $\widetilde{\mathbf{V}}_i$ and text $\widehat{\mathbf{s}}_i$. Specifically, the input $\mathbf{X}_i$ to the generator is

$$\mathbf{X}_i = [\overline{\mathbf{V}}_i, [sep], [dummy]], \tag{3.1}$$

where $[sep]$ and $[dummy]$ are respectively separation and dummy tokens. Remind that $\overline{\mathbf{V}}_i$ is a sequence of vectors with the number of segments M. Then the generator outputs

$$\widetilde{\mathbf{X}}_i = [\widetilde{\mathbf{V}}_i, [sep], \widehat{\mathbf{s}}_i], \tag{3.2}$$

where $\widehat{\mathbf{s}}_i$ can be regarded as the text representation that the generator predicts for a sequential video input $\overline{\mathbf{V}}_i$.

Finally, the visual representation $\widehat{\mathbf{v}}_i$ is obtained by mean-pooling the sequence representation $\widetilde{\mathbf{V}}_i$ to a single vector. Note that the adaptation step does not use the caption label but inputs a dummy token into the generator to obtain text information. Thus, we can train the language generator with video-only datasets. While the dummy token can be arbitrarily selected from the pretrained vocabulary, we resort to the start-of-sentence token for all reported experiments. As will be shown in Table 4.3, TAPM with the dummy token performs comparably with the ground truth captions.

3.4 The Sequential Coherence Loss

Visual storytelling is the problem of generating expressive, aligned, and coherent captions from a sequence of semantically connected visual inputs (*e.g.*videos

or photo streams). Consecutive images or video clips tend to share common backgrounds, characters, and objects.

This closeness makes those visual features similar, and as a result, the captions generated from them overlap one another. To make consecutive captions not too overlapped but still coherent, we introduce the *sequential coherence loss* to build text representation of each visual input.

The sequential coherence loss enforces the text representation of a clip to predict the visual representations within its closed neighborhood well. We divide the sequential coherence loss into three parts of the past, current, and future matching loss for a better explanation. First, the past matching loss projects the text representation $\widehat{\mathbf{s}}_i$ of video i by an FC layer f^p and makes it closer to the visual representation $\widehat{\mathbf{v}}_{i-1}$ of the previous video $i-1$ than the other videos, as in Figure 3.1. Second, the future matching loss is almost identical to the past matching loss except that it projects $\widehat{\mathbf{s}}_i$ with a different FC layer f^f and matches with the next visual representation $\widehat{\mathbf{v}}_{i+1}$. Finally, the current matching loss matches the current visual representation $\widehat{\mathbf{v}}_i$ with $\widehat{\mathbf{s}}_i$ through an FC layer f^c. They are similar in that the text representation is projected in the past, future, current visual space by an FC layer and then drives the embeddings of correct matches closer (*pull*) and those of wrong matches farther away from each other (*push*).

To implement this notion, we employ margin ranking losses between correct matches and other wrong ones. The final loss is the sum of the past, current,

and future matching losses as follows:

$$
\begin{aligned}
L_i = & \sum_{j \neq i-1} \max(0, 1 + \widehat{\mathbf{v}}_j * f^p(\widehat{\mathbf{s}}_i) - \widehat{\mathbf{v}}_{i-1} * f^p(\widehat{\mathbf{s}}_i)) \\
& + \sum_{j \neq i} \max(0, 1 + \widehat{\mathbf{v}}_j * f^c(\widehat{\mathbf{s}}_i) - \widehat{\mathbf{v}}_i * f^c(\widehat{\mathbf{s}}_i)) \\
& + \sum_{j \neq i+1} \max(0, 1 + \widehat{\mathbf{v}}_j * f^f(\widehat{\mathbf{s}}_i) - \widehat{\mathbf{v}}_{i+1} * f^f(\widehat{\mathbf{s}}_i)),
\end{aligned} \tag{3.3}
$$

where the operator $*$ denotes the cosine similarity, and j indicates the index for wrong matches.

3.5 Training with the adaptation Loss

Use of Language Model Outputs. As described in the previous sections, our adaptation losses use the visual representation processed with the language model rather than the visual encoder outputs. Using the language model outputs enables the adaptation losses to update the visual encoder in accordance with the language model. On the other hand, using the encoder outputs would update the visual encoder in isolation. In Table 4.3, we will show that adaptation using the encoder outputs (TAPM+VisualA) does not improve upon the baseline (TAPM-A), while adaptation on the language model outputs (TAPM) does. Thus, this scheme is crucial to train the visual encoder in coordination with the language model to benefit the target task.

Split-Training. We split the training process into two phases: the adaptation loss step and the caption generation loss step. First, the visual encoder is updated for a given number of epochs by the adaptation loss, while the text encoder and the language generator are fixed. Then, we jointly update all the components with the generation loss. By splitting the training process, we give the model a chance to optimize the simpler adaptation task long enough before

being presented with the harder generation objective. Fixing the language generator during the adaptation loss step prevents catastrophic forgetting of the language generation capability. Our ablation study in section 4.3 confirms that the split training leads to significant performance gains.

3.6 Finetuning and Inference

Target-Task Training. After adaptation training, we can finetune the language generator to the downstream captioning task with ground-truth data, where we input C pairs of video clips (or images) and text descriptions one by one: $\{\mathbf{V}_1, \mathbf{S}_1, \ldots, \mathbf{V}_C, \mathbf{S}_C\}$. We use the teacher forcing as the training scheme with the cross-entropy loss:

$$L_i^G = -\sum_{l=1}^{L}\sum_{v=1}^{V} y_{il}^v \log p_{il}^v, \tag{3.4}$$

where $v \in \{1, \ldots, V\}$ is the vocabulary index, p_{il} is the prediction probability for the l-th token in $\mathbf{S}_i$, and y_{il} is the ground truth label. Finally, the language model head generates a caption output, consisting of a single FC layer that maps each vector of the language model outputs $\widetilde{\mathbf{S}}_i$ to a softmax layer to obtain the word probability p_i of each token over vocabulary.

Cross-Modal Generation. At inference, our goal is to generate a coherent sequence of C sentences for a visual test sample $\{\mathbf{V}_1, \ldots, \mathbf{V}_C\}$. We first use the visual encoder to build the visual embedding $\overline{\mathbf{V}}_i$ for $i = 1, \ldots, C$. We then generate each sentence auto-regressively using the finetuned language generator. In the decoding step l for $\overline{\mathbf{V}}_i$ (*i.e.*, the i-th output sentence is generated up to $l-1$ words), the input to the language generator is $[\overline{\mathbf{V}}_i, [sep], [dummy], \mathbf{s}_{i1}, \ldots, \mathbf{s}_{il-1}]$. We can obtain the word probability p_{il} with the output of the language generator $\widetilde{\mathbf{s}}_{il-1}$, and finally select the next word $\mathbf{s}_{il} = \arg\max_v p_{il}$. We iterate this

until the end-of-sentence token [*eos*] appears, or the output sentence reaches the predefined maximum length.

Chapter 4

Experiments

We evaluate the TAPM approach in two visual storytelling tasks: sequential video captioning in LSMDC 2019 [1] and image captioning in VIST [2]. For both tasks, we achieve new state-of-the-art performance in both automatic evaluation (section 4.2) and human evaluation (section 4.4). We also perform various empirical analyses of our TAPM across various language models (section 4.3). Furthermore, we demonstrate that TAPM can benefit from additional visual-only datasets. TAPM is also extendable to other visual-linguistic tasks such as VQA and cross-modal retrieval, as shown in Appendix.

4.1 Experimental Setup

Datasets. The Multi-Sentence Description of LSMDC 2019 [1] is the task of generating consecutive captions for multiple short movie clips. For a given set of five clips, the model generates five sentences maintaining logical and contextual consistency. The dataset contains 128,085 clips from 200 movies and has four

splits; 20,283 training, 1,486 validation, 2,018 public test, and 1,923 blind test samples. Following the challenge protocol, we combine the train and validation split as training data. The official performance is evaluated on the blind test split hidden from participants, while ablation studies are conducted on the public test split.

VIST [2] is a visual storytelling dataset, including 10,117 Flickr albums with 210,819 unique photos. Each story of VIST contains five sequential images with the corresponding captions. We use the SIS (Stories of Images in Sequence) tier that has more storytelling elements. Ignoring broken images, we use 40,071 training, 4,988 validation, and 5,050 testing story samples. In all experiments, we use the training/test split of [2, 56, 57]. As in [57], we evaluate at the album level by allowing only one story candidate per album regardless of photo sequences.

Data Preprocessing. For LSMDC 2019, we use the ResNet [58] pretrained on ImageNet [59] to extract frame features as in [33, 60]. For the challenge submission and human evaluation, we add the I3D feature [61] pretrained on Kinetics [62] as done in the official baseline [28]. We equally segment a video clip into three subshots and represent each by mean-pooling the features of frames. For the challenge results, we use set level evaluation by concatenating all captions within a set of 5 clips as dictated by the organizers. For ablation study, we use individual sentence level evaluation to compare with non-sequential generation models fairly. For VIST, we use the same ResNet extractor and additional features of object bounding boxes from Faster R-CNN [63] pretrained on Visual Genome [64]. We choose at most 20 objects with the highest likelihood per image from the R-CNN [63] detection results. After processing each feature through the visual encoder, we concatenate all features along the temporal dimension with a special separator token between them. We tokenize and numericalize

the text using Byte Pair Encoding [65] for pretrained language models while using the whitespace tokenizer for the no pretrained models. In VIST, we use the default tokenizer to re-tokenize our generated samples for evaluation. We generate each caption with beam search up to 30 tokens and cut every ground truth sentence to the maximum length of 50 tokens for all experiments.

Metrics. We use three n-gram based metrics to evaluate our approach: CIDEr [66], METEOR [67] and ROUGE-L [68]. CIDEr captures consensus by applying Term Frequency Inverse Document Frequency (TF-IDF) weighting for each n-gram. METEOR scores the sequence matches with explicit alignment at the sentence level. ROUGE-L is a recall-based metric computed with the length of the longest common subsequence. For computing METEOR in VIST, we use the official VIST challenge evaluation code [1]. All the other metric scores are computed with the pycocoevalcap library [2].

Baselines. For LSMDC 2019, we compare our approach with the official baseline [1, 28]. We also adapt XE and AREL models [57] to LSMDC using the official codes. For VIST, we compare TAPM with eight state-of-the-art methods: GLACNet [69], h-attn-rank [56], Contextualize, Show and Tell (CST) [70], BLEU-RL [57], CIDEr-RL [57], GAN [57], AREL [57], StoryAnchor [71], HSRL [24], and INet [72]. The scores for BLEU-RL, CIDEr-RL, and AREL are referred from [57], while the results of GLACNet, CST, StoryAncher, HSRL, and INet are referred from the respective papers. We use XE and AREL as baselines for human evaluation on the VIST dataset. XE shares the architecture of AREL except for the lack of adversarial rewards. We use the publicly available codes for both models.

Hyperparameters. Unless we mention it explicitly, we fix all random seeds

[1] https://github.com/windx0303/VIST-Challenge-NAACL-2018
[2] https://github.com/tylin/coco-caption

Table 4.1 Quantitative results on the LSMDC 2019 [1] public and blind test set. XE and AREL do not report the blind test score because they are not challenge participants. C stands for CIDEr and M for METEOR. All tests are done on the set level.

	Public Test		Blind Test	
Models	C	M	C	M
Official Baseline [28]	7.0	12.0	6.9	11.9
XE [57]	7.2	11.5	-	-
AREL [57]	7.3	11.4	-	-
TAPM (ours)	**10.0**	**12.3**	**8.8**	**12.4**

Table 4.2 Quantitative results on the VIST [2] test set. R stands for ROUGE-L.

Models	C	M	R
Huang et al.[2]	-	31.4	-
h-attn-rank[56]	7.5	34.1	29.5
GLACNet[69]	-	30.1	-
CST[70]	5.1	34.4	29.2
BLEU-RL[57]	8.9	34.6	29.0
CIDEr-RL[57]	8.1	34.9	29.7
GAN[57]	9.1	35.0	29.5
AREL[57]	9.4	35.0	29.5
StoryAnchor[71]	9.9	35.5	30.0
HSRL[24]	10.7	35.2	30.8
INet[72]	10.0	35.6	29.7
TAPM (ours)	**13.8**	**37.2**	**33.1**

to 0. For training, we use Adam optimizer [73] with linear learning rate decay. The learning rate is $5e-5$, which is warmed up for the first 4000 steps. We apply 0.5 dropout on the language generator outputs. In all experiments, we use the batch size of 8. For LSMDC dataset we train the adaptation loss for 5 epochs, whereas we train for 3 epochs in case of VIST dataset. We train all models up to 30 epochs.

Table 4.3 Ablation results of our TAPM model on the LSMDC 2019 public test set and the VIST test set. The evaluations for LSMDC are done on the sentence level.

	LSMDC			VIST		
Models	C	M	R	C	M	R
Baseline[28]	11.90	8.25	-	-	-	-
Baseline+GPT-2[15]	8.65	7.75	19.90	-	-	-
TAPM (ours)	**15.37**	8.41	**20.21**	**8.3**	**34.1**	**30.2**
-A	14.54	8.27	19.89	4.8	33.6	29.9
+Cap	15.29	**8.47**	20.19	6.7	33.8	29.8
+VisualA	14.59	8.37	20.00	4.9	33.0	29.9
-Split	14.28	8.34	19.71	4.5	32.8	29.8
-A+Split	14.01	8.28	19.60	6.5	33.8	30.0

4.2 Quantitative Results

We use OpenAI GPT-2 [15] as our default language generator due to its best performance among other language models. We use beam search with a size of 3 for the results in this section and section 4.4 and use a greedy search for the results in section 4.3 for faster computation.

Table 4.1 outlines the results of sequential video captioning on the LSMDC 2019 blind test set. Our TAPM method outperforms the strong adversarial inference official baseline [28] as well as the XE and AREL model in all metrics. Notably, our method shows significant gaps in the CIDEr metric, which is designed to score human-likeness [66].

Table 4.2 compares the results of sequential image captioning on the VIST test set. We report the scores computed using only one story per album following previous works. Even without explicitly optimizing the language metrics, our method is competent in the automatic evaluation. In CIDEr, our approach exhibits significant performance gains over the best-performing model AREL [57]. Our model also achieves the highest ROUGE accuracy and on-par METEOR performance with the baselines.

Table 4.4 Comparison between language models on LSMDC 2019 public test set. C, M, and R denote CIDEr, METEOR, and ROUGE-L, respectively. All evaluations are on the sentence level.

	No Adaptation			Adaptation (No split-training)			Adaptation (split-train		
Models	C	M	R	C	M	R	C	M	R
Baseline [28]	11.90	8.25	-	-	-	-	-	-	-
LSTM-WT2	3.00	5.73	17.13	1.41	4.60	12.83	7.36	8.47	20.4
XLM [16]	10.05	7.09	19.01	7.50	6.95	17.66	13.11	8.00	20.0
GPT [54]	14.01	7.96	19.84	11.81	7.86	19.23	14.76	8.33	20.0
GPT-2	**14.54**	**8.27**	**19.89**	**14.28**	**8.34**	**19.71**	**15.37**	**8.41**	**20.2**

4.3 Further Analyses

We perform various empirical analyses of our TAPM model, including (i) ablation study to inspect the contributions of key ingredients and use of (ii) six other language models beyond GPT-2.

Ablation Study. We conduct an ablation study for the TAPM model in both LSMDC 2019 and VIST dataset. We test six variants: (i) (-A) removes the adaptation loss training, (ii) (+Cap) uses the ground truth captions instead of the dummy token, (iii) (+VisualA) applies the adaptation loss to the visual encoder output instead of the language generator output, (iv) (-Split) uses naive joint training of the adaptation and generation loss, (v) (-A+Split) is (-A) that uses split-training between the visual encoder and the generator,

Table 4.3 compares the results of the ablation variants. The performance of TAPM is comparable to that of TAPM+Cap, suggesting that adaptation with videos only is as successful as the supervision with the caption labels.

The slight performance drop from TAPM to TAPM-Split shows that naive joint training can be even worse than training without the adaptation loss. Significant degradation from TAPM-A to TAPM-A+Split proves the split training without the adaptation loss performs the worst. The results of TAPM+VisualA

show that applying adaptation loss to visual encoder outputs does not improve the caption quality. Hence, using language model outputs for adaptation is crucial. Our model, TAPM, performs the best when used as proposed.

Additionally, we replace the backbone of the baseline model [28] from the RNN encoder to GPT-2 pretrained language generator [15]. As shown in the table's first two rows, the modified model performs even worse than the original baseline. This performance drop verifies our claim that employing a stronger language model does not automatically lead to a better storytelling capability. A stronger textual prior may weaken the visual conditioning when the visually conditioned target data size is insufficient. Without a proper adaptation step, the model would generate less visually relevant captions when using a strong language model such as GPT-2. Hence, the performance improvement of TAPM is attributable to the adaptation step rather than the strength of the language model.

Other Language Models. We test the generalization capability of TAPM using three pretrained language models, including LSTM-WT2 [55], XLM [16], and GPT [54]. LSTM [55] is an extension of RNN enlarging its memory capacity. We pretrain an LSTM-based two-layer encoder-decoder architecture on the WikiText-2 dataset [74]. XLM [16] is a multilingual language model designed to exploit both monolingual data and aligned bilingual data. GPT [54] is the predecessor of GPT-2. Table 4.4 compares the result of different language models. For all models, split-training with the adaptation loss contributes to consistent improvement in the language metrics, while naive joint training results in performance drops in terms of CIDEr and METEOR. These results prove that our TAPM method can improve the visual storytelling performance of a wide range of language models. Furthermore, both the adaptation loss and the split training are necessary to achieve the enhancement.

Table 4.5 Results with additional visual-only data provided in the adaptation phase. The performance rises with the number of additional videos. C, M and R denotes CIDEr, METEOR and ROUGE-L, respectively.

Models	Videos	C	M	R
Baseline (Ours)	108,487	15.37	8.41	20.21
+ Additional LSMDC	10,053	15.49	8.51	20.26
+ Additional ActivityNet	480,860	16.48	8.67	20.35

Models	Scores
Human	1.085
Official Baseline [28]	4.015
TAPM (ours)	**3.670**

Table 4.6 Official human evaluation results on the LSMDC 2019 blind test set. Lower is better.

Additional Visual-Only Data. By not relying on ground-truth captions during the adaptation phase, we can exploit additional visual-only data. In Table 4.5, we perform experiments using additional video-only dataset to further improve TAPM in LSMDC. The generation performance increases along with the number of videos used, indicating that TAPM can use visual-only data to improve cross-modal generation capability.

4.4 Human Evaluation Results

We opt for human evaluation to robustly evaluate the captioning quality of our approach. As pointed out in [57], the automatic metrics often fail to capture

	TAPM *vs* XE			TAPM *vs* AREL		
Choice (%)	TAPM	XE	Tie	TAPM	AREL	Tie
Relevance	**59.9**	34.1	6.0	**61.3**	32.8	5.9
Expressiveness	**57.3**	32.3	10.4	**57.3**	34.0	8.7
Concreteness	**59.1**	30.3	10.7	**59.6**	30.4	10.0

Table 4.7 Human evaluation results on VIST. Higher is better.

expressiveness and coherence within a story. Please refer to [57] for details on the limitations of the language metrics for story evaluation.

Table 4.6 shows human evaluation results conducted by the LSMDC 2019 challenge organizers. For 150 random sets of clips, human annotators rate generated multi-sentence descriptions from 5 (worst) to 1 (best) based on how helpful they are for a blind person to understand what is happening in the movie. To account for variability in human decisions, they aggregate three human judgments per caption and report the median score. We observe that TAPM is superior to the strong adversarial baseline [28].

For VIST, we follow previous research [57] to perform the pairwise comparison test, comparing a pair of generated samples by two methods. We ask human annotators to choose a better story between the two models' outputs for three aspects: relevance, expressiveness, and concreteness. The judges can conclude that the two samples are equally good. We randomly select 150 photo sequences and collect the medians of scores from five workers per test sample. For baselines of XE and AREL, we reproduce the results using the code and parameters provided by the original authors.

Table 4.7 shows that our TAPM outperforms the baselines in all three aspects by large margins. The performance gain of our model is the most significant in terms of relevance. The gain suggests that the captions generated by TAPM reflect the pictorial narrative better than the baselines since the relevance measures how accurately the story describes what is happening in the image sequence.

4.5 Qualitative Results

Fig. 4.1 presents a VIST example to compare the captions of TAPM against the baselines. Our generated output can avoid using some wrong words like

GT	[female] is scared, she's getting married.	this is her first wedding photo.	and here she is with everyone in the wedding.	this is [male], her husband with location, the best man.	here's [male] and his family.
XE	the **bride** was so happy to be married.	it was a beautiful day for the wedding.	the bride and groom were very happy to be married.	the bride and groom were very happy to be married.	**the bride and groom pose for a picture**.
AREL	it was a beautiful day for the wedding.	i had a great time there.	the bride and groom were so happy to be married.	then the bride and groom walked down the aisle.	**the bride and groom pose for a picture**.
TAPM (Ours)	it was a beautiful day for the wedding.	they **went down the stairs** to **get to the reception**.	the bride and groom posed for pictures.	after the ceremony, the **groom and the groom's father** pose for a picture.	**all of the guests** were happy to be at the wedding.

Figure 4.1 Qualitative comparison of sequential image captioning between our method and selected baselines on the VIST dataset. Blue and red fonts indicate correct and erroneous descriptions, respectively. Green shows the coherence between sentences. In the second sentence generated by TAPM, the model explains why the couple is going down the stairs.

bride, unlike the baselines. Furthermore, TAPM notably captures the causal relationship between the images well. In the second picture, TAPM states the purpose of going down the stairs is to get to the reception and deduces that the ceremony is over with the third picture. The readers can find more examples in Appendix.

Chapter 5

Conclusion

We proposed the *Transitional Adaptation of Pretrained Model* (TAPM) method for harmonizing the pretrained language model with the visual encoder for vision-to-language generation tasks. Extensive experiments showed that the adaptation phase using the adaptation loss consistently improves the caption quality across several language models and loss types. Our model achieved competitive performance in both automatic metrics and human evaluation for two visual storytelling tasks: the multi-sentence description of LSMDC 2019 and the image storytelling of VIST. There are several directions beyond this work. First, we can explore other adaptation loss types to improve the visual understanding capability of the pretrained language models that have proven their strengths in many language tasks. Second, one can apply our method to other cross-modal generation tasks utilizing the pretrained language models beyond visual storytelling.

Appendix A

Overview

We provide the details of implementation and experiments that are not fully described in the main paper.

The outline of this material is as follows.

- Implementation Details
  - Computing Infrastructure
  - Random Seeds
  - Computational Efficiency
- Additional Experiments
  - Fill-in-the-Blank QA
  - Randomly Initialized Backbones
- AMT user interface
- Additional examples

Appendix B

Implementation Details

B.1 Computing Infrastructure

With the GPT-2-small model as the language generator, TAPM includes 751M parameters in total. The model takes approximately 30 minutes per epoch for training using a single NVIDIA TITAN RTX GPU.

We here summarize some information about computing infrastructure for our experiments.

- GPU: NVIDIA TITAN RTX
- CPU: Intel(R) Xeon(R) E5-2650 CPU
- OS : Ubuntu 16.04 LTS OS.
- RAM: SAMSUNG DDR4 8G
- Operating System: Ubuntu 16.04

Table B.1 Mean and standard deviations of TAPM using random seed $[0-4]$. Note that we fix the random seed to 0 in all other experiments.

	LSMDC			VIST		
Stats	C	M	R	C	M	R
mean	15.50	8.55	20.23	8.26	34.02	29.70
std	0.33	0.05	0.12	0.17	0.08	0.06

Table B.2 The number of parameters and GFLOPs.

Models	GFLOPs (G)	Params (M)
TAPM	5.766	62.3
-A	5.761	60.3

- Names and versions of relevant software libraries and frameworks: python $\geq$ 3.6 and PyTorch $\geq$ 1.3

All pretrained transformers are from the huggingface implementations (`https://github.com/huggingface/transformers`). See the source code for more details.

B.2 Random Seeds

Table B.1 shows that the performance of TAPM is stable across several random seeds.

B.3 Computational Efficiency

Table B.2 shows the number of parameters and GFLOPs (floating point operations) for training. Since the adaptation module (A) requires only 4 FC layers ($f_v^p, f_s^p, f_v^f, f_s^f$), it does not significantly affect computation complexity and training time. The adaptation module is not used for the inference time, so the inference time and complexity of TAPM and TAPM-A are exactly the same. Please note that our adaptation module does not contribute to the complexity of model inference.

Appendix C

Additional Experiments

C.1 Fill-in-the-Blank QA

We explore the generalizability of TAPM on another type of task. In Table C.1 we test TAPM with a videoQA task, specifically Fill-in-the-Blank QA task of LSMDC2017, beyond the sequential caption generation tasks in the original paper. The results show that our approach achieves the state-of-the-art performance for another multimodal task.

C.2 Randomly Initialized Backbones

Additionally, we explore how TAPM affects randomly initialized language models. In Table C.2, we test three randomly initialized language generators; LSTM-Scratch, QRNN-Scratch [17] and GPT-2-Scratch. As with pretrained language models, adaptation with split-training consistently improves caption quality across all language models. Even when there is no pretrained language information to adapt to, self-supervision may enhance robustness [76] and hence

Table C.1 Results on Fill-in-the-Blank QA task in LSMDC 2017.

Models	Accuracy
JsFusion [75]	45.52
Cross-Modal BERT −TAPM	50.10
Cross-Modal BERT +TAPM	52.53

Table C.2 Comparison between not pretrained language models on LSMDC 2019 public test set. C, M and R denotes CIDEr, METEOR and ROUGE-L, respectively. All evaluations are on the sentence level.

	No Adaptation			Adaptation (No split-training)			Adaptation (split-training)		
Models	C	M	R	C	M	R	C	M	R
Baseline [28]	11.90	8.25	-	-	-	-	-	-	-
LSTM-Scratch	5.13	6.77	19.34	3.67	5.95	18.51	7.90	7.70	19.45
QRNN-Scratch	1.48	5.65	16.29	3.01	5.73	17.13	7.05	7.25	18.58
GPT2-Scratch	4.17	5.94	16.97	4.01	6.03	17.18	12.68	8.27	20.08
GPT-2	**14.54**	**8.27**	**19.89**	**14.28**	**8.34**	**19.71**	**15.37**	**8.41**	**20.21**

generalization in sparse-signal datasets such as LSMDC.

Appendix D

AMT user interface

In our main paper, we conduct our human evaluation to compare different models' outputs on Amazon Mechanical Turk (AMT). Figure D.1,D.2,D.3 respectively shows the user interfaces for AMT instruction and human evaluation layouts for VIST and LSMDC 2019.

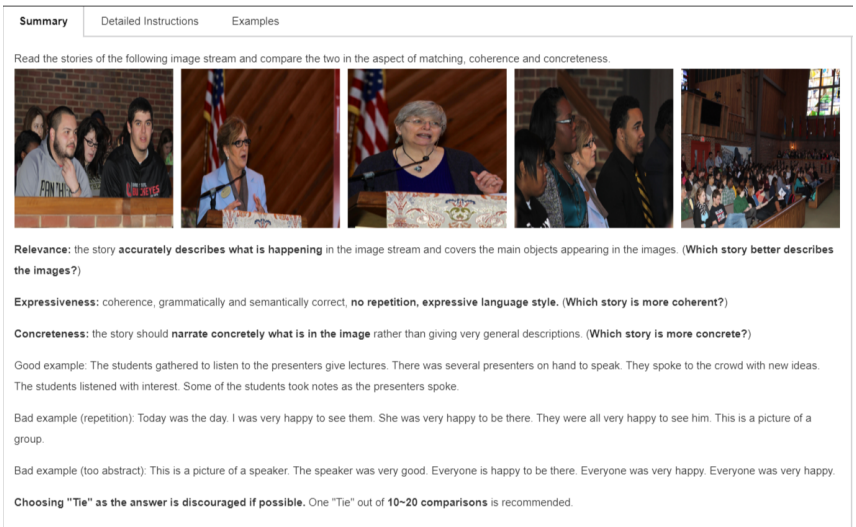


Figure D.1 The AMT Instruction for the turkers for the VIST model comparison.

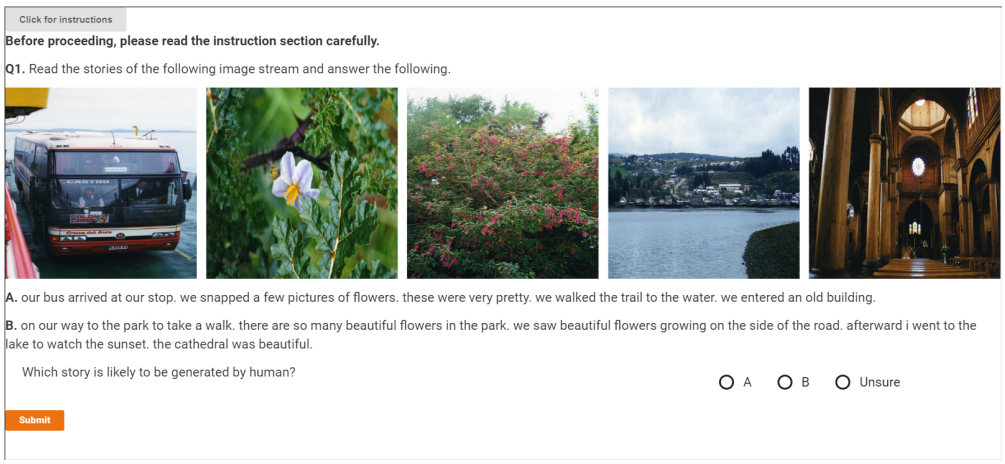


Figure D.2 The AMT human evaluation layout for the VIST model comparison.

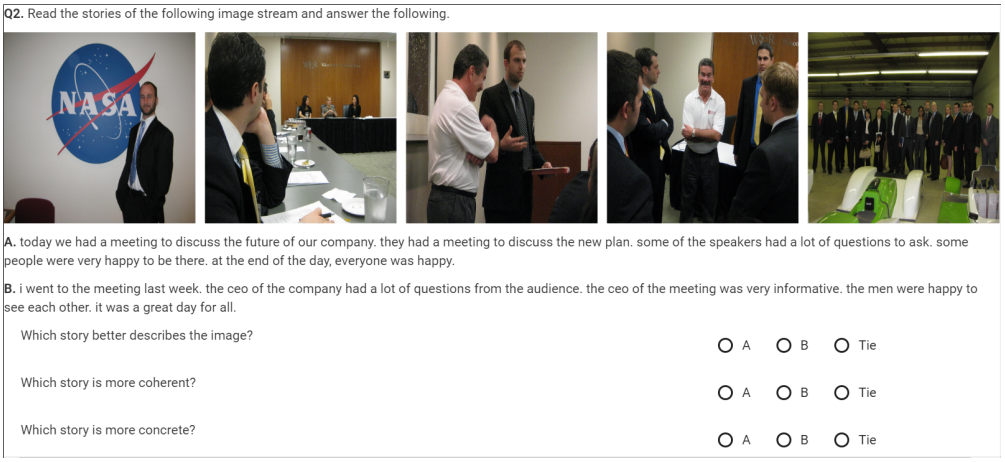


Figure D.3 The AMT human evaluation layout for the LSMDC 2019 model comparison.

Appendix E

Additional examples

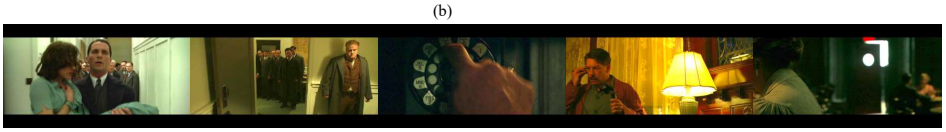
We provide additional examples to compare TAPM variants and with selected baselines qualitatively. Figure E.1,E.2 are from LSMDC 2019 experiments, while Figure E.3,E.4 are from VIST tests.

GT	someone grooves.	someone touches the screen.	someone clicks the mouse.	someone points to someone.	someone laughs to the kids.
TAPM -A	someone smiles.	***someone clicks on a computer.***	someone looks at the computer.	someone smiles.	someone smiles.
TAPM -Split	someone smiles.	someone looks at the screen.	someone looks at the screen.	someone smiles.	someone smiles.
TAPM (Ours)	someone gives someone a thumbs up.	***someone clicks on a message on the screen: "access denied. ".***	someone sits at a computer.	someone gives a thumbs up.	someone smiles and nods.

(a)

GT	in the control room.	someone sneers at someone.	someone looks sharply at someone.	someone puts his arm around someone.	someone faces someone.
TAPM -A	someone sits on a couch.	someone looks at someone.	someone looks at someone.	someone looks at someone.	someone looks at someone.
TAPM -Split	someone sits on a couch.	someone looks at someone.	someone looks at someone.	someone looks at the camera.	someone looks at someone.
TAPM (Ours)	***someone sits at a desk in the security hub.***	someone and someone exchange a look.	someone and someone watch from their desks.	someone grabs the camera.	someone looks up at someone.

(b)



GT	now, someone nods to someone.	in the interrogation room, the burly agent stays put beside the open door.	someone dials a payphone.	in his home, someone answers.	now, a woman sits at a bar.
TAPM -A	someone looks at someone.	someone walks into the room.	someone is sitting on the couch.	someone sits on the bed, looking at the glass.	someone looks at someone.
TAPM -Split	someone pulls her into the room.	someone looks at someone.	someone takes a drink.	someone sits on the couch.	someone looks at someone.
TAPM (Ours)	someone and someone are in the corridor.	someone looks at someone.	**someone hangs up the phone.**	**someone answers the phone.**	someone looks up at someone.

(c)

Figure E.1 The qualitative comparison between TAPM variants in the LSMDC 2019 dataset. Red indicates repetitions, blue/italic indicates interesting samples, and green/bold shows coherent sentences. In (a), TAPM tries to predict the message on the screen but nearly misses.

GT	another closes her eyes.	someone flips open another phone.	and pulls up its speed dial numbers.	someone appears at the top of the list.	someone glances at the executive, then presses a button.
TAPM -A	someone looks at someone.	someone walks to the door.	someone looks at the screen.	*someone reads the number.*	someone looks at the phone.
TAPM -Split	someone looks at someone.	someone looks at the camera.	someone looks at the screen.	*someone reads the number.*	someone looks at the phone.
TAPM (Ours)	someone and someone sit at the end of the long row of seats.	someone sees a man with a gun on a security guard.	someone sees a text message from someone.	*someone's phone shows a text message, "call me. ".*	someone answers his cell.

(d)

GT	someone glances toward the open front door, then takes a bite.	now, a man leads the woman and a plump brunette up the stairs.	the brunette lags behind.	now, a goateed man bicycles with the father.	a car darts out and nearly hits them.
TAPM -A	someone takes a sip of the coffee and then goes to the kitchen.	someone walks up to someone.	someone walks up to someone and starts to walk away.	someone rides a bike.	someone rides a bike.
TAPM -Split	someone pulls out a bottle of milk and starts to eat.	someone walks up to someone.	someone walks up to someone and starts to walk away.	someone rides a bike.	someone rides a bike.
TAPM (Ours)	someone eats a cookie.	someone and someone climb the stairs.	someone and someone run up to the fence.	**now, someone and someone ride on a bike.**	**someone rides up behind them.**

(e)

Figure E.2 The qualitative comparison between TAPM variants in the LSMDC 2019 dataset. Red indicates repetitions, blue/italic indicates interesting samples, and green/bold shows coherent sentences. In (d), TAPM takes a wrong guess for the message on the cell phone.

GT	we decided to take a walk around town to see all the local stores.	we started bright and early in the morning, it was such a beautiful day.	we stopped for lunch at the location location and had a wonderful sandwhich.	. we then went to copperfield's books and read and browsed for awhile.	finally, on our way back we stopped in to popeyes for some delicious fried chicken.
TAPM -A	the city was a great place to go.	the view from the window was amazing.	the building was very beautiful.	the restaurant was very nice.	we had a great time.
TAPM -Split	*we went to the location location location location locatio*	the city is beautiful.	the building was very nice.	the sign was very nice.	the place was very nice.
TAPM (Ours)	we went to the city today to see what it was like.	the windows were very old and had a lot of history.	we saw a lot of interesting things.	there was a lot of cool things to see.	afterward i went back to my hotel.

(a)

GT	a little driver wanted to take them to their destination.	the church was small and held a few guests.	members of the party were young and old.	the rings were basic but significant.	photos taken provided memories for a lifetime.
TAPM -A	the car was a little too big for my liking.	the church was beautiful.	the flowers were beautiful.	the bride was very happy to be able to touch the ring.	the bride and groom were happy to be married.
TAPM -Split	the boy was so excited to get his new car.	the church was beautiful.	**the bride was so happy.**	the bride and groom were so happy.	the bride and groom were happy to be married.
TAPM (Ours)	i took my son to the park yesterday.	there were a lot of beautiful sights to see.	**her favorite part of the wedding was the umbrella.**	the bride's ring looked so delicate and beautiful.	they are now married and ready to go home.

(b)

Figure E.3 The qualitative comparison between TAPM variants in the VIST dataset. Red indicates uninformative captions, blue/italic indicates language modelling failures, and green/bold shows coherent sentences. In (a), TAPM-Split shows a language modelling failure. Jointly training the adaptation loss with the generation loss could harm the language generation ability of the model. We see that full TAPM does not suffer from such issues. In (b), TAPM-Split and full TAPM try to describe the image within the context of wedding.

GT	the cruise ship look so majestic as it comes in the port.	friends are waiting to greet the passengers.	the port is all decked out for christmas, including this wonderful tree.	this lamp is so cool, i wonder where you find such a thing.	any of the passengers would like to get their picture taken professionally certainly have that option here.
AREL	we went on vacation to location.	we got to see a lot of people there.	*this is a picture of a tree.*	i had a great time there.	*this is a picture of a building.*
XE	we went to the location.	we took a trip to the local museum.	we saw a lot of flowers on the wall.	we saw a lot of interesting things to see.	we had a great time.
TAPM (Ours)	our cruise ship was ready for us.	the view from the deck was amazing.	i bought some flowers while i was there.	there was a lot of decorations there.	this sign was a great addition to the christmas tree.

(a)

GT	it was time for the halloween party and [male] the pirate was ready to go.	but [male] the pirate was very sad as [male] the balloon man showed up in a better costume.	[male] the balloon man had all kinds of fans.	a few that didn't see [male] the pirate were happy with [male] the balloon man.	but when [male] the pirate asked if [male] the balloon man can be his friend, everybody joined in too.
AREL	*this is a picture of a man.*	we had a lot of fun playing games.	there was a lot of fun dancing.	some of the performers were really cool.	*this is a picture of a group of people.*
XE	the halloween party was a lot of fun.	there were a lot of people there.	there were a lot of people there.	we had a great time.	at the end of the night, everyone had a great time.
TAPM (Ours)	i went to my friend's halloween party last night.	we played games and had fun.	some of the costumes were very creative.	after the party we all got together for a group photo.	it was a great halloween party and everyone had a great time.

(b)

Figure E.4 The qualitative comparison of TAPM and the selected baselines in the VIST dataset. Red indicates uninformative or misaligned captions, and blue/italic indicates isolated sentences.

Bibliography

[1] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele, "Movie description," *IJCV*, 2017.

[2] T.-H. K. Huang, F. Ferraro, N. Mostafazadeh, I. Misra, A. Agrawal, J. Devlin, R. Girshick, X. He, P. Kohli, D. Batra, C. L. Zitnick, D. Parikh, L. Vanderwende, M. Galley, and M. Mitchell, "Visual Storytelling," in *NAACL-HLT*, 2016.

[3] J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell, "Language models for image captioning: The quirks and what works," in *ACL*, 2015.

[4] M. Tanti, A. Gatt, and K. P. Camilleri, "Transfer learning from language models to image caption generators: Better models may not transfer better," *arxiv:1901.01216*, 2019.

[5] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to Sequence – Video to Text," in *ICCV*, 2015.

[6] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang, "Hierarchical Recurrent Neural Encoder for Video Representation with Application to Captioning," in *CVPR*, 2016.

[7] A. Rohrbach, M. Rohrbach, and B. Schiele, "The Long-Short Story of Movie Description," in *DAGM*, 2016.

[8] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing Videos by Exploiting Temporal Structure," in *ICCV*, 2015.

[9] Y. Yu, H. Ko, J. Choi, and G. Kim, "End-to-end Concept Word Detection for Video Captioning, Retrieval, and Question Answering," in *CVPR*, 2017.

[10] L. Zhou, Y. Zhou, J. J. Corso, R. Socher, and C. Xiong, "End-to-End Dense Video Captioning with Masked Transformer," in *CVPR*, 2018.

[11] S. Edunov, A. Baevski, and M. Auli, "Pre-trained language model representations for language generation," in *NAACL*, 2019.

[12] J. Yang, M. Wang, H. Zhou, C. Zhao, W. Zhang, Y. Yu, and L. Li, "Towards making the most of bert in neural machine translation," in *AAAI*, 2020.

[13] X. Liu, H. Li, J. Shao, D. Chen, and X. Wang, "Show, tell and discriminate: Image captioning by self-retrieval with partially labeled data," in *ECCV*, 2018.

[14] R. Luo, B. Price, S. Cohen, and G. Shakhnarovich, "Discriminability objective for training descriptive captions," in *CVPR*, 2018.

[15] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[16] A. C. Guillaume Lample, "Cross-lingual Language Model Pretraining," in *NIPS*, 2019.

[17] J. Bradbury, S. Merity, C. Xiong, and R. Socher, "Quasi-Recurrent Neural Networks," in *ICLR*, 2017.

[18] G. Kim, L. Sigal, and E. P. Xing, "Joint Summarization of Large-scale Collections of Web Images and Videos for Storyline Reconstruction," in *CVPR*, 2014.

[19] G. Kim and E. P. Xing, "Jointly Aligning and Segmenting Multiple Web Photo Streams for the Inference of Collective Photo Storylines," in *CVPR*, 2013.

[20] C. C. Park and G. Kim, "Expressing an Image Stream with a Sequence of Natural Sentences," in *NIPS*, 2015.

[21] C. C. Park, Y. Kim, and G. Kim, "Retrieval of Sentence Sequences for an Image Stream via Coherence Recurrent Convolutional Networks," *IEEE TPAMI*, 2018.

[22] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical Neural Story Generation," in *ACL*, 2018.

[23] P. Jain, P. Agrawal, A. Mishra, M. Sukhwani, A. Laha, and K. Sankaranarayanan, "Story Generation from Sequence of Independent Short Descriptions," in *SIGKDD Workshop on Machine Learning for Creativity (ML4Creativity)*, 2017.

[24] Q. Huang, Z. Gan, A. Celikyilmaz, D. Wu, J. Wang, and X. He, "Hierarchically Structured Reinforcement Learning for Topically Coherent Visual Story Generation," in *AAAI*, 2019.

[25] J. Tang, J. Wang, Z. Li, J. Fu, and T. Mei, "Show, Reward, and Tell: Adversarial Visual Story Generation," in *AAAI*, 2018.

[26] A. Fan, M. Lewis, and Y. Dauphin, "Strategies for structuring story generation," in *ACL*, 2019.

[27] S. Gella, M. Lewis, and M. Rohrbach, "A dataset for telling the stories of social media videos," in *EMNLP*, 2018.

[28] J. S. Park, M. Rohrbach, T. Darrell, and A. Rohrbach, "Adversarial Inference for Multi-Sentence Video Description," in *CVPR*, 2019.

[29] J. Lei, L. Wang, Y. Shen, D. Yu, T. L. Berg, and M. Bansal, "Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning," in *ACL*, 2020.

[30] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks," in *NIPS*, 2015.

[31] L. Zhang, F. Sung, F. Liu, T. Xiang, S. Gong, Y. Yang, and T. M. Hospedales, "Actor-Critic Sequence Training for Image Captioning," in *NIPS*, 2017.

[32] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved Image Captioning via Policy Gradient optimization of SPIDEr," in *ICCV*, 2017.

[33] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-Critical Sequence Training for Image Captioning," in *CVPR*, 2017.

[34] Z. Ren, X. Wang, N. Zhang, X. Lv, and L.-J. Li, "Deep Reinforcement Learning-based Image Captioning with Embedding Reward," in *CVPR*, 2017.

[35] L. Li and B. Gong, "End-to-End Video Captioning with Multitask Reinforcement Learning," in *WACV*, 2019.

[36] X. Wang, W. Chen, J. Wu, Y.-F. Wang, and W. Y. Wang, "Video Captioning via Hierarchical Reinforcement Learning," in *CVPR*, 2018.

[37] J. N. Tsitsiklis and B. Van Roy, "Analysis of temporal-diffference learning with function approximation," *NIPS*, 1996.

[38] C.-Y. Ma, Y. Kalantidis, G. AlRegib, P. Vajda, M. Rohrbach, and Z. Kira, "Learning to Generate Grounded Image Captions without Localization Supervision," *arXiv:1906.00283*, 2019.

[39] L. Zhou, Y. Kalantidis, X. Chen, J. J. Corso, and R. Marcus, "Grounded Video Description," in *CVPR*, 2019.

[40] R. R. Selvaraju, S. Lee, Y. Shen, H. Jin, D. Batra, and D. Parikh, "Taking a HINT: Leveraging Explanations to Make Vision and Language Models More Grounded," in *ICCV*, 2019.

[41] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "VideoBERT: A Joint Model for Video and Language Representation Learning," in *ICCV*, 2019.

[42] H. Tan and M. Bansal, "LXMERT: Learning Cross-Modality Encoder Representations from Transformers," in *EMNLP*, 2019.

[43] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks," in *NIPS*, 2019.

[44] P. Bhargava, "Adaptive transformers for learning multimodal representations," in *ACL SRW*, 2020.

[45] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "What does bert with vision look at?," in *ACL (short)*, 2020.

[46] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai, "Vl-bert: Pre-training of generic visual-linguistic representations," in *ICLR*, 2020.

[47] C. Zheng, Q. Guo, and P. Kordjamshidi, "Cross-modality relevance for reasoning on language and vision," in *ACL*, 2020.

[48] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, and J. Liu, "UNITER: UNiversal Image-TExt Representation Learning," *arXiv:1909.11740*, 2019.

[49] G. Li, N. Duan, Y. Fang, M. Gong, D. Jiang, and M. Zhou, "Unicoder-VL: A Universal Encoder for Vision and Language by Cross-modal Pre-training," in *AAAI*, 2020.

[50] C. Sun, F. Baradel, K. Murphy, and C. Schmid, "Learning Video Representations using Contrastive Bidirectional Transformer," *arXiv:1906.05743*, 2019.

[51] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier Nonlinearities Improve Neural Network Acoustic Models," in *ICML*, 2013.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *NIPS*, 2017.

[53] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *AISTATS*, 2010.

[54] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," *OpenAI blog*, 2018.

[55] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, 1997.

[56] L. Yu, M. Bansal, and T. L. Berg, "Hierarchically-Attentive RNN for Album Summarization and Storytelling," in *EMNLP*, 2017.

[57] X. Wang, W. Chen, Y.-F. Wang, and W. Y. Wang, "No Metrics Are Perfect: Adversarial Reward Learning for Visual Storytelling," in *ACL*, 2018.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016.

[59] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *IJCV*, 2015.

[60] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering," in *CVPR*, 2018.

[61] A. Z. Joao Carreira, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset," in *CVPR*, 2017.

[62] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The Kinetics Human Action Video Dataset," *arXiv:1705.06950*, 2017.

[63] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[64] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei, "Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations," *IJCV*, 2017.

[65] R. Sennrich, B. Haddow, , and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," in *ACL*, 2016.

[66] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based Image Description Evaluation," in *CVPR*, 2015.

[67] S. Banerjee and A. Lavie, "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments," in *ACL*, 2005.

[68] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *WAS*, 2004.

[69] T. Kim, M.-O. Heo, S. Son, K.-W. Park, and B.-T. Zhang, "GLAC Net: GLocal Attention Cascading Networks for Multi-image Cued Story Generation," *CoRR*, 2018.

[70] D. Gonzalez-Rico and G. Fuentes-Pineda, "Contextualize, Show and Tell: A Neural Visual Storyteller," *arXiv:1806.00738*, 2018.

[71] B. Zhang, H. Hu, and F. Sha, "The Steep Road to Happily Ever After: An Analysis of Current Visual Storytelling Models," in *ICCV19 CLVL workshop: 3rd Workshop on Closing the Loop Between Vision and Language*, 2019.

[72] Y. Jung, D. Kim, S. Woo, K. Kim, S. Kim, and I. S. Kweon, "Hide-and-Tell: Learning to Bridge Photo Streams for Visual Storytelling," in *AAAI*, 2020.

[73] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *ICLR*, 2015.

[74] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer Sentinel Mixture Models," in *ICLR*, 2017.

[75] Y. Yu, J. Kim, and G. Kim, "A joint sequence fusion model for video question answering and retrieval," in *ECCV*, 2018.

[76] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," in *NeurIPS*, 2019.

초록

시각-언어 생성 문제를 풀 때, 기존 모델들은 일반적으로 시각 인코더와 언어 생성기를 각 영역에서 선학습한 후 목표 문제에 미세조정한다. 그러나 이러한 직접적 이전 방식은 시각적 특정성과 언어적 유창성 간의 부조화를 낳을 수 있는데, 이는 시각과 언어 모델 각각이 공통되는 영역이 없는 대량의 시각과 언어 데이터에서 서로 별도로 학습되기 때문이다. 본 연구에서는 선학습과 미세조정 사이에 전이 적용 문제를 학습할 때 보다 어려운 목표 문제인 시각적 스토리텔링 문제에서 시각 인코더와 언어 모델을 조화시킬 수 있음을 밝힌다. 그 방법으로 제시한 TAPM은 언어 라벨 없이 시각적 입력값 간의 연결성 만을 파악하는 간단한 문제를 사용함으로서 멀티모달 모듈 간의 연결성을 확보한다. 연구결과를 종합해 볼 때, 제시된 적용 단계는 순차적 비디오 또는 이미지 캡셔닝 문제에서 다수 언어 모델의 성능을 크게 향상시켰다. 그 결과, 복수 문장 설명 문제인 LSMDC 2019 [1]와 이미지 스토리텔링 문제인 VIST [2]에서 자동 성능과 인적 평가 모두 최고 성능을 달성했다. 또한 추가적 실험으로 캡션의 질적 성능 향상이 특정 언어 모델에 국한되지 않는다는 점을 보였다.

주요어: 인공지능, 멀티모달 학습, 시각적 스토리텔링

학번: 2019-29077