Multi-modal Circulant Fusion for Video-to-Language and Backward

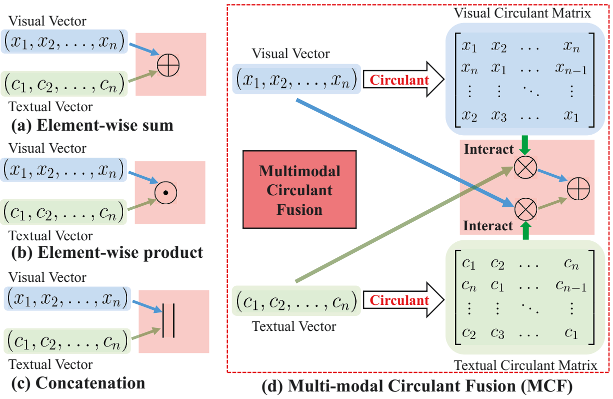
Aming Wu and **Yahong Han**
School of Computer Science and Technology, Tianjin University, Tianjin, China
{tjwam, yahong}@tju.edu.cn

Abstract

Multi-modal fusion has been widely involved in focuses on the modern artificial intelligence research, e.g., from visual content to languages and backward. Common-used multi-modal fusion methods mainly include element-wise product, element-wise sum, or even simply concatenation between different types of features, which are somewhat straightforward but lack in-depth analysis. Recent studies have shown fully exploiting interactions among elements of multi-modal features will lead to a further performance gain. In this paper, we put forward a new approach of multi-modal fusion, namely Multi-modal Circulant Fusion (MCF). Particularly, after reshaping feature vectors into circulant matrices, we define two types of interaction operations between vectors and matrices. As each row of the circulant matrix shifts one elements, with newly-defined interaction operations, we almost explore all possible interactions between vectors of different modalities. Moreover, as only regular operations are involved and defined a priori, MCF avoids increasing parameters or computational costs for multi-modal fusion. We evaluate MCF with tasks of video captioning and temporal activity localization via language (TALL). Experiments on MSVD and MSRVTT show our method obtains the state-of-the-art performance for video captioning. For TALL, by plugging into MCF, we achieve a performance gain of roughly 4.2% on TACoS.

1 Introduction

Multi-modal data are widely involved in recent focuses on artificial intelligence research, e.g., visual to languages [Yao *et al.*, 2015] and the backward of visual localization via language queries [Gao *et al.*, 2017]. Though many efforts have been made towards feature learning of images and texts, e.g., via convolutional neural networks (CNN) [He *et al.*, 2016] and recurrent neural networks (RNN) [Kiros *et al.*, 2015], respectively, in-depth analysis of multi-modal fusion has been unintentionally ignored. Common-used or somewhat straightforward multi-modal fusion methods mainly include element-wise sum (Fig. 1 (a)), element-wise product (Fig. 1 (b)), or even simply concatenation (Fig. 1 (c)) between different types of features [Fukui *et al.*, 2016; Yu *et al.*, 2017]. As feature vectors of different modalities lie in different feature spaces, interactions or correlations might not just exist among corresponding dimensions of multi-modal vectors. Besides no interactions with concatenation, element-wise sum or product only partially explores interactions or correlations among multi-modal features, which may burden the fusion performance.



(a) Element-wise sum

(b) Element-wise product

(c) Concatenation

(d) Multi-modal Circulant Fusion (MCF)

Figure 1: Different ways of multi-modal fusion. (a), (b), and (c) indicate three common fusion methods. (d) is the proposed multi-modal circulant fusion (MCF) method. '$\oplus$' denotes element-wise sum. '$\odot$' denotes element-wise product. '$||$' denotes concatenation. '$\otimes$' denotes multiplication operation.

Recent studies [Fukui *et al.*, 2016; Yu *et al.*, 2017] have shown fully exploiting interactions among elements of multi-modal features will lead to a further performance gain. Authors in [Fukui *et al.*, 2016; Yu *et al.*, 2017] develop multi-modal bilinear pooling to capture pairwise interactions between multi-modal feature dimensions. As bilinear pooling defines p parameterized projection matrices, where p is the dimension of the output fused features, additional huge number of parameters are introduced into the model. Although count sketch [Fukui *et al.*, 2016] or matrix factorization [Yu *et al.*, 2017] were employed to shrink projection matrices, more computational costs were introduced again. Moreover, empirical studies in [Fukui *et al.*, 2016] show that the performance gain from multi-modal fusion is guaranteed only when the dimension p of the output fused features is high, which means there still needs a large number of parameterized projection matrices. Therefore, if we skipped the parameterized projection matrices and defined a fully-interacted module of multi-modal fusion a priori, e.g., constituted with only regular operations like element-wise sum or product, we expect to get a good fusion performance with fewer parameters or lower computational costs.

In this paper, we propose a new module of Multi-modal Circulant Fusion (MCF) to fully exploit interactions among multi-modal features. In Fig. 1 (d) we show the main idea of MCF by taking the fusion of visual and textual vectors as an example. In particular, after reshaping visual or textual vectors into circulant matrices, respectively, we define two types of interaction operations between original feature vectors and the reshaped circulant matrices. Finally, we use element-wise sum to obtain the joint representation of these two cross-fused vectors. As each row of the circulant matrix shifts one elements, with newly-defined interaction operations, we almost explore all possible interactions between vectors of different modalities. Note that, as only regular operations are involved in the MCF and the proposed MCF is defined a priori, we avoid introducing new parameters or increasing computational costs for multi-modal fusion.

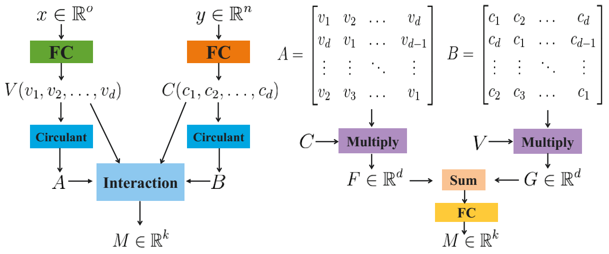
In the experiments, we extensively evaluate the proposed MCF with tasks of video captioning and temporal activity localization via language (TALL) [Gao *et al.*, 2017]. Experimental results on MSVD [Chen and Dolan, 2011] and MSRVTT [Chen and Dolan, 2011] show that our method could obtain the state-of-the-art performance of video captioning. For TALL, by plugging into the MCF, we achieve a performance improvement of roughly 4.2% on TACoS [Regneri *et al.*, 2013].

2 Related Works

As the original bilinear method [Tenenbaum and Freeman, 1997] needs a huge number of parameters and is not applied into a multi-modal task [Peng *et al.*, 2018; Zhang *et al.*, 2018], the method MCB [Fukui *et al.*, 2016] uses the count sketch to reduce the number of parameters of bilinear pooling and employs bilinear pooling to solve multi-modal fusion. However, MCB always outputs a high-dimensional fused feature which increases computational cost of following processes. Whereas the work in [Yu *et al.*, 2017] proposes a bilinear pooling method based on matrix factorization to combine multi-modal features.

Recent advances in video captioning mainly follow the encoding-decoding framework and generate captions via RNN [Zhu *et al.*, 2017; Yang *et al.*, 2017]. Though strengths in representation ability, CNN has not been well exploited in video captioning. In this paper, we propose a coarse-to-fine multi-stage convolutional network which includes a MCF and many convolutional decoders that generate finer video descriptions. In order to reduce the risk of vanishing gradients, inspired by the work [Zhang *et al.*, 2016], we enforce intermediate supervisions for each stage. Experimental results on MSVD and MSRVTT show the effectiveness of our method.

As a backward direction of video captioning, Temporal Activity Localization via Language (TALL) [Gao *et al.*, 2017] is given a temporally untrimmed video and a natural language query, and then determines the start and end moments for the described activity inside the video. By plugging MCF into the architecture of [Gao *et al.*, 2017], we obtain a performance improvement owing to a better multi-modal fusion.


$x \in \mathbb{R}^o$
FC
$V(v_1, v_2, \ldots, v_d)$
Circulant
A
$y \in \mathbb{R}^n$
FC
$C(c_1, c_2, \ldots, c_d)$
Circulant
B
Interaction
$M \in \mathbb{R}^k$
$A = \begin{bmatrix} v_1 & v_2 & \ldots & v_d \\ v_d & v_1 & \ldots & v_{d-1} \\ \vdots & \vdots & \ddots & \vdots \\ v_2 & v_3 & \ldots & v_1 \end{bmatrix}$
$B = \begin{bmatrix} c_1 & c_2 & \ldots & c_d \\ c_d & c_1 & \ldots & c_{d-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_2 & c_3 & \ldots & c_1 \end{bmatrix}$
C
Multiply
V
Multiply
$F \in \mathbb{R}^d$
Sum
$G \in \mathbb{R}^d$
FC
$M \in \mathbb{R}^k$

(a) Multi-modal Circulant Fusion (MCF) **(b) The detail of Interaction in MCF**

Figure 2: The flowchart of Multimodal Circulant Fusion (MCF).

3 Multi-modal Circulant Fusion

The detailed procedures of MCF are illustrated in Fig. 2. Given two feature vectors in different modalities, e.g., the visual features $x \in \mathbb{R}^o$ and the textual features $y \in \mathbb{R}^n$, to reduce computational cost, we first utilize two projection matrix $W_1 \in \mathbb{R}^{d \times o}$ and $W_2 \in \mathbb{R}^{d \times n}$ $(d \leq min(o, n))$ to project the two input features to a lower dimensional space.

$$\begin{aligned} V &= xW_1^T \\ C &= yW_2^T \end{aligned} \tag{1}$$

where W_1^T and W_2^T are the transpose of W_1 and W_2.

Then we use the projection vector $V \in \mathbb{R}^d$ and $C \in \mathbb{R}^d$ to construct circulant matrix $A \in \mathbb{R}^{d \times d}$ and $B \in \mathbb{R}^{d \times d}$.

$$\begin{aligned} A &= circ(V) \\ B &= circ(C) \end{aligned} \tag{2}$$

where $circ(b)$ denotes converting b to a circulant matrix.

In order to make elements in projection vector and circulant matrix fully interact, we explore two different multiplication operations. The first is shown in Eq. (3) and uses matrix multiplication between circulant matrix and projection vector.

$$\begin{aligned} F &= CA \\ G &= VB \end{aligned} \tag{3}$$

The second is to have projection vector and each row vector of circulant matrix do element-wise product. The procedures are shown as follows:

$$\begin{aligned} F &= \frac{1}{d} \sum_{i=1}^{d} a_i \odot C \\ G &= \frac{1}{d} \sum_{i=1}^{d} b_i \odot V \end{aligned} \tag{4}$$

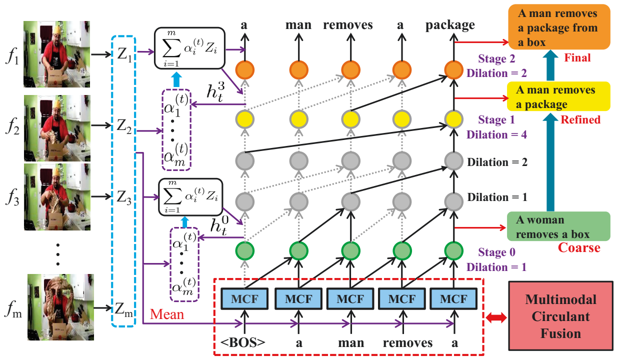


Figure 3: Illustration of multi-stage sequential decoder. For this decoder, we first use MCF to obtain joint representation of visual feature and word embedding feature. Then we take the joint representation as input of this decoder. 'Coarse', 'Refined', and 'Final' indicate three stages of decoder. The corresponding increasingly improved video description are show in green, yellow, and orange.

where $a_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}^d$ are row vector of circulant matrix A and B. $\odot$ denotes operation of element-wise product. It is noted that we do not introduce new parameters in the multiplication operation.

Finally, through a projection matrix $W_3 \in \mathbb{R}^{d \times k}$, we convert the element-wise sum vector of $F \in \mathbb{R}^d$ and $G \in \mathbb{R}^d$ to target vector $M \in \mathbb{R}^k$.

4 MCF for Video Captioning

In this section, we develop a new framework for video captioning, in which we construct convolutional encoder and decoder for video-to-language translation. Note that, in the decoder, we take MCF as a base layer for coarse decoding, stacked on which with layered dilations for refined and final decoding. Thus we construct a Multi-stage Decoder with MCF.

4.1 Convolutional Encoding Network

Feature Extraction. We use pre-trained convolutional networks to extract feature for each of the m video frames, which results in a vector $X_i \in \mathbb{R}^q$ for the i-th frame.

Discriminative Enhancing. Given two consecutive frame features X_i and X_{i+1} $(i = 0, 2, \cdots)$, we first compute interframe differences $diff$. Then by a $ReLU$ operation, we add the positive values of $diff$ to X_{i+1} and the absolute value of negative values in $diff$ to X_i. Thus, we enlarge the discrimination gap between X_i and X_{i+1}.

$$diff = X_{i+1} - X_i, \tag{5}$$

$$V_{i+1} = X_{i+1} + ReLU(diff), \tag{6}$$

$$V_i = X_i + ReLU(-diff), \tag{7}$$

$$ReLU(x) = \begin{cases} x, & \text{if } x > 0; \\ 0, & \text{else} \end{cases} \tag{8}$$

where $i = 0, 2, \cdots$. $V_i \in \mathbb{R}^q$ is the enhancing result of X_i.

Reconstruction Network. In this work, we construct a reconstruction network to learn a compact representation Z_i for each video frame. The procedures are shown as follows:

$$\begin{aligned} Z_i &= ReLU(W_E * V_i + b_E) \\ R_i &= W_D * Z_i + b_D \\ L_{recon} &= ||X_i - R_i||_2^2 \end{aligned} \tag{9}$$

where $i = 0, 1, \cdots$. $W_E \in \mathbb{R}^{1 \times q \times r}$ denotes convolutional filter and $r \leq q$, $b_E \in \mathbb{R}^r$ and $b_D \in \mathbb{R}^q$ are the bias parameters. $Z_i \in \mathbb{R}^r$ is the learned compact representation. $W_D \in \mathbb{R}^{1 \times r \times q}$ is the reconstruction filter and $R_i \in \mathbb{R}^q$ denotes reconstruction result. L_{recon} denotes reconstruction loss between R_i and X_i and $|| \cdot ||_2$ denotes an ℓ_2-norm. $*$ denotes a convolutional operator.

4.2 Multi-stage Convolutional Decoder with MCF

As introduced above, we stack many dilated convolutional layers [Chen *et al.*, 2017] to form a coarse-to-fine multi-stage decoder (Fig. 3). In the following, we denote by $\hat{Y}^j = \{\hat{Y}_0^j, ..., \hat{Y}_{T-1}^j\}$ the predicted word sequence of the j-th stage of decoder, where $j \in \{0, ..., N_f\}$ and $(N_f + 1)$ is the number of stages. We denote by $Y = \{Y_0, ..., Y_{T-1}\}$ the target word sequence, where T denotes sequence length. Note that we treat the initial stage $j = 0$ as coarse decoder and above stages as refined decoders. $[a, b]$ represents the concatenation of a and b. And '$\odot$' is an element-wise product operation, '$*$' denotes a convolutional operator. $Z = \{Z_0, Z_1, \cdots\}$. The $\sigma(\cdot)$ is a sigmoid function.

MCF as a Coarse Decoder. In the bottom stage ($j = 0$), we use one dilated convolutional layer to learn a coarse decoder. At each time step $t \in [0, T-1]$, the input to coarse decoder consists of previous target word Y_{t-1} and mean vector Z_{mean} of encoding output Z. As the inputs are two different modalities, we first use MCF to obtain the joint representation M_t^0. The operation of coarse decoder can be described as follows:

$$\begin{aligned} M_t^0 &= MCF(Z_{mean}, Y_{t-1}) \\ h_t^0 &= tanh(w_f^0 * H_t^0 + b_f^0) \odot \sigma(w_g^0 * H_t^0 + b_g^0) \end{aligned} \tag{10}$$

where $H_t^0 = [M_{t-1}^0, M_t^0]$. $MCF(a, b)$ represents using MCF to fuse a and b. w_f^0 and w_g^0 denote convolutional filters on the 0th layer. b_f^0 and b_g^0 denote bias on the 0th layer.

Refined Decoder. In this paper, our refined decoder consists of two stages. The first stage contains three dilated convolutional layers. And the second stage only includes one dilated convolutional layer which stacks on top of first stage. We take prediction of the second stage as final description.

For the first refined decoder, we first use the output h_t^0 of coarse decoder to compute visual attention $\varphi_t^0(Z)$. The operation of first layer in this refined decoder is shown as follows:

$$\begin{aligned} M_t^1 &= w_1 * [h_t^0, \varphi_t^0(Z)] + b_1 \\ H_t^1 &= [M_{t-1}^1, M_t^1] \\ h_t^1 &= tanh(w_f^1 * H_t^1 + b_f^1) \odot \sigma(w_g^1 * H_t^1 + b_g^1) \end{aligned} \tag{11}$$

where w_1 is learnable filter to convert the channel of concatenated representation. w_f^1 and w_g^1 denote convolutional filters on the 1th layer. b_f^1 and b_g^1 denote bias on the 1th layer.

Then, operations of the next two layers in first refined decoder are shown as follows:

$$
\begin{aligned}
H_t^l &= [h_t^{l-1}, h_{t-rl}^{l-1}] \\
h_t^l &= tanh(w_f^l * H_t^l + b_f^l) \odot \sigma(w_g^l * H_t^l + b_g^l)
\end{aligned} \tag{12}
$$

where rl represents dilated rate of the layer l. h_t^{l-1} denotes the output of $(l-1)$-th layer at time step t. w_f^l and w_g^l denote convolutional filters on the layer l. b_f^l and b_g^l are bias.

For the second refined decoder, the procedures can be described as follows:

$$
\begin{aligned}
M_t^{L+1} &= h_t^L + \varphi_t^1(Z) \\
H_t^{L+1} &= [M_t^{L+1}, M_{t-2}^{L+1}] \\
Mid1 &= tanh(w_f^{L+1} * H_t^{L+1} + b_f^{L+1}) \\
Mid2 &= \sigma(w_g^{L+1} * H_t^{L+1} + b_g^{L+1}) \\
h_t^{L+1} &= Mid1 \odot Mid2
\end{aligned} \tag{13}
$$

where L denotes the number of layers of first refined stage. h_t^L denotes the output of layer L at time step t. w_f^{L+1} and w_g^{L+1} denote convolutional filters of layer $L+1$. b_f^{L+1} and b_g^{L+1} are bias. $\varphi_t^1(Z)$ represents attention computed by h_t^L.

In this section, we use the method [Yao *et al.*, 2015] to compute attention (Fig. 3). Besides, based on different size of filter and dilated rate, we use different number of zero vectors to pad the input of each layer.

Finally, the t-th generated word $\hat{Y}_t^1$ of the first refined stage and $\hat{Y}_t^2$ of the second refined stage are computed as follows:

$$
\begin{aligned}
\hat{Y}_t^1 &\sim softmax(w_p(h_t^L + \varphi_t^1(Z)) + b_p) \\
\hat{Y}_t^2 &\sim softmax(w_p(h_t^{L+1} + \varphi_t^1(Z)) + b_p)
\end{aligned} \tag{14}
$$

where w_p and b_p are learnable projection matrix and bias.

Training Loss. For each stage j, we employ a cross-entropy (XE) loss.

$$
L_{XE}^j(\theta_{0:j}) = -\sum_{t=0}^{T-1} log(p_{\theta_{0:j}}(Y_t|Y_{0:t-1}, Z)) \tag{15}
$$

where Y_t is the ground-truth word at time t, $\theta_{0:j}$ are parameters up to the j-th stage decoder, and $p_{\theta_{0:j}}(Y_t|Y_{0:t-1}, Z)$ is the output probability of word Y_t given by the previous word $Y_{0:t-1}$ and encoding output Z.

The training loss L_{train} is computed as follows:

$$
L_{train} = \beta_2 L_{recon} + \beta_1 \sum_{j=0}^{N_f} \lambda_j L_{XE}^j(\theta_{0:j}) \tag{16}
$$

where λ_j, β_1, and β_2 are hyber-parameters.

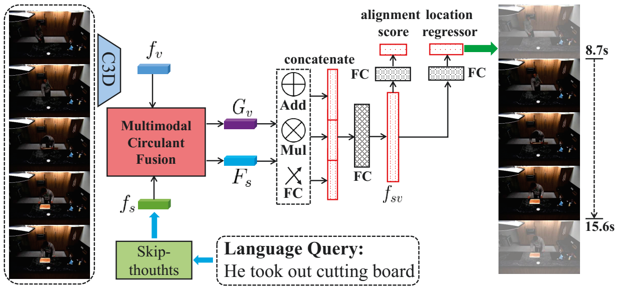


Figure 4: The architecture with MCF for the task of temporal activity localization via language query.

5 MCF for TALL

For temporal activity localization via language query, we plug MCF into the architecture of work [Gao *et al.*, 2017]. Concretely, we first feed visual feature f_v and textual feature f_s into MCF (Fig. 4). For MCF, after obtaining $F \in \mathbb{R}^d$ and $G \in \mathbb{R}^d$ by Eq. (4), we use a projection matrix $W_M \in \mathbb{R}^{d \times k}$ to convert F and G to $F_s \in \mathbb{R}^k$ and $G_v \in \mathbb{R}^k$.

$$
\begin{aligned}
F_s &= f_s W_M \\
G_v &= f_v W_M
\end{aligned} \tag{17}
$$

The following operations are same as that of the work [Gao *et al.*, 2017].

6 Evaluation on Video Captioning

We evaluate the benefit of MCF on two video captioning datasets. All results all evaluated by metrics of BLEU, METEOR, and CIDEr [Yao *et al.*, 2015].

6.1 Dataset and Implementation Details

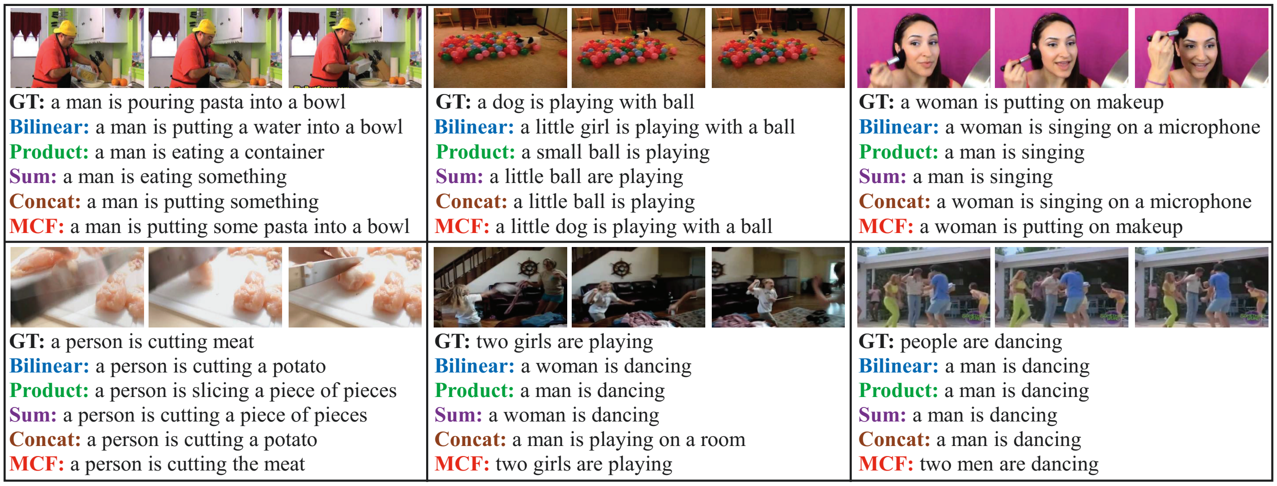
Datasets. MSVD [Chen and Dolan, 2011] contains 1,970 video clips. We use 1,200 clips for training, 100 clips for validation, and 670 clips for testing. MSRVTT [Xu *et al.*, 2016] contains 10,000 video clips. We use 6,513 clips for training, 497 clips for validation, and 2,990 clips for testing.

Video Processing. For the MSVD dataset, we select 40 frames from each video and feed them into GoogLeNet [Szegedy *et al.*, 2015] to extract a 1,024 dimensional representation. For the MSRVTT dataset, we select 20 frames from each video and feed them into GoogLeNet and ResNet-152 [He *et al.*, 2016] to extract 1,024 and 2,048 dimensional representation, respectively.

Encoding Network. In the encoder, we set the channel r (in Eq. (9)) of the encoding temporal output Z_i to 512.

Decoding Network. For the multi-stage decoder, we use five dilated layers with dilated rate 1, 1, 2, 4 and 2. The number of filter channel is set to 512, 256, 256, 512 and 512, respectively. The width of filter is set to 2. For MCF, we set $W_1 \in \mathbb{R}^{256 \times 512}$, $W_2 \in \mathbb{R}^{256 \times 512}$ (in Eq. (1)) and $W_3 \in \mathbb{R}^{256 \times 512}$.

Training Details. The vocabulary size is 12,596 for MSVD and 23,308 for MSRVTT, respectively. We use Adam optimizer with an initial learning rate of 1×10^{-3}. We empirically set β_1 and β_2 to 0.9 and 0.1, respectively. And λ_0, λ_1



GT: a man is pouring pasta into a bowl
Bilinear: a man is putting a water into a bowl
Product: a man is eating a container
Sum: a man is eating something
Concat: a man is putting something
MCF: a man is putting some pasta into a bowl

GT: a dog is playing with ball
Bilinear: a little girl is playing with a ball
Product: a small ball is playing
Sum: a little ball are playing
Concat: a little ball is playing
MCF: a little dog is playing with a ball

GT: a woman is putting on makeup
Bilinear: a woman is singing on a microphone
Product: a man is singing
Sum: a man is singing
Concat: a woman is singing on a microphone
MCF: a woman is putting on makeup

GT: a person is cutting meat
Bilinear: a person is cutting a potato
Product: a person is slicing a piece of pieces
Sum: a person is cutting a piece of pieces
Concat: a person is cutting a potato
MCF: a person is cutting the meat

GT: two girls are playing
Bilinear: a woman is dancing
Product: a man is dancing
Sum: a woman is dancing
Concat: a man is playing on a room
MCF: two girls are playing

GT: people are dancing
Bilinear: a man is dancing
Product: a man is dancing
Sum: a man is dancing
Concat: a man is dancing
MCF: two men are dancing

Figure 5: Examples of the generated video captions on MSVD. These captions are generated from multi-stage CNN which uses different multi-modal fusion method. 'GT' represents ground truth. 'Bilinear', 'Product', 'Sum' and 'Concat' represent bilinear pooling, element-wise product, element-wise sum and concatenation, respectively.

and λ_2 are set to 0.2, 0.2, and 0.6, respectively. Note that we do not conduct beam search in testing.

6.2 Experimental Results

MSVD Dataset. On MSVD dataset, we compare our method with other methods. The results are shown in Table 1.

It can be seen that our method outperforms all above methods on the metric of METEOR and CIDEr. Particularly, for the work [Song *et al.*, 2017], they use multi-layer LSTM as the decoder. Our convolutional sequential decoder outperforms the performance of hLSTMmat [Song *et al.*, 2017]. This shows that our method is effective.

Method	BLEU@4	METEOR	CIDEr
S2VT [Venugopalan *et al.*, 2015]	-	29.20	-
C3D+LSTM-E [Pan *et al.*, 2016b]	41.70	29.90	-
VGG+p-RNN [Yu *et al.*, 2016]	44.30	31.10	62.10
Tempor-attention [Yao *et al.*, 2015]	41.92	29.60	51.67
G+Bi-GRU-RCN$_1$ [Ballas *et al.*, 2016]	48.42	31.70	65.38
G+HRNE [Pan *et al.*, 2016a]	43.80	33.10	-
MAMRNN [Li *et al.*, 2017]	41.40	32.20	53.90
Boundary [Baraldi *et al.*, 2017]	42.50	32.40	63.50
G+hLSTMmat [Song *et al.*, 2017]	48.50	31.90	-
G+MVRM [Zhu *et al.*, 2017]	**49.45**	33.39	75.45
G+MCNN+MCF-element-wise product	45.65	33.56	73.86
G+MCNN+MCF-matrix multiply	46.46	**33.72**	**75.46**

Table 1: Comparison with other models on MSVD. Here 'G' denotes GoogLeNet. 'MCNN' represents our multi-stage CNN. 'MCF-element-wise product' and 'MCF-matrix multiply' mean we respectively use element-wise product and matrix multiplication in our MCF. All values are measured by percentage (%).

MSRVTT dataset. On MSRVTT dataset, we compare our method with representative methods. Results are shown in Table 2. Compared with all above methods which use a single kind of visual feature as input, our method obtains the best performance on METEOR and CIDEr metric. Besides, compared with the work [Xu *et al.*, 2017] which uses many kinds of features as input, our method outperforms its performance. This shows that the performance of our method on a single kind of visual feature is valid.

Ablation Analysis. We respectively use bilinear pooling [Fukui *et al.*, 2016], concatenation, element-wise sum and element-wise product to replace MCF used in our multi-stage CNN and keep other components of multi-stage CNN and their parameter settings unchanged. The output dimension of bilinear pooling is set to 512. The results are shown in Table 3. On MSVD and MSRVTT dataset, we can see that MCF outperforms all above fusion methods on the metric of METEOR and CIDEr. This shows that based on our convolutional architecture, MCF is effective for video captioning. Besides, through Table 1 and Table 2, we find that for our method, using matrix multiplication in MCF outperforms the performance of using element-wise product.

Method	BLEU@4	METEOR	CIDEr
MA-LSTM [Xu *et al.*, 2017]	36.5	26.5	41.0
G+LSTM [Venugopalan *et al.*, 2014]	34.6	24.6	-
C3D+SA [Yao *et al.*, 2015]	36.1	25.7	-
R+S2VT [Venugopalan *et al.*, 2015]	31.4	25.7	35.2
R+hLSTMat [Song *et al.*, 2017]	**38.3**	26.3	-
G+MCNN+MCF-matrix multiply	36.1	26.4	39.2
R+MCNN+MCF-matrix multiply	38.1	**27.2**	**42.1**
R+MCNN+MCF-element-wise product	37.7	27.1	41.3

Table 2: Comparison with other models on MSRVTT. Here 'G', 'R', and 'C' denote GoogLeNet, ResNet, and C3D. 'MCF-matrix multiply' and 'MCF-element-wise product' represent we respectively use matrix multiplication and element-wise product in our MCF. 'MCNN' represents our multi-stage CNN for video captioning.

In Fig. 5, we show some video captioning examples generated by multi-stage CNN which uses different fusion method. We can see that the captions generated by multi-stage CNN using MCF are better than that generated by multi-stage CNN using other fusion methods. Particularly, taking the first and third results as examples, our method successfully identifies 'pasta' and 'makeup', while are better than above fusion

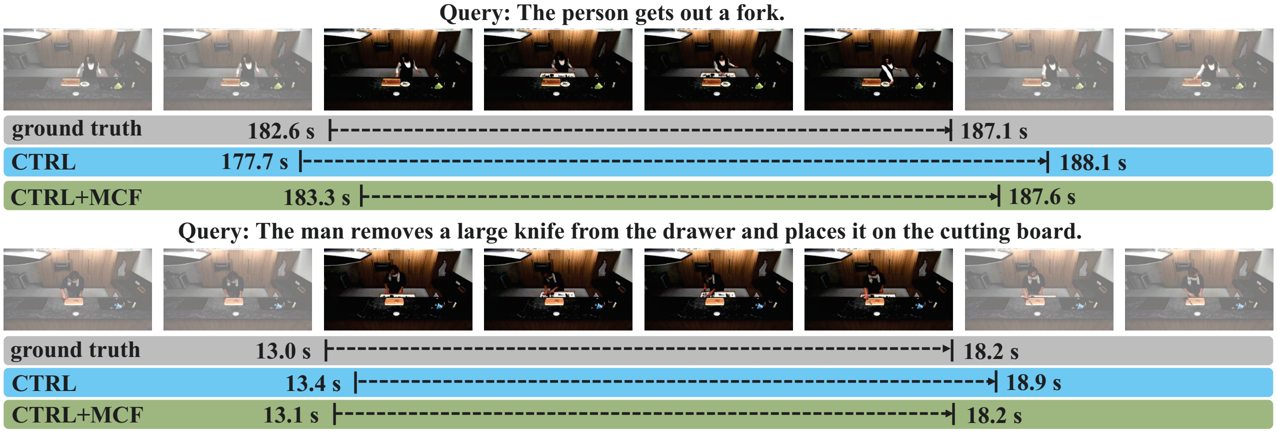


Figure 6: Examples of regression results on TACoS. The gray row shows the ground truth for the given query. The blue row shows the prediction of CTRL. The green row shows the prediction of our method.

methods. This also demonstrates that based on our architecture, MCF is an effective fusion method for video captioning.

Fusion Method (dataset)	BLEU@4	METEOR	CIDEr
Element-wise sum (MSVD)+G	**47.13**	32.71	71.20
Element-wise product (MSVD)+G	46.21	32.67	71.92
Concatenation (MSVD)+G	45.25	32.90	70.64
Bilinear Pooling (MSVD)+G	43.61	32.46	71.11
MCF-matrix multiply (MSVD)+G	46.46	**33.72**	**75.46**
Element-wise sum (MSRVTT)+R	37.5	26.4	40.0
Element-wise product (MSRVTT)+R	36.8	26.6	39.3
Concatenation (MSRVTT)+R	36.2	26.3	39.9
Bilinear Pooling (MSRVTT)+R	37.5	26.7	40.9
MCF-matrix multiply (MSRVTT)+R	**38.1**	**27.2**	**42.1**

Table 3: The results using different fusion method in our multi-stage CNN. Here 'G' and 'R' denote GoogleNet and ResNet feature. 'MCF-matrix multiply' denote using matrix multiplication in MCF.

7 Evaluation on TALL

7.1 Dataset and Implementation Details

We evaluate the benefit of MCF on TACoS dataset [Regneri *et al.*, 2013]. This dataset contains 127 videos. Each video includes two type of annotations. The first is activity labels with temporal location (start and end frame). The second is language descriptions. In total, there are 17,344 pairs of sentence and video clips. We split it in 50% for training, 25% for validation and 25% for test. In experiment, we set $W_1 \in \mathbb{R}^{96\times1024}$, $W_2 \in \mathbb{R}^{96\times1024}$ (in Eq. (1)), and $W_M \in \mathbb{R}^{96\times1024}$ (in Eq. (17)). All other parameter settings are same as those of the work [Gao *et al.*, 2017].

7.2 Experimental Results

Evaluation Metric. We employ the metric used by [Regneri *et al.*, 2013; Gao *et al.*, 2017] to compute 'R@n, IoU=m'.

Comparison with other methods. We compare our method with other methods on TACoS and report the result for $IoU \in \{0.1, 0.3, 0.5\}$ and Recall@$\{1,\ 5\}$. The results are shown in Table 4. 'Random', 'Verb' and 'Verb+Obj' are used by work [Gao *et al.*, 2017] to make contrast. 'VSA-RNN' and 'VST-STV' [Karpathy and Fei-Fei, 2015] leverage images and their descriptions to learn about their correspondences. 'CTRL-p' and 'CTRL-np' denote using parameterized and non-parameterized regression loss to train CTRL [Gao *et al.*, 2017]. And we use element-wise product in MCF.

It can be seen from Table 4 that our method outperforms all above methods on most metrics. Particularly, our method achieves a relative improvement of 4.2% on the metric R@5 with IoU=0.1. This shows that MCF is an effective fusion method for this task.

Method	R@1 IoU=0.5	R@1 IoU=0.3	R@1 IoU=0.1	R@5 IoU=0.5	R@5 IoU=0.3	R@5 IoU=0.1
Random	0.83	1.81	3.28	3.57	7.03	15.09
Verb	1.62	2.62	6.71	3.72	6.36	11.87
Verb+Obj	8.25	11.24	14.69	16.46	21.50	26.60
VSA-RNN	4.78	6.91	8.84	9.10	13.90	19.05
VSA-STV	7.56	10.77	15.01	15.50	23.92	32.82
CTRL-p	11.85	17.59	23.71	23.05	33.19	47.51
CTRL-np	**13.30**	18.32	24.32	25.42	36.69	48.73
CTRL+MCF-m	13.05	17.08	23.16	**25.74**	35.62	48.86
CTRL+MCF-p	12.53	**18.64**	**25.84**	24.73	**37.13**	**52.96**

Table 4: Comparison of different methods on TACoS. 'MCF-m' and 'MCF-p' denote using matrix multiplication and element-wise product in MCF. All values are measured by percentage (%).

In Fig. 6, we show some examples about TALL task. Particularly, taking the second result as example, our method accurately determines the start and end time for the query which includes two actions.

8 Conclusion

In this paper, we propose the Multi-modal Circulant Fusion (MCF) to combine visual and text representations. We test the MCF on video captioning and TALL tasks. Experimental results on three datasets demonstrate the effectiveness of our method.

Acknowledgments

This work was supported by the NSFC (under Grant U1509206, 61472276, U1736219).

References

[Ballas *et al.*, 2016] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *ICLR*, 2016.

[Baraldi *et al.*, 2017] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. *CVPR*, 2017.

[Chen and Dolan, 2011] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, pages 190–200, 2011.

[Chen *et al.*, 2017] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017.

[Fukui *et al.*, 2016] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *EMNLP*, pages 457–468, 2016.

[Gao *et al.*, 2017] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *ICCV*, Oct 2017.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, June 2016.

[Karpathy and Fei-Fei, 2015] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, pages 3128–3137, 2015.

[Kiros *et al.*, 2015] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *NIPS*, pages 3294–3302. 2015.

[Li *et al.*, 2017] Xuelong Li, Bin Zhao, and Xiaoqiang Lu. Mam-rnn: Multi-level attention model based rnn for video captioning. In *IJCAI*, 2017.

[Pan *et al.*, 2016a] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *CVPR*, pages 1029–1038, 2016.

[Pan *et al.*, 2016b] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *CVPR*, pages 4594–4602, 2016.

[Peng *et al.*, 2018] Yuxin Peng, Jinwei Qi, Xin Huang, and Yuxin Yuan. Ccl: Cross-modal correlation learning with multigrained fusion by hierarchical network. *IEEE Transactions on Multimedia*, 20(2):405–420, 2018.

[Regneri *et al.*, 2013] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *TACL*, 1:25–36, 2013.

[Song *et al.*, 2017] Jingkuan Song, Zhao Guo, Lianli Gao, Wu Liu, Dongxiang Zhang, and Heng Tao Shen. Hierarchical lstm with adjusted temporal attention for video captioning. *arXiv preprint arXiv:1706.01231*, 2017.

[Szegedy *et al.*, 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015.

[Tenenbaum and Freeman, 1997] Joshua B Tenenbaum and William T Freeman. Separating style and content. In *NIPS*, pages 662–668, 1997.

[Venugopalan *et al.*, 2014] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *NAACL*, 2014.

[Venugopalan *et al.*, 2015] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *ICCV*, pages 4534–4542, 2015.

[Xu *et al.*, 2016] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*, pages 5288–5296, 2016.

[Xu *et al.*, 2017] Jun Xu, Ting Yao, Yongdong Zhang, and Tao Mei. Learning multimodal attention lstm networks for video captioning. In *ACM MM*, pages 537–545. ACM, 2017.

[Yang *et al.*, 2017] Ziwei Yang, Yahong Han, and Zheng Wang. Catching the temporal regions-of-interest for video captioning. In *Proceedings of the 2017 ACM on Multimedia Conference*, pages 146–153. ACM, 2017.

[Yao *et al.*, 2015] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *ICCV*, pages 4507–4515, 2015.

[Yu *et al.*, 2016] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *CVPR*, pages 4584–4593, 2016.

[Yu *et al.*, 2017] Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. Multi-modal factorized bilinear pooling with co-attention learning for visual question answering. In *ICCV*, Oct 2017.

[Zhang *et al.*, 2016] Yuting Zhang, Kibok Lee, and Honglak Lee. Augmenting supervised neural networks with unsupervised objectives for large-scale image classification. In *ICML*, pages 612–621, 2016.

[Zhang *et al.*, 2018] Hanwang Zhang, Yulei Niu, and Shih-Fu Chang. Grounding referring expressions in images by variational context. In *CVPR*, 2018.

[Zhu *et al.*, 2017] Linchao Zhu, Zhongwen Xu, and Yi Yang. Bidirectional multirate reconstruction for temporal modeling in videos. In *CVPR*, July 2017.