

Sequence-to-Sequence Learning via Shared Latent Representation

Xu Shen,[†] Xinmei Tian,[†] Jun Xing,[★] Yong Rui,[‡] Dacheng Tao^{*}

[†]CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application Systems,
University of Science and Technology of China, China

[★] Institute for Creative Technologies, University of Southern California

[‡] Lenovo Research

^{*} UBTECH Sydney Artificial Intelligence Institute, SIT, FEIT, University of Sydney, Australia
shenxu@mail.ustc.edu.cn, xinmei@ustc.edu.cn, junxnui@gmail.com,
yongrui@lenovo.com, dacheng.tao@sydney.edu.au

Abstract

Sequence-to-sequence learning is a popular research area in deep learning, such as video captioning and speech recognition. Existing methods model this learning as a mapping process by first encoding the input sequence to a fixed-sized vector, followed by decoding the target sequence from the vector. Although simple and intuitive, such mapping model is *task-specific*, unable to be directly used for different tasks. In this paper, we propose a star-like framework for general and flexible sequence-to-sequence learning, where different types of media contents (the peripheral nodes) could be encoded to and decoded from a *shared latent representation* (SLR) (the central node). This is inspired by the fact that human brain could learn and express an abstract concept in different ways. The media-invariant property of SLR could be seen as a high-level regularization on the intermediate vector, enforcing it to not only capture the latent representation intra each individual media like the auto-encoders, but also their transitions like the mapping models. Moreover, the SLR model is *content-specific*, which means it only needs to be trained once for a dataset, while used for different tasks. We show how to train a SLR model via dropout and use it for different sequence-to-sequence tasks. Our SLR model is validated on the Youtube2Text and MSR-VTT datasets, achieving superior performance on video-to-sentence task, and the first sentence-to-video results.

Introduction

Sequence-to-sequence learning recently gains enormous attention both academically and commercially, and has been successfully used to develop various practical and powerful applications, such as machine translation (Sutskever, Vinyals, and Le 2014), speech recognition (Graves, Mohamed, and Hinton 2013) and video captioning (Venugopalan et al. 2015a). This has been greatly advanced with the increasing power of recurrent neural networks, especially the LSTM (Hochreiter and Schmidhuber 1997), for sequential processing. Despite the diversity of sequences, existing methods share a common mapping framework (Figure 1a), where the input sequence is first encoded to a fixed-sized vector, then the vector is decoded to the output se-

Copyright © 2018, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

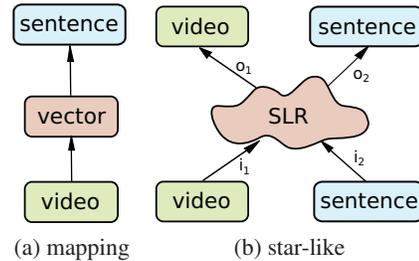


Figure 1: Concept illustration of different models. (a) the mapping model first encodes the video into a fixed-sized vector, followed by decoding a sentence from the vector. (b) the star-like model learns a shared latent representation (SLR) among the video and sentence, and could reconstruct them from the SLR. The star-like mode could be used for different learning tasks, such as video-to-sentence learning when activating the video input (i_1) and sentence output (o_2).

quence. However, these mapping models are task-specific, which means we need to train different models separately (e.g. English-to-Chinese and Chinese-to-English). It may bring more computation and time cost and deployment difficulty as the model number increases.

In human brain, information collected and extracted from different channels (e.g. visual and auditory information) are first mapped to a common inferior front cortex area, then this cortex area sends signals to the control cortex for further processing to drive various motor controls (Bear, Connors, and Paradiso 1996). This motivates us to propose a star-like framework for sequence-to-sequence learning (Figure 1b). In the framework, the peripheral nodes are the various media sequences (e.g. text, audio and video) that could be encoded to and decoded from the central node called the shared latent representation (SLR). In particular, the SLR could be learned from a single or multiple media sources, and generate the same (e.g. sentence-to-sentence) or different (video-to-sentence) medias. The star structure decouples the input from output, enabling a general and flexible framework for various sequence learning applications. Different from

the task-specific models, the SLR-based model is content-specific. It only needs to be trained once on a dataset, while used for different tasks by specifying the target input and output sequences.

The star framework contains three components: multi-source encoding, SLR learning and multi-source decoding. The media sources are first processed independently to extract their high-level features, which are then concatenated and encoded to get the SLR distribution via variational auto-encoder (Kingma and Welling 2013). From the distribution, a random SLR is sampled and decoded to reconstruct different media sequences separately. In particular, the input and output source channels could be manually turned on or off according to the target task, e.g. turn on the input video and output sentence channel for video captioning.

The goal is to train the SLR to be media-invariant. For instance, we should get close SLRs from either a video or its captions. To train such a model, we extend the dropout technique on unit-level (Srivastava et al. 2014) to channel-level to gradually reduce the dependency on each input source when inferring the SLR. Structure-wise, a SLR model combines multiple sequence-to-sequence variational auto-encoders. Thus, the dropout could also be interpreted as performing model averaging.

In this paper, we focus on the study of video and sentence sequences, particularly the video captioning task (Venugopalan et al. 2015a; Pan et al. 2016; Yao et al. 2015), to validate our SLR model. In summary, this paper makes the following contributions:

- We present a star-like model for general and flexible sequence-to-sequence learnings. It only needs to be trained once on a dataset, while used for different tasks.
- We propose the novel concept of shared latent representation that captures the high-level information shared among different media sources.
- We show how to train a SLR model via channel-level dropout.
- Our proposed model is evaluated on the Youtube2Text and MSR-VTT datasets, achieving state-of-the-art performance on video-to-sentence task, and the first sentence-to-video results.

Related Work

Although Deep Neural Networks work well whenever large labeled training sets are available, they cannot be used to map sequences to sequences. With the advance of recurrent neural networks (RNNs), especially LSTMs, a large amount of work on applications of sequence-to-sequence learning came up. Sutskever et al. (Sutskever, Vinyals, and Le 2014) presented a general end-to-end approach to sequence learning, which uses a multilayered LSTM to encode the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. Following this approach, various applications were developed, such as conversation modeling (Vinyals and Le 2015) and video captioning (Venugopalan et al. 2015a). Most recently, Ha and Eck (Ha and

Eck) exploited the sequence learning approach to design the sketch-rnn, a RNN able to construct stroke-based drawings of common objects. With the use of variational auto-encoder (Kingma and Welling 2013), the sketch-rnn could generate various drawings. Instead of encoding the input sequence with LSTMs like (Venugopalan et al. 2015a), (Pan et al. 2016) adopted a 3D-CNN to extract the high-level features of each video frame, followed by mean pooling them to get the global information. In particular, when the input and output sequences are synchronized, such as speech recognition (Graves et al. 2006) and lipreading (Assael et al. 2016) tasks, the learning could be conducted locally in real-time (e.g. via connectionist temporal classification) without encoding the whole input sequence in advance.

Attention (Bahdanau, Cho, and Bengio 2014) is an effective mechanism to enhance the sequence processing by selectively focusing on parts of the input sequence, as exemplified by the neural machine translation (Luong, Pham, and Manning 2015) and video description (Yao et al. 2015). It has been shown particularly powerful to handle long sequence, such as the question answering task (Hermann et al. 2015). Besides, attention could also be used to transform the fix-sized inputs or outputs into sequences. By mimicking the foveation of the human eye, (Gregor et al. 2015; Xu et al. 2015) used the spatial attention mechanism to generate and captioning images.

Multimodal studies learning from multiple sources, such as speech recognition from the visual and audio inputs (Ngiam et al. 2011), and video description based on the audio and broad topic (category) information (Ramanishka et al. 2016). With more clues from different sources, the multimodal learning usually achieves better performance than a single-modal. Instead of learning from multiple sources, we explore the shared high-level concept among them.

Preliminaries

Variational Auto-Encoder

A VAE consists of two networks that encode a data sample x to a latent representation z and decode the latent representation back to data space, respectively:

$$\begin{aligned} z &\sim \mathbf{encoder}(x) = q(z|x) \\ \tilde{x} &\sim \mathbf{decoder}(z) = p(x|z) \end{aligned} \quad (1)$$

The VAE regularizes the encoder by imposing a prior over the latent distribution $p(z)$, typically $z \sim \mathcal{N}(0, I)$, where I is the identity matrix. The VAE loss is defined as:

$$\begin{aligned} \mathcal{L}_{VAE} &= -\mathbb{E}_{q(z|x)} \left[\log \frac{p(x|z)p(z)}{q(z|x)} \right] = \mathcal{L}_{like} + \mathcal{L}_{prior} \\ \mathcal{L}_{like} &= -\mathbb{E}_{q(z|x)} [\log p(x|z)] \\ \mathcal{L}_{prior} &= D_{KL}(q(z|x)||p(z)), \end{aligned} \quad (2)$$

where \mathcal{L}_{like} measures the expected negative reconstruction error, and \mathcal{L}_{prior} (the KL divergence of the approximate posterior from the prior) acts as a prior regularizer.

Long Short Term Memory

Recurrent neural networks (RNNs) are able to process input sequences of arbitrary length via the recursive application of a transition function. However, it suffers from the exploding or vanishing gradients problem (Bengio, Simard, and Frasconi 1994). The LSTM architecture (Hochreiter and Schmidhuber 1997) addresses this problem of learning long-term dependencies by introducing a memory cell c_t that is able to preserve state over long periods of time, and an input gate i_t , a forget gate f_t and an output gate o_t to control the information flow (Zaremba and Sutskever 2014):

$$\begin{aligned}
 i_t &= \sigma \left(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)} \right) \\
 f_t &= \sigma \left(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)} \right) \\
 o_t &= \sigma \left(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)} \right) \\
 u_t &= \tanh \left(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)} \right) \\
 c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\
 h_t &= o_t \odot \tanh(c_t),
 \end{aligned} \tag{3}$$

where x_t and h_t are the input and hidden state at the current time step, σ is the logistic sigmoid function and \odot is elementwise multiplication. Intuitively, the forget gate controls the extent to which the previous memory cell is forgotten, the input gate controls how much each unit is updated, and the output gate controls the exposure of the internal memory state. The model can learn to represent information over multiple time scales by controlling the gating variables.

Dropout

Dropout is a recently introduced algorithm for training neural networks (Srivastava et al. 2014). In its simplest form, on each presentation of each training example, each feature detector unit is deleted randomly with probability 0.5. The remaining weights are trained by backpropagation. The procedure is repeated for each example and each training epoch, sharing the weights at each iteration. After the training phase is completed, predictions are produced by halving all the weights. The motivation and intuition behind the algorithm is to prevent overfitting associated with the coadaptation of feature detectors. By randomly dropping out neurons, the procedure prevents any neuron from relying excessively on the output of any other neuron, forcing it instead to rely on the population behavior of its inputs.

Method

We elaborate the general SLR model and its learning method in this section. For simplicity, we only discuss the common case of two different media sources like video and sentence shown in Figure 1b. Both the model and learning method could be easily generalized to other cases.

Framework

As illustrated in Figure 2, the SLR model mainly consists of 3 stages: multi-source encoding, SLR learning and multi-source decoding, and could be manually configured to han-

dle different target tasks. We elaborate these components below.

Multi-source Encoding During the first stage, we extract the high-level feature x_i of each media source \mathcal{S}^i separately, and concatenate them into a hidden state x :

$$\begin{aligned}
 x_i &= \text{encoder}_i(\mathcal{S}^i), i \in \{1, 2\} \\
 x &= [x_1; x_2]
 \end{aligned} \tag{4}$$

where each encoder_i of media source \mathcal{S}^i may differ from one another, such as encoding the videos via mean pooling of the high level CNN features (Pan et al. 2016) and the sentences via LSTMs (Sutskever, Vinyals, and Le 2014).

SLR Learning We use a variational auto-encoder (Kingma and Welling 2013) to learn the distribution of the SLR. Specifically, the concatenated hidden state x is projected into two vectors μ and $\hat{\sigma}$, each of size N_z , using a fully connected layer ($W_\mu, W_\sigma \in \mathcal{R}^{N_z \times N_x}$). We convert $\hat{\sigma}$ into a non-negative standard deviation parameter σ using an exponential operation. We use μ and σ , along with $\mathcal{N}(0, I)$, a vector of IID Gaussian variables of size N_z , to construct a random vector, $z \in \mathbb{R}^{N_z}$:

$$\begin{aligned}
 \mu &= W_\mu x + b_\mu \\
 \hat{\sigma} &= W_\sigma x + b_\sigma \\
 \sigma &= \exp\left(\frac{\hat{\sigma}}{2}\right) \\
 z &= \mu + \sigma \odot \mathcal{N}(0, I)
 \end{aligned} \tag{5}$$

Under this encoding scheme, the SLR vector z is not a deterministic output, but a random vector conditioned on the input media sources.

Multi-source Decoding In the last stage, we adopt a LSTM structure to decode each target sequence from the sampled SLR:

$$\begin{aligned}
 [h_0^i; c_0^i] &= \tanh(W_z^i z + b_z^i) \\
 [h_t^i; c_t^i] &= \text{LSTM}_i(\mathcal{S}_{t-1}^i, [h_{t-1}^i; c_{t-1}^i]) \\
 \mathcal{S}_t^i &= \text{decoder}_i(h_t^i)
 \end{aligned} \tag{6}$$

where the initial hidden and cell states $[h_0^i; c_0^i]$ of the LSTMs are the output of a single layer network conditional on z , the LSTM is defined in Equation 3, and i represents the identity of different target sequences. Similarly, each decoder_i are also media-dependent, such as a softmax layer to predict the probabilities of each word in sentence and a non-linear mapping layer ($f(W h_t + b)$) mapping to the pixels of each frame for a video sequence.

Model Configuration The input and output source channels could be manually turned on or off to achieve different applications (at least one input channel is active). In particular, when an input channel \mathcal{S}^i is inactive, we set its high-level feature x_i in Equation 4 to all zeros. The configuration

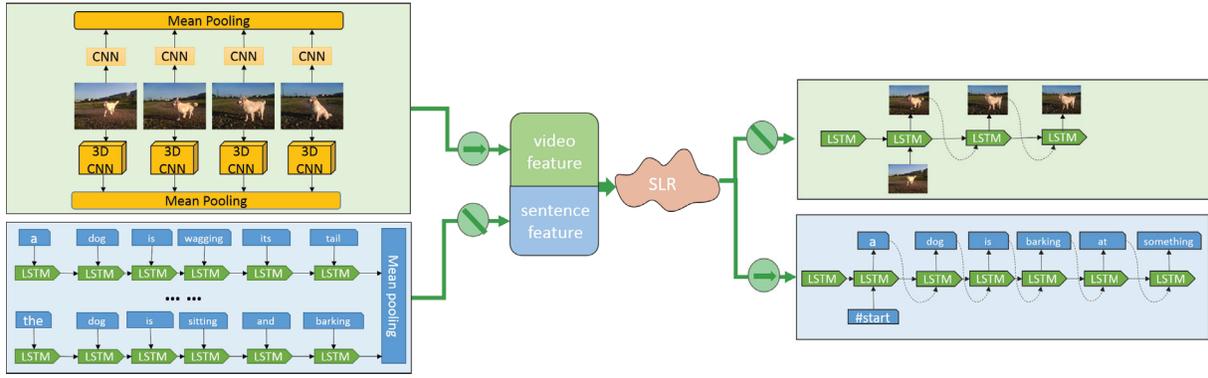


Figure 2: The SLR model overview. The sentence and video sources are first processed independently to get their high-level features (blue and green boxes in the middle) via LSTM and mean pooling methods. The features are then concatenated and encoded to get the SLR distribution via a variational auto-encoder. Finally, a random SLR, sampled from the distribution, is decoded via LSTMs to reconstruct different media sequences separately. The input and output media channels could be manually turned on or off (green circles) to handle different tasks, such as turning on the input video and output sentence for video captioning.

could also be made adaptive based on the context. For example, trained on a Chinese-English language dataset, our model could be used for bi-directional translations, and the channels could be dynamically turned on or off as people talk to each other.

Learning

The overall learning could be divided into three phases – full-model learning (FL), partial-model learning (PL), and testing. In the FL phase, all the input and output media channels are active, while in the PL phase, only part of the input and output channels are available. Both the learning phases could be trained end-to-end, similar to the traditional sequence-to-sequence variational auto-encoders. In the testing phase, the states of input and output channels are configured according to the target task(s) of the dataset.

Suppose $\mathcal{S}^a = \{\mathcal{S}_i^a\}$ and $\mathcal{S}^b = \{\mathcal{S}_j^b\}$ are two different media sources in a dataset, and their matching correspondences are known. In particular, such correspondences may be inter-media (e.g. \mathcal{S}_i^a is similar to \mathcal{S}_j^b) and intra-media (e.g. \mathcal{S}_i^a is similar to \mathcal{S}_j^a). For example, in Youtube2Text dataset, a video could be described by multiple similar sentences (e.g. Figure 2).

Loss This whole SLR model could be trained end-to-end. The training loss is similar as the variational auto-encoder described in Equation 2, except that the reconstruction error \mathcal{L}_{like} is composed of multiple output channels:

$$\mathcal{L}_{like} = \lambda_a \mathcal{L}_a + \lambda_b \mathcal{L}_b \quad (7)$$

where λ_a and λ_b are weighting parameters of reconstruction error \mathcal{L}_a and \mathcal{L}_b , respectively. Please refer to the experiment for the detailed examples.

Full-model Learning In the FL phase, we randomly input a sequence-pair $(\mathcal{S}_i^a, \mathcal{S}_j^b)$, and train the model by decoding

loss from both \mathcal{S}_i^a and \mathcal{S}_j^b , where \mathcal{S}_i^a and \mathcal{S}_j^b are the intra-media matching of \mathcal{S}_i^a and \mathcal{S}_j^b , respectively, subject to availability. If there is no matching sequence of \mathcal{S}_i^a or \mathcal{S}_j^b , we simply set $\mathcal{S}_i^a = \mathcal{S}_i^a$ or $\mathcal{S}_j^b = \mathcal{S}_j^b$. Using a similar, instead of same, output sequence for loss calculation could prevent overfitting and enhance the distribution learning capability.

Partial-model Learning In order to train the SLR model for general-purpose, we randomly dropout the input and output channels. To keep a smooth transition from the FL to PL phase, we select input from $(\mathcal{S}_i^a, \mathcal{S}_j^b)$, $(\mathcal{S}_i^a, \mathbf{0})$ or $(\mathbf{0}, \mathcal{S}_j^b)$ with probabilities of 0.5, 0.25 and 0.25, respectively, and keep all the output channels. After enough training steps, we configure the input and output channels based on the testing task(s) for target training, which helps achieve better performance in testing.

Testing Finally, we input the given media sequence(s) (e.g. \mathcal{S}_i^a) to the SLR model, and decode the target output sequence(s) (e.g. \mathcal{S}_j^b) from the sampled SLR.

Experiments

In this section, we apply our SLR framework mainly on video and sentence generation task (shown in Fig. 2). Models are tested on the Microsoft Research Video Description Corpus (YouTube2Text) (Guadarrama et al. 2013) and the MSRVT dataset (Xu et al. 2016). In video to sentence and sentence to sentence tasks, generated sentences are evaluated by BLEU@N (Papineni et al. 2001) and METEOR (Lavie and Agarwal 2005). In video to video and sentence to video tasks, generated videos are evaluated by mean squared error (MSE) between ground truth video frames and predicted video frames. All the examples presented in the following sections are randomly drawn from YouTube2Text dataset. We will detail the experimental settings, compared models and results in the following sections.

Experimental Settings

Data The YouTube2Text dataset contains 1,970 videos and about 40 English sentences for each video. Following previous works, we randomly split 1,200 videos for training, 100 for validation and 670 videos for testing as in (Yao et al. 2015). The MSRVTT dataset contains 6,513 videos for training, 497 videos for validation and 2,990 videos for testing and each video corresponds to 20 descriptions. On both datasets, we take the output of 4096-way fc6 layer from the 19-layer VGG (Simonyan and Zisserman 2014) and 4096-way fc6 layer output of C3D(Tran et al. 2015) as a 2D/3D CNN representation of the frames, respectively. Each video is split into clips with 16 continuous frames and with a maximum of 45 clips. Each sentence is represented by a vector of words (at most 35 words), and each word is preprocessed by lowercasing and encoded by a one-hot vector. In our experiments, we use all the words of the training set to build the vocabulary. Specifically, we have 9855 words for YouTube2Text and 23668 words for MSRVTT.

Model As shown in Fig. 2, sentence features (h_s) are encoded by LSTMs with 512 hidden units. The mean-pooling of the concatenated 2D/3D CNN features (9216D) of input video are first embedded into 512D by a fully-connected layer with \tanh activation:

$$h_v = \tanh(W_{emb}h_{concat} + b_{emb}) \quad (8)$$

i.e. $x_1 = h_s \in R^{512}$, $x_2 = h_v \in R^{512}$ and $x \in R^{1024}$ in Eq. 4. Additionally, in order to avoid the scale imbalance between video representation and sentence representation, we apply L2 normalization on x_1 and x_2 before concatenation. Then x is mapped to a 512D vector by variational autoencoder in Eq. 5 ($\mu, \sigma, z \in R^{512}$). We use LSTM with 512 hidden units ($h_t^i, c_t^i \in R^{512}$ in Eq. 6) as decoders for both sentence and video. Specifically, decode LSTM for video are required to reconstruct the down sampled frames (target frames are down sampled to 36×64 on Youtube2Text and 64×48 on MSRVTT), and decode LSTM for sentence are required to predict the corresponding probabilities of each word in the vocabulary.

Learning As aforementioned, we first do full-model learning, then do partial-model learning, and finally comes to testing. We use an initial learning rate 0.0001 on the YouTube2Text and 0.001 learning rate on the MSRVTT dataset for the full-model learning stage, and decay the learning rate by 10 in the partial-model learning stage. The full model learning stage is trained for 20 epochs on the YouTube2Text and 60 epochs on the MSRVTT dataset. The partial model learning stage is trained for 80/40 epochs on the YouTube2Text and the MSRVTT dataset, respectively. Finally, we fine-tune the learned model on the specific task (i.e. video-to-sentence) for 20 epochs. We train the model by Adam optimizer with 100 mini batch size. Gradients of parameters are clipped to maximum 35 L2 norm. In the full model learning stage, we use a weighted sum of sentence generation loss, video generation loss, and latent loss of the variational auto-encoder:

$$\mathcal{L} = \lambda_v \mathcal{L}_{video} + \lambda_s \mathcal{L}_{sentence} + \lambda_p \mathcal{L}_{prior} \quad (9)$$

where $\lambda_v = 1$, $\lambda_s = 1$, $\lambda_p = 0.01$, \mathcal{L}_{video} is computed by average Euclidean distance between predicted frames and

ground truth frames; $\mathcal{L}_{sentence}$ is computed by softmax loss of predicted probabilities; \mathcal{L}_{prior} is computed by the KL divergence between latent vector and the normal distribution (Eq. 2). In the partial learning stage, we randomly dropout the input video/sentence feature. In the testing stage, we set $\lambda_v = 0$ for sentence generation tasks and $\lambda_s = 0$ for video generation tasks.

Compared models

- Long-Short Term Memory(LSTM): Directly decode target videos/sentences from the representation of input videos/ sentences by LSTM.
- Soft Attention (SA)(Yao et al. 2015): A weighted attention mechanism is used to determine the focused frames when decoding sentences from input videos.
- Sequence-to-Sequence - Video to Text (S2VT) (Venugopalan et al. 2015b): Encoding and decoding of inputs and word representations are learned jointly via a single LSTM.
- Long Short Term Memory with visual semantic embedding (LSTM-E)(Pan et al. 2016): A relevance loss between embedded video and sentence representation is introduced and then is jointly learned with the coherence loss.
- Sequence-to-Sequence Learning with variational auto-encoder (S2S+VAE): A distribution is learned from input representation by variational auto-encoder first, then target videos/ sentences are decoded from that distribution.
- Sequence-to-Sequence Learning via shared latent representation (S2S+SLR): A shared latent representation is learned from both videos and sentences first, then both videos and sentences are decoded from the latent representation.

Model	METEOR	BLEU @1	BLEU @2	BLEU @3	BLEU @4
LSTM	26.9	69.8	53.3	42.1	31.2
SA	29.6	80.0	64.7	52.6	42.2
S2VT	29.8	-	-	-	-
LSTM-E	31.0	78.8	66.0	55.4	45.3
S2S+VAE	30.2	75.3	63.3	53.9	43.5
S2S+SLR	33.4	79.8	67.9	57.8	47.2

Table 1: METEOR and BLEU@N scores for video captioning (video-to-sentence) models on Youtube2Text. All values are reported as percentage.

Video-to-Sentence

In this section, we compare different models for the video to sentence task. The results on Youtube2Text and MSRVTT are summarized in Table 1 and Table 2. Results of compared models are copied from published literatures (Venugopalan et al. 2015b; Yao et al. 2015; Pan et al. 2016; Xu et al. 2016). The improvement of S2S+SLR method compared with LSTM-E indicates that an explicitly shared

Model	METEOR	BLEU @1	BLEU @2	BLEU @3	BLEU @4
LSTM	29.5	79.9	64.9	52.1	40.1
SA	29.9	81.5	65.0	52.5	40.5
S2S+VAE	30.3	79.3	65.0	52.9	41.5
S2S+SLR	32.8	80.8	66.1	54.3	44.1

Table 2: METEOR and BLEU@N scores for video captioning (video-to-sentence) models on MSRVT. All values are reported as percentage.

latent representation is better for inferring descriptions. Additionally, we can see that S2S+VAE also outperforms mapping models, e.g. LSTM, SA and S2VT. This phenomenon shows that a generative model is better for modelling the description diversity among different persons. Fig. 3 shows some examples. The interesting point of the results is that S2S+SLR tends to generate more abstract concepts in the descriptions. Specifically, the outputs of SLR model are not restricted to the content and the ground truth descriptions, but some interesting inference and more general perceptions.

Video-to-Video

The mean squared error (MSE) between ground truth video frames and predicted video frames generated by different models are shown in Table 3. Because all other models are designed for video to sentence task only, we only consider LSTM model as a baseline here. As generating video frames from a fixed representation only is quite challenging and the results divers a lot. We feed the first ground truth frame into the decoding LSTM to ensure reasonable and consistent predictions are generated. The results in these two tables show that shared latent representation learned from both video and sentence representations can help to solve this challenging task.

Model	MSE(YouTube2Text)	MSE(MSRVTT)
LSTM	460.5	471.2
S2S+VAE	450.3	462.3
S2S+SLR	398.9	423.4

Table 3: MSE scores for video reconstruction (video-to-video) models on Youtube2Text and MSRVT.

Sentence-to-Sentence

For the sentence to sentence task, sentences are encoded by an encoding LSTM and decoded by another LSTM. The encoding, decoding and embedding are learned jointly. For each target sentence, the input sentence is randomly sampled from other descriptions of the same video. Quantitative results are presented in Table 4 and Table 5. As all other models are designed for video to sentence task only, we only consider LSTM model as a baseline here. We can see that shared latent representation learned from video channel also helps to decode similar sentences. Some examples of LSTM and our model are presented in Fig. 4. We can see that model

tends to generate more brief and more abstract sentences with shared latent representation learning.

Model	METEOR	BLEU @1	BLEU @2	BLEU @3	BLEU @4
LSTM	36.3	82.0	71.5	61.5	49.9
S2S+VAE	37.2	83.2	73.1	62.8	52.4
S2S+SLR	39.4	84.6	75.4	64.9	54.5

Table 4: BLEU@N and METEOR scores for sentence-to-sentence models on Youtube2Text. All values are reported as percentage.

Model	METEOR	BLEU @1	BLEU @2	BLEU @3	BLEU @4
LSTM	33.5	80.9	69.2	56.3	46.1
S2S+VAE	34.8	81.3	70.6	57.8	47.5
S2S+SLR	35.7	82.3	71.4	58.9	48.6

Table 5: BLEU@N and METEOR scores for sentence-to-sentence models on MSRVT. All values are reported as percentage.

Sentence-to-Video

Since all other models are designed for video to sentence task only, we only consider LSTM model as a baseline here. Sentence to video is the most challenging task among video and sentence learning tasks, because sentence only contains much less information compared with video. In this situation, latent shared representation from both video and sentence helps a lot compared with other models, achieving more than 10% improvement as shown in Table 6. Consequently, latent shared representation learning provides a promising way to solve the sentence to video problem. Some examples on YouTube2Text are shown in Fig. 5. Though the predicted video frames are a little bit blur, we can still see that the SLR model begin to learn some general things in the sentence.

Model	MSE(YouTube2Text)	MSE(MSRVTT)
LSTM	481.4	490.6
S2S+VAE	472.3	481.0
S2S+SLR	417.6	431.2

Table 6: MSE scores for sentence-to-video models on Youtube2Text and MSRVT.

Conclusion

This paper proposed a novel sequence-to-sequence learning method. Instead of formulating the learning as a one-to-one mapping process, we learned the shared latent representation among multiple sources and how each media source could be encoded to and decoded from it. This was inspired by the fact that human brain could learn and express an abstract

	LSTM: a women is adding flour to water S2S+SLR: a women is mixing ingredients	GroundTruth: a women is mixing flour and water A women pours flour into bowl of water A women is mixing some ingredients in a bowl
	LSTM: a cat is meowing S2S+SLR: a cat is playing with a spot	GroundTruth: A cat is looking at a wall A white cat is meowing at a spot of light on the wall
	LSTM: a man is doing pushup S2S+SLR: a man is dancing	GroundTruth: A man is doing exercise a woman is seated watching a man do pushups A man is working out
	LSTM: a women is slicing S2S+SLR: a man is cutting a piece of meat	GroundTruth: The women is cutting beacon A women is cutting flesh with knife
	LSTM: a dog is barking S2S+SLR: a dog is walking	GroundTruth: A dog is running around A dog barks and runs behind a tree

Figure 3: Examples of video to sentence samples. The videos are represented by sampled frames. The output sentences are generated by LSTM and our S2S+SLR model. Ground truth sentences are randomly selected.

Input	Output	Ground Truth
A man is lifting weights	LSTM: A man lifts weights S2S+SLR: a man is doing exercise	A man is lifting weights A man is weight lifting A man is doing exercises Someone is working out A man is lifting barbells over his head The man lifted weights above his head
A man is adding pepper to sliced carrots	LSTM: a man is seasoning carrots S2S+SLR: a man is cutting a carrot	A man is seasoning carrots A man is adding pepper to sliced carrots A person is pouring seasoning on to a tray of carrots The chef sprinkled seasonings on the carrots A man is mixing carrot A man is sprinkling black paper on carrot pieces
A lady making noodles in the kitchen	LSTM: a women is boiling noodles S2S+SLR: a women is cooking	A women puts noodles into boiling water A women is making noodles Noodles are boiling A women is adding pasta to a pot of water A female making how to make noodle dish A lady making noodles in the kitchen

Figure 4: Examples of generating sentences from given sentences. The input sentences are randomly sampled from descriptions of the same video. The output sentences are generated by LSTM and our S2S+SLR model. Ground truth sentences are randomly selected from descriptions of the same video.



Figure 5: Examples of sentence-to-video samples on YouTube2Text. The first row shows ground truth video frames. The second and third rows present video frames predicted by S2S+SLR and LSTM models. As this task is too challenging, the generated frames are a little blur. We will keep working on this challenging topic in the future.

concept in different ways. In contrast to task-specific mapping models, our star-like model could be used for different tasks with only one training. Thus, it could be manually configured to handle a specific task, or set adaptive in a changing environment, e.g. multi-directional translation during a conversation. Our model was validated on the challenging video-sentence dataset, and achieved superior results in different tasks. We believe our model could be directly applied to other media sources, including both sequence (e.g. audio) and fix-sized vectors (e.g. images).

Acknowledgments

This work was supported by National Key Research and Development Program of China 2017YFB1002203, NSFC No.61572451, No.61390514, and No. 61632019, Youth Innovation Promotion Association CAS CX2100060016, Fok Ying Tung Education Foundation WF2100060004, and Australian Research Council Projects FL-170100117, DP-180103424, DP-140102164, LP-150100671.

References

- Assael, Y. M.; Shillingford, B.; Whiteson, S.; and de Freitas, N. 2016. Lipnet: Sentence-level lipreading. *CoRR* abs/1611.01599.
- Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473.
- Bear, M. F.; Connors, B. W.; and Paradiso, M. A. 1996. *Neuroscience: Exploring the Brain*. Williams & Wilkins.
- Bengio, Y.; Simard, P.; and Frasconi, P. 1994. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.* 157–166.
- Graves, A.; Fernández, S.; Gomez, F.; and Schmidhuber, J. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 369–376.
- Graves, A.; Mohamed, A.; and Hinton, G. E. 2013. Speech recognition with deep recurrent neural networks. *CoRR* abs/1303.5778.
- Gregor, K.; Danihelka, I.; Graves, A.; and Wierstra, D. 2015. DRAW: A recurrent neural network for image generation. *CoRR* abs/1502.04623.
- Guadarrama, S.; Krishnamoorthy, N.; Malkarnenkar, G.; Venugopalan, S.; Mooney, R.; Darrell, T.; and Saenko, K. 2013. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *ICCV*.
- Ha, D., and Eck, D. A neural representation of sketch drawings. *arxiv preprint*.
- Hermann, K. M.; Kociský, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. *CoRR* abs/1506.03340.
- Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Comput.* 1735–1780.
- Kingma, D. P., and Welling, M. 2013. Auto-Encoding Variational Bayes. *ArXiv e-prints*.
- Lavie, A., and Agarwal, A. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of ACL Workshop*, 65–72.
- Luong, M.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. *CoRR* abs/1508.04025.
- Ngiam, J.; Khosla, A.; Kim, M.; Nam, J.; Lee, H.; and Ng, A. Y. 2011. Multimodal deep learning. In *ICML*, 689–696.
- Pan, Y.; Mei, T.; Yao, T.; Li, H.; and Rui, Y. 2016. Jointly modeling embedding and translation to bridge video and language. *CVPR*.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2001. Bleu: A method for automatic evaluation of machine translation. In *ACL*, 311–318. Association for Computational Linguistics.
- Ramanishka, V.; Das, A.; Park, D. H.; Venugopalan, S.; Hendricks, L. A.; Rohrbach, M.; and Saenko, K. 2016. Multimodal video description. In *ACM MM*, 1092–1096.
- Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.
- Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 1929–1958.
- Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. *CoRR* abs/1409.3215.
- Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. *ICCV*.
- Venugopalan, S.; Rohrbach, M.; Donahue, J.; Mooney, R. J.; Darrell, T.; and Saenko, K. 2015a. Sequence to sequence - video to text. *CoRR* abs/1505.00487.
- Venugopalan, S.; Rohrbach, M.; Donahue, J.; Mooney, R. J.; Darrell, T.; and Saenko, K. 2015b. Sequence to sequence - video to text. In *ICCV*, 4534–4542.
- Vinyals, O., and Le, Q. V. 2015. A neural conversational model. *CoRR* abs/1506.05869.
- Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A. C.; Salakhutdinov, R.; Zemel, R. S.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. *CoRR* abs/1502.03044.
- Xu, J.; Mei, T.; Yao, T.; and Rui, Y. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Yao, L.; Torabi, A.; Cho, K.; Ballas, N.; Pal, C.; Larochelle, H.; and Courville, A. 2015. Describing videos by exploiting temporal structure. In *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE.
- Zaremba, W., and Sutskever, I. 2014. Learning to execute. *CoRR* abs/1410.4615.