Long-term Recurrent Convolutional Networks for Visual Recognition and Description

Jeff Donahue[⋆] Lisa Anne Hendricks[⋆] Sergio Guadarrama[⋆] Marcus Rohrbach[⋆*]
Subhashini Venugopalan[†] Kate Saenko[‡] Trevor Darrell[⋆*]
[†]UT Austin, Austin, TX, vsub@cs.utexas.edu
[‡]UMass Lowell, Lowell, MA, saenko@cs.uml.edu
[⋆]UC Berkeley, [*]ICSI, Berkeley, CA, {jdonahue, lisa_anne, sguada, rohrbach, trevor}@eecs.berkeley.edu

Abstract

Models based on deep convolutional networks have dominated recent image interpretation tasks; we investigate whether models which are also recurrent, or "temporally deep", are effective for tasks involving sequences, visual and otherwise. We develop a novel recurrent convolutional architecture suitable for large-scale visual learning which is end-to-end trainable, and demonstrate the value of these models on benchmark video recognition tasks, image description and retrieval problems, and video narration challenges. In contrast to current models which assume a fixed spatio-temporal receptive field or simple temporal averaging for sequential processing, recurrent convolutional models are "doubly deep" in that they can be compositional in spatial and temporal "layers". Such models may have advantages when target concepts are complex and/or training data are limited. Learning long-term dependencies is possible when nonlinearities are incorporated into the network state updates. Long-term RNN models are appealing in that they directly can map variable-length inputs (e.g., video frames) to variable length outputs (e.g., natural language text) and can model complex temporal dynamics; yet they can be optimized with backpropagation. Our recurrent long-term models are directly connected to modern visual convnet models and can be jointly trained to simultaneously learn temporal dynamics and convolutional perceptual representations. Our results show such models have distinct advantages over state-of-the-art models for recognition or generation which are separately defined and/or optimized.

1. Introduction

Recognition and description of images and videos is a fundamental challenge of computer vision. Dramatic progress has been achieved by supervised convolutional models on image recognition tasks, and a number of extensions to process video have been recently proposed. Ideally, a video model should allow processing of variable length input sequences, and also provide for variable length outputs, including generation of full-length sentence descriptions that go beyond conventional one-versus-all prediction tasks. In this paper we propose *long-term recurrent convolutional networks* (LRCNs), a novel architecture for visual recognition and description which combines convolutional layers and long-range temporal recursion and is end-to-end trainable (see Figure 1). We instantiate our architecture for specific video activity recognition, image caption genera-

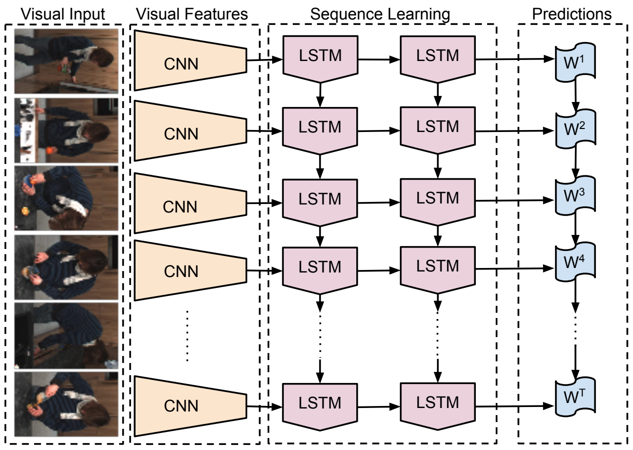


Figure 1: We propose *Long-term Recurrent Convolutional Networks* (LRCNs), a class of architectures leveraging the strengths of rapid progress in CNNs for visual recognition problem, and the growing desire to apply such models to time-varying inputs and outputs. LRCN processes the (possibly) variable-length visual input (left) with a CNN (middle-left), whose outputs are fed into a stack of recurrent sequence models (*LSTMs*, middle-right), which finally produce a variable-length prediction (right).

tion, and video description tasks as described below.

To date, CNN models for video processing have successfully considered learning of 3-D spatio-temporal filters over raw sequence data [13, 2], and learning of frame-to-frame representations which incorporate instantaneous optic flow or trajectory-based models aggregated over fixed windows or video shot segments [16, 33]. Such models explore two extrema of perceptual time-series representation learning: either learn a fully general time-varying weighting, or apply simple temporal pooling. Following the same inspiration that motivates current deep convolutional models, we advocate for video recognition and description models which are also deep over temporal dimensions; i.e., have temporal recurrence of latent variables. RNN models are well known to be "deep in time"; e.g., explicitly so when unrolled, and form implicit compositional representations in the time domain. Such "deep" models predated deep spatial convolution models in the literature [31, 44].

Recurrent Neural Networks have long been explored in perceptual applications for many decades, with varying results. A significant limitation of simple RNN models which strictly integrate state information over time is known as the "vanishing gradient" effect: the ability to backpropogate an error signal through a long-range temporal interval becomes increasingly impossible in practice. A class of models which enable long-range learning was first proposed in [12], and augments hidden state with nonlinear mechanisms to cause state to propagate without modification, be updated, or be reset, using simple memory-cell like neural gates. While this model proved useful for several tasks, its utility became apparent in recent results reporting large-scale learning of speech recognition [10] and language translation models [38, 5].

We show here that long-term recurrent convolutional models are generally applicable to visual time-series modeling; we argue that in visual tasks where static or flat temporal models have previously been employed, long-term RNNs can provide significant improvement when ample training data are available to learn or refine the representation. Specifically, we show LSTM-type models provide for improved recognition on conventional video activity challenges and enable a novel end-to-end optimizable mapping from image pixels to sentence-level natural language descriptions. We also show that these models improve generation of descriptions from intermediate visual representations derived from conventional visual models.

We instantiate our proposed architecture in three experimental settings (see Figure 3). First, we show that directly connecting a visual convolutional model to deep LSTM networks, we are able to train video recognition models that capture complex temporal state dependencies (Figure 3 left; Section 4). While existing labeled video activity datasets may not have actions or activities with extremely complex time dynamics, we nonetheless see improvements on the order of 4% on conventional benchmarks.

Second, we explore direct end-to-end trainable image to sentence mappings. Strong results for machine translation tasks have recently been reported [38, 5]; such models are encoder/decoder pairs based on LSTM networks. We propose a multimodal analog of this model, and describe an architecture which uses a visual convnet to encode a deep state vector, and an LSTM to decode the vector into an natural language string (Figure 3 middle; Section 5). The resulting model can be trained end-to-end on large-scale image and text datasets, and even with modest training provides competitive generation results compared to existing methods.

Finally, we show that LSTM decoders can be driven directly from conventional computer vision methods which predict higher-level discriminative labels, such as the semantic video role tuple predictors in [30] (Figure 3 right; Section 6). While not end-to-end trainable, such models offer architectural and performance advantages over previous statistical machine translation-based approaches, as reported below.

We have realized a generalized "LSTM"-style RNN model in the widely-adopted open source deep learning framework *Caffe* [14], incorporating the specific LSTM units of [46, 38, 5].

2. Background: Recurrent Neural Networks (RNNs)

Traditional RNNs (Figure 2, left) can learn complex temporal dynamics by mapping input sequences to a sequence of hidden states, and hidden states to outputs via the following recurrence equations (Figure 2, left):

$$\begin{aligned} h_t &= g(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \\ z_t &= g(W_{hz}h_t + b_z) \end{aligned}$$

where g is an element-wise non-linearity, such as a sigmoid or hyperbolic tangent, x_t is the input, $h_t \in \mathbb{R}^N$ is the hidden state with N hidden units, and y_t is the output at time t. For a length T input sequence $\langle x_1, x_2, ..., x_T \rangle$, the updates above are computed sequentially as h_1 (letting $h_0 = 0$), y_1, h_2, y_2, ..., h_T, y_T.

Though RNNs have proven successful on tasks such as speech recognition [42] and text generation [37], it can be difficult to train them to learn long-term dynamics, likely due in part to the vanishing and exploding gradients problem [12] that can result from propagating the gradients down through the many layers of the recurrent network, each corresponding to a particular timestep. LSTMs provide a solution by incorporating memory units that allow the network to learn when to forget previous hidden states and when to update hidden states given new information.

As research on LSTMs has progressed, hidden units with varying connections within the memory unit have been proposed. We use the LSTM unit as described in [45] (Figure 2, right), which is a slight simplification of the one described in [10]. Letting $\sigma(x) = (1 + e^{-x})^{-1}$ be the *sigmoid* nonlinearity which squashes real-valued inputs to a $[0, 1]$ range, and letting $\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 2\sigma(2x) - 1$ be the *hyperbolic tangent* nonlinearity, similarly squashing its inputs to a $[-1, 1]$ range, the LSTM updates for timestep t given inputs x_t, h_{t-1}, and c_{t-1} are:

$$\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
g_t &= \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \phi(c_t)
\end{aligned}$$

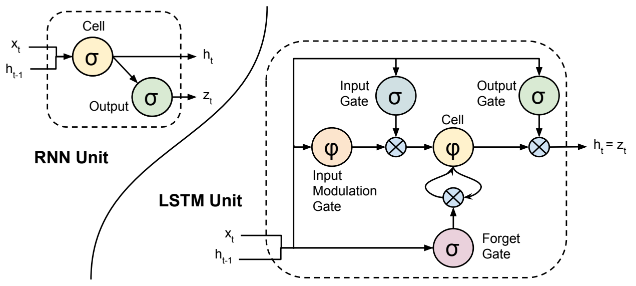


Figure 2: A diagram of a basic RNN cell (left) and an LSTM memory cell (right) used in this paper (from [45], a slight simplification of the architecture described in [9], which was derived from the LSTM initially proposed in [12]).

In addition to a hidden unit $h_t \in \mathbb{R}^N$, the LSTM includes an input gate $i_t \in \mathbb{R}^N$, forget gate $f_t \in \mathbb{R}^N$, output gate $o_t \in \mathbb{R}^N$, input modulation gate $g_t \in \mathbb{R}^N$, and memory cell $c_t \in \mathbb{R}^N$. The memory cell unit c_t is a summation of two things: the previous memory cell unit c_{t-1} which is modulated by f_t, and g_t, a function of the current input and previous hidden state, modulated by the input gate i_t. Because i_t and f_t are sigmoidal, their values lie within the range $[0, 1]$, and i_t and f_t can be thought of as knobs that the LSTM learns to selectively forget its previous memory or consider its current input. Likewise, the output gate o_t learns how much of the memory cell to transfer to the hidden state. These additional cells enable the LSTM to learn extremely complex and long-term temporal dynamics the RNN is not capable of learning. Additional depth can be added to LSTMs by stacking them on top of each other, using the hidden state of the LSTM in layer $l - 1$ as the input to the LSTM in layer l.

Recently, LSTMs have achieved impressive results on language tasks such as speech recognition [10] and machine translation [38, 5]. Analogous to CNNs, LSTMs are attractive because they allow end-to-end fine-tuning. For example, [10] eliminates the need for complex multi-step pipelines in speech recognition by training a deep bidirectional LSTM which maps spectrogram inputs to text. Even with no language model or pronunciation dictionary, the model produces convincing text translations. [38] and [5] translate sentences from English to French with a multi-layer LSTM encoder and decoder. Sentences in the source language are mapped to a hidden state using an encoding LSTM, and then a decoding LSTM maps the hidden state to a sequence in the target language. Such an encoder decoder scheme allows sequences of different lengths to be mapped to each other. Like [10] the sequence-to-sequence architecture for machine translation circumvents the need for language models.

The advantages of LSTMs for modeling sequential data in vision problems are twofold. First, when integrated with current vision systems, LSTM models are straightforward to fine-tune end-to-end. Second, LSTMs are not confined to fixed length inputs or outputs allowing simple modeling for sequential data of varying lengths, such as text or video. We next describe a unified framework to combine LSTMs with deep convolutional networks to create a model which is both spatially and temporally deep.

3. Long-term Recurrent Convolutional Network (LRCN) model

This work proposes a Long-term Recurrent Convolutional Network (LRCN) model combinining a deep hierarchical visual feature extractor (such as a CNN) with a model that can learn to recognize and synthesize temporal dynamics for tasks involving sequential data (inputs or outputs), visual, linsguistical or otherwise. Figure 1 depicts the core of our approach. Our LRCN model works by passing each visual input v_t (an image in isolation, or a frame from a video) through a feature transformation $\phi_V(v_t)$ parametrized by V to produce a fixed-length vector representation $\phi_t \in \mathbb{R}^d$. Having computed the feature-space representation of the visual input sequence $\langle \phi_1, \phi_2, ..., \phi_T \rangle$, the sequence model then takes over.

In its most general form, a sequence model parametrized by W maps an input x_t and a previous timestep hidden state h_{t-1} to an output z_t and updated hidden state h_t. Therefore, inference must be run sequentially (i.e., from top to bottom, in the *Sequence Learning* box of Figure 1), by computing in order: $h_1 = f_W(x_1, h_0) = f_W(x_1, 0)$, then $h_2 = f_W(x_2, h_1)$, etc., up to h_T. Some of our models stack multiple LSTMs atop one another as described in Section 2.

The final step in predicting a distribution $P(y_t)$ at timestep t is to take a softmax over the outputs z_t of the sequential model, producing a distribution over the (in our case, finite and discrete) space C of possible per-timestep outputs:

$$P(y_t = c) = \frac{\exp(W_{zc} z_{t,c} + b_c)}{\sum\limits_{c' \in C} \exp(W_{zc} z_{t,c'} + b_c)}$$

The success of recent very deep models for object recognition [22, 34, 39] suggests that strategically composing many "layers" of non-linear functions can result in very powerful models for perceptual problems. For large T, the above recurrence indicates that the last few predictions from a recurrent network with T timesteps are computed by a very "deep" (T-layered) non-linear function, suggesting that the resulting recurrent model may have similar representational power to a T-layer neural network. Critically, however, the sequential model's weights W are reused at every timestep, forcing the model to learn generic timestep-to-timestep dynamics (as opposed to dynamics directly conditioned on t, the sequence index) and preventing the parameter size from growing in proportion to the maximum number of timesteps.

In most of our experiments, the visual feature transformation ϕ corresponds to the activations in some layer of a large CNN. Using a visual transformation $\phi_V(.)$ which is time-invariant and independent at each timestep has the important advantage of making the expensive convolutional inference and training parallelizable over all timesteps of the input, facilitating the use of fast contemporary CNN implementations whose efficiency relies on independent batch processing, and end-to-end optimization of the visual and sequential model parameters V and W.

We consider three vision problems (activity recognition, image description and video description) which instantiate one of the following broad classes of sequential learning tasks:

1. Sequential inputs, fixed outputs (Figure 3, left): $\langle x_1, x_2, ..., x_T \rangle \mapsto y$. The visual activity recognition problem can fall under this umbrella, with videos of arbitrary length T as input, but with the goal of predicting a single label like *running* or *jumping* drawn from a fixed vocabulary.

2. Fixed inputs, sequential outputs (Figure 3, middle): $x \mapsto \langle y_1, y_2, ..., y_T \rangle$. The image description problem fits in this category, with a non-time-varying image as input, but a much larger and richer label space consisting of *sentences* of any length.

3. Sequential inputs and outputs (Figure 3, right): $\langle x_1, x_2, ..., x_T \rangle \mapsto \langle y_1, y_2, ..., y_{T'} \rangle$. Finally, it's easy to imagine tasks for which both the visual input and output are time-varying, and in general the number of input and output timesteps may differ (i.e., we may have $T \neq T'$). In the video description task, for example, the input and output are both sequential, and the number of frames in the video should not constrain the length of (number of words in) the natural-language description.

In the previously described formulation, each instance has T inputs $\langle x_1, x_2, ..., x_T \rangle$ and T outputs $\langle y_1, y_2, ..., y_T \rangle$. We describe how we adapt this formulation in our hybrid model to tackle each of the above three problem settings. With sequential inputs and scalar outputs, we take a late fusion approach to merging the per-timestep predictions $\langle y_1, y_2, ..., y_T \rangle$ into a single prediction y for the full sequence. With fixed-size inputs and sequential outputs, we simply duplicate the input x at all T timesteps $x_t := x$ (noting this can be done cheaply due to the time-invariant visual feature extractor). Finally, for a sequence-to-sequence problem with (in general) different input and output lengths, we take an "encoder-decoder" approach inspired by [46]. In this approach, one sequence model, the *encoder*, is used to map the input sequence to a fixed-length vector, then another sequence model, the *decoder*, is used to unroll this vector to sequential outputs of arbitrary length. Under this model, the system as a whole may be thought of as having $T + T'$ timesteps of input and output, wherein the input is processed and the decoder outputs are ignored for the first T timesteps, and the predictions are made and "dummy" inputs are ignored for the latter T' timesteps.

Under the proposed system, the weights (V, W) of the model's visual and sequential components can be learned jointly by maximizing the likelihood of the ground truth outputs y_t conditioned on the input data and labels up to that point $(x_{1:t}, y_{1:t-1})$ In particular, we minimize the negative log likelihood $\mathcal{L}(V, W) = -\log P_{V,W}(y_t | x_{1:t}, y_{1:t-1})$ of the training data (x, y).

One of the most appealing aspects of the described system is the ability to learn the parameters "end-to-end," such that the parameters V of the visual feature extractor learn to pick out the aspects of the visual input that are relevant to the sequential classification problem. We train our LRCN models using stochastic gradient descent with momentum, with backpropagation used to compute the gradient $\nabla \mathcal{L}(V, W)$ of the objective $\mathcal{L}$ with respect to all parameters (V, W).

We next demonstrate the power of models which are both deep in space and deep in time by exploring three applications: activity recognition, image description, and video description.

4. Activity recognition

Activity recognition is an example of the first sequential learning task described above; T individual frames are inputs into T convolutional networks which are then connected to a single-layer LSTM with 256 hidden units. A large body of recent work has proposed deep architectures

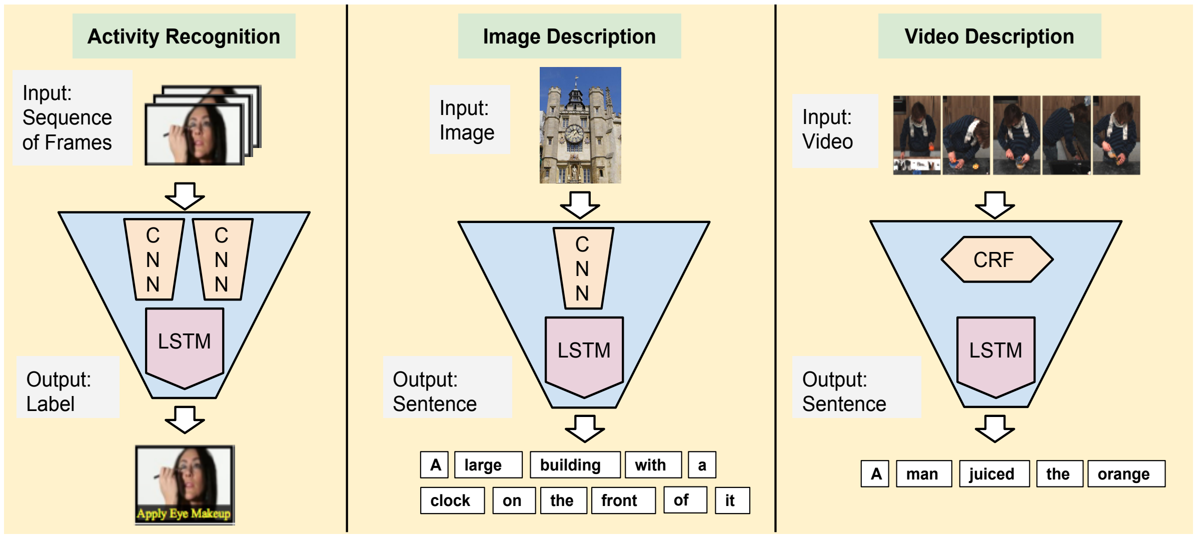


Figure 3: Task-specific instantiations of our LRCN model for activity recognition, image description, and video description.

for activity recognition ([16, 33, 13, 2, 1]). [33, 16] both propose convolutional networks which learn filters based on a stack of N input frames. Though we analyze clips of 16 frames in this work, we note that the LRCN system is more flexible than [33, 16] since it is not constrained to analyzing fixed length inputs and could potentially learn to recognize complex video sequences (*e.g.*, cooking sequences as presented in 6). [1, 2] use recurrent neural networks to learn temporal dynamics of either traditional vision features ([1]) or deep features ([2]), but do not train their models end-to-end and do not pre-train on larger object recognition databases for important performance gains.

We explore two variants of the LRCN architecture: one in which the LSTM is placed after the first fully connected layer of the CNN (LRCN-fc$_6$) and another in which the LSTM is placed after the second fully connected layer of the CNN (LRCN-fc$_7$). We train the LRCN networks with video clips of 16 frames. The LRCN predicts the video class at each time step and we average these predictions for final classification. At test time, we extract 16 frame clips with a stride of 8 frames from each video and average across clips.

We also consider both RGB and flow inputs. Flow is computed with [4] and transformed into a "flow image" by centering x and y flow values around 128 and multiplying by a scalar such that flow values fall between 0 and 255. A third channel for the flow image is created by calculating the flow magnitude. The CNN base of the LRCN is a hybrid of the *Caffe* [14] reference model, a minor variant of *AlexNet* [22], and the network used by Zeiler & Fergus [47]. The net is pre-trained on the 1.2M image ILSVRC-2012 [32] classification training subset of the ImageNet [7] dataset, giving the network a strong initialization to facilitate faster training and prevent over-fitting to the relatively small video datasets. When classifying center crops, the top-1 classification accuracy is 60.2% and 57.4% for the hybrid and *Caffe* reference models, respectively. In our baseline model, T video frames are individually classified by a CNN. As in the LSTM model, whole video classification is done by averaging scores across all video frames.

4.1. Evaluation

We evaluate our architecture on the UCF-101 dataset [36] which consists of over 12,000 videos categorized into 101 human action classes. The dataset is split into three splits, with a little under 8,000 videos in the training set for each split. We report accuracy for split-1.

Figure 1, columns 2-3, compare video classification of our proposed models (LRCN-fc$_6$, LRCN-fc$_7$) against the baseline architecture for both RGB and flow inputs. Each LRCN network is trained end-to-end. To determine if end-to-end training is necessary, we also train a LRCN-fc$_6$ network in which only the LSTM parameters are learned. The fully fine-tuned network increases performance from 70.47% to 71.12%, demonstrating that end-to-end fine-tuning is indeed beneficial. The LRCN-fc$_6$ network yields the best results for both RGB and flow and improves upon the baseline network by 2.12 % and 4.75% respectively.

RGB and flow networks can be combined by computing a weighted average of network scores as proposed in [33]. Like [33], we report two weighted averages of the predictions from the RGB and flow networks in Table 1 (right). Since the flow network outperforms the RGB network, weighting the flow network higher unsurprisingly leads to better accuracy. In this case, LRCN outperforms the baseline single-frame model by 3.88%.

The LRCN shows clear improvement over the baseline single-frame system and approaches the accuracy achieved by other deep models. [33] report the results on UCF-101

Model	Input Type		Weighted Average	
	RGB	Flow	$1/2, 1/2$	$1/3, 2/3$
Single frame	65.40	53.20	–	–
Single frame (ave.)	69.00	72.20	75.71	79.04
LRCN-fc$_6$	**71.12**	**76.95**	81.97	**82.92**
LRCN-fc$_7$	70.68	69.36	–	–

Table 1: Activity recognition: Comparing single frame models to LRCN networks for activity recognition in the UCF-101 [36] dataset, with both RGB and flow inputs. Our LRCN model consistently and strongly outperforms a model based on predictions from the underlying convolutional network architecture alone.

by computing a weighted average between flow and RGB networks (86.4% for split 1 and 87.6% averaging over all splits). Though [16] does not report numbers on the separate splits of UCF-101, the average split accuracy is 65.4% which is substantially lower than our LRCN model.

5. Image description

In contrast to activity recognition, the static image description task only requires a single convolutional network since the input consists of a single image. A variety of deep and multi-modal models [8, 35, 19, 20, 15, 25, 20, 18] have been proposed for image description; in particular, [20, 18] combine deep temporal models with convolutional representations. [20], utilizes a "vanilla" RNN as described in Section 2, potentially making learning long-term temporal dependencies difficult. Contemporaneous with and most similar to our work is [18], which proposes a different architecture that uses the hidden state of an LSTM encoder at time T as the encoded representation of the length T input sequence. It then maps this sequence representation, combined with the visual representation from a convnet, into a joint space from which a separate decoder predicts words. This is distinct from our arguably simpler architecture, which takes as per-timestep input a copy of the static input image, along with the previous word. We present empirical results showing that our integrated LRCN architecture outperforms these prior approaches, none of which comprise an end-to-end optimizable system over a hierarchy of visual and temporal parameters.

We now describe our instantiation of the LRCN architecture for the image description task. At each timestep, both the image features and the previous word are provided as inputs to the sequential model, in this case a stack of LSTMs (each with 1000 hidden units), which is used to learn the dynamics of the time-varying output sequence, natural language. At timestep t, the input to the bottom-most LSTM is the embedded ground truth word from the previous timestep w_{t-1}. For sentence generation, the input becomes a sample $\hat{w}_{t-1}$ from the model's predicted distribution at the previous timestep. The second LSTM in the stack fuses the outputs of the bottom-most LSTM with the image representation $\phi_V(x)$ to produce a joint representation of the visual and language inputs up to time t. (The visual model $\phi_V(x)$ used in this experiment is the base *Caffe* [14] reference model, very similar to the well-known *AlexNet* [22], pre-trained on ILSVRC-2012 [32] as in Section 4.) Any further LSTMs in the stack transform the outputs of the LSTM below, and the fourth LSTM's outputs are inputs to the softmax which produces a distribution over words $p(w_t|w_{1:t-1})$.

Following [19], we refer to the use of the bottom-most LSTM to exclusively process the language input (with no visual input) as the *factored* version of the model, and study the importance of this by comparing it to an *unfactored* variant. See Figure 4 for details on the variants we study.

Without any explicit language modeling or defined syntax structure, the described LRCN system learns mappings from pixel intensity values to natural language descriptions that are often semantically descriptive and grammatically correct.

5.1. Evaluation

We evaluate our image description model on both image retrieval and image annotation generation. We first show the effectiveness of our model by quantitatively evaluating it on the image retrieval task proposed by [26] and seen in [25, 15, 35, 8, 18]. Our model is trained on the combined training sets of the Flickr30k [28] (28,000 training images) and COCO2014 [24] dataset (80,000 training images). We report results on Flickr30k [28], with 30,000 images and five sentence annotations per image. We use 1000 images each for test and validation and the remaining 28,000 for training.

Image retrieval results are recorded in Table 2 and report median rank, **Med***r*, of the first retrieved ground truth image and Recall@K, the number of sentences for which the correct image is retrieved in the top-K. Our model consistently outperforms the strong baselines from recent work [18, 25, 15, 35, 8] as can be seen in Table 2. Here, we make a note that the new *OxfordNet* model in [18] outperforms our model on the retrieval task. However, *OxfordNet* [18] utilizes a better-performing convolutional network to get the additional edge over the base *ConvNet* [18]. The strength of our temporal model (and integration of the temporal and visual models) can be more directly measured against the *ConvNet* [18] result, which uses the same base CNN architecture [22] pretrained on the same data.

In Table 3, we report image-to-caption retrieval results for each of the architectural variants in Figure 4, as well as a four-layer version (LRCN$_{4f}$) of the factored model. Based on the facts that LRCN$_{2f}$ outperforms the LRCN$_{4f}$ model, and LRCN$_{1u}$ outperforms LRCN$_{2u}$, there seems to be little to be gained from naively stacking additional LSTM layers atop an existing network. On the other hand, a compari-

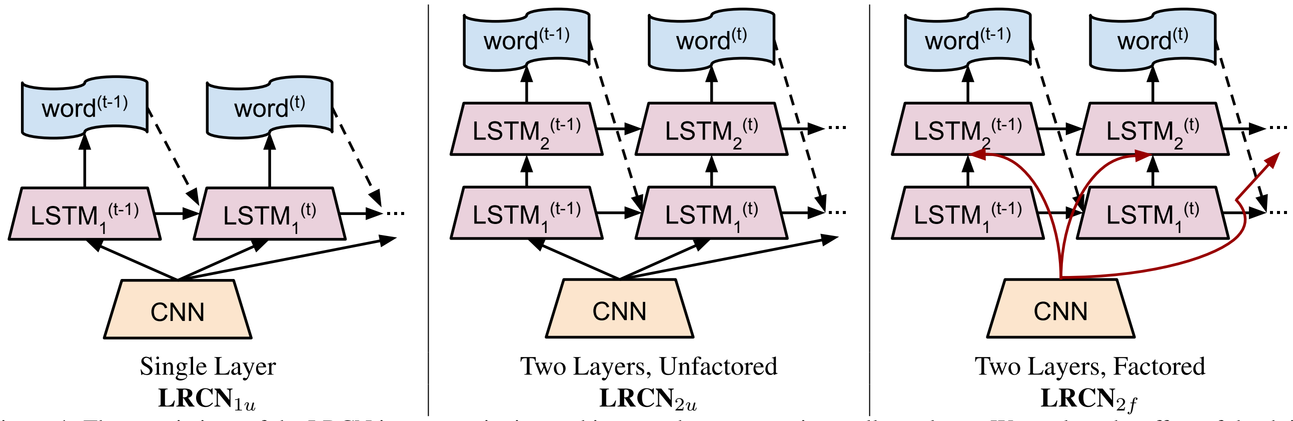


Figure 4: Three variations of the LRCN image captioning architecture that we experimentally evaluate. We explore the effect of depth in the LSTM stack, and the effect of the "factorization" of the modalities (also explored in [19]).

	R@1	**R@5**	**R@10**	**Med*r***
DeViSE [8]	6.7	21.9	32.7	25
SDT-RNN [35]	8.9	29.8	41.1	16
DeFrag [15]	10.3	31.4	44.5	13
m-RNN [25]	12.6	31.2	41.5	16
ConvNet [18]	10.4	31.0	43.7	14
LRCN$_{2f}$ (ours)	**17.5**	**40.3**	**50.8**	**9**

Table 2: Image description: Caption-to-image retrieval results for the Flickr30k [28] dataset. **R@K** is the average recall at rank K (high is good). **Med*r*** is the median rank (low is good). Note that [18] achieves better retrieval performance using a stronger CNN architecture see text.

	R@1	**R@5**	**R@10**	**Med*r***
LRCN$_{1u}$	14.1	31.3	39.7	24
LRCN$_{2u}$	3.8	12.0	17.9	80
LRCN$_{2f}$	**17.5**	**40.3**	**50.8**	**9**
LRCN$_{4f}$	15.8	37.1	49.5	10

Table 3: Flickr30k caption-to-image retrieval results for variants of the LRCN architectures. See Figure 4 for diagrams of these architectures. The results indicate that the "factorization" is important to the LRCN's retrieval performance, while simply stacking additional LSTM layers does not seem to improve performance.

son of the LRCN$_{2f}$ and LRCN$_{2u}$ results indicatees that the "factorization" in the architecture is quite important to the model's retrieval performance.

To evaluate sentence generation, we use the BLEU [27] metric which was designed for automated evaluation of statistical machine translation. BLEU is a modified form of precision that compares N-gram fragments of the hypothesis translation with multiple reference translations. We use BLEU as a measure of similarity of the descriptions. The unigram scores (B-1) account for the adequacy of (or the information retained) by the translation, while longer N-gram scores (B-2, B-3) account for the fluency. We compare our results with [25] (on Flickr30k), and two strong

	Flickr30k [28]			
	B-1	B-2	B-3	B-4
m-RNN [25]	54.79	23.92	19.52	-
1NN fc$_8$ base (ours)	37.34	18.66	9.39	4.88
1NN fc$_7$ base (ours)	38.81	20.16	10.37	5.54
LRCN (ours)	**58.72**	**39.06**	**25.12**	**16.46**
	COCO 2014 [24]			
	B-1	B-2	B-3	B-4
1NN fc$_8$ base (ours)	46.04	26.20	14.95	8.70
1NN fc$_7$ base (ours)	47.47	27.55	15.96	9.36
LRCN (ours)	**62.79**	**44.19**	**30.41**	**21.00**

Table 4: Image description: Sentence generation results (BLEU scores (%) – *ours* are adjusted with the brevity penalty) for the Flickr30k [28] and COCO 2014 [24] test sets.

nearest neighbor baselines computed using AlexNet fc$_7$ and fc$_8$ layer outputs. We used 1-nearest neighbor to retrieve the most similar image in the training database and average the BLEU score over the captions. The results on Flickr30k are reported in Table 4. Additionally, we report results on the new COCO2014 [24] dataset which has 80,000 training images, and 40,000 validation images. Similar to Flickr30k, each image is annotated with 5 or more image annotations. We isolate 5,000 images from the validation set for testing purposes and the results are reported in Table 4.

Based on the B-1 scores in Table 4, generation using LRCN performs comparably with m-RNN [25] in terms of the information conveyed in the description. Furthermore, LRCN significantly outperforms the baselines and the m-RNN with regard to the fluency (B-2, B-3) of the generation, indicating the LRCN retains more of the bigrams and trigrams from the human-annotated descriptions.

In addition to standard quantitative evaluations, we also employ Amazon Mechnical Turkers (AMT) to evaluate the generated sentences. Given an image and a set of descriptions from different models, we ask Turkers to rank the sentences based on correctness, grammar and relevance.

	Correctness	Grammar	Relevance
TreeTalk [23]	4.08	4.35	3.98
OxfordNet [18]	3.71	3.46	3.70
NN [18]	**3.44**	3.20	**3.49**
LRCN fc_8 (ours)	3.74	3.19	3.72
LRCN ft (ours)	**3.47**	**3.01**	**3.50**
Captions	2.55	3.72	2.59

Table 5: Image description: Human evaluator rankings from 1-6 (low is good) averaged for each method and criterion. We evaluated on 785 Flickr images selected by the authors of [18] for the purposes of comparison against this similar contemporary approach.

We compared sentences from our model to the ones made publicly available by [18]. As seen in Table 5, our fine-tuned (ft) LRCN model performs on par with the Nearest Neighbour (NN) on correctness and relevance, and better on grammar. We show example sentence generations in Figure 6.

6. Video description

In video description we must generate a variable length stream of words, similar to Section 5. [11, 30, 17, 3, 6, 17, 40, 41] propose methods for generating sentence descriptions for video, but to our knowledge we present the first application of deep models to the vision description task.

The LSTM framework allows us to model the video as a variable length input stream as discussed in Section 3. However, due to limitations of available video description datasets we take a different path. We rely on more "traditional" activity and video recognition processing for the input and use LSTMs for generating a sentence.

We first distinguish the following architectures for video description (see Figure 5). For each architecture, we assume we have predictions of objects, subjects, and verbs present in the video from a CRF based on the full video input. In this way, we observe the video as whole at each time step, not incrementally frame by frame.

(a) LSTM encoder & decoder with CRF max. (Figure 5(a)) The first architecture is motivated by the video description approach presented in [30]. They first recognize a semantic representation of the video using the maximum a posterior estimate (MAP) of a CRF taking in video features as unaries. This representation, *e.g.* ⟨person,cut,cutting board⟩, is then concatenated to a input sentence (*person cut cutting board*) which is translated to a natural sentence (*a person cuts on the board*) using phrase-based statistical machine translation (SMT) [21]. We replace the SMT with an LSTM, which has shown state-of-the-art performance for machine translation between languages [38, 5]. The architecture (shown in Figure 5(a)) has an encoder LSTM (orange) which encodes the one-hot vector (binary index vector in a vocabulary) of the input sentence as done in [38]. This allows for variable-length inputs. (Note that the input sentence might have a different number of words than elements of the semantic representation.) At the end of the encoder stage, the final hidden unit must remember all necessary information before being input into the decoder stage (pink) in which the hidden representation is decoded into a sentence, one word at each time step. We use the same two-layer LSTM for encoding and decoding.

Architecture	Input	BLEU
SMT [30]	CRF max	24.9
SMT [29]	CRF prob	26.9
(a) LSTM Encoder-Decoder (ours)	CRF max	25.3
(b) LSTM Decoder (ours)	CRF max	27.4
(c) LSTM Decoder (ours)	CRF prob	**28.8**

Table 6: Video description: Results on detailed description of TACoS multilevel[29], in %, see Section 6 for details.

(b) LSTM decoder with CRF max. (Figure 5(b)) In this variant we exploit that the semantic representation can be encoded as a single fixed length vector. We provide the entire visual input representation at each time step to the LSTM, analogous to how an entire image is provided as an input to the LSTM in image description.

(c) LSTM decoder with CRF prob. (Figure 5(c)) A benefit of using LSTMs for machine translation compared to phrase-based SMT [21] is that it can naturally incorporate probability vectors during training *and* test time which allows the LSTM to learn uncertainties in visual generation rather than relying on MAP estimates. The architecture is the the same as in (b), but we replace max predictions with probability distributions.

6.1. Evaluation

We evaluate our approach on the TACoS multilevel [29] dataset, which has 44,762 video/sentence pairs (about 40,000 for training/validation). We compare to [30] who use max prediction as well as a variant presented in [29] which takes CRF probabilities at test time and uses a word lattice to find an optimal sentence prediction. Since we use the max prediction as well as the probability scores provided by [29], we have an identical visual representation. [29] uses dense trajectories [43] and SIFT features as well as temporal context reasoning modeled in a CRF.

Table 6 shows the BLEU-4 score. The results show that (1) the LSTM outperforms an SMT-based approach to video description; (2) the simpler decoder architecture (b) and (c) achieve better performance than (a), likely because the input does not need to be memorized; and (3) our approach achieves 28.8%, clearly outperforming the best reported number of 26.9% on TACoS multilevel by [29].

More broadly, these results show that our architecture

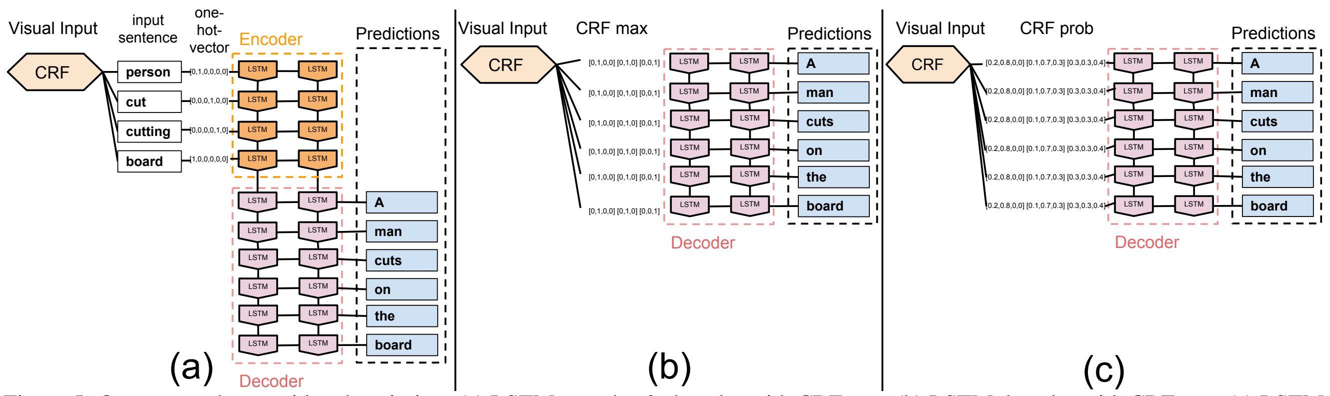


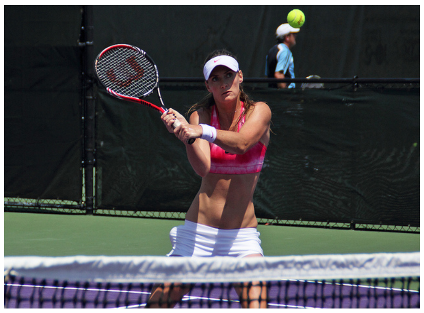
Figure 5: Our approaches to video description. (a) LSTM encoder & decoder with CRF max (b) LSTM decoder with CRF max (c) LSTM decoder with CRF probabilities. (For larger figure zoom or see supplemental).

is not restricted to deep neural networks inputs but can be cleanly integrated with other fixed or variable length inputs from other vision systems.

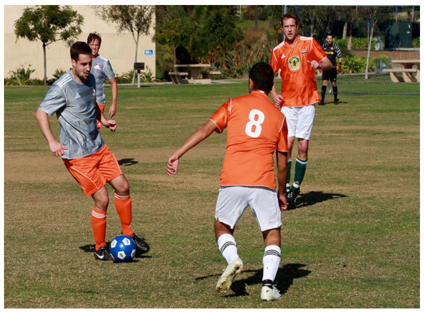
7. Conclusions

We've presented LRCN, a class of models that is both spatially and temporally deep, and has the flexibility to be applied to a variety of vision tasks involving sequential inputs and outputs. Our results consistently demonstrate that by learning sequential dynamics with a deep sequence model, we can improve on previous methods which learn a deep hierarchy of parameters only in the visual domain, and on methods which take a fixed visual representation of the input and only learn the dynamics of the output sequence.

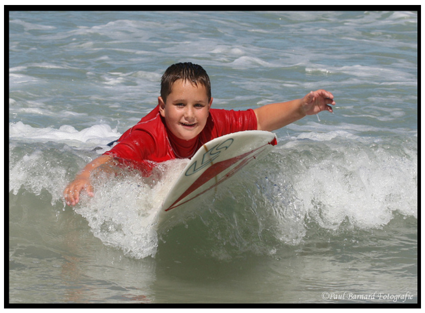
As the field of computer vision matures beyond tasks with static input and predictions, we envision that "doubly deep" sequence modeling tools like LRCN will soon become central pieces of most vision systems, as convolutional architectures recently have. The ease with which these tools can be incorporated into existing visual recognition pipelines makes them a natural choice for perceptual problems with time-varying visual input or sequential outputs, which these methods are able to produce with little input preprocessing and no hand-designed features.



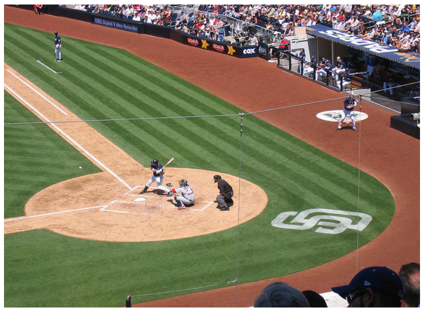
A female tennis player in action on the court.



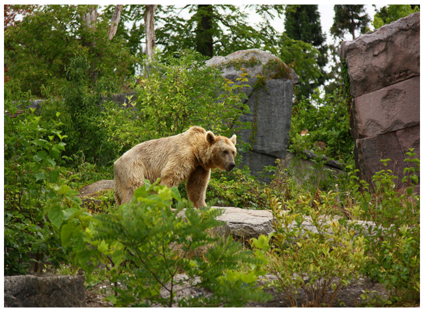
A group of young men playing a game of soccer



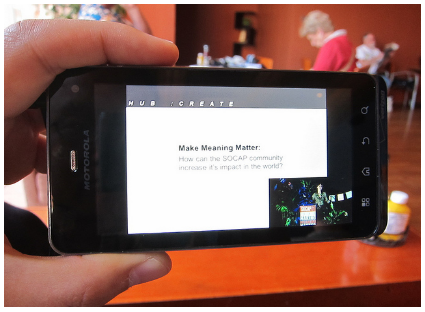
A man riding a wave on top of a surfboard.



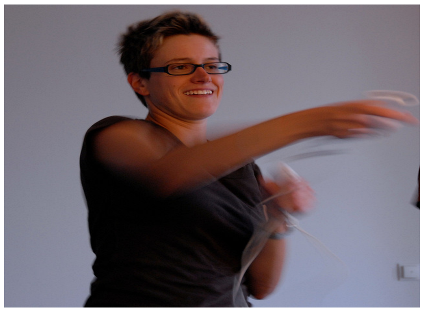
A baseball game in progress with the batter up to plate.



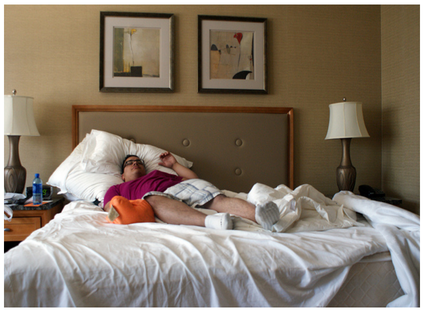
A brown bear standing on top of a lush green field.



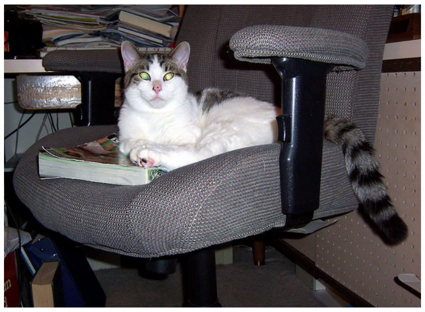
A person holding a cell phone in their hand.



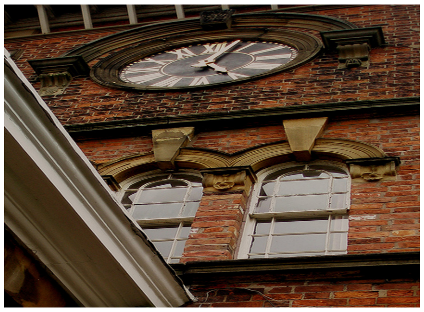
A close up of a person brushing his teeth.



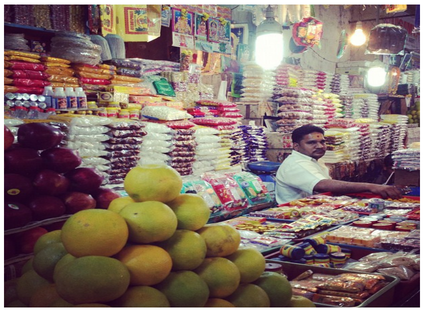
A woman laying on a bed in a bedroom.



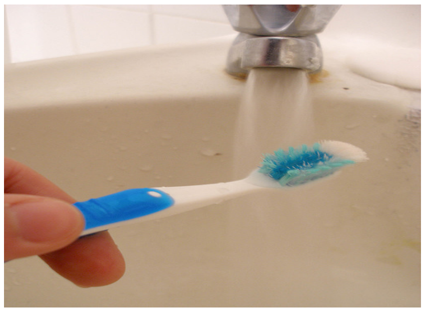
A black and white cat is sitting on a chair.



A large clock mounted to the side of a building.

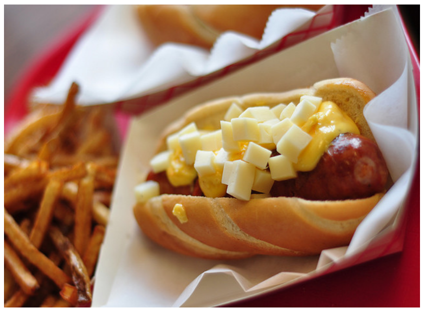


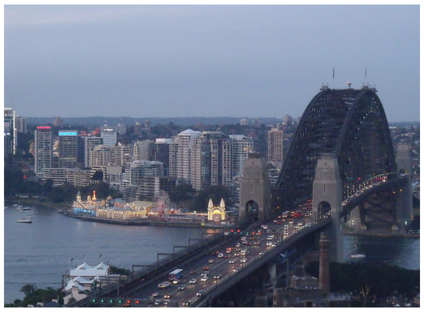
A bunch of fruit that are sitting on a table.

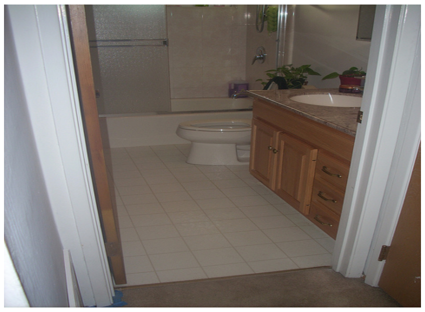


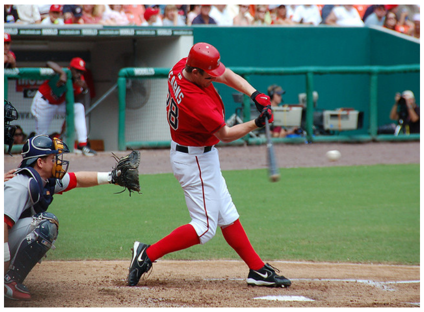
A toothbrush holder sitting on top of a white sink.

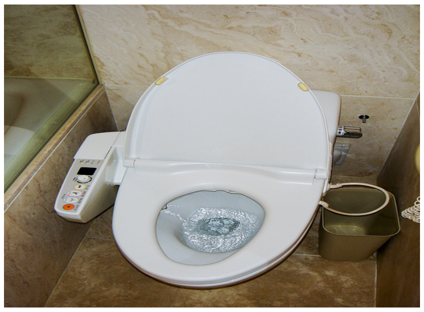
Figure 6: Image description: images with corresponding captions generated by our finetuned LRCN model. These are images 1-12 of our randomly chosen validation set from COCO 2014 [24] (see Figure 7 for images 13-24). We used beam search with a beam size of 5 to generate the sentences, and display the top (highest likelihood) result above.

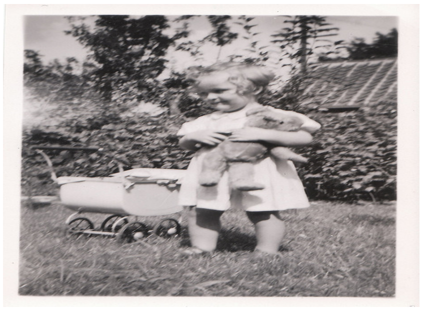

A close up of a hot dog on a bun.

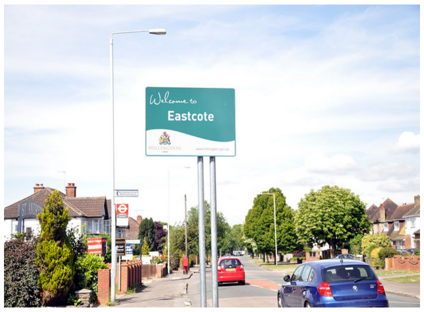

A boat on a river with a bridge in the background.


A bath room with a toilet and a bath tub.

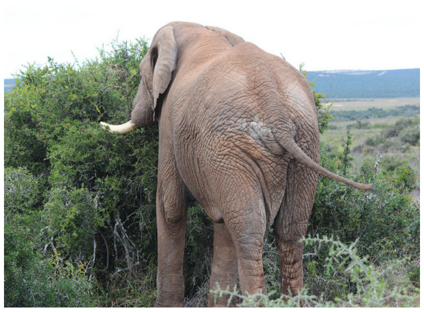

A man that is standing in the dirt with a bat.

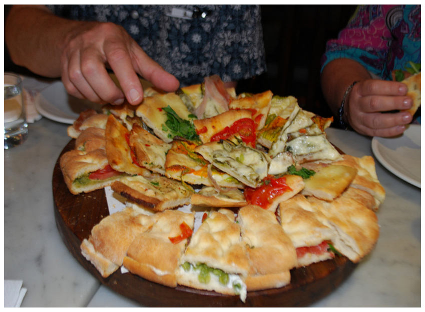

A white toilet sitting in a bathroom next to a trash can.

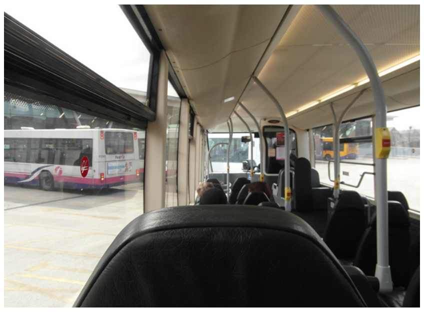

Black and white photograph of a woman sitting on a bench.

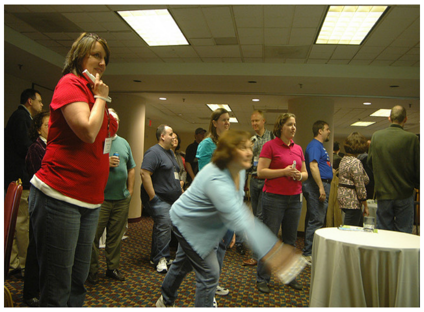


A group of people walking down a street next to a traffic light.


An elephant standing in a grassy area with tree in the background.


A close up of a plate of food with broccoli.


A bus parked on the side of a street next to a building.


A group of people standing around a table.

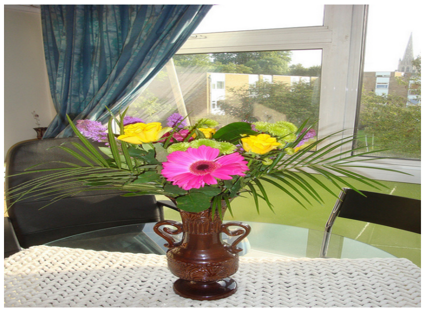

A vase filled with flower sitting on a table.

Figure 7: Image description: images 13-24 (and LRCN-generated captions) from the set described in Figure 6.

Acknowledgements

The authors thank Oriol Vinyals for valuable advice and helpful discussion throughout this work. This work was supported in part by DARPA's MSEE and SMISC programs, NSF awards IIS-1427425, and IIS-1212798, IIS-1116411, Toyota, and the Berkeley Vision and Learning Center. Marcus Rohrbach was supported by a fellowship within the FITweltweit-Program of the German Academic Exchange Service (DAAD).

References

[1] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Action classification in soccer videos with long short-term memory recurrent neural networks. In *ICANN*. 2010. 5

[2] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. In *Human Behavior Understanding*. 2011. 2, 5

[3] A. Barbu, A. Bridge, Z. Burchill, D. Coroian, S. Dickinson, S. Fidler, A. Michaux, S. Mussman, S. Narayanaswamy, D. Salvi, L. Schmidt, J. Shangguan, J. M. Siskind, J. Waggoner, S. Wang, J. Wei, Y. Yin, and Z. Zhang. Video in sentences out. In *UAI*, 2012. 8

[4] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*. 2004. 5

[5] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. 2, 3, 8

[6] P. Das, C. Xu, R. Doell, and J. Corso. Thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *CVPR*, 2013. 8

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[8] A. Frome, G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. DeViSE: A deep visual-semantic embedding model. In *NIPS*, 2013. 6, 7

[9] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013. 3

[10] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *ICML*, 2014. 2, 3

[11] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. YouTube2Text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shoot recognition. In *ICCV*, 2013. 8

[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 1997. 2, 3

[13] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):221–231, 2013. 2, 5

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014. 2, 5, 6

[15] A. Karpathy, A. Joulin, and L. Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. *NIPS*, 2014. 6, 7

[16] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 2, 5, 6

[17] M. U. G. Khan, L. Zhang, and Y. Gotoh. Human focused video description. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011. 8

[18] R. Kiros, R. Salakhuditnov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 6, 7, 8

[19] R. Kiros, R. Salakhutdinov, and R. Zemel. Multimodal neural language models. In *ICML*, 2014. 6, 7

[20] R. Kiros, R. Zemel, and R. Salakhutdinov. Multimodal neural language models. In *Proc. NIPS Deep Learning Workshop*, 2013. 6

[21] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *ACL*, 2007. 8

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 4, 5, 6

[23] P. Kuznetsova, V. Ordonez, T. L. Berg, U. C. Hill, and Y. Choi. Treetalk: Composition and compression of trees for image descriptions. *Transactions of the Association for Computational Linguistics*, 2(10):351–362, 2014. 8

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014. 6, 7, 10

[25] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille. Explain images with multimodal recurrent neural networks. *arXiv preprint arXiv:1410.1090*, 2014. 6, 7

[26] P. Y. Micah Hodosh and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *JAIR*, 47:853–899, 2013. 6

[27] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *ACL*, 2002. 7

[28] M. H. Peter Young, Alice Lai and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2:67–78, 2014. 6, 7

[29] A. Rohrbach, M. Rohrbach, W. Qiu, A. Friedrich, M. Pinkal, and B. Schiele. Coherent multi-sentence video description with variable level of detail. In *GCPR*, 2014. 8

[30] M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. Translating video content to natural language descriptions. In *ICCV*, 2013. 2, 8

[31] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985. 2

[32] 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, 2014. 5, 6

[33] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. *arXiv preprint arXiv:1406.2199*, 2014. 2, 5, 6

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 4

[35] R. Socher, Q. Le, C. Manning, and A. Ng. Grounded compositional semantics for finding and describing images with sentences. In *NIPS Deep Learning Workshop*, 2013. 6, 7

[36] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 5, 6

[37] I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *ICML*, 2011. 2

[38] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014. 2, 3, 8

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 4

[40] C. C. Tan, Y.-G. Jiang, and C.-W. Ngo. Towards textually describing complex video contents with audio-visual concept classifiers. In *MM*, 2011. 8

[41] J. Thomason, S. Venugopalan, S. Guadarrama, K. Saenko, and R. J. Mooney. Integrating language and vision to generate natural language descriptions of videos in the wild. In *COLING*, 2014. 8

[42] O. Vinyals, S. V. Ravuri, and D. Povey. Revisiting recurrent neural networks for robust ASR. In *ICASSP*, 2012. 2

[43] H. Wang, A. Kläser, C. Schmid, and C. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 2013. 8

[44] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1989. 2

[45] W. Zaremba and I. Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014. 3

[46] W. Zaremba, I. Sutskever, and O. Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014. 2, 4

[47] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*. 2014. 5