• Classification tasks
• **Classification tasks**
• **Structured input to structured output tasks**
  o Machine translation or other NLP tasks
  o Image captioning
Language modeling using RNN

• Compute the probability of a sentence $s = (w_1, w_2, \ldots, w_T)$

\[ p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t|w_1, \ldots, w_{t-1}) \]

• RNN
Recap: Forward propagation in RNN

Recurrent connections between hidden units; output every time step

\[ a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)} , \]
\[ h^{t} = \tanh(a^{(t)}) , \]
\[ o^{(t)} = c + Vh^{(t)} , \]
\[ \hat{y}^{(t)} = \text{softmax}(o^{(t)}) \]

Softmax to get normalized probabilities

[Fig 10.3]
Language modeling using RNN

• Compute the probability of a sentence $s = (w_1, w_2, \ldots, w_T)$

$$p(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} p(w_t|w_1, \ldots, w_{t-1})$$

Conditional probability

• RNN

$$p(w_{t+1} = w|w_1, \ldots, w_t) = g_{\theta}^w(h_t, w_t)$$

Probability of the next word being ‘$w$’
Conditional language model

- \( p(w_{t+1} = w | w_1, \ldots, w_t) = g_\theta^w(h_t, w_t) \)
- \( h_t = \varphi_\theta(h_{t-1}, w_t, c) \)
Recap: Encoder-Decoder
Sequence-to-Sequence Architecture

• Map variable-length input sequence to variable-length output sequence
• Machine translation [Cho et al., 2014] [Sutskever et al., 2014]
Encoder-Decoder
Sequence-to-Sequence Architecture

Encoder (reader or input) RNN processes input sequence $x=(x^{(1)}, \ldots, x^{(nx)})$ and emits context $C$
Encoder-Decoder
Sequence-to-Sequence Architecture

Encoder (reader or input) RNN processes input sequence \( x=(x^{(1)}, \ldots, x^{(nx)}) \) and emits context \( C \).

Decoder (writer or output) RNN is conditioned on the context \( C \) to generate output sequence \( y=(y^{(1)}, \ldots, y^{(ny)}) \).
Encoder-Decoder
[Grid]-to-[Sequence] Architecture

Encoder (reader or input) [CNN] processes input image $x$ and emits context $C$

Decoder (writer or output) RNN is conditioned on the context $C$ to generate output sequence $y=(y^{(1)}, \ldots, y^{(ny)})$
Encoder-Decoder

[Grid]-to-[Sequence] Architecture

Encoder (reader or input) [CNN] processes input image $x$ and emits context $C$.

Decoder (writer or output) RNN is conditioned on the context $C$ to generate output sequence $y=(y^{(1)}, \ldots, y^{(ny)})$.

The context model is too simple to guarantee that spatial, temporal, or spatio-temporal structures of input are preserved.
Attention mechanisms allow the system to sequentially focus on different subsets of the input (Cho et al., 2015).
Attention mechanism

• A structured representation of input
• e.g., a set of fixed-size vectors known as “context set”
  \[ C = \{ c_1, c_2, \ldots, c_M \} \]
• Attention model: another neural network to map hidden state to context vector
Attention model

\[ e_i^t = f_{Attn} \left( z_{t-1}, c_i, \{ \alpha_j^{t-1} \}_{j=1}^M \right) \]

- Soft attention: softmax over context vectors in context set
- Hard attention: one best match

MC sampling [Xu et al., 2015]

\[ \alpha_i^t = \frac{\exp(e_i^t)}{\sum_{j=1}^{M} e_j^t} \]

Natural for gradient back-propagation
Attention model

\[ e_i^t = f_{Att}\left(z_{t-1}, c_i, \left\{ \alpha_j^{t-1} \right\}_{j=1}^M \right) \]

Hidden state \( z \)

Attention weight \( \alpha \)

Hidden state \( z \)

Attention weight \( \alpha \)

\[ c^t = \varphi\left(\left\{ c_i \right\}_{i=1}^M, \left\{ \alpha_i^t \right\}_{i=1}^M \right) \]

- Soft attention: softmax over context vectors in context set
- Hard attention: one best match

MC sampling

Natural for gradient back-propagation

\[ \alpha_i^t = \frac{\exp(e_i^t)}{\sum_{j=1}^M e_j^t} \]

e: score of context \( c_i \) at time \( t \)

e.g., weighted sum
Conditional RNN language model

\[ c^t = \varphi \left( \left\{ c_i \right\}_{i=1}^M, \left\{ \alpha_i^t \right\}_{i=1}^M \right) = \sum_{i=1}^M \alpha_i c_i \]

- Computing context vector every time step instead of using a fixed-length context vector
- \( h_t = \varphi_\theta(h_{t-1}, x_t, c_t) \)
Image captioning

• Representation of input image:
  – Activation of the last **fully-connected** hidden layer as context vector in **simple** encoder-decoder model
  – Activation of the last **convolutional** layer to use **attention** mechanism
[Karpathy & Fei-Fei 2015]

Generate dense descriptions of images using multimodal embedding

Figure 1. Motivation/Concept Figure: Our model treats language as a rich label space and generates descriptions of image regions.
Representing images

• Bounding box detection using:
  – R-CNN + pretrain on ImageNet + finetuning on 200 classes of ImageNet Detection Challenge

R-CNN: Regions with CNN features

[1/31/18]

Girshick CVPR’14

4096 activations of fully connected layer right before classification
Representing images

• Top 19 bounding boxes + entire input image = 20

• 1 image $\rightarrow$ 20 h-dimensional vectors
  \[ v = W_m[\text{CNN}_{\theta_c}(I_b)] + b_m \]
Recap: Bidirectional RNNs

Forward in time

Backward in time

[Fig. 10.11]
Representing sentences

• Bidirectional RNN
• Left to right & right to left context
• Each input word $\rightarrow$ 1-of-k vector
• Encode into h-D vector (the same embedding space as images)
Alignment

- Training set:
  - $k$: image index
  - $l$: sentence index
- Multimodal h-D embedding
- Image $\rightarrow v_1, \ldots, v_{20}$
- Sentence n words $\rightarrow s_1, \ldots, s_n$
- Similarity between image region & word based on dot product $v_k^T s_t$

\[
S_{k,l} = \sum_{i \in g_l} \max_{i \in g_k} v_i^T s_t
\]  
(Eq. 8)

Figure 3. Diagram for evaluating the image-sentence score $S_{kl}$. Object regions are embedded with a CNN (left). Words (enriched by their context) are embedded in the same multimodal space with a BRNN (right). Pairwise similarities are computed with inner products (magnitudes shown in grayscale) and finally reduced to image-sentence score with Equation 8.
Multimodal RNN for text generation

- Image CNN at \( t_0 \)
- START & END: special tokens
- Each word encoded into a vector
- Predict next word as probability distribution over dictionary + END

Figure 4. Diagram of our multimodal Recurrent Neural Network generative model. The RNN takes a word, the context from previous time steps and defines a distribution over the next word in the sentence. The RNN is conditioned on the image information at the first time step. START and END are special tokens.
Qualitative result

20 occurrences of “man in black shirt”
60 occurrences of “is playing guitar”
Additional sample results

a woman sitting on a couch with a dog
logprob: -9.05

a cat is sitting on a couch with a remote control
logprob: -12.45
Show & Tell
[Vinyals et al., 2015]

Simple encoder-decoder model using fixed-length context vector
Show & Tell
[Vinyals et al., 2015]

CNN:
Inception V1-3
Batch Normalization
Show & Tell
[Vinyals et al., 2015]
Show, Attend, & Tell
[Xu et al., 2015]

1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation
Figure 3. Visualization of the attention for each generated word. The rough visualizations obtained by upsampling the attention weights and smoothing. (top) “soft” and (bottom) “hard” attention (note that both models generated the same captions in this example).
Show, Attend, & Tell
[Xu et al., 2015]

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.
Image captioning with attributes (LSTM-A) [Yao et al., 2017]

• CNN-RNN encoder-decoder model
• Predefined set of high-level attributes
• Multiple instance learning with inter-attribute correlations
Image captioning with attributes (LSTM-A)
[Yao et al., 2017]

Figure 1. Five variants of our LSTM-A framework (better viewed in color).


Table 2. Leaderboard of the published state-of-the-art image captioning models on the online COCO testing server, where B@N, M, R, and C are short for BLEU@N, METEOR, ROUGE-L, and CIDEr-D scores. All values are reported as percentage (%).

<table>
<thead>
<tr>
<th>Model</th>
<th>B@1</th>
<th>B@2</th>
<th>B@3</th>
<th>B@4</th>
<th>M</th>
<th>R</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM-A3 (Ours)</td>
<td>78.7</td>
<td>93.7</td>
<td>62.7</td>
<td>86.7</td>
<td>47.6</td>
<td>76.5</td>
<td>35.6</td>
</tr>
<tr>
<td>Watson Multimodal [24]</td>
<td>77.3</td>
<td>92.9</td>
<td>60.9</td>
<td>85.6</td>
<td>46.1</td>
<td>75.1</td>
<td>34.4</td>
</tr>
<tr>
<td>G-RMI(PG-SPIDER-TAG) [14]</td>
<td>75.1</td>
<td>91.6</td>
<td>59.1</td>
<td>84.2</td>
<td>44.5</td>
<td>73.8</td>
<td>33.1</td>
</tr>
<tr>
<td>MetaMind/VT-GT [15]</td>
<td>74.8</td>
<td>92.0</td>
<td>58.4</td>
<td>84.5</td>
<td>44.4</td>
<td>74.4</td>
<td>33.6</td>
</tr>
<tr>
<td>rewriteset [35]</td>
<td>72.0</td>
<td>90.0</td>
<td>55.0</td>
<td>81.2</td>
<td>41.4</td>
<td>70.5</td>
<td>31.3</td>
</tr>
<tr>
<td>ATT [38]</td>
<td>73.1</td>
<td>90</td>
<td>56.5</td>
<td>81.5</td>
<td>42.4</td>
<td>70.9</td>
<td>31.6</td>
</tr>
<tr>
<td>Google [31]</td>
<td>71.3</td>
<td>89.5</td>
<td>54.2</td>
<td>80.2</td>
<td>40.7</td>
<td>69.4</td>
<td>30.9</td>
</tr>
<tr>
<td>MSK Captivator [4]</td>
<td>71.5</td>
<td>90.7</td>
<td>54.3</td>
<td>81.9</td>
<td>40.7</td>
<td>71</td>
<td>30.8</td>
</tr>
</tbody>
</table>
How much data do we need to achieve decent performance in image captioning?
30K images + 150K captions

P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. TACL 2014.

Figure 1: Two images from our data set and their five captions.
MS COCO 2014

330K images x 5 captions

http://farm9.staticflickr.com/8463/8127485677_6d8d40496a_z.jpg

http://cocodataset.org
Testing on images outside datasets

[Google’s Show & Tell]

Courtesy: Andy Tsai
Testing on images outside datasets

[Show, Attend, & Tell]

Courtesy: Junjiao Tian
There’s a lot of room to improve
Next

• Wednesday papers:
  – Project presentation – Afshaan
  – Word2vec (Krishna)
  – Skip-thought vector (Satyen)

• Project midterm report