Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models

Jamie Ryan Kiros, Ruslan Salakhutdinov, Richard Zemel

Presentation by David Madras

University of Toronto

January 25, 2017
Image Captioning

???????
Image Retrieval

Google search for "a cat jumping off a bookshelf".
Introduction: Captioning and Retrieval

- **Image captioning**: the challenge of generating descriptive sentences for images
- Must consider spatial relationships between objects
- Also should generate grammatical, sensible phrases
- **Image retrieval** is related: given a query sentence, find the most relevant pictures in a database

![Caption Example: A cat jumping off a bookshelf](image)

**Figure 1**: Caption Example: A cat jumping off a bookshelf
Approaches to Captioning

1. Template based methods
   ▶ Begin with several pre-determined sentence templates
   ▶ Fill these in with object detection, analyzing spatial relationships
   ▶ Less generalizable, captions don’t feel very fluid, ”human”

2. Composition-based methods
   ▶ Extract and re-compose components of relevant, existing captions
   ▶ Try to find the most ”expressive” components
   ▶ e.g. TREETALK [Kuznetsova et al., 2014] - uses tree fragments

3. Neural Network Methods
   ▶ Sample from a conditional neural language model
   ▶ Generate description sentence by conditioning on the image

The paper we’ll talk about today fits (unsurprisingly) into the Neural Network Methods category.
High-Level Approach

- Kiros et al. take approach inspired by translation: images and text are different "languages" that can express the same concept.
- Sentences and images are embedded in same representation space; similar underlying concepts should have similar representations.
- To caption an image:
  1. Find that image's embedding
  2. Sample a point near that embedding
  3. Generate text from that point
- To do image retrieval for a sentence:
  1. Find that sentence's embedding
  2. Do a nearest neighbour search in the embedding space for images in our database
Encoder-Decoder Model

- An encoder-decoder model has two components
  - **Encoder functions** which transform data into a representation space
  - **Decoder functions** which transform a vector from representation space into data

![Network Diagram](image)

**Figure 2:** The basic encoder-decoder structure
Encoder-Decoder Model

- Kiros et al. learn these functions using neural networks. Specifically:
  - **Encoder for sentences**: recurrent neural network (RNN) with long short-term memory (LSTM)
  - **Encoder for images**: convolutional neural network (CNN)
  - **Decoder for sentences**: Structure-Content Neural Language Model
  - No decoder for images in this model - that’s a separate question

**Figure 3**: The basic encoder-decoder structure
Obligatory Model Architecture Slide

Figure 4: The model for captioning/retrieval proposed by Kiros et al.
Recurrent Neural Networks (RNNs)

- Recurrent neural networks have loops in them
- We propagate information between time steps
- Allows us to use neural networks on **sequential, variable-length** data
- Our current state is influenced by input *and* all past states

*Figure 5: A basic (vanilla) RNN*

Image from Andrej Karpathy
Recurrent Neural Networks (RNNs)

- By unrolling the network through time, an RNN has similar structure to a feedforward NN
- Weights are shared throughout time - can lead to vanishing/exploding gradient problem
- RNN’s are Turing-complete - can simulate arbitrary programs (...in theory)

![Figure 6: RNN unrolled through time](Image from Chris Olah)
RNNs for Language Models

- Language is a natural application for RNNs, as it takes a sequential, variable-length form

*Image from Jamie Kiros*
RNNs for Conditional Language Models

- We can condition our sentences on an alternate input

Image from Jamie Kiros
RNNs for Language Models: Encoders

- We can use RNNs to encode sentences in a high-dimensional representation space

![Diagram showing RNNs encoding a sentence](Image from Jamie Kiros)
Long Short-Term Memory (LSTM)

Learning long-term dependencies with RNNs can be difficult
- LSTM cells [Hochreiter, 1997] can do a better job at this
- The network explicitly learns how much to ”remember” or ”forget” at each time step
- LSTMs also help with the vanishing gradient problem

Image from Alex Graves
Learning Multimodal Distributed Representations

- Jointly optimize text/image encoders for images $x$, captions $v$
- $s(x, v)$ is cosine similarity, and $v_k$ are a set of random captions which do not describe image $x$

$$\min_{\theta} \sum_{x,k} \max(0, \alpha - s(x, v) + s(x, v_k)) + \sum_{v,k} \max(0, \alpha - s(v, x) + s(v, x_k))$$

- Maximize similarity between $x$’s embedding and its descriptions’, and minimize similarity to all other sentences
That’s the encoding half of the model - any questions?

Now we’ll talk about the decoding half.

The authors describe two types of models: log-bilinear and multiplicative.

The model they ultimately use is based on the more complex multiplicative model, but I think it’s helpful to explain both.
Log-bilinear neural language models

- In sentence generation, we model the probability of the next word given the previous words - $P(w_n|w_{1:n-1})$
- We can represent each word as a $K$-dimensional vector $w_i$
- In an LBL, we make a linear prediction of $w_n$ with

$$\hat{r} = \sum_{i=1}^{n-1} C_i w_i$$

where $\hat{r}$ is the predicted representation of $w_n$, and $C_i$ are context parameter matrices for each index
- We then use a softmax over all word representations $r_i$ to get a probability distribution over the vocabulary

$$P(w_n = i|w_{1:n-1}) = \frac{\exp(\hat{r}^T w_i + b_i)}{\sum_j^V \exp(\hat{r}^T w_j + b_j)}$$

- We learn $C_i$ through gradient descent
Suppose we have auxiliary vector $u$, e.g., an image embedding. We will model $P(w_n | w_{1:n-1}, u)$ by finding $F$ latent factors to explain the multimodal embedding space. Let $T \in \mathcal{R}^{V \times K \times G}$ be a tensor, where $V$ is vocabulary size, $K$ is word embedding dimension, $G$ is the dimension of $u$, i.e., the number of slices of $T$. We can model $T$ as a tensor factorizable into three matrices (where $W^{ij} \in \mathcal{R}^{I \times J}$)

$$T_u = (W^{fv})^T \cdot \text{diag}(W^{fg} u) \cdot W^{fk}$$

By multiplying the two outer matrices from above, we get $E = (W^{fk})^T \cdot W^{fv}$, a word embedding matrix independent of $u$. 
Multiplicative neural language models

- As in the LBL, we predict the next word representation with

\[ \hat{r} = \sum_{i=1}^{n-1} C_i E_{w_i} \]

where \( E_{w_i} \) is word \( w_i \)'s embedding, and \( C_i \) is a context matrix

- We use a softmax to get a probability distribution

\[
P(w_n = i|w_1:n-1,u) = \frac{\exp(W^{fv}(::i)f + b_i)}{\sum_j \exp(W^{fv}(::j)f + b_j)}
\]

where factor outputs \( f = (W^{fk}\hat{r}) \cdot (W^{fg}u) \) depend on \( u \)

- Effectively, this model replaces the word embedding matrix \( R \) from the LBL with the tensor \( T \), which depends on \( u \)
This model, proposed by Kiros et al. is a form of multiplicative neural language model.
We condition on a vector $v$, as above.
However, $v$ is an additive function of "content" and "structure" vectors.
- The content vector $u$ may be an image embedding.
- The structure vector $t$ is an input series of POS tags.

We are modelling $P(w_n|w_{1:n-1}, t_{n:n+k}, u)$
- Previous words and future structure
Structure-Content Neural Language Models

- We can predict a vector $\hat{v}$ of combined structure and content information (the $T$’s are context matrices)

$$
\hat{v} = \max \left( \sum_{n}^{n+k} (T(i)t_i) + T_uu + b, 0 \right)
$$

- We continue as with the multiplicative model described above
- Note that the content vector $u$ can represent an image or a sentence - using a sentence embedding as $u$, we can learn on text alone
Caption Generation

1. Embed image
2. Use image embedding and closest images/sentences in dataset to make bag of concepts
3. Get set of all "medium-length" POS sequences
4. Sample a concept conditioning vector and a POS sequence
5. Compute MAP estimate from SC-NLM
6. Generate 1000 descriptions, rank top 5 using scoring function
   ▶ Embed description
   ▶ Get cosine similarity between sentence and image embeddings
   ▶ Kneser-Ney trigram model trained on large corpus - compute log-prob of sentence
   ▶ Average the cosine similarity and the trigram model scores
Experiments: Retrieval

- Trained on Flickr8K/Flickr30K
- Each image has 5 caption sentences
- Metric is Recall-K - how often is correct caption returned in top K results? (or vice versa)
- Best results are state-of-the-art, using OxfordNet features

<table>
<thead>
<tr>
<th>Model</th>
<th>Image Annotation</th>
<th></th>
<th></th>
<th></th>
<th>Image Search</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
<td>R@10</td>
<td>Med r</td>
<td>R@1</td>
<td>R@5</td>
<td>R@10</td>
<td>Med r</td>
</tr>
<tr>
<td>Random Ranking</td>
<td>0.1</td>
<td>0.6</td>
<td>1.1</td>
<td>631</td>
<td>0.1</td>
<td>0.5</td>
<td>1.0</td>
<td>500</td>
</tr>
<tr>
<td>SDT-RNN [6]</td>
<td>4.5</td>
<td>18.0</td>
<td>28.6</td>
<td>32</td>
<td>6.1</td>
<td>18.5</td>
<td>29.0</td>
<td>29</td>
</tr>
<tr>
<td>† DeViSE [5]</td>
<td>4.8</td>
<td>16.5</td>
<td>27.3</td>
<td>28</td>
<td>5.9</td>
<td>20.1</td>
<td>29.6</td>
<td>29</td>
</tr>
<tr>
<td>† SDT-RNN [6]</td>
<td>6.0</td>
<td>22.7</td>
<td>34.0</td>
<td>23</td>
<td>6.6</td>
<td>21.6</td>
<td>31.7</td>
<td>25</td>
</tr>
<tr>
<td>DeFrag [15]</td>
<td>5.9</td>
<td>19.2</td>
<td>27.3</td>
<td>34</td>
<td>5.2</td>
<td>17.6</td>
<td>26.5</td>
<td>32</td>
</tr>
<tr>
<td>† DeFrag [15]</td>
<td>12.6</td>
<td>32.9</td>
<td>44.0</td>
<td>14</td>
<td>9.7</td>
<td>29.6</td>
<td>42.5</td>
<td>15</td>
</tr>
<tr>
<td>m-RNN [7]</td>
<td>14.5</td>
<td>37.2</td>
<td>48.5</td>
<td>11</td>
<td>11.5</td>
<td>31.0</td>
<td>42.4</td>
<td>15</td>
</tr>
<tr>
<td>Our model</td>
<td>13.5</td>
<td>36.2</td>
<td>45.7</td>
<td>13</td>
<td>10.4</td>
<td>31.0</td>
<td>43.7</td>
<td>14</td>
</tr>
<tr>
<td>Our model (OxfordNet)</td>
<td><strong>18.0</strong></td>
<td><strong>40.9</strong></td>
<td><strong>55.0</strong></td>
<td><strong>8</strong></td>
<td><strong>12.5</strong></td>
<td><strong>37.0</strong></td>
<td><strong>51.5</strong></td>
<td><strong>10</strong></td>
</tr>
</tbody>
</table>

Figure 7: Flickr8K retrieval results
Experiments: Retrieval

- Trained on Flickr8K/Flickr30K
- Each image has 5 caption sentences
- Metric is Recall-K - how often is correct caption returned in top K results? (or vice versa)
- Best results are state-of-the-art, using OxfordNet features

<table>
<thead>
<tr>
<th>Model</th>
<th>Image Annotation</th>
<th>Image Search</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td>Random Ranking</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>† DeViSE [5]</td>
<td>4.5</td>
<td>18.1</td>
</tr>
<tr>
<td>† SDT-RNN [6]</td>
<td>9.6</td>
<td>29.8</td>
</tr>
<tr>
<td>† DeFrag [15]</td>
<td>14.2</td>
<td>37.7</td>
</tr>
<tr>
<td>† DeFrag + Finetune CNN [15]</td>
<td>16.4</td>
<td>40.2</td>
</tr>
<tr>
<td>m-RNN [7]</td>
<td>18.4</td>
<td>40.2</td>
</tr>
<tr>
<td>Our model</td>
<td>14.8</td>
<td>39.2</td>
</tr>
<tr>
<td>Our model (OxfordNet)</td>
<td><strong>23.0</strong></td>
<td><strong>50.7</strong></td>
</tr>
</tbody>
</table>

Figure 8: Flickr30K retrieval results
Qualitative Results - Caption Generation Successes

- Generation is difficult to evaluate quantitatively.

- A car is parked in the middle of nowhere.
- A wooden table and chairs arranged in a room.
- There is a cat sitting on a shelf.
- A ferry boat on a marina with a group of people.
- A little boy with a bunch of friends on the street.
Qualitative Results - Caption Generation Failures

- Generation is difficult to evaluate quantitatively

- the two birds are trying to be seen in the water. (can't count)
- a giraffe is standing next to a fence in a field. (hallucination)
- a parked car while driving down the road. (contradiction)
- the handlebars are trying to ride a bike rack. (nonsensical)
- a woman and a bottle of wine in a garden. (gender)
Qualitative Results - Analogies

- We can do analogical reasoning, modelling an image as roughly the sum of its components

- dog + cat =

- cat + dog =

- plane + bird =

- man + woman =
Qualitative Results - Analogies

- We can do analogical reasoning, modelling an image as roughly the sum of its components

- blue + red =
- blue + yellow =
- yellow + red =
- white + red =

Nearest images
Qualitative Results - Analogies

- We can do analogical reasoning, modelling an image as roughly the sum of its components

- day + night =

- flying + sailing =

- bowl + box =

- box + bowl =

Nearest images
Conclusions

- In their paper, Kiros et al. present a model for image captioning and retrieval
- The model is inspired by translation systems, and aims to jointly embed images and their captions in the same space
- To decode from the representation space, we condition on an auxiliary content vector (such as an image or sentence representation) and a structure vector (such as POS tags)
- Since the publication of this paper, advances have been made on related problems, such as:
  - Image generation from a given caption
  - Attention-based captioning
  - State of the art caption generation on the MS-COCO dataset are Google’s model (Show and Tell: A Neural Image Caption Generator, 2015) and MSR’s model (From Captions to Visual Concepts and Back, 2015) with 32% of captions passing the Turing test, compared to 16% for this model
Questions?

Thanks for your attention!