Deep Learning for Question Answering

Mohit Iyyer
Who wrote the song “Kiss from a Rose”? 
Who wrote the song “Kiss from a Rose”?

Question Analysis: POS/Parsing/NER
Who wrote the song “Kiss from a Rose”?
Who wrote the song “Kiss from a Rose”?

Question Analysis: POS/Parsing/NER

Query Formulation/Template Extraction

Knowledge Base Search/Candidate Answer Generation
Who wrote the song “Kiss from a Rose”?

Question Analysis: POS/Parsing/NER

Query Formulation/Template Extraction

Knowledge Base Search/Candidate Answer Generation

Evidence Retrieval/Candidate Scoring
Who wrote the song “Kiss from a Rose”?

1. Question Analysis: POS/Parsing/NER
2. Query Formulation/Template Extraction
3. Knowledge Base Search/Candidate Answer Generation
4. Evidence Retrieval/Candidate Scoring
5. Answer Type Selection
Who wrote the song “Kiss from a Rose”?
Who wrote the song “Kiss from a Rose”?

Seal

Knowledge Base Search/ Candidate Answer Generation

Evidence Retrieval/ Candidate Scoring

Final Ranking

Answer Type Selection

Question Analysis: POS/Parsing/NER

Query Formulation/ Template Extraction
Can we replace all of these modules with a single neural network?

Who wrote the song “Kiss from a Rose”? Seal
Can we replace all of these modules with a single neural network?

Who wrote the song “Kiss from a Rose”? Seal
Outline

• Briefly: deep learning + NLP basics
• Factoid QA
• Reasoning-based QA
• Visual QA
• Future directions!
Neural Networks for NLP
Let’s start with words

- Represent words by low-dimensional vectors called embeddings

- e.g., president $\rightarrow [0.23, 1.3, -0.3, 0.43]$
Computing Vectors for Questions

• How do we compose word embeddings into vectors that capture the meanings of questions?

Who wrote Macbeth?
Computing Vectors for Questions

• How do we compose word embeddings into vectors that capture the meanings of questions?

\[ g( \text{Who} \ \text{wrote} \ \text{Macbeth} \ ? ) = \]
Computing Vectors for Questions

• How do we compose word embeddings into vectors that capture the meanings of questions?

Who wrote Macbeth?

Neural Net!
Recurrent Neural Networks

Who
\( c_1 \)
directed
\( c_2 \)
Predator
\( c_3 \)
?
\( c_4 \)
Recurrent Neural Networks

\[ h_1 = f(W \left[ \begin{array}{c} c_1 \\ \cdots \end{array} \right]) \]

Who
\( c_1 \)
directed
\( c_2 \)
Predator
\( c_3 \)
? 
\( c_4 \)
Recurrent Neural Networks

\[ h_1 = f(W \begin{bmatrix} \cdots \end{bmatrix}) \]

Hidden layer

Who \( c_1 \)
directed \( c_2 \)
Predator \( c_3 \)
? \( c_4 \)

Who directed Predator?
Recurrent Neural Networks

\[ h_1 = f(W \begin{bmatrix} \cdots \end{bmatrix}c_1) \quad h_2 = f(W \begin{bmatrix} h_1 \end{bmatrix}c_2) \]

Hidden layer

Who \quad \text{directed} \quad \text{Predator} \quad ?

\begin{align*}
c_1 & \quad c_2 & \quad c_3 & \quad c_4
\end{align*}
Recurrent Neural Networks

\[ h_1 = f(W \begin{bmatrix} c_1 \\ \vdots \end{bmatrix}) \quad h_2 = f(W \begin{bmatrix} h_1 \\ c_2 \end{bmatrix}) \quad h_3 = f(W \begin{bmatrix} h_2 \\ c_3 \end{bmatrix}) \]
Recurrent Neural Networks

\[ h_1 = f(W \begin{bmatrix} \cdots \\ c_1 \end{bmatrix}) \quad h_2 = f(W \begin{bmatrix} h_1 \\ c_2 \end{bmatrix}) \quad h_3 = f(W \begin{bmatrix} h_2 \\ c_3 \end{bmatrix}) \quad h_4 = f(W \begin{bmatrix} h_3 \\ c_4 \end{bmatrix}) \]
Recurrent Neural Networks

who

directed

Predator

softmax: predict answer

\[
\begin{align*}
  h_1 &= f(W \begin{bmatrix} \cdots \end{bmatrix} c_1) \\
  h_2 &= f(W \begin{bmatrix} h_1 \end{bmatrix} c_2) \\
  h_3 &= f(W \begin{bmatrix} h_2 \end{bmatrix} c_3) \\
  h_4 &= f(W \begin{bmatrix} h_3 \end{bmatrix} c_4)
\end{align*}
\]

Hidden layer
Who

directed

Predator

?
Recurrent Neural Networks

More complex variants: LSTMs, GRUs

\[ h_1 = f(W \begin{bmatrix} \cdots \end{bmatrix} c_1) \quad h_2 = f(W \begin{bmatrix} h_1 \end{bmatrix} c_2) \quad h_3 = f(W \begin{bmatrix} h_2 \end{bmatrix} c_3) \quad h_4 = f(W \begin{bmatrix} h_3 \end{bmatrix} c_4) \]

Hidden layer

softmax: predict answer
Recursive Neural Networks

• $g$ can also depend on a parse tree of the input text sequence
Recursive Neural Networks

- $g$ can also depend on a parse tree of the input text sequence.

$$h_1 = f(W \begin{bmatrix} c_2 \\ c_3 \end{bmatrix})$$
Recursive Neural Networks

- \( g \) can also depend on a parse tree of the input text sequence

\[
h_1 = f(W \begin{bmatrix} c_2 \\ c_3 \end{bmatrix})
\]

\[
h_2 = f(W \begin{bmatrix} h_1 \\ c_4 \end{bmatrix})
\]
Recursive Neural Networks

- $g$ can also depend on a parse tree of the input text sequence.

$$h_3 = f(W \begin{bmatrix} c_1 \\ h_2 \end{bmatrix})$$

$$h_2 = f(W \begin{bmatrix} h_1 \\ c_4 \end{bmatrix})$$

$$h_1 = f(W \begin{bmatrix} c_2 \\ c_3 \end{bmatrix})$$

Who \hspace{2em} directed \hspace{2em} Predator \hspace{2em} ?

$c_1$ \hspace{2em} $c_2$ \hspace{2em} $c_3$ \hspace{2em} $c_4$
Recursive Neural Networks

- \( g \) can also depend on a parse tree of the input text sequence.

\[
\begin{align*}
h_3 &= f(W \begin{bmatrix} c_1 \\ h_2 \end{bmatrix}) \\
h_2 &= f(W \begin{bmatrix} h_1 \\ c_4 \end{bmatrix}) \\
h_1 &= f(W \begin{bmatrix} c_2 \\ c_3 \end{bmatrix}) \\
\end{align*}
\]

softmax: predict answer
Deep Averaging Networks
Deep Averaging Networks

\[
\text{av} = \sum_{i=1}^{4} \frac{c_i}{4}
\]
Deep Averaging Networks

\[ z_1 = f(W_1 \cdot av) \]

\[ av = \sum_{i=1}^{4} \frac{c_i}{4} \]
Deep Averaging Networks

\[ z_2 = f(W_2 \cdot z_1) \]

\[ z_1 = f(W_1 \cdot av) \]

\[ av = \sum_{i=1}^{4} \frac{c_i}{4} \]

Who \hspace{1cm} directed \hspace{1cm} Predator \hspace{1cm} ?

C_1 \hspace{1cm} C_2 \hspace{1cm} C_3 \hspace{1cm} C_4
Deep Averaging Networks

softmax: predict answer

\[ z_2 = f(W_2 \cdot z_1) \]

\[ z_1 = f(W_1 \cdot \text{av}) \]

\[ \text{av} = \sum_{i=1}^{4} \frac{c_i}{4} \]

Who \quad \text{directed} \quad \text{Predator} \quad ?

C_1 \quad C_2 \quad C_3 \quad C_4
Softmax Answer Classification

- Multinomial logistic regression

\[ \hat{y}_p = \text{softmax}(W_{\text{ans}} \cdot h_q) \]

\[ \text{softmax}(q) = \frac{\exp q}{\sum_{j=1}^{k} \exp q_j} \]

- Output is a distribution over a finite set of answers

- Later on: a max-margin answer ranking approach can yield better results
How do we train these models?

• Model parameters learned through variants of *backpropagation* (Rumelhart et al., 1986; Goller and Kuchler, 1996) given QA pairs as input

• In theory, use the chain rule to compute partial derivatives of the error function with respect to every parameter

• In practice, use Theano (or Torch) and never have to compute any derivatives by hand!
Application 1: Quiz Bowl
Factoid QA

• Given a description of an entity, identify the person, place, or thing discussed.

• Neural nets never previously applied to this task

• Traditionally approached using information retrieval, querying huge knowledge bases for the answer
Quiz Bowl
Quiz Bowl

This creature has female counterparts named Penny and Gown.
Quiz Bowl

This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.
This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature’s counterparts include Daemon on the Berkeley Software Distribution, or BSD.
This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature’s counterparts include Daemon on the Berkeley Software Distribution, or BSD.

For ten points, name this mascot of the Linux operating system, a penguin whose name refers to formal male attire.
This creature has female counterparts named Penny and Gown.

This creature appears dressed in Viking armor and carrying an ax when he is used as the mascot of PaX, a least privilege protection patch.

This creature’s counterparts include Daemon on the Berkeley Software Distribution, or BSD.

For ten points, name this mascot of the Linux operating system, a penguin whose name refers to formal male attire.

Answer: Tux
Simple Approach!

Identify this mascot of Linux…

Tux
Simple Approach!

Identify this mascot of Linux…

Tux
Two Neural Models

• Dependency-tree recursive neural network (DT-RNN)

• Deep averaging network (DAN)

• Both models are initialized with pretrained word2vec embeddings and have the same hidden layer dimensionality for fair comparison
Experimental Datasets

- History: 4,415 QA pairs with 16,895 sentences and 451 unique answers
- Literature: 5,685 QA pairs with 21,549 sentences and 595 unique answers
Choosing an Error Function

• Answers can appear as part of question text (e.g., a question on *World War II* might mention the *Battle of the Bulge* and vice versa)

• Instead of using a *softmax* output layer, can we take advantage of these co-occurrences by modeling answers and questions in the same vector space?
Max-Margin Objective

- Replace **softmax** output layer with a contrastive max-margin function

- Given a question \(q\) with correct answer \(a\) and an incorrect answer \(b\), the loss is

\[
\max(0, 1 - x_a \cdot h_q + x_b \cdot h_q)
\]
Q: Name this fascist leader of Italy

- Benito Mussolini

Walt Disney

Rihanna
Geometric Intuition

Q: Name this fascist leader of Italy

Benito Mussolini

Maximize inner product

Walt Disney

Rihanna
Geometric Intuition

Q: name this fascist leader of Italy
Benito Mussolini

Minimize inner product
Walt Disney

Maximize inner product
Rihanna
## Experiments

<table>
<thead>
<tr>
<th>Model</th>
<th>History</th>
<th></th>
<th></th>
<th>Literature</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pos 1</td>
<td>Pos 2</td>
<td>Full</td>
<td>Pos 1</td>
<td>Pos 2</td>
<td>Full</td>
</tr>
<tr>
<td>BOW-DT</td>
<td>35.4</td>
<td>57.7</td>
<td>60.2</td>
<td>24.4</td>
<td>51.8</td>
<td>55.7</td>
</tr>
<tr>
<td>IR</td>
<td>37.5</td>
<td>65.9</td>
<td>71.4</td>
<td>27.4</td>
<td>54.0</td>
<td>61.9</td>
</tr>
<tr>
<td>DAN</td>
<td>46.4</td>
<td>70.8</td>
<td>71.8</td>
<td>35.3</td>
<td>67.9</td>
<td>69.0</td>
</tr>
<tr>
<td>DT-RNN</td>
<td>47.1</td>
<td>72.1</td>
<td>73.7</td>
<td>36.4</td>
<td>68.2</td>
<td>69.1</td>
</tr>
</tbody>
</table>

**DAN** is 20 times faster to train than **DT-RNN**
Learning a Vector Space
Exact String Matches

• One current weakness of neural models

In this poem, the narrator meets a “traveller from an antique land” who tells of a statue with a “wrinkled lip, and sneer of cold command”.

• Practical solution: train language model on original source material / Wikipedia and combine with output of neural network
QA: Man vs. Machine

• Scaled up a **DAN** to handle ~100k Q/A pairs with ~5k unique answers! Also added thousands of Wikipedia sentence/page-title pairs

• To play against humans, we need to decide not only what answer to give but also when we are confident enough to buzz in.

• Another classifier re-ranks the top 200 guesses of the **DAN** using language model features to decide whether to buzz on any of them or wait for more clues
V1: tied team of ex-Jeopardy champions 200-200
V2: defeated Ken Jennings
300-160
Code available!

DT-RNN code: cs.umd.edu/~miyyer/qblearn
DAN code: github.com/miyyer/dan
Full quiz bowl system code: github.com/miyyer/qb
Video of Ken Jennings match: youtu.be/kTXJCEvCDYk

Application 2:
Reasoning-based QA
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
Naïve Neural Approach

![Diagram of Naïve Neural Approach](http://smerity.com/articles/2015/keras_qa.html)
Naïve Neural Approach

[Diagram showing the process of word vectors fed into an RNN for both the story and query, resulting in a merged output which is then passed through a softmax layer to predict the answer.]

Problems

• Doesn’t scale to long / complex question types

• RNNs/LSTMs are very bad at remembering facts from the distant past!

• Solution: add an external memory component that learns to store important facts and reason about them
Dynamic Memory Networks

• Collaboration with Richard Socher and colleagues from MetaMind

• Extends simple RNNs with an iterative attention mechanism that focuses on one fact at a time and enables transitive reasoning

1. Compute vector $s_i$ for every sentence in input and vector $q$ for the question using recurrent network $A$.
1. Compute vector $\mathbf{s}_i$ for every sentence in input and vector $\mathbf{q}$ for the question using recurrent network $A$

2. Compute an attention score $a_i$ for every sentence

$$a_i = G(s_i, m_{t-1}, q)$$
1. Compute vector $s_i$ for every sentence in input and vector $q$ for the question using recurrent network $A$

2. Compute an attention score $a_i$ for every sentence

   $$a_i = G(s_i, m_{t-1}, q)$$

3. Compute an episodic memory $m_t$ by weighting each $s_i$ with its corresponding $a_i$ and passing them through another recurrent network $B$
1. Compute vector $s_i$ for every sentence in input and vector $q$ for the question using recurrent network $A$

2. Compute an *attention score* $a_i$ for every sentence

\[ a_i = G(s_i, m_{t-1}, q) \]

3. Compute an *episodic memory* $m_t$ by weighting each $s_i$ with its corresponding $a_i$ and passing them through another recurrent network $B$

4. Repeat until network $B$ outputs a “finished reading” signal
1. Compute vector $s_i$ for every sentence in input and vector $q$ for the question using recurrent network A

2. Compute an attention score $a_i$ for every sentence

$$a_i = G(s_i, m_{t-1}, q)$$

3. Compute an episodic memory $m_t$ by weighting each $s_i$ with its corresponding $a_i$ and passing them through another recurrent network B

4. Repeat until network B outputs a “finished reading” signal

5. Feed final episodic memory $m$ to a softmax layer to predict the answer
John moved to the bedroom. Mary grabbed the football there. Sandra journeyed to the bedroom. Sandra went back to the hallway. Mary moved to the garden. Mary journeyed to the office.

Where is the football?
John moved to the bedroom. Mary grabbed the football there. Sandra journeyed to the bedroom. Sandra went back to the hallway. Mary moved to the garden. Mary journeyed to the office.

Where is the football?
John moved to the bedroom. Mary grabbed the football there.
Sandra journeyed to the bedroom. Sandra went back to the hallway.
Mary moved to the garden. Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
John moved to the bedroom.
Mary grabbed the football there.
Sandra journeyed to the bedroom.
Sandra went back to the hallway.
Mary moved to the garden.
Mary journeyed to the office.

Where is the football?
Evaluation: FB bAbi

- 20 very simple tasks (e.g., counting, basic deduction, induction, coreference)

- DMNs solve 18 out of 20 tasks with over 95% accuracy, comparable to other baselines that use hand-engineered features (e.g., n-grams, positional features)

- Can also be applied to many other NLP tasks (what is the sentiment of this sentence? what is this sentence’s translation in French?)
Application 3: Visual QA
• Is this truck considered “vintage”?
• Does the road look new?
• What kind of tree is behind the truck?
VisualQA Dataset

• collaboration between Virginia Tech and Microsoft Research

• questions were created and answered by Amazon Turkers (800k questions on 250k images)

“We have built a smart robot. It understands a lot about images. It can recognize and name all the objects, it knows where the objects are, it can recognize the scene (e.g., kitchen, beach), people’s expressions and poses, and properties of objects (e.g., color of objects, their texture). Your task is to stump this smart robot!”
Brief Aside: ConvNets

Convolutional Layers:
slide a set of small filters over the image

Pooling Layers:
reduce dimensionality of representation

image: https://cs231n.github.io/convolutional-networks/
\[ \text{ConvNet}() = \]
$ConvNet\left(\right) = \text{softmax: predict ‘truck’}$
Naïve VisualQA

• $i = \text{ConvNet}(\text{image}) > \text{use an existing network trained for image classification and freeze weights}$

• $q = RNN(\text{question}) > \text{learn weights}$

• answer = $\text{softmax}(\langle i; q \rangle)$
Visual Attention

- Use the question representation $q$ to determine where in the image to look

How many benches are shown?
Visual Attention

• Use the question representation $q$ to determine where in the image to look

How many benches are shown?
How many benches are shown?
How many benches are shown?
attention over final convolutional layer in network: 196 boxes, captures color and positional information

How many benches are shown?
attention over final convolutional layer in network: 196 boxes, captures color and positional information

softmax:

predict answer

How many benches are shown?
Issues

• Visual attention is more complicated than textual attention; requires many more QA pairs than are currently available

• focusing on more than one “box” at a time is difficult for the current model; perhaps an iterative attention mechanism like the DMN’s can solve this problem

• Work in progress, full evaluation coming soon!
Closing Thoughts
Future Trends

• Neural networks with attention mechanisms are cutting-edge models with broad applications!

• With more data and bigger networks, we can begin to answer more complex questions

• Multi-task learning, such as a single model that learns to reason over both text and images
A Major Limitation

- All of these networks generalize very poorly to new facts or information at test-time, would fail at:

xxwf moved to the rfecs.
dawas grabbed the gndsa there.
gfdg journeyed to the klnmkb.
gfdg went back to the aqqzs.
dawas moved to the mnsh.
dawas journeyed to the taaaed.

Where is the gndsa?
Constants vs. Variables

• Currently, every word in a question is represented with an embedding.

• This doesn’t make much sense for numbers, proper names, or other entities

An amusement park sells 2 kinds of tickets. Tickets for children cost $1.50. Adult tickets cost $4. On a certain day, 278 people entered the park. On that same day the admission fees collected totaled $792.

How many children were admitted on that day?
Thanks! Questions?

And thanks to my advisors, Jordan Boyd-Graber at U. Colorado and Hal Daumé III at UMD, and to Richard Socher and colleagues from MetaMind.