RNN extensions, Memory Addressing and Attention

Eilif Solberg

21.09.2018

Outline

Composing RNNs

Bidirectional RNNs

Motivation:

- Want to include future context

Bidirectional RNNs

Motivation:

- Want to include future context
- Could solve with time-delay for predictions, though need to specify fixed context.

Bidirectional RNNs

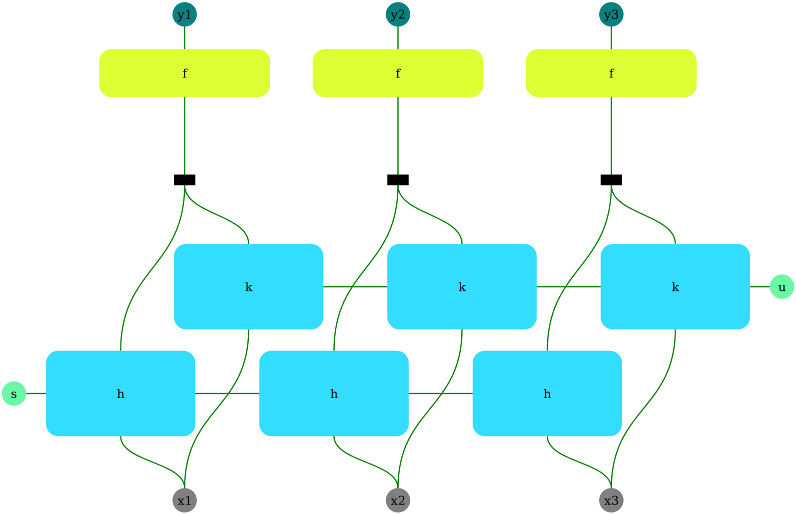
Motivation:

- Want to include future context
- Could solve with time-delay for predictions, though need to specify fixed context.

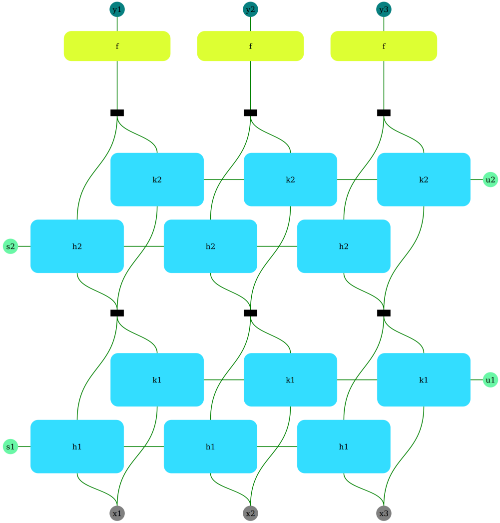
Assumes tight coupling between prediction at time t and input at time t.

- e.g. speech-to-text, text-to-speech

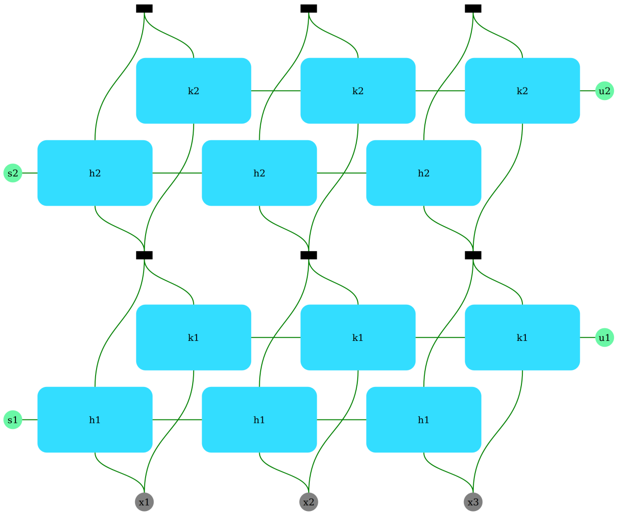
Bidirectional RNN - single layer



Bidirectional RNN - two layers



Bidirectional RNN - feature extraction



Encoder-decoder

For sequence-to-sequence problems with loose coupling between sequences

- prediction at time t not directly related to input at time t.

Encoder-decoder

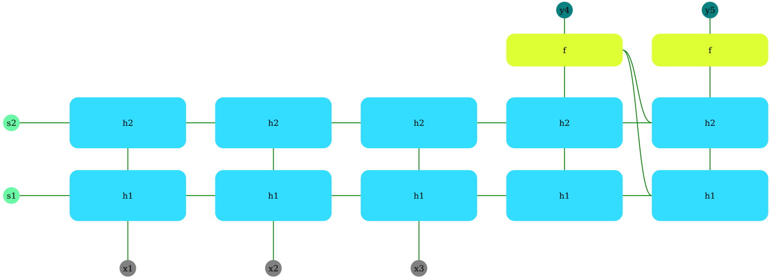
For sequence-to-sequence problems with loose coupling between sequences

- prediction at time t not directly related to input at time t.

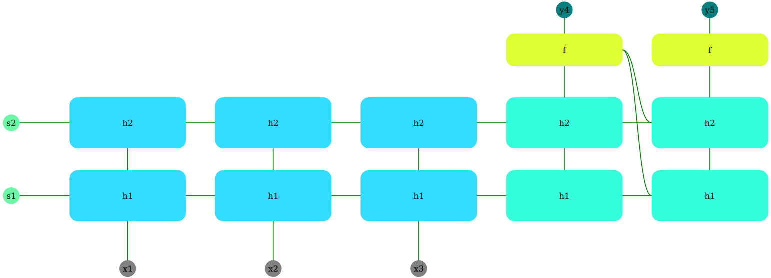
Example: sentence translation

1. Encode the "meaning" of sentence in source language into intermediate representation
2. Decode the "meaning" of the sentence into a representation in the target language

Encoder-decoder, shared RNN



Encoder-decoder, separate RNN



Encoder-decoder conclusion

Assume N source and N target languages.

- Want to be able to translate between any two of them

Encoder-decoder conclusion

Assume N source and N target languages.

- Want to be able to translate between any two of them
- Possible to share encoder and decoder?

Encoder-decoder conclusion

Assume N source and N target languages.

- Want to be able to translate between any two of them
- Possible to share encoder and decoder?

Newer models include attention

- Bidirectional RNN may then be used as *encoder*

Recursive neural networks

RNN Memory extensions

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \ldots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \ldots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

How do we *address* memory?

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \ldots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

How do we *address* memory?

Location

- Specify *where* to get information, e.g. index $j \in \{1, \ldots, J\}$
  - "Give me the content at memory cell 4"

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \dots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

How do we *address* memory?

Location

- Specify *where* to get information, e.g. index $j \in \{1, \dots, J\}$
  - "Give me the content at memory cell 4"
- *Direct* addressing

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \ldots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

How do we *address* memory?

Location

- Specify *where* to get information, e.g. index $j \in \{1, \ldots, J\}$
  - "Give me the content at memory cell 4"
- *Direct* addressing

Content

- Specify what kind of information through a *query* q
  - "When did the french revolution start?"

Addressing: location vs content-based

Assume we have memory M with memory cells $M_1, \ldots, M_J$.

- E.g. $M_j \in \mathbb{R}^n$

How do we *address* memory?

Location

- Specify *where* to get information, e.g. index $j \in \{1, \ldots, J\}$
  - "Give me the content at memory cell 4"
- *Direct* addressing

Content

- Specify what kind of information through a *query* q
  - "When did the french revolution start?"
- *Indirect* addressing

Content-based addressing

- Memory M with memory cells $M_1, \ldots, M_J$.

Content-based addressing

- Memory M with memory cells $M_1, \dots, M_J$.
- query $q \in \mathbb{R}^d$

Content-based addressing

- Memory M with memory cells $M_1, \ldots, M_J$.
- query $q \in \mathbb{R}^d$
- key function K, e.g. $K : \mathbb{R}^n \to \mathbb{R}^d$

Content-based addressing

- Memory M with memory cells $M_1, \ldots, M_J$.
- query $q \in \mathbb{R}^d$
- key function K, e.g. $K : \mathbb{R}^n \to \mathbb{R}^d$
- matching function f, e.g. inner product function

$$\alpha_j = f(q, K(M_j))$$

Content-based addressing

- Memory M with memory cells $M_1, \dots, M_J$.
- query $q \in \mathbb{R}^d$
- key function K, e.g. $K : \mathbb{R}^n \to \mathbb{R}^d$
- matching function f, e.g. inner product function

$$\alpha_j = f(q, K(M_j))$$

What is the returned result of our query?

Content-based addressing

- Memory M with memory cells $M_1, \ldots, M_J$.
- query $q \in \mathbb{R}^d$
- key function K, e.g. $K : \mathbb{R}^n \to \mathbb{R}^d$
- matching function f, e.g. inner product function

$$\alpha_j = f(q, K(M_j))$$

What is the returned result of our query?

$$p = \text{softmax}(\alpha)$$

$$v(q, M) = M_j \text{ with probability } p_j \qquad \text{hard addressing}$$

$$v(q, M) = \sum_{j=1}^{J} p_j M_j \qquad \text{soft addressing}$$

Content-based addressing

- Memory M with memory cells $M_1, \ldots, M_J$.
- query $q \in \mathbb{R}^d$
- key function K, e.g. $K : \mathbb{R}^n \to \mathbb{R}^d$
- matching function f, e.g. inner product function

$$\alpha_j = f(q, K(M_j))$$

What is the returned result of our query?

$$p = \mathrm{softmax}(\alpha)$$

$$v(q, M) = M_j \text{ with probability } p_j \qquad \text{hard addressing}$$

$$v(q, M) = \sum_{j=1}^{J} p_j M_j \qquad \text{soft addressing}$$

Where does the query vector q come from?

RNN example with external memory - read operation

RNN example with external memory - read operation

Perform query based on current state

$$q^t = Q^{(r)}(s^t)$$

RNN example with external memory - read operation

Perform query based on current state

$$q^t = Q^{(r)}(s^t)$$

Extract key for each memory cell

$$k_j^t = K^{(r)}(M_j^{t-1})$$

RNN example with external memory - read operation

Perform query based on current state

$$q^t = Q^{(r)}(s^t)$$

Extract key for each memory cell

$$k_j^t = K^{(r)}(M_j^{t-1})$$

Calculate how well memory cell match query

$$\alpha_j^t = f(q^t, k_j^t)$$

RNN example with external memory - read operation

Perform query based on current state

$$q^t = Q^{(r)}(s^t)$$

Extract key for each memory cell

$$k_j^t = K^{(r)}(M_j^{t-1})$$

Calculate how well memory cell match query

$$\alpha_j^t = f(q^t, k_j^t)$$

Get resulting vector r^t by

$$p^t = \text{softmax}(\alpha^t)$$

$$r^t = v(q^t, M^{t-1}) = \sum_{j=1}^{J} p_j^t M_j^{t-1}$$

RNN example with external memory - write operation

RNN example with external memory - write operation

Need to decide *what* to write in addition to *where*

RNN example with external memory - write operation

Need to decide *what* to write in addition to *where*

- *Where* can be decided as with read operation
  - Separate functions $Q^{(w)}$ and $K^{(w)}$.

RNN example with external memory - write operation

Need to decide *what* to write in addition to *where*

- *Where* can be decided as with read operation
  - Separate functions $Q^{(w)}$ and $K^{(w)}$.
- *What*: e.g. function W

$$w^t = W(s^t)$$

RNN example with external memory - write operation

Need to decide *what* to write in addition to *where*

- *Where* can be decided as with read operation
  - Separate functions $Q^{(w)}$ and $K^{(w)}$.

- *What*: e.g. function W

$$w^t = W(s^t)$$

How to make update?

RNN example with external memory - write operation

Need to decide *what* to write in addition to *where*

- *Where* can be decided as with read operation
  - Separate functions $Q^{(w)}$ and $K^{(w)}$.
- *What*: e.g. function W

$$w^t = W(s^t)$$

How to make update?

$$M_j^t = (1 - p_j^w) M_j^{t-1} + p_j^w w^t \qquad \text{overwrite}$$

$$M_j^t = M_j^{t-1} + p_j^w w^t \qquad \text{residual update}$$

Exists corresponding *hard* update rules

RNN example with external memory - how to use it

Update function:

$$s^t = h(x^t, s^{t-1}, y^{t-1}, r^{t-1})$$

RNN example with external memory - how to use it

Update function:

$$s^t = h(x^t, s^{t-1}, y^{t-1}, r^{t-1})$$

Could also add directly to output function

$$y^t = f(s^t, r^t)$$

External memory - multiple read/write *heads*

- E.g. define N query, key function pairs $(Q_1^{(r)}, K_1^{(r)}), \ldots,$ $(Q_N^{(r)}, K_N^{(r)})$
  - Concatenate all of the retrieved vectors, $r^t = (r_1^t, \ldots, r_n^t)$.

External memory - multiple read/write *heads*

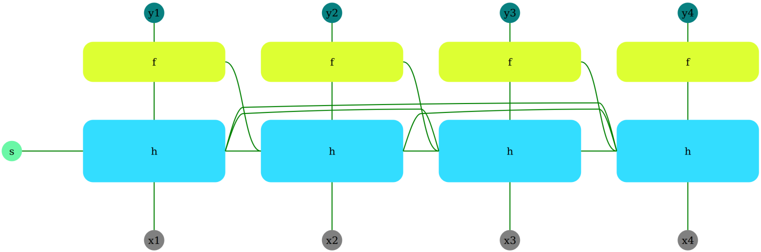
- E.g. define N query, key function pairs $(Q_1^{(r)}, K_1^{(r)})$, $\ldots$, $(Q_N^{(r)}, K_N^{(r)})$
  - Concatenate all of the retrieved vectors, $r^t = (r_1^t, \ldots, r_n^t)$.
- Write operations, need to resolve possible conflicts in updates

External memory - multiple read/write *heads*

- E.g. define N query, key function pairs $(Q_1^{(r)}, K_1^{(r)})$, ..., $(Q_N^{(r)}, K_N^{(r)})$
  - Concatenate all of the retrieved vectors, $r^t = (r_1^t, \ldots, r_n^t)$.
- Write operations, need to resolve possible conflicts in updates
- May use same matching function

Attending to previous states I

Attending to previous states I



Attending to previous states II

$$s^t = h(x^t, (s^1, \ldots, s^{t-1}), y^{t-1})$$

Attending to previous states II

$$s^t = h(x^t, (s^1, \ldots, s^{t-1}), y^{t-1})$$

Do query with respect to "memory cells" $(s^1, \ldots, s^{t-1})$.

$$\begin{aligned} \alpha_i^t &= f(Q(s^{t-1}, x^t), K(s^i)) \\ p^t &= \mathrm{softmax}(\alpha^t) \\ \tilde{s}^{t-1} &= \sum_{i=1}^{t-1} p_i^t s^i \end{aligned}$$

Attending to previous states II

$$s^t = h(x^t, (s^1, \ldots, s^{t-1}), y^{t-1})$$

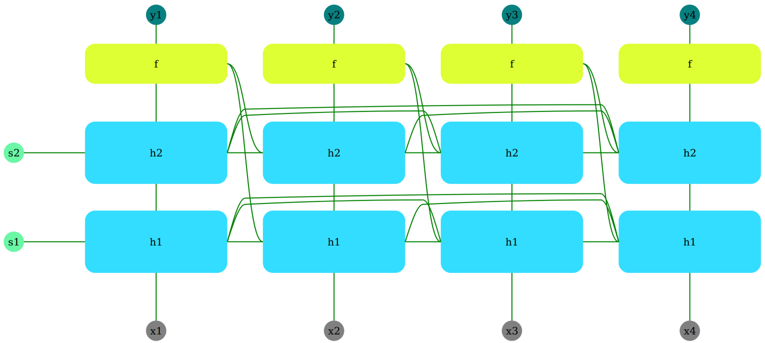
Do query with respect to "memory cells" $(s^1, \ldots, s^{t-1})$.

$$\begin{aligned}
\alpha_i^t &= f(Q(s^{t-1}, x^t), K(s^i)) \\
p^t &= \mathrm{softmax}(\alpha^t) \\
\tilde{s}^{t-1} &= \sum_{i=1}^{t-1} p_i^t s^i
\end{aligned}$$

Then proceed with "previous state" $\tilde{s}^{t-1}$

$$s^t = h(x^t, \tilde{s}^{t-1}, y^{t-1})$$

Attending to previous states III



Attention

Image captioning with RNN and content-based attention

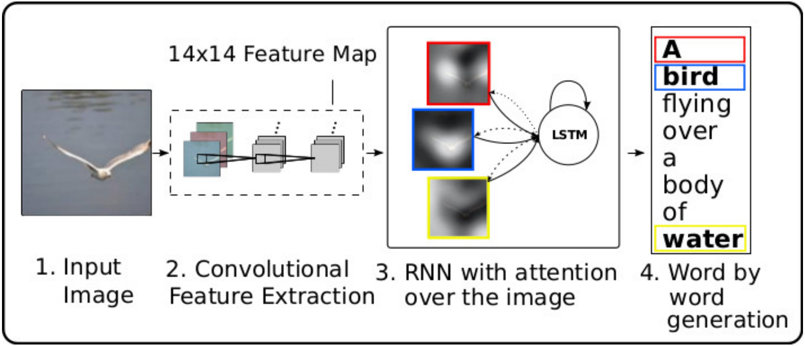


Figure: Illustration from "Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International conference on machine learning. 2015."

Image captioning with RNN and content-based attention

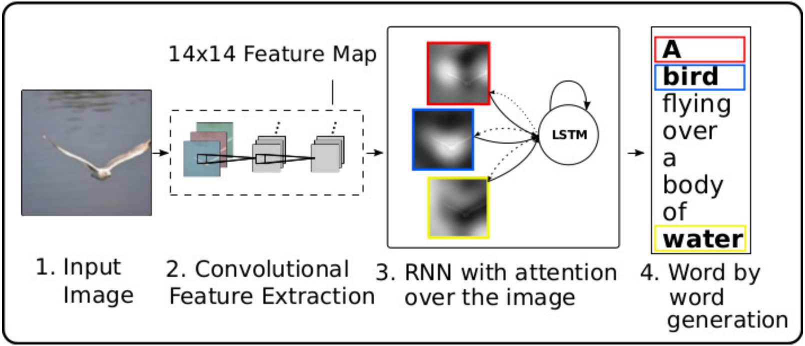


Figure: Illustration from "Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International conference on machine learning. 2015."

- Content based addressing with 14×14 conv features as "memory"

Image captioning with RNN and content-based attention

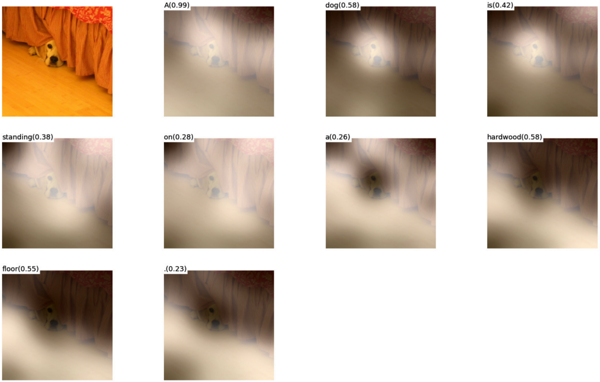


Figure: Illustration from "Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International conference on machine learning. 2015."

Image classification with RNN and location-based attention

Pseudoalgorithm:

[1] A *glimpse* is here defined as a crop of the image

Image classification with RNN and location-based attention

Pseudoalgorithm:

- Start with center/random glimpse[1] with center l^0

[1] A *glimpse* is here defined as a crop of the image

Image classification with RNN and location-based attention

Pseudoalgorithm:

- Start with center/random glimpse[1] with center l^0
- For $t = 1, \ldots, \tau$
  1. Extract glimpse with center at l^{t-1}.
  2. Extract features for location, e.g. with convnet
  3. Update state of RNN
  4. - if $t < \tau$: predict next glimpse center l^t
     - else: Make prediction/classification based on l^τ

[1] A *glimpse* is here defined as a crop of the image

Image classification with RNN and location-based attention

Pseudoalgorithm:

- Start with center/random glimpse[1] with center l^0
- For $t = 1, \ldots, \tau$
  1. Extract glimpse with center at l^{t-1}.
  2. Extract features for location, e.g. with convnet
  3. Update state of RNN
  4. - if $t < \tau$: predict next glimpse center l^t
     - else: Make prediction/classification based on l^τ
- How to encode l^t?

[1] A *glimpse* is here defined as a crop of the image

Image classification with RNN and location-based attention

Pseudoalgorithm:

- Start with center/random glimpse[1] with center l^0
- For $t = 1, \ldots, \tau$
  1. Extract glimpse with center at l^{t-1}.
  2. Extract features for location, e.g. with convnet
  3. Update state of RNN
  4. - if $t < \tau$: predict next glimpse center l^t
     - else: Make prediction/classification based on l^τ
- How to encode l^t?
- Glimpse policy trained with reinforcement learning (policy gradient)!

[1] A *glimpse* is here defined as a crop of the image

Image classification with RNN and location-based attention

Pseudoalgorithm:

- Start with center/random glimpse[1] with center l^0
- For $t = 1, \ldots, \tau$
  1. Extract glimpse with center at l^{t-1}.
  2. Extract features for location, e.g. with convnet
  3. Update state of RNN
  4. - if $t < \tau$: predict next glimpse center l^t
     - else: Make prediction/classification based on l^τ
- How to encode l^t?
- Glimpse policy trained with reinforcement learning (policy gradient)!

Usually extract some lower resolution crops as well.

[1] A *glimpse* is here defined as a crop of the image