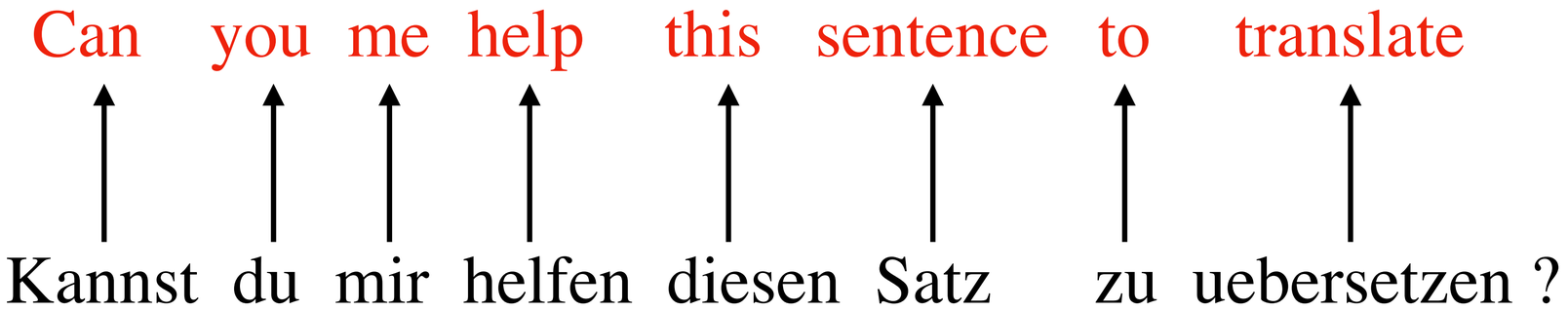
8.4

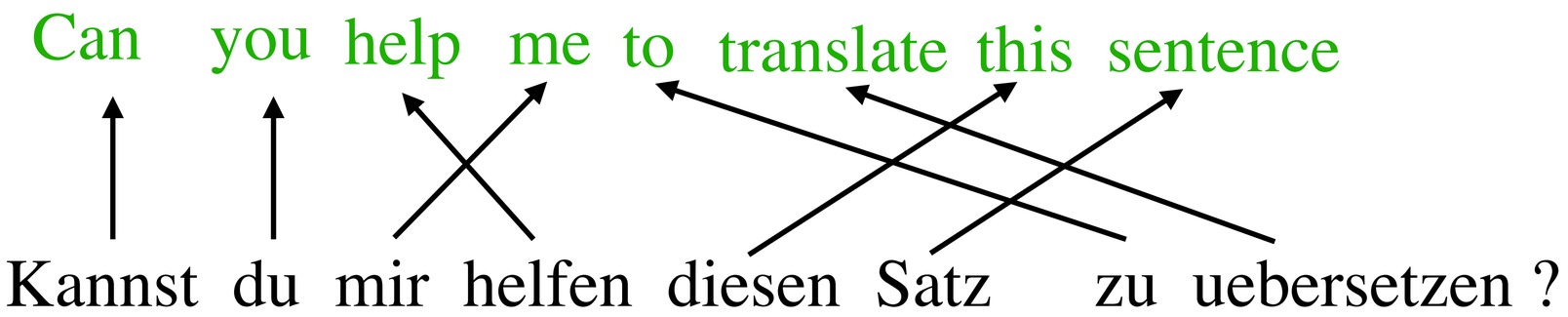
The Transformer Architecture

Part 3: Paying Attention to Different Parts of the Input Sequence

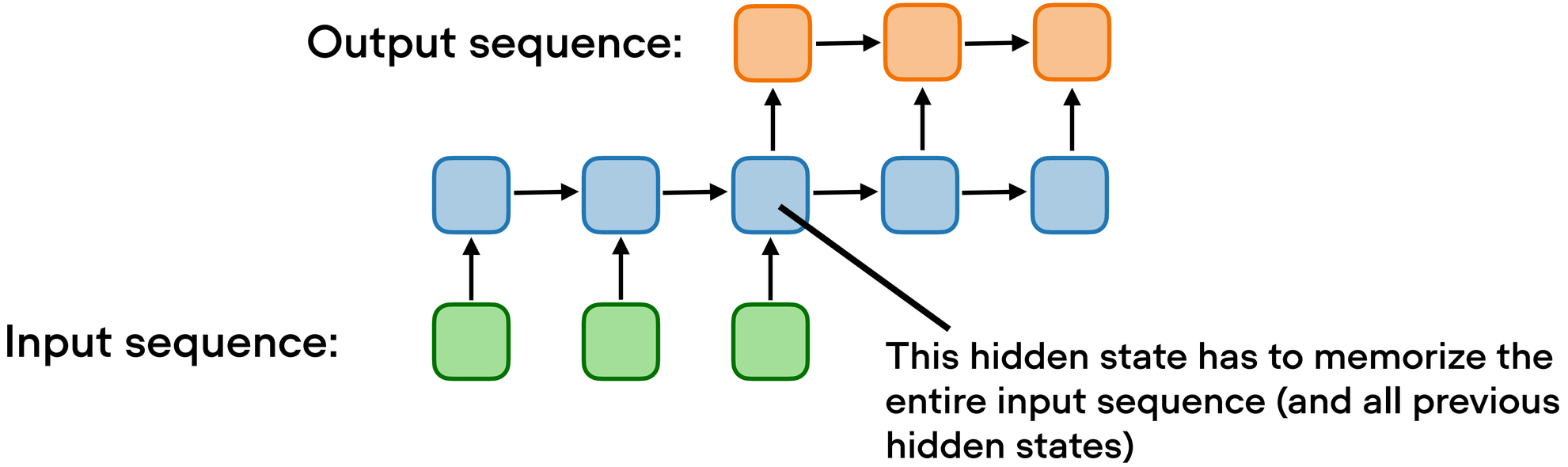
Sebastian Raschka and the Lightning AI Team

We can't translate sentences word by word!





RNN for Seq2Seq tasks (e.g., language translation)



The approach does not work well for longer sequences.

Attention was developed to address that!

Computer Science > Computation and Language

[Submitted on 1 Sep 2014 (v1), last revised 19 May 2016 (this version, v7)]

Neural Machine Translation by Jointly Learning to Align and Translate

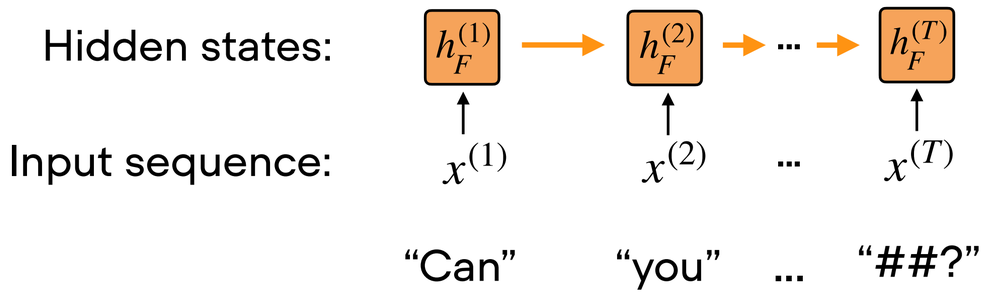
Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio

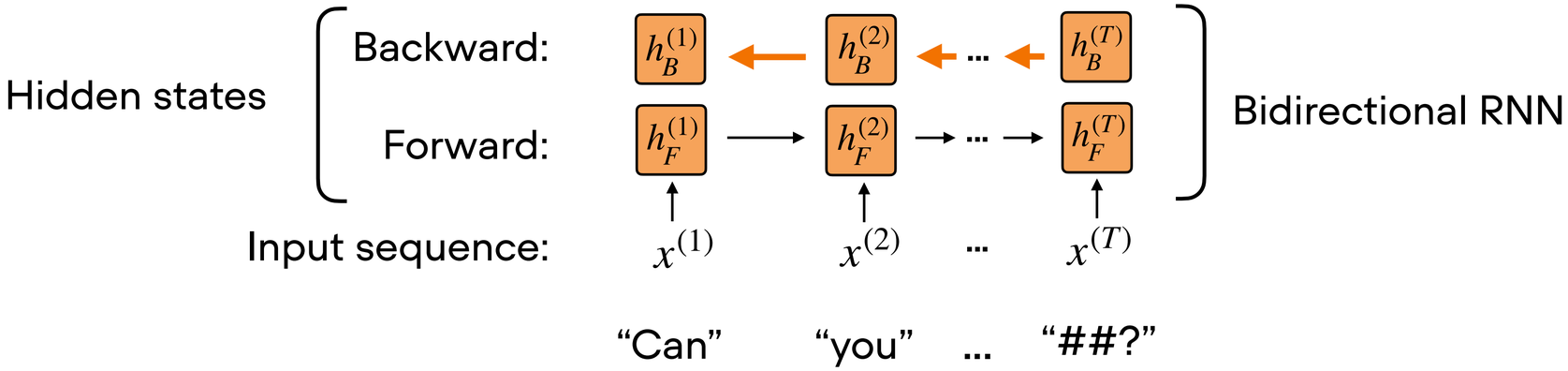
Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder-decoders and consists of an encoder that encodes a source sentence into a fixed-length vector from which a decoder generates a translation. In this paper, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder-decoder architecture, and propose to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly. With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of English-to-French translation. Furthermore, qualitative analysis reveals that the (soft-)alignments found by the model agree well with our intuition.

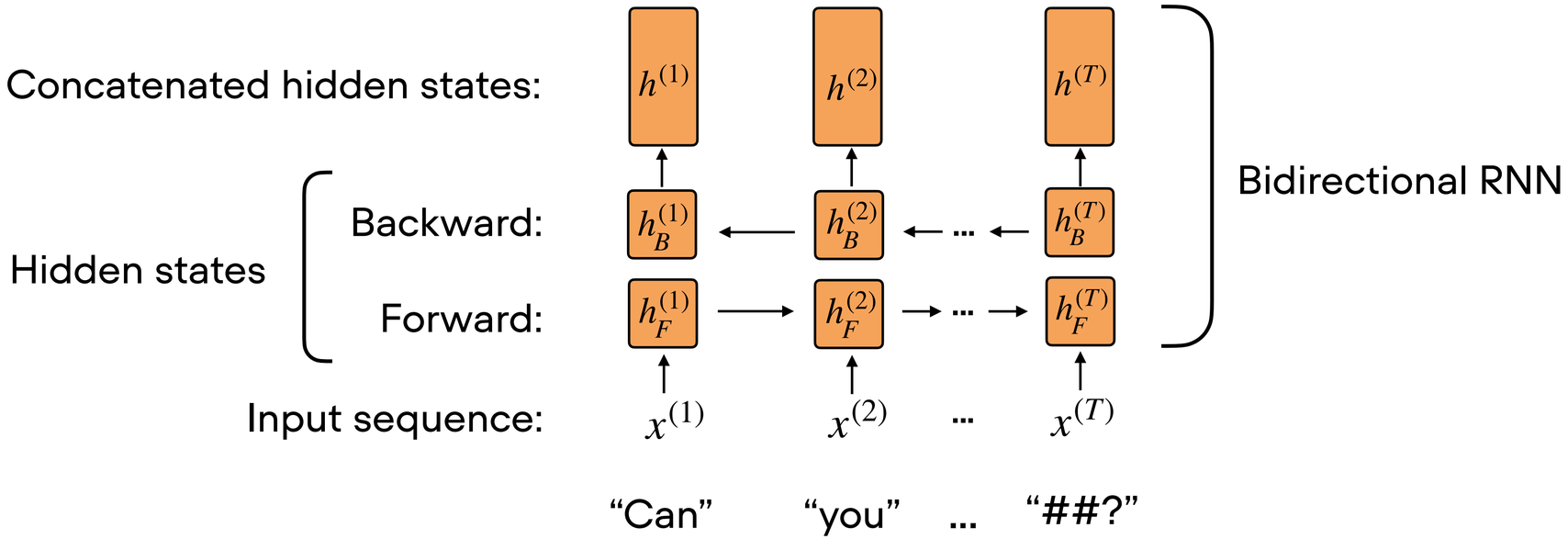
Comments: Accepted at ICLR 2015 as oral presentation

Idea: create context vectors that contain information about the whole sequence

Use attention scores to weigh the importance of each word at the current step

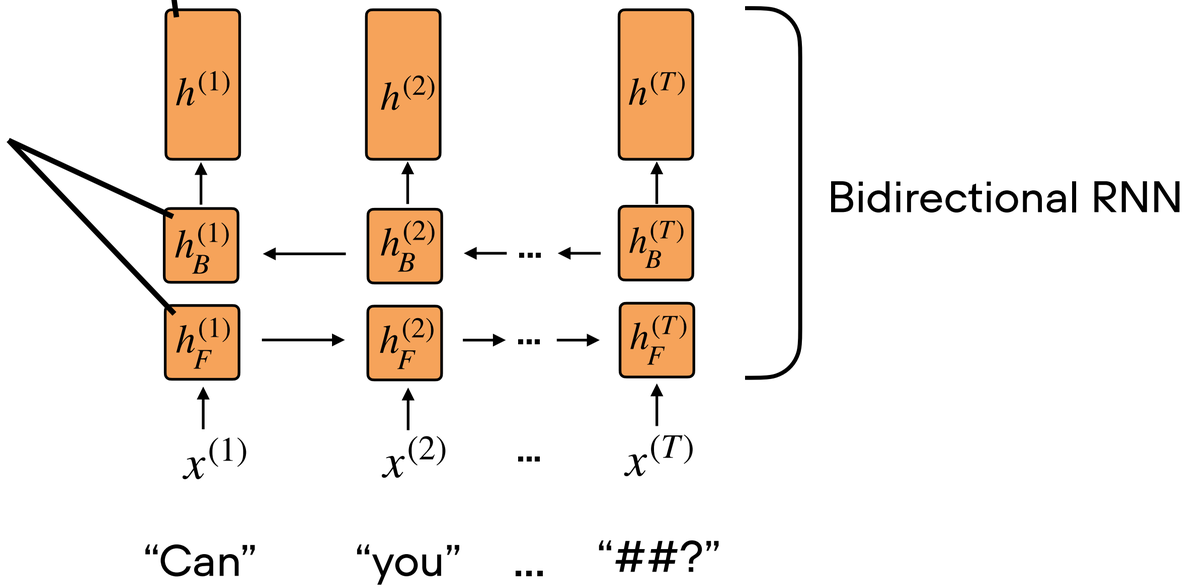

Hidden states:
$h_F^{(1)}$
$h_F^{(2)}$
...
$h_F^{(T)}$
Input sequence:
$x^{(1)}$
$x^{(2)}$
...
$x^{(T)}$
“Can”
“you”
...
“##?”

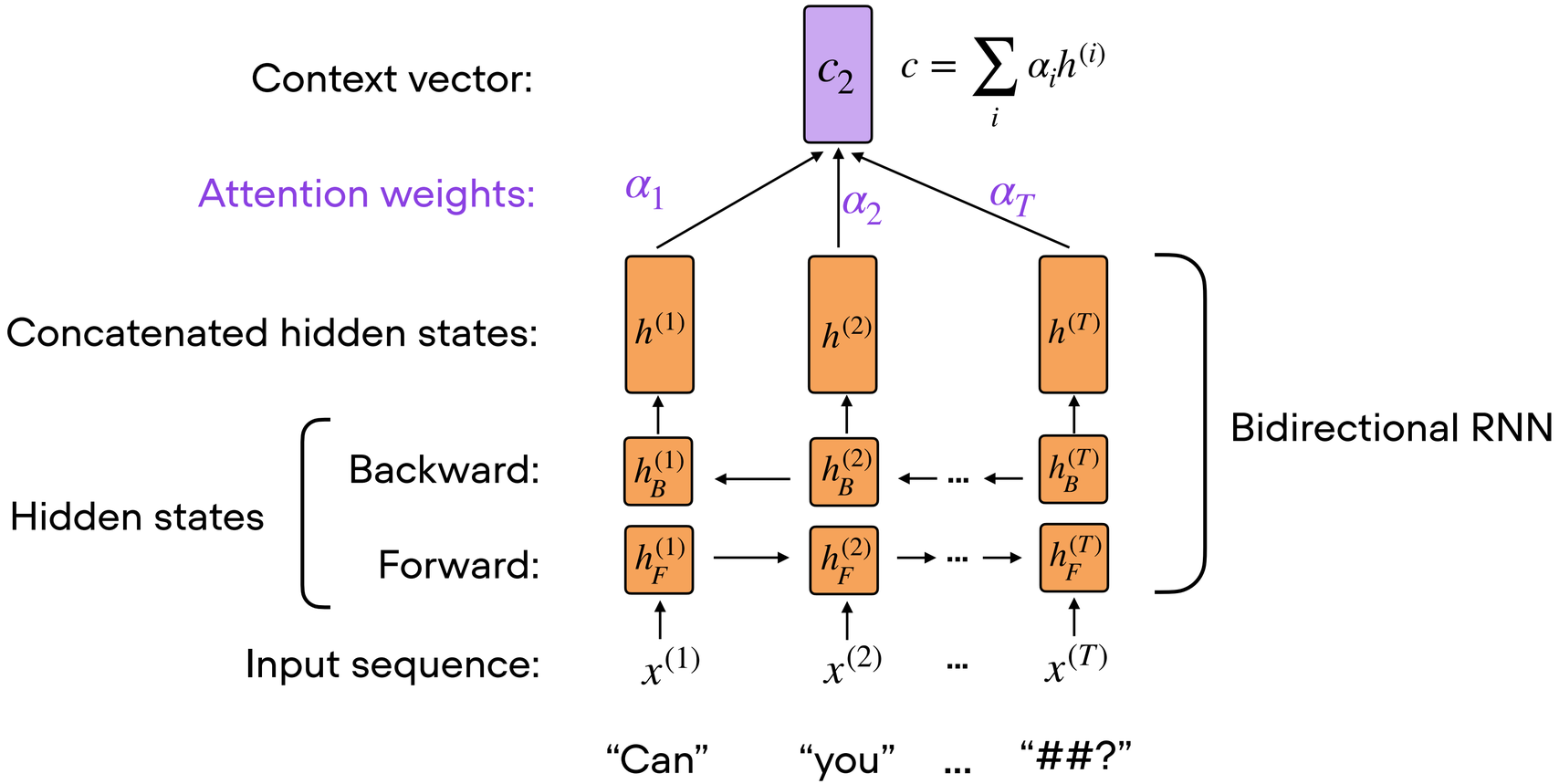

Hidden states
Backward:
Forward:
$h_B^{(1)}$
$h_B^{(2)}$
...
$h_B^{(T)}$
$h_F^{(1)}$
$h_F^{(2)}$
...
$h_F^{(T)}$
Bidirectional RNN
Input sequence:
$x^{(1)}$
$x^{(2)}$
...
$x^{(T)}$
“Can”
“you”
...
“##?”

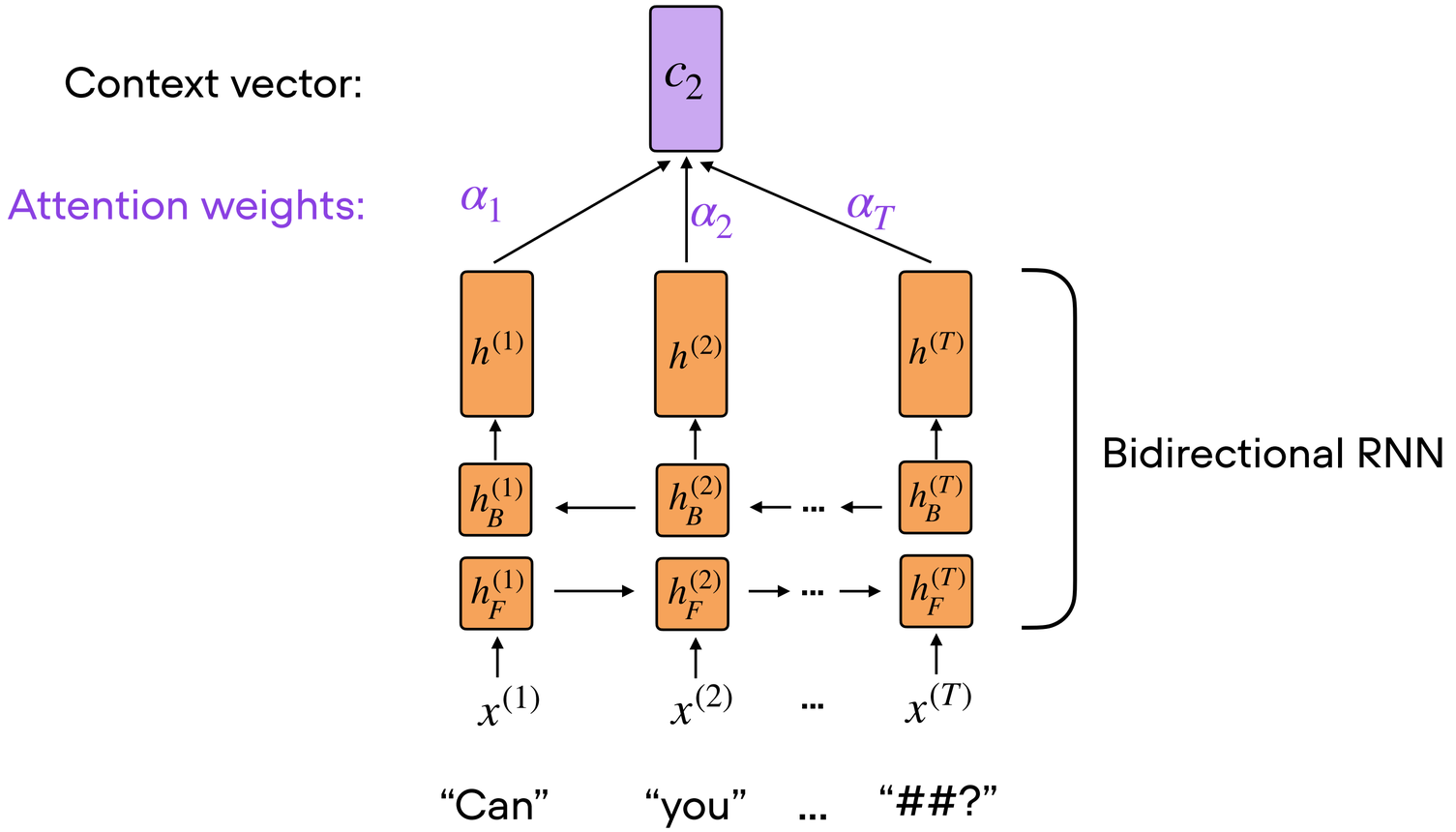

Concatenated hidden states:
Hidden states
Backward:
Forward:
Input sequence:
$h^{(1)}$
$h^{(2)}$
$h^{(T)}$
$h_B^{(1)}$
$h_B^{(2)}$
$h_B^{(T)}$
$h_F^{(1)}$
$h_F^{(2)}$
$h_F^{(T)}$
$x^{(1)}$
$x^{(2)}$
$x^{(T)}$
...
Bidirectional RNN
“Can”
“you”
...
“##?”

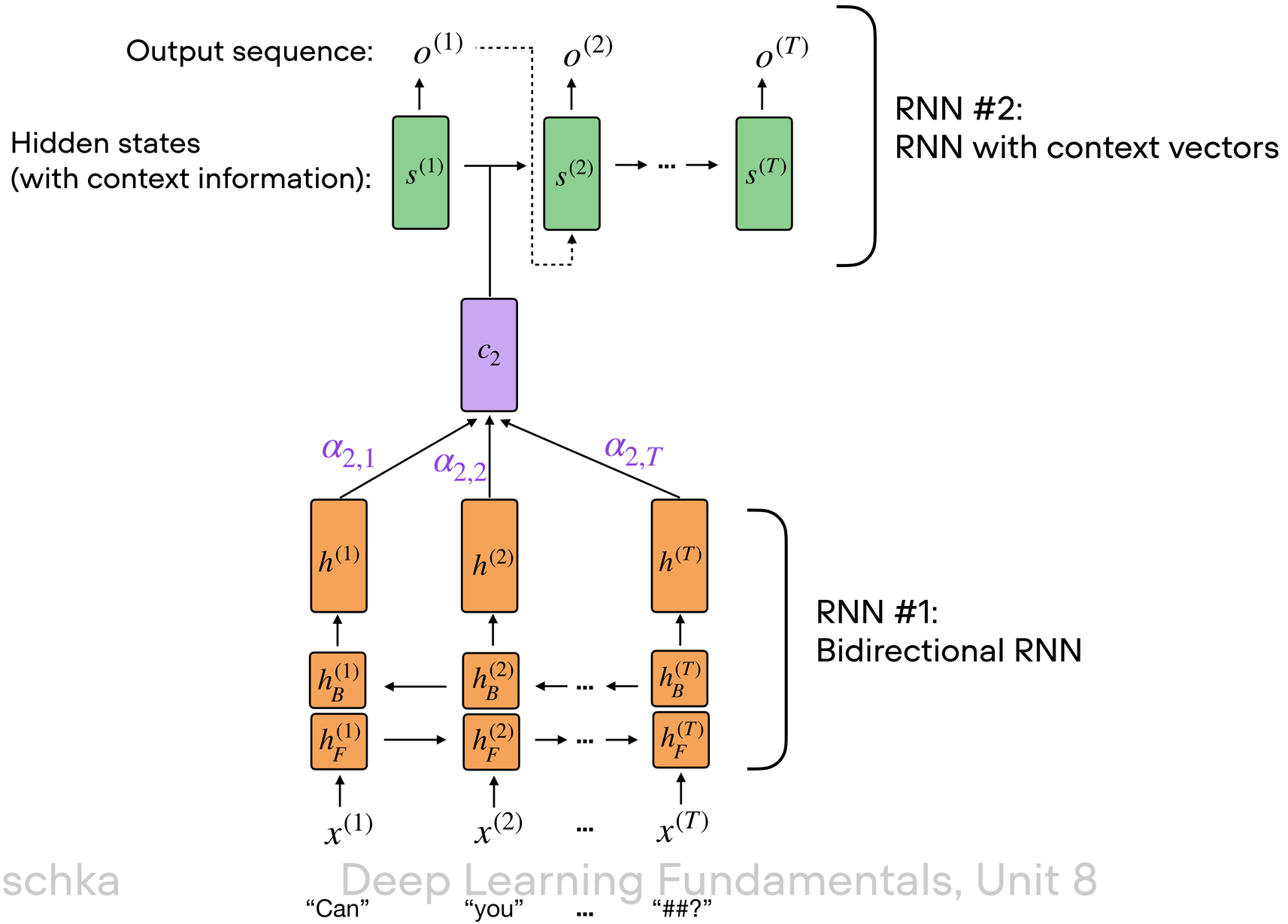
Then the concatenated hidden state is a 256-dimensional vector

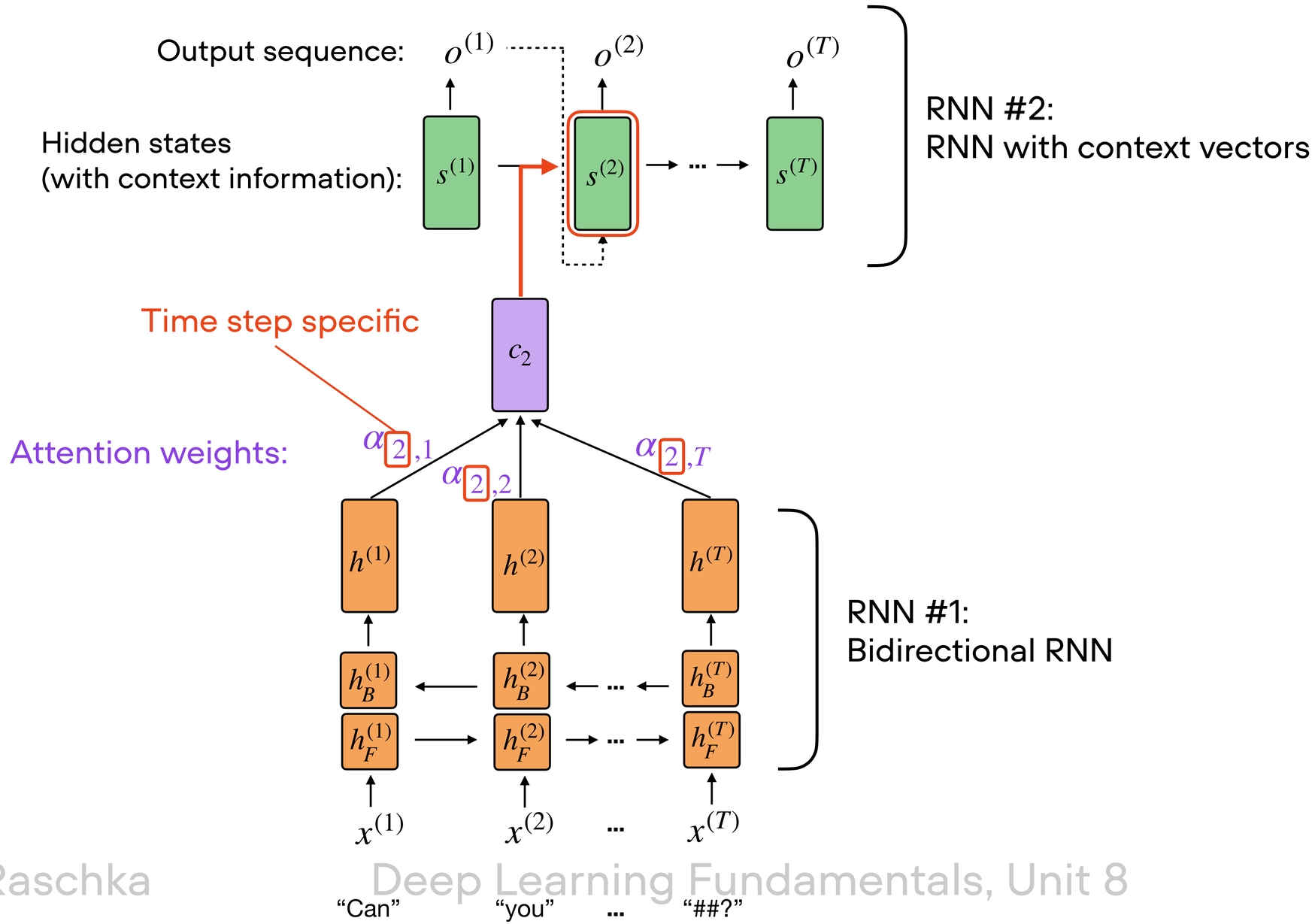
For example, suppose each hidden state is a 128-dimensional vector




Context vector:
c_2
$c = \sum_i \alpha_i h^{(i)}$
Attention weights:
α_1
α_2
α_T
Concatenated hidden states:
$h^{(1)}$
$h^{(2)}$
$h^{(T)}$
Bidirectional RNN
Hidden states
Backward:
$h_B^{(1)}$
$h_B^{(2)}$
...
$h_B^{(T)}$
Forward:
$h_F^{(1)}$
$h_F^{(2)}$
...
$h_F^{(T)}$
Input sequence:
$x^{(1)}$
$x^{(2)}$
...
$x^{(T)}$
“Can”
“you”
...
“##?”


Context vector:
Attention weights:
c_2
α_1
α_2
α_T
$h^{(1)}$
$h^{(2)}$
$h^{(T)}$
$h_B^{(1)}$
$h_B^{(2)}$
$h_B^{(T)}$
$h_F^{(1)}$
$h_F^{(2)}$
$h_F^{(T)}$
Bidirectional RNN
$x^{(1)}$
$x^{(2)}$
$x^{(T)}$
"Can"
"you"
...
"##?"


Output sequence:
$o^{(1)}$
$o^{(2)}$
$o^{(T)}$
Hidden states
(with context information):
$s^{(1)}$
$s^{(2)}$
...
$s^{(T)}$
RNN #2:
RNN with context vectors
c_2
$\alpha_{2,1}$
$\alpha_{2,2}$
$\alpha_{2,T}$
$h^{(1)}$
$h^{(2)}$
$h^{(T)}$
$h_B^{(1)}$
$h_B^{(2)}$
$h_B^{(T)}$
$h_F^{(1)}$
$h_F^{(2)}$
$h_F^{(T)}$
RNN #1:
Bidirectional RNN
$x^{(1)}$
$x^{(2)}$
$x^{(T)}$
"Can"
"you"
"##?"


Output sequence:
$o^{(1)}$
$o^{(2)}$
$o^{(T)}$
Hidden states
(with context information):
$s^{(1)}$
$s^{(2)}$
...
$s^{(T)}$
RNN #2:
RNN with context vectors
Time step specific
c_2
Attention weights:
$\alpha_{2,1}$
$\alpha_{2,2}$
$\alpha_{2,T}$
$h^{(1)}$
$h^{(2)}$
$h^{(T)}$
$h_B^{(1)}$
$h_B^{(2)}$
$h_B^{(T)}$
$h_F^{(1)}$
$h_F^{(2)}$
$h_F^{(T)}$
RNN #1:
Bidirectional RNN
$x^{(1)}$
$x^{(2)}$
$x^{(T)}$
"Can"
"you"
"##?"

The focus is not always on the current input word

Cornell University

We gratefully acknowledge support from the Simons Foundation and member institutions.

arXiv > cs > arXiv:1409.0473

Search... All fields Search

Help | Advanced Search

Computer Science > Computation and Language

[Submitted on 1 Sep 2014 (v1), last revised 19 May 2016 (this version, v7)]

Neural Machine Translation by Jointly Learning to Align and Translate

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio

Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder–decoders and consists of an encoder that encodes a source sentence into a fixed-length vector from which a decoder generates a translation. In this paper, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder–decoder architecture, and propose to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly. With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of English-to-French translation. Furthermore, qualitative analysis reveals that the (soft-)alignments found by the model agree well with our intuition.

Comments: Accepted at ICLR 2015 as oral presentation

Download:

- PDF
- Other formats

(license)

Current browse context: cs.CL

< prev | next >

new | recent | 1409

Change to browse by:

cs

cs.LG

cs.NE

stat

stat.ML

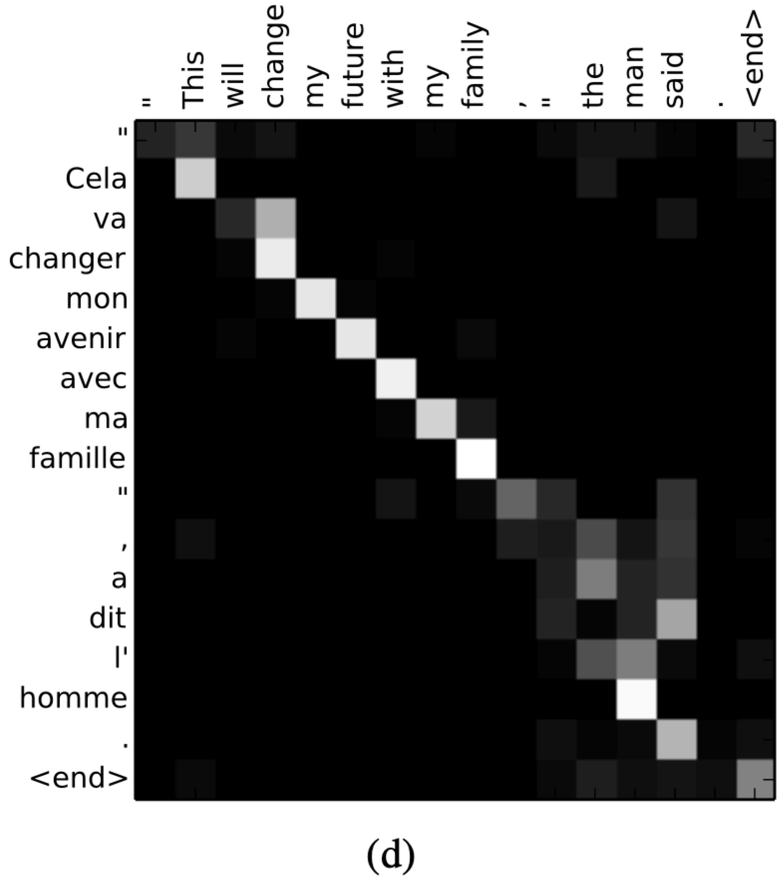
References & Citations

- NASA ADS
- Google Scholar
- Semantic Scholar

28 blog links (what is this?)

DBLP - CS Bibliography

listing | bibtex



(d)

Back to transformers

Next: Computing the attention weights