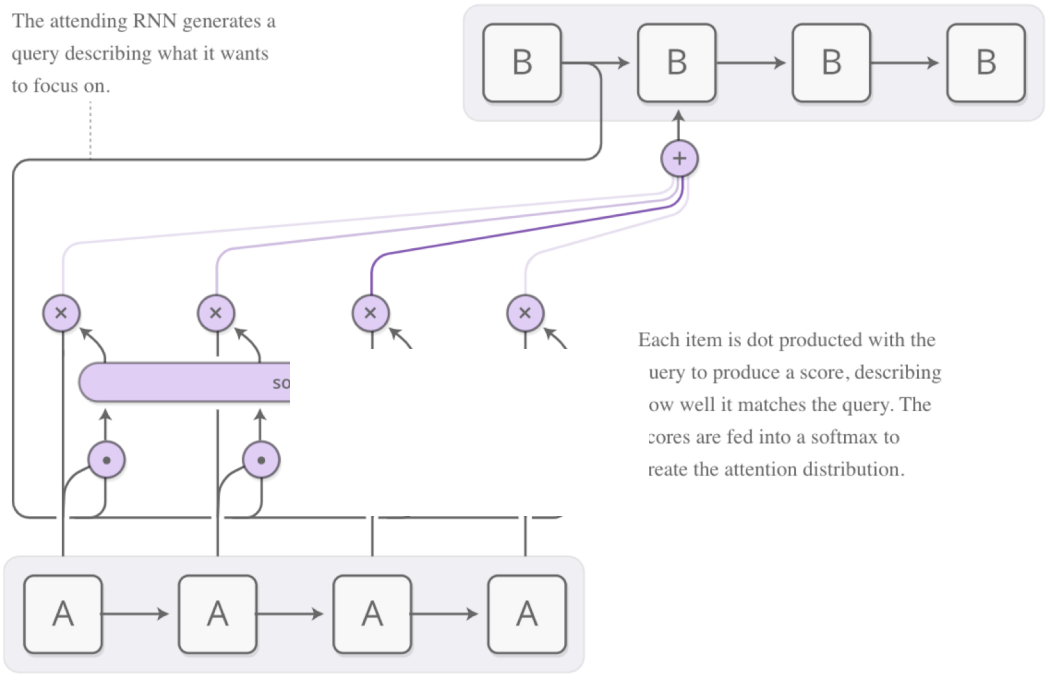
# Sequence-to-sequence models with attention

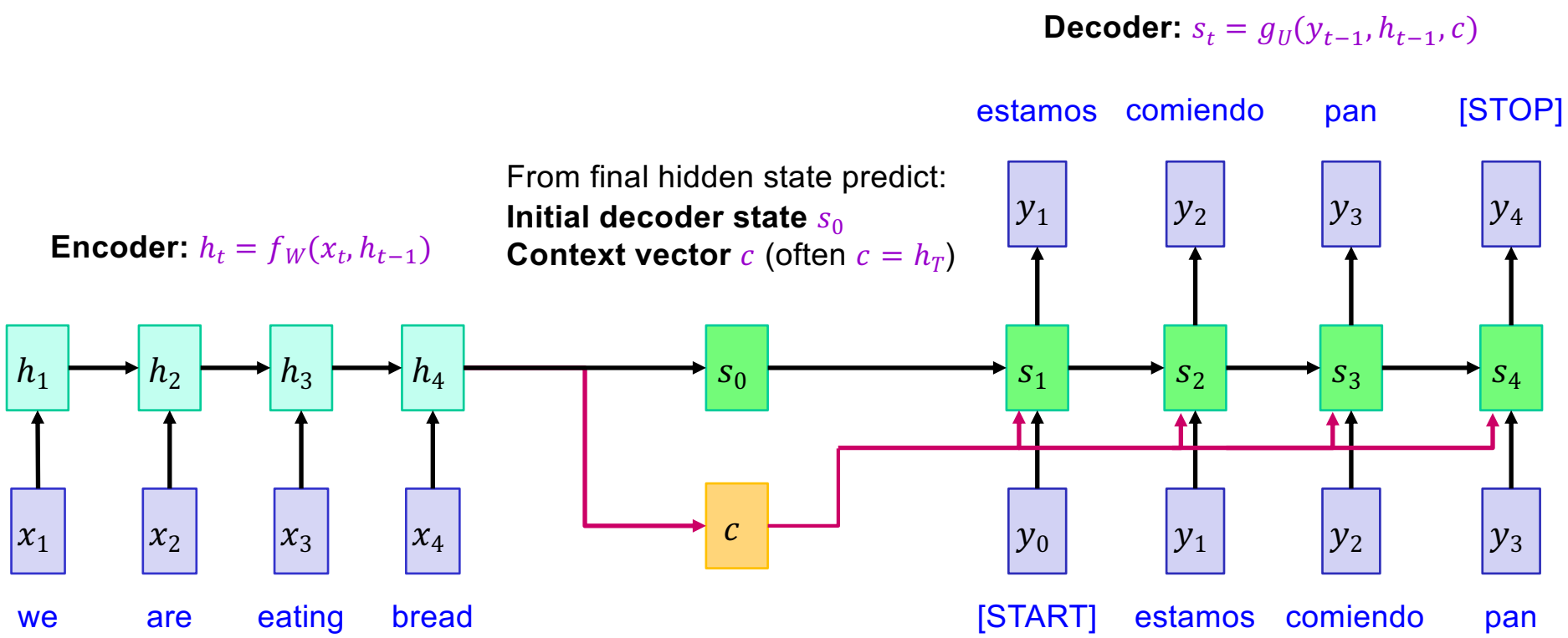



Many slides adapted from J. Johnson

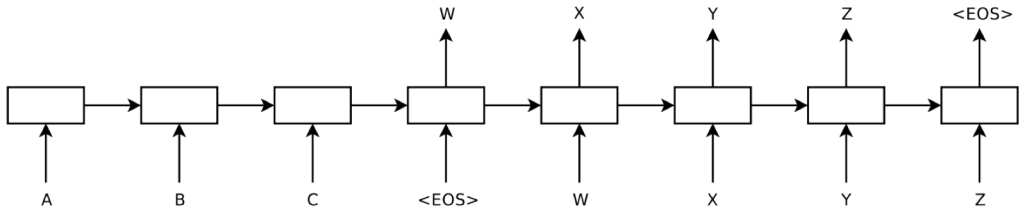
# Outline

- Vanilla seq2seq with RNNs
- Seq2seq with RNNs and attention
- Image captioning with attention
- Convolutional seq2seq with attention

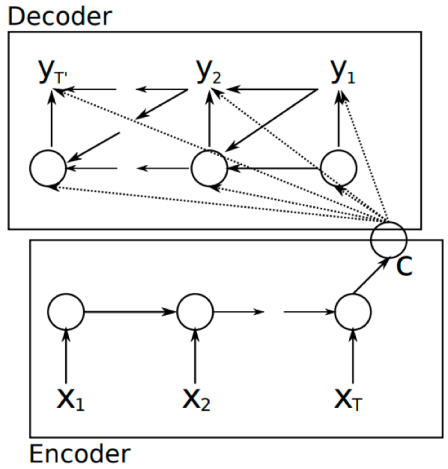
# Sequence-to-sequence with RNNs



# Sequence-to-sequence with RNNs

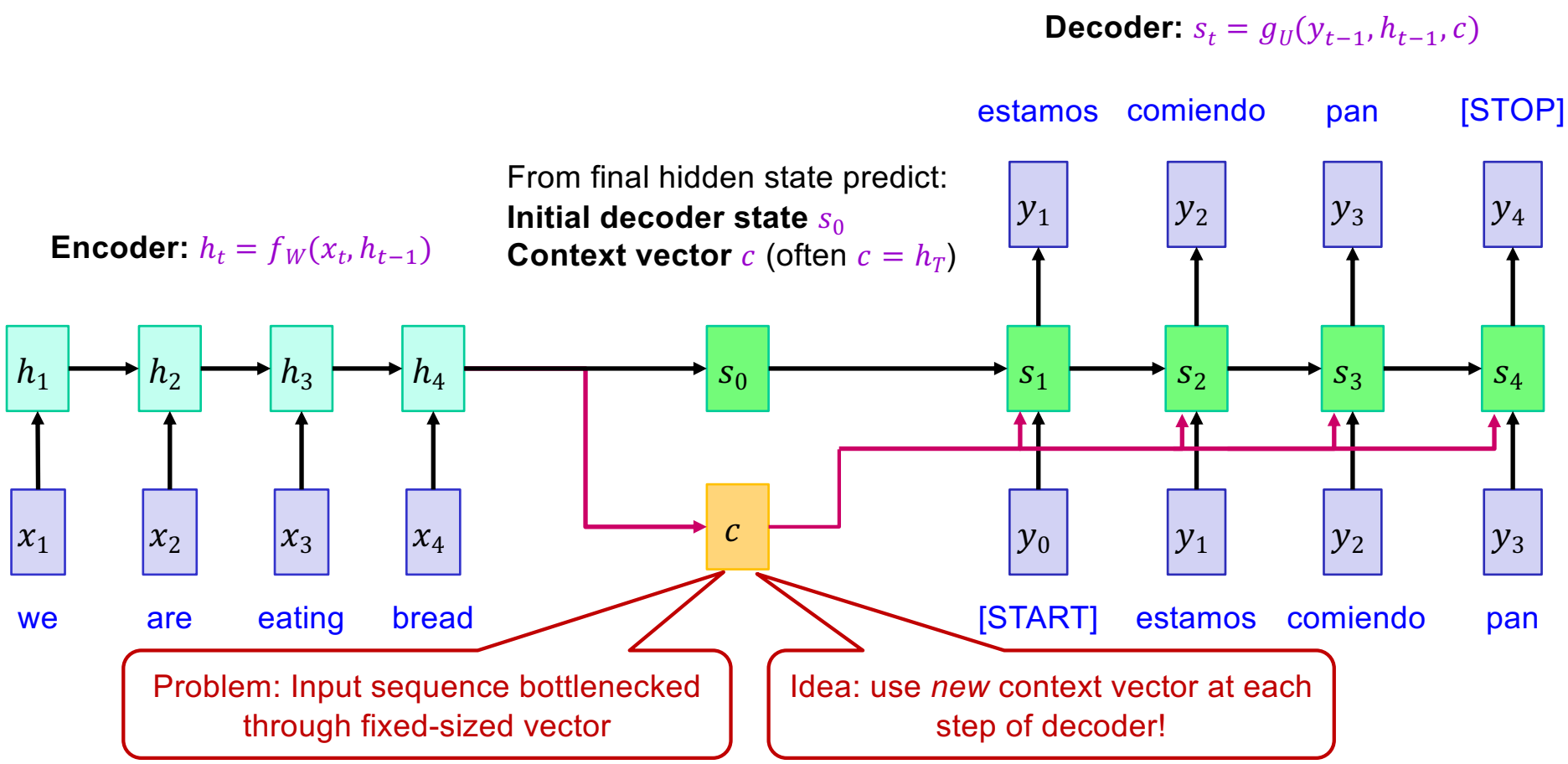



I. Sutskever, O. Vinyals, Q. Le, Sequence to Sequence Learning with Neural Networks, NeurIPS 2014




K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, ACL 2014

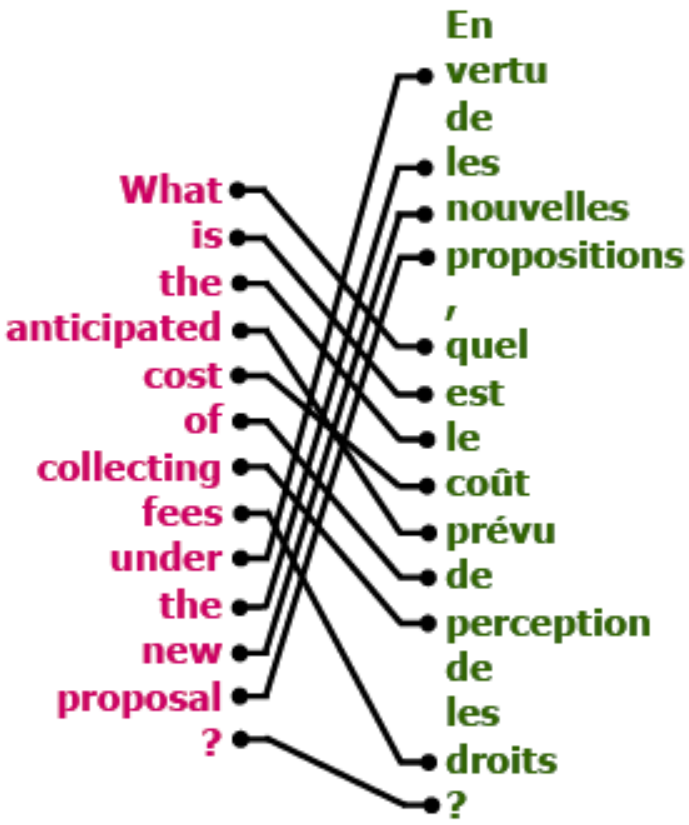
# Sequence-to-sequence with RNNs




A. Sutskever, O. Vinyals, Q. Le, Sequence to sequence learning with neural networks, NeurIPS 2014

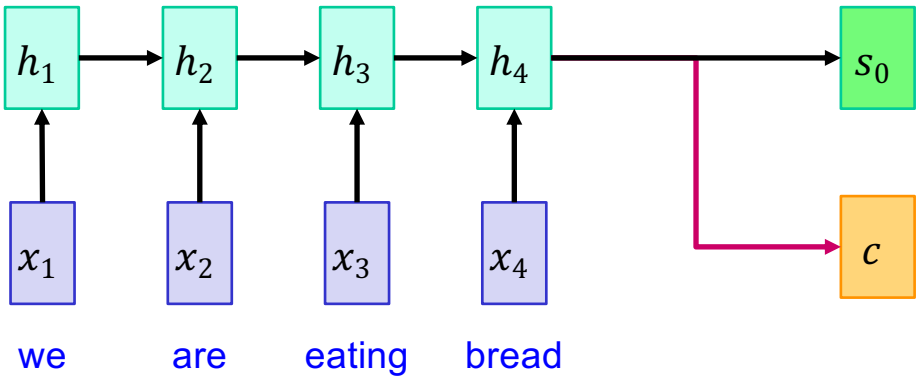
# Sequence-to-sequence with RNNs *and attention*

- Intuition: translation requires *alignment*

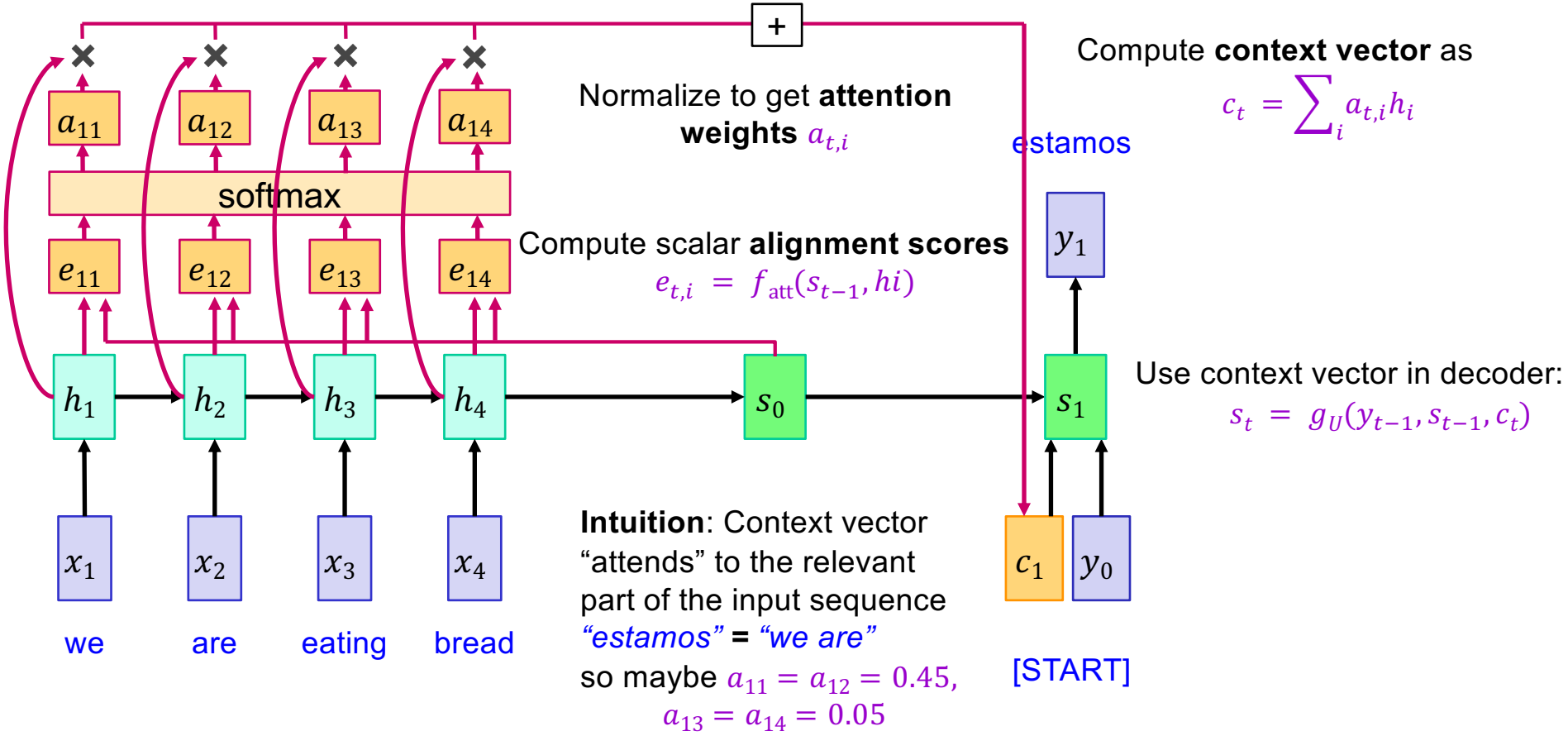


# Sequence-to-sequence with RNNs and attention

- At each timestep of decoder, context vector “looks at” different parts of the input sequence

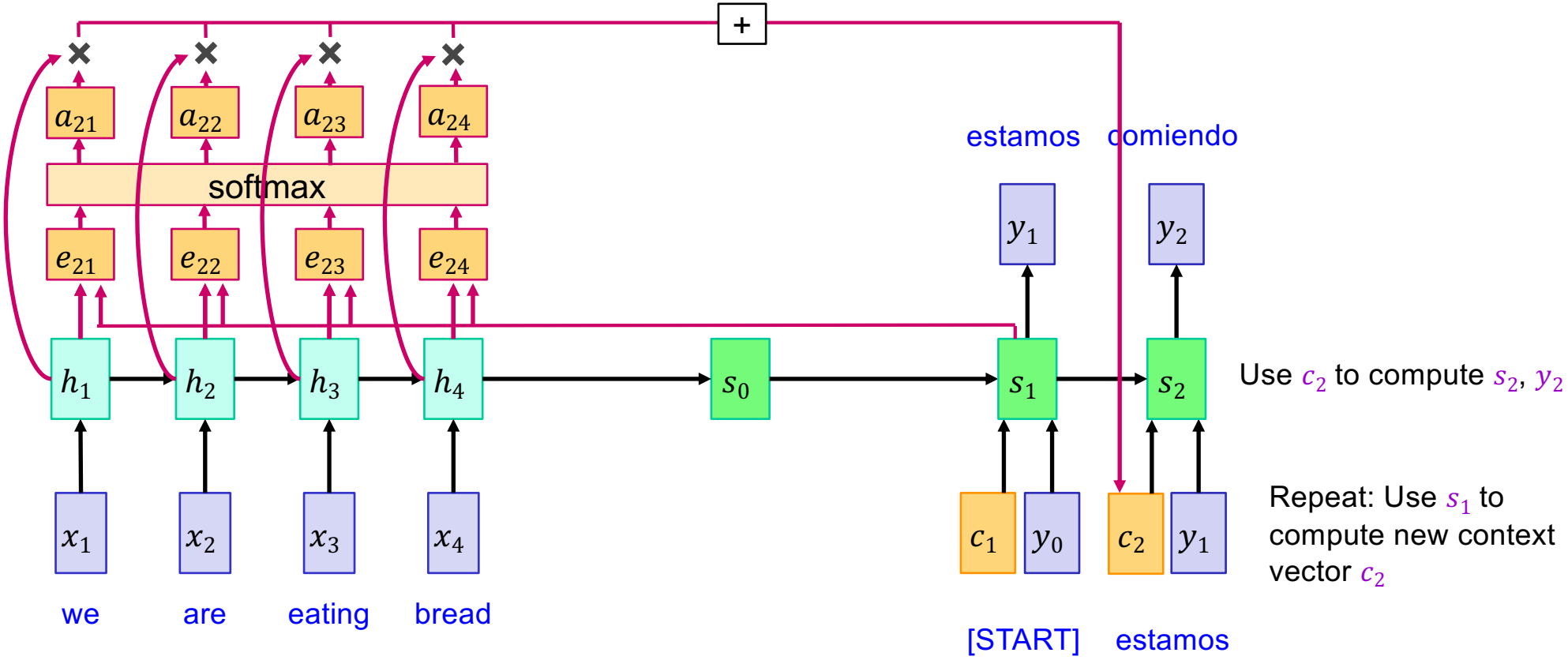


# Sequence-to-sequence with RNNs and attention

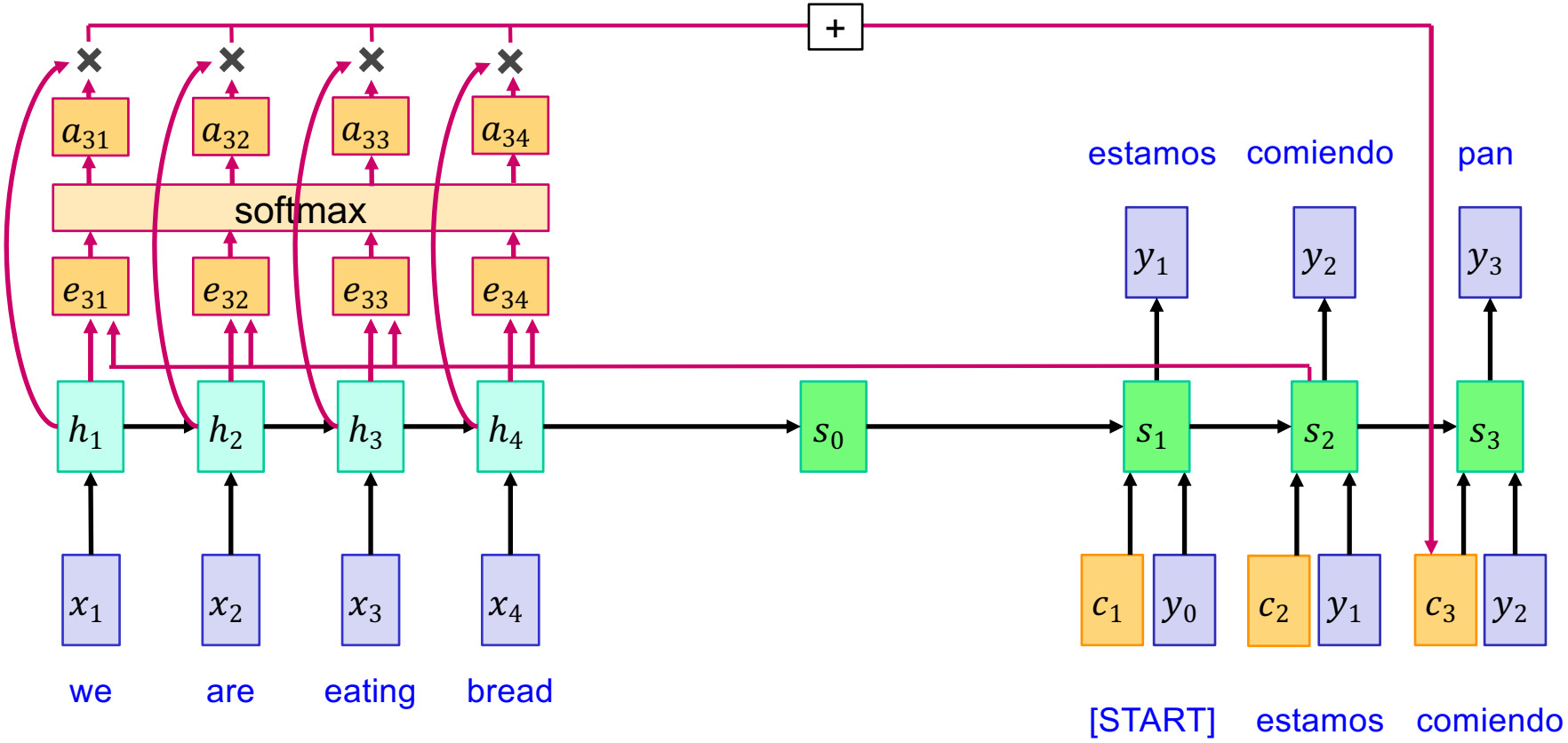



D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

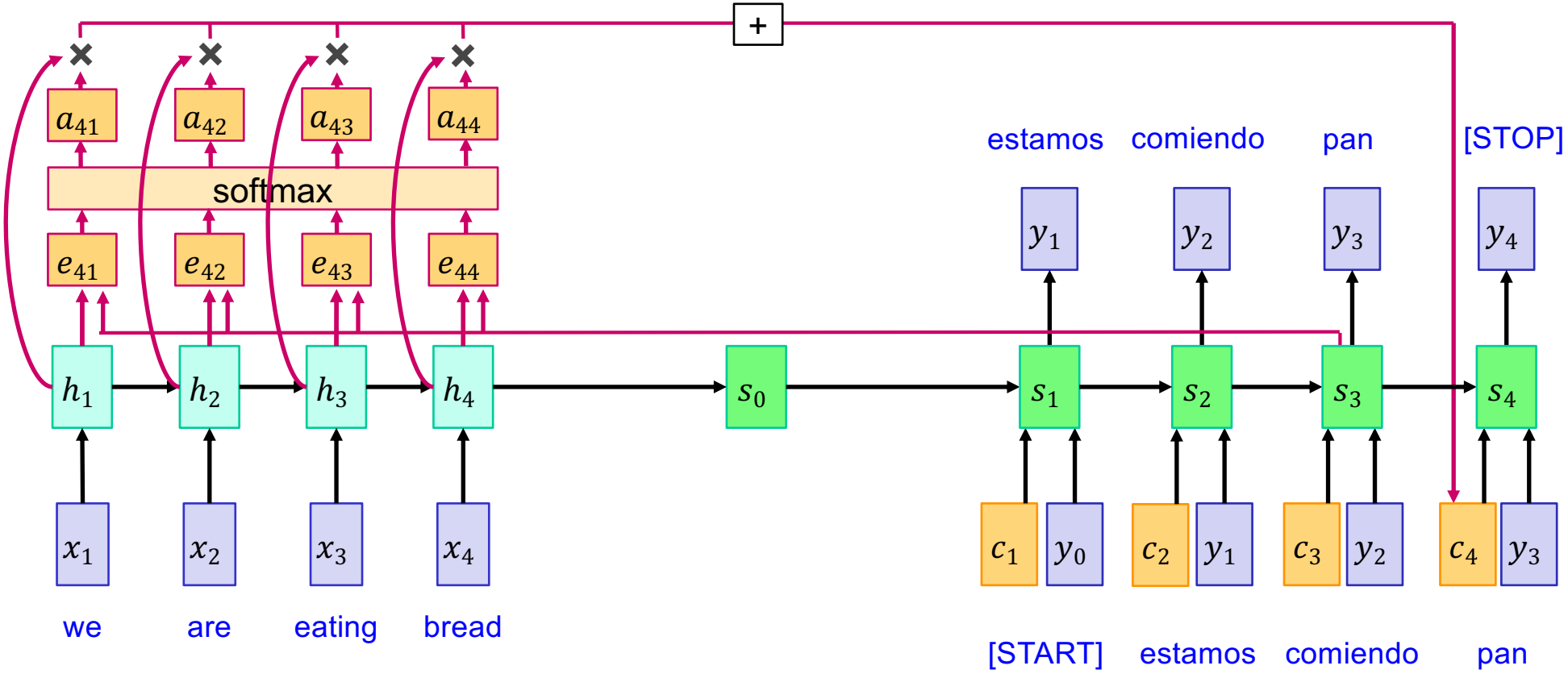
# Sequence-to-sequence with RNNs and attention



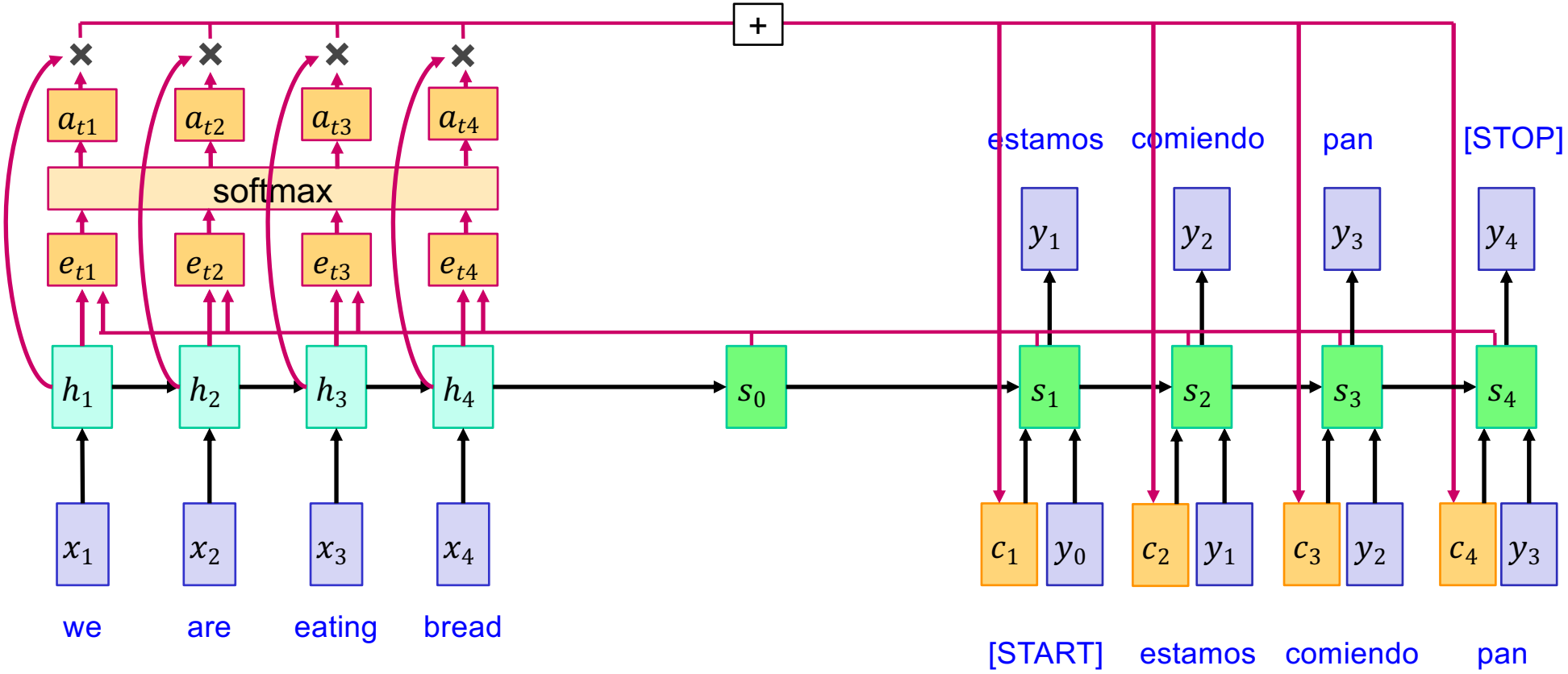
# Sequence-to-sequence with RNNs and attention



# Sequence-to-sequence with RNNs and attention

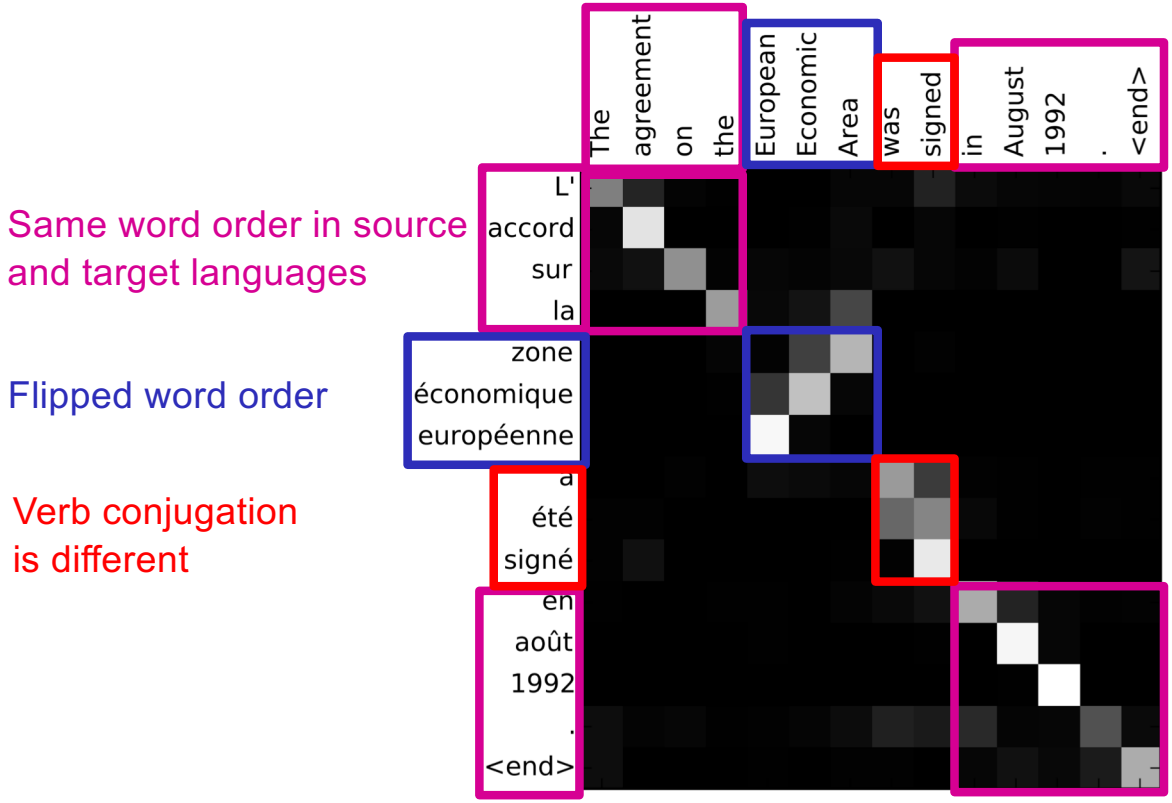


# Sequence-to-sequence with RNNs and attention



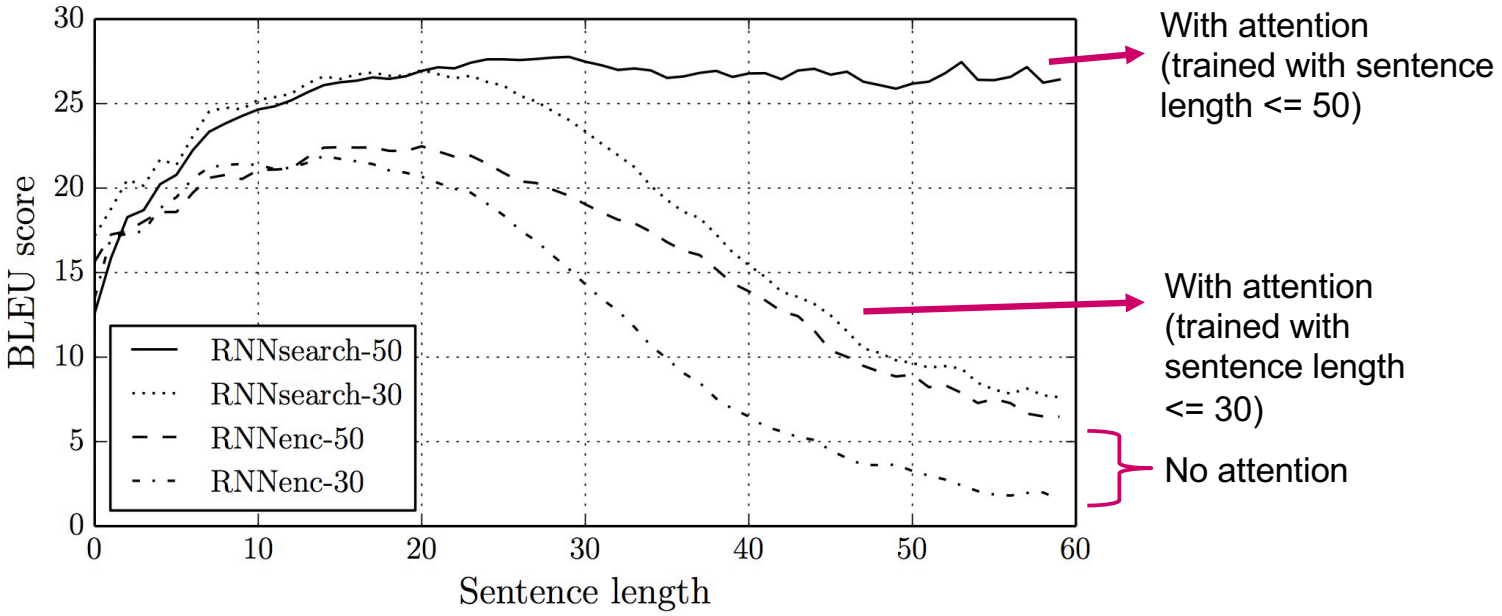
# Sequence-to-sequence with RNNs and attention

- Visualizing attention weights:




D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

# Quantitative evaluation




D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

# Google Neural Machine Translation (GNMT)

## Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation

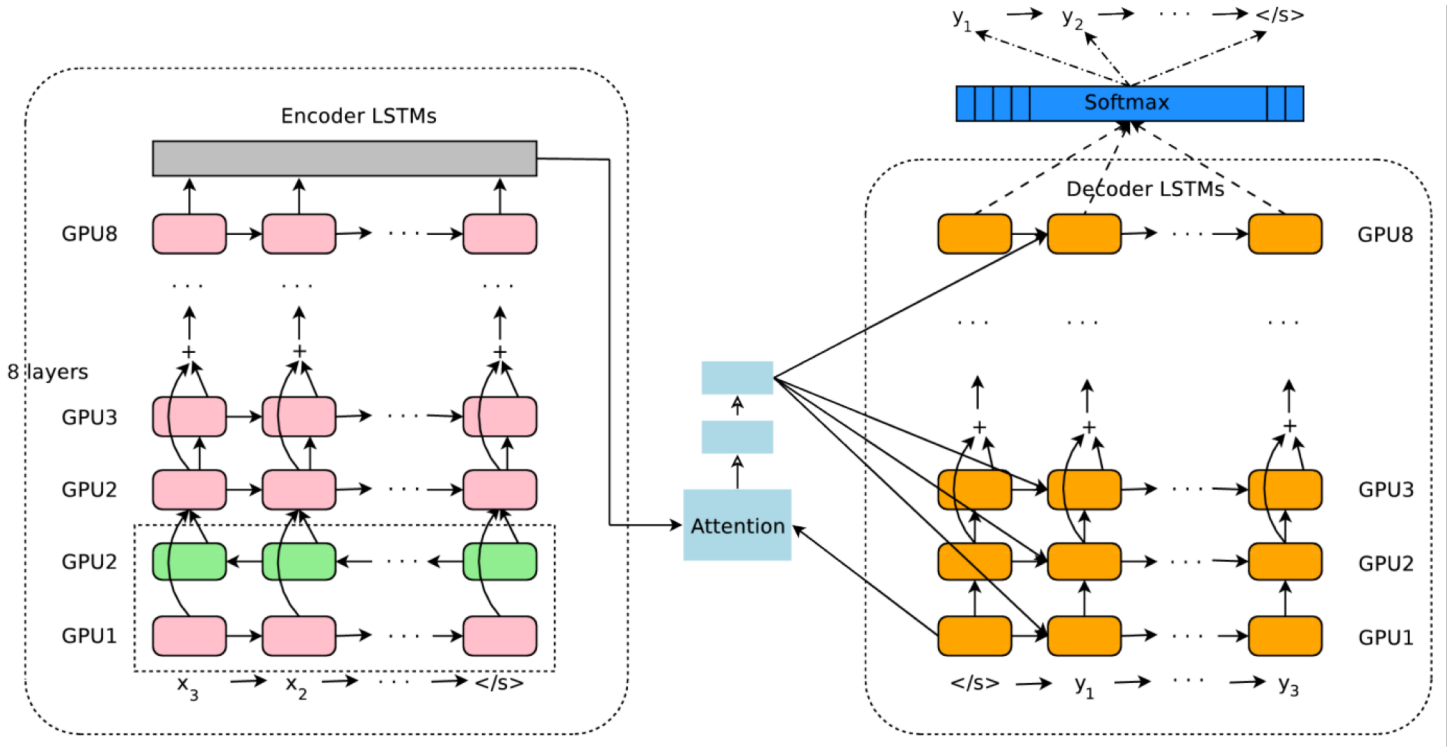
Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi
yonghui,schuster,zhifengc,qvl,mnorouzi@google.com

Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey,
Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser,
Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens,
George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa,
Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean

Y. Wu et al., Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, arXiv 2016

https://www.nytimes.com/2016/12/14/magazine/the-great-ai-awakening.html

# Google Neural Machine Translation (GNMT)




Y. Wu et al., [Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation](), arXiv 2016

# Google Neural Machine Translation (GNMT)

- **Standard training objective:** maximize log-likelihood of ground truth output given input:

$$\sum_i \log P_W(Y_i^* | X_i)$$

  - Not related to task-specific reward function (e.g., BLEU score)
  - Does not encourage "better" incorrect sentences to get better likelihood

- **Refinement objective:** expectation of rewards over possible predicted sentences $Y$:

$$\sum_i \sum_Y P_W(Y | X_i)\, R(Y, Y_i^*)$$

  - Use variant of BLEU score to compute reward
  - Reward is not differentiable -- need *reinforcement learning* to train (initialize with ML-trained model)

# Google Neural Machine Translation (GNMT)

- Human evaluation results on production data (500 randomly sampled sentences from Wikipedia and news websites)

Table 10: Mean of side-by-side scores on production data

| | PBMT | GNMT | Human | Relative Improvement |
|---|---|---|---|---|
| English → Spanish | 4.885 | 5.428 | 5.550 | 87% |
| English → French | 4.932 | 5.295 | 5.496 | 64% |
| English → Chinese | 4.035 | 4.594 | 4.987 | 58% |
| Spanish → English | 4.872 | 5.187 | 5.372 | 63% |
| French → English | 5.046 | 5.343 | 5.404 | 83% |
| Chinese → English | 3.694 | 4.263 | 4.636 | 60% |

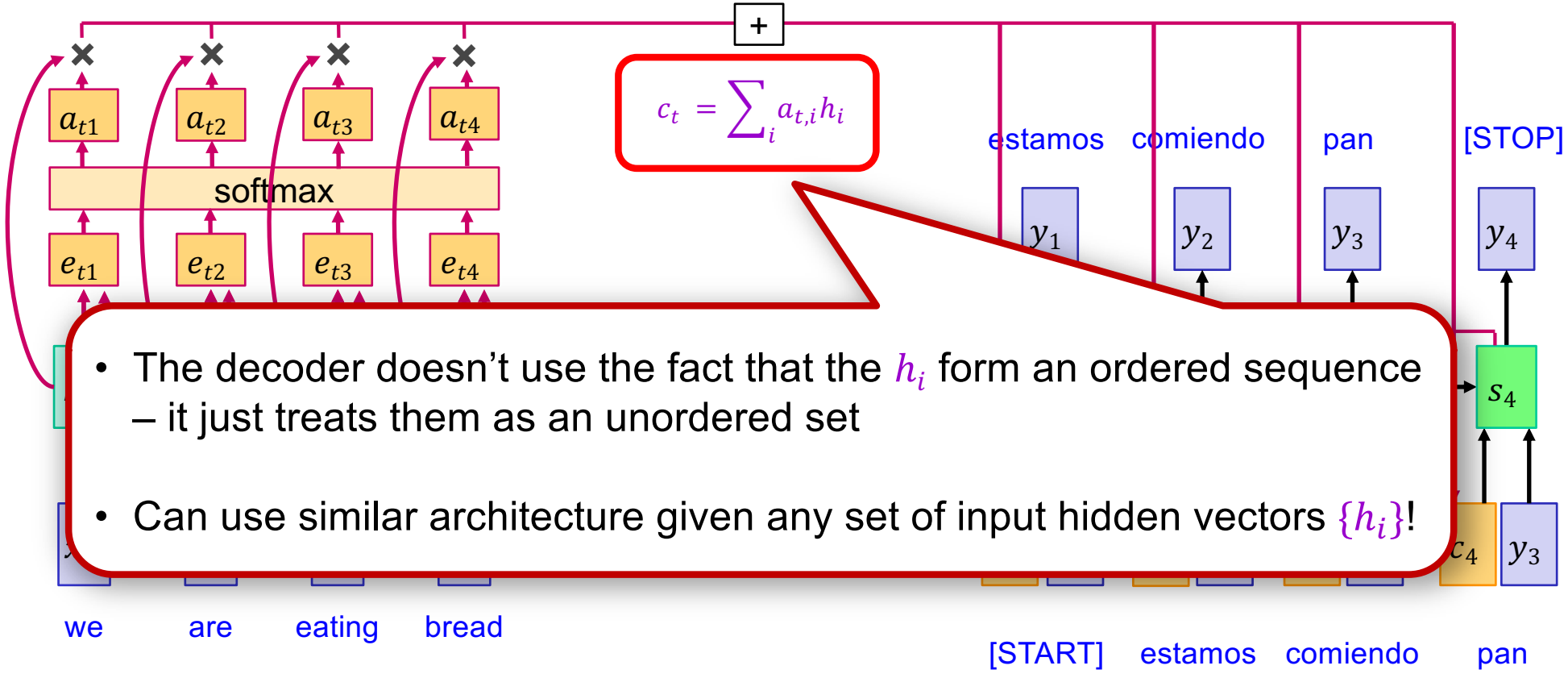
**Side-by-side scores:** range from 0 ("completely nonsense translation") to 6 ("perfect translation"), produced by human raters fluent in both languages

**PBMT:** Translation by phrase-based statistical translation system used by Google
**GNMT:** Translation by GNMT system
**Human:** Translation by humans fluent in both languages

# Outline

- Vanilla seq2seq with RNNs
- Seq2seq with RNNs and attention
- Image captioning with attention

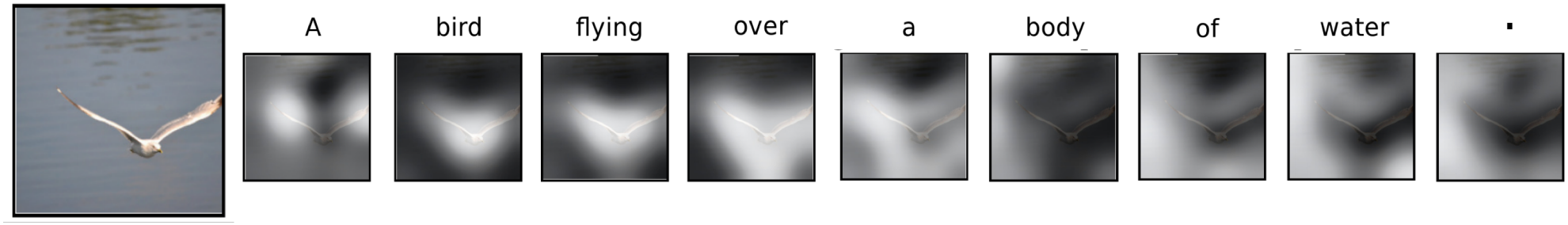
# Generalizing attention




- The decoder doesn't use the fact that the $h_i$ form an ordered sequence – it just treats them as an unordered set
- Can use similar architecture given any set of input hidden vectors $\{h_i\}$!

# Image captioning with RNNs and attention

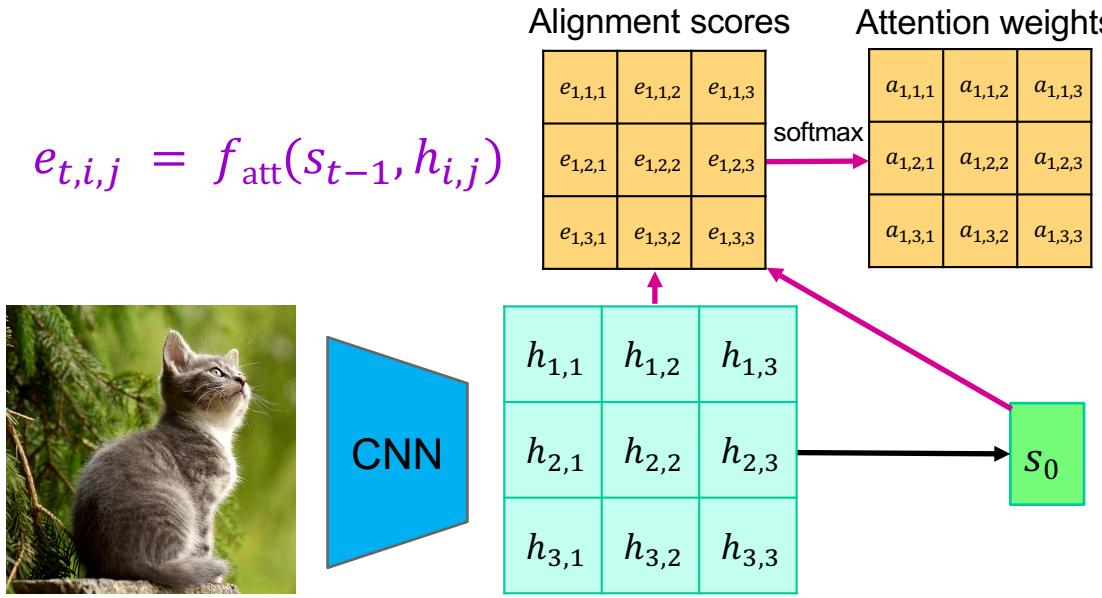
- Idea: pay attention to different parts of the image when generating different words
- Automatically learn this grounding of words to image regions without direct supervision




K. Xu et al., Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, ICML 2015

# Image captioning with RNNs and attention

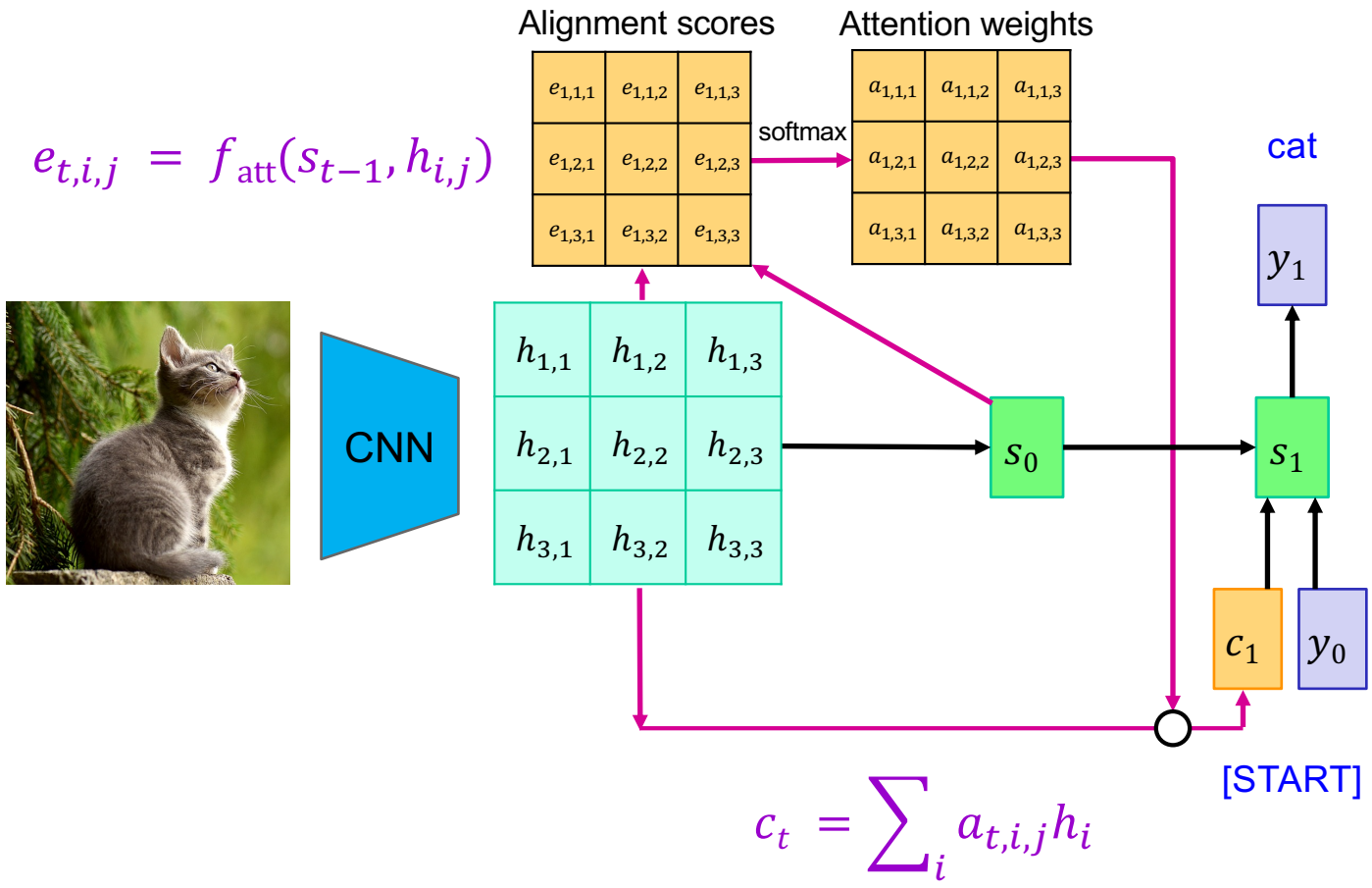
$$e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j})$$



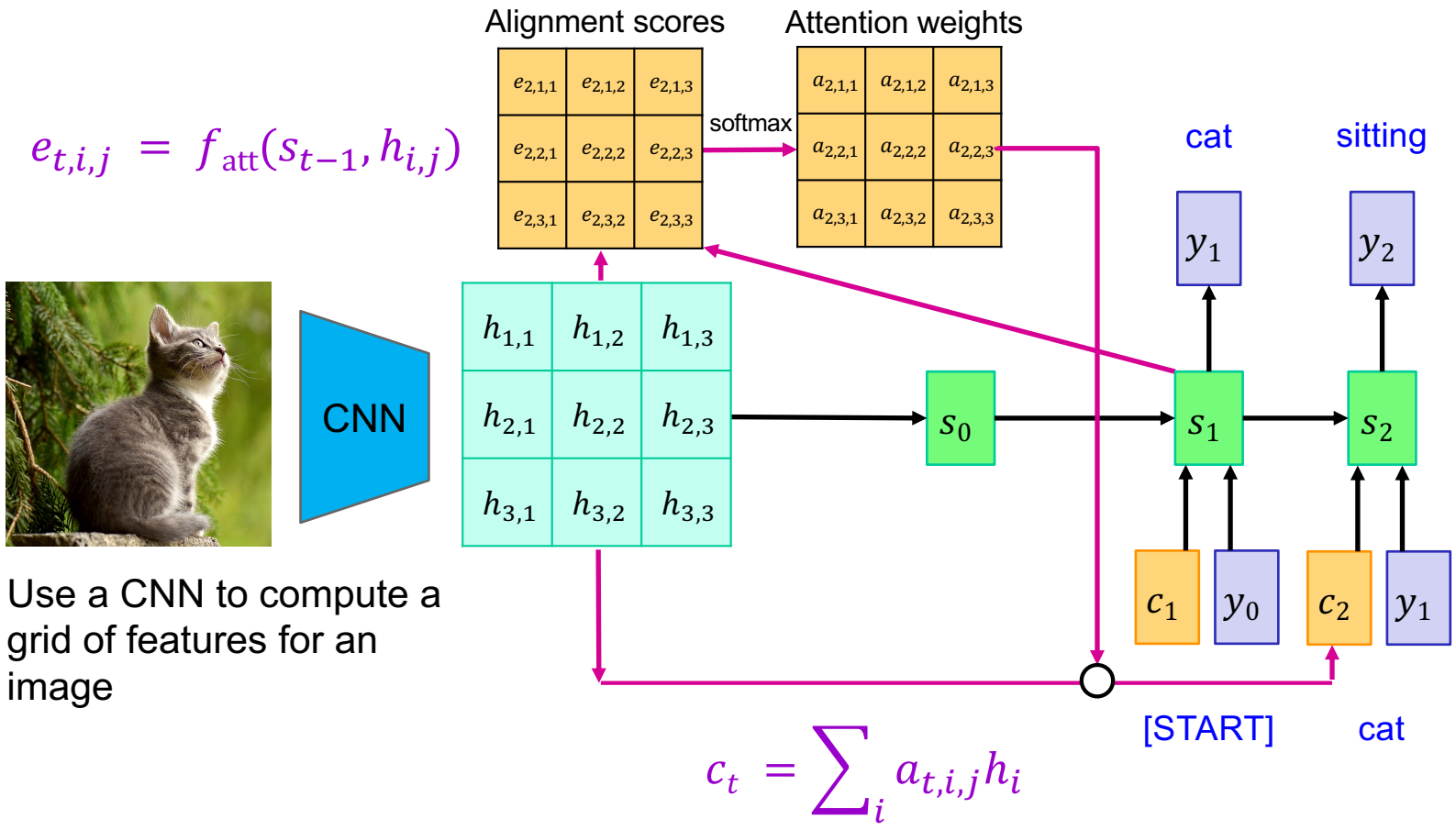

Use CNN to extract a grid of features

K. Xu et al., Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, ICML 2015

# Image captioning with RNNs and attention

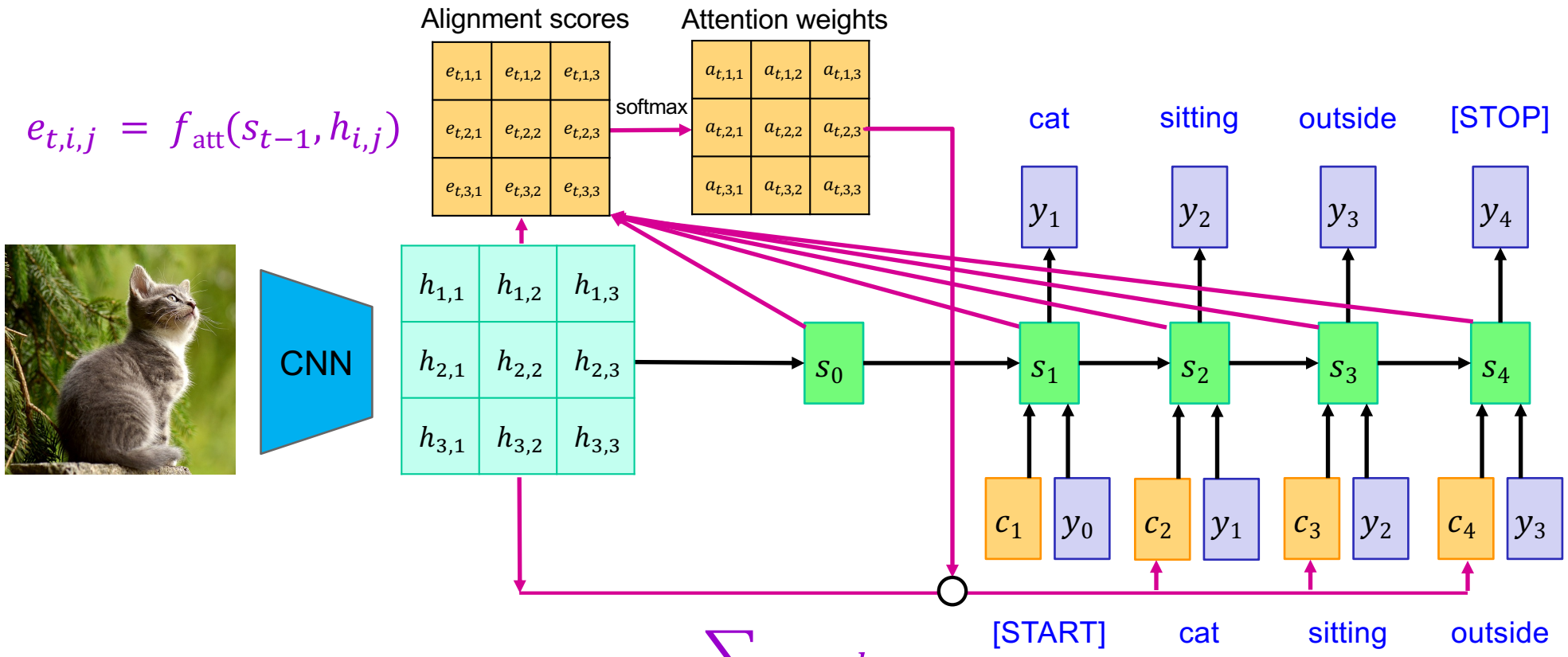


# Image captioning with RNNs and attention




Use a CNN to compute a grid of features for an image

# Image captioning with RNNs and attention



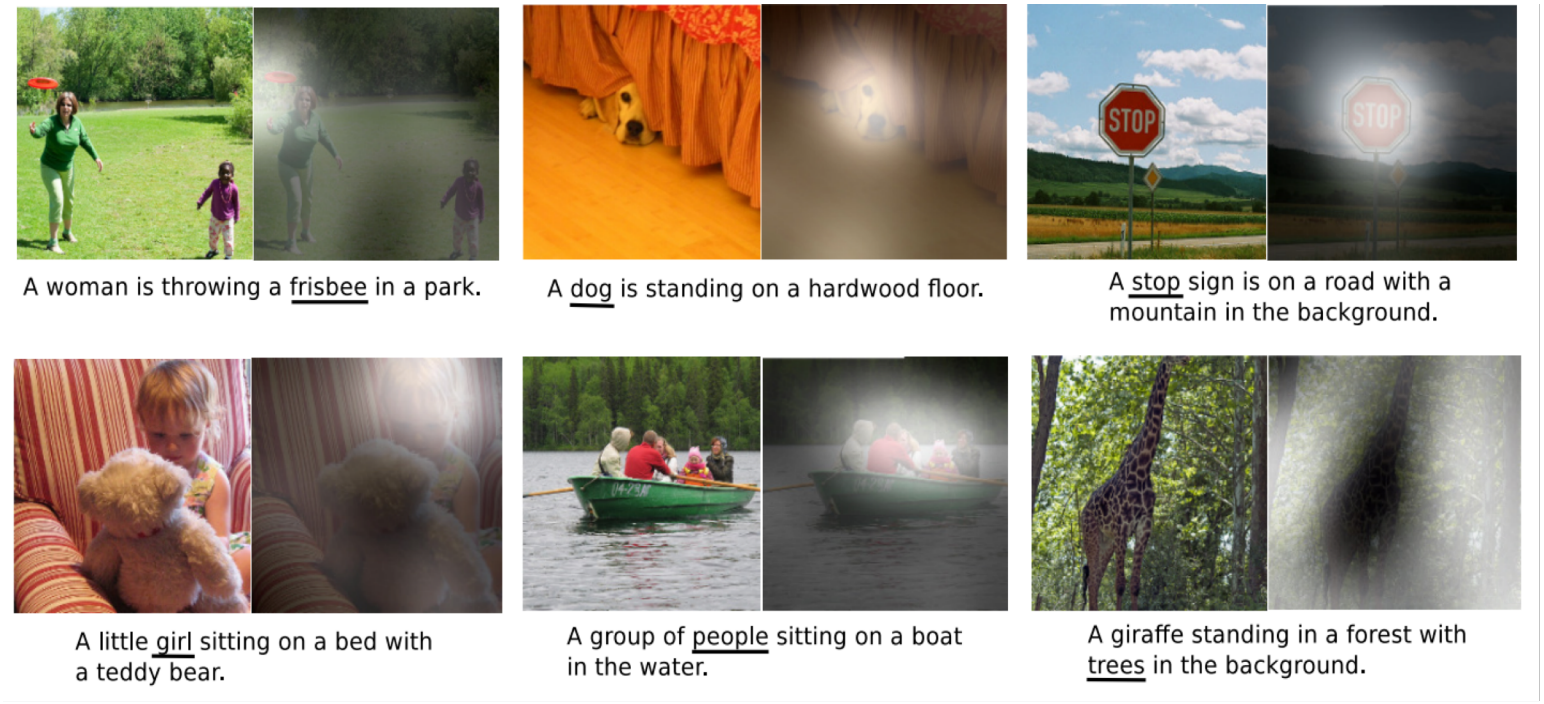

$$e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j})$$

$$c_t = \sum_i a_{t,i,j} h_i$$

Each time step of decoder uses a different context vector that looks at different parts of the input image

# Example results

- Good captions



# Example results

- Mistakes

A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and a hat on a skateboard.

A person is standing on a beach with a surfboard.

A woman is sitting at a table with a large pizza.

A man is talking on his cell phone while another man watches.

# Quantitative results

| Dataset | Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR |
|---|---|---|---|---|---|---|
| Flickr8k | Google NIC | 63 | 41 | 27 | - | - |
| | Soft-Attention | **67** | 44.8 | 29.9 | 19.5 | 18.93 |
| | Hard-Attention | **67** | **45.7** | **31.4** | **21.3** | **20.30** |
| Flickr30k | Google NIC | 66.3 | 42.3 | 27.7 | 18.3 | - |
| | Soft-Attention | 66.7 | 43.4 | 28.8 | 19.1 | **18.49** |
| | Hard-Attention | **66.9** | **43.9** | **29.6** | **19.9** | 18.46 |
| COCO | Google NIC | 66.6 | 46.1 | 32.9 | 24.6 | - |
| | Soft-Attention | 70.7 | 49.2 | 34.4 | 24.3 | **23.90** |
| | Hard-Attention | **71.8** | **50.4** | **35.7** | **25.0** | 23.04 |

Source

# X, Attend, and Y

"**Show, attend, and tell**" *(Xu et al, ICML 2015)*
Look at image, attend to image regions, produce question

"**Ask, attend, and answer**" *(Xu and Saenko, ECCV 2016)*
"**Show, ask, attend, and answer"** *(Kazemi and Elqursh, 2017)*
Read text of question, attend to image regions, produce answer

"**Listen, attend, and spell**" *(Chan et al, ICASSP 2016)*
Process raw audio, attend to audio regions while producing text

"**Listen, attend, and walk**" *(Mei et al, AAAI 2016)*
Process text, attend to text regions, output navigation commands

"**Show, attend, and interact**" *(Qureshi et al, ICRA 2017)*
Process image, attend to image regions, output robot control commands

"**Show, attend, and read**" *(Li et al, AAAI 2019)*
Process image, attend to image regions, output text

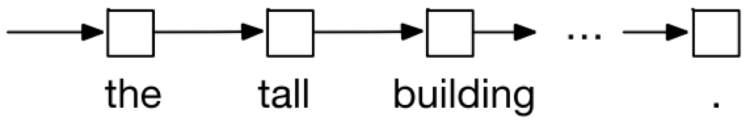
Source: J. Johnson

# Outline

- Vanilla seq2seq with RNNs
- Seq2seq with RNNs and attention
- Image captioning with attention
- Convolutional seq2seq with attention

# Recurrent vs. convolutional sequence models

- **Recurrent models:**
  - Treat input as ordered sequence (inherently sequential processing)
  - Build up context using the hidden vector




- **Convolutional models:**
  - Treat input as a grid indexed by time and feature dimension
  - Build up context using multiple layers of convolutions
  - Processing can be parallel at training time, but convolutions must be *causal*

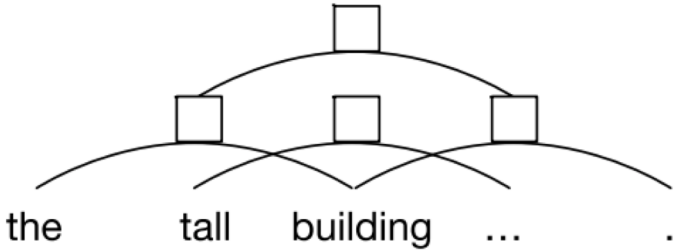



Image source

# WaveNet

- Goal: generate raw audio
  - Represented as sequence of 16-bit integer values (can be quantized to 256 discrete levels), 16K samples per second
- Applications: text-to-speech, music generation
  - Also works for speech recognition

Figure 1: A second of generated speech.

A. van den Oord et al., WaveNet: A generative model for raw audio, arXiv 2016

# WaveNet

- Training time: compute predictions of all timesteps in parallel (conditioned on ground truth)

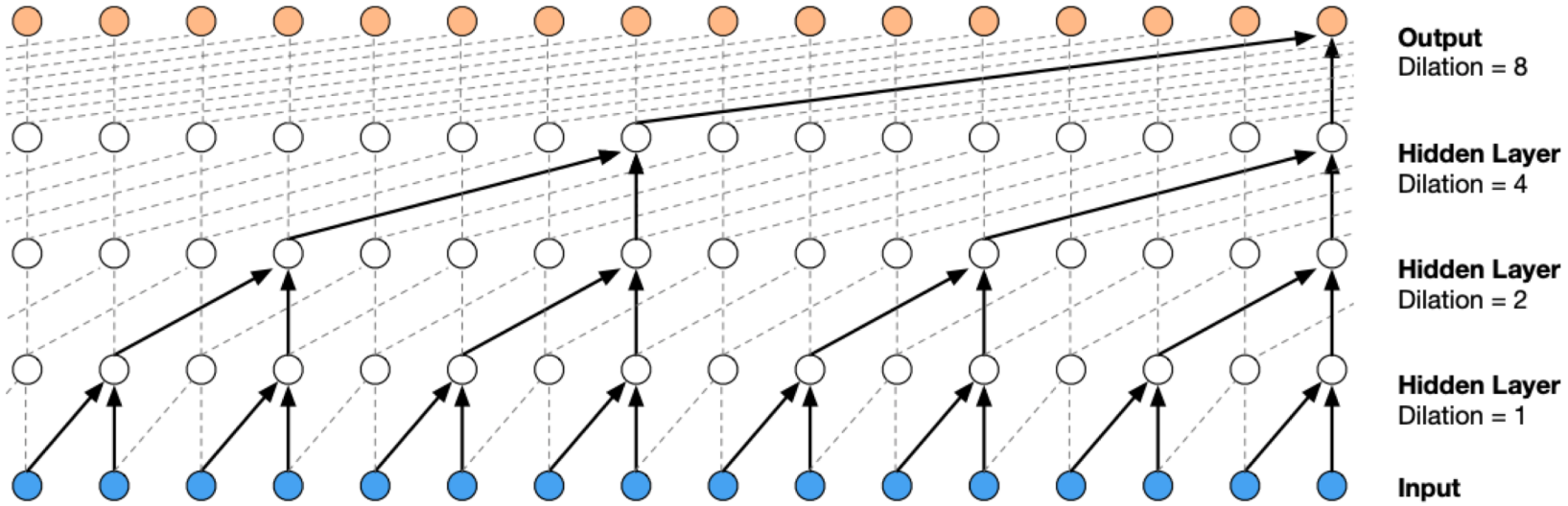



Image source

# WaveNet

- Test time: feed each predicted sample back into the model to make prediction at next timestep

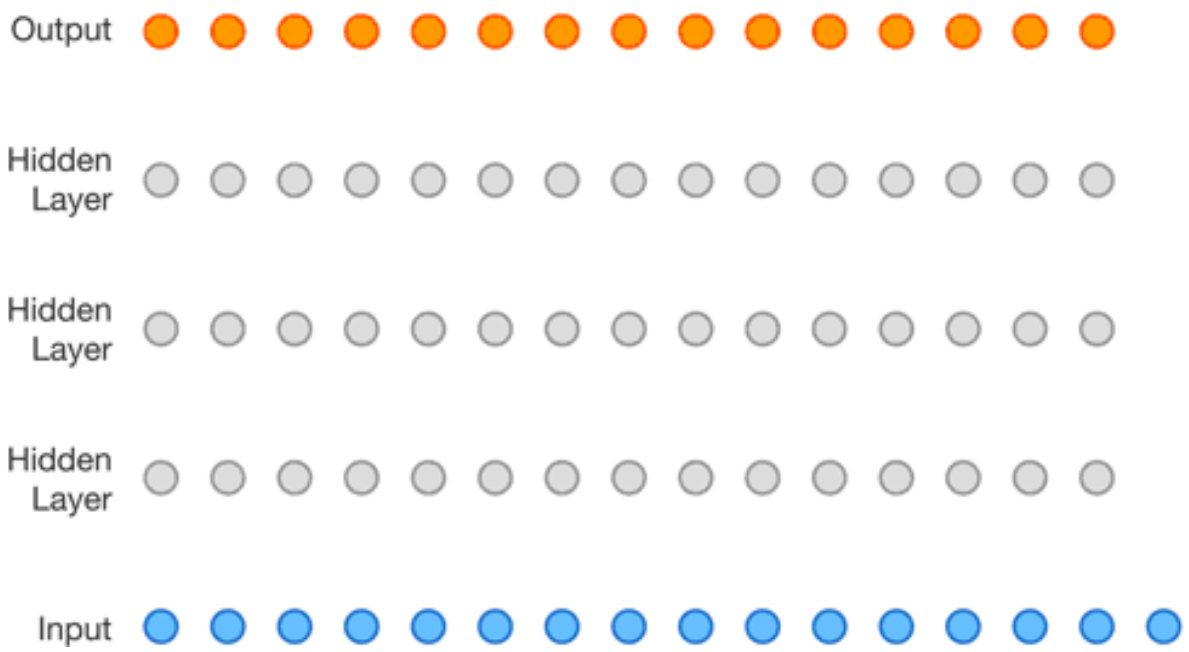



Image source

# WaveNet: Results

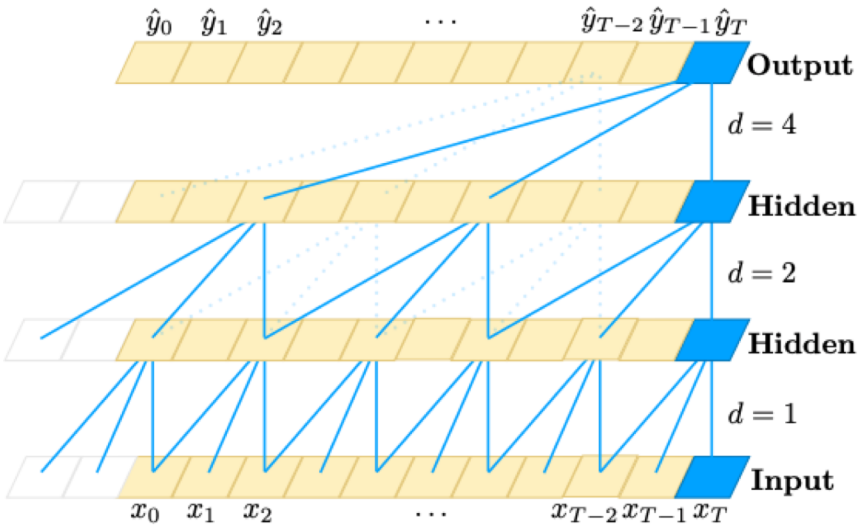
- Text-to-speech with different speaker identities:

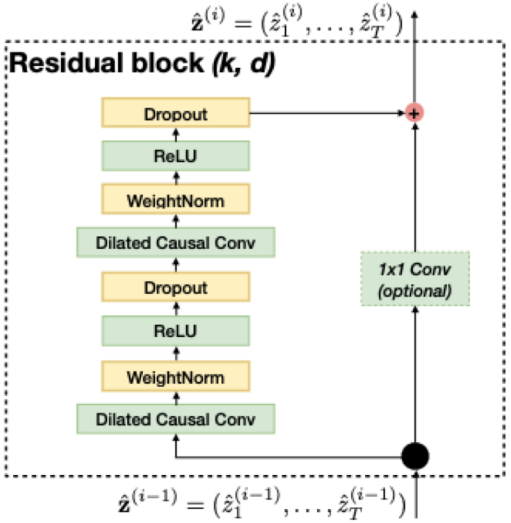

- Generated sample of classical piano music:

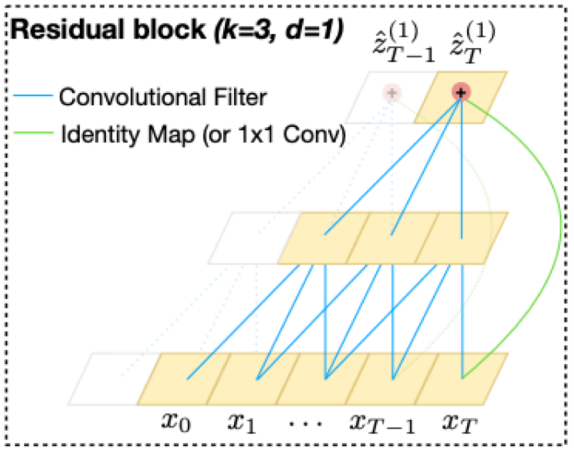
https://deepmind.com/blog/article/wavenet-generative-model-raw-audio

# Temporal convolutional networks (TCNs)

- TCNs can be competitive with RNNs for a variety of sequence modeling tasks

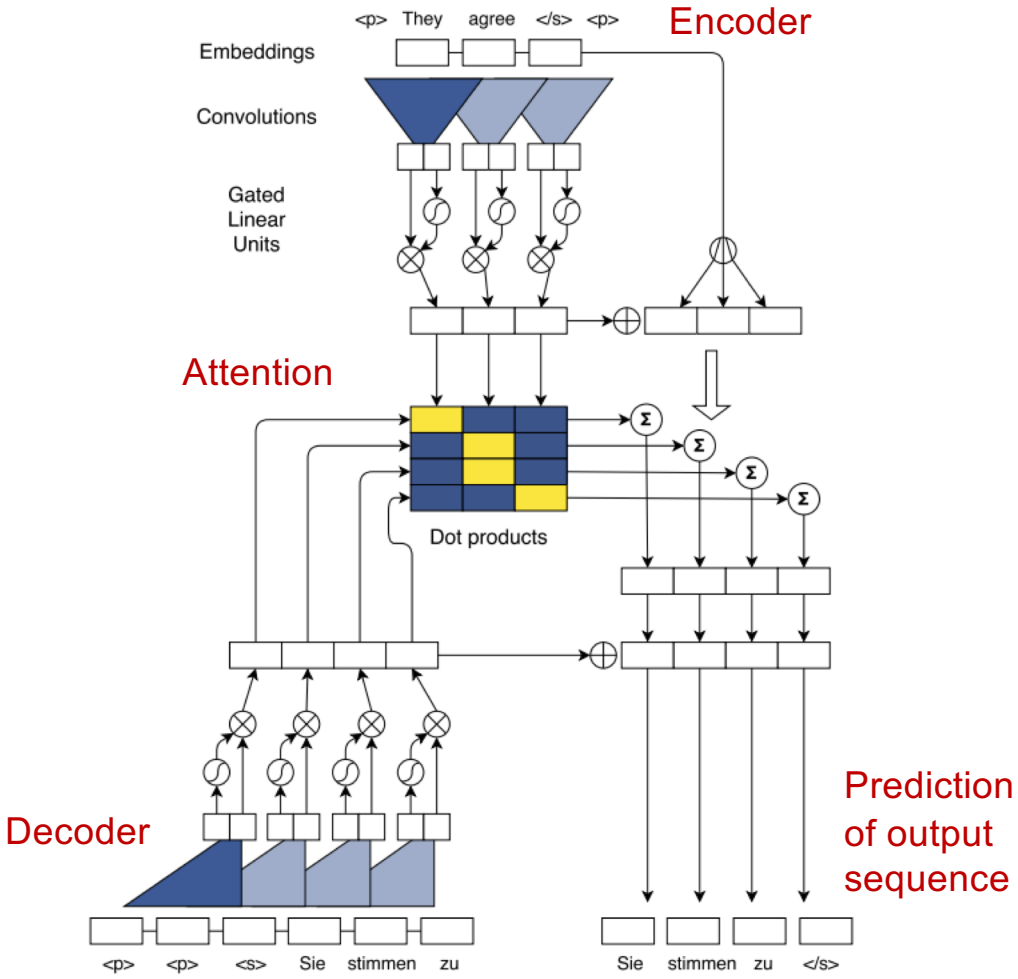



S. Bai, J. Kolter, and V. Koltun, An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, arXiv 2018

# Convolutional seq2seq with attention



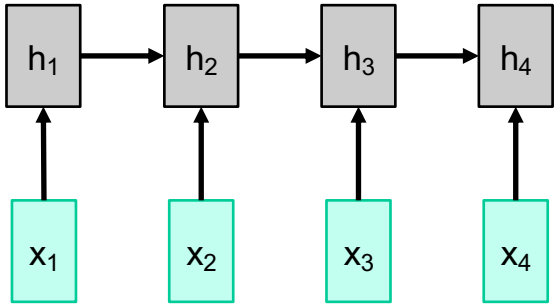

| WMT'14 English-German | BLEU |
|---|---|
| Luong et al. (2015) LSTM (Word 50K) | 20.9 |
| Kalchbrenner et al. (2016) ByteNet (Char) | 23.75 |
| Wu et al. (2016) GNMT (Word 80K) | 23.12 |
| Wu et al. (2016) GNMT (Word pieces) | 24.61 |
| ConvS2S (BPE 40K) | 25.16 |

| WMT'14 English-French | BLEU |
|---|---|
| Wu et al. (2016) GNMT (Word 80K) | 37.90 |
| Wu et al. (2016) GNMT (Word pieces) | 38.95 |
| Wu et al. (2016) GNMT (Word pieces) + RL | 39.92 |
| ConvS2S (BPE 40K) | 40.51 |

J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. Dauphin, Convolutional sequence to sequence learning, ICML 2017

# Different ways of processing sequences

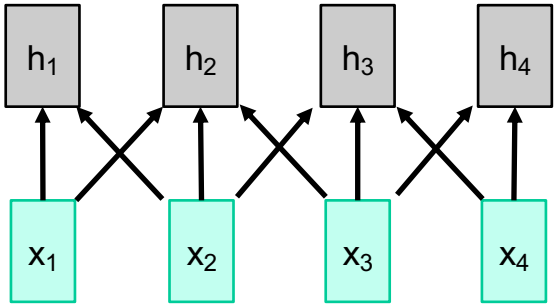
## RNN




Works on **ordered sequences**

- Pros: Good at long sequences: After one RNN layer, $h_T$ "sees" the whole sequence
- Cons: Not parallelizable: need to compute hidden states sequentially

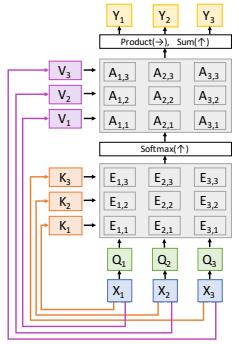
## 1D convolutional network




Works on **multidimensional grids**

- Con: Bad at long sequences: Need to stack many conv layers for outputs to "see" the whole sequence
- Pro: Highly parallel: Each output can be computed in parallel

## Transformer



- Works on **sets of vectors**
- Pro: Good at long sequences: after one self-attention layer, each output "sees" all inputs!
- Pro: Highly parallel: Each output can be computed in parallel
- Con: Very memory-intensive