Attention Networks

Bryan Pardo
Deep Learning
Northwestern University

Language model recap

The goal: predict the next token

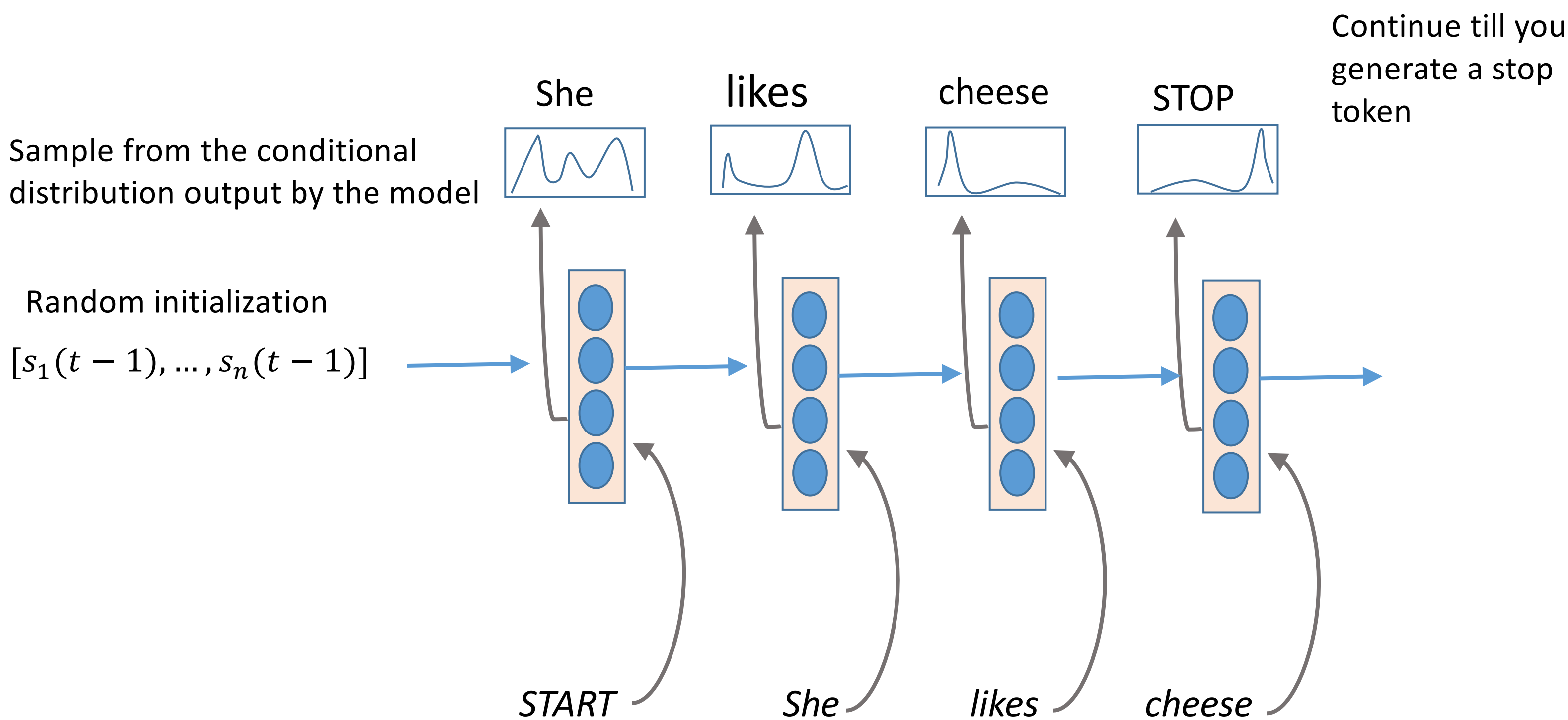
- Each sentence is its own label.

- Given “An apple is…”, predict “good” as the next word.

- Our model output will be a probability distribution over the 1003 element vector (top 1000 words + start + start + other).

- We can use cross-entropy loss, comparing the one-hot vector to the probability vector output by the model.

Generating a new sentence with the model



Let’s think about translation

What’s an “embedding”?

- An embedding is representation of some thing as a d-dimensional vector of real values.

- Typically we want an embeddings to have meaningful “groupings”, if we plot the embeddings of objects in a d-dimensional space.

- Here’s an embedding for people: telephone number.
  - Does it group people well? If so, how are they grouped?

- What’s an embedding for words?

What is “conditioning”?

- Term borrowed from statistics & probability (i.e. a conditional distribution)
- Our language model (or seq2seq model) outputs a probability distribution over the set of words, we want to modify our probability estimates about the next word based on what we know about the prior sequence. This modification is called “conditioning”

UNCONDITIONED: P(next_word) : just about any word is likely

CONDITIONED: P(next_word | “Two plus two equals”) : “four” is most likely

Going from English to Spanish

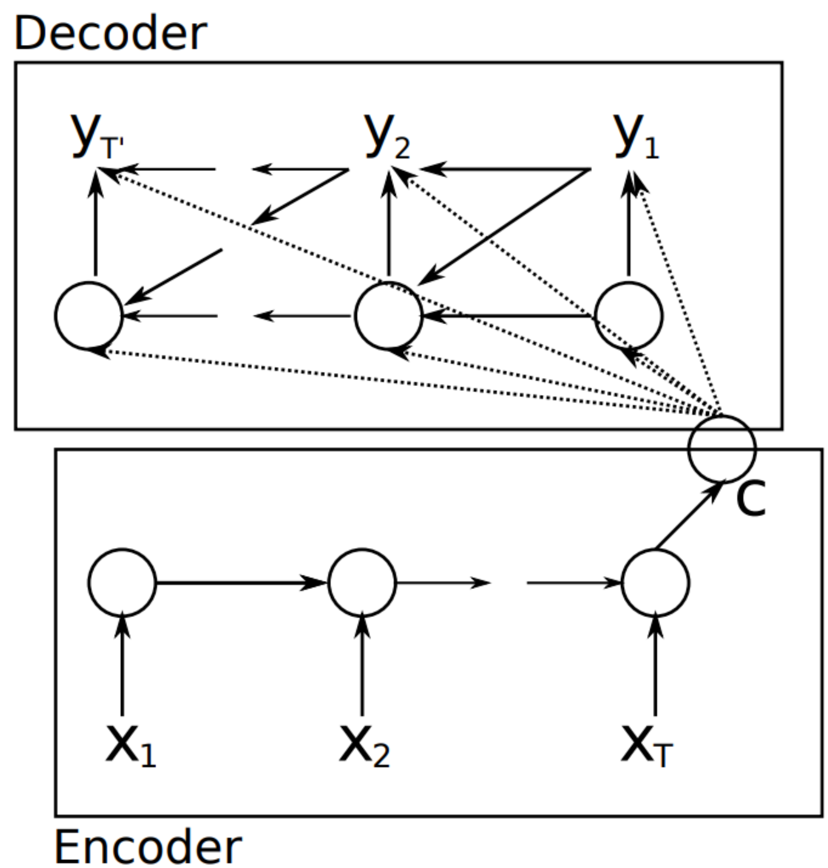
- English: She likes cheese.

- Spanish: A ella le gusta el queso.

- What’s different about these two sentences?
  - Number of words.
  - Ordering of words.
  - Which noun is doing the acting.

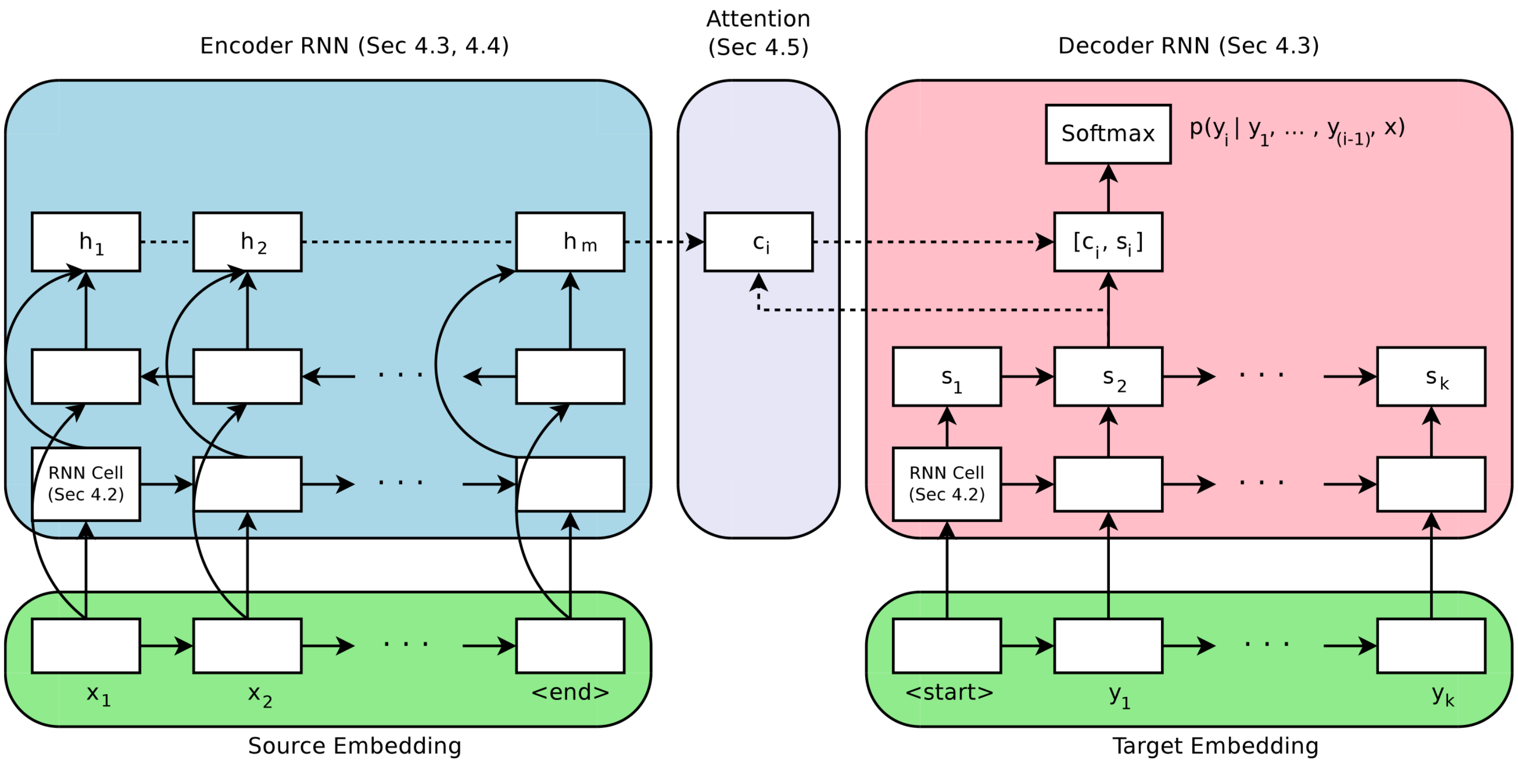
A two-part network

- We need something to wrap up the ”meaning” of the sentence in a way that gets past all these surface grammatical differences
- We need something to turn that “meaning” into the language of our choice.
- This is an encoder/decoder network.



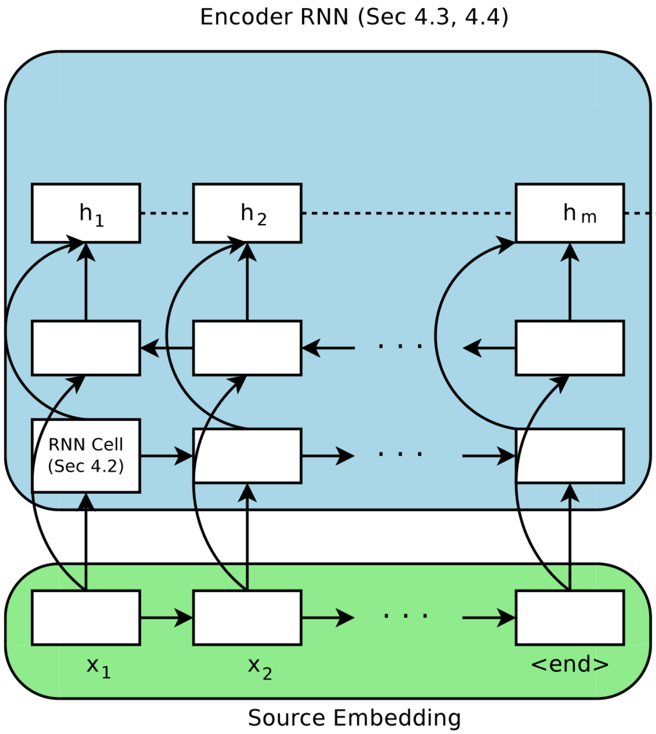
Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*.

A general framework for seq2seq (with attention)



Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

The encoder



Each **h** is a vector concatenating the hidden state of all the cells in the forward direction and also the cells in the backward direction.

Note that it is bi-directional

Typically this RNN is an LSTM (or a GRU)

Each is a token (e.g. a word) encoded in an embedding (e.g. one-hot encoding)

Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

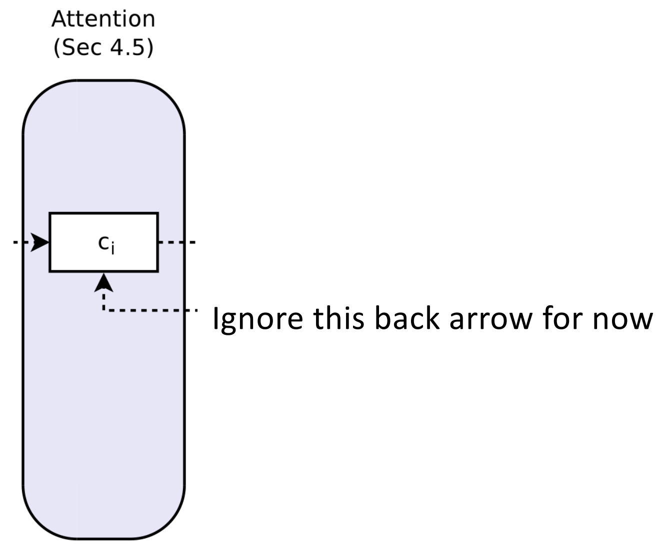
Making a fixed-length context vector

The context is a function of the encoder's full output sequence

$$c = q(h_1, h_2, \ldots, h_m)$$

The first seq2seq model* just used the last state as the context

$$q(h_1, h_2, \ldots, h_m) = h_m$$



* Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*.

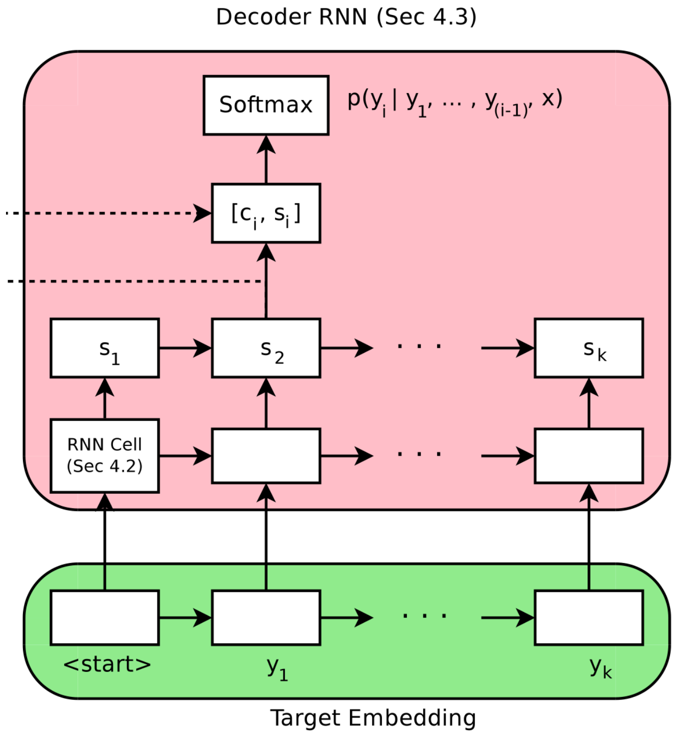
Using context in generation

The final output is a probability distribution over the classes (words) of the output language

The context is introduced to the encoder here

s is the state of the decode (a vector of RNN outputs). Its size depends on the the layer size.

The previous token output by the decoder



Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

Using decoder state to guide "attention"

The context at decoder step i is attention-weighted sum of the full sequence of encoder states.

$$c_i = \sum_j \alpha_{ij} h_j$$

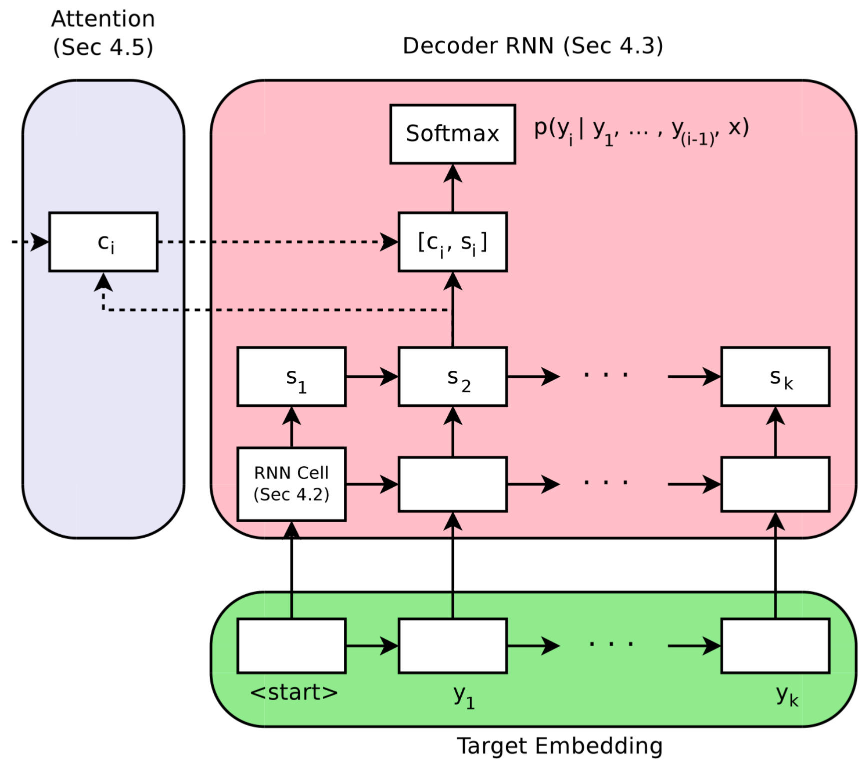
Attention weights are normalized with a softmax.

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

Un-normalized attention for each encoder step is determined by a small feed-forward network

$$e_{ij} = \tanh(W_s s_i + W_h h_j)$$

W_s and W_h are learned weight matrices



Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

Questions about our network

- The single-layer feed-forward network that implements attention attention is called “additive” attention. Can you say why?

- What are some implications of these design choices for….
  - The ability to completely ignore an uninteresting portion of the context?
  - The ability to selectively attend to any portion of the input sequence?

- How does the context (aka attention) from state *i*-1 influence the decoder state *i*? How is that different from the previous architecture

Multiplicative attention

The context at decoder step *i* is attention-weighted sum of the full sequence of encoder states.

$$c_i = \sum_j \alpha_{ij} h_j$$

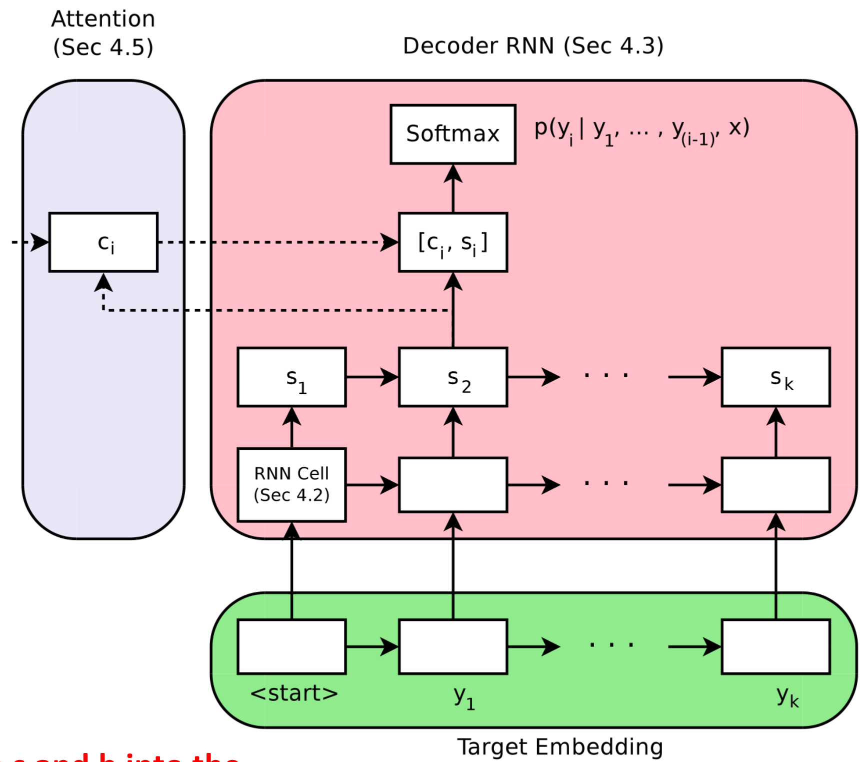
Attention weights are normalized with a softmax.

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

Weights are determined by multiplying the decoder state by the hidden state.

$$e_{ij} = \langle W_s s_i, W_h h_j \rangle$$

W_s , W_h **: Learned matrices that also transform s and h into the right shape for multiplication.**



Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

How much does attention help?

Attention	newstest2013	Params
Mul-128	22.03 ± 0.08 (22.14)	65.73M
Mul-256	22.33 ± 0.28 (22.64)	65.93M
Mul-512	21.78 ± 0.05 (21.83)	66.32M
Mul-1024	18.22 ± 0.03 (18.26)	67.11M
Add-128	22.23 ± 0.11 (22.38)	65.73M
Add-256	22.33 ± 0.04 (22.39)	65.93M
Add-512	$\mathbf{22.47} \pm 0.27$ (22.79)	66.33M
Add-1028	22.10 ± 0.18 (22.36)	67.11M
None-State	9.98 ± 0.28 (10.25)	64.23M
None-Input	11.57 ± 0.30 (11.85)	64.49M

Table 5: BLEU scores on newstest2013, varying the type of attention mechanism.

- BLEU scores are a measure of language translation quality
- Higher is better.
- Additive attention with hidden-state vectors of size 512 were the best
- Despite this, multiplicative attention is the most-used now.

Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

Have I mentioned residual connections?

- How do you decide how many layers your network should have?
- Too shallow: not enough representational power.
- Too deep risks slow convergence, poor local minimum

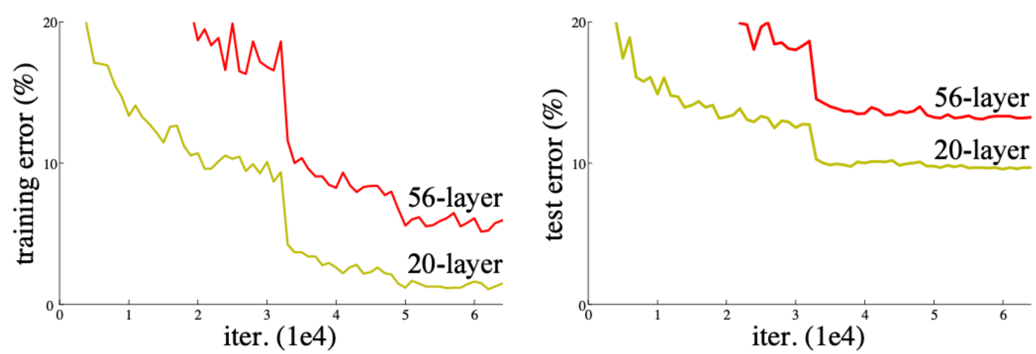
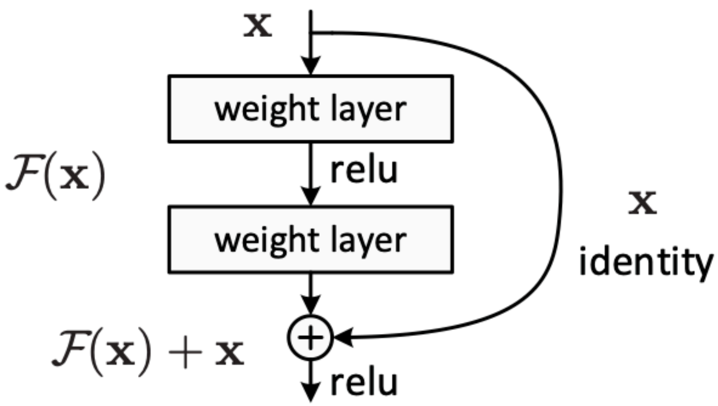


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778)

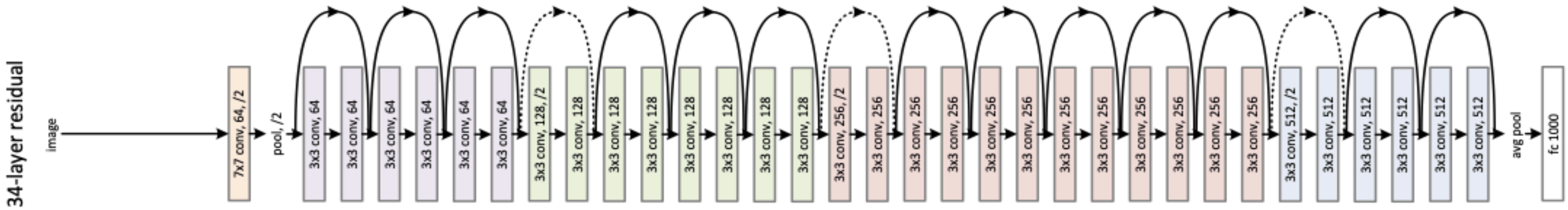
ResNets



- Build a network out of skippable layers
- Sum the output of the previous layer with the output of this weight layer, before applying the activation function
- If this layer’s weights aren’t adding anything, bypass it
- This allows VERY DEEP nets.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778)

A 34 layer ResNet



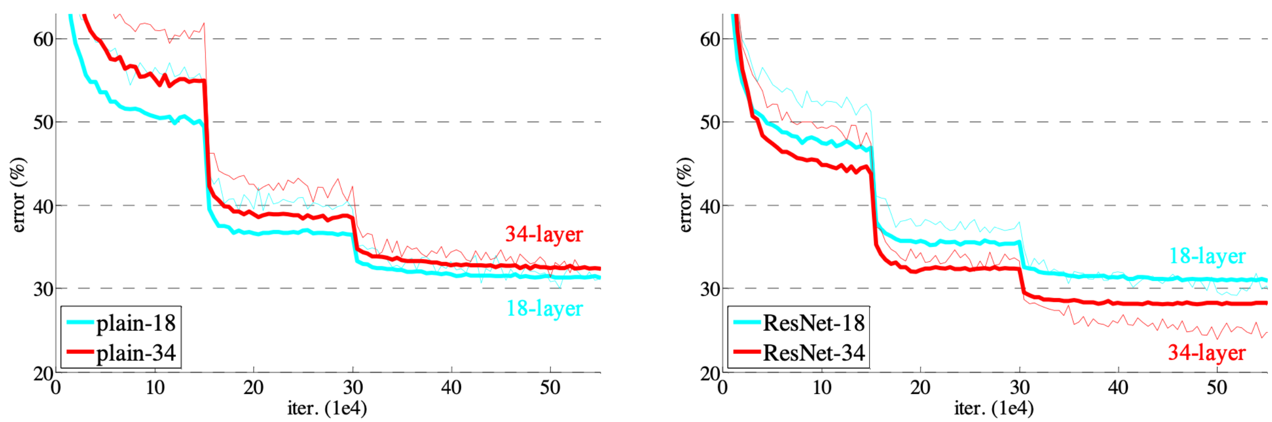


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778)

Residual connections in Seq2Seq models

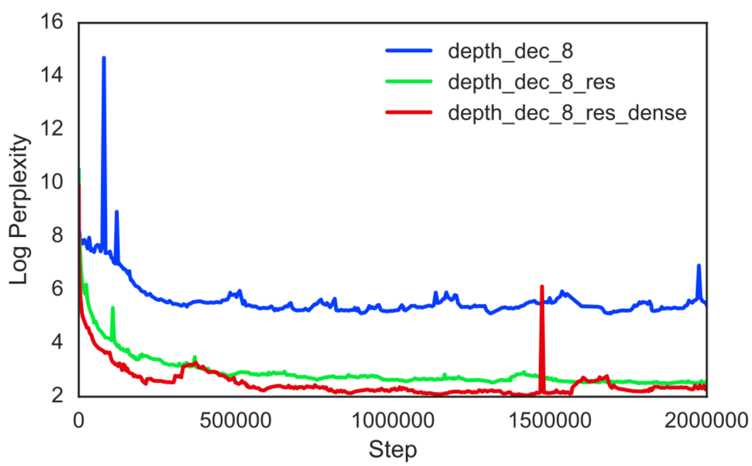


Figure 2: Training plots for deep decoder with and without residual connections, showing log perplexity on the eval set.

Depth	**newstest2013**	**Params**
Enc-2	21.78 ± 0.05 (21.83)	66.32M
Enc-4	**21.85** ± 0.32 (22.23)	69.47M
Enc-8	21.32 ± 0.14 (21.51)	75.77M
Enc-8-Res	19.23 ± 1.96 (21.97)	75.77M
Enc-8-ResD	17.30 ± 2.64 (21.03)	75.77M
Dec-1	21.76 ± 0.12 (21.93)	64.75M
Dec-2	21.78 ± 0.05 (21.83)	66.32M
Dec-4	**22.37** ± 0.10 (22.51)	69.47M
Dec-4-Res	17.48 ± 0.25 (17.82)	68.69M
Dec-4-ResD	21.10 ± 0.24 (21.43)	68.69M
Dec-8	01.42 ± 0.23 (1.66)	75.77M
Dec-8-Res	16.99 ± 0.42 (17.47)	75.77M
Dec-8-ResD	20.97 ± 0.34 (21.42)	75.77M

Table 3: BLEU scores on newstest2013, varying the encoder and decoder depth and type of residual connections.

Britz, D., Goldie, A., Luong, M. T., & Le, Q. (2017). Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.