EECS 498/598: Deep Learning
Lecture 6. Language applications

Honglak Lee

02/15/2019

Outline

- **Application: Text Classification**
- Application: Language Modeling
- Application: Machine Translation (Seq2Seq)
  - Machine Translation with Attention
- Application: Question Answering
- Advanced Attention mechanism: Transformers
- Pre-training/Transfer Learning

Text Classification

Application: Sentiment analysis

Customer Reviews

Speech and Language Processing, 2nd Edition

15 Reviews

5 star: (8)
4 star: (3)
3 star: (3)
2 star: (0)
1 star: (1)

Average Customer Review

(15 customer reviews)

Share your thoughts with other customers

Create your own review

The most helpful favorable review

4 of 4 people found the following review helpful

Great introductions and reference book

I read the first edition of that book and it is terrific. The second edition is much more adapted to current research. Statistical methods in NLP are more detailed and some syntax-based approaches are presented. My specific interest is in machine translation and dialogue systems. Both chapters are extensively rewritten and much more elaborated. I believe this book is...

Read the full review ›

Published on August 9, 2008 by carheg

› See more **5 star**, **4 star** reviews

Vs.

The most helpful critical review

37 of 37 people found the following review helpful

Good description of the problems in the field, but look elsewhere for practical solutions

The authors have the challenge of covering a vast area, and they do a good job of highlighting the hard problems within individual sub-fields, such as machine translation. The availability of an accompanying Web site is a strong plus, as is the extensive bibliography, which also includes links to freely available software and resources.

Now for the...

Read the full review ›

Published on April 2, 2009 by P. Nadkarni

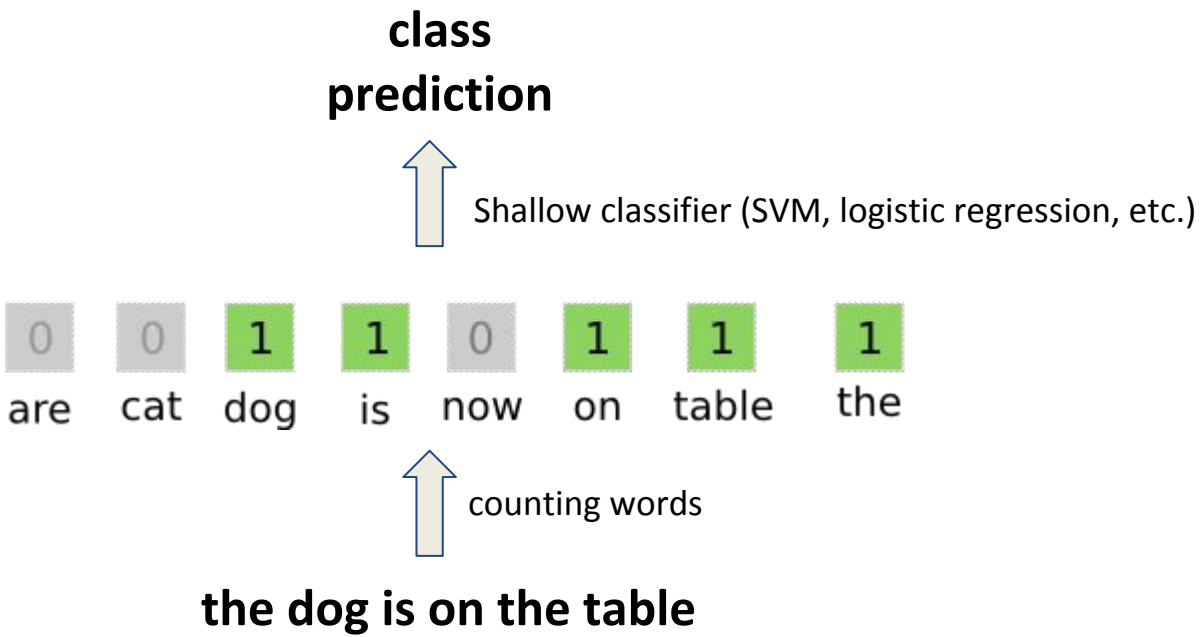
› See more **3 star**, 2 star, **1 star** reviews

Slides credit: Marina Neves

Baseline: Bag of words

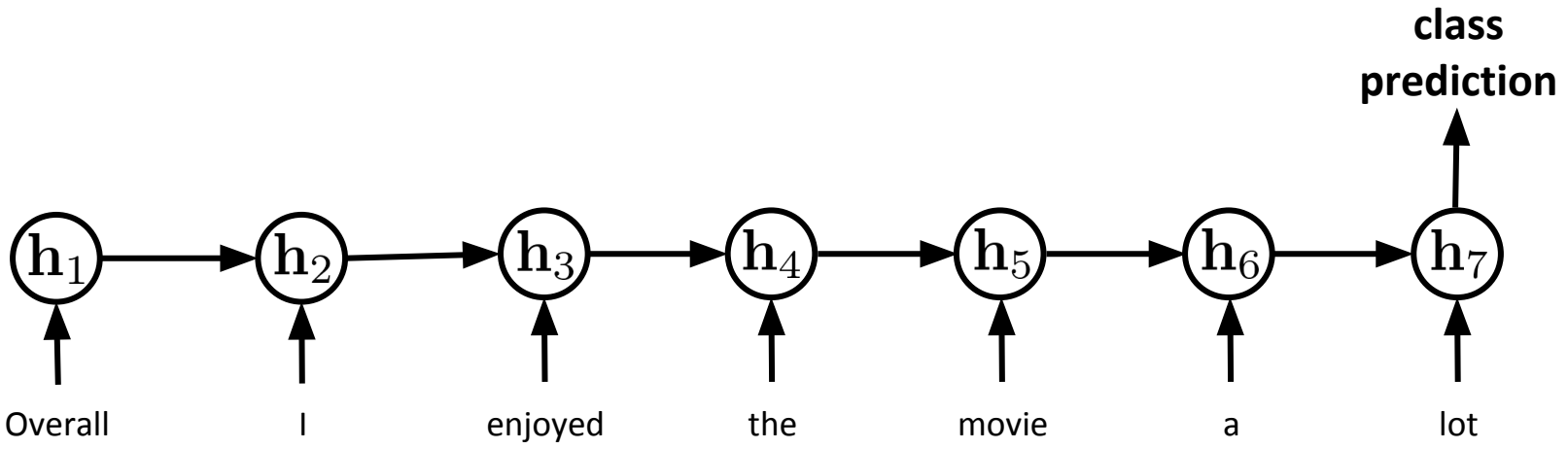
counting # of each word in the vocabulary

→ feature for shallow classifier



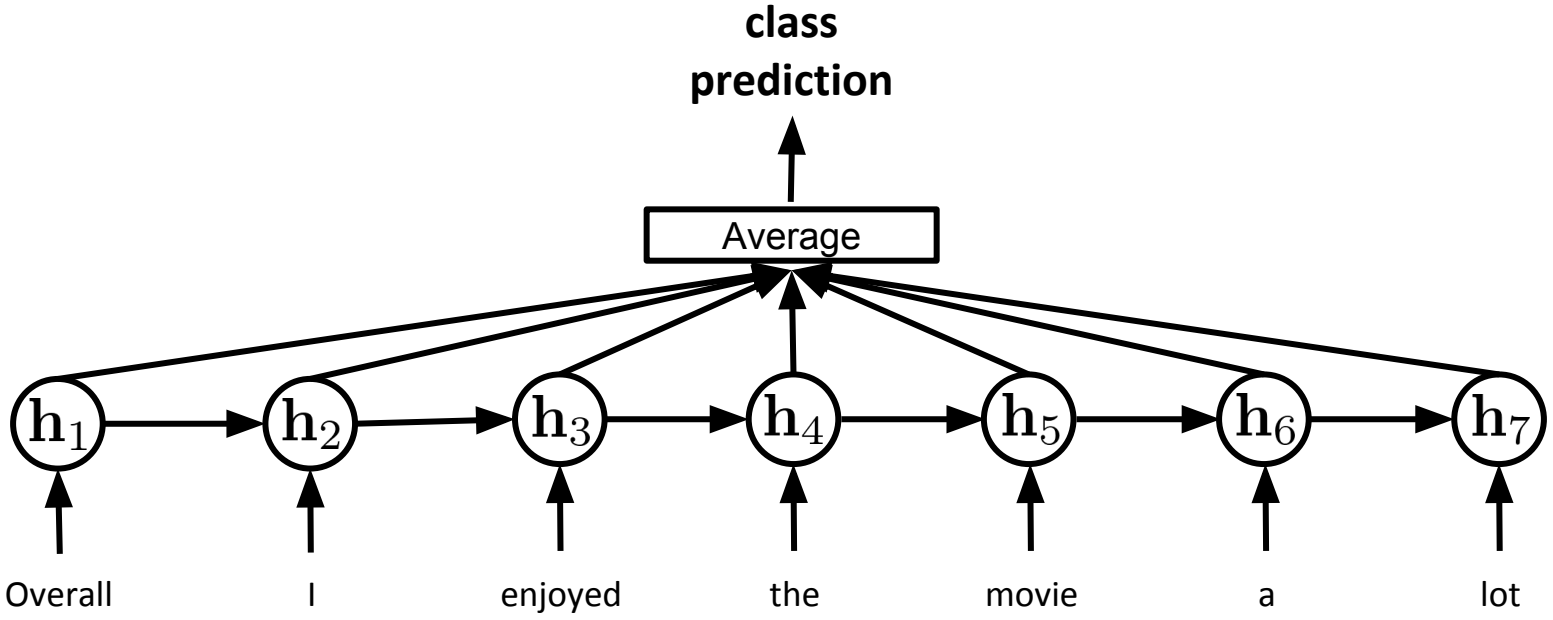
RNN/LSTM for text classification

- Read a full sentence (or paragraph) and predict the class label



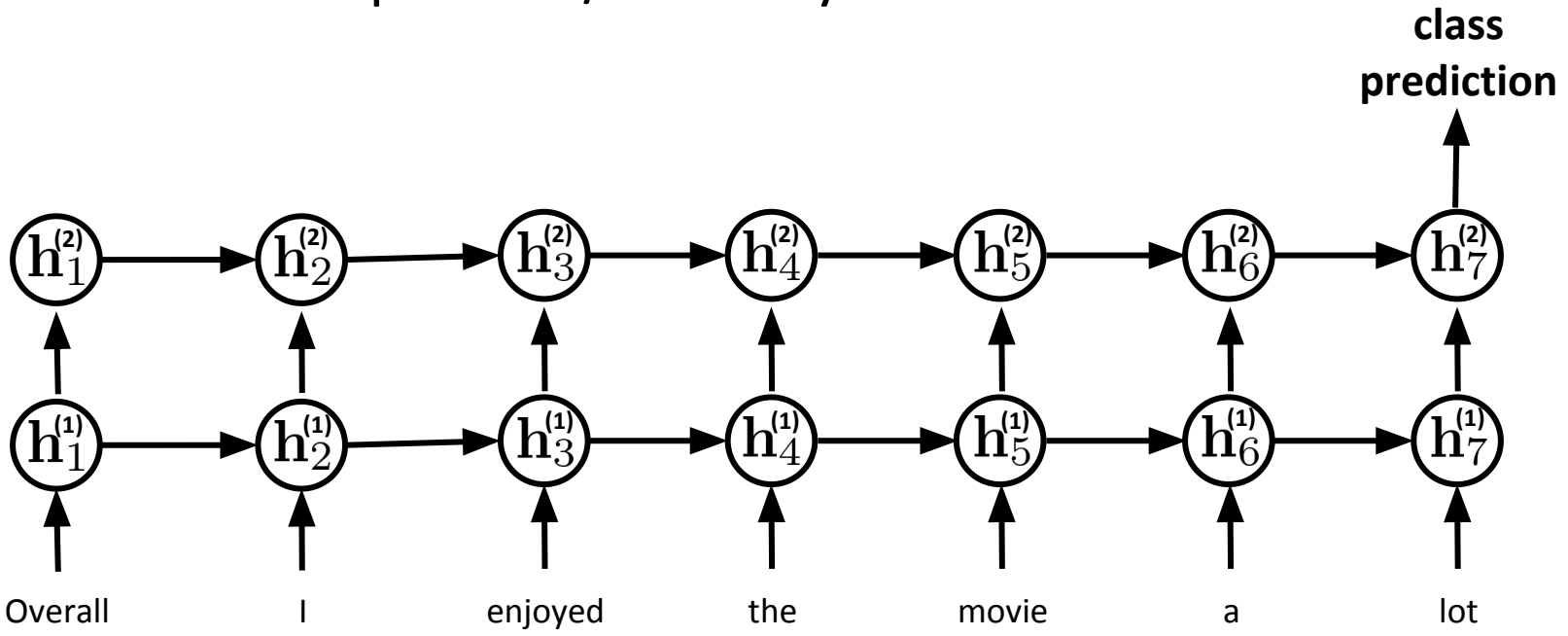
RNN/LSTM for text classification

- Read a full sentence (or paragraph) and predict the class label
- Variations: (average/max) pooling over time



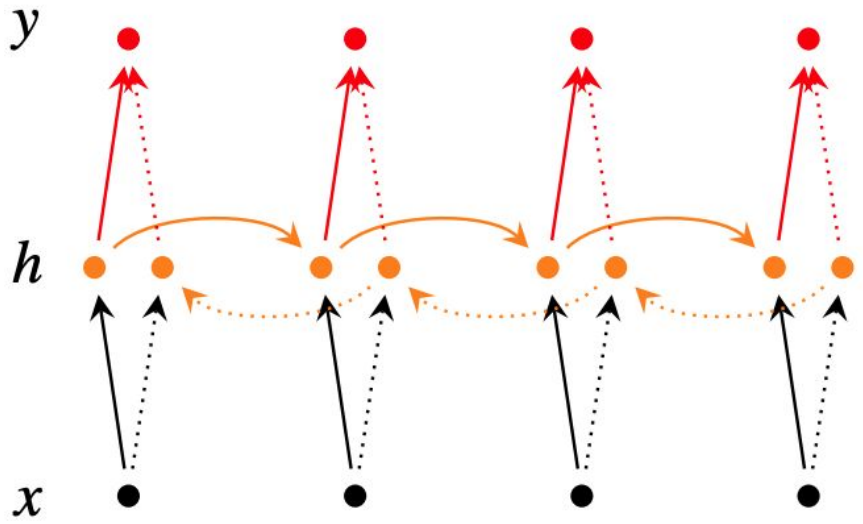
RNN/LSTM for text classification

- Read a full sentence (or paragraph) and predict the class label
- Variation: multiple RNN/LSTM layers



Bi-directional RNN (or LSTM)

Problem: For classification you want to incorporate information from words both preceding and following



$$\vec{h}_t = f(\vec{W}x_t + \vec{V}\vec{h}_{t-1} + \vec{b})$$

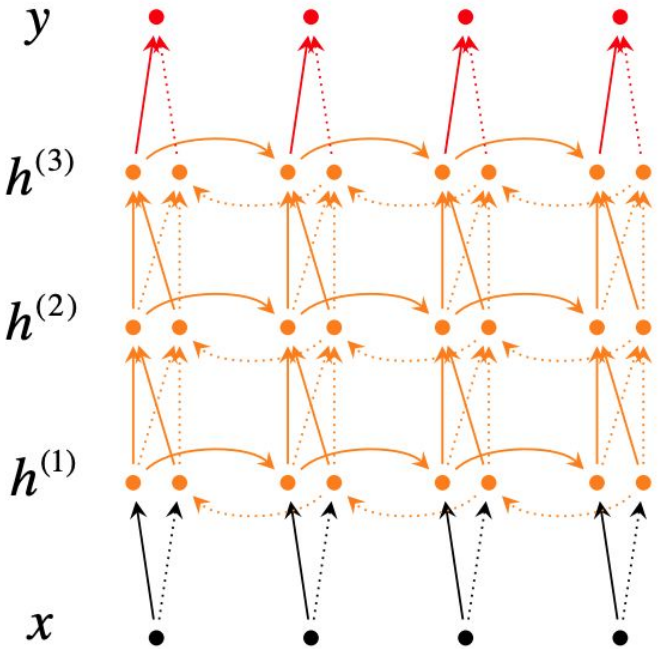
$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

$$y_t = g(U[\vec{h}_t; \overleftarrow{h}_t] + c)$$

$h = [\vec{h}; \overleftarrow{h}]$ now represents (summarizes) the past and future around a single token.

Slides credit: Richard Socher

Bi-directional RNN (or LSTM)



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

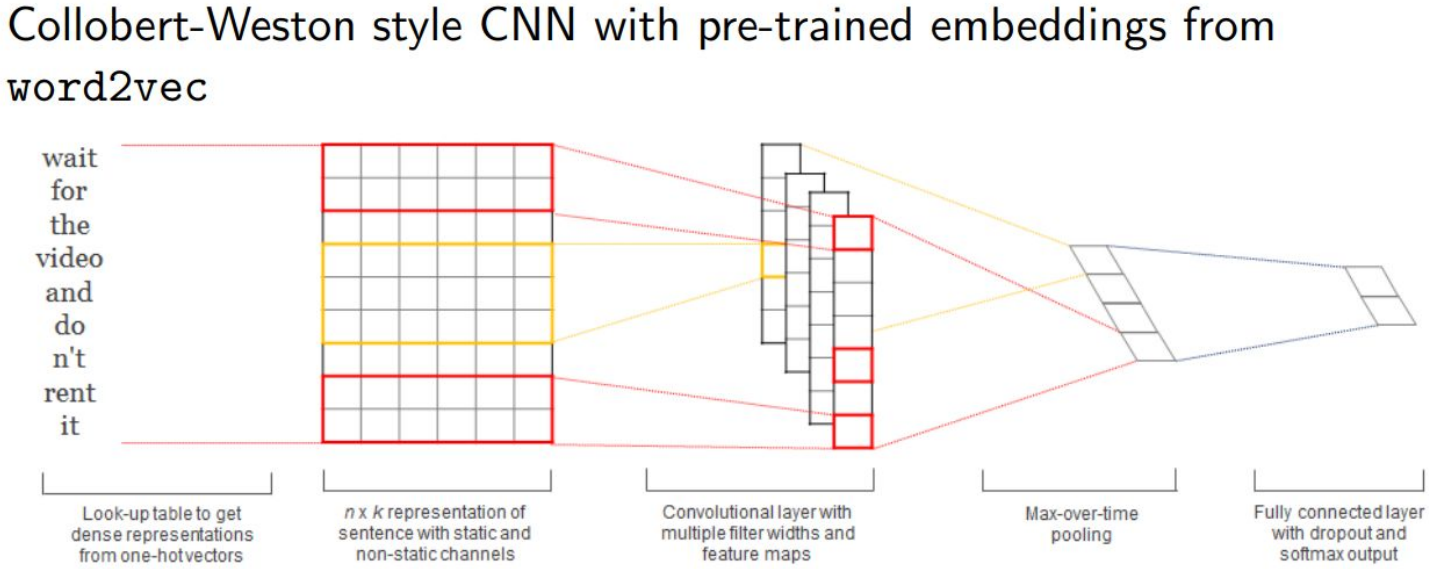
$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$y_t = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

Each memory layer passes an intermediate sequential representation to the next.

Slides credit: Richard Socher

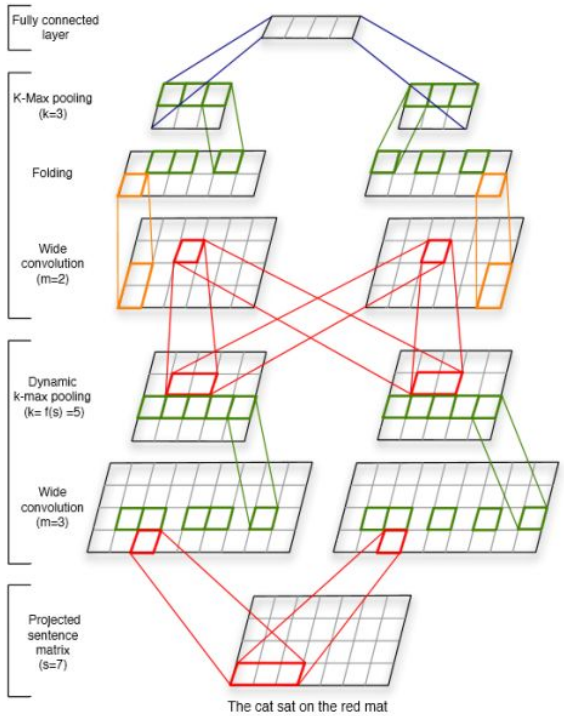
Convolutional Neural Network for text classification



Kim, Yoon. "Convolutional neural networks for sentence classification." *EMNLP 2014*.

Slides credit: Yoon Kim

Dynamic Convolutional Neural Network



Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A convolutional neural network for modelling sentences". 2014

Slides credit: Yoon Kim

Model Comparison

- **Bag of words**: Surprisingly good baseline for simple classification problems. Especially if followed by a few layers!

- **CNNs**: Good for classification, unclear how to incorporate phrase level annotation (can only take a single label), need zero padding for shorter phrases, hard to interpret, easy to parallelize on GPUs

- **RNNs**: Most cognitively plausible (reading from left to right), lots of improvements with gates (GRUs, LSTMs, etc.)

Slides credit: Richard Socher

Model Comparison

Method	MR	SST-2	SST-5	TREC	SUBJ	IMDB
SVM (Socher et al., 2013)	—	79.4	40.7	—	—	—
NB (Socher et al., 2013)	—	81.8	41.0	—	—	—
NBSVM-bi (Wang and Manning, 2012)	79.4	—	—	—	93.2	91.2
SVM_S (Silva et al., 2011)	—	—	—	95.0	—	—
Standard-RNN (Socher et al., 2013)	—	82.4	43.2	—	—	—
MV-RNN (Socher et al., 2012)	79.0	82.9	44.4	—	—	—
RNTN (Socher et al., 2013)	—	85.4	45.7	—	—	—
DRNN (Irsoy and Cardie, 2014)	—	86.6	49.8	—	—	—
Standard-LSTM (Tai et al., 2015)	—	86.7	45.8	—	—	—
bi-LSTM (Tai et al., 2015)	—	86.8	49.1	—	—	—
Tree-LSTM (Tai et al., 2015)	—	88.0	51.0	—	—	—
SA-LSTM (Dai and Le, 2015)	80.7	—	—	—	—	92.8
DCNN (Kalchbrenner et al., 2014)	—	86.8	48.5	93.0	—	—
CNN-MC (Kim, 2014)	81.1	88.1	47.4	92.2	93.2	—
MVCNN (Yin and Schütze, 2015)	—	89.4	49.6	—	93.9	—
Dep-CNN (Ma et al., 2015)	81.9	—	49.5	95.4	—	—
Neural-BoW (Kalchbrenner et al., 2014)	—	80.5	42.4	88.2	—	—
DAN (Iyyer et al., 2015)	80.3	86.3	47.7	—	—	89.4
Paragraph-Vector (Le and Mikolov, 2014)	—	87.8	48.7	—	—	92.6
WRRBM+BoW(bnc) (Dahl et al., 2012)	—	—	—	—	—	89.2
Full+Unlabeled+BoW(bnc) (Maas et al., 2011)	—	—	—	—	88.2	88.9
DSCNN	81.5	89.1	49.7	95.4	93.2	90.2
DSCNN-Pretrain	82.2	88.7	50.6	95.6	93.9	90.7

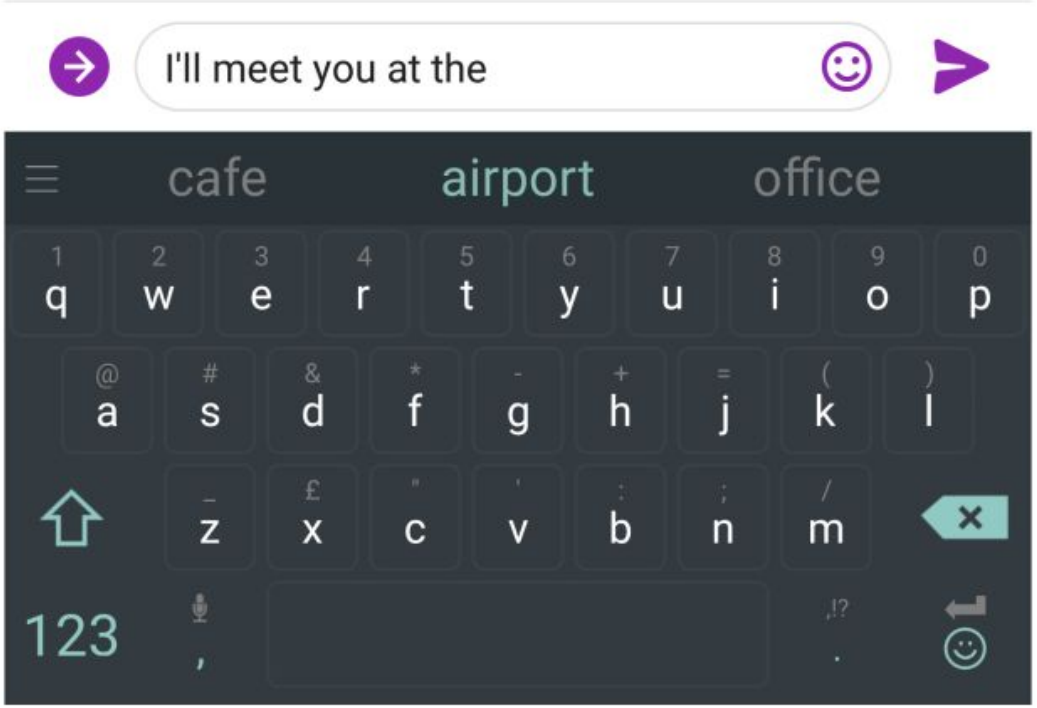
Rui Zhang, Honglak Lee, and Dragomir Radev. "Dependency sensitive convolutional neural networks for modeling sentences and documents." *NAACL 2016*.

Outline

- Application: Text Classification
- **Application: Language Modeling**
- Application: Machine Translation (Seq2Seq)
  - Machine Translation with Attention
- Application: Question Answering
- Advanced Attention mechanism: Transformers
- Pre-training/Transfer Learning

Language Modeling with RNNs

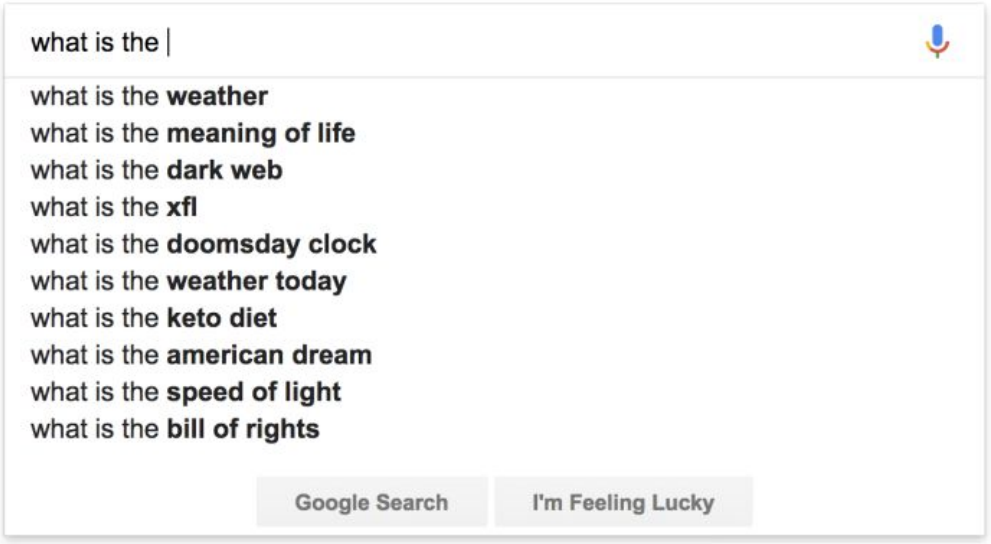
- You use language models every day!



Slides credit: Abigail See

Language Modeling with RNNs

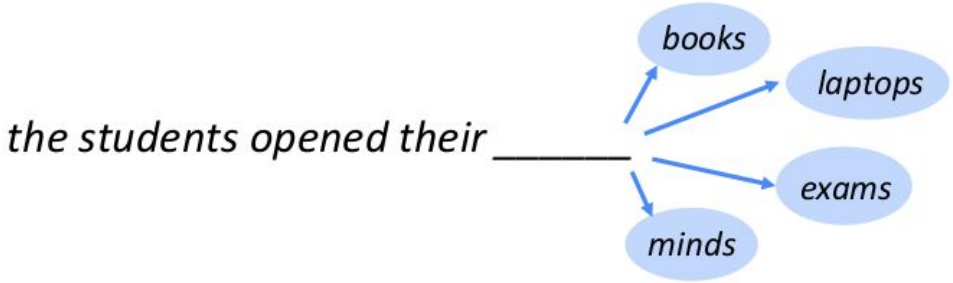
- You use language models every day!



Slides credit: Abigail See

Language Modeling with RNNs

- **Language Modeling** is the task of predicting what word comes next.



- More formally: given a sequence of words $x^{(1)}, x^{(2)}, ..., x^{(t)}$

$$P(x^{(t+1)}|x^{(t)}, ..., x^{(1)})$$

  where $x^{(t+1)}$ can be any word in the vocabulary $V = \{w_1, ..., w_{|V|}\}$

- A system that does this is called a **Language Model**.

Slides credit: Abigail See

Language Modeling with RNNs

- You can also think of a Language Model as a system that **assigns probability to a piece of text**.

- For example, if we have some text $x^{(1)}, x^{(2)}, \ldots, x^{(T)}$, then the probability of this text according to the language model is:

$$P(x^{(1)}, x^{(2)}, ..., x^{(T)}) = P(x^{(1)}) \times P(x^{(2)}|x^{(1)}) \times ... \times P(x^{(T)}|x^{(T-1)}, ..., x^{(1)})$$

Slides credit: Abigail See

Language Modeling with RNNs

- You can also think of a Language Model as a system that **assigns probability to a piece of text**.

- For example, if we have some text $x^{(1)}, x^{(2)}, \ldots, x^{(T)}$, then the probability of this text according to the language model is:

$$P(x^{(1)}, x^{(2)}, \ldots, x^{(T)}) = P(x^{(1)}) \times P(x^{(2)}|x^{(1)}) \times \ldots \times P(x^{(T)}|x^{(T-1)}, \ldots, x^{(1)})$$

$$= \prod_{t=1}^{T} P(x^{(t)}|x^{(t-1)}, \ldots, x^{(1)})$$

What the LM provides

Slides credit: Abigail See

Evaluating Language Models

- The standard **evaluation metric** for Language Models is **perplexity**

$$\text{perplexity} = \prod_{t=1}^{T} \left(\frac{1}{P_{\text{LM}}(\boldsymbol{x}^{(t+1)} | \ \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})} \right)^{1/T}$$

Normalized by number of words

Inverse probability of corpus, according to Language Model

- This is equal to the exponential of the cross-entropy loss:

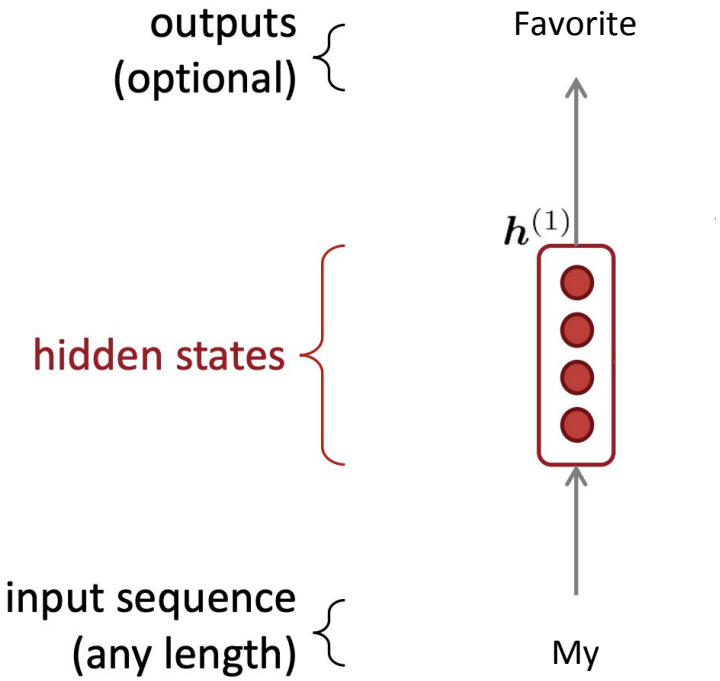
$$J(\theta) = \prod_{t=1}^{T} \left(\frac{1}{\hat{\boldsymbol{y}}^{(t)}_{\boldsymbol{x}_{t+1}}} \right)^{1/T} = \exp\left(\frac{1}{T} \sum_{t=1}^{T} -\log \hat{\boldsymbol{y}}^{(t)}_{\boldsymbol{x}_{t+1}} \right) = \exp(J(\theta))$$

Lower perplexity is better!

Slides credit: Abigail See

Language Model General Architecture

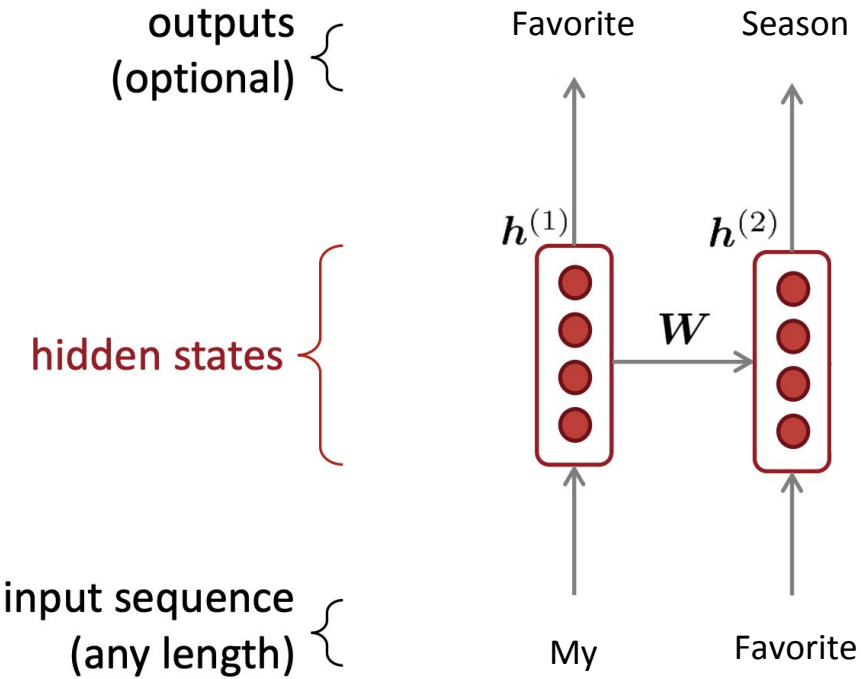
- RNN that reads one token (in this case words) at a time and predicts the next



Slides credit: Abigail See

Language Model General Architecture

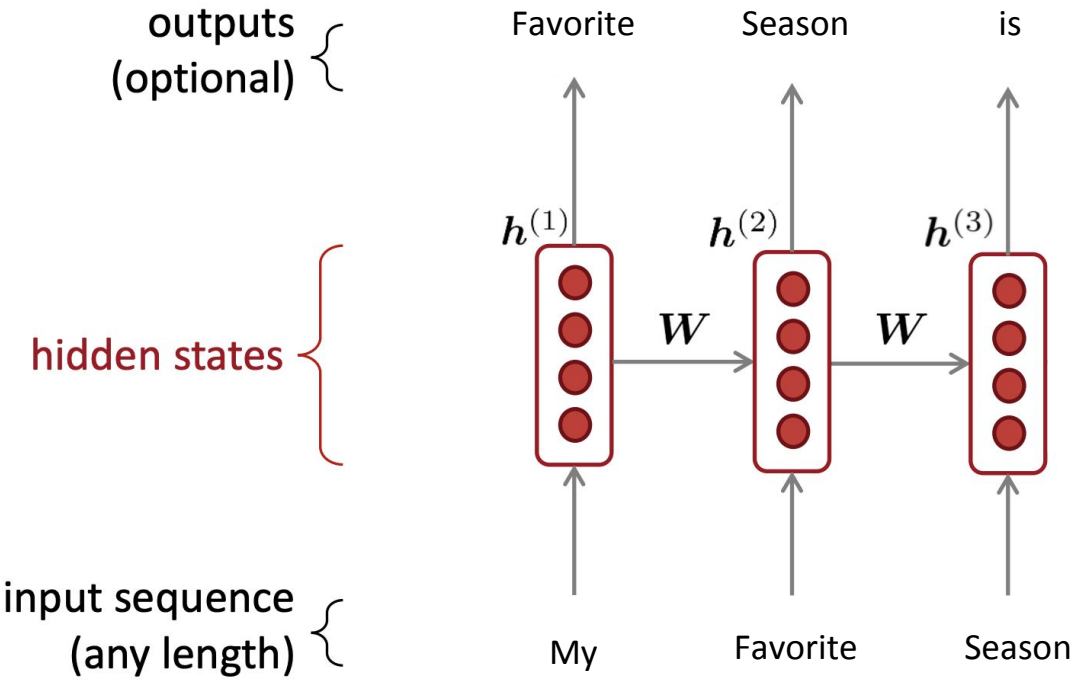
- RNN that reads one token (in this case words) at a time and predicts the next



Slides credit: Abigail See

Language Model General Architecture

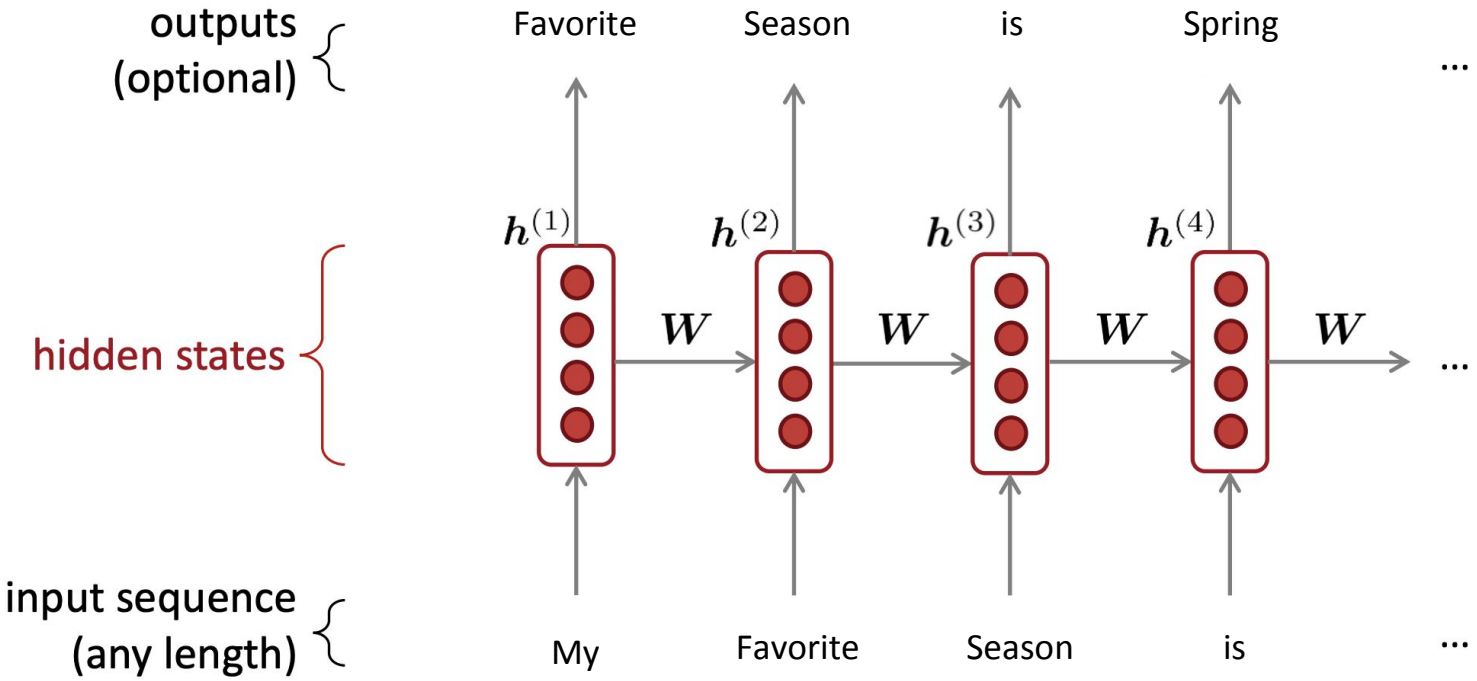
- RNN that reads one token (in this case words) at a time and predicts the next



Slides credit: Abigail See

Language Model General Architecture

- RNN that reads one token (in this case words) at a time and predicts the next



Slides credit: Abigail See

RNNs Perplexity

- RNNs greatly improve perplexity

n-gram model →

Increasingly complex RNNs ↓

Model	Perplexity
Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6
RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013)	51.3
RNN-2048 + BlackOut sampling (Ji et al., 2015)	68.3
Sparse Non-negative Matrix factorization (Shazeer et al., 2015)	52.9
LSTM-2048 (Jozefowicz et al., 2016)	43.7
2-layer LSTM-8192 (Jozefowicz et al., 2016)	30
Ours small (LSTM-2048)	43.9
Ours large (2-layer LSTM-2048)	39.8

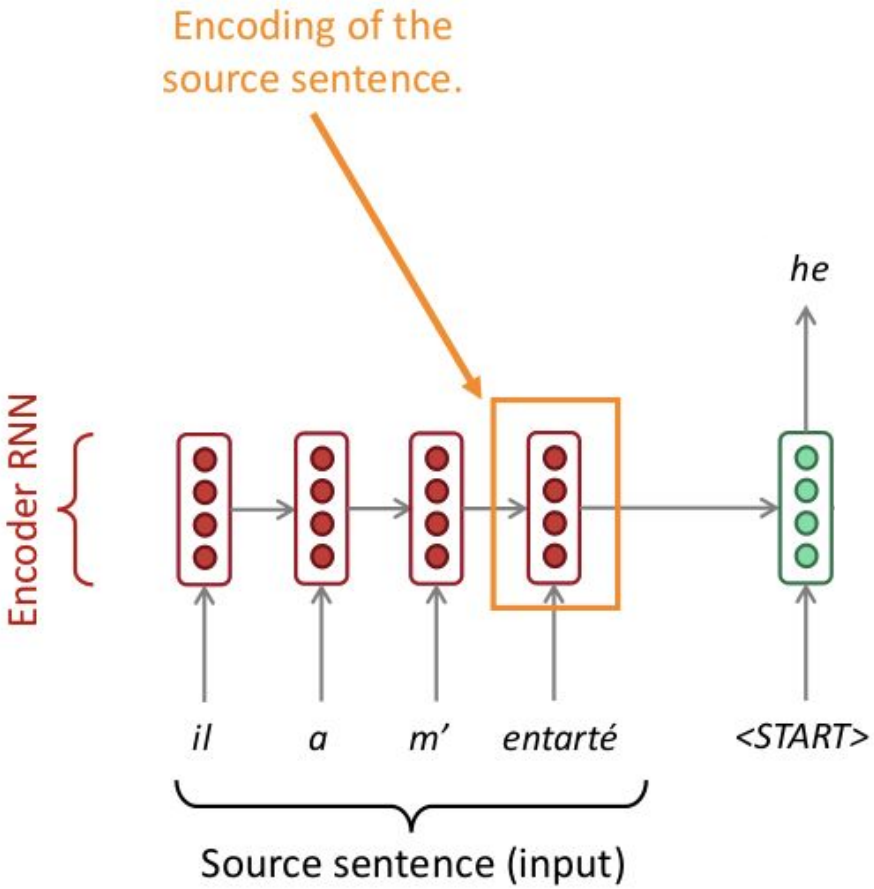
↓ Perplexity improves (lower is better)

Source: https://research.fb.com/building-an-efficient-neural-language-model-over-a-billion-words/

Outline

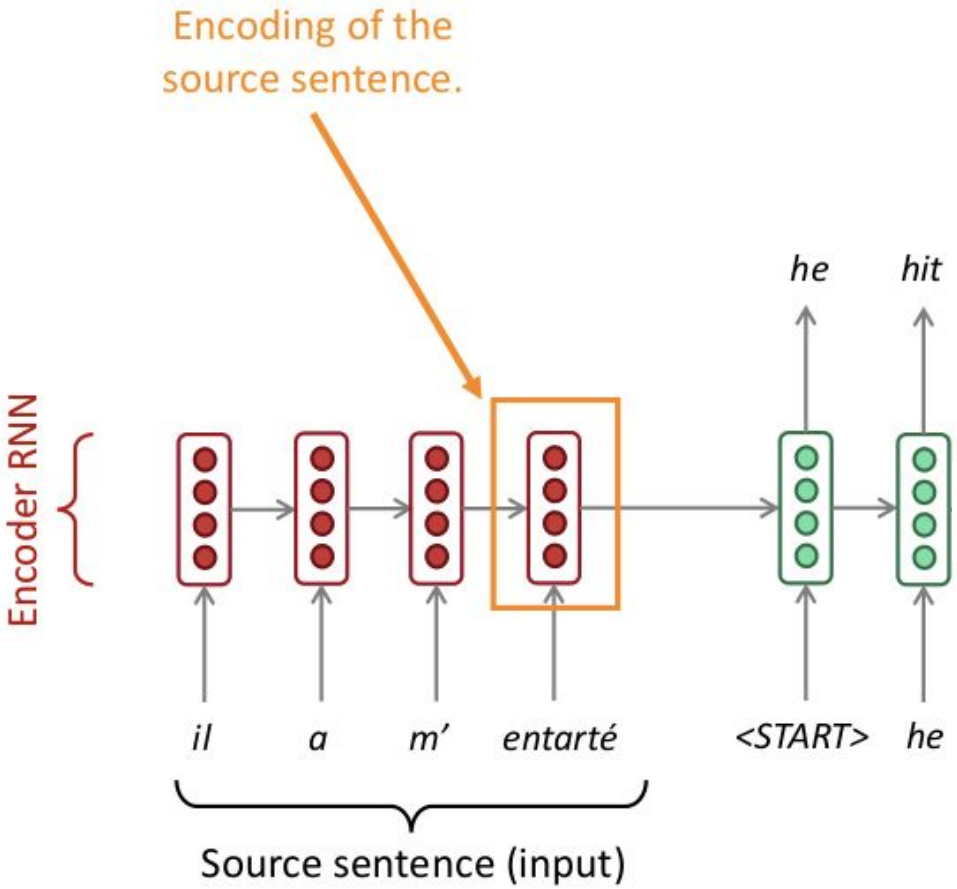
- Application: Text Classification
- Application: Language Modeling
- **Application: Machine Translation (Seq2Seq)**
  - Machine Translation with Attention
- Application: Question Answering
- Advanced Attention mechanism: Transformers
- Pre-training/Transfer Learning

Seq2seq: Neural Machine Translation



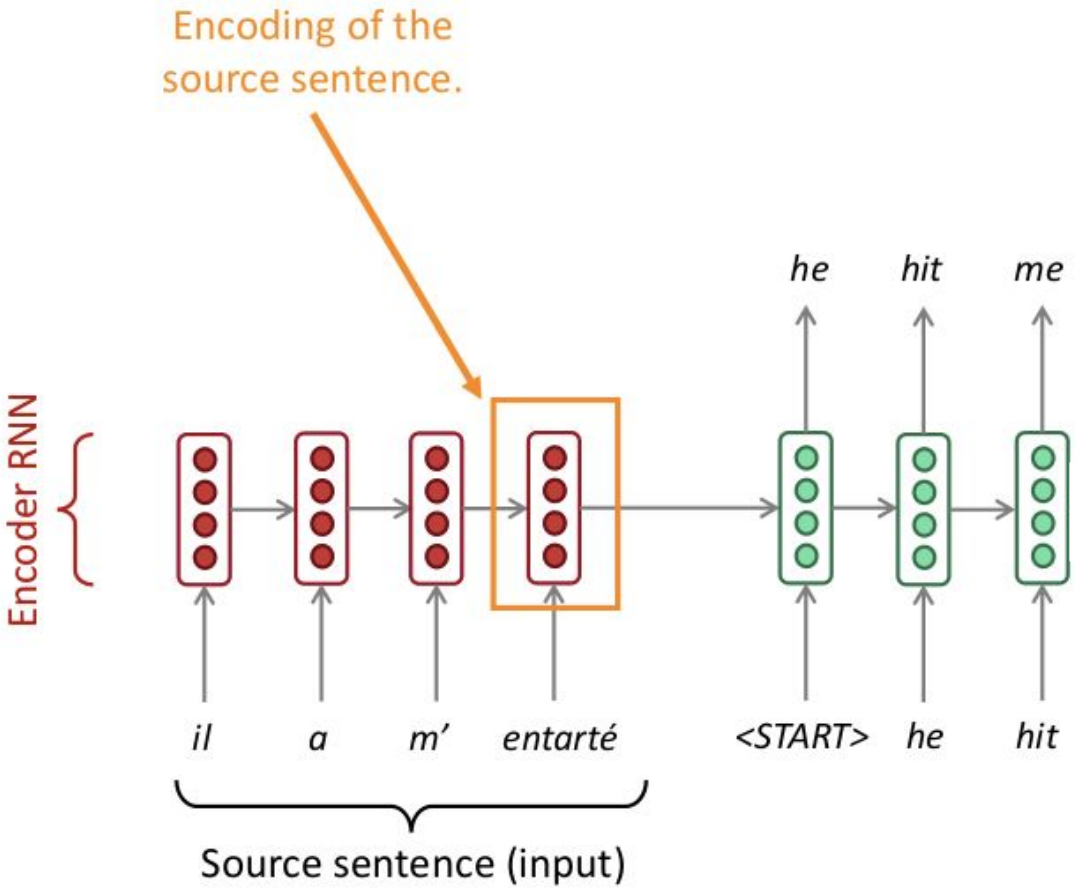
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



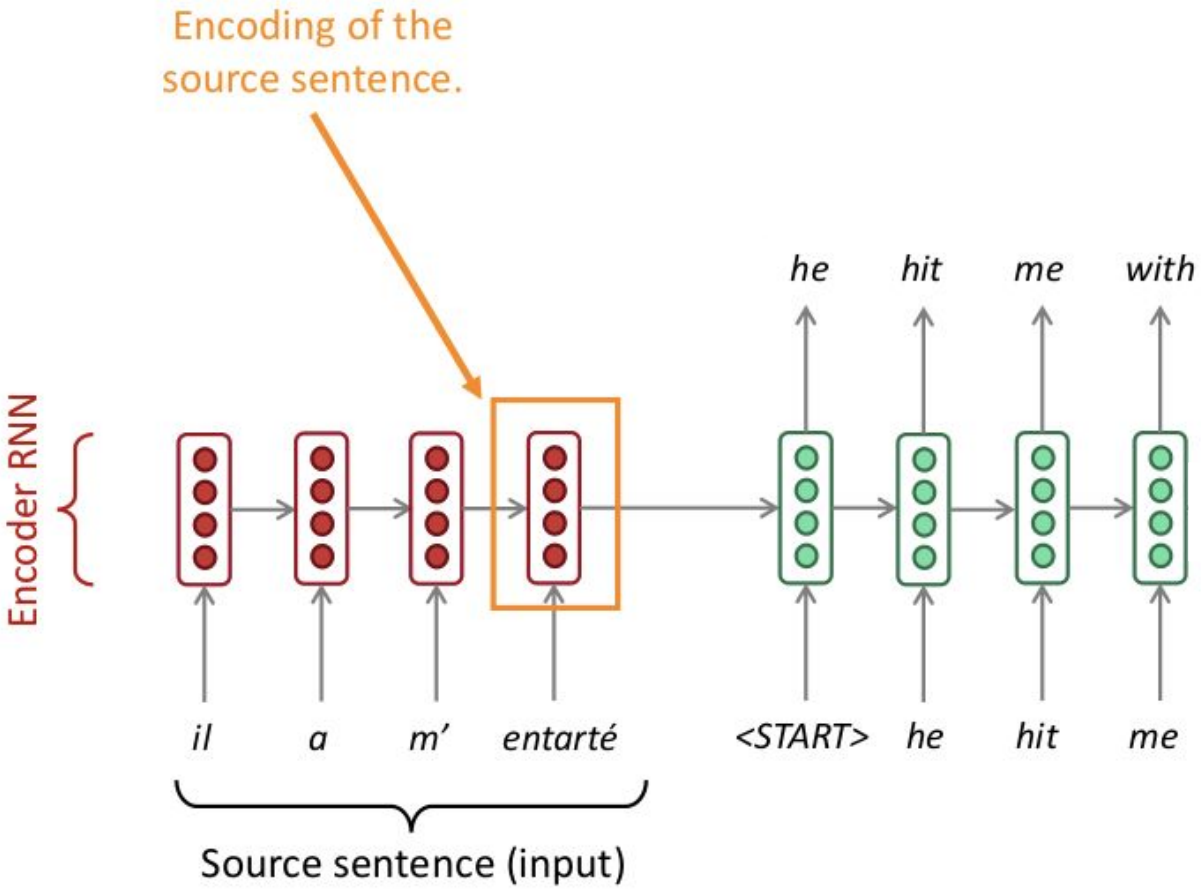
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



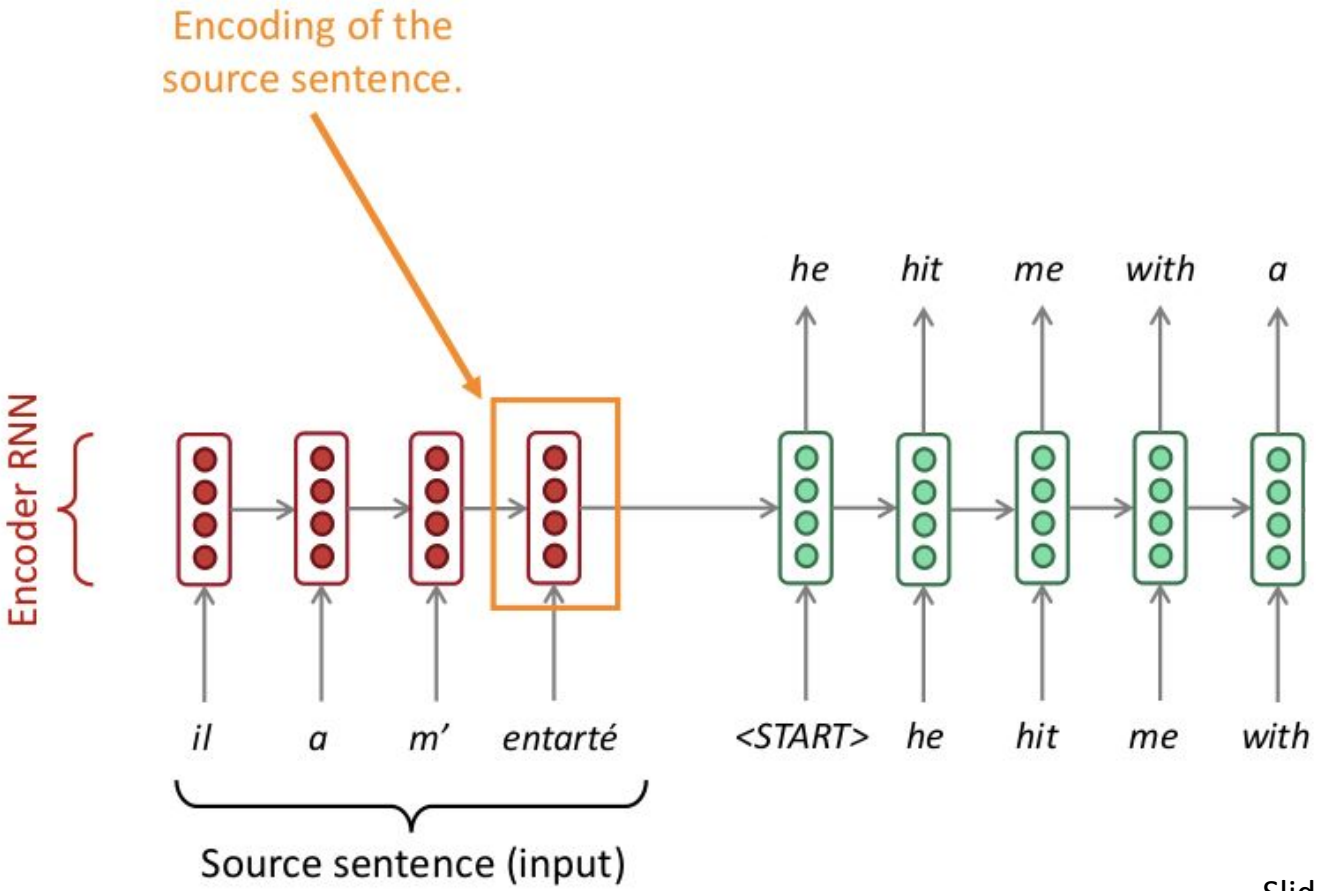
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



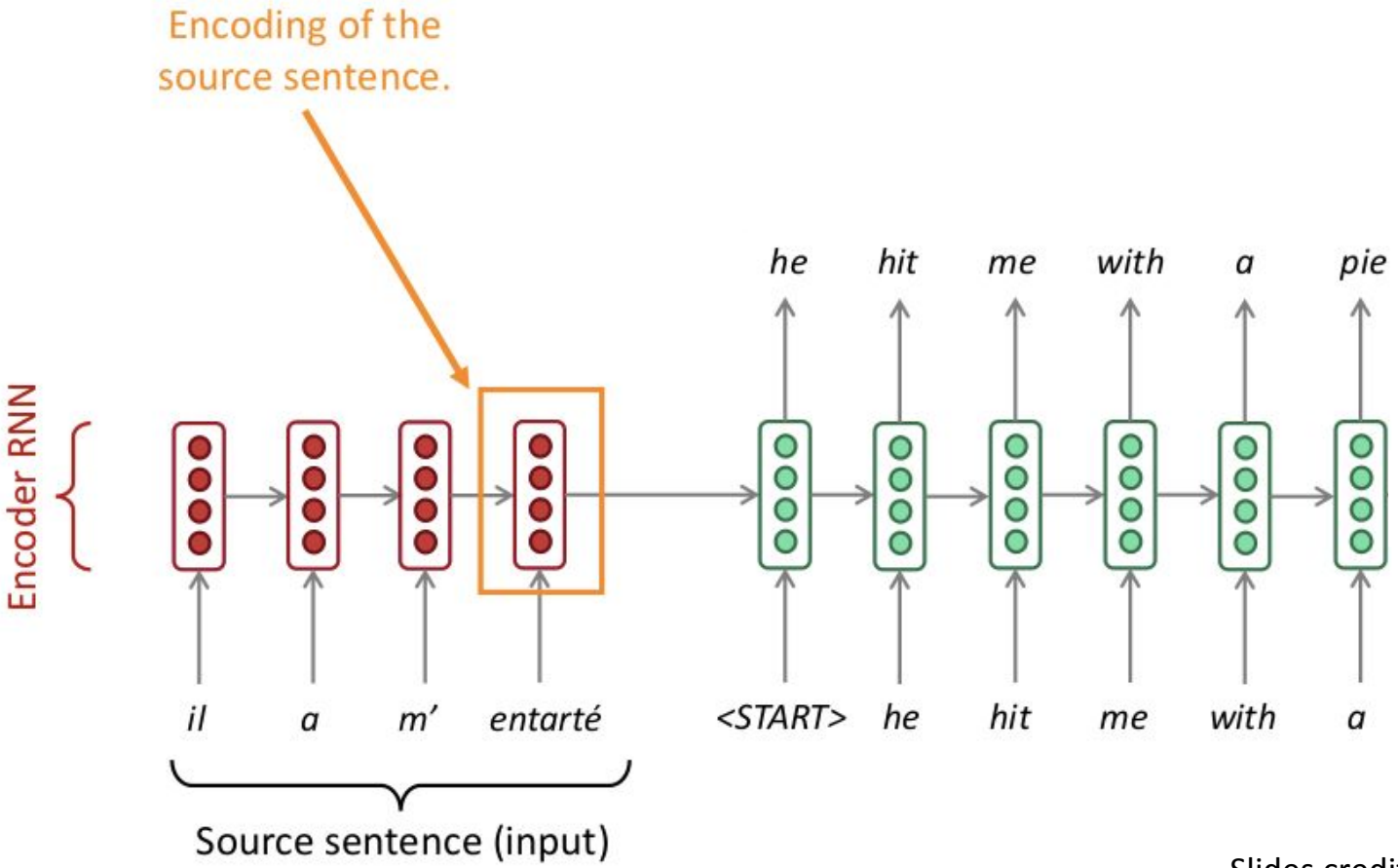
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



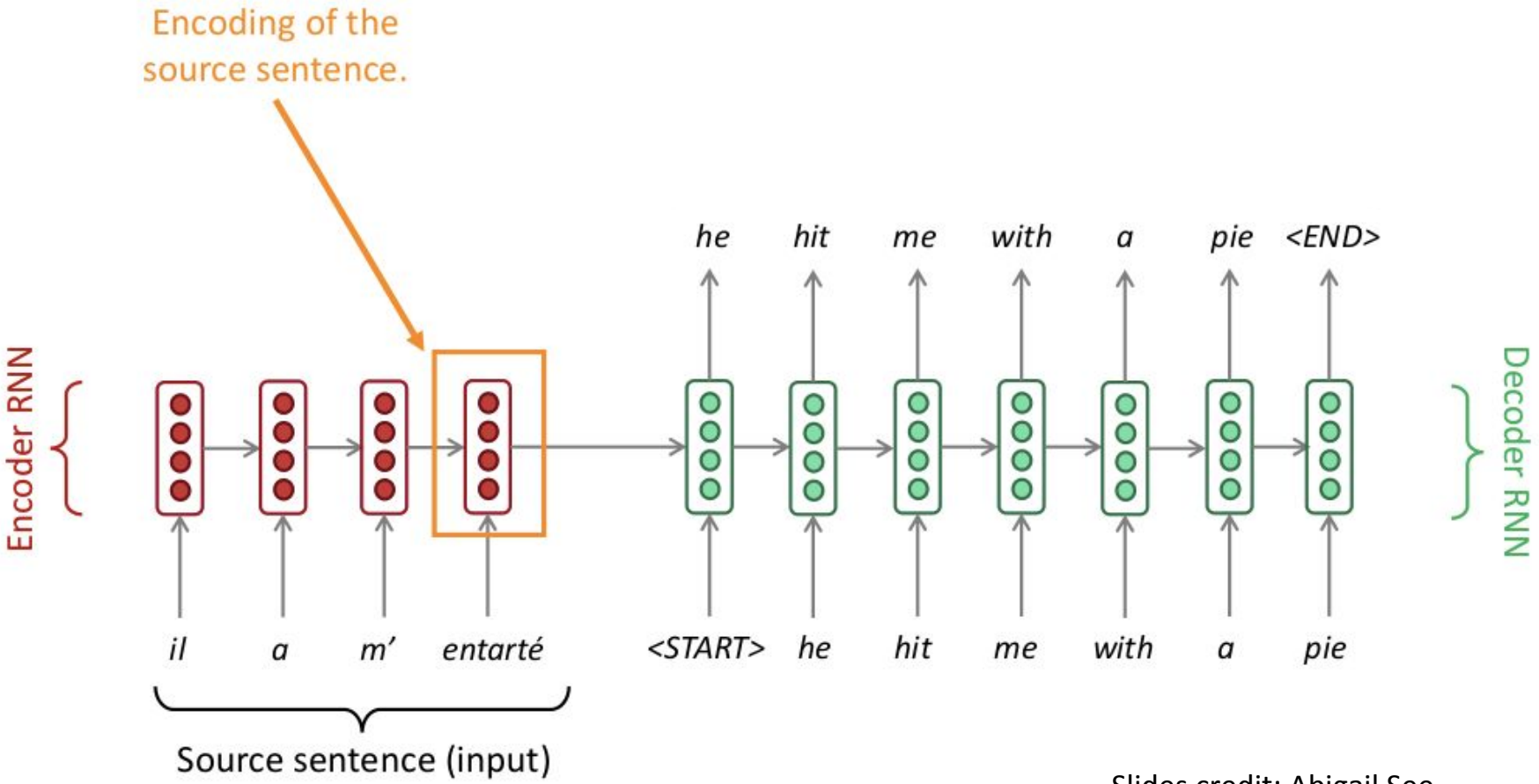
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



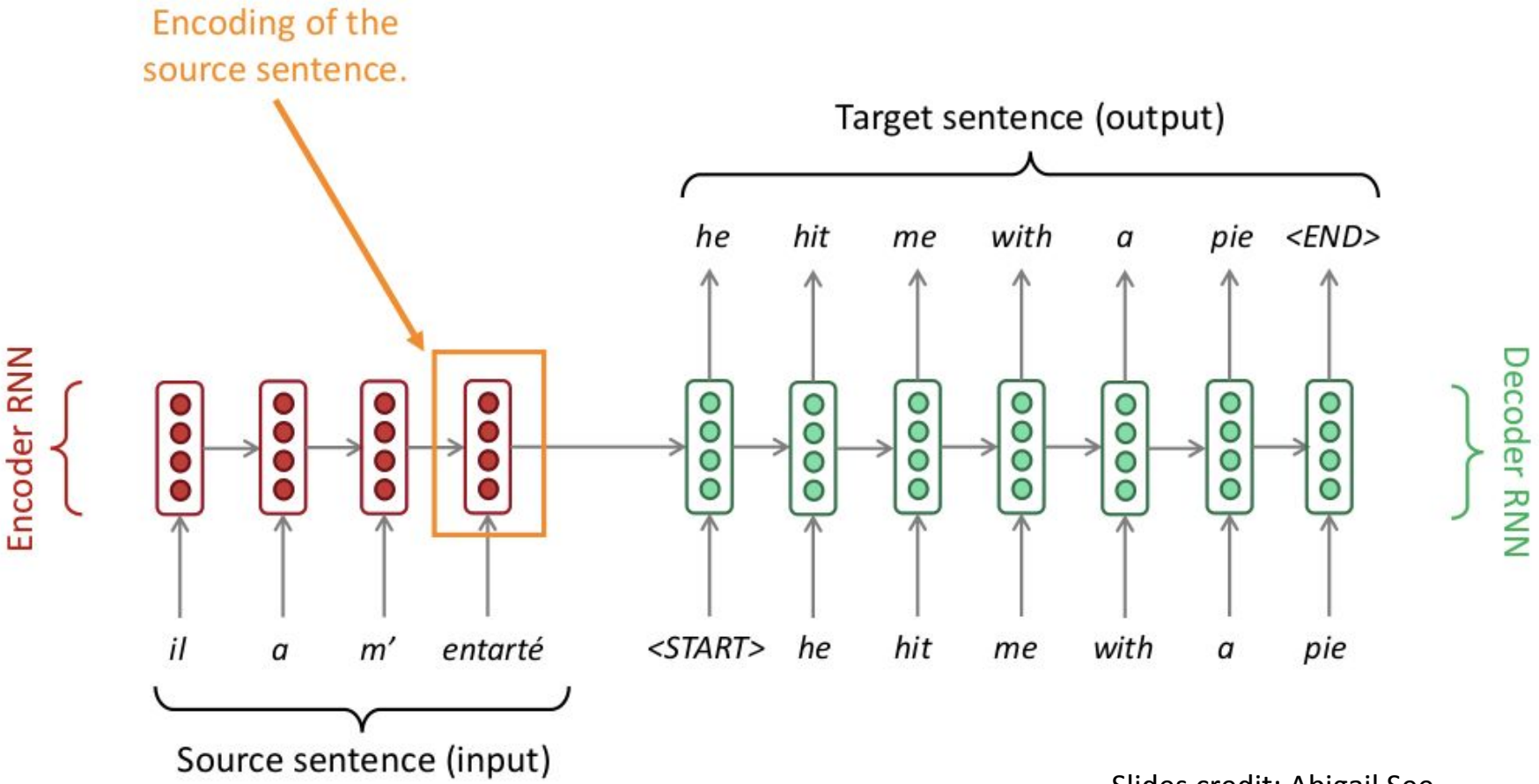
Slides credit: Abigail See

Seq2seq: Neural Machine Translation



Slides credit: Abigail See

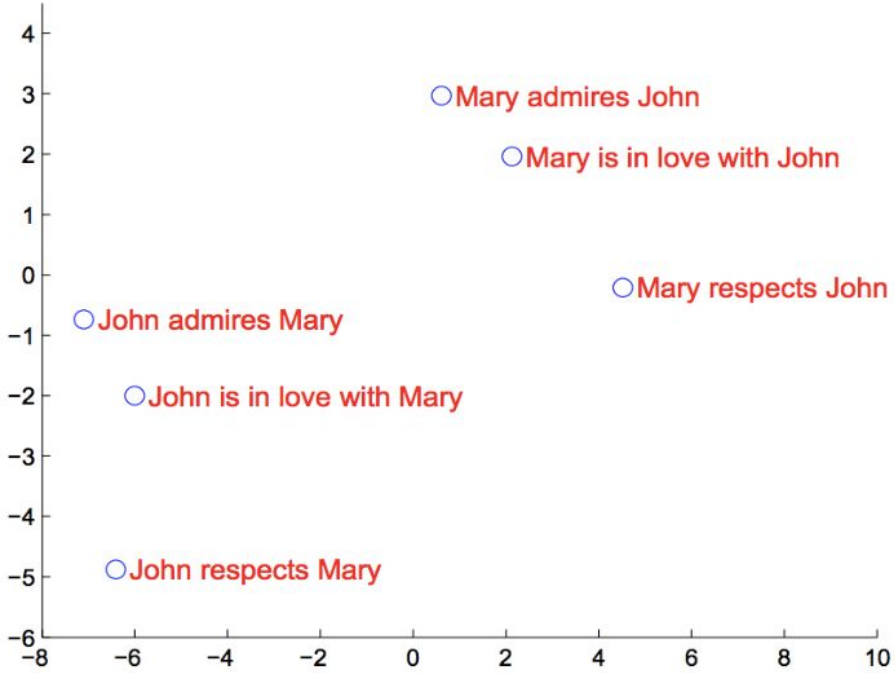
Seq2seq: Neural Machine Translation



Slides credit: Abigail See

Seq2seq: Neural Machine Translation

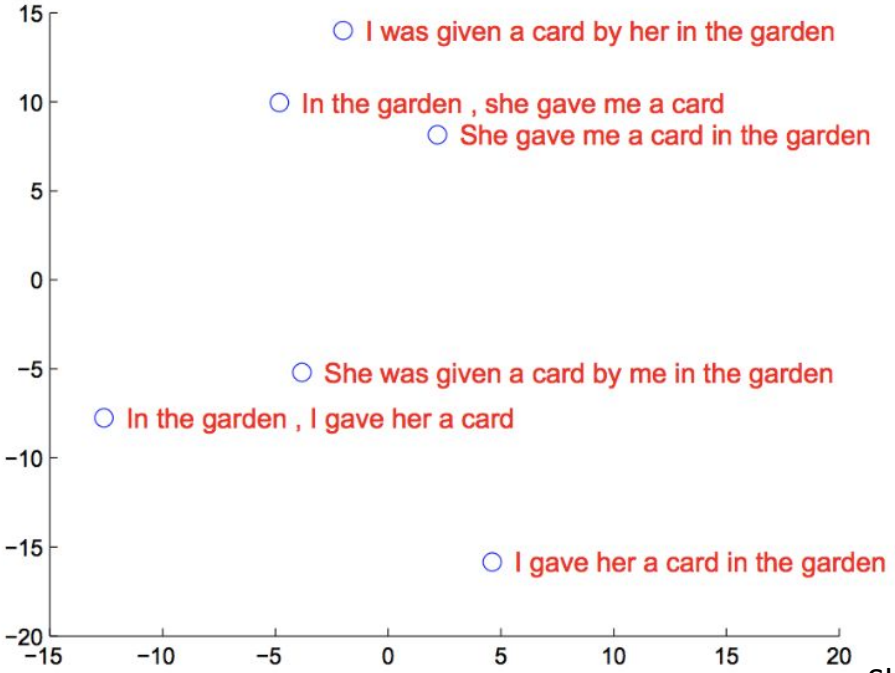
- Representation learning



Slides credit: Ilya Sutskever

Seq2seq: Neural Machine Translation

- Representation learning



Slides credit: Ilya Sutskever

Seq2seq: Neural Machine Translation

- **French-to-English:**
  - **French:** Les avionneurs se querellent au sujet de la largeur des sièges alors que de grosses commandes sont en jeu.
  - **NMT:** Aircraft manufacturers are concerned about the width of seats while large orders are at stake.
  - **TRUTH:** Jet makers feud over seat width with big orders at stake.

Slides credit: Ilya Sutskever

Seq2seq: Neural Machine Translation

- **French-to-English:**
  - **French:** La dispute fait rage entre les grands constructeurs aéronautiques à propos de la largeur des sièges de la classe touriste sur les vols long courriers , ouvrant la voie à une confrontation amère lors du salon aéronautique de Dubaï qui a lieu de mois-ci.
  - **NMT:** The dispute is raging between large aircraft manufacturers on the size of the tourist seats on the long-haul flights , leading to a bitter confrontation at the Dubai Airshow in the month of October.
  - **TRUTH:** A row has flared up between leading plane makers over the width of tourist-class seats on long-distance flights , setting the tone for a bitter confrontation at this month's Dubai Airshow.

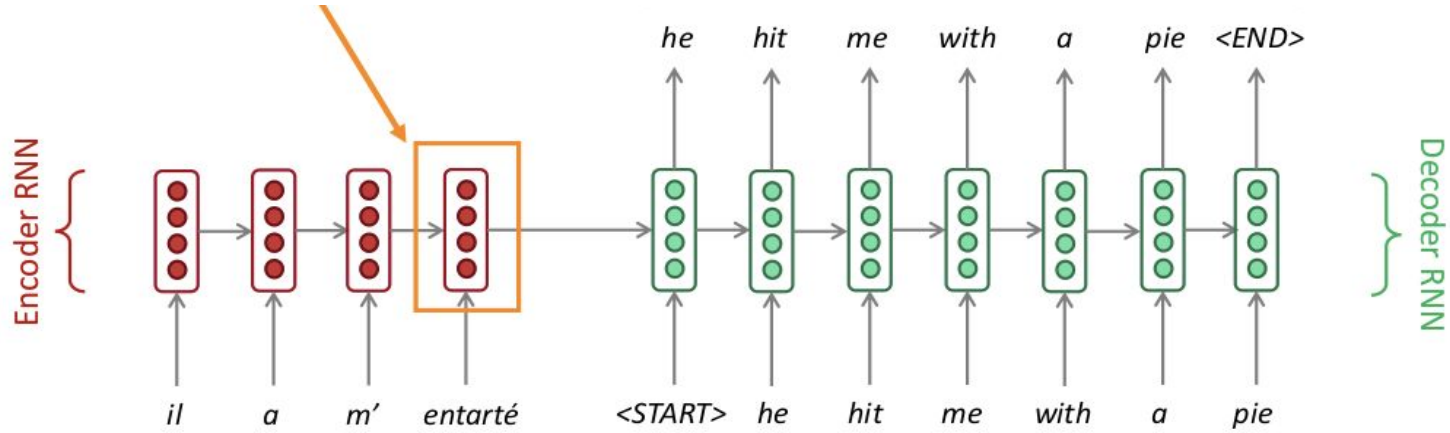
Slides credit: Ilya Sutskever

Outline

- Application: Text Classification
- Application: Language Modeling
- **Application: Machine Translation (Seq2Seq)**
  - **Machine Translation with Attention**
- Application: Question Answering
- Advanced Attention mechanism: Transformers
- Pre-training/Transfer Learning

Seq2seq - The issue with long inputs

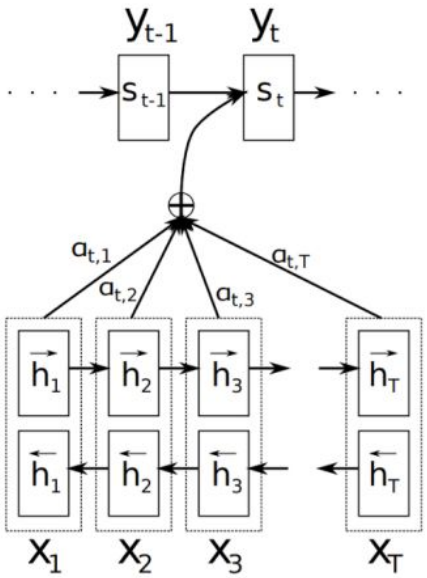
- Same embedding informs the entire output
- Needs to capture all the information about the input regardless of its length

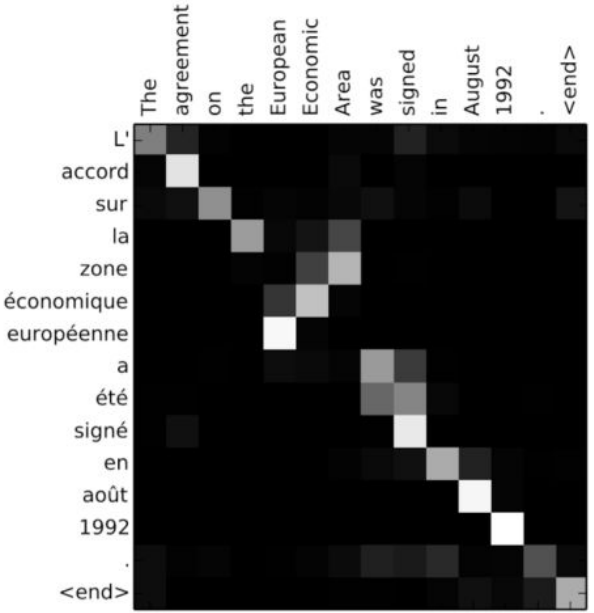


- Is there a better way to pass the information from encoder to the decoder?

Credit slides: Abigail See

Seq2seq with Attention

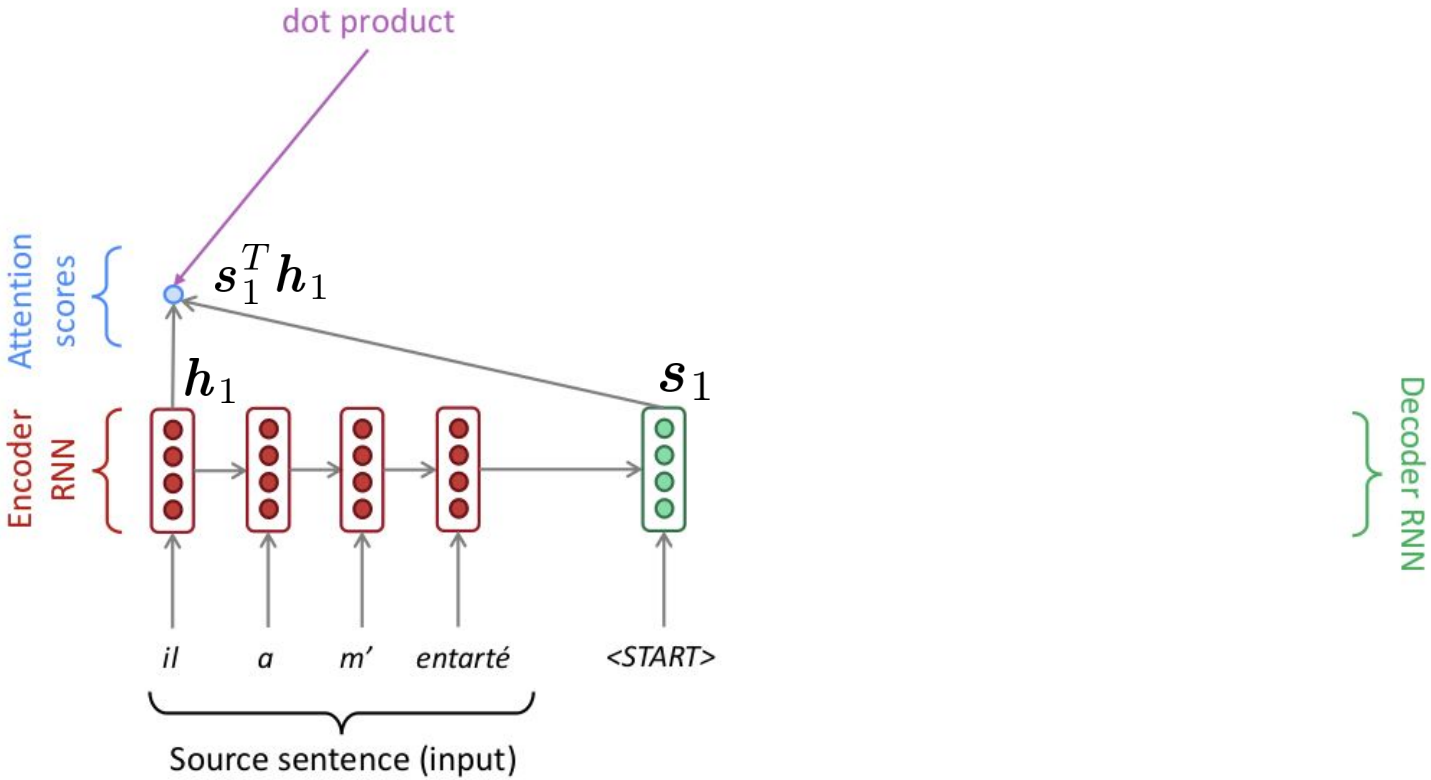




Bahdanau, D., et al. "Neural Machine Translation by Jointly Learning to Align and Translate." *ICLR (2015)*

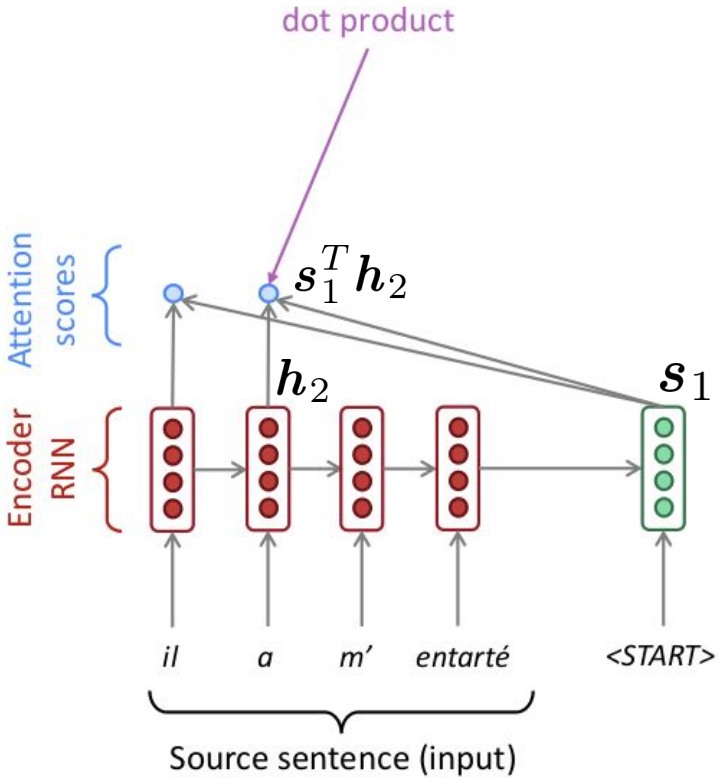
Credit slides: Russ Salakhutdinov

Seq2seq with Attention



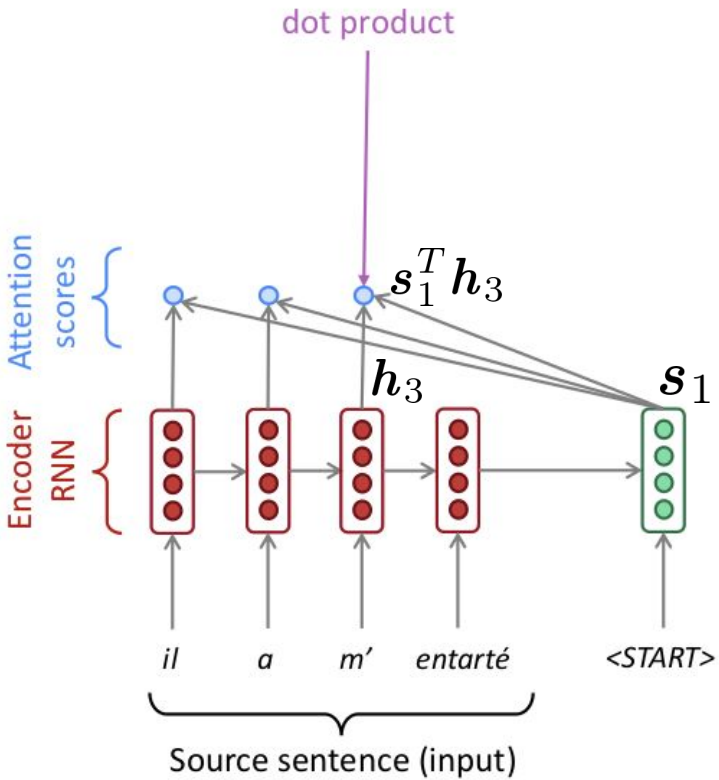
Credit slides: Abigail See

Seq2seq with Attention



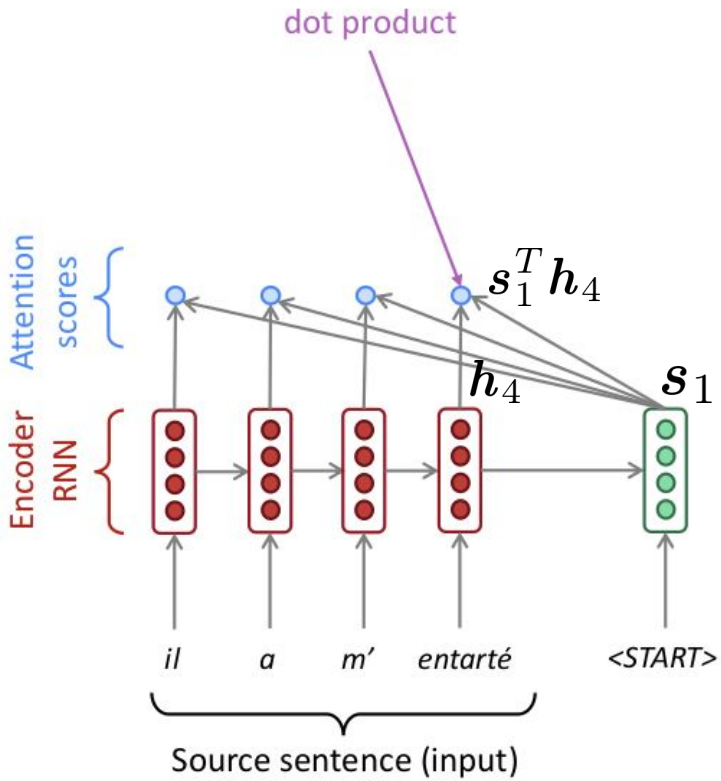
Credit slides: Abigail See

Seq2seq with Attention



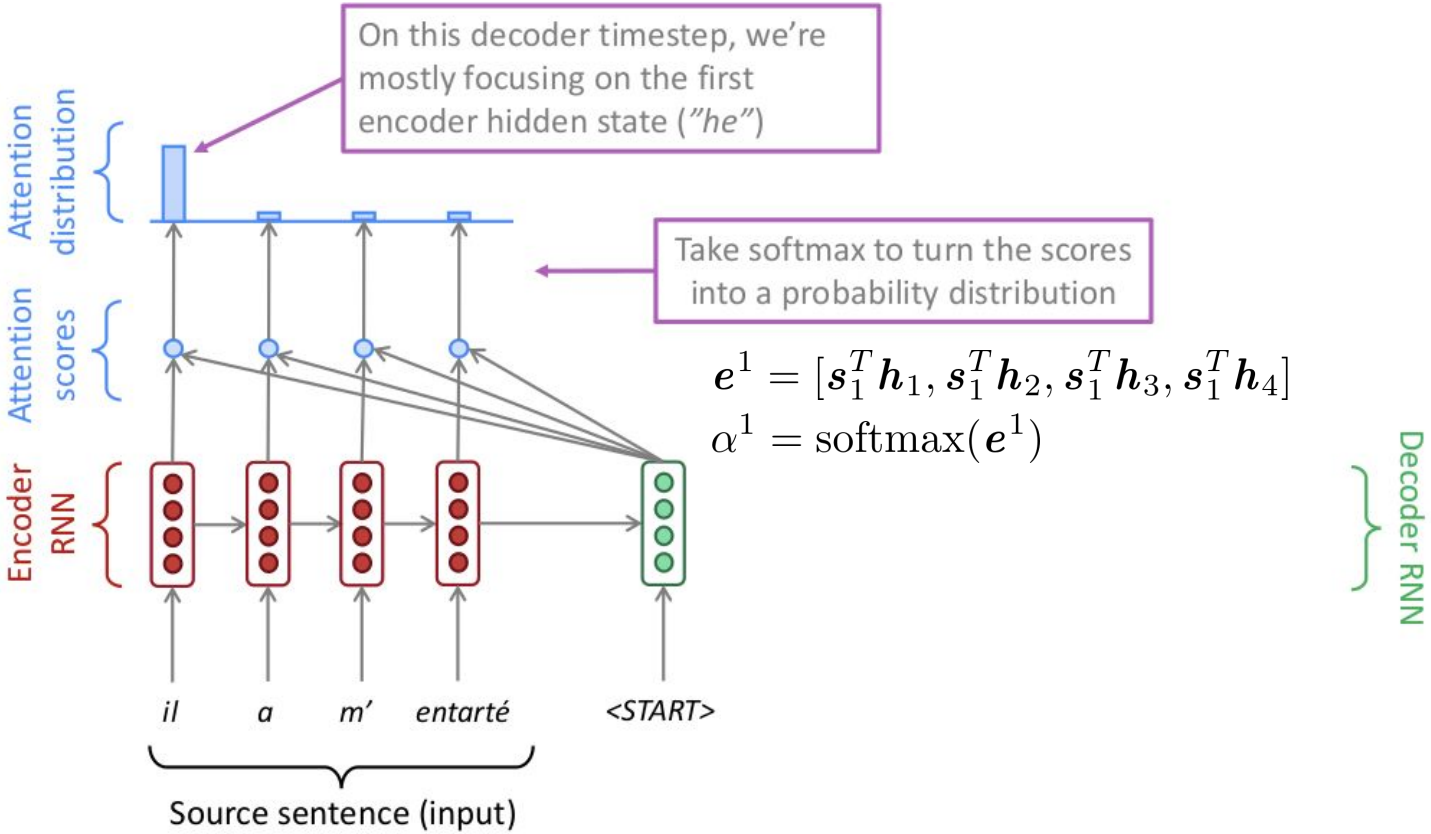
Credit slides: Abigail See

Seq2seq with Attention



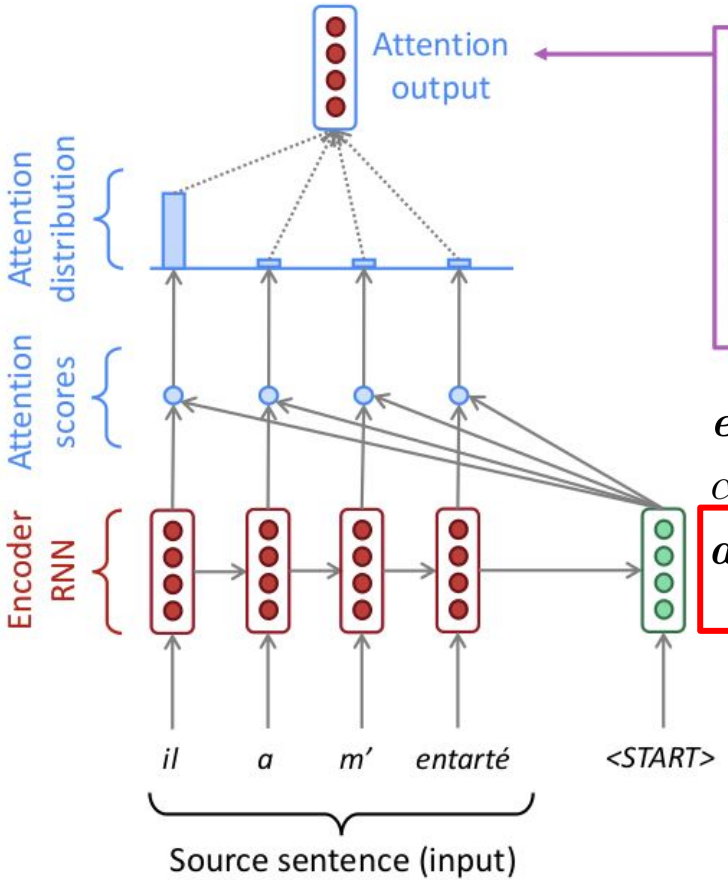
Credit slides: Abigail See

Seq2seq with Attention



Credit slides: Abigail See

Seq2seq with Attention



Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information from the hidden states that received high attention.

$$\boldsymbol{e}^1 = [\boldsymbol{s}_1^T \boldsymbol{h}_1, \boldsymbol{s}_1^T \boldsymbol{h}_2, \boldsymbol{s}_1^T \boldsymbol{h}_3, \boldsymbol{s}_1^T \boldsymbol{h}_4]$$

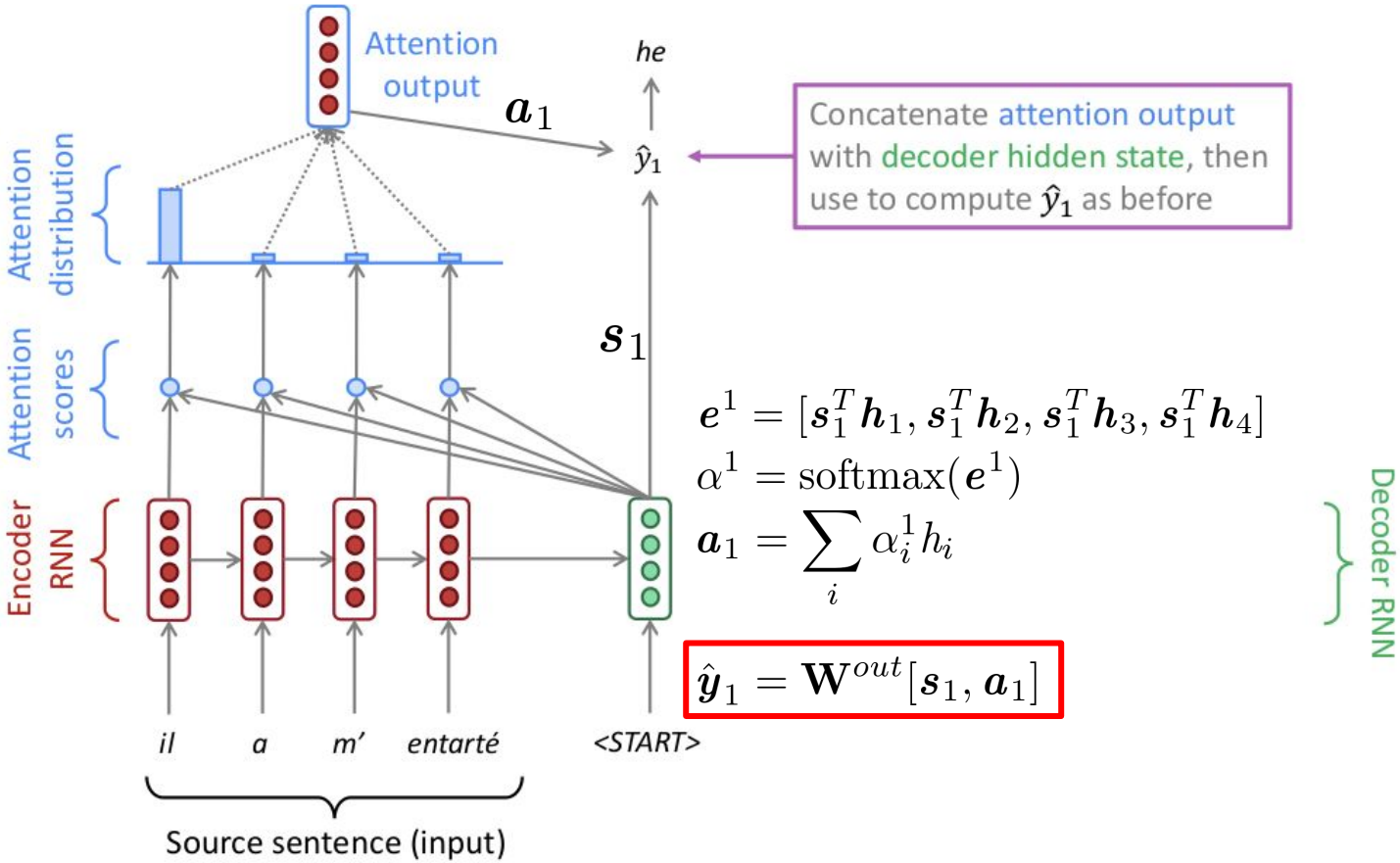
$$\alpha^1 = \text{softmax}(\boldsymbol{e}^1)$$

$$\boldsymbol{a}_1 = \sum_i \alpha_i^1 h_i$$

Decoder RNN

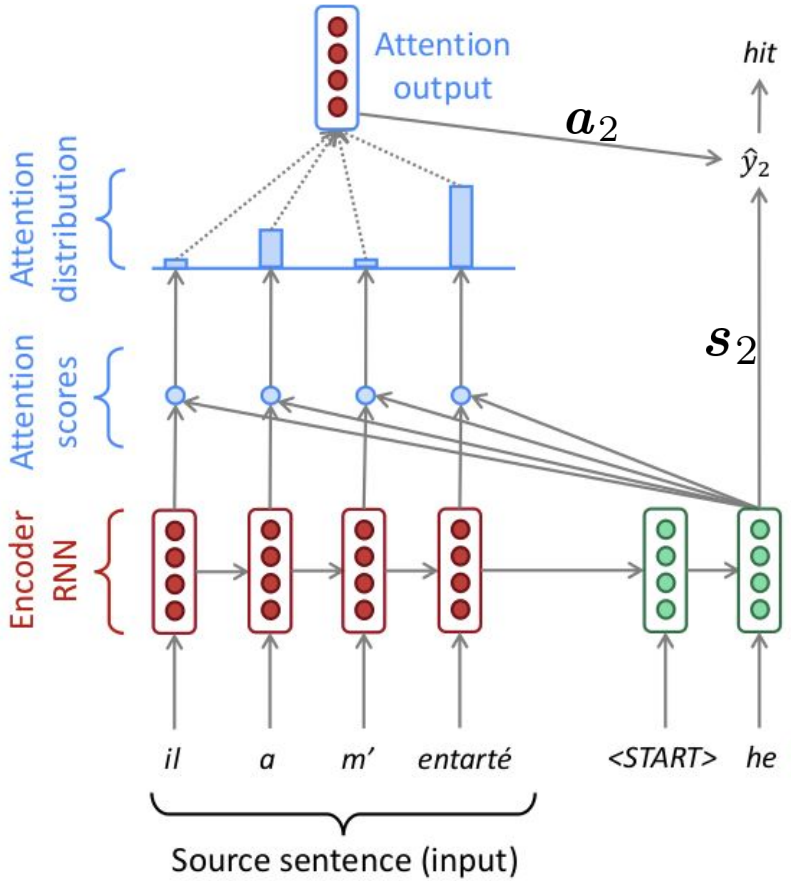
Credit slides: Abigail See

Seq2seq with Attention



Credit slides: Abigail See

Seq2seq with Attention

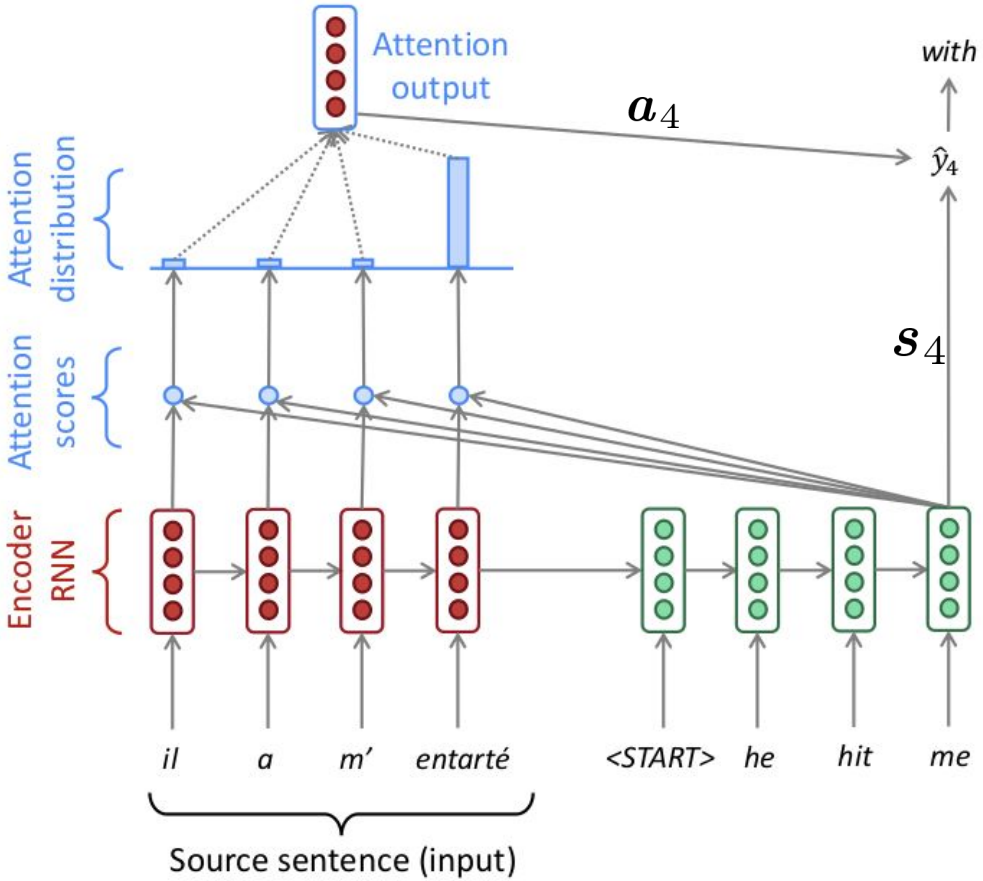


$$e^2 = [s_2^T h_1, s_2^T h_2, s_2^T h_3, s_2^T h_4]$$
$$\alpha^2 = \text{softmax}(e^2)$$
$$a_2 = \sum_i \alpha_i^2 h_i$$

$$\hat{y}_2 = \mathbf{W}^{out}[s_2, a_2]$$

Credit slides: Abigail See

Seq2seq with Attention

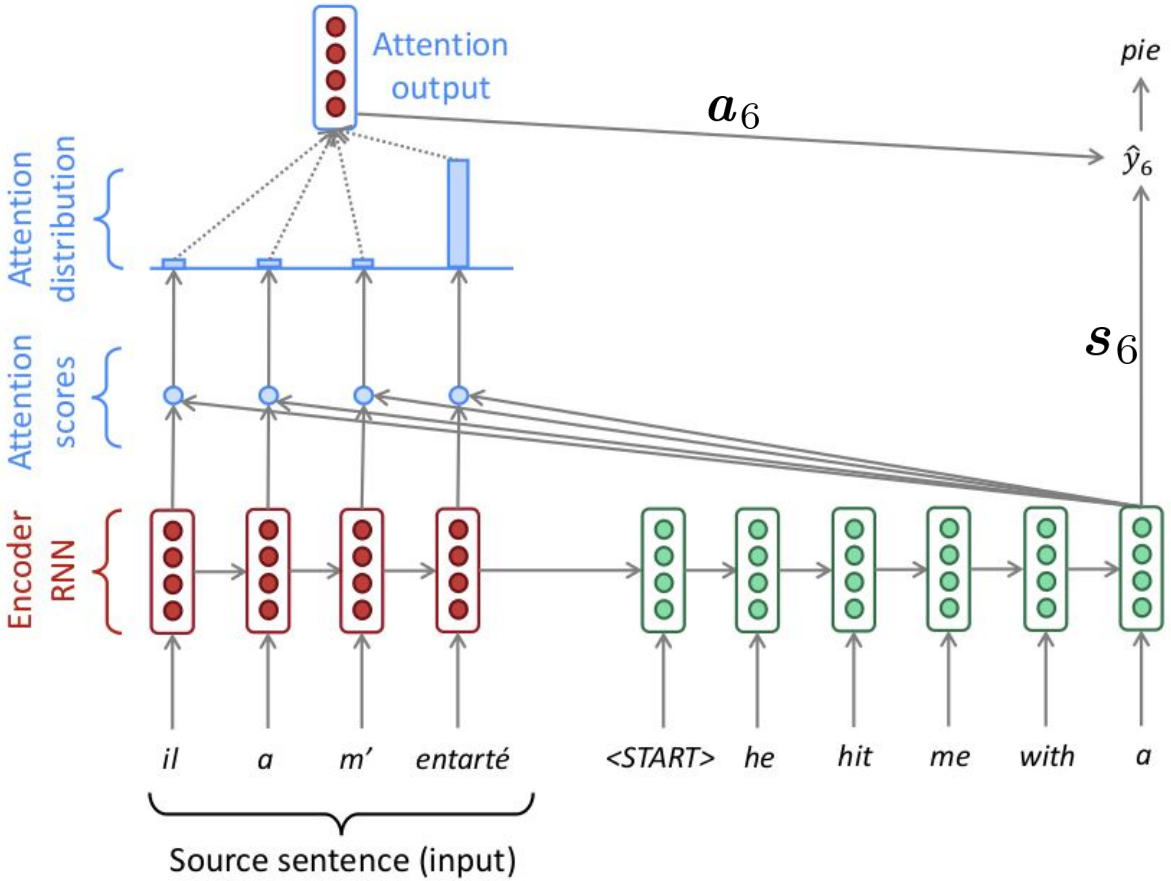


$$e^4 = [s_4^T h_1, s_4^T h_2, s_4^T h_3, s_4^T h_4]$$
$$\alpha^4 = \text{softmax}(e^4)$$
$$a_4 = \sum_i \alpha_i^4 h_i$$

$$\hat{y}_4 = \mathbf{W}^{out}[s_4, a_4]$$

Credit slides: Abigail See

Seq2seq with Attention



$$e^6 = [s_6^T h_1, s_6^T h_2, s_6^T h_3, s_6^T h_4]$$
$$\alpha^6 = \text{softmax}(e^6)$$
$$a_6 = \sum_i \alpha_i^6 h_i$$

$$\hat{y}_6 = \mathbf{W}^{out}[s_6, a_6]$$

Credit slides: Abigail See

Seq2seq with Attention

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

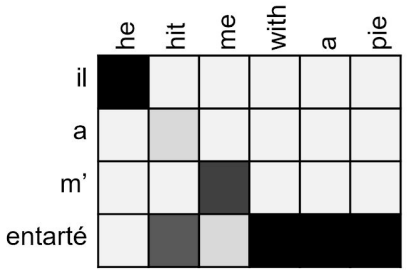
- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

Credit slides: Abigail See

Attention is Great!

- Attention significantly improves NMT performance
  - It's very useful to allow decoder to focus on certain parts of the source
- Attention solves the bottleneck problem
  - Attention allows decoder to look directly at source; bypass bottleneck
- Attention helps with vanishing gradient problem
  - Provides shortcut to faraway states
- Attention provides some interpretability
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get (soft) alignment for free!
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself



Credit slides: Abigail See

Attention is a General Deep Learning Technique

- We've seen that attention is a great way to improve the sequence-to-sequence model for Machine Translation.
- However: You can use attention in many architectures (not just seq2seq) and many tasks (not just MT)

- **More general definition of attention**:
  - Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.

- We sometimes say that the query *attends to* the values.
- For example, in the seq2seq + attention model, each decoder hidden state (query) *attends to* all the encoder hidden states (values).

Credit slides: Abigail See

Attention is a General Deep Learning Technique

More general definition of attention:

Given a set of vector *values*, and a vector *query*, **attention** is a technique to compute a weighted sum of the values, dependent on the query.

Intuition:

- The weighted sum is a *selective summary* of the information contained in the values, where the query determines which values to focus on.
- Attention is a way to obtain a *fixed-size representation of an arbitrary set of representations* (the values), dependent on some other representation (the query).

Credit slides: Abigail See

There are several attention variants

- We have some *values* $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and a *query* $\boldsymbol{s} \in \mathbb{R}^{d_2}$

- Attention always involves:
  1. Computing the *attention scores* $\boldsymbol{e} \in \mathbb{R}^N$
  2. Taking softmax to get *attention distribution* α:

$$\alpha = \text{softmax}(\boldsymbol{e}) \in \mathbb{R}^N$$

  3. Using attention distribution to take weighted sum of values:

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1}$$

  thus obtaining the *attention output* $\boldsymbol{a}$ (sometimes called the *context vector*)

There are multiple ways to do this

Credit slides: Abigail See

There are several attention variants

There are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and $\boldsymbol{s} \in \mathbb{R}^{d_2}$:

- Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
  - Note: this assumes $d_1 = d_2$
  - This is the version we saw earlier
- Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
  - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
- Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
  - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a weight vector.
  - d_3 (the attention dimensionality) is a hyperparameter

"Deep Learning for NLP Best Practices", Ruder, 2017. http://ruder.io/deep-learning-nlp-best-practices/index.html#attention
"Massive Exploration of Neural Machine Translation Architectures", Britz et al, 2017, https://arxiv.org/pdf/1703.03906.pdf

Credit slides: Abigail See

Outline

- Application: Text Classification
- Application: Language Modeling
- Application: Machine Translation (Seq2Seq)
  - Machine Translation with Attention
- **Application: Question Answering**
- Advanced Attention mechanism: Transformers
- Pre-training/Transfer Learning

Machine Comprehension

- "A machine **comprehends** a passage of **text** if, for any **question** regarding that text that can be **answered** correctly by a majority of native speakers, that machine can provide a string which those speakers would agree both answers that question, and does not contain information irrelevant to that question."

Towards the Machine Comprehension of Text: An Essay

Christopher J.C. Burges
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA

December 23, 2013

Slides credit: Christopher Manning

Reading Comprehension

Passage (*P*) + Question (*Q*) ⟶ Answer (*A*)

P

Alyssa got to the beach after a long trip. She's from Charlotte. She traveled from Atlanta. She's now in Miami. She went to Miami to visit some friends. But she wanted some time to herself at the beach, so she went there first. After going swimming and laying out, she went to her friend Ellen's house. Ellen greeted Alyssa and they both had some lemonade to drink. Alyssa called her friends Kristin and Rachel to meet at Ellen's house…….

Q Why did Alyssa go to Miami?

A To visit some friends

Slides credit: Christopher Manning

Stanford Question Answering Dataset (SQuAD)

Question: Which team won Super Bowl 50?

Passage

Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California.

100k examples

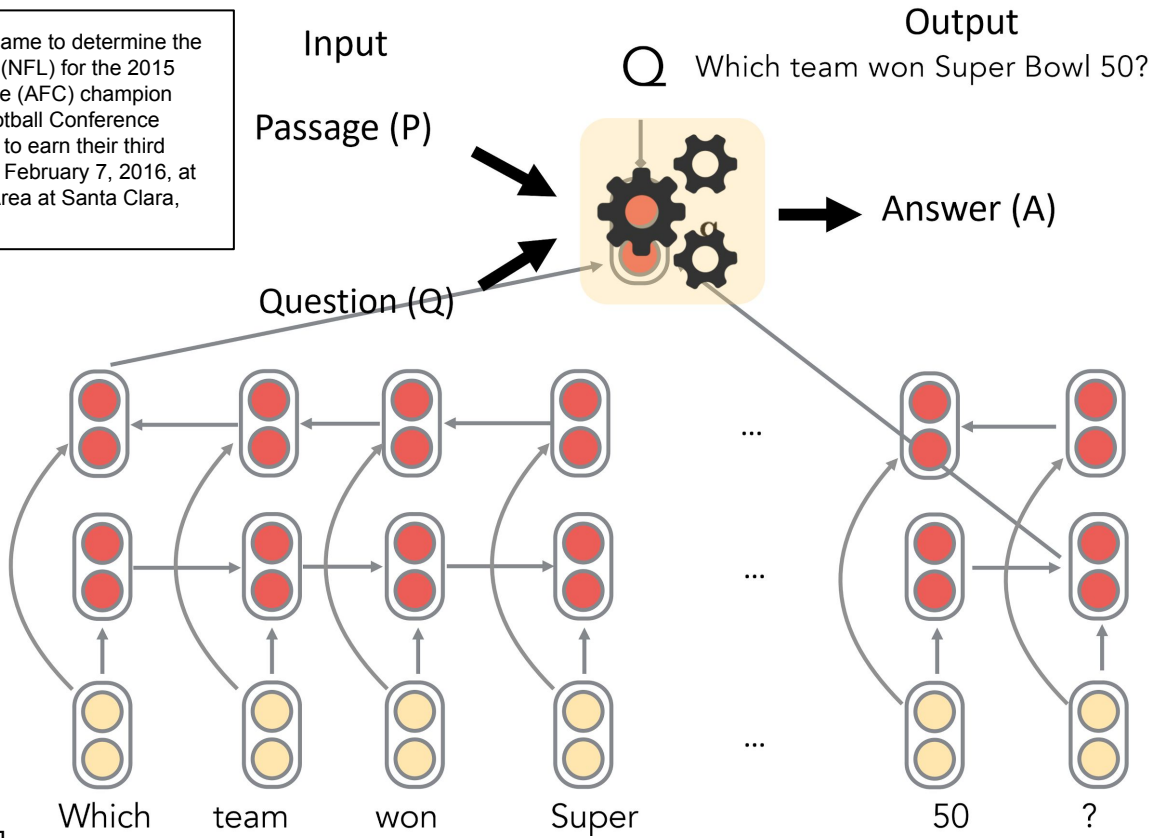
Answer must be a span in the passage

A.k.a. extractive question answering

Slides credit: Christopher Manning

Stanford Attentive Reader

Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California.

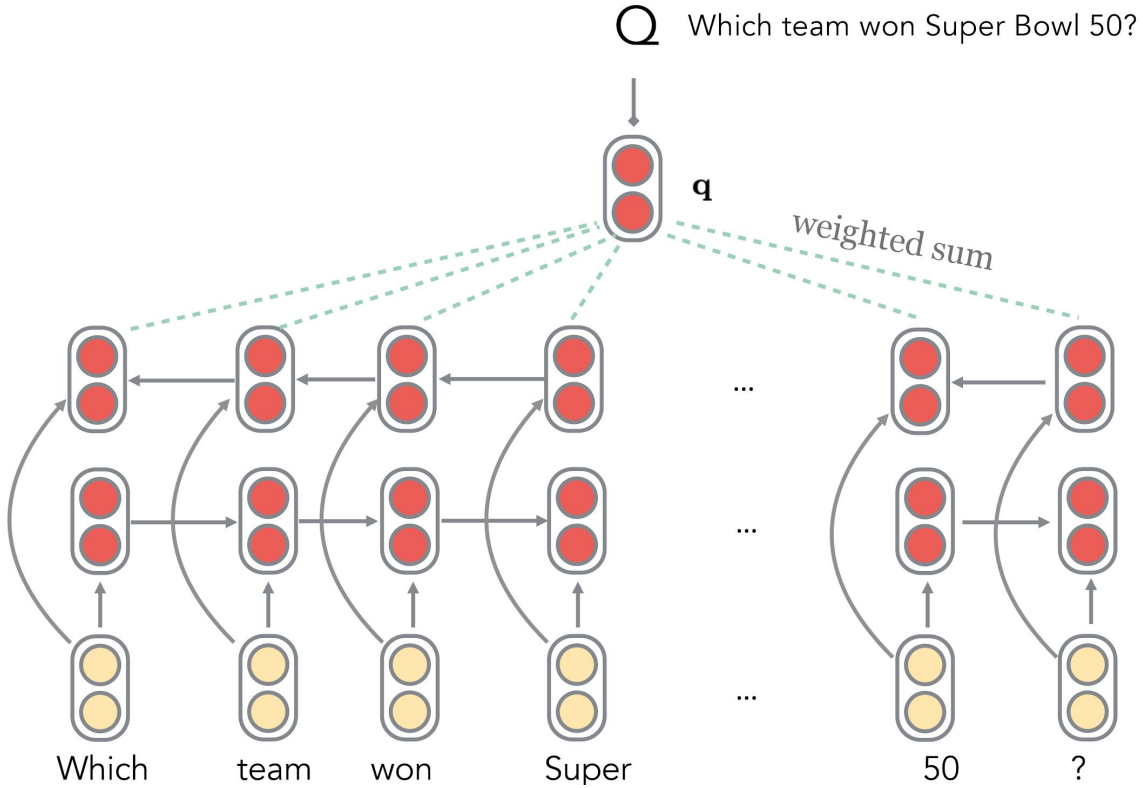


[Chen, Bolton, & Manning 2016]
[Chen, Fisch, Weston & Bordes 2017]
[Chen 2018]

Slides credit: Christopher Manning

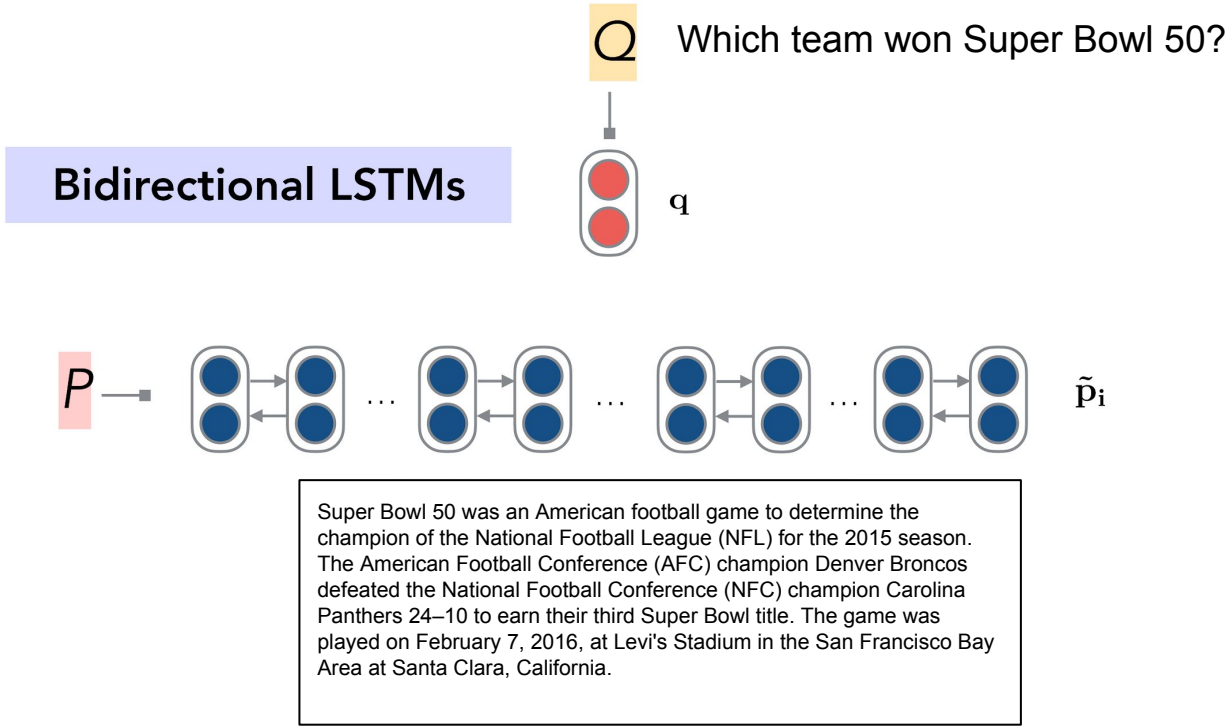
Stanford Attentive Reader

(Chen et. al, 2016; Chen et. al, 2017)



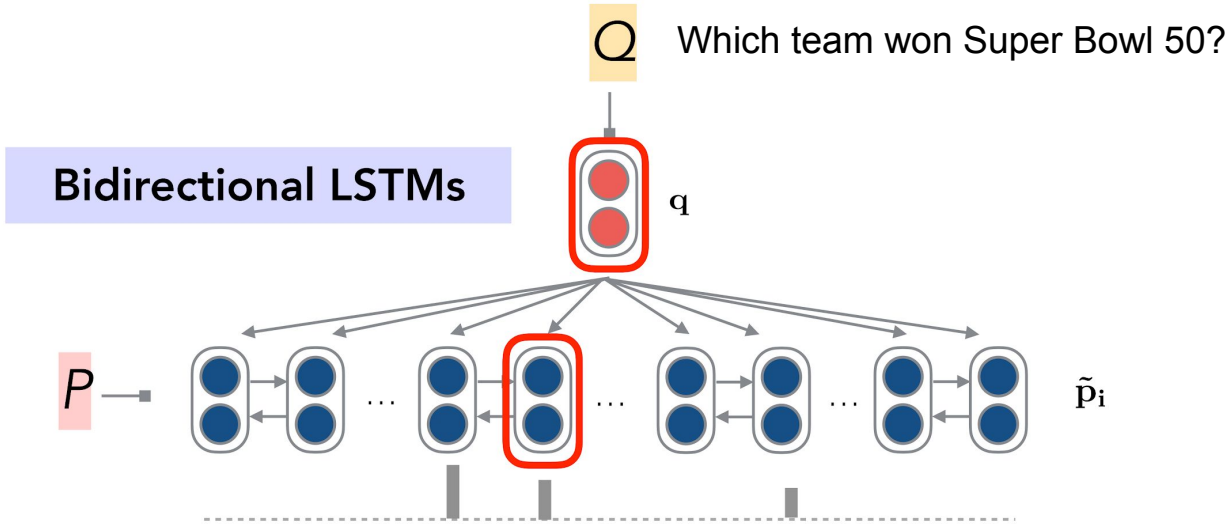
Slides credit: Christopher Manning

Stanford Attentive Reader



Slides credit: Christopher Manning

Stanford Attentive Reader



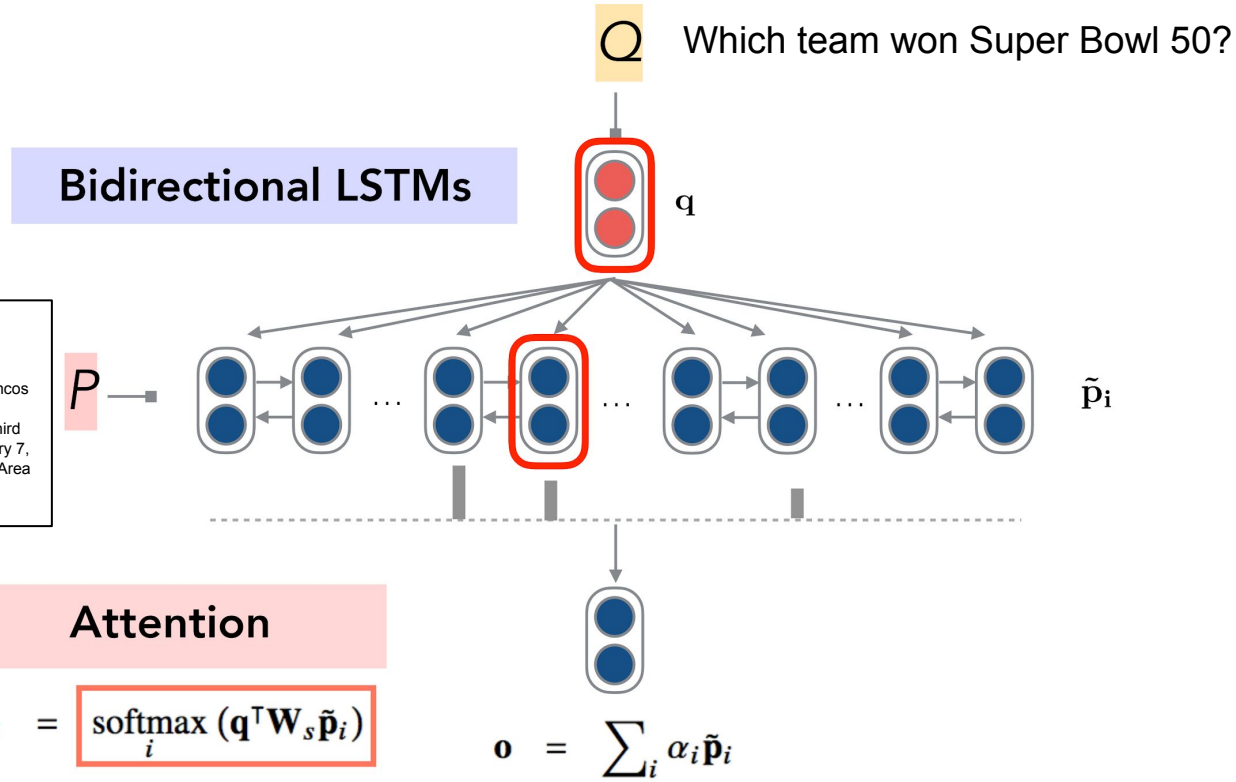
Attention

$$\alpha_i = \underset{i}{\text{softmax}}\left(\mathbf{q}^\intercal \mathbf{W}_s \tilde{\mathbf{p}}_i\right)$$

Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California.

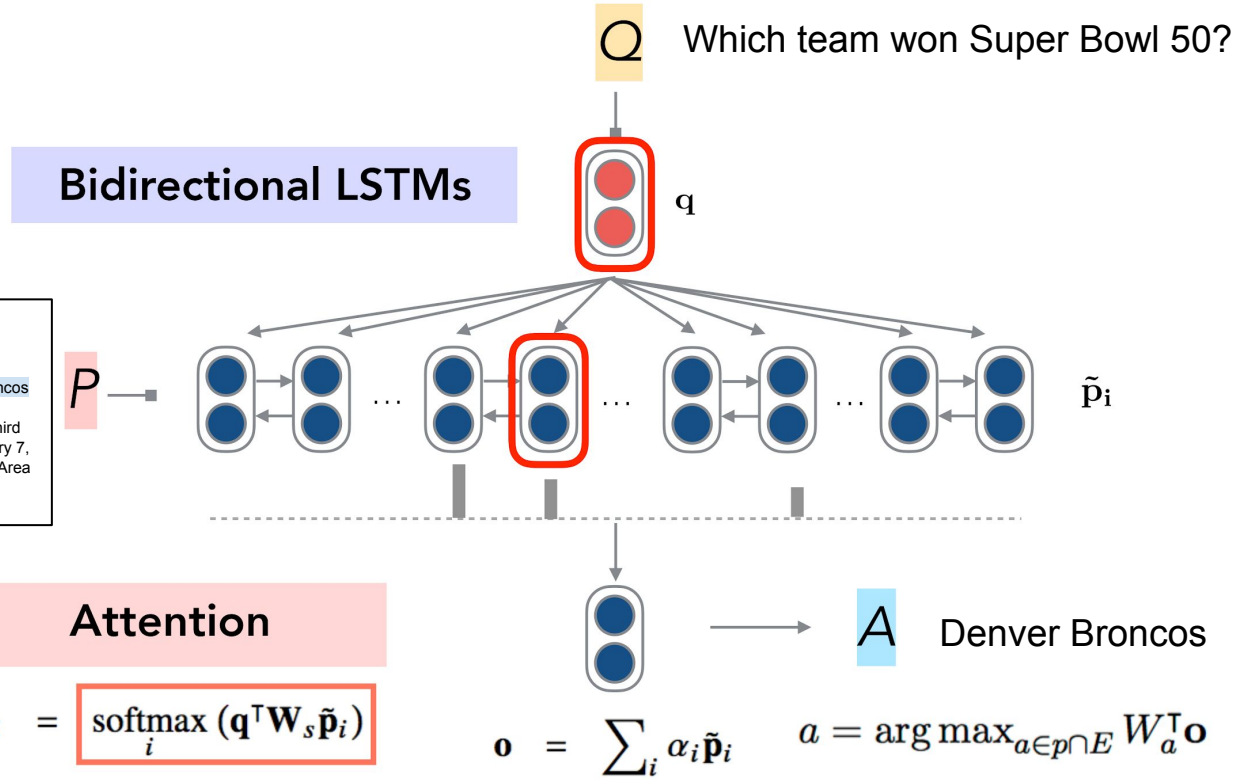
Slides credit: Christopher Manning

Stanford Attentive Reader



Slides credit: Christopher Manning

Stanford Attentive Reader



Slides credit: Christopher Manning

Stanford Attentive Reader++

- $\mathbf{p}_i$: Vector representation of each token in passage

Made from concatenation of

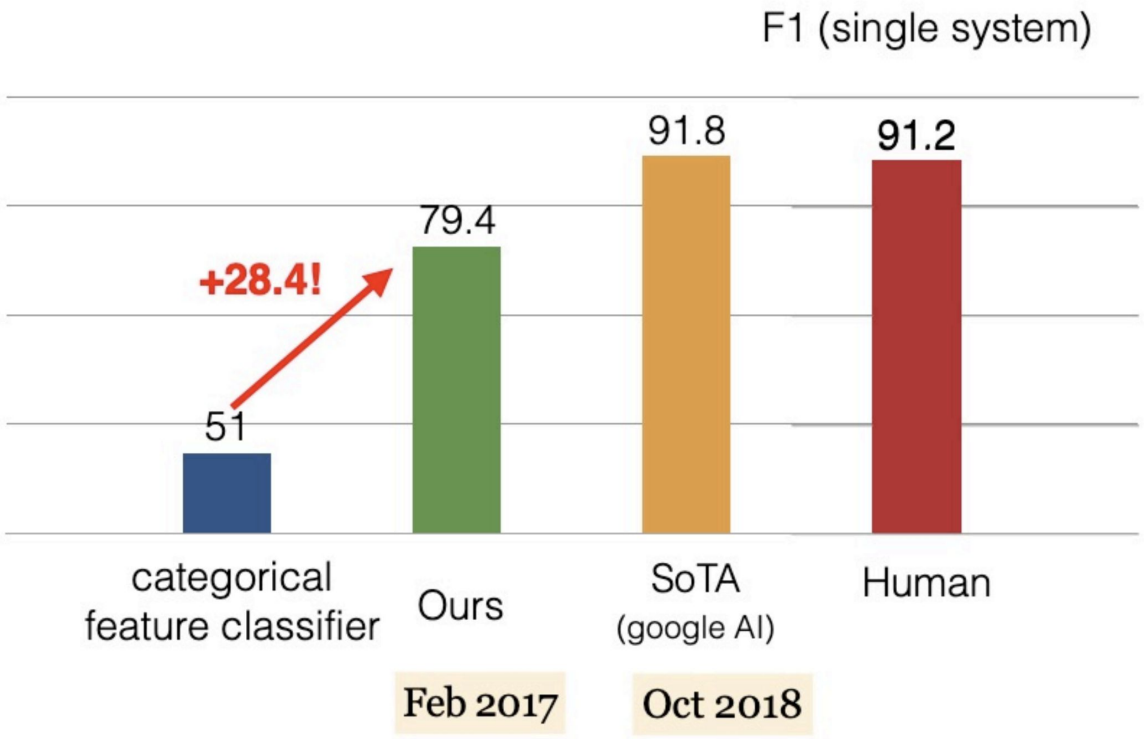
- Word embedding (GloVe 300d)
- Linguistic features: POS & NER tags, one-hot encoded
- Term frequency (unigram probability)
- Exact match: whether the word appears in the question
  - 3 binary features: exact, uncased, lemma
- Aligned question embedding ("car" vs "vehicle")

$$f_{align}(p_i) = \sum_j a_{i,j}\mathbf{E}(q_j) \qquad q_{i,j} = \frac{\exp(\alpha(\mathbf{E}(p_i)) \cdot \alpha(\mathbf{E}(q_j)))}{\sum_{j'} \exp(\alpha(\mathbf{E}(p_i)) \cdot \alpha(\mathbf{E}(q'_j)))}$$

Where α is a simple one layer FFNN

Slides credit: Christopher Manning

Stanford Attentive Reader++



Slides credit: Christopher Manning

SQuAD Leaderboard 2.0/1.1

Leaderboard (SQuAD2.0)

SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph. How will your system compare to humans on this task?

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jan 15, 2019	BERT + MMFT + ADA (ensemble) *Microsoft Research Asia*	**85.082**	**87.615**
2 Jan 10, 2019	BERT + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	84.292	86.967
3 Dec 13, 2018	BERT finetune baseline (ensemble) *Anonymous*	83.536	86.096
4 Dec 16, 2018	Lunet + Verifier + BERT (ensemble) *Layer 6 AI NLP Team*	83.469	86.043
4 Dec 21, 2018	PAML+BERT (ensemble model) *PINGAN GammaLab*	83.457	86.122
5 Dec 15, 2018	Lunet + Verifier + BERT (single model) *Layer 6 AI NLP Team*	82.995	86.035

SQuAD1.1 Leaderboard

Since the release of SQuAD1.0, the community has made rapid progress, with the best models now rivaling human performance on the task. Here are the ExactMatch (EM) and F1 scores evaluated on the test set of v1.1.

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Oct 05, 2018	BERT (single model) *Google AI Language* https://arxiv.org/abs/1810.04805	85.083	91.835
2 Sep 09, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.356	91.202
2 Sep 26, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.954	91.677
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490
4 Jul 08, 2018	r-net (ensemble) *Microsoft Research Asia*	84.003	90.147

https://rajpurkar.github.io/SQuAD-explorer/

Outline

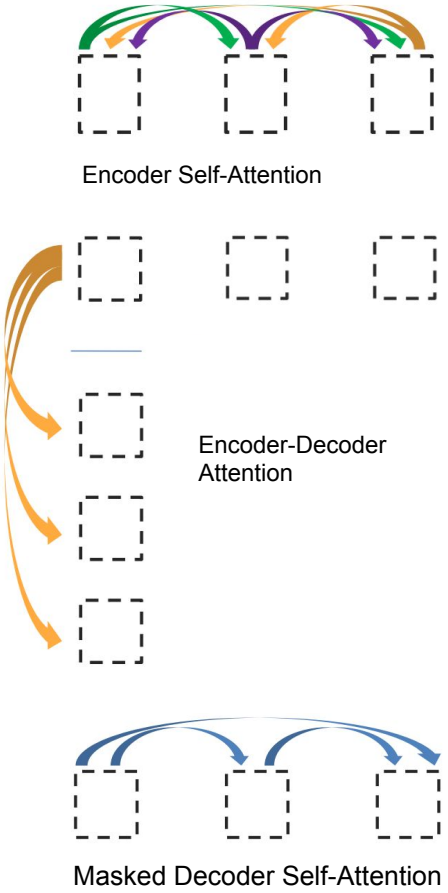
- Application: Text Classification
- Application: Language Modeling
- Application: Machine Translation (Seq2Seq)
  - Machine Translation with Attention
- Application: Question Answering
- **Advanced Attention mechanism: Transformers**
- Pre-training/Transfer Learning

Transformers
Attention is all you need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

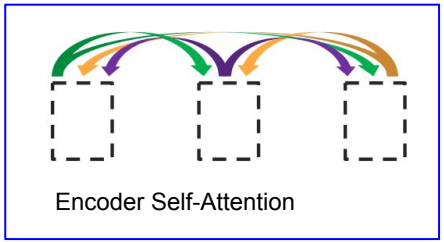
NIPS 2017

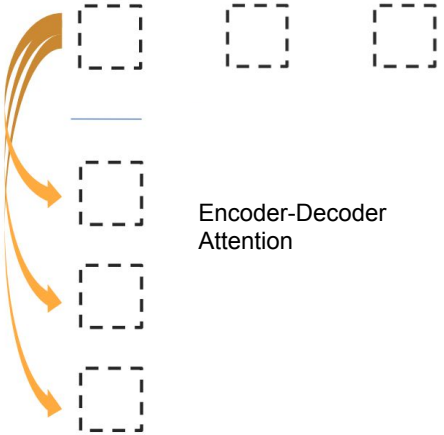
Overview - Visualization

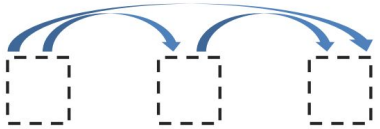


https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Overview - Visualization

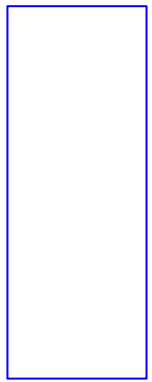




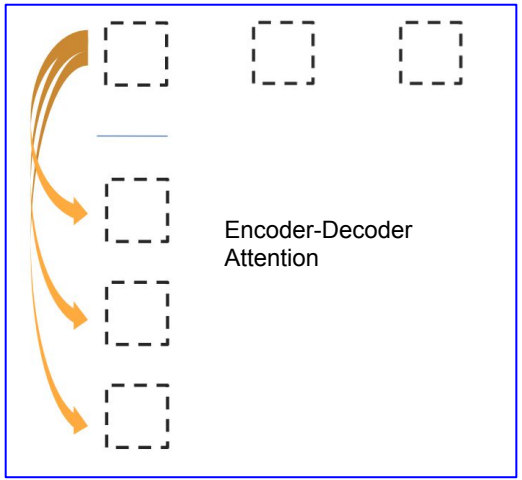


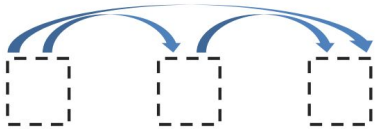
https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Overview - Visualization



Encoder Self-Attention

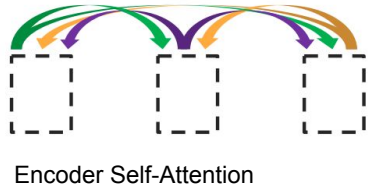




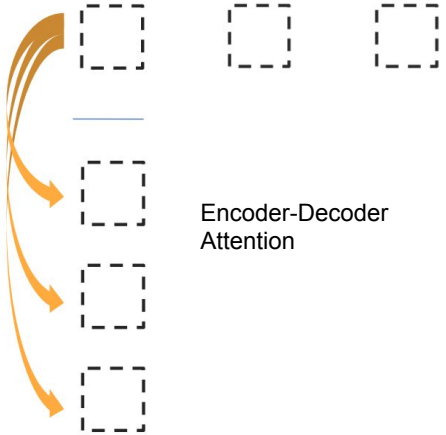
Masked Decoder Self-Attention

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

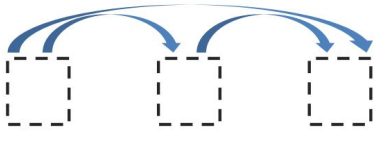
Overview - Visualization



Encoder Self-Attention



Encoder-Decoder Attention

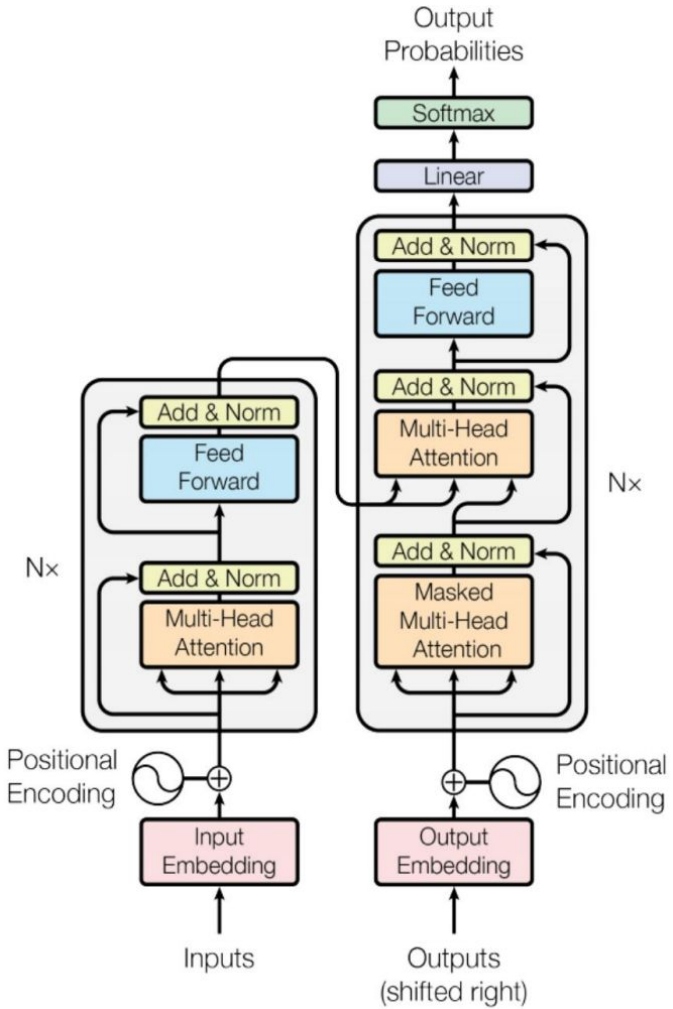


Masked Decoder Self-Attention

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

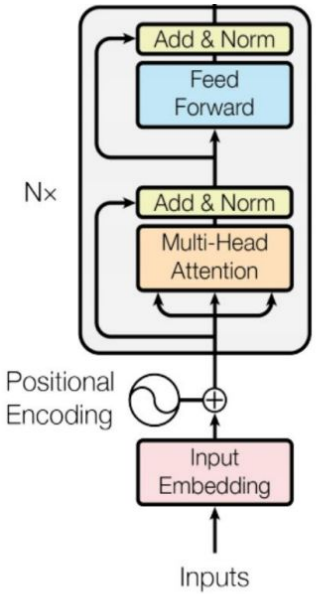
The Transformer

- Architecture
- Key features
  - Multi-head attention
  - Residual connections
  - Layer norm
  - Position-wise feedforward layers



Encoder

- Two sub-layers
  - Output of each sub-layer: LayerNorm(x+Sublayer(x))
- Sublayer(.) takes in a sequence of embeddings and outputs an analogous sequence of embeddings of the same dimensionality

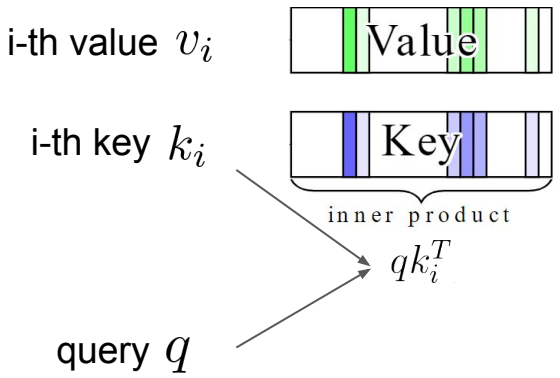


* Layer Normalization (Ba et al. 2016): apply instance-level normalization for each layer by aggregating over all hidden units

$$\bar{a}_i^l = \frac{g_i^l}{\sigma_i^l}\left(a_i^l - \mu_i^l\right) \qquad \mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l \qquad \sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2}$$

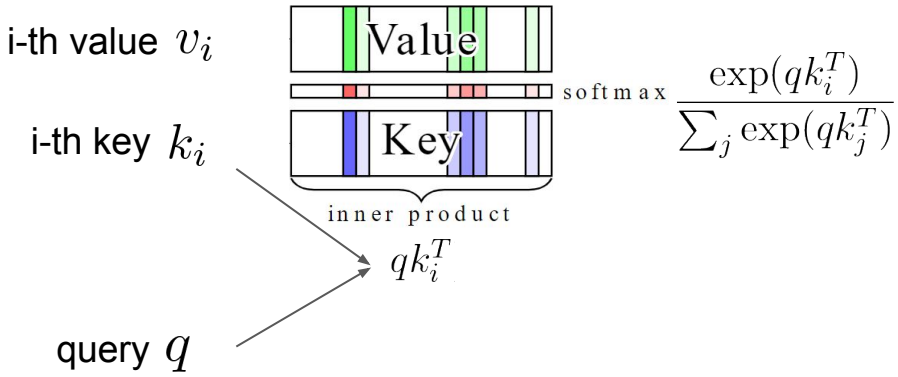
Attention

- Mapping a query (q) and a set of key-value (k, v) pairs to an output
- Eg:
  - Query q and set of key-value pairs (k_i, v_i)
  - Dot-product attention
    - Query and key have to be of same dimension
    - Value can have any dimensionality



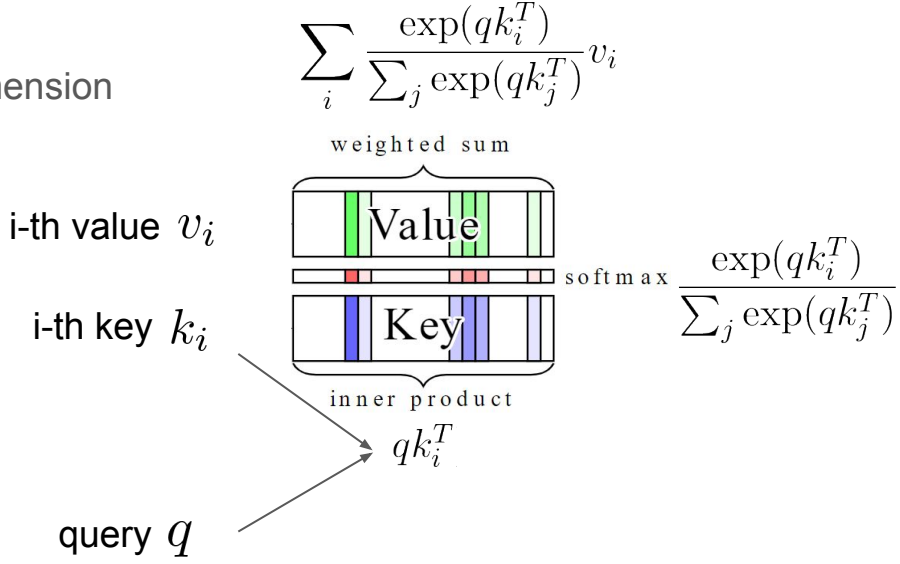
Attention

- Mapping a query (q) and a set of key-value (k, v) pairs to an output
- Eg:
  - Query q and set of key-value pairs (k_i, v_i)
  - Dot-product attention
    - Query and key have to be of same dimension
    - Value can have any dimensionality



Attention

- Mapping a query (q) and a set of key-value (k, v) pairs to an output
- Eg:
  - Query q and set of key-value pairs (k_i, v_i)
  - Dot-product attention
    - Query and key have to be of same dimension
    - Value can have any dimensionality



Attention

- Mapping a query (q) and a set of key-value (k, v) pairs to an output
- Eg:
  - Query q and set of key-value pairs (k_i, v_i)
  - Dot-product attention
    - Query and key have to be of same dimension
    - Value can have any dimensionality

$$\sum_i \frac{\exp(qk_i^T)}{\sum_j \exp(qk_j^T)} v_i$$

- Can be efficiently implemented using matmul

$$\text{softmax}(QK^T)V$$

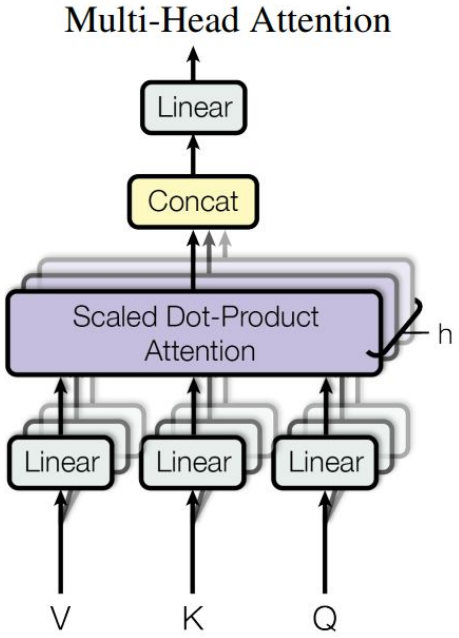
- Scaled dot-product attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{d_k}\right)V$$

d_k: dimension of the keys

Attention

- Contrast with convolution
  - Feature-maps: Learn different detectors
  - Attention: Just one detector
- Multi-head attention
  - Learn several detectors/feature-maps
  - Multiple attention layers in parallel



$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

Multi-head attention

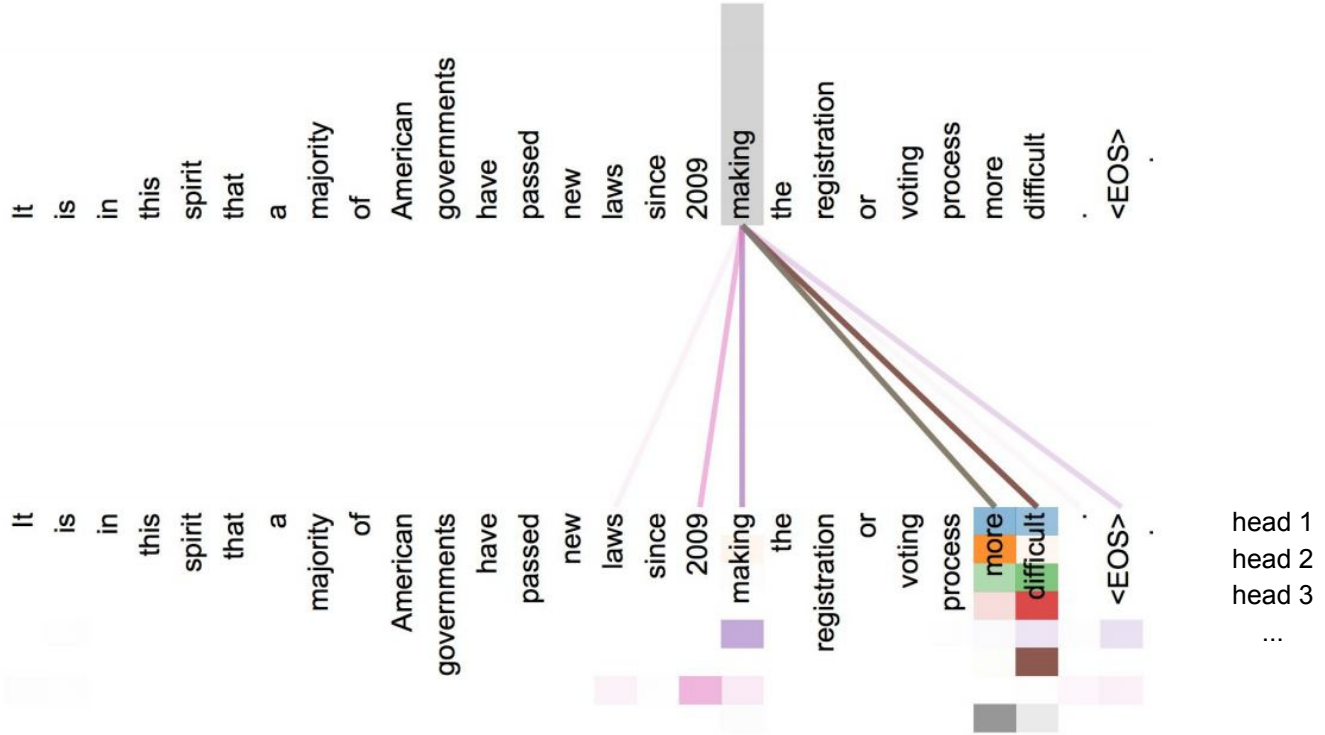


Figure source: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Self-attention

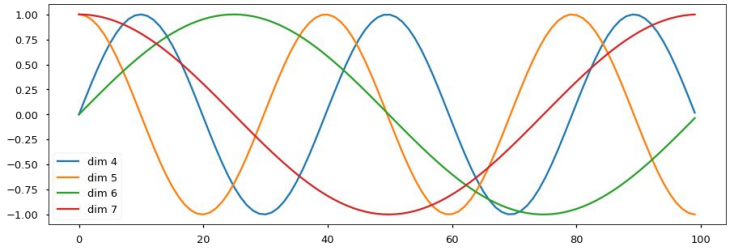
- Constant path length between any two positions
- Variable-sized perceptive field
- Gating/multiplication enables crisp error propagation
- Trivial to parallelize (per layer)

Encoder Self-Attention

Positional encoding

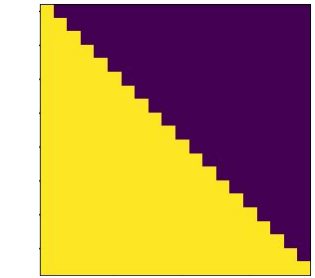
- Attention has no notion of location
  - If (key, value) pairs are permuted, we get the same result
- Introduce positional parameters
  - Add them to word embeddings at the bottom
- Choices
  - Trained parameters
  - Fixed parameters

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

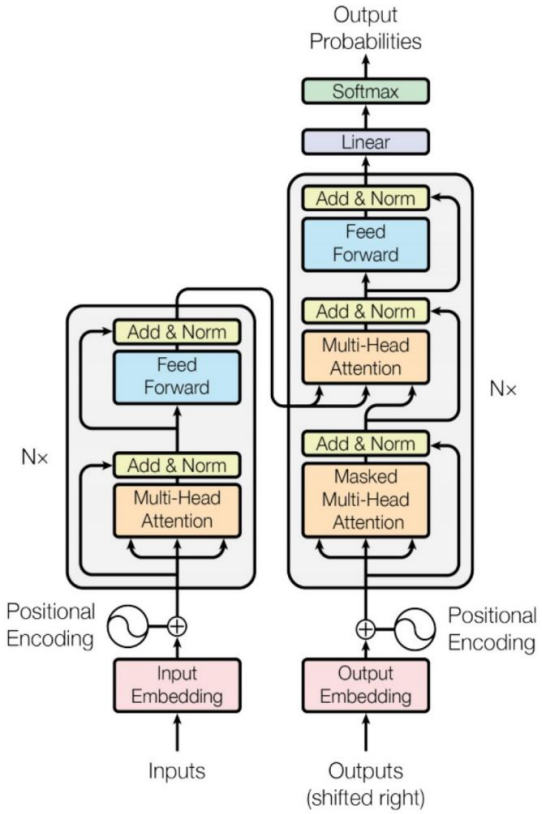


Decoder

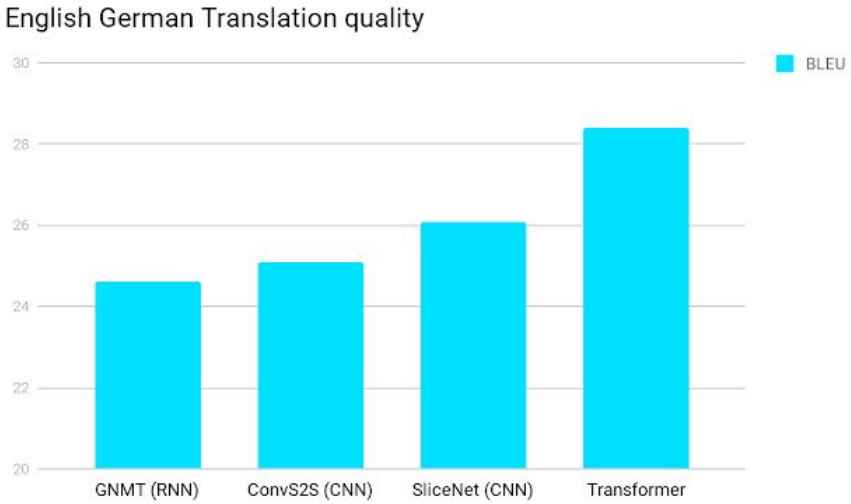
- One additional sub-layer
- Masked attention
  - Avoid seeing future inputs
  - Apply triangle mask
- Decoder-Encoder attention

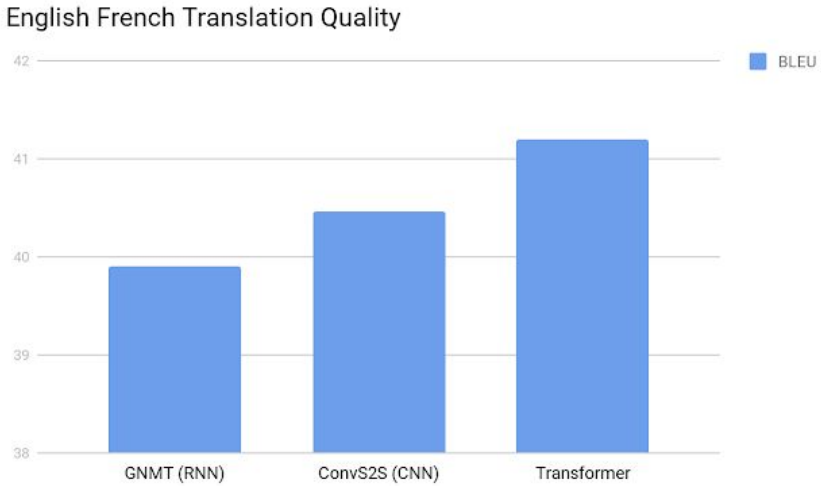


Masked Decoder Self-Attention



Machine Translation performance





BLEU scores (higher is better) of single models on the standard WMT newstest2014 dataset

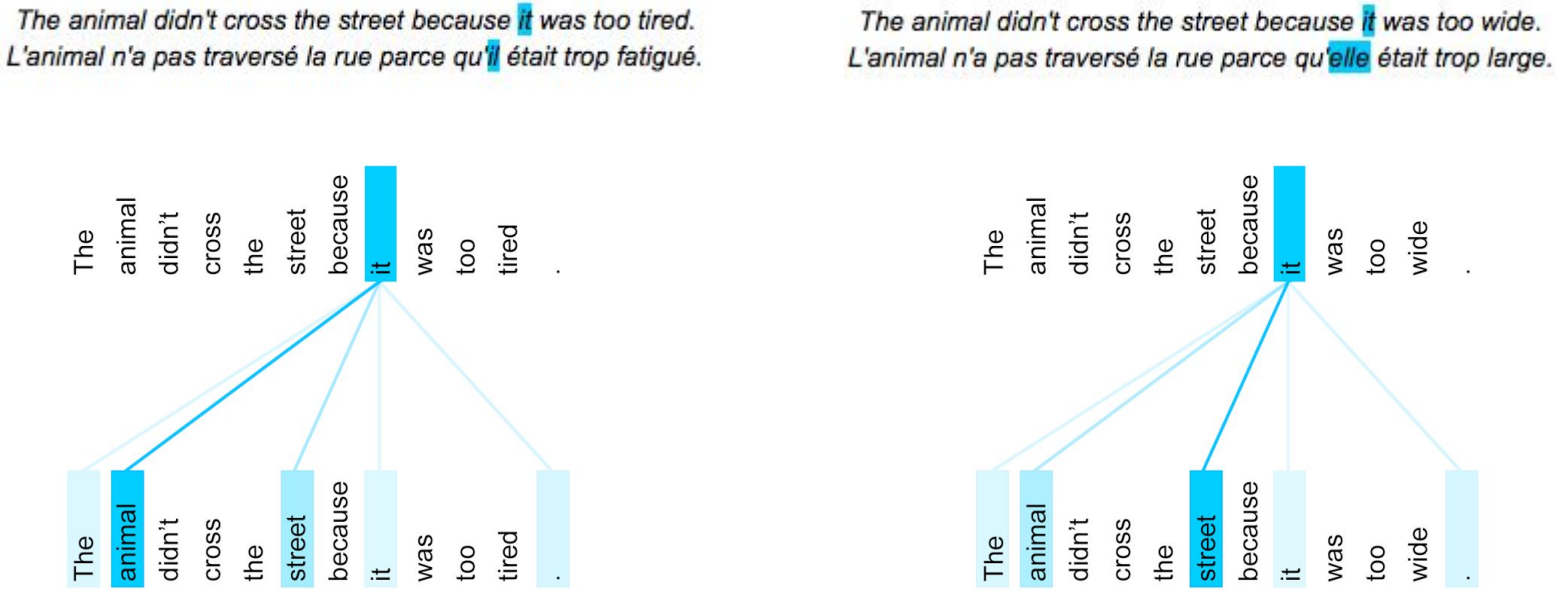
Training Details:

- Word-piece vocabulary
- Training on 8 P100s
- Learning rate decay with warmup
- Dropout, label smoothing

Figure source: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Visualization

- MT example - Coreference



The encoder self-attention distribution for the word “it” from the 5th to the 6th layer of a Transformer trained on English to French translation (one of eight attention heads).

Figure source: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Outline

- Application: Text Classification
- Application: Language Modeling
- Application: Machine Translation (Seq2Seq)
  - Machine Translation with Attention
- Application: Question Answering
- Advanced Attention mechanism: Transformers
- **Pre-training/Transfer Learning**

Overview

Question: How can we train a good representation of text for many downstream supervised tasks?

Solutions: pretrain text embedding with unsupervised learning tasks

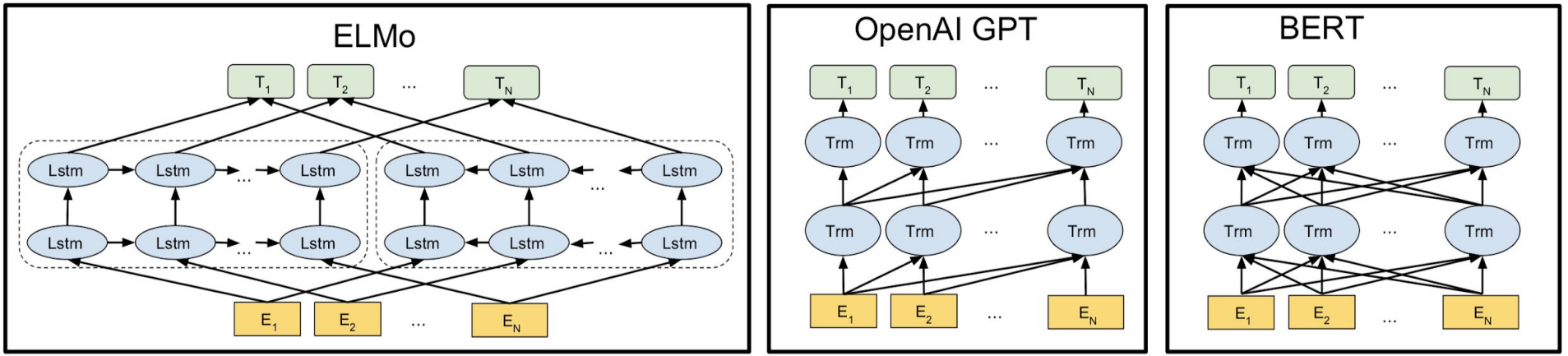
- Language modeling
- Fill-in-the-blank
- Next sentence prediction

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.
Devlin et al. "BERT: Pre-training of deep bidirectional transformers for language understanding." (2018).
ELMo: Peters et al. (2018). Deep contextualized word representations.

Overview

E_i: input embedding at time i

T_i: output embedding at time i

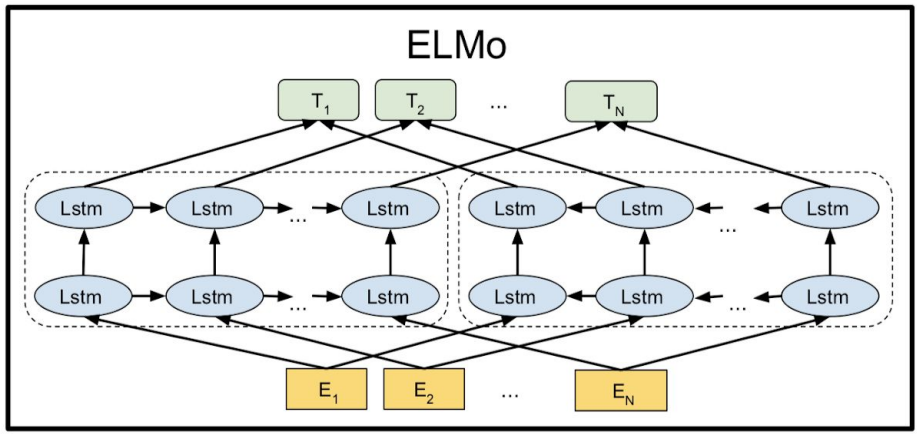


- ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features for downstream tasks.
- OpenAI GPT uses a left-to-right Transformer.
- BERT uses a bidirectional Transformer.

Source: Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv* (2018).

ELMO

- Learn ‘contextualized’ word representations
  - Make the representation of a word sensitive to the context in which it appears
- word representation = function of entire input sentence
- TLDR
  - Train bi-directional language model on large corpus
  - Given a sentence, run it through the trained LM and augment word representations with hidden layer representations

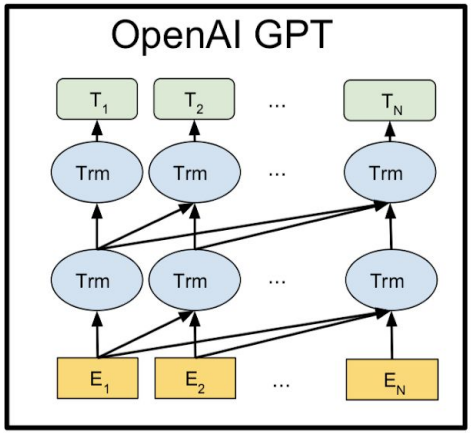


ELMo: Peters et al. (2018). Deep contextualized word representations.

Open AI GPT

- Tokens $\mathcal{U} = \{u_1, \ldots, u_n\}$
- LM objective $L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$
- Multi-layer Transformer decoder
  - Previous tokens $U = (u_{-k}, \ldots, u_{-1})$

$$h_0 = UW_e + W_p$$
$$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$
$$P(u) = \texttt{softmax}(h_n W_e^T)$$



Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

Supervised fine-tuning

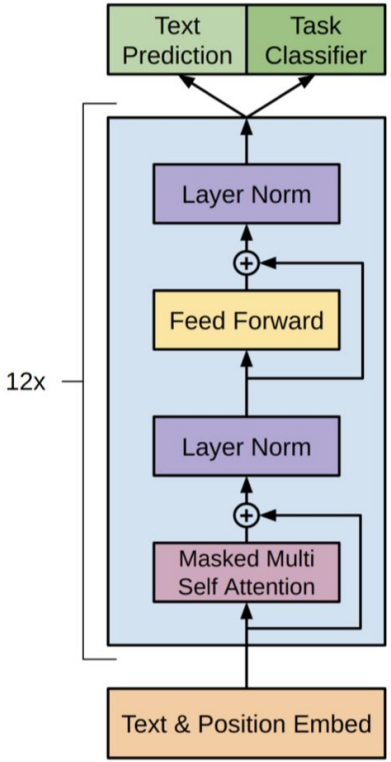
- Final transformer block activations h_l^m
- Supervised objective

$$P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y)$$

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m)$$

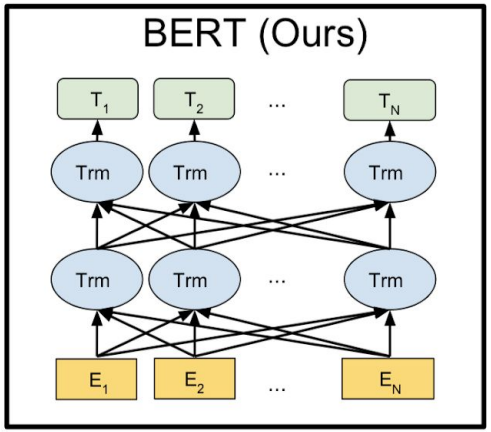
- Multi-task objective

$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$



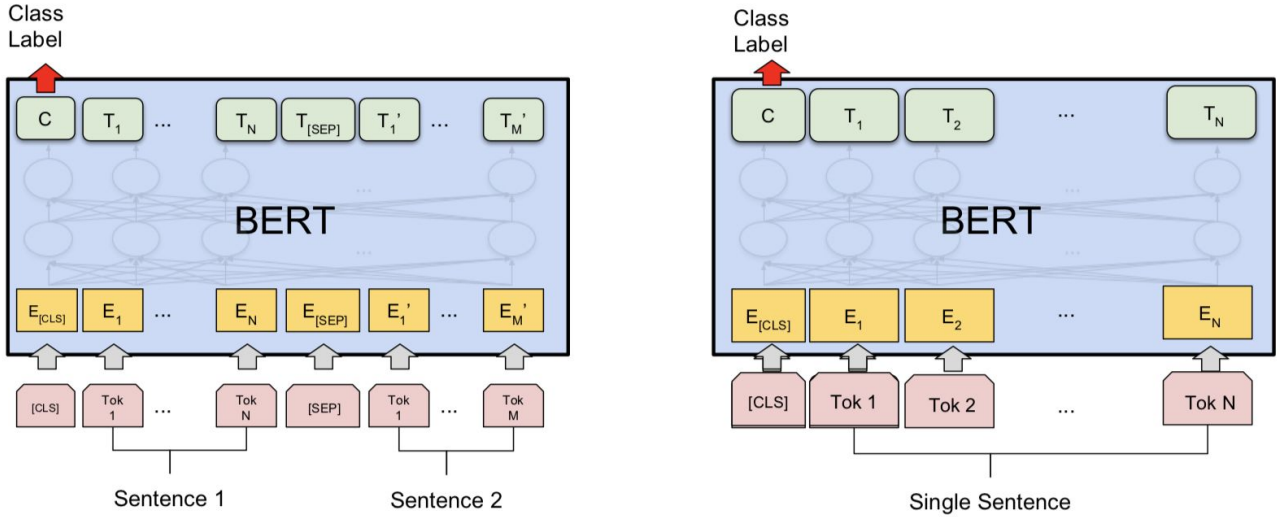
BERT

- Prior LM training does not use bi-directional context
  - LM training is generally uni-directional
- Tasks
  - Fill-in-the-blank task
  - Next sentence prediction
- Bi-directional transformer
  - Replace words to be predicted by a blank token
  - Perform prediction task for only the blank positions



Fine-tuning

- Sequence-level tasks



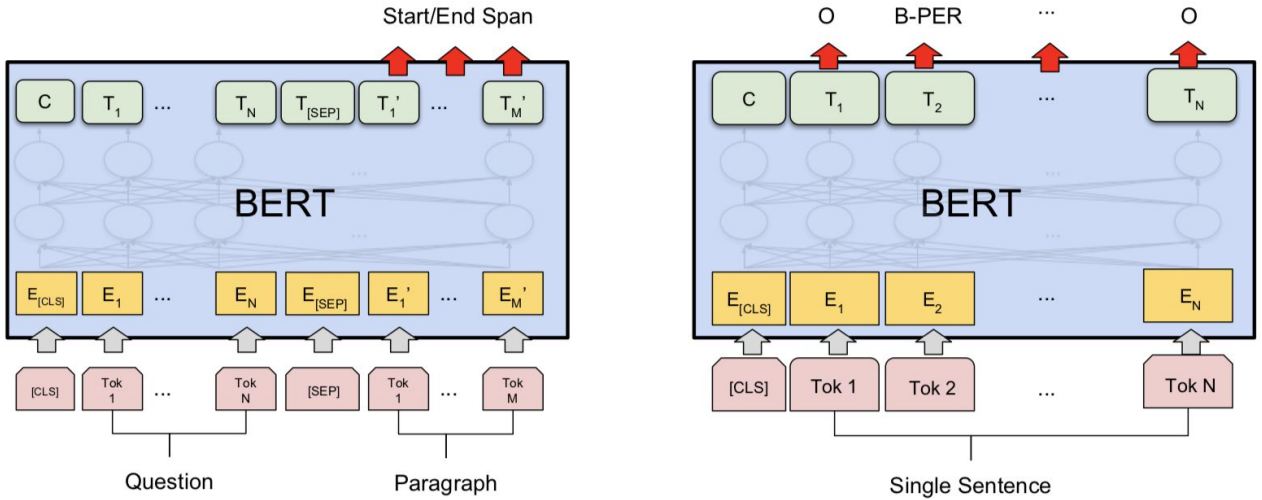
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

Figure source: Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv* (2018).

Fine-tuning

- Token-level tasks



(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Figure source: Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv* (2018).

GLUE benchmark

https://gluebenchmark.com/leaderboard

BERT

OpenAI GPT

ELMo

	Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI	RTE	WNLI	AX
+	1	Jacob Devlin	BERT: 24-layers, 1024-hidden, 16-heads		80.4	60.5	94.9	85.4/89.3	87.6/86.5	89.3/72.1	86.7	85.9	91.1	70.1	65.1	39.6
	2	Alec Radford	Singletask Pretrain Transformer		72.8	45.4	91.3	75.7/82.3	82.0/80.0	88.5/70.3	82.1	81.4	88.1	56.0	53.4	29.8
+	3	Samuel Bowman	BiLSTM+ELMo+Attn		70.5	36.0	90.4	77.9/84.9	75.1/73.3	84.7/64.8	76.4	76.1	79.9	56.8	65.1	26.5
	4	GLUE Baselines	BiLSTM+ELMo+Attn		68.9	18.9	91.6	77.3/83.5	72.8/71.1	83.5/63.3	75.6	75.9	81.7	61.2	65.1	22.6
			GenSen		66.6	7.7	83.1	76.6/83.0	79.3/79.2	82.9/59.8	71.4	71.3	82.3	59.2	65.1	20.6
			Single Task BiLSTM+ELMo		66.2	35.0	90.2	69.0/80.8	64.0/60.2	85.7/65.6	72.9	73.4	69.4	50.1	65.1	19.5
			BiLSTM+Attn		65.7	0.0	85.0	75.1/83.7	73.9/71.8	84.3/63.6	72.2	72.1	82.1	61.7	63.7	24.6
			BiLSTM+ELMo		64.9	27.5	89.6	76.2/83.5	67.0/65.9	78.5/57.8	67.1	68.0	66.7	55.7	62.3	19.2
			Single Task BiLSTM+ELMo+Attn		64.8	35.0	90.2	68.8/80.2	55.5/52.5	86.5/66.1	76.9	76.7	61.1	50.3	65.1	27.9

Why LM pre-training is helpful

- Generative model learns to perform many of the evaluation tasks to improve language modeling capability
- Eg: Sentiment analysis
  - “*I liked the movie, it was very ____*”

Summary

- Applications Case Studies:
  - Text Classification
  - Language Modeling
  - Machine Translation
  - Question Answering
- Architectures/Methods:
  - CNN, RNN/LSTMs (multilayer/bidirectional LSTM variants)
  - Sequence-to-sequence
  - Attention mechanism
- Advanced Attention: Transformers
- Pre-training/Transfer Learning: ELMo, GPT, BERT