Interpreting, Training, and Distilling Seq2Seq Models

Alexander Rush (@harvardnlp)

(with Yoon Kim, Sam Wiseman, Hendrik Strobelt, Yuntian Deng, Allen Schmaltz)

http://www.github.com/harvardnlp/seq2seq-talk/

VE RI TAS

at

Carnegie Mellon University
Language Technologies Institute

Sequence-to-Sequence

- Machine Translation (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014b; Cho et al., 2014; Bahdanau et al., 2014; Luong et al., 2015)
- Question Answering (Hermann et al., 2015)
- Conversation (Vinyals and Le, 2015) (Serban et al., 2016)
- Parsing (Vinyals et al., 2014)
- Speech (Chorowski et al., 2015; Chan et al., 2015)
- Caption Generation (Karpathy and Li, 2015; Xu et al., 2015; Vinyals et al., 2015)
- Video-Generation (Srivastava et al., 2015)
- NER/POS-Tagging (Gillick et al., 2016)
- Summarization (Rush et al., 2015)

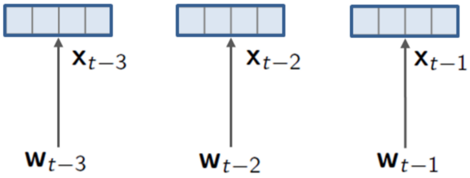
Sequence-to-Sequence

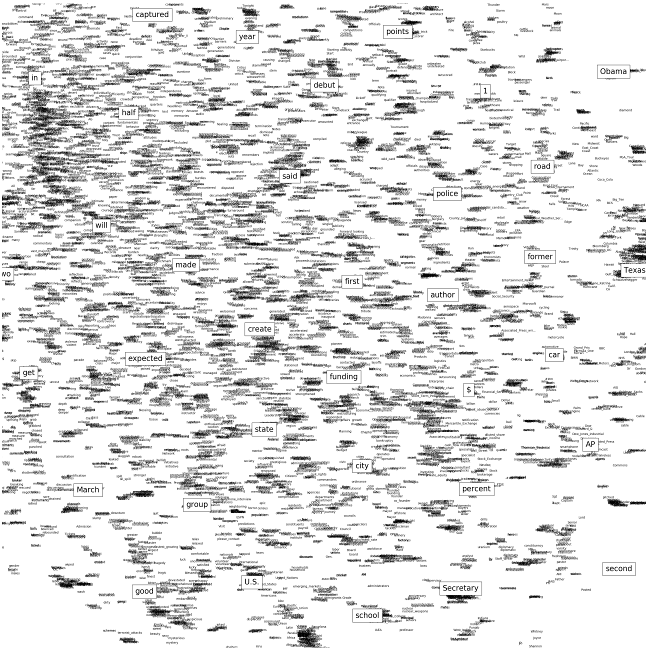
- Machine Translation (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014b; Cho et al., 2014; Bahdanau et al., 2014; Luong et al., 2015)
- Question Answering (Hermann et al., 2015)
- Conversation (Vinyals and Le, 2015) (Serban et al., 2016)
- Parsing (Vinyals et al., 2014)
- Speech (Chorowski et al., 2015; Chan et al., 2015)
- Caption Generation (Karpathy and Li, 2015; Xu et al., 2015; Vinyals et al., 2015)
- Video-Generation (Srivastava et al., 2015)
- NER/POS-Tagging (Gillick et al., 2016)
- Summarization (Rush et al., 2015)

Seq2Seq Neural Network Toolbox

Embeddings	sparse features	$\Rightarrow$	dense features
RNNs	feature sequences	$\Rightarrow$	dense features
Softmax	dense features	$\Rightarrow$	discrete predictions

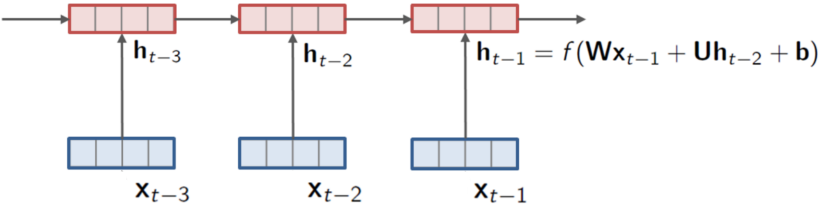
Embeddings sparse features $\Rightarrow$ dense features



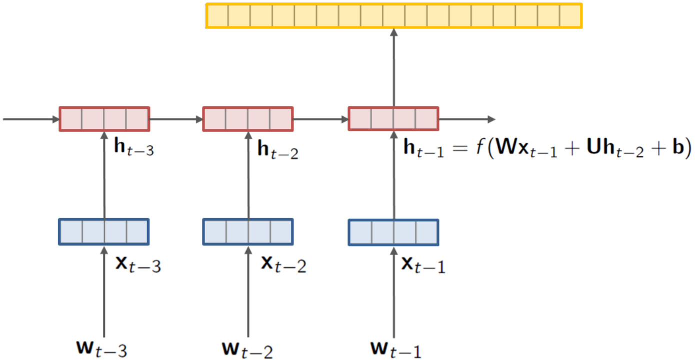


[Words Vectors]

RNNs/LSTMs feature sequences $\Rightarrow$ dense features



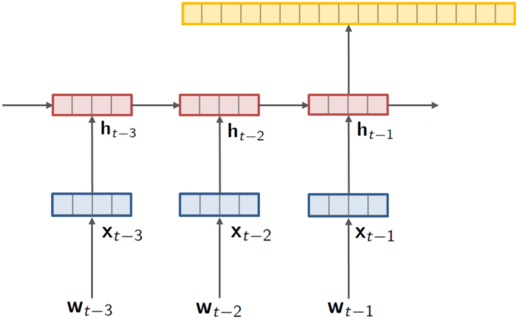
LM/Softmax dense features $\Rightarrow$ discrete predictions



$$p(w_t | w_1, \ldots, w_{t-1}; \theta) = \text{softmax}(\mathbf{W}_{out}\mathbf{h}_{t-1} + \mathbf{b}_{out})$$

$$p(w_{1:T}) = \prod_t p(w_t | w_1, \ldots, w_{t-1})$$

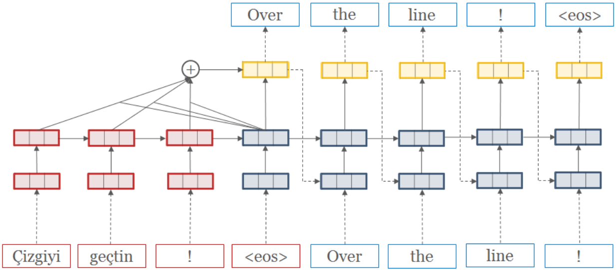
Contextual Language Model / "seq2seq"



- Key idea, contextual language model based on encoder x:

$$p(w_{1:T}|x) = \prod_t p(w_t|w_1, \ldots, w_{t-1}, x)$$

Actual Seq2Seq / Encoder-Decoder / Attention-Based Models

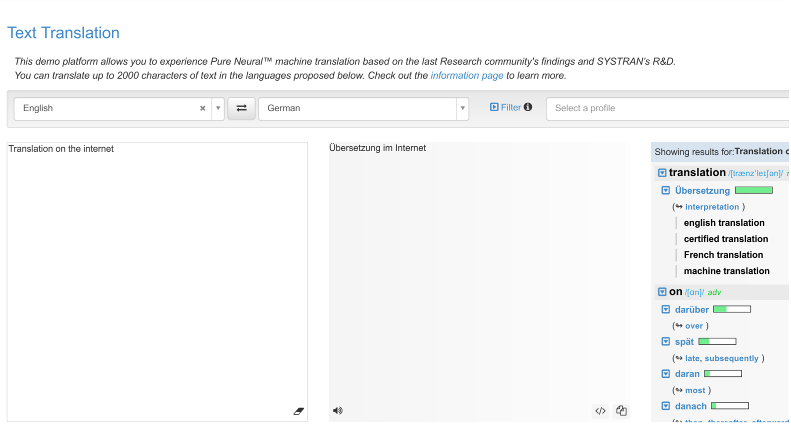


- Different encoders, attention mechanisms, input feeding, ...
- Almost all models use LSTMs or other gated RNNs
- Large multi-layer networks necessary for good performance.
  - 4 layer, 1000 hidden dims is common for MT

Seq2Seq-Attn

- HarvardNLP's open-source system (Yoon Kim)
  http://github.com/harvardnlp/seq2seq-attn

- Used by SYSTRAN for 32 language pairs (Crego et al., 2016)



Seq2Seq Applications: Neural Summarization (Rush et al., 2015)

Source (First Sentence)

Russian Defense Minister Ivanov called Sunday for the creation of a joint front for combating global terrorism.

Target (Title)

Russia calls for joint front against terrorism.

- (Mou et al., 2015) (Cheng and Lapata, 2016) (Toutanova et al., 2016) (Wang et al., 2016b) (Takase et al., 2016), among others
- Used by Washington Post to suggest headlines (Wang et al., 2016a)

Seq2Seq Applications: Neural Summarization (Rush et al., 2015)

Source (First Sentence)

Russian Defense Minister Ivanov called Sunday for the creation of a joint front for combating global terrorism.

Target (Title)

Russia calls for joint front against terrorism.

- (Mou et al., 2015) (Cheng and Lapata, 2016) (Toutanova et al., 2016) (Wang et al., 2016b) (Takase et al., 2016), among others
- Used by Washington Post to suggest headlines (Wang et al., 2016a)

Seq2Seq Applications: Grammar Correction (Schmaltz et al., 2016)

Source (Original Sentence)

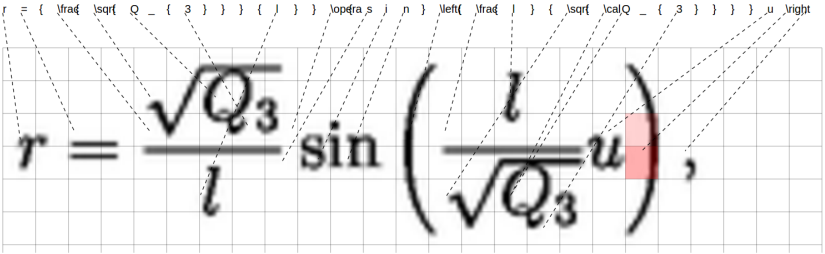
There is no a doubt, tracking systems has brought many benefits in this information age .

Target (Corrected Sentence)

There is no doubt, tracking systems have brought many benefits in this information age .

- 1st on BEA'11 grammar correction task (Daudaravicius et al., 2016)

Seq2Seq Applications: Im2Markup (Deng and Rush, 2016)



[Latex Example]

[Project]

This Talk

- How can we **interpret** these learned hidden representations?
- How should we **train** these style of models?
- How can we **shrink** these models for practical applications?

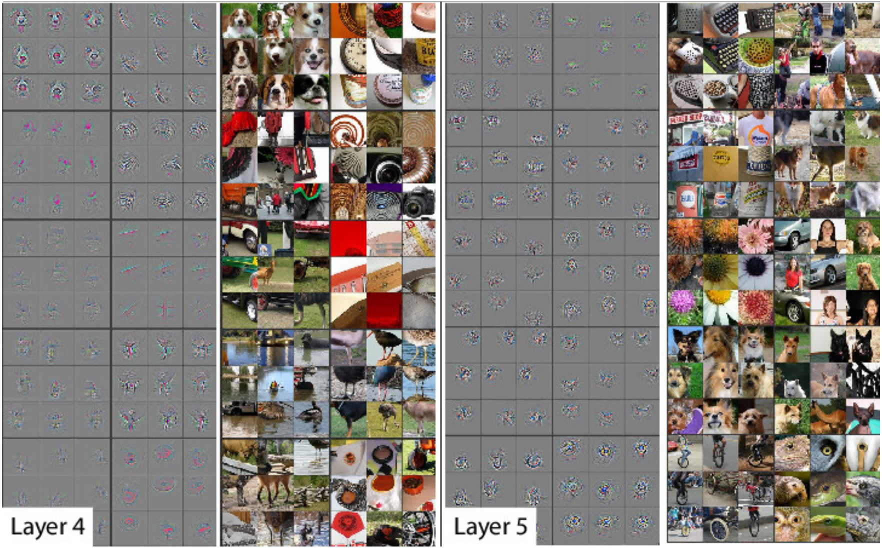
This Talk

- How can we **interpret** these learned hidden representations?

LSTMVis lstm.seas.harvard.edu

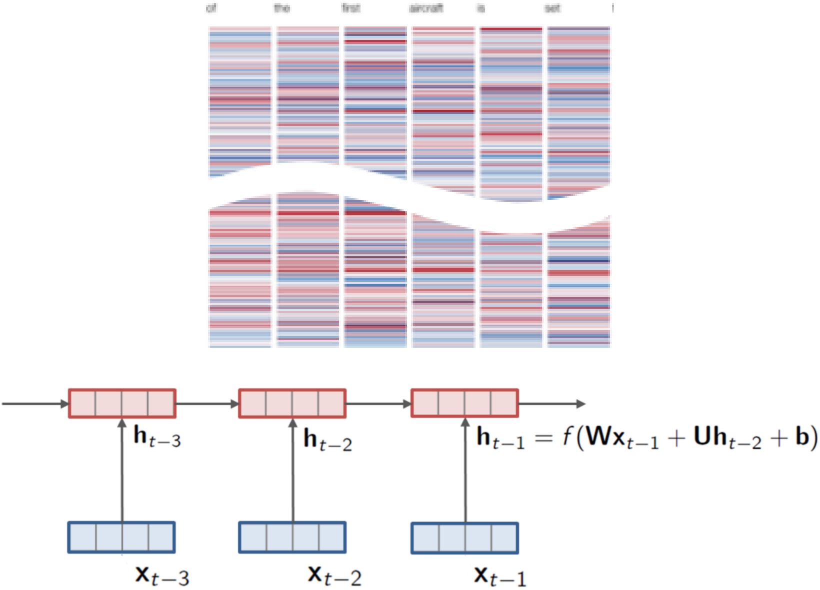
(Strobelt et al., 2016)

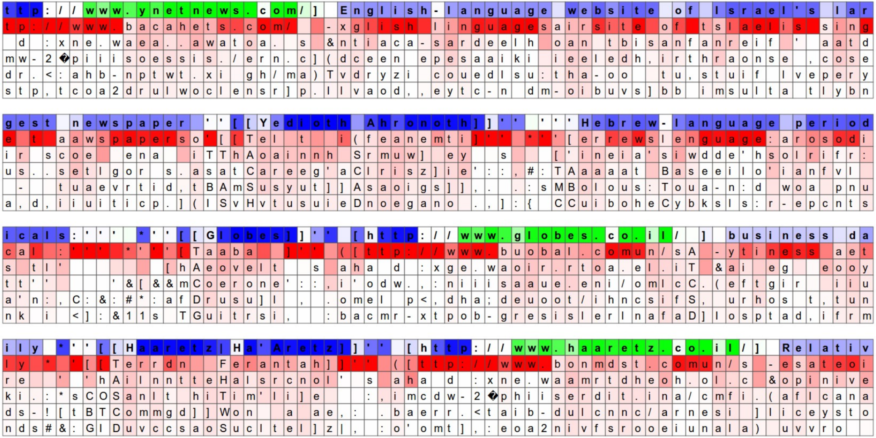
- How should we **train** these style of models? (Wiseman and Rush, 2016)
- How can we **shrink** these models for practical applications? (Kim and Rush, 2016)


Layer 4
Layer 5

(?)

Vector-Space RNN Representation





(Karpathy et al., 2015)

Example 1: Synthetic (Finite-State) Language

alphabet: ( ) 0 1 2 3 4

corpus: (1 (2) ()) 0 (((3)) 1)

- Numbers are randomly generated, must match nesting level.
- Train a predict-next-word language model (decoder-only).

$$p(w_t|w_1, \ldots, w_{t-1})$$

[Parens Example]

Example 2: Real Language

alphabet: all english words

corpus: Project Gutenberg Children's books

- Train a predict-next-word language model (decoder-only).

$$p(w_t|w_1, \ldots, w_{t-1})$$

[LM Example]

Example 3: Seq2Seq Encoder

alphabet: all english words

corpus: Summarization

- Train a full seq2seq model, examine *encoder* LSTM.

[Summarization Example]

This Talk

- How can we **interpret** these learned hidden representations? (Strobelt et al., 2016)
- How should we **train** these style of models?

Sequence-to-Sequence Learning as Beam-Search Optimization

(Wiseman and Rush, 2016)

- How can we **shrink** these models for practical applications (Kim and Rush, 2016)?

Seq2Seq Notation

- x; source input
- $\mathcal{V}$; vocabulary
- w_t; random variable for the t-th target token with support $\mathcal{V}$
- $y_{1:T}$; ground-truth output
- $\hat{y}_{1:T}$; predicted output
- $p(w_{1:T} \mid x; \theta) = \prod_t p(w_t | w_{1:t-1}, x; \theta)$; model distribution

Seq2Seq Details

Train Objective: Given source-target pairs $(x, y_{1:T})$, minimize NLL of each word independently, conditioned on *gold* history $y_{1:t-1}$

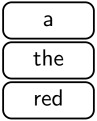
$$\mathcal{L}_{\mathsf{NLL}}(\theta) = -\sum_t \log p(w_t = y_t | y_{1:t-1}, x; \theta)$$

Test Objective: Structured prediction

$$\hat{y}_{1:T} = \arg\max_{w_{1:T}} \sum_t \log p(w_t | w_{1:t-1}, x; \theta)$$

- Typical to approximate the $\arg\max$ with beam-search

Beam Search $(K = 3)$



For $t = 1 \dots T$:

- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$

a

the

red

For $t = 1 \dots T$:

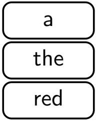
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1} | x) + \log p(w | \hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

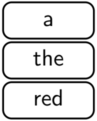
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \color{red}{\log p(\hat{y}^{(k)}_{1:t-1}|x)} + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \ldots T$:

- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \color{red}{\log p(w|\hat{y}^{(k)}_{1:t-1}, x)}$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$

a
the
red

For $t = 1 \ldots T$:

- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\color{red}{\hat{y}^{(1:K)}_{1:t}} \leftarrow \text{K-}\arg\max\, s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$

a

the

red

For $t = 1 \ldots T$:

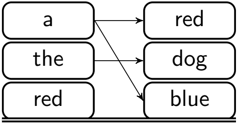
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1} | x) + \log p(w | \hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \textcolor{red}{\text{K-}\arg\max}\ s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

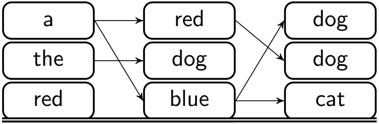
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1} | x) + \log p(w | \hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

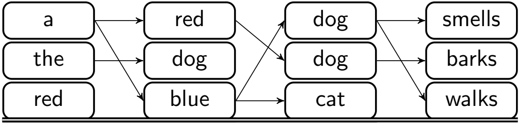
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

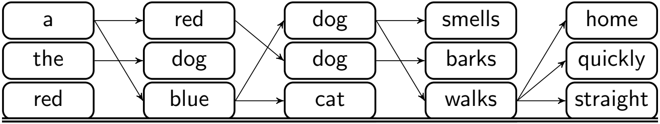
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1} | x) + \log p(w | \hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg max } s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

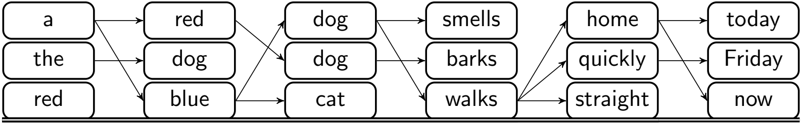
- For all k and for all possible output words w:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

- Update beam:

$$\hat{y}^{(1:K)}_{1:t} \leftarrow \text{K-arg}\max\ s(w, \hat{y}^{(k)}_{1:t-1})$$

Beam Search $(K = 3)$



For $t = 1 \dots T$:

- For all k and for all possible output words w:

$$s(w, \hat{y}_{1:t-1}^{(k)}) \leftarrow \log p(\hat{y}_{1:t-1}^{(k)}|x) + \log p(w|\hat{y}_{1:t-1}^{(k)}, x)$$

- Update beam:

$$\hat{y}_{1:t}^{(1:K)} \leftarrow \text{K-arg max } s(w, \hat{y}_{1:t-1}^{(k)})$$

Problem

How should we train sequence models?

Related Work

- Approaches to Exposure Bias, Label Bias:
  - Data as Demonstrator, Scheduled Sampling (Venkatraman et al., 2015; Bengio et al., 2015)
  - Globally Normalized Transition-Based Networks (Andor et al., 2016)
- RL-based approaches
  - MIXER (Ranzato et al., 2016)
  - Actor-Critic (Bahdanau et al., 2016)

Problem

How should we train sequence models?

Related Work

- Approaches to Exposure Bias, Label Bias:
  - Data as Demonstrator, Scheduled Sampling (Venkatraman et al., 2015; Bengio et al., 2015)
  - Globally Normalized Transition-Based Networks (Andor et al., 2016)
- RL-based approaches
  - MIXER (Ranzato et al., 2016)
  - Actor-Critic (Bahdanau et al., 2016)

Issue #1: Train/Test Mismatch (cf., (Ranzato et al., 2016))

$$\text{NLL}(\theta) = -\sum_t \log p(w_t = y_t | y_{1:t-1}, x; \theta)$$

(a) Training conditions on *true* history ("Exposure Bias")

(b) Train with word-level NLL, but evaluate with BLEU-like metrics

Idea #1: Train with beam-search

- Use a loss that incorporates sequence-level costs

Issue #1: Train/Test Mismatch (cf., (Ranzato et al., 2016))

$$\mathsf{NLL}(\theta) = -\sum_t \log p(w_t = y_t | y_{1:t-1}, x; \theta)$$

(a) Training conditions on *true* history ("Exposure Bias")

(b) Train with word-level NLL, but evaluate with BLEU-like metrics

Idea #1: Train with beam-search

- Use a loss that incorporates sequence-level costs

BSO Idea #1: Use a loss that incorporates sequence-level costs

$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

- $y_{1:t}$ is the gold prefix; $\hat{y}_{1:t}^{(K)}$ is the K'th prefix on the beam
- $\Delta(\hat{y}_{1:t}^{(K)})$ allows us to scale loss by badness of predicting $\hat{y}_{1:t}^{(K)}$

BSO Idea #1: Use a loss that incorporates sequence-level costs

$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

- $y_{1:t}$ is the gold prefix; $\hat{y}_{1:t}^{(K)}$ is the K'th prefix on the beam
- $\Delta(\hat{y}_{1:t}^{(K)})$ allows us to scale loss by badness of predicting $\hat{y}_{1:t}^{(K)}$

BSO Idea #1: Use a loss that incorporates sequence-level costs

$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

- $y_{1:t}$ is the gold prefix; $\hat{y}_{1:t}^{(K)}$ is the K'th prefix on the beam
- $\Delta(\hat{y}_{1:t}^{(K)})$ allows us to scale loss by badness of predicting $\hat{y}_{1:t}^{(K)}$

BSO Idea #1: Use a loss that incorporates sequence-level costs

$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

- $y_{1:t}$ is the gold prefix; $\hat{y}_{1:t}^{(K)}$ is the K'th prefix on the beam
- $\Delta(\hat{y}_{1:t}^{(K)})$ allows us to scale loss by badness of predicting $\hat{y}_{1:t}^{(K)}$

Issue #2: Seq2Seq models next-word probabilities:

$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

(a) Sequence score is sum of locally normalized word-scores; gives rise to “Label Bias” (Lafferty et al., 2001)

(b) What if we want to train with sequence-level constraints?

Idea #2: Don’t locally normalize

Issue #2: Seq2Seq models next-word probabilities:

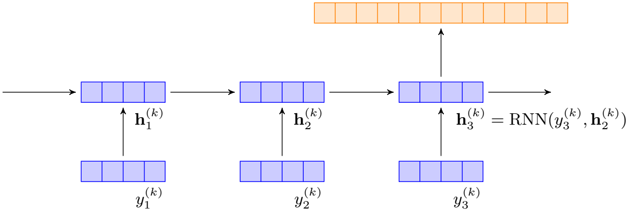
$$s(w, \hat{y}^{(k)}_{1:t-1}) \leftarrow \log p(\hat{y}^{(k)}_{1:t-1}|x) + \log p(w|\hat{y}^{(k)}_{1:t-1}, x)$$

(a) Sequence score is sum of locally normalized word-scores; gives rise to "Label Bias" (Lafferty et al., 2001)

(b) What if we want to train with sequence-level constraints?

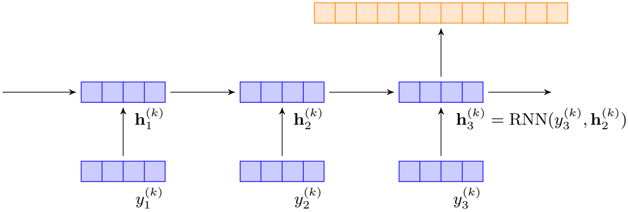
Idea #2: Don't locally normalize

BSO Idea #2: Don't locally normalize



$$s(w, \hat{y}_{1:t-1}^{(k)}) = \log p(\hat{y}_{1:t-1}^{(k)} | x) + \log \mathrm{softmax}(\mathbf{W}_{out}\, \mathbf{h}_{t-1}^{(k)} + \mathbf{b}_{out})$$

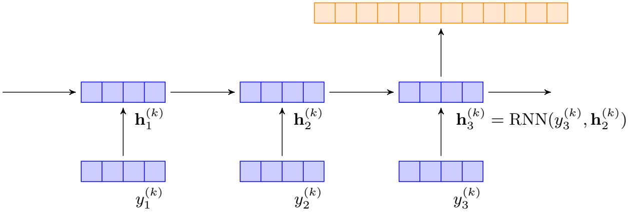
BSO Idea #2: Don't locally normalize



$$s(w, \hat{y}_{1:t-1}^{(k)}) = \cancel{\log p(\hat{y}_{1:t-1}^{(k)}|x) + \log \mathrm{softmax}(\mathbf{W}_{out}\, \mathbf{h}_{t-1}^{(k)} + \mathbf{b}_{out})}$$

$$= \mathbf{W}_{out}\, \mathbf{h}_{t-1}^{(k)} + \mathbf{b}_{out}$$

BSO Idea #2: Don't locally normalize

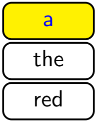


$$s(w, \hat{y}^{(k)}_{1:t-1}) = \cancel{\log p(\hat{y}^{(k)}_{1:t-1}|x) + \log \mathrm{softmax}(\mathbf{W}_{out}\,\mathbf{h}^{(k)}_{t-1} + \mathbf{b}_{out})}$$

$$= \mathbf{W}_{out}\,\mathbf{h}^{(k)}_{t-1} + \mathbf{b}_{out}$$

- Can set $s(w, \hat{y}^{(k)}_{1:t-1}) = -\infty$ if $(w, \hat{y}^{(k)}_{1:t-1})$ violates a hard constraint

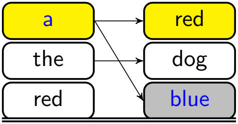
Beam Search Optimization



$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

- Color Gold: target sequence y
- Color Gray: violating sequence $\hat{y}^{(K)}$

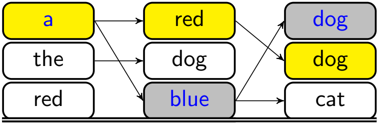
Beam Search Optimization



$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

- Color Gold: target sequence y
- Color Gray: violating sequence $\hat{y}^{(K)}$

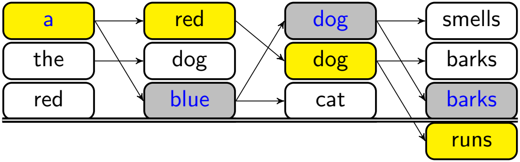
Beam Search Optimization



$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

- Color Gold: target sequence y
- Color Gray: violating sequence $\hat{y}^{(K)}$

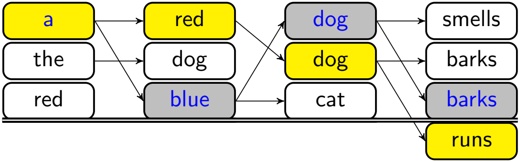
Beam Search Optimization



$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

- Color Gold: target sequence y
- Color Gray: violating sequence $\hat{y}^{(K)}$

Beam Search Optimization

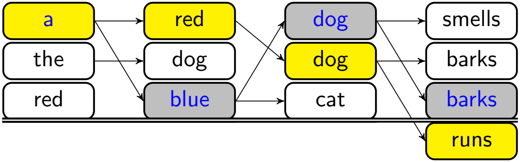


$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

LaSO (Daumé III and Marcu, 2005):

- If no margin violation at $t-1$, update beam as usual
- Otherwise, update beam with sequences prefixed by $y_{1:t-1}$

Beam Search Optimization

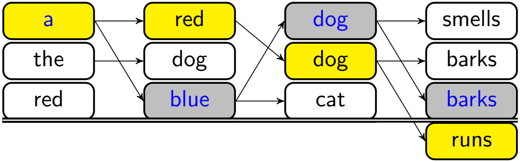


$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

LaSO (Daumé III and Marcu, 2005):

- If no margin violation at $t-1$, update beam as usual
- Otherwise, update beam with sequences prefixed by $y_{1:t-1}$

Beam Search Optimization

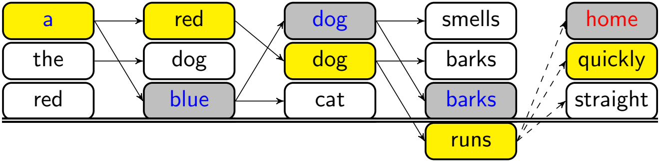


$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)})\right]$$

LaSO (Daumé III and Marcu, 2005):

- If no margin violation at $t-1$, update beam as usual
- Otherwise, update beam with sequences prefixed by $y_{1:t-1}$

Beam Search Optimization

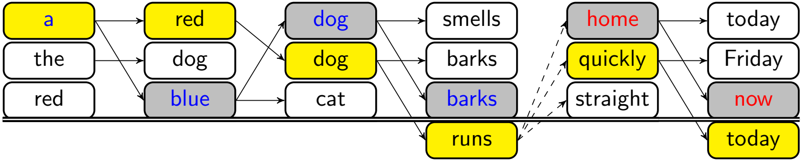


$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

LaSO (Daumé III and Marcu, 2005):

- If no margin violation at $t-1$, update beam as usual
- Otherwise, update beam with sequences prefixed by $y_{1:t-1}$

Beam Search Optimization

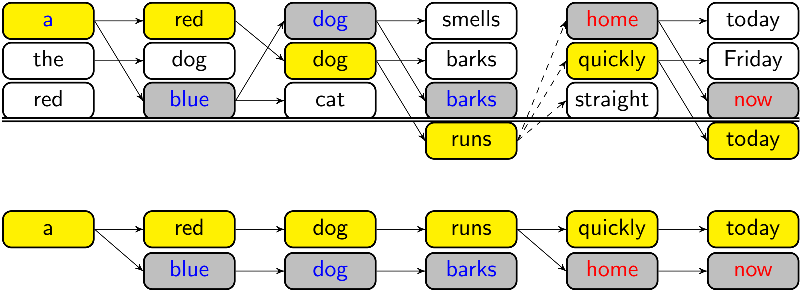


$$\mathcal{L}(\theta) = \sum_t \Delta(\hat{y}_{1:t}^{(K)}) \left[1 - s(y_t, y_{1:t-1}) + s(\hat{y}_t^{(K)}, \hat{y}_{1:t-1}^{(K)}) \right]$$

LaSO (Daumé III and Marcu, 2005):

- If no margin violation at $t-1$, update beam as usual
- Otherwise, update beam with sequences prefixed by $y_{1:t-1}$

Backpropagation over Structure



Experiments

- Word Ordering, Dependency Parsing, Machine Translation

- Uses LSTM encoders and decoders, attention, input feeding
- All models trained with Adagrad (Duchi et al., 2011)
- Pre-trained with NLL; K increased gradually
- "BSO" uses unconstrained search; "ConBSO" uses constraints

	$K_e = 1$	$K_e = 5$	$K_e = 10$
	Word Ordering (BLEU)		
seq2seq	25.2	29.8	31.0
BSO	28.0	33.2	34.3
ConBSO	**28.6**	**34.3**	**34.5**
	Dependency Parsing (UAS/LAS)[1]		
seq2seq	**87.33/82.26**	88.53/84.16	88.66/84.33
BSO	86.91/82.11	91.00/**87.18**	91.17/**87.41**
ConBSO	85.11/79.32	**91.25**/86.92	**91.57**/87.26
	Machine Translation (BLEU)		
seq2seq	22.53	24.03	23.87
BSO, SB-Δ, K_t=6	**23.83**	**26.36**	**25.48**
XENT	17.74	20.10	20.28
DAD	20.12	22.25	22.40
MIXER	20.73	21.81	21.83

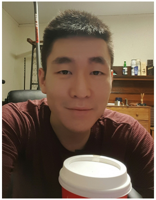
[1] Note Andor et al. (2016) have SOA, with 94.41/92.55.

This Talk

- How can we **interpret** these learned hidden representations? (Strobelt et al., 2016)
- How should we **train** these style of models? (Wiseman and Rush, 2016)
- How can we **shrink** these models for practical applications?

Sequence-Level Knowledge Distillation

(Kim and Rush, 2016)



[Google unleashes deep learning tech on language with **Neural** ...]

TechCrunch - Sep 27, 2016

Google has been working on a **machine** learning **translation** technique for years, and today is its official debut. The Google **Neural Machine** ...

Google **Translate** now converts Chinese into English with **neural** ...

VentureBeat - Sep 27, 2016

Google announces **Neural Machine Translation**

The Stack - Sep 28, 2016

Google announces **Neural Machine Translation** to improve Google ...

Highly Cited - ZDNet - Sep 27, 2016

Google is using **Neural** Networks for Chinese to English **machine** ...

Opinion - Firstpost - Sep 28, 2016

Google announces **neural** network to improve **machine translation**

In-Depth - Seeking Alpha - Sep 27, 2016

ZDNet VentureBeat The Stack Geektime Ubergizmo Science Mag...

View all

SYSTRAN: 1st software provider to launch a **Neural Machine** ...

GlobeNewswire (press release) - Oct 17, 2016

In December, SYSTRAN will communicate the feedback received on Pure **Neural** TM **Machine Translation**, its roadmap and time to market plan ...

Iconic Integrates Custom **Neural Machine Translation** Into ...

Slator (press release) (subscription) - Oct 6, 2016

Dublin – October 6, 2016 – Iconic **Translation** Machines (Iconic), a leading Irish **machine translation** (MT) software and solutions provider, today ...

Neural Machine Translation

Excellent results on many language pairs, but need large models

- Original seq2seq paper (Sutskever et al., 2014a): 4-layers/1000 units
- Deep Residual RNNs (Zhou et al., 2016) : 16-layers/512 units
- Google's NMT system (Wu et al., 2016): 8-layers/1024 units

Beam search + ensemble on top

$\Longrightarrow$ Deployment is challenging!

Neural Machine Translation

Excellent results on many language pairs, but need large models

- Original seq2seq paper (Sutskever et al., 2014a): 4-layers/1000 units
- Deep Residual RNNs (Zhou et al., 2016) : 16-layers/512 units
- Google's NMT system (Wu et al., 2016): 8-layers/1024 units

Beam search + ensemble on top

$\Longrightarrow$ Deployment is challenging!

Related Work: Compressing Deep Models

- **Pruning**: Prune weights based on importance criterion (LeCun et al., 1990; Han et al., 2016; See et al., 2016)
- **Knowledge Distillation**: Train a *student* model to learn from a *teacher* model (Bucila et al., 2006; Ba and Caruana, 2014; Hinton et al., 2015; Kuncoro et al., 2016). (Sometimes called "dark knowledge")

Knowledge Distillation (Bucila et al., 2006; Hinton et al., 2015)

- Train a *larger teacher* model first to obtain teacher distribution $q(\cdot)$
- Train a *smaller student* model $p(\cdot)$ to mimic the teacher

Word-Level Knowledge Distillation

Teacher distribution: $q(w_t \mid y_{1:t-1})$

$$\mathcal{L}_{\text{NLL}} = -\sum_t \sum_{k \in \mathcal{V}} \mathbb{1}\{y_t = k\} \log p(w_t = k \mid y_{1:t-1}; \theta)$$

$$\mathcal{L}_{\text{WORD-KD}} = -\sum_t \sum_{k \in \mathcal{V}} q(w_t = k \mid y_{1:t-1}) \log p(w_t = k \mid y_{1:t-1}; \theta)$$

Knowledge Distillation (Bucila et al., 2006; Hinton et al., 2015)

- Train a *larger teacher* model first to obtain teacher distribution $q(\cdot)$
- Train a *smaller student* model $p(\cdot)$ to mimic the teacher

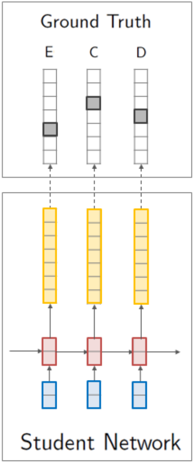
Word-Level Knowledge Distillation

Teacher distribution: $q(w_t \mid y_{1:t-1})$

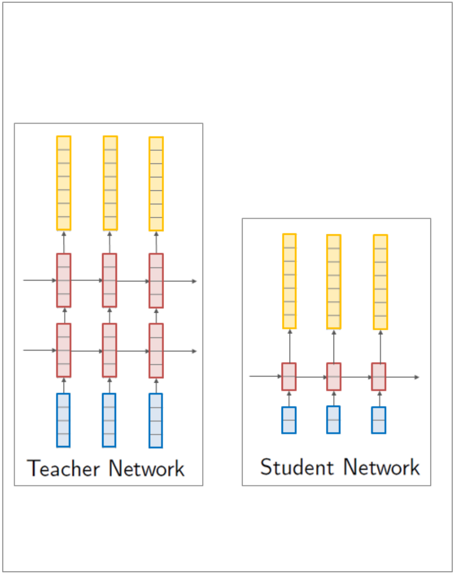
$$\mathcal{L}_{\text{NLL}} = -\sum_{t} \sum_{k \in \mathcal{V}} \mathbb{1}\{y_t = k\} \log p(w_t = k \mid y_{1:t-1}; \theta)$$

$$\mathcal{L}_{\text{WORD-KD}} = -\sum_{t} \sum_{k \in \mathcal{V}} q(w_t = k \mid y_{1:t-1}) \log p(w_t = k \mid y_{1:t-1}; \theta)$$

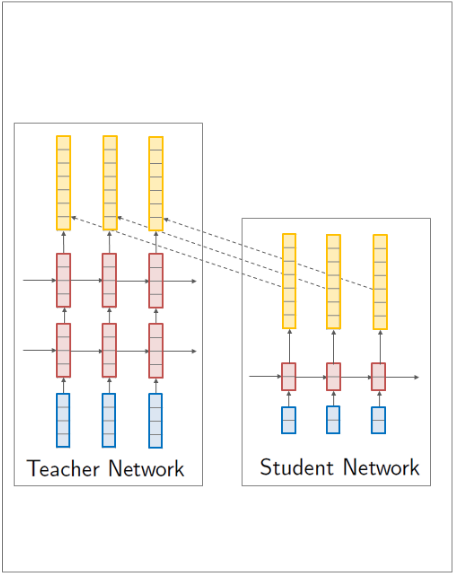
No Knowledge Distillation



Word-Level Knowledge Distillation



Word-Level Knowledge Distillation



Word-Level Knowledge Distillation Results

English $\rightarrow$ German (WMT 2014)

Model	BLEU
4×1000 Teacher	19.5
2×500 Baseline (No-KD)	17.6
2×500 Student (Word-KD)	17.7
2×300 Baseline (No-KD)	16.9
2×300 Student (Word-KD)	17.6

This Work: Sequence-Level Knowledge Distillation

$$\mathcal{L}_{\text{NLL}} = -\sum_{t} \sum_{k \in \mathcal{V}} \mathbb{1}\{y_t = k\} \log p(w_t = k \mid y_{1:t-1})$$

$$\mathcal{L}_{\text{WORD-KD}} = -\sum_{t} \sum_{k \in \mathcal{V}} q(w_t = k \mid y_{1:t-1}) \log p(w_t = k \mid y_{1:t-1})$$

Instead minimize cross-entropy, between q and p implied *sequence*-distributions

$$\mathcal{L}_{\text{SEQ-KD}} = -\sum_{w_{1:T} \in \mathcal{V}^T} q(w_{1:T} \mid x) \log p(w_{1:T} \mid x)$$

Sum over an exponentially-sized set $\mathcal{V}^T$.

This Work: Sequence-Level Knowledge Distillation

$$\mathcal{L}_{\text{NLL}} = -\sum_{t}\sum_{k\in\mathcal{V}} \mathbb{1}\{y_t = k\} \log p(w_t = k \mid y_{1:t-1})$$

$$\mathcal{L}_{\text{WORD-KD}} = -\sum_{t}\sum_{k\in\mathcal{V}} q(w_t = k \mid y_{1:t-1}) \log p(w_t = k \mid y_{1:t-1})$$

Instead minimize cross-entropy, between q and p implied *sequence*-distributions

$$\mathcal{L}_{\text{SEQ-KD}} = -\sum_{w_{1:T}\in\mathcal{V}^T} q(w_{1:T} \mid x) \log p(w_{1:T} \mid x)$$

Sum over an exponentially-sized set $\mathcal{V}^T$.

Sequence-Level Knowledge Distillation

Approximate $q(w \,|\, x)$ with mode

$$q(w_{1:T} \,|\, x) \approx \mathbb{1}\{\arg\max_{w_{1:T}} q(w_{1:T} \,|\, x)\}$$

Approximate mode with beam search

$$\hat{y} \approx \arg\max_{w_{1:T}} q(w_{1:T} \,|\, x)$$

Simple model: train the student model on $\hat{y}$ with NLL

Sequence-Level Knowledge Distillation

Approximate $q(w \mid x)$ with mode

$$q(w_{1:T} \mid x) \approx \mathbb{1}\{\arg\max_{w_{1:T}} q(w_{1:T} \mid x)\}$$

Approximate mode with beam search

$$\hat{y} \approx \arg\max_{w_{1:T}} q(w_{1:T} \mid x)$$

Simple model: train the student model on $\hat{y}$ with NLL

Sequence-Level Knowledge Distillation

Approximate $q(w \mid x)$ with mode

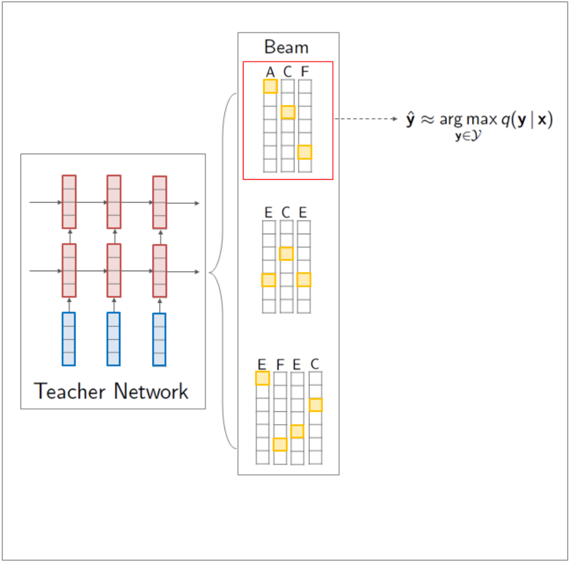
$$q(w_{1:T} \mid x) \approx \mathbb{1}\{\arg\max_{w_{1:T}} q(w_{1:T} \mid x)\}$$

Approximate mode with beam search

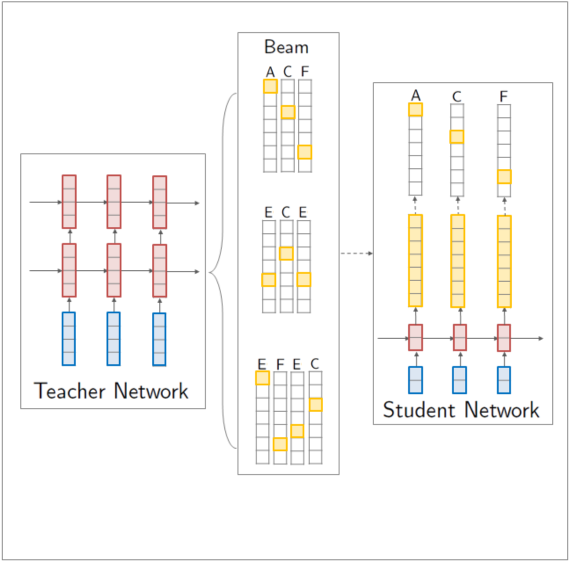
$$\hat{y} \approx \arg\max_{w_{1:T}} q(w_{1:T} \mid x)$$

Simple model: train the student model on $\hat{y}$ with NLL

Sequence-Level Knowledge Distillation



Sequence-Level Knowledge Distillation



Sequence-Level Interpolation

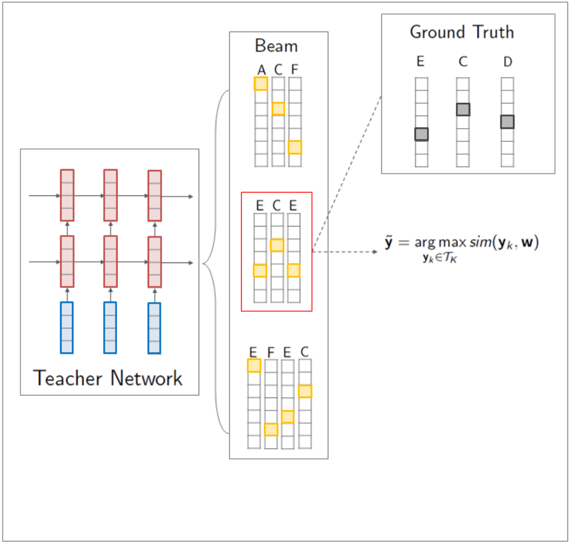
Word-level knowledge distillation

$$\mathcal{L} = \alpha \mathcal{L}_{\text{WORD-KD}} + (1 - \alpha) \mathcal{L}_{\text{NLL}}$$

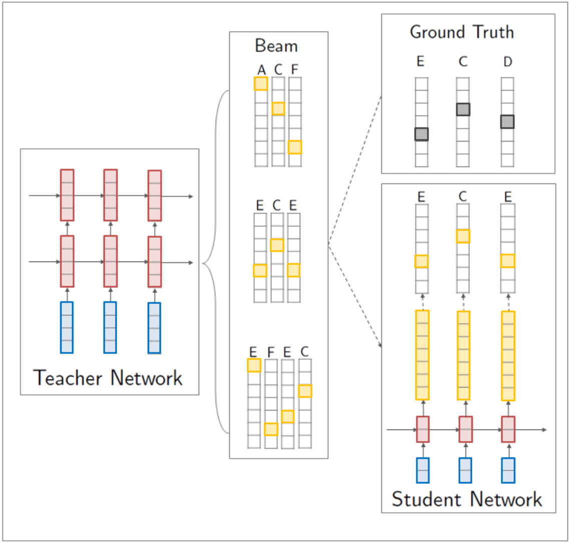
Training the student towards the mixture of teacher/data distributions.

How can we incorporate ground truth data at the sequence-level?

Sequence-Level Interpolation



Sequence-Level Interpolation



Experiments on English $\rightarrow$ German (WMT 2014)

- Word-KD: Word-level Knowledge Distillation
- Seq-KD: Sequence-level Knowledge Distillation with beam size $K = 5$
- Seq-Inter: Sequence-level Interpolation with beam size $K = 35$. Fine-tune from pretrained Seq-KD (or baseline) model with smaller learning rate.

Results: English $\rightarrow$ German (WMT 2014)

Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	−	19.5	−	6.7	1.3%
2×500						
Student	14.7	−	17.6	−	8.2	0.9%

Results: English $\rightarrow$ German (WMT 2014)

Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	–	19.5	–	6.7	1.3%
2×500						
Student	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%

Results: English $\rightarrow$ German (WMT 2014)

Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	–	19.5	–	6.7	1.3%
2×500						
Student	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%
Seq-KD	18.9	+4.2	19.0	+1.4	22.7	16.9%

Results: English $\rightarrow$ German (WMT 2014)

Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	–	19.5	–	6.7	1.3%
2×500						
Student	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%
Seq-KD	18.9	+4.2	19.0	+1.4	22.7	16.9%
Seq-Inter	18.9	+4.2	19.3	+1.7	15.8	7.6%

Results: English $\rightarrow$ German (WMT 2014)

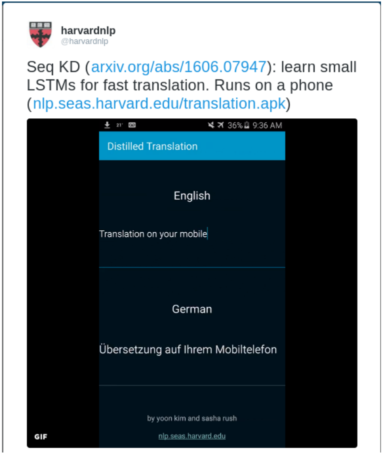
Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	–	19.5	–	6.7	1.3%
Seq-Inter	19.6	+1.9	19.8	+0.3	10.4	8.2%
2×500						
Student	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%
Seq-KD	18.9	+4.2	19.0	+1.4	22.7	16.9%
Seq-Inter	18.9	+4.2	19.3	+1.7	15.8	7.6%

Results: English $\to$ German (WMT 2014)

Model	BLEU$_{K=1}$	$\Delta_{K=1}$	BLEU$_{K=5}$	$\Delta_{K=5}$	PPL	$p(\hat{\mathbf{y}})$
4×1000						
Teacher	17.7	–	19.5	–	6.7	1.3%
Seq-Inter	19.6	+1.9	19.8	+0.3	10.4	8.2%
2×500						
Student	14.7	–	17.6	–	8.2	0.9%
Word-KD	15.4	+0.7	17.7	+0.1	8.0	1.0%
Seq-KD	18.9	+4.2	19.0	+1.4	22.7	16.9%
Seq-Inter	18.9	+4.2	19.3	+1.7	15.8	7.6%

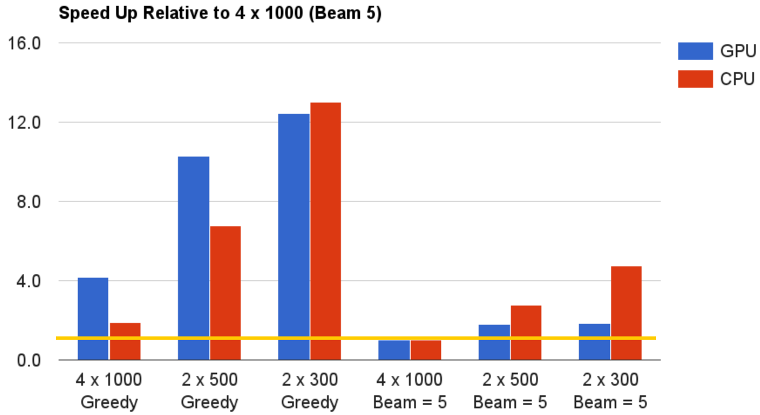
Many more experiments (different language pairs, combining configurations, different sizes etc.) in paper

An Application



[App]

Decoding Speed



Combining Knowledge Distillation and Pruning

Number of parameters still large for student models (mostly due to word embedding tables)

- 4×1000: 221 million
- 2×500: 84 million
- 2×300: 49 million

Prune student model: Same methodology as See et al. (2016)

- Prune $x\%$ of weights based on absolute value
- Fine-tune pruned model (crucial!)

Combining Knowledge Distillation and Pruning

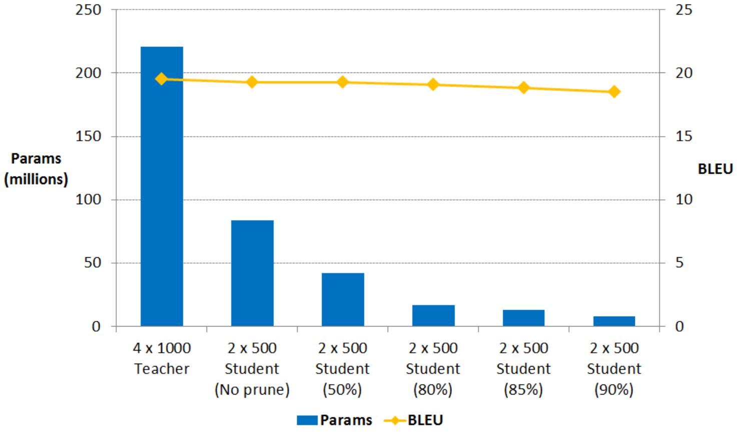
Number of parameters still large for student models (mostly due to word embedding tables)

- 4×1000: 221 million
- 2×500: 84 million
- 2×300: 49 million

Prune student model: Same methodology as See et al. (2016)

- Prune $x\%$ of weights based on absolute value
- Fine-tune pruned model (crucial!)

Combining Knowledge Distillation and Pruning

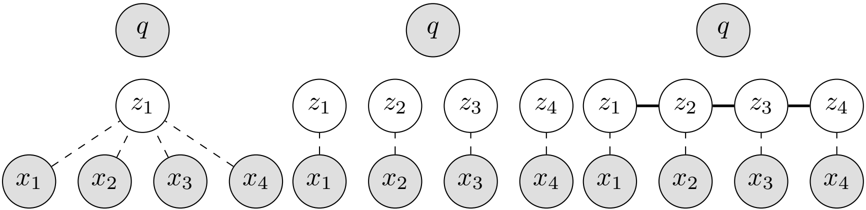


Conclusion: Other work

- How can we **interpret** these learned hidden representations?
  - Lei et al. (2016) other methods for interpreting decisions (as opposed to states).
- How should we **train** these style of models?
  - Lee et al. (2016) CCG parsing (backprop through search is a thing now/again)
- How can we **shrink** these models for practical applications?
  - Live deployment: (greedy) student outperforms (beam search) teacher. (Crego et al., 2016)
  - Can compress an ensemble into a single model (Kuncoro et al., 2016)

Coming Work

- Structured Attention Networks (Kim et al 2016)



(a) (b) (c)

Thanks!

References I

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally Normalized Transition-Based Neural Networks. *arXiv*, cs.CL.

Ba, L. J. and Caruana, R. (2014). Do Deep Nets Really Need to be Deep? In *Proceedings of NIPS*.

Bahdanau, D., Brakel, P., Xu, K., Goyal, A., Lowe, R., Pineau, J., Courville, A., and Bengio, Y. (2016). An Actor-Critic Algorithm for Sequence Prediction.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179.

References II

Bucila, C., Caruana, R., and Niculescu-Mizil, A. (2006). Model Compression. In *Proceedings of KDD*.

Chan, W., Jaitly, N., Le, Q., and Vinyals, O. (2015). Listen, Attend and Spell. *arXiv:1508.01211*.

Cheng, J. and Lapata, M. (2016). Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of EMNLP*.

Chorowski, J., Bahdanau, D., and Serdyuk, D. (2015). Attention-based models for speech recognition. *Advances in Neural*.

Crego, J., Kim, J., and Senellart, J. (2016). Systran's pure neural machine translation system. *arXiv preprint arXiv:1602.06023*.

References III

Daudaravicius, V., Banchs, R. E., Volodina, E., and Napoles, C. (2016). A Report on the Automatic Evaluation of Scientific Writing Shared Task. *NAACL BEA11 Workshop*, pages 53–62.

Daumé III, H. and Marcu, D. (2005). Learning as search optimization: approximate large margin methods for structured prediction. In *Proceedings of the Twenty-Second International Conference on Machine Learning* {*(ICML*} *2005)*, pages 169–176.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Gillick, D., Brunk, C., Vinyals, O., and Subramanya, A. (2016). Multilingual Language Processing from Bytes. In *Proceedings of NAACL*.

References IV

Han, S., Mao, H., and Dally, W. J. (2016). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *Proceedings of ICLR*.

Hermann, K., Kocisky, T., and Grefenstette, E. (2015). Teaching machines to read and comprehend. *Advances in Neural*.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv:1503.0253*.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *EMNLP*, pages 1700–1709.

Karpathy, A., Johnson, J., and Li, F.-F. (2015). Visualizing and understanding recurrent networks. *ICLR Workshops*.

Karpathy, A. and Li, F.-F. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.

References V

Kim, Y. and Rush, A. M. (2016). Sequence-Level Knowledge Distillation.

Kuncoro, A., Ballesteros, M., Kong, L., Dyer, C., and Smith, N. A. (2016). Distilling an Ensemble of Greedy Dependency Parsers into One MST Parser. In *Proceedings of EMNLP*.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289.

LeCun, Y., Denker, J. S., and Solla, S. A. (1990). Optimal Brain Damage. In *Proceedings of NIPS*.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*, number September, page 11.

References VI

Mou, L., Yan, R., Li, G., Zhang, L., and Jin, Z. (2015). Backward and forward language modeling for constrained sentence generation. *arXiv preprint arXiv:1512.06612*.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. (2016). Sequence Level Training with Recurrent Neural Networks. *ICLR*, pages 1–15.

Rush, A. M., Chopra, S., and Weston, J. (2015). A Neural Attention Model for Abstractive Sentence Summarization. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (September):379–389.

Schmaltz, A., Kim, Y., Rush, A. M., and Shieber, S. M. (2016). Sentence-Level Grammatical Error Identification as Sequence-to-Sequence Correction.

See, A., Luong, M.-T., and Manning, C. D. (2016). Compression of Neural Machine Translation via Pruning. In *Proceedings of CoNLL*.

References VII

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2016). Building End-to-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of AAAI*.

Srivastava, N., Mansimov, E., and Salakhutdinov, R. (2015). Unsupervised Learning of Video Representations using LSTMs. *Proceedings of ICML*.

Strobelt, H., Gehrmann, S., Huber, B., Pfister, H., and Rush, A. M. (2016). Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks.

Sutskever, I., Vinyals, O., and Le, Q. (2014a). Sequence to Sequence Learning with Neural Networks.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014b). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Takase, S., Suzuki, J., Okazaki, N., Hirao, T., and Nagata, M. (2016). Neural headline generation on abstract meaning representation.

References VIII

Toutanova, K., Tran, K. M., and Amershi, S. (2016). A dataset and evaluation metrics for abstractive compression of sentences and short paragraphs.

Venkatraman, A., Boots, B., Hebert, M., and Bagnell, J. (2015). DATA AS DEMONSTRATOR with Applications to System Identification. *pdfs.semanticscholar.org*.

Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2014). Grammar as a Foreign Language. In *arXiv*, pages 1–10.

Vinyals, O. and Le, Q. (2015). A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and Tell: A Neural Image Caption Generator. In *Proceedings of CVPR*.

Wang, S., Han, S., and Rush, A. M. (2016a). Headliner. *Computation+Journalism*.

References IX

Wang, T., Chen, P., Amaral, K., and Qiang, J. (2016b). An experimental study of lstm encoder-decoder model for text simplification. *arXiv preprint arXiv:1609.03663*.

Wiseman, S. and Rush, A. M. (2016). Sequence-to-Sequence Learning as Beam-Search Optimization.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv:1606.09.08144*.

References X

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *ICML*.

Zhou, J., Cao, Y., Wang, X., Li, P., and Xu, W. (2016). Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation. In *Proceedings of TACL*.