Discriminative and Neural Translation Models

July 2, 2015
Noisy Channel MT

\[ p(e) \]

source  \[\rightarrow\] English
Noisy Channel MT

$p(e)$

source

$\rightarrow$ English

$p(g \mid e)$

$\rightarrow$ German
Noisy Channel MT

\[ p(e) \]

\[
\begin{align*}
    e^* &= \arg \max_e p(e | g) \\
    &= \arg \max_e \frac{p(g | e) \times p(e)}{p(g)} \\
    &= \arg \max_e p(g | e) \times p(e)
\end{align*}
\]
Noisy Channel MT

\[ e^* = \arg \max_e p(e \mid g) \]

\[ = \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)} \]

\[ = \arg \max_e p(g \mid e) \times p(e) \]
And Beyond...

\[
e^* = \arg \max_e p(e \mid g)
\]

\[
= \arg \max_e \frac{p(g \mid e) \times p(e)}{p(g)}
\]

\[
= \arg \max_e p(g \mid e) \times p(e)
\]

\[
= \arg \max_e \log p(g \mid e) + \log p(e)
\]
And Beyond…

\[ e^* = \arg \max \limits_{e} p(e \mid g) \]

\[ = \arg \max \limits_{e} \frac{p(g \mid e) \times p(e)}{p(g)} \]

\[ = \arg \max \limits_{e} p(g \mid e) \times p(e) \]

Does this look familiar?

\[ = \arg \max \limits_{e} \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} \log p(g \mid e) \\ \log p(e) \end{bmatrix} \begin{bmatrix} w^T \\ h(g,e) \end{bmatrix} \]
MT Linear Model

$-\log p(g|e)$

$-\log p(e)$
MT Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
MT Linear Model

\[-\log p(g \mid e)\]

\[-\log p(e)\]
MT Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
MT Linear Model

-\log p(g|e)

Improvement 1:

change \( \vec{w} \) to find better translations
MT Linear Model

$-\log p(g|e)$

$\overrightarrow{\vec{w}}$

$-\log p(e)$
MT Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
MT Linear Model

\[-\log p(g|e)\]

\[-\log p(e)\]
MT Linear Model

Improvement 2:

Add dimensions to make points separable
MT Linear Model

\[ e^* = \arg \max_{e} w^\top h(g, e) \]

• Improve the modeling capacity of the noisy channel in two ways
  • Reorient the weight vector *(parameter optimization)*
  • Add new dimensions *(new features)*

• Questions
  • What features? \[ h(g, e) \]
  • How do we set the weights? \[ w \]
What Features?

• Two solutions

• **Engineer features**
  As a language expert, what do I think is likely to be useful for distinguishing good from bad translations?

• **Learn features**
  Let data decide what features should be used to solve the problem.
What are Features?

• A **feature function** is a function from objects to a real values

• A “good” feature function will map **inputs that behave similarly** (from the task perspective) into similar values

• Example feature functions:
  
  - the number of verbs in the translation?
  - 1 if the first word is capitalized, 0 otherwise
  - the log language model probability of a sequence
horse
horse
$\mathbf{v}_{\text{horse}}$
\[ w_{\text{horse}} = P v_{\text{horse}} \]
$w_{horse} = P v_{horse}$
Words as Vectors
Words as Vectors

\[
\begin{align*}
\mathbf{w}_{\text{train}} & \quad \mathbf{w}_{\text{truck}} \\
\mathbf{w}_{\text{car}} & \quad \mathbf{w}_{\text{horse}} \\
\mathbf{w}_{\text{dog}} & \quad \mathbf{w}_{\text{cat}} \\
\mathbf{w}_{\text{translation}} &
\end{align*}
\]
Words as Vectors

$w_{\text{train}}$, $w_{\text{truck}}$, $w_{\text{car}}$, $w_{\text{horse}}$, $w_{\text{dog}}$, $w_{\text{cat}}$, $w_{\text{translation}}$
Features for MT

- For MT we want to represent translations (not words) as vectors.

- The space of translations (not just good ones, but bad ones too!) is huge, so designing this vector space is huge.

- The noisy channel says: “let’s represent translations as two-dimensional vectors.”

- Until a year ago, MT was pretty close to that…
Features for MT

• Target side features
  • log p(e)  [ n-gram language model ]
  • Number of words in hypothesis
  • Non-English character count

• Source + target features
  • log relative frequency e|f of each rule  [ log #(e,f) - log #(f) ]
  • log relative frequency f|e of each rule  [ log #(e,f) - log #(e) ]
  • “lexical translation” log probability e|f of each rule  [ ≈ log p_{model1}(e|f) ]
  • “lexical translation” log probability f|e of each rule  [ ≈ log p_{model1}(f|e) ]

• Other features
  • Count of rules/phrases used
  • Reordering pattern probabilities
Computing Features: Locality

• **Local features**
  *Feature values only depend on rule*

• **Source-contextual features**
  *Feature values depend on rule’s target side and the entire source document*

• **Non-local features**
  *Feature values depend on other translation decisions used*
  
  *Example: (n-gram) language model*
Why do we care?

- Searching for the best solution with local and source-contextual features is inexpensive
  - Use dynamic programming / shortest path
- Non-local features effectively mean that decisions you make “early” in the search affect how much each decision later will cost
  - NP-hard maximization problem in general
- You will get to (attempt to) solve this in the lab today
Parameter Learning
Hypothesis Space
Hypothesis Space
We assume a decoder that computes:

$$\langle e^*, a^* \rangle = \arg \max_{(e,a)} w^\top h(g, e, a)$$
We assume a decoder that computes:
\[ \langle e^*, a^* \rangle = \arg \max_{\langle e,a \rangle} w^T h(g, e, a) \]

And \( K \)-best lists of, that is:
\[ \{ \langle e^*_i, a^*_i \rangle \}_{i=1}^{i=K} = \arg \max_{\langle e,a \rangle} i^{th} \] Standard algorithms exist for both of these.
Learning Weights

- Try to match the reference translation exactly
- Conditional random field
  - Maximize the score of the reference translations
  - “Average” over the different ways of producing the same translation variables
Problems

• These methods give “full credit” when the model \textit{exactly} produces the reference and no credit otherwise

• What is the problem with this?
Cost-Sensitive Training

• Assume we have a cost function that gives a score for how good/bad a translation is

\[ \ell(\hat{e}, E) \mapsto [0, 1] \]

• Optimize the weight vector by making reference to this function
K-Best List Example
K-Best List Example

\[ h_1 \]

\[ h_2 \]

- \[ 0.8 \leq \ell < 1.0 \]
- \[ 0.6 \leq \ell < 0.8 \]
- \[ 0.4 \leq \ell < 0.6 \]
- \[ 0.2 \leq \ell < 0.4 \]
- \[ 0.0 \leq \ell < 0.2 \]
Training as Classification

- **Pairwise Ranking Optimization**
  - Reduce training problem to *binary classification* with a linear model

- **Algorithm**
  - For $i = 1$ to $N$
    - Pick random pair of hypotheses (A,B) from $K$-best list
    - Use cost function to determine if is A or B better
    - Create $i$th training instance
  - Train binary linear classifier
The diagram shows a scatter plot with two axes, $h_1$ and $h_2$. The points are color-coded based on the range of their $\ell$ values:

- **Red**: $0.8 \leq \ell < 1.0$
- **Dark Red**: $0.6 \leq \ell < 0.8$
- **Orange**: $0.4 \leq \ell < 0.6$
- **Green**: $0.2 \leq \ell < 0.4$
- **Light Green**: $0.0 \leq \ell < 0.2$

The points labeled #1 to #10 represent different data points in the scatter plot.
Worse!

- 0.8 ≤ ℓ < 1.0
- 0.6 ≤ ℓ < 0.8
- 0.4 ≤ ℓ < 0.6
- 0.2 ≤ ℓ < 0.4
- 0.0 ≤ ℓ < 0.2
Worse!

- 0.8 ≤ ℓ < 1.0
- 0.6 ≤ ℓ < 0.8
- 0.4 ≤ ℓ < 0.6
- 0.2 ≤ ℓ < 0.4
- 0.0 ≤ ℓ < 0.2
Better!
Better!

- $0.8 \leq \ell < 1.0$
- $0.6 \leq \ell < 0.8$
- $0.4 \leq \ell < 0.6$
- $0.2 \leq \ell < 0.4$
- $0.0 \leq \ell < 0.2$
Worse!

- 0.8 ≤ ℓ < 1.0
- 0.6 ≤ ℓ < 0.8
- 0.4 ≤ ℓ < 0.6
- 0.2 ≤ ℓ < 0.4
- 0.0 ≤ ℓ < 0.2
Better!
Fit a linear model
Fit a linear model
K-Best List Example

\[ w \]

- \( 0.8 \leq \ell < 1.0 \)
- \( 0.6 \leq \ell < 0.8 \)
- \( 0.4 \leq \ell < 0.6 \)
- \( 0.2 \leq \ell < 0.4 \)
- \( 0.0 \leq \ell < 0.2 \)
Why do this?

Table 2: Effect of maximum entropy training for alignment template approach (WP: word penalty feature, CLM: class-based language model (five-gram), MX: conventional dictionary).

<table>
<thead>
<tr>
<th></th>
<th>objective criteria [%]</th>
<th>subjective criteria [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SER</td>
<td>WER</td>
</tr>
<tr>
<td>Baseline($\lambda_m = 1$)</td>
<td>86.9</td>
<td>42.8</td>
</tr>
<tr>
<td>ME</td>
<td>81.7</td>
<td>40.2</td>
</tr>
<tr>
<td>ME+WP</td>
<td>80.5</td>
<td>38.6</td>
</tr>
<tr>
<td>ME+WP+CLM</td>
<td>78.1</td>
<td>38.3</td>
</tr>
<tr>
<td>ME+WP+CLM+MX</td>
<td>77.8</td>
<td>38.4</td>
</tr>
</tbody>
</table>

Discriminative
Questions?
What Features?

• Two solutions

• **Engineer features**
  As a language expert, what do I think is likely to be useful for distinguishing good from bad translations?

• **Learn features**
  Let data decide what features should be used to solve the problem.
  We will use neural networks for this.
Learning Features with Neural Networks

• Neural networks take very “small” features (e.g., one word, one pixel, one slice of spectrogram) and compose them into representations of larger, more abstract units

• Compute a representation of a sentence from words

• Compute a representation of the contents of an image from pixels
Bengio et al. (2003)

\[
p(e) = \prod_{i=1}^{|e|} p(e_i | e_{i-n+1}, \ldots, e_{i-1})
\]

\[
p(e_i | e_{i-n+1}, \ldots, e_{i-1}) =
\]

\[
e_{i-1}
\]
Bengio et al. (2003)

\[
p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1})
\]

\[
p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \begin{pmatrix} c \end{pmatrix}
\]
Bengio et al. (2003)

\[
p(e) = \prod_{i=1}^{\mid e \mid} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1})
\]

\[
p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) =
\]

\[
te_{i-1} \begin{bmatrix} \mathbf{C} \end{bmatrix} \mathbf{c}_{e_{i-1}}
\]

“word embedding”
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{\vert e \vert} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[
\begin{array}{ccc}
\text{C} & \text{C} & \text{C} \\
\text{C} & \text{C} & \text{C} \\
\end{array}
\]

\[ c_{e_{i-1}} \]

\[ c_{e_{i-2}} \]
Bengio et al. (2003)

\[
p(e) = \prod_{i=1}^{\vert e \vert} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1})
\]

\[
p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = 
\]

\[
\begin{array}{ccc}
e_{i-1} & \mathbf{C} & c_{e_{i-1}} \\
e_{i-2} & \mathbf{C} & c_{e_{i-2}} \\
e_{i-3} & \mathbf{C} & c_{e_{i-3}}
\end{array}
\]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[ \begin{array}{ccc}
\cdots & \mathbf{C} & \cdots \\
\vdots & \vdots & \vdots \\
\cdot & \cdot & \cdot \\
 e_{i-3} & \mathbf{C} & e_{i-3} \\
 e_{i-2} & \mathbf{C} & e_{i-2} \\
 e_{i-1} & \mathbf{C} & e_{i-1}
\end{array} \]

\[ x=x \]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

![Diagram showing the relationship between \( e_{i-1} \), \( e_{i-2} \), \( e_{i-3} \), and \( X = x \).]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{\lvert e \rvert} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[ h = \tanh(Wc_e) \]
Bengio et al. (2003)

\[
p(e) = \prod_{i=1}^{\left|e\right|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1})
\]

\[
p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) =
\]

\[
h = \tanh(Wc_e + b)
\]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[ \text{tanh} \]

\[ x=x \]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[ x = x \]

\[ \text{tanh} \]

\[ \text{softmax} \]
Bengio et al. (2003)

\[ p(e) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) \]

\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

![Diagram](image)
\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]
\[ = \frac{\exp \mathbf{v}^\top_{e_i} h(e_{i-1}, e_{i-2}, e_{i-3})}{\sum_{e' \in V} \exp \mathbf{v}^\top_{e'} h(e_{i-1}, e_{i-2}, e_{i-3})} \]
\[ p(e_i \mid e_{i-n+1}, \ldots, e_{i-1}) = \]

\[
\begin{align*}
\frac{\exp \mathbf{v}^\top \mathbf{h}(e_{i-1}, e_{i-2}, e_{i-3})}{\sum_{e' \in V} \exp \mathbf{v}^\top_{e'} \mathbf{h}(e_{i-1}, e_{i-2}, e_{i-3})}
\end{align*}
\]

\textit{h is a feature function that extracts a vector of features from a trigram of words.}
What Features?

- Two solutions

- **Engineer features**
  As a language expert, what do I think is likely to be useful for distinguishing good from bad translations?

- **Learn features (induce features)**
  Let data decide what features should be used to solve the problem.
  We will use neural networks for this.
Learning Features

• Neural networks are, of course, nonlinear models with particular parametric forms

• But the view that they induce features is important because while tasks do different things, we often use the same features in many tasks

  • Key opportunity: some tasks have a lot more data to learn features from

  • Multitask learning: learn the parameters for many tasks jointly

• The go-to “auxiliary” task for language is: language modeling
Devlin et al. (2014)

• Turn Bengio et al. (2003) into a translation model

• Conditional model; generate the next English word conditioned on
  • The previous $n$ English words you generated
  • The aligned source word, and its $m$ neighbors
\[ p(e \mid f, a) = \prod_{i=1}^{|e|} p(e_i \mid e_{i-2}, e_{i-1}, f_{a_i-1}, f_{a_i}, f_{a_i+1}) \]

\[ p(e_i \mid e_{i-2}, e_{i-1}, f_{a_i-1}, f_{a_i}, f_{a_i+1}) = \]

**Source alignment & its context**
(m=4)

**Target context**
(n=4)
Devlin et al. (2014)

<table>
<thead>
<tr>
<th>BOLT Test</th>
<th>Ar-En</th>
<th>BLEU</th>
<th>% Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Simple Hier.” Baseline</td>
<td></td>
<td>33.8</td>
<td></td>
</tr>
<tr>
<td>S2T/L2R NNJM (Dec)</td>
<td></td>
<td>38.4</td>
<td>100%</td>
</tr>
<tr>
<td>Source Window=7</td>
<td></td>
<td>38.3</td>
<td>98%</td>
</tr>
<tr>
<td>Source Window=5</td>
<td></td>
<td>38.2</td>
<td>96%</td>
</tr>
<tr>
<td>Source Window=3</td>
<td></td>
<td>37.8</td>
<td>87%</td>
</tr>
<tr>
<td>Source Window=0</td>
<td></td>
<td>35.3</td>
<td>33%</td>
</tr>
<tr>
<td>Layers=384x768x768</td>
<td></td>
<td>38.5</td>
<td>102%</td>
</tr>
<tr>
<td>Layers=192x512</td>
<td></td>
<td>38.1</td>
<td>93%</td>
</tr>
<tr>
<td>Layers=128x128</td>
<td></td>
<td>37.1</td>
<td>72%</td>
</tr>
<tr>
<td>Vocab=64,000</td>
<td></td>
<td>38.5</td>
<td>102%</td>
</tr>
<tr>
<td>Vocab=16,000</td>
<td></td>
<td>38.1</td>
<td>93%</td>
</tr>
<tr>
<td>Vocab=8,000</td>
<td></td>
<td>37.3</td>
<td>83%</td>
</tr>
<tr>
<td>Activation=Rectified Lin.</td>
<td></td>
<td>38.5</td>
<td>102%</td>
</tr>
<tr>
<td>Activation=Linear</td>
<td></td>
<td>37.3</td>
<td>76%</td>
</tr>
</tbody>
</table>
Neural Translation

- We know how to use neural networks to condition on unbounded history (e.g., RNNs, LSTMs) without engineering features (e.g., $n$-grams)

I → drank → a →
Neural Translation

• We know how to use neural networks to condition on unbounded history (e.g., RNNs, LSTMs) without engineering features (e.g., \(n\)-grams)

• Translation models can be thought of as a “conditional” language model; i.e., a sequence model that conditions on its past + a source sentence
Neural Translation

• We know how to use neural networks to condition on unbounded history (e.g., RNNs, LSTMs) without engineering features (e.g., $n$-grams)

• Translation models can be thought of as a “conditional” language model; i.e., a sequence model that conditions on its past + a source sentence
Neural Translation

- We know how to use neural networks to condition on unbounded history (e.g., RNNs, LSTMs) without engineering features (e.g., $n$-grams).

- Translation models can be thought of as a “conditional” language model; i.e., a sequence model that conditions on its past + a source sentence.

```
Ich trank eine Cola
```

```
I drank a Coke
```
Neural MT

- Two big questions:
  - How do we represent the sentence / pieces of the sentence?
  - How do we influence the prediction of the language model component?
The sun is shining

"Additive" architecture
Encoding Sentences

"Recurrent" architecture

Sutskever et al.
Encoding Sentences

“Recursive” architecture

Socher et al.
Encoding Sentences

"Convolutional" architecture

The sun is shining

Blunsom et al.
Decoding Vectors

- High level decision
  - The input is represented as a single vector
  - The representation of the input evolves over time (usually computed via some “attention” mechanism)

- Many low-level decisions
  - Do we just have two inputs to the RNN at each time step?
  - Do we use the input to “gate” the RNN recurrences?
Figure from Mei, Bansal, and Walter (2015)
Figure from Mei, Bansal, and Walter (2015)
Figure from Mei, Bansal, and Walter (2015)
(a) Encoder-Decoder

(b) LSTM Encoder

(c) LSTM Decoder

Figure from Mei, Bansal, and Walter (2015)
Kalchbrenner & Blunsom (2013)
Kalchbrenner & Blunsom (2013)
may i have a wake-up call at seven tomorrow morning?

CLM

II

明天 早上 七点 叫醒 我 好 吗 ?
I'd like to have a room under thirty dollars a night.

我想要一晚三十美元以下的房间。
I want a late thirties under $’s room.

I'd like to have a room under thirty dollars a night.

我想要一晚三十美元以下的房间。
you have to do something about it.

不 想想 办法 的 话 我 会 为难的。
i can't urinate.

CLM

不 想想 办法 的 话 我 会 为难的。
# Encoder/Decoder model

## English-German WMT2015

<table>
<thead>
<tr>
<th>System</th>
<th>Submitter</th>
<th>System Notes</th>
<th>Constraint</th>
<th>Run Notes</th>
<th>BLEU</th>
<th>BLEU (11b)</th>
<th>BLEU-cased</th>
<th>BLEU-cased (11b)</th>
<th>TER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural MT (contrastive) (Details)</td>
<td>UMontreal-MILA University of Montreal</td>
<td>Neural machine translation + 5-gram LM (re-ranking)</td>
<td>yes</td>
<td></td>
<td>25.2</td>
<td>25.2</td>
<td>24.9</td>
<td>24.9</td>
<td>0.665</td>
</tr>
<tr>
<td>Neural MT (primary) (Details)</td>
<td>UMontreal-MILA University of Montreal</td>
<td>Neural machine translation (primary) without monolingual corpus</td>
<td>yes</td>
<td>Primary (ensemble)</td>
<td>25.2</td>
<td>25.2</td>
<td>24.8</td>
<td>24.8</td>
<td>0.659</td>
</tr>
<tr>
<td>uedin-syntax (Details)</td>
<td>rsennrich University of Edinburgh</td>
<td>Moses string-to-tree with ParZu for syntactic annotation.</td>
<td>yes</td>
<td></td>
<td>24.5</td>
<td>24.5</td>
<td>24.0</td>
<td>24.0</td>
<td>0.667</td>
</tr>
<tr>
<td>LIMSI/KIT Joint Submission (Details)</td>
<td>jan.niehues LIMSI/KIT</td>
<td></td>
<td>yes</td>
<td></td>
<td>24.1</td>
<td>24.1</td>
<td>23.6</td>
<td>23.6</td>
<td>0.664</td>
</tr>
<tr>
<td>uedin-pbt-wmt15-en-de (Details)</td>
<td>Matthias Huck University of Edinburgh</td>
<td>Phrase-based Moses</td>
<td>yes</td>
<td></td>
<td>23.3</td>
<td>23.3</td>
<td>22.8</td>
<td>22.8</td>
<td>0.681</td>
</tr>
</tbody>
</table>
## Encoder/Decoder model

### German-English WMT2015

<table>
<thead>
<tr>
<th>System</th>
<th>Submitter</th>
<th>System Notes</th>
<th>Constraint</th>
<th>Run Notes</th>
<th>BLEU</th>
<th>BLEU (11b)</th>
<th>BLEU-cased</th>
<th>BLEU-cased (11b)</th>
<th>TER</th>
</tr>
</thead>
<tbody>
<tr>
<td>uedin-pbt-wmt15-de-en</td>
<td>Matthias Huck, University of</td>
<td>Phrase-based Moses</td>
<td>yes</td>
<td>with LDC Gigaword</td>
<td>30.3</td>
<td>30.3</td>
<td>29.3</td>
<td>29.3</td>
<td>0.582</td>
</tr>
<tr>
<td>KIT primary</td>
<td>eunah.cho, KIT</td>
<td>Phrase-based translation system using GigaWord</td>
<td>yes</td>
<td></td>
<td>30.4</td>
<td>30.4</td>
<td>29.1</td>
<td>29.1</td>
<td>0.583</td>
</tr>
<tr>
<td>RWTH primary</td>
<td>Jan-Thorsten Peter, RWTH Aachen University</td>
<td></td>
<td>yes</td>
<td></td>
<td>30.3</td>
<td>30.3</td>
<td>28.9</td>
<td>28.9</td>
<td>0.580</td>
</tr>
<tr>
<td>Moses Phrase-Based</td>
<td>jhu-smt, Johns Hopkins University</td>
<td>Basic phrase-based system (pre-reordering)</td>
<td>yes</td>
<td></td>
<td>29.8</td>
<td>29.8</td>
<td>28.7</td>
<td>28.7</td>
<td>0.588</td>
</tr>
<tr>
<td>uedin-syntax-baseline</td>
<td>uedin-maria, University of Edinburgh</td>
<td>GHKM tree-to-string system. WMT14 primary. Gigaword</td>
<td>yes</td>
<td>[1500-9]</td>
<td>29.9</td>
<td>29.9</td>
<td>28.7</td>
<td>28.7</td>
<td>0.590</td>
</tr>
<tr>
<td>uedin-syntax-contrastive</td>
<td>uedin-maria, University of Edinburgh</td>
<td>GHKM string-to-tree. 2 bilingual neural LMs. 1 class based LM. Gigaword</td>
<td>yes</td>
<td>[1500-12]</td>
<td>29.8</td>
<td>29.8</td>
<td>28.7</td>
<td>28.7</td>
<td>0.586</td>
</tr>
<tr>
<td>Neural MT</td>
<td>UMontreal-MILA, University of Montreal</td>
<td>Neural machine translation + 5-gram LM (re-ranking)</td>
<td>yes</td>
<td></td>
<td>28.9</td>
<td>28.9</td>
<td>27.9</td>
<td>27.9</td>
<td>0.591</td>
</tr>
<tr>
<td>Neural MT (primary)</td>
<td>UMontreal-MILA, University of Montreal</td>
<td>Neural machine translation (primary) without monolingual corpus</td>
<td>yes</td>
<td></td>
<td>28.5</td>
<td>28.5</td>
<td>27.6</td>
<td>27.6</td>
<td>0.590</td>
</tr>
</tbody>
</table>
Questions?