Discriminative End-to-End Translation

Trevor Cohn

University of Sheffield

November, 2011
We’d like to model Machine Translation as an instance of structured prediction:

- We observe the input and output, but not the process in between.
- Probabilistic inference provides a natural method for modelling this latent structure.
Challenges for Discriminative SMT

- We need something to discriminate:
  - There is no gold standard.
  - There are often millions of ways of reaching the reference from the source.

- Parallel corpora are very noisy and contain non-literal translations.

- We must be able to train on tens of thousands of sentences (at least!) in order to estimate interesting

Compared to MERT

- Different objective: likelihood vs BLEU score of best translation.
- Allows for more features.
- Needs to be trained on larger datasets.
Derivational Ambiguity

Which derivation should we optimise?

```
the red hat
|
le chapeau
```
Derivational Ambiguity

Which derivation should we optimise?

the red hat
le chapeau

the red hat
le chapeau
Derivational Ambiguity

Which derivation should we optimise?

- the red hat
  - le chapeau

- the red hat
  - le chapeau

- the red hat
  - le chapeau
Derivational Ambiguity

Which derivation should we optimise?

\[
\begin{align*}
\text{the} & \quad \text{red} & \quad \text{hat} \\
\text{le} & \quad \text{chapeau}
\end{align*}
\]

\[
\begin{align*}
\text{the} & \quad \text{red} & \quad \text{hat} \\
\text{le} & \quad \text{chapeau}
\end{align*}
\]

\[
\begin{align*}
\text{the} & \quad \text{red} & \quad \text{hat} \\
\text{le} & \quad \text{chapeau}
\end{align*}
\]

\[
\begin{align*}
\text{the} & \quad \text{red} & \quad \text{hat} \\
\text{le} & \quad \text{chapeau}
\end{align*}
\]

Trevor Cohn (Sheffield)
Aims

- Globally train a translation model with binary features.
- Scale to more than hundreds of sentences.
- And use an underlying model structure which is known to perform well (Hiero).
- Incorporate a language model (LM) (Hard! requires approximation).
Synchronous Context Free Grammar

SCFG

\[ S \rightarrow \langle X_1, X_1 \rangle \]
\[ X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \]
\[ X \rightarrow \langle \text{Sie, She} \rangle \]
\[ X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \]
\[ X \rightarrow \langle \text{trinken, drink} \rangle \]
\[ X \rightarrow \langle \text{will, wants to} \rangle \]

Example Derivation

\[ S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \]
\[ \Rightarrow \langle \text{Sie X}_3, \text{She X}_3 \rangle \Rightarrow \langle \text{Sie X}_4 X_5, \text{She X}_4 X_5 \rangle \]
\[ \Rightarrow \langle \text{Sie will X}_5, \text{She wants to X}_5 \rangle \Rightarrow \langle \text{Sie will X}_6 X_7, \text{She wants to X}_7 X_6 \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee X}_7, \text{She wants to X}_7 \text{ a cup of coffee} \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle \]
Synchronous Context Free Grammar

**SCFG**

\[
S \rightarrow \langle X_1, X_1 \rangle \\
X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \\
X \rightarrow \langle \text{Sie, She} \rangle \\
X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \\
X \rightarrow \langle \text{trinken, drink} \rangle \\
X \rightarrow \langle \text{will, wants to} \rangle
\]

**Example Derivation**

\[
S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \\
\Rightarrow \langle \text{Sie } X_3, \text{ She } X_3 \rangle \Rightarrow \langle \text{Sie } X_4 X_5, \text{ She } X_4 X_5 \rangle \\
\Rightarrow \langle \text{Sie will } X_5, \text{ She wants to } X_5 \rangle \Rightarrow \langle \text{Sie will } X_6 X_7, \text{ She wants to } X_7 X_6 \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee } X_7, \text{ She wants to } X_7 \text{ a cup of coffee} \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle
\]
Synchronous Context Free Grammar

SCFG

\[ S \rightarrow \langle X_1, X_1 \rangle \]
\[ X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \]
\[ X \rightarrow \langle \text{Sie, She} \rangle \]
\[ X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \]

Example Derivation

\[ S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \]
\[ \Rightarrow \langle \text{Sie X}_3, \text{She X}_3 \rangle \Rightarrow \langle \text{Sie X}_4 X_5, \text{She X}_4 X_5 \rangle \]
\[ \Rightarrow \langle \text{Sie will X}_5, \text{She wants to X}_5 \rangle \Rightarrow \langle \text{Sie will X}_6 X_7, \text{She wants to X}_7 X_6 \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee X}_7, \text{She wants to X}_7 \text{ a cup of coffee} \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle \]
Synchronous Context Free Grammar

SCFG

\[ \begin{align*}
S & \rightarrow \langle X_1, X_1 \rangle \\
X & \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \\
X & \rightarrow \langle \text{Sie, She} \rangle \\
X & \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle
\end{align*} \]

\[ \begin{align*}
X & \rightarrow \langle X_1 X_2, X_1 X_2 \rangle \\
X & \rightarrow \langle \text{will, wants to} \rangle \\
X & \rightarrow \langle \text{trinken, drink} \rangle
\end{align*} \]

Example Derivation

\[ \begin{align*}
S \Rightarrow \langle X_1, X_1 \rangle & \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \\
& \Rightarrow \langle \text{Sie X}_3, \text{She X}_3 \rangle \\
& \Rightarrow \langle \text{Sie will X}_5, \text{She wants to X}_5 \rangle \\
& \Rightarrow \langle \text{Sie will eine Tasse Kaffee X}_7, \text{She wants to drink a cup of coffee} \rangle
\end{align*} \]
Synchronous Context Free Grammar

SCFG

\[ S \rightarrow \langle X_{1}, X_{1} \rangle \]
\[ X \rightarrow \langle X_{1} X_{2}, X_{1} X_{2} \rangle \]
\[ X \rightarrow \langle Sie, She \rangle \]
\[ X \rightarrow \langle eine Tasse Kaffee, a cup of coffee \rangle \]
\[ X \rightarrow \langle will, wants to \rangle \]
\[ X \rightarrow \langle trinken, drink \rangle \]

Example Derivation

\[ S \Rightarrow \langle X_{1}, X_{1} \rangle \Rightarrow \langle X_{2} X_{3}, X_{2} X_{3} \rangle \]
\[ \Rightarrow \langle Sie X_{3}, She X_{3} \rangle \Rightarrow \langle Sie X_{4} X_{5}, She X_{4} X_{5} \rangle \]
\[ \Rightarrow \langle Sie will X_{5}, She wants to X_{5} \rangle \Rightarrow \langle Sie will X_{6} X_{7}, She wants to X_{7} X_{6} \rangle \]
\[ \Rightarrow \langle Sie will eine Tasse Kaffee X_{7}, She wants to trinken, a cup of coffee \rangle \]
\[ \Rightarrow \langle Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee \rangle \]
Synchronous Context Free Grammar

SCFG

\[ S \rightarrow \langle X_1, X_1 \rangle \]
\[ X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \]
\[ X \rightarrow \langle \text{Sie, She} \rangle \]
\[ X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \]
\[ X \rightarrow \langle \text{will, wants to} \rangle \]
\[ X \rightarrow \langle \text{trinken, drink} \rangle \]

Example Derivation

\[ S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \]
\[ \Rightarrow \langle \text{Sie X}_3, \text{She X}_3 \rangle \Rightarrow \langle \text{Sie X}_4 X_5, \text{She X}_4 X_5 \rangle \]
\[ \Rightarrow \langle \text{Sie will X}_5, \text{She wants to X}_5 \rangle \Rightarrow \langle \text{Sie will X}_6 X_7, \text{She wants to X}_7 X_6 \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee X}_7, \text{She wants to X}_7 \text{ a cup of coffee} \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle \]
Synchronous Context Free Grammar

### SCFG

\[
S \rightarrow \langle X_1, X_1 \rangle \\
X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \\
X \rightarrow \langle \text{Sie, She} \rangle \\
X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \\
X \rightarrow \langle \text{trinken, drink} \rangle \\
X \rightarrow \langle \text{will, wants to} \rangle \\
\]

### Example Derivation

\[
S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \\
\Rightarrow \langle \text{Sie } X_3, \text{ She } X_3 \rangle \Rightarrow \langle \text{Sie } X_4 X_5, \text{ She } X_4 X_5 \rangle \\
\Rightarrow \langle \text{Sie will } X_5, \text{ She wants to } X_5 \rangle \Rightarrow \langle \text{Sie will } X_6 X_7, \text{ She wants to } X_7 X_6 \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee } X_7, \text{ She wants to } X_7 \text{ a cup of coffee} \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle \\
\]
Synchronous Context Free Grammar

SCFG

\[ S \rightarrow \langle X_1, X_1 \rangle \]
\[ X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \]
\[ X \rightarrow \langle \text{Sie, She} \rangle \]
\[ X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \]
\[ X \rightarrow \langle \text{will, wants to} \rangle \]
\[ X \rightarrow \langle \text{trinken, drink} \rangle \]

Example Derivation

\[ S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \]
\[ \Rightarrow \langle \text{Sie X}_3, \text{She X}_3 \rangle \Rightarrow \langle \text{Sie X}_4 X_5, \text{She X}_4 X_5 \rangle \]
\[ \Rightarrow \langle \text{Sie will X}_5, \text{She wants to X}_5 \rangle \Rightarrow \langle \text{Sie will X}_6 X_7, \text{She wants to X}_7 X_6 \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee X}_7, \text{She wants to X}_7 a \text{ cup of coffee} \rangle \]
\[ \Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle \]
Synchronous Context Free Grammar

**SCFG**

\[
S \rightarrow \langle X_1, X_1 \rangle \\
X \rightarrow \langle X_1 X_2, X_2 X_1 \rangle \\
X \rightarrow \langle \text{Sie, She} \rangle \\
X \rightarrow \langle \text{eine Tasse Kaffee, a cup of coffee} \rangle \\
X \rightarrow \langle \text{will, wants to} \rangle \\
X \rightarrow \langle \text{trinken, drink} \rangle
\]

**Example Derivation**

\[
S \Rightarrow \langle X_1, X_1 \rangle \Rightarrow \langle X_2 X_3, X_2 X_3 \rangle \\
\Rightarrow \langle \text{Sie } X_3, \text{ She } X_3 \rangle \Rightarrow \langle \text{Sie } X_4 X_5, \text{ She } X_4 X_5 \rangle \\
\Rightarrow \langle \text{Sie will } X_5, \text{ She wants to } X_5 \rangle \Rightarrow \langle \text{Sie will } X_6 X_7, \text{ She wants to } X_7 X_6 \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee } X_7, \text{ She wants to } X_7 \text{ a cup of coffee} \rangle \\
\Rightarrow \langle \text{Sie will eine Tasse Kaffee trinken, She wants to drink a cup of coffee} \rangle
\]
Discriminative Model: Parametric form

**Conditional probability of a derivation**

\[
p_\Lambda(d, e|f) = \frac{\exp \sum_k \lambda_k H_k(d, e, f)}{Z_\Lambda(f)}.
\]

**Conditional probability of a translation**

\[
p_\Lambda(e|f) = \sum_{d \in \Delta(e,f)} p_\Lambda(d, e|f).
\]

- We model the distribution over all derivations with a log-linear model.
- We marginalise out the derivations:
  - Important for estimating sparse derivation dependent features.
Discriminative Model: Parametric form

### Conditional probability of a derivation

\[
p_\Lambda(d, e|f) = \frac{\exp \sum_k \lambda_k H_k(d, e, f)}{Z_\Lambda(f)}.
\]

### Conditional probability of a translation

\[
p_\Lambda(e|f) = \sum_{d \in \Delta(e, f)} p_\Lambda(d, e|f).
\]

- We model the distribution over all derivations with a log-linear model.
- We marginalise out the derivations:
  - Important for estimating sparse derivation dependent features.
Features

The features must decompose with the rules:

\[ H_k(d, e, f) = \sum_{r \in d} h_k(f, r, q(r, d)) \]

- Any part of the source may be used for features:
  - Source syntax,
  - morphology,
  - lexical context etc.
Features: Non-local Context for the LM

Example Derivation

- $q(r, d)$ returns the order $m$ target lexical context of the rule in the derivation.
- The target side of the rule and its lexical context can then be used in features.
Approximating $Z_{\Lambda}(f)$

We must approximate the space of all possible derivations.

- We can exactly train the model without the LM context features ($p^{-LM}_{\Lambda}$).
- Including the LM context features ($p^{-LM}_{\Lambda}$) makes computing $Z_{\Lambda}(f)$ intractable ($|f|^{11}$ for a trigram LM).
Approximating $Z_\Lambda(f)$

\[ Z_\Lambda(f) \approx \tilde{Z}_\Lambda(f) = \sum_e \sum_{d \in \{ \subset \Delta(e, f) \}} \exp \sum_k \lambda_k H_k(d, e, f) \]

We must approximate the space of all possible derivations.

- Obvious approach: Use a beam search to produce a pruned chart of possible derivations.
Approximating $Z_\Lambda(f)$

$$Z_\Lambda(f) \approx \tilde{Z}_\Lambda(f) = \sum_e \sum_{d \in \Delta(e, f)} \exp \sum_k \lambda_k H_k(d, e, f)$$

We must approximate the space of all possible derivations.

- Alternative: Build a chart from sampled derivations:
  - Draw samples from an exact model trained without the LM ($p_{\Lambda}^{-LM}$),
  - create a chart from the union of these derivations with the LM context included,
  - this creates a less biased sample of derivations.
Approximating $Z_A(f)$: Using sampling
Approximating $Z_{\Lambda}(f)$: Using sampling
Approximating $Z_\Lambda(f)$: Using sampling

- $X_{[1,2]}$ Everything
- $X_{[1,3]}$ Everything and
- $X_{[1,4]}$ Everything *
- $X_{[3,4]}$ everyone
- $X_{[2,3]}$ anything
- $X_{[4,5]}$ is
- $X_{[5,6]}$ possible
- $S_{[1,6]}$ Everything *
- $S_{[1,6]}$ conceivable

Trevor Cohn (Sheffield) Discriminative SMT
Regularisation

Prior

\[ p_0^{+LM}(\lambda_k) \propto e^{-\frac{|\lambda_k - \lambda_{LM}|^2}{2\sigma^2}} \]

- The mean parameters of the Gaussian prior are set to \( \Lambda^{-LM} \).
- Any features that fall outside the approximated model will simply retain the weight assigned by \( \Lambda^{-LM} \).
- The prior will penalise substantial deviations away from \( \Lambda^{-LM} \).

Intuition: rule parameters should not change substantially with the inclusion of language model features.
Training

Objective function

\[ \mathcal{L} = \sum_{(e_i, f_i) \in D} \log p_{\Lambda}^{+LM}(e_i | f_i) + \sum_k \log p_0^{+LM}(\lambda_k) \]

- First train an exact model \( p_{\Lambda}^{-LM} \),
- \( p_{\Lambda}^{-LM} \) is then used for sampling and in the prior.
- Dynamic programming is used to efficiently calculate these feature expectations.
Experimental Details

All experiments are on data from the IWSLT Chinese-English evaluation.

- **Training:**
  - 40k training sentences pairs drawn from a travel domain.
  - 38k have a reachable reference.

- **Development and Testing**
  - 2005 test set used for evaluation.
  - 2004 test set used for development.
  - Each has 16 references: one translation and 15 paraphrases.

- Features on each rule, word penalty and LM (2.9M features).

- Evaluation with *Bleu*. 
Evaluation: Beam approximation

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_\Lambda^{-LM}(e</td>
<td>f)$</td>
</tr>
<tr>
<td>$p_\Lambda^{+LM}(e</td>
<td>f) (\tilde{Z}_{\Lambda}^{beam}(f))$</td>
</tr>
<tr>
<td>$p_\Lambda^{+LM}(e</td>
<td>f) (\tilde{Z}_{\Lambda}^{sam}(f))$</td>
</tr>
</tbody>
</table>

- Both techniques for incorporating the LM are effective.
- Sampling has a slight advantage over the use of a beam.
- Sampling technique results in 50% smaller charts.
Evaluation: Beam approximation

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{\Lambda}^{-LM}(e</td>
<td>f)$</td>
</tr>
<tr>
<td>$p_{\Lambda}^{+LM}(e</td>
<td>f) (\tilde{Z}_{\Lambda}^{beam}(f))$</td>
</tr>
<tr>
<td>$p_{\Lambda}^{+LM}(e</td>
<td>f) (\tilde{Z}_{\Lambda}^{sam}(f))$</td>
</tr>
</tbody>
</table>

- Both techniques for incorporating the LM are effective.
- Sampling has a slight advantage over the use of a beam.
- Sampling technique results in 50% smaller charts.
Evaluation: Beam approximation

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p^+_{\Lambda}^{LM}(e</td>
<td>f)$</td>
</tr>
<tr>
<td>$p^+_{\Lambda}^{LM}(e</td>
<td>f) (Z_{\Lambda}^{beam}(f))$</td>
</tr>
<tr>
<td>$p^+_{\Lambda}^{LM}(e</td>
<td>f) (Z_{\Lambda}^{sam}(f))$</td>
</tr>
</tbody>
</table>

- Both techniques for incorporating the LM are effective.
- Sampling has a slight advantage over the use of a beam.
- Sampling technique results in 50% smaller charts.
Comparison with MERT

<table>
<thead>
<tr>
<th>Model</th>
<th>$\text{BLEU}^{\text{multi-ref}}$</th>
<th>$\text{BLEU}^{\text{FirstRef}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_-^{LM}(e</td>
<td>f)$</td>
<td>35.2</td>
</tr>
<tr>
<td>$p_+^{LM}(e</td>
<td>f)$</td>
<td>44.6</td>
</tr>
<tr>
<td>MERT ($\text{BLEU}^{\text{NIST}}$)</td>
<td>44.5</td>
<td>30.2</td>
</tr>
</tbody>
</table>

- Minimum Error Rate Training (MERT) is the standard discriminative training method for SMT.
- Our model optimises likelihood, not $\text{BLEU}$.
- The brevity penalty makes direct comparison difficult.
- Our training regime achieves comparable performance to MERT.
Source side syntactic features

We exploit our models ability to incorporate sparse feature over the source sentence:

- We extract source syntax fragments for frequent (>1) rule applications.
- Resulting in 4.2M syntactic features extracted (for a total of 7.1M).
- This avoids the issues associated with directly incorporating syntax into the grammar.
Source side syntactic features

Example

Example Derivation:
(Step 1) $X_1 \rightarrow <$货币 兑换处, currency exchange office$>
(Step 2) $X_2 \rightarrow <$[X$_1$在 哪里 ？, Where is the [X$_1$]?$>

Where is the currency exchange office ？
Source side syntactic features

Example

Example Derivation:

(Step 1) $X_1 \rightarrow <\text{货币 兑换处, currency exchange office}>$

(Step 2) $X_2 \rightarrow <[X_1] \text{ 在 哪里 ？, Where is the [X_1] ？}>

Example Syntax feature =

for Step 1

Example Derivation:

货币 兑换处

在 哪里 ？

where is the currency exchange office

货币 兑换处

currency exchange office

Trevor Cohn (Sheffield)  Discriminative SMT  November, 2011
Source side syntactic features

Example

Example Derivation:
(Step 1) $X_1 \rightarrow <\text{货币 兑换处, currency exchange office}>$
(Step 2) $X_2 \rightarrow <[X_1] \text{ ?}, Where is the } [X_1] \text{ ?}>

Example Syntax feature =
for Step 2

Example Derivation:

Where is the currency exchange office?
Source side syntactic features

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{\Lambda}^{+LM}(e</td>
<td>f)$</td>
</tr>
<tr>
<td>$p_{\Lambda}^{+LM}(e</td>
<td>f)$ + syntax</td>
</tr>
</tbody>
</table>

- The model is able to gain advantage from sparse syntactic features, without being overwhelmed by the noise.
Summary

In this section we have:

- Shown that conditional latent variable graphical models can be scaled to a complex task such as translation.
- Demonstrated the flexibility of this modelling framework by estimating millions of noisy source conditioned syntactic features.
- Achieved results comparable to current state-of-the-art models.
Future Directions

Questions:

- How do we approximate $Z$ such that we can bound the error?
- How do we train undirected graphical models with unsupervised objective on large scale tasks?
- What objective function gives the best performance/scalability tradeoffs?

Acknowledgements

Thanks to Phil Blunsom and Miles Osborne.