Contents

• What is a syntactic model?

• What’s wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
What is a syntactic model?

• Hierarchical Phrase-Based Model
  – String-to-string
  – Non-terminals are unlabelled
    \[ X \rightarrow \text{habe } X_1 \text{ gegessen} \ # \text{have eaten } X_1 \]

• Tree-to-string Model
  – Source non-terminals are labelled
    • match input parse tree
      \[ S \rightarrow \text{habe } \textbf{NP}_1 \text{ gegessen} \ # \text{have eaten } \textbf{NP}_1 \]
What is a syntactic model?

• Hierarchical Phrase-Based Model
  – String-to-string
  – Non-terminals are unlabelled

\[
X \rightarrow \text{habe } X_1 \text{ gegessen } \# \text{ have eaten } X_1
\]

• Tree-to-string Model
  – Source non-terminals are labelled
    • match input parse tree

\[
S \rightarrow \text{habe } \textbf{NP}_1 \text{ gegessen } \# \text{ have eaten } \textbf{NP}_1
\]
What is a syntactic model?

• Hierarchical Phrase-Based Model
  – String-to-string
  – Non-terminals are unlabelled
    
    \[
    X \rightarrow \text{habe } X_1 \text{ gegessen} \quad \# \text{have eaten } X_1
    \]

• Tree-to-string Model
  – Source non-terminals are labelled
    • match input parse tree
      
      \[
      S \rightarrow \text{habe } \textbf{NP}_1 \text{ gegessen} \quad \# \text{have eaten } \textbf{NP}_1
      \]
What is a syntactic model?

• Hierarchical Phrase-Based Model
  – String-to-string
  – Non-terminals are unlabelled
    \[ X \rightarrow \text{habe } X_1 \text{ gegessen} \ # \text{ have eaten } X_1 \]

• Tree-to-string Model
  – Source non-terminals are labelled
    • match input parse tree
      \[ S \rightarrow \text{habe } NP_1 \text{ gegessen} \ # \text{ have eaten } NP_1 \]
What is a syntactic model?

• Hierarchical Phrase-Based Model
  – String-to-string
  – Non-terminals are unlabelled
    \[ X \rightarrow \text{habe } X_1 \text{ gegessen } \# \text{ have eaten } X_1 \]

• Tree-to-string Model
  – Source non-terminals are labelled
    • match input parse tree
      \[ S \rightarrow \text{habe } \textbf{NP}_1 \text{ gegessen } \# \text{ have eaten } \textbf{NP}_1 \]
Contents

• What is a syntactic model?

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
What’s Wrong with Syntax?

<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree-to-string</td>
<td>27.02</td>
<td>57.68</td>
</tr>
<tr>
<td>Tree-to-tree</td>
<td>22.23</td>
<td>54.05</td>
</tr>
<tr>
<td>Moses (phrase-based)</td>
<td>30.18</td>
<td>58.13</td>
</tr>
</tbody>
</table>

Evaluation of French-English MT System
(Ambati and Lavie, 2009)
Hierarchical Model

according to János Veres, this would be in the first quarter of 2008 possible.
Hierarchical Model

Hierarchical Model according to János Veres, this would be in the first quarter of 2008 possible.
Hierarchical Model

according to János Veres, this would be in the first quarter of 2008 possible.
János Veres wäre dies im ersten Quartal 2008 möglich.
Tree-to-String Model

according to János Veres would be this in the first quarter of 2008 possible.
Tree-to-String Model

according to János Veres would be this in the first quarter of 2008 possible.
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Why Use Syntactic Models?

• Decrease decoding time
  – Derivation constrained by source parse tree
Why Use Syntactic Models?

• Decrease decoding time
  – Derivation constrained by source parse tree

• Long-range reordering during decoding
  – rules covering more words than max-span limit
Why Use Syntactic Models?

• Decrease decoding time
  – Derivation constrained by source parse tree

• Long-range reordering during decoding
  – rules covering more words than max-span limit

• Other rule-forms
  – 3+ non-terminals
  – consecutive non-terminals
  – non-lexicalized rules
Why Use Syntactic Models?

• Decrease decoding time
  – Derivation constrained by source parse tree

• Long-range reordering during decoding
  – rules covering more words than max-span limit

• Other rule-forms
  – 3+ non-terminals
  – consecutive non-terminals
  – non-lexicalized rules

\[ X \rightarrow S_1 O_2 V_3 \quad \# \quad S_1 V_3 O_2 \]
\[ X \rightarrow PRO_1 PRO_2 \text{ aime bien} \quad \# \quad PRO_1 \text{ like PRO}_2 \]
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)
Other Syntactic Models

• Syntax-Augmented MT (SAMT)
  – Not constrained only to parse tree
  – (Zollmann and Venugopal, 2006)

• Binarization
  – Restructure and relabel parse parse tree
  – (Wang et al, 2010)

• Forest-based translation
  – Recover from parse errors
  – (Mi et al, 2008)

• Soft constraint
  – Reward/Penalize derivations which follows parse structure
  – (Chiang 2010)

• Ignore Syntax (occasionally)
Mixed-Syntax Model

• Tree-to-string model
  – input is a parse tree

• Roles of non-terminals
  – Constrain derivation to parse constituents
  – State information
    • Consistent node label on target derivation
    • hypotheses with different head NT cannot be recombined
Mixed-Syntax Model

• Tree-to-string model
  – input is a parse tree

• Roles of non-terminals
  – Constrain derivation to parse constituents
    • Can sometime have no constraints
  – State information
    • Consistent node label on target derivation
    • hypotheses with different head NT cannot be recombined
    • always X
Mixed-Syntax Model

Example Translation Rules

• Naïve syntax model

  \[ VP \rightarrow \text{VVFIN}_1 \text{ zu VVINF}_2 \# \text{ to VVFIN}_2 \text{ VVINF}_1 \]

• Mixed-Syntax Model

  \[ VP \rightarrow X_1 \text{ zu VVINF}_2 \# X \rightarrow \text{ to } X_2 X_1 \]
Mixed-Syntax Model
Example Translation Rules

• Naïve syntax model
  \[ VP \rightarrow VVFIN_1 \text{ zu } VVINF_2 \# \rightarrow VVFIN_2 VVINF_1 \]

• Mixed-Syntax Model
  \[ VP \rightarrow X_1 \text{ zu } VVINF_2 \# X \rightarrow \rightarrow X_2 X_1 \]
Mixed-Syntax Model
Example Translation Rules

• Naïve syntax model
  \[ VP \rightarrow \text{VVFIN}_1 \text{ zu VVINF}_2 \# \text{ to VVFIN}_2 \text{ VVINF}_1 \]

• Mixed-Syntax Model
  \[ VP \rightarrow X_1 \text{ zu VVINF}_2 \# X \rightarrow \text{ to } X_2 \ X_1 \]
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Extraction

• Allow rules
  – Max 3 non-terminals
  – Adjacent non-terminals
    • At least 1 NT must be syntactic
  – Non-lexicalized rules
Example Rules Extracted

| Rule                                      | Fractional Count | p(t | s) |
|-------------------------------------------|------------------|-------|
| **Syntactic Rules**                      |                  |       |
| VP $\rightarrow$ NP$_1$ VVINF$_2$ $\#$ X$\rightarrow$ X$_2$ X$_1$ | 167.3            | 68%   |
| **Mixed Rules**                           |                  |       |
| VP $\rightarrow$ X$_1$ VZ$_2$ $\#$ X$\rightarrow$ X$_2$ X$_1$ | 63.3             | 64%   |
| VP $\rightarrow$ X$_1$ zu VVINF$_2$ $\#$ X$\rightarrow$ to X$_2$ X$_1$ | 39.9             | 56%   |
| TOP $\rightarrow$ NP$_1$ X$_2$ $\#$ X$\rightarrow$ X$_1$ X$_2$   | 43.1             | 92%   |
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Synchronous CFG

Input:

```
NP  S
 |   |
PRO VP
 |   |
NE PRO VB NE
 |   |   |
je ne lui vu pas
```

The input can be translated as follows:

Derivation 1:
```
S NP 1 VP 2
X X X X
```

Derivation 2:
```
S NP PRO je VP 2
X X X
```
Synchronous CFG

Input:

```
S
├── NP
│   └── PRO
│       └── je
├── VP
│   ├── NE
│   │   └── ne
│   ├── PRO
│   │   └── lui
│   └── VB
│       └── vu
└── NE
    └── pas
```

Rules:

- $S \rightarrow NP_1 \ VP_2$  
  \hspace*{1cm} # NP_1 \ VP_2$
- $NP \rightarrow je$  
  \hspace*{1cm} # I$
- $PRO \rightarrow lui$  
  \hspace*{1cm} # him$
- $VB \rightarrow vu$  
  \hspace*{1cm} # see$
- $VP \rightarrow ne \ PRO_1 \ VB_2 \ pas$  
  \hspace*{1cm} # did not \ VB_2 \ PRO_1$
Synchronous CFG

Input:

```
S
  NP  VP
  PRO NE PRO VB NE
  je  ne lui vu pas
```

Derivation:

```
S
  NP1 VP2
  PRO
  je
  NP
  VP
  PRO
  lui
  VB
  vu
  NE
  pas
```

Rules:

```
S \rightarrow NP_1 \text{ VP}_2 \quad \# \text{ NP}_1 \text{ VP}_2
NP \rightarrow \text{ je} \quad \# \text{ I}
PRO \rightarrow \text{ lui} \quad \# \text{ him}
VB \rightarrow \text{ vu} \quad \# \text{ see}
VP \rightarrow \text{ ne PRO}_1 \text{ VB}_2 \text{ pas} \quad \# \text{ did not VB}_2 \text{ PRO}_1
```
4.2.10 Example Decoding with the Mixed-Syntax Model

Supposing a parsed input sentence and the mixed-syntax grammar, below:

Input:

\[
\text{S} \quad \text{NP} \quad \text{VP} \quad \text{NE} \\
\text{PRO} \quad \text{PRO} \quad \text{VB} \quad \text{NE} \\
\text{je} \quad \text{ne} \quad \text{lui} \quad \text{vu} \quad \text{pas} \\
\]

Grammar:

\[
\text{S} \to \text{NP}_1 \text{VP}_2 \\
\text{NP} \to \text{je} \\
\text{PRO} \to \text{lui} \quad \text{# him} \\
\text{VB} \to \text{vu} \quad \text{# see} \\
\text{VP} \to \text{ne PRO}_1 \text{VB}_2 \text{pas} \quad \text{# did not VB}_2 \text{PRO}_1
\]

The input can be translated as follows:

Derivation 1:

\[
\text{S} \quad \text{NP}_1 \quad \text{VP}_2 \\
\text{X} \quad \text{X}_1 \quad \text{X}_2 \\
\]

Derivation 2:

\[
\text{S} \quad \text{NP} \quad \text{PRO} \quad \text{je} \\
\text{X} \quad \text{X}_1 \quad \text{X}_2 \\
\]

Rules:

\[
\text{S} \to \text{NP}_1 \text{VP}_2 \quad \text{# NP}_1 \text{VP}_2 \\
\text{NP} \to \text{je} \quad \text{# I} \\
\text{PRO} \to \text{lui} \quad \text{# him} \\
\text{VB} \to \text{vu} \quad \text{# see} \\
\text{VP} \to \text{ne PRO}_1 \text{VB}_2 \text{pas} \quad \text{# did not VB}_2 \text{PRO}_1
\]
Synchronous CFG

Input:

```
S
  NP
  PRO j
  NP
  VP
  PRO ne
  NE leur
  VP
  VB vu
  NP
  PRO pas
```

Grammar:

```
S → NP₁ VP₂
NP → je
PRO → lui
VB → vu
VP → ne PRO₁ VB₂ pas
```

Derivation:

```
S
  NP
  VP
  NP
  VP
  NP
  VP
```

Rules:

- `S → NP₁ VP₂` # NP₁ VP₂
- `NP → je` # I
- `PRO → lui` # him
- `VB → vu` # see
- `VP → ne PRO₁ VB₂ pas` # did not VB₂ PRO₁
4.2. Model

4.2.10 Example Decoding with the Mixed-Syntax Model

Supposing a parsed input sentence and the mixed-syntax grammar, below:

```
S  NP 1  VP 2
NP  PRO  NE  PRO  VB  NE
    je  ne  lui  vu  pas

S  NP 1  VP 2
NP  PRO  ne  VP 2  pas
    lui

S  NP  VP
NP  PRO  I  VP 2
    did  not  VB 2  PRO
    him
```

The input can be translated as follows:

Derivation 1:

Derivation 2:

Rules:

- $S \rightarrow NP_1 \ VP_2$  # $NP_1 \ VP_2$
- $NP \rightarrow je$  # I
- $PRO \rightarrow lui$  # him
- $VB \rightarrow vu$  # see
- $VP \rightarrow ne \ PRO_1 \ VB_2 \ pas$  # did not $VB_2 \ PRO_1$
Input:

```
S
  NP  VP
  |    |
  PRO NE  PRO VB NE
  |    |    |    |
  je  ne lui vu pas
```

Grammar:

```
S → NP1 VP2, X → X1 X2

PRO → je, X → I

PRO → lui, X → him

VP → ne PRO1 VB2 pas # did not VB2 PRO1
```

Derivation:

```
S
  NP VP
  |    |
  PRO NE VP
  |    |    |
  je ne PRO VB pas
```

Rules:

- \( S \rightarrow NP_1 VP_2 \) # NP_1 VP_2
- \( NP \rightarrow je \) # I
- \( PRO \rightarrow lui \) # him
- \( VB \rightarrow vu \) # see
- \( VP \rightarrow ne PRO_1 VB_2 pas \) # did not VB_2 PRO_1

Synchronous CFG
Synchronous CFG

Input:

Derivation:

Rules:

S → NP₁ VP₂  # NP₁ VP₂
NP → je  # I
PRO → lui  # him
VB → vu  # see
VP → ne PRO₁ VB₂ pas  # did not VB₂ PRO₁
Mixed-Syntax Model

Input:

```
S
 /\  
NP VP
 /\  
PRO NE PRO VB NE
 /\  
je ne lui vu pas
```

The input can be translated as follows:

Derivation 1:
```
S NP VP, X X 1 X 2
```

Derivation 2:
```
S NP PRO je VP 2, X X I X 2
```

Grammar:
```
S → NP 1 VP 2, X → X 1 X 2

PRO → je, X → I

PRO → lui, X → him

VB → vu, X → see

VP → ne X 1 pas, X → did not X 1 X → PRO 1 VB 2, X → X 2 X 1
```
Mixed-Syntax Model

Input:

```
S
  NP
    PRO je
    VP
      NE ne
      PRO lui
      VB vu
      NE pas
```

Rules:

- \( S \rightarrow NP_1 VP_2 \)  \# \( X \rightarrow NP_1 VP_2 \)
- \( PRO \rightarrow je \)  \# \( X \rightarrow I \)
- \( PRO \rightarrow lui \)  \# \( X \rightarrow \text{him} \)
- \( VB \rightarrow vu \)  \# \( X \rightarrow \text{see} \)
- \( VP \rightarrow ne X_1 pas \)  \# \( \text{did not} X_1 \)
- \( X \rightarrow PRO_1 VB_2 \)  \# \( X \rightarrow X_2 X_1 \)
### Mixed-Syntax Model

#### Input:

```
S
 NP  VP
  PRO NE  PRO VB NE
   je  ne  lui  vu  pas
```

#### Derivation:

```
S, X
 NP1  VP2  X1  X2
```

#### Rules:

- `S → NP1 VP2`  # `X → NP1 VP2`
- `PRO → je`  # `X → I`
- `PRO → lui`  # `X → him`
- `VB → vu`  # `X → see`
- `VP → ne X1 pas`  # `did not X1`
- `X → PRO1 VB2`  # `X → X2 X1`
Mixed-Syntax Model

Input:

```
S
NP  VP
PRO NE PRO VB NE
  je  ne lui vu pas
```

Grammar:

```
S → NP₁ VP₂, X → NP₁ VP₂
PRO → je, X → I
PRO → lui, X → him
VB → vu, X → see
VP → ne X₁ pas, # did not X₁
X → PRO₁ VB₂, # X → X₂ X₁
```

Derivation:

```
S
NP  VP
PRO VP₂
  je
```

```
S
NP  VP
PRO VP₂
  je I
```

```
S
NP  VP
PRO VP₂
  Je
VP
  ne
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```

```
S
NP  VP
PRO VP₂
  Je
VP
  Ne did not
```
Mixed-Syntax Model

Input:

```
S
 /\  
NP  VP
 |  |
PRO NE PRO VB NE
  |  |
  je ne lui vu pas
```

Grammar:

```
S → NP_1 VP_2
PRO → je
PRO → lui
VB → vu
VP → ne X_1 pas
X → PRO_1 VB_2
```

The input can be translated as follows:

Derivation 1:
```
S NP_1 VP_2 , X
```

Derivation 2:
```
S NP_1 PRO_1 VP_2 , X
```

Derivation 3:
```
S NP_1 PRO_1 VP_2 , X
```

Rules:

```
S → NP_1 VP_2 # X → NP_1 VP_2
PRO → je # X → I
PRO → lui # X → him
VB → vu # X → see
VP → ne X_1 pas # did not X_1
X → PRO_1 VB_2 # X → X_2 X_1
```
4.2. Model

4.2.10 Example Decoding with the Mixed-Syntax Model

Supposing a parsed input sentence and the mixed-syntax grammar, below:

Input:

\[
S \\
NP \\
PRO \quad \text{je} \\
NP \\
PRO \quad \text{ne} \\
VP \\
\text{NE} \quad \text{lui} \\
\text{NE} \quad \text{vu} \\
\text{NE} \quad \text{pas} \\
\]

Grammar:

\[
S \rightarrow NP_1 \ VP_2 \\
PRO \rightarrow \text{je} \\
PRO \rightarrow \text{lui} \\
VB \rightarrow \text{vu} \\
VP \rightarrow \text{ne} \ X_1 \ pas \\
X \rightarrow \text{PRO}_1 \ VB_2 \\
\]

Derivation 1:

\[
S \rightarrow NP_1 \ VP_2 \\
\]

Derivation 2:

\[
S \rightarrow NP_1 \ VP_2 \\
PRO \rightarrow \text{je} \\
\]

Derivation 3:

\[
S \rightarrow NP_1 \ VP_2 \\
\]

Derivation 4:

\[
S \rightarrow NP_1 \ VP_2 \\
\]

Derivation 5:

\[
S \rightarrow NP_1 \ VP_2 \\
\]

Derivation 6:

\[
S \rightarrow NP_1 \ VP_2 \\
\]

Notice in derivation 2 that the non-terminal label from the parent node: NP: does not match the label of the child's root node: PRO. - However: since both labels cover the source word: je: this is a valid inference.

In derivation 3: the free non-terminal does not span a source syntactic constituent. This is valid but permitted only for undecorated non-terminals: denoted by the symbol X.

Notice also that the completed source derivation 6 is not isomorphic to the original source parse. The mixed-syntax model allows derivations which are suited for the primary objective of translation: rather than being constrained to replicate the source parse.
4.2. Model

4.2.10 Example Decoding with the Mixed-Syntax Model

Supposing a parsed input sentence and the mixed-syntax grammar, below:

Input:

\[ S \rightarrow NP_1 VP_2, X \rightarrow NP_1 VP_2 \]

\[ PRO \rightarrow je, \quad X \rightarrow I \]

\[ PRO \rightarrow lui, \quad X \rightarrow him \]

\[ VB \rightarrow vu, \quad X \rightarrow see \]

\[ VP \rightarrow ne X_1 pas, \quad # \text{ did not } X_1 \]

\[ X \rightarrow PRO_1 VB_2, \quad # X \rightarrow X_2 X_1 \]

The input can be translated as follows:

Derivation 1:

\[ S \rightarrow NP_1 VP_2, X \rightarrow NP_1 VP_2 \]

Derivation 2:

\[ S \rightarrow NP_1 VP_2, X \rightarrow I \]

Derivation 3:

\[ S \rightarrow NP_1 VP_2, X \rightarrow him \]

Notice in derivation 2 that the non-terminal label from the parent node: NP: does not match the label of the child's root node: PRO - However: since both labels cover the source word: je: this is a valid inference-

In derivation 3: the free non-terminal does not span a source syntactic constituent - This is valid but permitted only for undecorated non-terminals: denoted by the symbol X -

Notice also that the completed source derivation 6 is not isomorphic to the original source parse - The mixed-syntax model allows derivations which is suited for the primary objective of translation: rather than being constrained to replicate the source parse.

4.2.11 Tree-to-String Model

Huang et al. (52)36a described a tree-to-string model using a tree-transducer which
Mixed-Syntax Model

Input:

```
S
| NP
| PRO
| je
|  
| VP
| NE
| ne
|  
| PRO
| lui
|  
| VB
| vu
|  
| NE
| pas
```

Derivation:

```
S
| NP        |
| PRO X   |
| je X     |
| ne X     |
| lui X    |
| VP X     |
| ne X pas |
```

Rules:

- \( S \rightarrow NP_1 VP_2 \)  
- \( PRO \rightarrow je \)  
- \( PRO \rightarrow lui \)  
- \( VB \rightarrow vu \)  
- \( VP \rightarrow ne X_1 pas \)  
- \( X \rightarrow PRO_1 VB_2 \)  
- \( X \rightarrow PRO_1 VP_2 \)  
- \( X \rightarrow NP_1 VP_2 \)  
- \( X \rightarrow I \)  
- \( X \rightarrow him \)  
- \( X \rightarrow see \)  
- \( X \rightarrow did \)  
- \( X \rightarrow not \)  
- \( X \rightarrow see \)  
- \( X \rightarrow him \)
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
# Experiment

## German-English Corpus

<table>
<thead>
<tr>
<th></th>
<th>German</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train Sentences</td>
<td>82,306</td>
<td></td>
</tr>
<tr>
<td>Words</td>
<td>2,034,373</td>
<td>1,965,325</td>
</tr>
<tr>
<td>Tune Sentences</td>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>Test Sentences</td>
<td>1026</td>
<td></td>
</tr>
</tbody>
</table>

- **Trained:** News Commentary 2009
- **Tuned:** held out set
- **Tested:** *nc test 2007*
## Results

### Using constituent parse

#### German-English

<table>
<thead>
<tr>
<th>Model</th>
<th># rules</th>
<th>%BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>61.2m</td>
<td>15.9</td>
</tr>
<tr>
<td>Tree-to-String</td>
<td>4.7m</td>
<td>14.9</td>
</tr>
<tr>
<td>Mixed Syntax</td>
<td>128.7m</td>
<td>16.7</td>
</tr>
</tbody>
</table>

#### English-German

<table>
<thead>
<tr>
<th>Model</th>
<th># rules</th>
<th>%BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>84.6m</td>
<td>10.2</td>
</tr>
<tr>
<td>Mixed Syntax</td>
<td>175.0m</td>
<td>10.6</td>
</tr>
</tbody>
</table>
Example
Hierarchical Model

according to János Veres, this would be in the first quarter of 2008 possible.
Example

Mixed-Syntax Model

According to János Veres this would be possible in the first quarter of 2008.
Example

Mixed Syntax

according to János Veres this would be possible in the first quarter of 2008.
Example

Mixed Syntax

according to János Veres this would be possible in the first quarter of 2008.
Example
Mixed Syntax

according to János Veres this would be possible in the first quarter of 2008.
Example

Mixed Syntax

according to János Veres this would be possible in the first quarter of 2008.
Chunk Tags

• Advantages of Shallow Tags
  – Don’t need Treebank
  – More reliable

• Disadvantages
  – Not a tree structure
    • We don’t rely on tree structure
# Results

## Shallow Tags

### German-English

<table>
<thead>
<tr>
<th>Model</th>
<th># rules</th>
<th>%BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>64.3m</td>
<td>16.3</td>
</tr>
<tr>
<td>Mixed Syntax</td>
<td>254.5m</td>
<td>16.8</td>
</tr>
</tbody>
</table>
# Larger Training Corpus

## German-English Corpus

<table>
<thead>
<tr>
<th></th>
<th>German</th>
<th>English</th>
<th>Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Train</strong></td>
<td>Sentences</td>
<td>1,446,224</td>
<td>Europarl v5</td>
</tr>
<tr>
<td></td>
<td>Words</td>
<td>37,420,876</td>
<td>39,464,626</td>
</tr>
<tr>
<td><strong>Tune</strong></td>
<td>Sentences</td>
<td>1910</td>
<td>dev2006</td>
</tr>
<tr>
<td><strong>Test</strong></td>
<td>Sentences</td>
<td>1920</td>
<td>nc test2007 v2</td>
</tr>
<tr>
<td>(in-domain)</td>
<td></td>
<td>1042</td>
<td>devtest2006</td>
</tr>
<tr>
<td>(out-of-domain)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Larger Training Corpus

German-English

<table>
<thead>
<tr>
<th>Model</th>
<th># rules</th>
<th>In-domain (BLEU)</th>
<th>Out-of-domain (BLEU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>500m</td>
<td>22.1</td>
<td>16.5</td>
</tr>
<tr>
<td>Mixed Syntax (original)</td>
<td>2664m</td>
<td>21.6</td>
<td>16.3</td>
</tr>
<tr>
<td>Mixed Syntax (new extraction)</td>
<td>1435m</td>
<td>22.7</td>
<td>17.8</td>
</tr>
</tbody>
</table>
Contents

• What’s Wrong with Syntax?

• Which syntax model to use?

• Why use syntactic models?

• Mixed-Syntax Model
  – Extraction
  – Decoding
  – Results

• Future Work
Create your own label
Dumb labels

ich bitte Sie, sich zu einer Schweigeminute zu erheben.
Create your own label
Dumb labels

ich bitte Sie, sich zu einer Schweigeminute zu erheben.
Create your own labels

Dumb labels

ich bitte Sie, sich zu einer Schweigeminute zu erheben.

<table>
<thead>
<tr>
<th>Model</th>
<th>In-domain (BLEU)</th>
<th>Out-of-domain (BLEU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>22.1</td>
<td>16.5</td>
</tr>
<tr>
<td>Dumb Labels</td>
<td>22.0</td>
<td>16.3</td>
</tr>
</tbody>
</table>
Create your own labels

Labels motivated by reordering

Labelling patterns:

1. VMFIN...VVINF EOS
2. VVINF und ... VVINF
3. VAFIN ... (VVPP or VVINF) EOS
4. , PRELS ... VVINF EOS
5. EOS ... zu VVINF

Example:

ich bitte Sie , sich zu einer Schweigeminute zu erheben .

label 5

... werde ich dem Vorschlag von Herrn Evans folgen .

label 3
Create your own labels

Labels motivated by reordering

<table>
<thead>
<tr>
<th>Model</th>
<th>In-domain (BLEU)</th>
<th>Out-of-domain (BLEU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical</td>
<td>22.1</td>
<td>16.5</td>
</tr>
<tr>
<td>Dumb Labels</td>
<td>22.0</td>
<td>16.3</td>
</tr>
<tr>
<td>Reordering Labels</td>
<td>22.1</td>
<td>16.9</td>
</tr>
</tbody>
</table>
Conclusion

• Mixed-Syntax Model
  – SCFG-based decoding
  – Hierarchical phrase-based v. tree-to-string
  – Generality v. specificity

• Syntax Models
  – Many variations
  – Won’t automatically make MT better
  – Question
    • which syntactic information?
    • how do we use it?
    • why use it?