Neural network parsing

Daniël de Kok & Erhard Hinrichs

Recap

$c \leftarrow c_0(S)$
while $\neg isFinal(c)$ **do**
  $t \leftarrow g(c)$
  $c \leftarrow t(c)$
end while

- c: parser state
- c_0: initial parser state
- t: transition
- g: parsing guide

- Next question: what function do we use as g?
- Answer: a probabilistic model: $p(t|c)$
- Benefits:
  - Well-understood methods for optimization.
  - Provides a good basis for beam search.

Data-driven models

Introduction

- Goal of of a guide during parsing: predict for a given c the best transition t.
- Using properties (**features**) of the parser state.

Example (parsing features)

Consider following two parsing features:

- f_1: 1 if $\sigma = [\cdots|\text{ART}|\text{NN}]$, 0 otherwise
- f_2: 1 if $\sigma = [\cdots|\text{APPR}|\text{NE}]$, 0 otherwise

Example (parsing features)

Consider following two parsing features:

- f_1: 1 if $\sigma = [\cdots|\text{ART}|\text{NN}]$, 0 otherwise
- f_2: 1 if $\sigma = [\cdots|\text{APPR}|\text{NE}]$, 0 otherwise

For a parser state with $\sigma = [\ldots, \text{APPR}, \text{NE}]$:

- $f_1(c_i) = 0$
- $f_2(c_i) = 1$
- $\mathbf{x}_i = [0, 1]$

Example (parsing features)

Consider following two parsing features:

- f_1: 1 if $\sigma = [\cdots|\text{ART}|\text{NN}]$, 0 otherwise
- f_2: 1 if $\sigma = [\cdots|\text{APPR}|\text{NE}]$, 0 otherwise

For a parser state with $\sigma = [\ldots, \text{ART}, \text{NN}]$:

- $f_1(c_i) = 1$
- $f_2(c_i) = 0$
- $\mathbf{x}_i = [1, 0]$

Feature-based rules

- Given the features:
  - f_1: 1 if $\sigma = [\cdots|\text{ART}|\text{NN}]$, 0 otherwise
  - f_2: 1 if $\sigma = [\cdots|\text{APPR}|\text{NE}]$, 0 otherwise
- $\textsc{Left-Arc}_{\text{DET}}$ when $f_1 = 1$

Feature-based rules

- Given the features:
  - f_1: 1 if $\sigma = [\cdots|\text{ART}|\text{NN}]$, 0 otherwise
  - f_2: 1 if $\sigma = [\cdots|\text{APPR}|\text{NE}]$, 0 otherwise
- $\textsc{Left-Arc}_{\text{DET}}$ when $f_1 = 1$
- $\textsc{Right-Arc}_{\text{PN}}$ when $f_2 = 1$

Data-driven learning

For each sentence S_i and the corresponding gold standard dependency structure g_i:

1. Parse S_i using an oracle to reproduce G_i.
2. Extract the transition sequence $C_{0,m}$ and the corresponding transitions $T_{0,m}$.
3. For each pair $(c_i, t_i) \in (C_{0,m}, T_{0,m})$:
    1. Convert c_i to a feature vector $\mathbf{x}_i \in \mathbb{R}^d$.
    2. Convert t_i to a natural number $y_i \in \mathbb{N}$.
    3. $(\mathbf{x}_i, y_i)$ is a training instance.

Example

Transition	σ	β	A
	[ROOT]	[Staatsanwalt, ...]	$\emptyset$
SH	[ROOT, Staatsanwalt]	[muß, ⋯]	$\emptyset$
SH	[ROOT, Staatsanwalt, muß]	[AWO-Konten, ⋯]	$\emptyset$
$\mathrm{LA_{SUBJ}}$	[ROOT, muß]	[AWO-Konten, ⋯]	$A_1 = \{(\text{muß}, \text{SUBJ}, \text{Staatsanwalt})\}$
SH	[ROOT, AWO-Konten]	[prüfen]	A_1
SH	[ROOT, AWO-Konten, prüfen]	[]	A_1
$\mathrm{LA_{OBJA}}$	[ROOT, muß, prüfen]	[]	$A_2 = A_1 \cup \{(\text{prüfen}, \mathit{OBJA}, \text{AWO-Konten})\}$
$\mathrm{RA_{AUX}}$	[ROOT, muß]	[]	$A_3 = A_2 \cup \{(\text{muß}, \mathit{AUX}, \text{prüfen})\}$
$\mathrm{RA_{ROOT}}$	[ROOT]	[]	$A_4 = A_3 \cup \{(\text{ROOT}, \mathit{ROOT}, \text{muß})\}$

Example

Transition	σ	β	A
	[ROOT]	[Staatsanwalt, ...]	$\emptyset$
SH	[ROOT, Staatsanwalt]	[muß, $\cdots$]	$\emptyset$
SH	[ROOT, Staatsanwalt, muß]	[AWO-Konten, $\cdots$]	$\emptyset$
$\mathrm{LA}_{\mathrm{SUBJ}}$	[ROOT, muß]	[AWO-Konten, $\cdots$]	$A_1 = \{(\text{muß, SUBJ, Staatsanwalt})\}$
SH	[ROOT, AWO-Konten]	[prüfen]	A_1
SH	[ROOT, AWO-Konten, prüfen]	[]	A_1
$\mathrm{LA}_{\mathrm{OBJA}}$	[ROOT, muß, prüfen]	[]	$A_2 = A_1 \cup \{(\text{prüfen}, \mathit{OBJA}, \text{AWO-Konten})\}$
$\mathrm{RA}_{\mathrm{AUX}}$	[ROOT, muß]	[]	$A_3 = A_2 \cup \{(\text{muß}, \mathit{AUX}, \text{prüfen})\}$
$\mathrm{RA}_{\mathrm{ROOT}}$	[ROOT]	[]	$A_4 = A_3 \cup \{(\text{ROOT}, \mathit{ROOT}, \text{muß})\}$

- The tuple $< \sigma, \beta, A >$ on each line is a parser state.
- The transition on the next line is the corresponding class.

Features

- Features are extracted from a parser state c_i using feature templates.
- Features can use any portion of the parser state:
  - Any token on the stack/buffer.
  - Attributes of a token, such as its part-of-speech tag or lemma.

Example feature set (Kübler, McDonald, and Nivre 2009)

Address	Form	Lemma	POS-tag	Features	Deprel
σ_0	+	+	+	+	
σ_1			+		
LDEP[σ_0]					+
RDEP[σ_0]					+
β_0	+	+	+	+	
β_1	+		+		
β_2			+		
β_3			+		
LDEP[β_0]					+
RDEP[β_0]					+

Feature vectors

- To use techniques from linear algebra, we represent each feature as a fixed component in a vector.

Feature vectors

- To use techniques from linear algebra, we represent each feature as a fixed component in a vector.
- We can represent four features in a vector of four components:

a. $\sigma = [\cdots|\text{die}|\text{AWO}]$
b. $\sigma = [\cdots|\text{in}|\text{Japan}]$
c. $\sigma = [\cdots|\text{das}|\text{Auto}]$
d. $\sigma = [\cdots|\text{in}|\text{Sofia}]$
e. $\sigma = [\cdots|\text{Sofia}],\ \beta = [\text{gewesen}|\cdots]$

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

Feature vectors

- The parser state
  - $\sigma = [\cdots, \text{in}, \text{Sofia}]$
  - $\beta = [\text{gewesen}, \cdots]$
- Would correspond to the following feature vector:

a. $\sigma = [\cdots|\text{die}|\text{AWO}]$
b. $\sigma = [\cdots|\text{in}|\text{Japan}]$
c. $\sigma = [\cdots|\text{das}|\text{Auto}]$
d. $\sigma = [\cdots|\text{in}|\text{Sofia}]$
e. $\sigma = [\cdots|\text{Sofia}],\ \beta = [\text{gewesen}|\cdots]$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Softmax regression

Basic linear model

- We introduce a matrix $\mathbf{W} \in \mathbb{R}^{|T| \times |F|}$, where:
  - $|T|$: the number of transitions.
  - $|F|$: the number of features.

Basic linear model

- We introduce a matrix $\mathbf{W} \in \mathbb{R}^{|T| \times |F|}$, where:
  - $|T|$: the number of transitions.
  - $|F|$: the number of features.
- Each row of $\mathbf{W}$ is a transition-specific weighting of feature vectors.

Basic linear model

- We introduce a matrix $\mathbf{W} \in \mathbb{R}^{|T| \times |F|}$, where:
  - $|T|$: the number of transitions.
  - $|F|$: the number of features.
- Each row of $\mathbf{W}$ is a transition-specific weighting of feature vectors.
- If $\mathbf{x}$ is the feature vector of a parser state, then a simple guide would be:

$$g(\mathbf{x}) = \arg\max_{t \in T} (\mathbf{W}\mathbf{x})_t$$

The softmax function

The **softmax** function squashes the components of a vector $\mathbf{u}$, such that the components sum up to 1:

$$\mathrm{softmax}(\mathbf{u})_i = \frac{e^{\mathbf{u}_i}}{\sum_k e^{\mathbf{u}_k^u}}$$

The softmax function

The **softmax** function squashes the components of a vector $\mathbf{u}$, such that the components sum up to 1:

$$\mathrm{softmax}(\mathbf{u})_i = \frac{e^{\mathbf{u}_i}}{\sum_k e^{\mathbf{u}_k^u}}$$

- Softmax can represent a categorical probability distribution $p(y|\mathbf{x})$:
  - All output values are in the range $(0, 1)$.
  - $\sum_k \mathrm{softmax}(\mathbf{u})_k = 1$
  - $\mathrm{softmax}(\mathbf{u})_y$ is the probability of class y.

Softmax for classification

To convert our earlier scoring function, $\mathbf{Wx}$ to a probabilistic model:

Softmax for classification

To convert our earlier scoring function, $\mathbf{Wx}$ to a probabilistic model: apply softmax.

Softmax for classification

To convert our earlier scoring function, $\mathbf{W}\mathbf{x}$ to a probabilistic model: apply softmax.

$$
\begin{aligned}
p(y|x) &= \mathrm{softmax}(u)_y \\
&= \mathrm{softmax}(\mathbf{W}\mathbf{x})_y \\
&= \frac{e^{\mathbf{W}_y \mathbf{x}}}{\sum_k e^{\mathbf{W}_k \mathbf{x}}}
\end{aligned}
$$

Softmax for classification

$$p(y|x) = \mathrm{softmax}(u)_y = \mathrm{softmax}(\mathbf{W}\mathbf{x})_y = \frac{e^{\mathbf{W}_y \mathbf{x}}}{\sum_k e^{\mathbf{W}_k \mathbf{x}}}$$

Known as:

- **logistic regression** in statistics;
- **softmax layer** in neural networks;
- **conditional maximum entropy model** in CL.

Computing the softmax regression

Given $\mathbf{x} \in \mathbb{R}^4$ and $y \in \mathbb{R}^3$:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} & W_{1,4} \\ W_{2,1} & W_{2,2} & W_{2,3} & W_{2,4} \\ W_{3,1} & W_{3,2} & W_{3,3} & W_{3,4} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \right)$$

$$= \text{softmax}\left(\begin{bmatrix} W_{1,1} \cdot x_1 + W_{1,2} \cdot x_2 + W_{1,3} \cdot x_3 + W_{1,4} \cdot x_4 \\ W_{2,1} \cdot x_1 + W_{2,2} \cdot x_2 + W_{2,3} \cdot x_3 + W_{2,4} \cdot x_4 \\ W_{3,1} \cdot x_1 + W_{3,2} \cdot x_2 + W_{3,3} \cdot x_4 + W_{3,4} \cdot x_4 \end{bmatrix} \right)$$

Example

$$\mathbf{W} = \begin{bmatrix} 0.3 & -0.1 & 0.4 & -0.2 \\ -0.2 & 0.6 & 0.5 & -0.9 \\ 0.3 & -0.1 & 0.1 & 0.4 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Example

$$\mathbf{W} = \begin{bmatrix} 0.3 & -0.1 & 0.4 & -0.2 \\ -0.2 & 0.6 & 0.5 & -0.9 \\ 0.3 & -0.1 & 0.1 & 0.4 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{W}\mathbf{x}^\intercal = \begin{bmatrix} 0.1 & -1.1 & 0.7 \end{bmatrix}$$

Example

$$\mathbf{W} = \begin{bmatrix} 0.3 & -0.1 & 0.4 & -0.2 \\ -0.2 & 0.6 & 0.5 & -0.9 \\ 0.3 & -0.1 & 0.1 & 0.4 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{W}\mathbf{x}^\intercal = \begin{bmatrix} 0.1 & -1.1 & 0.7 \end{bmatrix}$$

$$\text{softmax}(\mathbf{W}\mathbf{x})^\intercal = \begin{bmatrix} 0.32 & 0.10 & 0.58 \end{bmatrix}$$