

Multitask Pointer Network for Discontinuous Constituent Parsing
An application to the German language

James Yu
jby21@cam.ac.uk

Faculty of Economics
University of Cambridge

15th January 2024

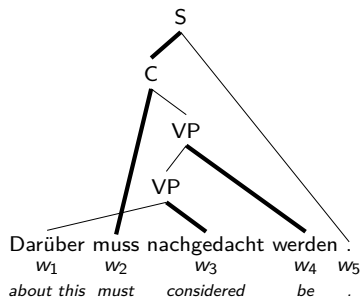
Why use a neural network to model grammar?

Linguistic preliminaries: constituent trees

Let $w = (w_1, \dots, w_L)$ be a sentence.

Definition

- A *constituent tree* is a rooted tree whose leaves are the words $(w_i)_{i=1}^L$ and internal nodes are constituents satisfying some constraints.
- A *constituent* is a triple (Z, \mathcal{Y}, h) containing, respectively, its label, yield, and lexical head.
- A constituent is *discontinuous* if its yield is not contiguous.

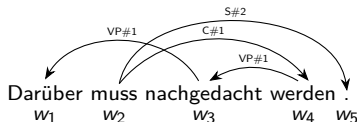
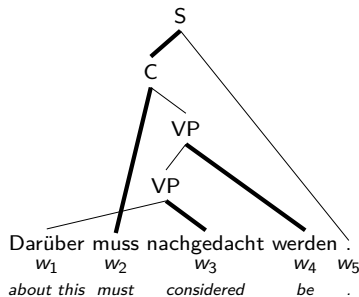


Reduction to dependency parsing

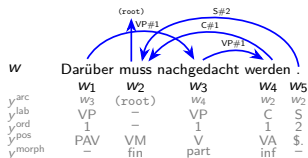
Definition

A *dependency tree* is a rooted tree spanning the words in the sentence $(w_i)_{i=1}^L$. Each edge is labelled and connects a *head word* (parent) to a *dependency* (child).

Fernández-González and Martins (2015) show that constituent trees are isomorphic to dependency trees in which the edges contain information about constituent labels and attachment order.



- Regressor: sentence $(w_i)_{i=1}^L$.
- Regressand: dependency tree, parts of speech and morphology.
- Bottom-up approach: think of *arcs* going from every child w_i to its parent $y_i^{\text{arc}} \in w \setminus w_i$.



Denote the regressand by $y = (y_i)_{i=1}^L$ where $y_i = (y_i^{\text{arc}}, y_i^{\text{lab}}, y_i^{\text{ord}}, y_i^{\text{pos}}, y_i^{\text{morph}})$ and with mild abuse of notation let $y_{<i} = (y_1, \dots, y_i)$ for each $i = 2, \dots, L$ and $y_{<1} = 0$.

Assumption

For each $i = 1, \dots, L$, the random variables $y_i^{\text{lab}}, y_i^{\text{ord}}, y_i^{\text{pos}}$ and y_i^{morph} are mutually independent conditional on $y_i^{\text{arc}}, y_{<i}$ and w .

We can decompose the conditional probability of y given w :

$$\begin{aligned}
 p_w(y) &= \prod_{i=1}^L p_w(y_i \mid y_{<i}) \\
 &= \prod_{i=1}^L \left\{ p_w(y_i^{\text{arc}} \mid y_{<i}) p_w(y_i^{\text{lab}} \mid y_i^{\text{arc}}, y_{<i}) \right. \\
 &\quad \cdot p_w(y_i^{\text{ord}} \mid y_i^{\text{arc}}, y_{<i}) p_w(y_i^{\text{pos}} \mid y_i^{\text{arc}}, y_{<i}) p_w(y_i^{\text{morph}} \mid y_i^{\text{arc}}, y_{<i}) \left. \right\}.
 \end{aligned}$$

Given an input sentence $w = (w_i)_{i=1}^L$ we generate *embeddings* $\omega = (\omega_i)_{i=1}^L$, where

$$\omega_i = \mathbf{WordEmbed}(w_i) \oplus \mathbf{CharEmbed}(w_i) \oplus \mathbf{BertEmbed}(w_i).$$

- **CharEmbed** is implemented using a CNN á la Chiu and Nichols (2016).
- BERT model pre-trained on German text by Chan et al. (2020).

Encoder: feed embeddings through a multi-layer bi-directional LSTM with skip-connections and dropout:

$$\mathbf{e} = (\mathbf{e}_i)_{i=0,\dots,n} = \mathbf{BiLSTM}(\omega).$$

(\mathbf{e}_0 represents the root pseudo-node.)

Decoder: feed embeddings through a single-layer uni-directional LSTM with dropout:

$$\mathbf{d} = (\mathbf{d}_i)_{i=1,\dots,n} = \mathbf{LSTM}(\omega).$$

The pointer network: bi-affine attention mechanism

Drawing on Dozat and Manning (2016) and Vinyals et al. (2015).

- Obtain dimension-reduced representations:

$$\mathbf{e}^{\text{arc}} = \text{MLP}_{\text{enc}}^{\text{arc}}(\mathbf{e}); \quad \mathbf{d}^{\text{arc}} = \text{MLP}_{\text{dec}}^{\text{arc}}(\mathbf{d}).$$

- Obtain latent features \mathbf{v}^{arc} :

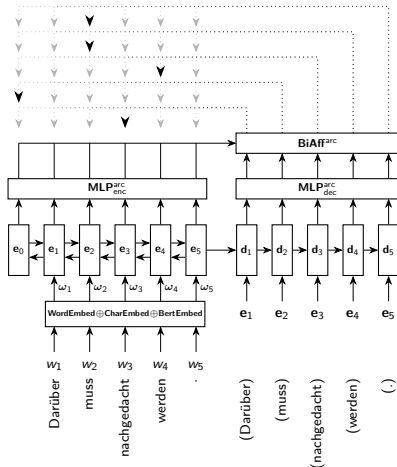
$$\begin{aligned} \mathbf{v}_{i,j}^{\text{arc}} &= \text{BiAff}^{\text{arc}}(\mathbf{e}_i^{\text{arc}}, \mathbf{d}_j^{\text{arc}}) \\ &:= \mathbf{e}_i^{\text{arcT}} \mathbf{U}_{\text{h-d}}^{\text{arc}} \mathbf{d}_j^{\text{arc}} \\ &\quad + \mathbf{e}_i^{\text{arcT}} \mathbf{U}_{\text{h-h}}^{\text{arc}} \mathbf{e}_i^{\text{arc}} + \mathbf{d}_j^{\text{arcT}} \mathbf{U}_{\text{d-d}}^{\text{arc}} \mathbf{d}_j^{\text{arc}} \\ &\quad + U_{\text{h}}^{\text{arc}} \mathbf{e}_i^{\text{arc}} + U_{\text{d}}^{\text{arc}} \mathbf{d}_j^{\text{arc}} + \mathbf{u}_{\text{bias}}^{\text{arc}}. \end{aligned}$$

- Obtain attention logits:

$$s_{i,j}^{\text{arc}} = \mathbf{u}_{\text{agg}}^{\text{arcT}} \tanh(\mathbf{v}_{i,j}^{\text{arc}}).$$

Fixing child w_j , the vector $\text{softmax}(s_{:,j}^{\text{arc}})$ can be interpreted as an estimated probability distribution over potential parents.

$$\hat{p}^{\text{arc}}(w_i \mid y_{<j}, w) = \text{softmax}(s_{:,j}^{\text{arc}})_i.$$



Part-of-speech and morphology: bi-affine classifier

Drawing on Dozat and Manning (2016).

- Encoder and decoder are shared across tasks. The remaining network is separately trained.
- Classification via a bi-affine architecture allows modelling of class probabilities conditional on arcs.

Example: part-of-speech classification.

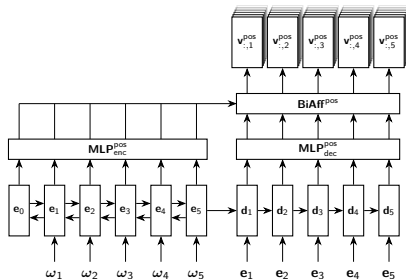
$$\mathbf{e}^{\text{pos}} = \text{MLP}_{\text{enc}}^{\text{pos}}(\mathbf{e}); \quad \mathbf{d}^{\text{pos}} = \text{MLP}_{\text{dec}}^{\text{pos}}(\mathbf{d}).$$

- Obtain class *logits* \mathbf{v}^{pos} :

$$\begin{aligned} \mathbf{v}_{i,j}^{\text{pos}} &= \text{BiAff}^{\text{pos}}(\mathbf{e}_i^{\text{pos}}, \mathbf{d}_j^{\text{pos}}) \\ &:= \mathbf{e}_i^{\text{posT}} \mathbf{U}_{\text{h-d}}^{\text{pos}} \mathbf{d}_j^{\text{pos}} + \mathbf{e}_i^{\text{posT}} \mathbf{U}_{\text{h-h}}^{\text{pos}} \mathbf{e}_i^{\text{pos}} + \mathbf{U}_{\text{h}}^{\text{pos}} \mathbf{e}_i^{\text{pos}} \\ &\quad + \mathbf{d}_j^{\text{posT}} \mathbf{U}_{\text{d-d}}^{\text{pos}} \mathbf{d}_j^{\text{pos}} + \mathbf{U}_{\text{d}}^{\text{pos}} \mathbf{d}_j^{\text{pos}} + \mathbf{u}_{\text{bias}}^{\text{pos}}. \end{aligned}$$

Fixing child w_j , the vector $\text{softmax}(\mathbf{v}_{i,j}^{\text{pos}})$ can be interpreted as an estimated probability distribution over its parts-of-speech conditional on having an arc to w_j over potential parents.

$$\hat{p}^{\text{pos}}(c \mid w_i, y_{<j}, w) = \text{softmax}(\mathbf{v}_{i,j}^{\text{pos}})_c.$$



Inference

Given $w = (w_i)_{i=1}^L$, estimate $y = (y_i)_{i=1}^L$ via *maximum likelihood*:

$$\hat{y} = \arg \max_y \left\{ \prod_{i=1}^L \left\{ \hat{p}_w(y_i^{\text{arc}} | y_{<i}) \hat{p}_w(y_i^{\text{lab}} | y_i^{\text{arc}}, y_{<i}) \right. \right. \\ \left. \left. \cdot \hat{p}_w(y_i^{\text{ord}} | y_i^{\text{arc}}, y_{<i}) \hat{p}_w(y_i^{\text{pos}} | y_i^{\text{arc}}, y_{<i}) \hat{p}_w(y_i^{\text{morph}} | y_i^{\text{arc}}, y_{<i}) \right\} \right\}.$$

The feasible region is very large, so the maximisation is approximated via *beam search*.

Training

The training process involves minimising the *cross-entropy* between the estimated distribution \hat{p} and the empirical distribution present in the dataset $(w^n, y^n)_{n=1}^N$.

$$\min_{\hat{p}} \left\{ \sum_{n=1}^N \log \prod_{i=1}^{L_n} \left\{ \hat{p}_w(y_i^{\text{narc}} | y_{<i}^n) \hat{p}_w(y_i^{\text{nlab}} | y_i^{\text{narc}}, y_{<i}^n) \right. \right. \\ \left. \left. \cdot \hat{p}_w(y_i^{\text{nord}} | y_i^{\text{narc}}, y_{<i}^n) \hat{p}_w(y_i^{\text{npos}} | y_i^{\text{narc}}, y_{<i}^n) \hat{p}_w(y_i^{\text{nmorph}} | y_i^{\text{narc}}, y_{<i}^n) \right\} \right\} \\ = \min_{\hat{p}} \left\{ \text{loss}^{\text{arc}} + \text{loss}^{\text{lab}} + \text{loss}^{\text{ord}} + \text{loss}^{\text{pos}} + \text{loss}^{\text{morph}} \right\}.$$

This is approximated via *SGD* with *Nesterov momentum*.

The TIGER treebank is a widely-used corpus of $\sim 50\,000$ constituency trees. Its main textual basis is the *Frankfurter Rundschau*.

- 97 % of the dataset is usable as training examples, with train/dev/test split of 80/10/10 %.

Model (with BERT)	F1	Disc. F1
F.-González & G.-Rodríguez (2022)	89.8	71.0
Corro (2020)	90.0	62.1
Chen & Komachi (2023)	89.6	70.9
This work	89.5	82.2

Table: Comparison of overall F1-score and F1-score measured only on discontinuous constituents (Disc. F1), calculated using disco-dop as standard practice

Model	pos	morph (avr)
Kondratyuk et al. (2018)	98.58	98.97
Müller et al. (2013)	98.20	98.27
Schnabel & Schütze (2014)	97.50	97.76
This work	99.16	99.54

Table: Comparison of part-of-speech (pos) and morphology accuracies (%). Morphology accuracies are the average of accuracies for case, degree, gender, mood, number, person and tense.



Chan, B., Möller, T., Pietsch, M., & Soni, T. (2020). *German BERT (bert-base-german-cased)*. Retrieved September 1, 2023, from <https://huggingface.co/bert-base-german-cased>



Chen, Z., & Komachi, M. (2023). Discontinuous combinatory constituency parsing. *Transactions of the Association for Computational Linguistics*, 11, 267–283.



Chiu, J. P., & Nichols, E. (2016). Named entity recognition with bidirectional lstm-cnns. *Transactions of the association for computational linguistics*, 4, 357–370.



Corro, C. (2020). Span-based discontinuous constituency parsing: A family of exact chart-based algorithms with time complexities from $O(n^6)$ down to $O(n^3)$. *Empirical Methods in Natural Language Processing*.



Dozat, T., & Manning, C. D. (2016). Deep biaffine attention for neural dependency parsing. *International Conference on Learning Representations*.



Fernández-González, D., & Gómez-Rodríguez, C. (2022). Multitask pointer network for multi-representational parsing. *Knowledge-Based Systems*, 236, 107760.



Fernández-González, D., & Martins, A. F. (2015). Parsing as reduction. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1523–1533.



Kondratyuk, D., Gavenčiak, T., Straka, M., & Hajič, J. (2018). Lemmatag: Jointly tagging and lemmatizing for morphologically-rich languages with brnns. *arXiv preprint arXiv:1808.03703*.



Müller, T., Schmid, H., & Schütze, H. (2013). Efficient higher-order crfs for morphological tagging. *Proceedings of the 2013 conference on empirical methods in natural language processing*, 322–332.



Schnabel, T., & Schütze, H. (2014). FLORS: Fast and simple domain adaptation for part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 2, 15–26.



Vinyals, O., Fortunato, M., & Jaitly, N. (2015). Pointer networks. *Advances in neural information processing systems*, 28.