Lexicalized Constituency Parsing for Middle Dutch: Low-resource Training and Cross-Domain Generalization

Yiming Liang, Fang Zhao
Universiteit Gent, Université Paris Cité & Laboratoire de linguistique formelle
Blandijnberg 2 9000 Gent Belgium, 8 Rue Albert Einstein 75013 Paris France
yiming.liang@ugent.be, fang.zhao@etu.u-paris.fr

Abstract

Recent years have seen growing interest in applying neural networks and contextualized word embeddings to the parsing of historical languages. However, most advances have focused on dependency parsing, while constituency parsing for low-resource historical languages like Middle Dutch has received little attention. In this paper, we adapt a transformer-based constituency parser to Middle Dutch, a highly heterogeneous and low-resource language, and investigate methods to improve both its in-domain and cross-domain performance. We show that joint training with higher-resource auxiliary languages increases F1 scores by up to 0.73, with the greatest gains achieved from languages that are geographically and temporally closer to Middle Dutch. We further evaluate strategies for leveraging newly annotated data from additional domains, finding that fine-tuning and data combination yield comparable improvements, and our neural parser consistently outperforms the currently used PCFG-based parser for Middle Dutch. We further explore feature-separation techniques for domain adaptation and demonstrate that a minimum threshold of approximately 200 examples per domain is needed to effectively enhance cross-domain performance.

Keywords: Middle Dutch, constituency parsing, transformers, low-resource languages, domain adaptation

1. Introduction

Neural networks with contextualized word representations have proven effective for PoS tagging and syntactic parsing (Kitaev and Klein, 2018; Han and Eisenstein, 2019; Lim et al., 2020, among others), reducing the manual effort needed to build linguistic treebanks, which are crucial for quantitative linguistic studies and digital humanities research. Recently, there has been a growing interest in applying them to parse lesser-resourced languages like historical languages (e.g., Grobol and Crabbé, 2021, for Old French; Kulick et al., 2022, for Early Modern English).

While most advances have focused on dependency parsing (e.g., Vania et al., 2019; Zhang, 2022; Grobol et al., 2022), much less attention has been paid to constituency parsing, as the latter one has been argued to be more difficult in terms of accuracy and the efficiency of parsing algorithms (Kübler et al., 2009; Cross and Huang, 2016). However, constituency-parsed corpora remain widely used in linguistic research, in particular among historical linguists, because they provide the hierarchical structure of sentences in terms of smaller units that better align with many influential syntactic frameworks (e.g., traditional Phrase-Structure Grammar (Bloomfield, 1933), Generative Grammar (Chomsky, 1957)), and are easily accessed and explored with the toolkit *CorpusSearch 2* [1] (Randall, 2010) provided by the University of Pennsylvania. Constituency parsing has also been shown to provide useful structured input for downstream NLP tasks and improve performance in lots of semantic tasks, such as Semantic Role Labeling (Wang et al., 2019; Bastianelli et al., 2020), Target Identification (Bastianelli et al., 2020) and nested Named Entity Recognition (Wang and Lu, 2018; Fu et al., 2021; Yang and Tu, 2022).

Despite its importance, there has been little work on applying transformer-based models and contextualized word representations to historical language constituency parsing (but see Kulick et al., 2022; Sapp et al., 2023; Nie et al., 2023), and no work has been focused on Middle Dutch parsing. The majority of Penn-style historical corpora still rely on rule-based shallow parsing or statistical PCFG parsers, which require extensive manual effort for postprocessing and correction (Booth et al., 2020; Farasyn et al., 2022). In this paper, we focus on constituency parsing of Middle Dutch (1150-1500), a low-resource language with only around 4,000 syntactically annotated sentences of high heterogeneity in spelling, syntax and genres. We apply the Berkeley Neural parser (*Benepar*, Kitaev and Klein 2018; Kitaev et al. 2019) combined with BERT representations (Devlin et al., 2019) to Middle Dutch, and investigate how annotated corpora from richer-resourced languages and domain-adaptation techniques can be leveraged to train a constituency parser that performs well both in-domain and on out-of-domain texts under low-resource conditions. Concretely, first, we train *Benepar* on the largest available parsed text *Etstoel* (approximately 2,000 sentences), and examine whether incorporating PoS-tag prediction as an auxiliary task **(Phase I)**

[1] https://corpussearch.sourceforge.net/CS.html

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 10276–10290
11–16 May 2026.

and joint training with auxiliary languages can improve in-domain performance **(Phase II)**. Second, based on the best-performing configuration, we explore its zero-shot and few-shot generalisation to new texts and genres, comparing data combination and fine-tuning strategies. We further experiment with feature separation techniques (Kim et al., 2016; Li et al., 2020, 2022) to evaluate their role in low-resource constituency parsing (**Phase III**). We also compare the performance of *Benepar* with the statistical PCFG *Bikel* parser (Bikel, 2002, 2004), currently in use for parsing Middle Dutch texts.

We address two main research questions:

RQ1: *How to effectively train a neural constituency parser when annotated data are limited?* We find that while including PoS-tag prediction as an auxiliary task brings no additional benefit, joint training with auxiliary languages substantially improves accuracy (up to +0.73 F1 and at best 86.21 F1), especially when the auxiliary language is geographically and temporally close to Middle Dutch.

RQ2: *How well does the parser generalize to new texts and genres, and how to improve cross-domain performance under high heterogeneity?* Our experiments show that *Benepar* strongly outperforms the statistical *Bikel* parser even in zero-shot conditions. Fine-tuning and data-combined retraining with sentences from the new domain yield comparable results, and the parser begins to show improvement with as few as 10 examples and surpasses 70 F1 after 100 examples. We further examine feature separation techniques for domain adaptation and find that, with around 200 examples per domain, cross-domain performance exceeds 74.7 F1 across all three new domains, while smaller datasets yield little or no improvement.

Overall, our results not only produce a state-of-the-art Middle Dutch parser, but also yield broader insights into neural constituency parsing for low-resource and highly heterogeneous historical languages, particularly in the context of incremental treebank construction, where annotated data become available over time[2].

2. Related work

2.1. Constituency parsing

Constituency parsing, which seeks to represent the structure of sentences in terms of hierarchical phrases/constituents, has seen continuous progress from statistical probabilistic context-free grammar (PCFGs) (Charniak, 1997; Collins, 1999; Bikel, 2004) to neural approaches (Cross and Huang, 2016; Choe and Charniak, 2016; Crabbé, 2015; Coavoux and Crabbé, 2017), and more recently, to self-attentive models with contextualised word embeddings (Kitaev and Klein, 2018; Kitaev et al., 2019; Zhou and Zhao, 2019). While these advances have led to highly accurate parsers for modern languages with abundant annotated data such as the Penn Treebank (PTB), much less attention has been given to historical or low-resource languages, where annotated corpora are scarce and domain variation is high. In fact, Penn-style historical treebanks are often created using rule-based shallow parsers, which rely heavily on manual effort for creating rules and performing corrections on the output (Booth et al., 2020; Farasyn et al., 2022; Arnardóttir and Ingason, 2020). Among the very few historical treebanks that use an automatic parser, the Historical High German corpus (IPCHG, Sapp et al., 2024) uses *Benepar* (Kitaev and Klein, 2018; Kitaev et al., 2019) for automatic parsing: they first train *Benepar* in richer-resourced Middle Low German (CHLG, Booth et al. 2020) and apply it to the target Early New High German (ENHG), and subsequently retrain the parser on combined data consisting of CHLG and the manually corrected ENHG trees (Sapp et al., 2023). Their experiments yield promising results (F1 65 at best), but the out-of-domain performance is far from satisfactory (below F1 50 in general), and the cross-domain performance remains unknown. Other research also explores *Benepar* on large historical Penn-style corpora: Arnardóttir and Ingason (2020) report that *Benepar* achieves an F1 of 84.74 when trained and evaluated on Icelandic Parsed Historical Corpus (73,000 matrix clauses, Rögnvaldsson et al., 2012); Kulick et al. (2022) achieve an F1 of 90.53 with 31 function tags trained on Parsed Corpus of Early Modern English (28,000 sentences, Kroch et al. 2004). Nie et al. (2023) propose a delexicalized version of *Benepar* and report an F1 of 64.72 in a zero-shot setup, transferring a model trained on Modern High German (TIGER, Brants et al. 2004) to Middle High German. However, no attempts have yet been made to apply constituency parsing to Middle Dutch or Historical Dutch more generally.

2.2. Domain adaptation

Since Middle Dutch collectively refers to several dialects spoken over several centuries and there was no standardized variety (Hüning and Vogl, 2009), it exhibits substantial variation in spelling and syntax across texts. This makes domain adaptation particularly crucial for parsing Middle Dutch, as models can suffer from dramatic performance drop when new texts and genres differ from training data (Ramponi and Plank, 2020; Marzinotto et al., 2019; Joshi et al., 2018). Therefore, we review domain adaptation techniques, with a particular attention to low-resource settings.

[2]The code for our experiments is available via https://github.com/Mehechiger/Goeiemiddutch

A long-standing strand of domain adaptation approaches rests on explicitly separating domain-invariant features from domain-specific ones (e.g., Daumé III, 2007; Kim et al., 2016; Sato et al., 2017; Li et al., 2020, 2022). In particular, the so-called *shared-private* model places a *shared* encoder for domain-invariant features alongside one or more *private* encoders for domain-specific signals (Kim et al., 2016). Sato et al. (2017) first apply adversarial training (Goodfellow et al., 2014; Ganin and Lempitsky, 2015) to learn domain-invariant features in parsing. They design a gating mechanism to mix representations from the shared and private encoders, then they feed the mixed representations to the parser network. However, while these methods consistently improve performance in most cases, they are harmful when target data are scarce. Indeed, target domain encoders and gates may not be well optimized when lacking data, which in turn harms generalization.

Li et al. (2020, 2022) publish a series of studies tackling the low-resource setting in domain adaptation in dependency parsing. Following Bousmalis et al. (2016), Li et al. (2020, 2022) enforce orthogonality constraints to encourage the domain-specific features to be mutually exclusive with the shared features to reduce redundancies in the shared and private feature spaces. To alleviate underfitting due to the lack of target domain labeled data, Li et al. (2020) introduce fused target-domain word representations, which combine source and target private representations as the final domain-specific representations when the input word is from the target domain. From a similar angle, Li et al. (2022) apply a dynamic matching network (Jang et al., 2019) on the shared-private model to let the target encoder mimick well-trained source features, leveraging the in-depth relevance of domain-specific encoders and thus alleviating target domain underfitting. While these methods have proven effective for small datasets, their lowest-resource domain studied consists of 1,645 examples. It is therefore interesting to explore the effectiveness of these methods in even lower-resource conditions like Middle Dutch, and the amount of target data required.

3. Data

3.1. Parsed Data of Middle Dutch

Middle Dutch is the term used for the language varieties used between approximately 1150 and 1500 in the territory covered nowadays by the Netherlands and Flanders region of Belgium. Due to the absence of standard variety of Dutch and the dominance of Latin or French in writing, administration and nobility, Middle Dutch is a collection of dialects spoken over several centuries, and featured by "a huge variation in the grammatical structure, the pronunciation and the spelling" (Hüning and Vogl, 2009, p. 257), which implies big challenges for domain adaptation of the parser from one text to another. At the morpho-syntactic level, as stated in Kerckvoorde (1993); Hüning and Vogl (2009), Middle Dutch is most characterized by: 1) a rigid V2 order in main clause and a more flexible word order in subordinate clauses, whereas the finite verb is placed at the end of the subordinate clause in Modern Dutch; 2) more flexibility of word order within a nouns phrase (NP) because of modifier postpositioning; 3) a much richer inflection system with a four-case-inflectional paradigm on nouns, adjectives, articles and numerals, which disappeared from the 17th century on. As no word embeddings are available for Middle Dutch, the language's flexible word order and rich inflectional morphology may pose additional challenges for parsing and word representation, particularly when adapting a pre-trained BERT model trained on later stages of Dutch. An additional complexity comes from the nested embedded clauses and long sentences featured by legal charters, one of the most well-documented textual genres for Middle Dutch.

Our experiments are based on the *Penn-style Treebank of Middle Dutch* (Simonenko and Liang, p.c.), which contains the following Middle Dutch parsed and corrected texts as of the day of the study:

- *Etstoel*: a sample of 15th-century legal charters from the Etstoel-Drenthe corpus (van Kemenade and Postma, p.c.), which contain manually corrected annotations of PoS tags, lemmas, and constituency syntactic trees.
- *CRM14*: a sample of legal charters composed in the 14th century taken from the corpus *Het Corpus Van Reenen-Mulder van 14e-eeuwse Middelnederlandse oorkonden* (van Reenen and Mulder, p.c.), which contains manually corrected annotations of PoS tags and lemmas.[3]
- *Tafel* ("Tafel van den Kersten Ghelove"): written by Dirck van Delft in 1404, religious texts
- *Trappen* ("Seven Trappen"): written by Jan van Ruusbroec at 1359-1362, religious texts

A parsing tree from *Etstoel* is illustrated by Figure 1. To the best of our knowledge, these are the only parsed texts of Middle Dutch. More texts are being parsed and manually checked as part of the *Penn-style Treebank of Middle Dutch*. The four available texts are PoS-tagged and lemmatised either by being manually annotated by philologists, or using the GaLAHaD PoS-tagger and lemmatiser[4], and au-

[3] https://middelnederlands.nl/corpora/crm14/

[4] https://github.com/instituutnederlandsetaal/galahad

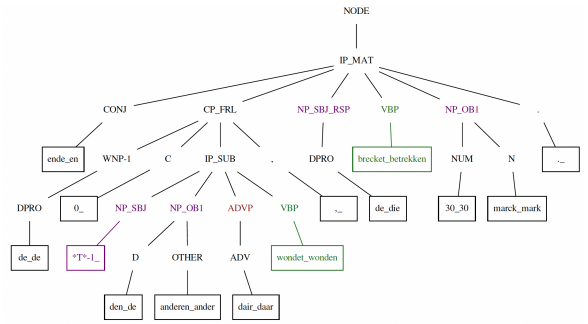


Figure 1: The constituency tree of the sentence *ende de den anderen dair wondet, de brecket 30 marck* "and the one who wounds the other there shall pay (or owes) 30 marks" from Etstoel

tomatically parsed with Bikel parser[5] (Bikel, 2002, 2004). All automatic annotations, including PoS tags, lemmas, and constituency trees, are subsequently corrected and carefully reviewed by human annotators. These texts are annotated following the Penn standard for historical English (Santorini, 2022), with adaptations for Dutch-specific conventions.

3.2. Data for Auxiliary Languages

It has been shown that cross-lingual transfer generally improves adaptation from high-resource modern languages to low-resource languages (Wolf et al., 2020; Ruder, 2019; Lauscher et al., 2020; Nie et al., 2023; Kitaev et al., 2019). Since Middle Dutch has limited annotated data, we experiment with the following Penn-style treebanks for auxiliary languages, differing in data sizes (cf. Table 3) and relatedness to Middle Dutch:

- Historical treebanks:
  - Historical English (AllHE): Old English (OE, Taylor et al. 2003), Middle English (ME, Kroch and Taylor 2000), Early Modern English (EModE, Kroch et al. 2004), Modern British English (ModBE, Kroch et al. 2016)
  - Historical High German (AllHG): Middle High German (MHG), Early New High German (ENHG), both from Sapp et al. (2024)
  Middle Low German (CHLG, Booth et al. 2020)
  - Historical French (AllHF): Old French (OF), Middle French (MF), both from Martineau and Santorini (2010); Kroch and Santorini (2010)
  - Historical Portuguese (HP, Galves and Faria 2017)
- Modern treebanks: Modern English (PTB, Bies et al. 2015), Modern Chinese Treebank (CTB, Xue et al. 2013), Modern Dutch (LassySmall, van Noord et al. 2013)

[5]https://dbparser.github.io/dbparser/

All historical treebanks follow the Penn standard for Historical English (Santorini, 2022) for constituency parsing. Given the wide chronological span of historical languages and the syntactic changes that occur over time, the syntactic structures can differ substantially across historical periods. Therefore, most historical treebanks focus on one period. For corpora that are not explicitly divided, we separate them according to conventionally established periods, as in the cases of Historical High German and Historical French. Although modern treebanks also follow Penn-style, their syntactic tags and annotation conventions differ non-negligibly from those of historical treebanks. For this reason, we do not combine Modern British English and PTB, despite their temporal proximity.

4. Experiments

4.1. Data Preparation

For the first research question, which concerns low-resource parser training, we train the *Benepar* parser only on *Etstoel*, the largest parsed text. We explore two strategies: Part-of-Speech tags prediction as an auxiliary task (Phase I) and auxiliary language training (Phase II). At the end of each phase, the best-performing option will be kept for the next phase. After obtaining the best parser at the end of Phase II, we address the second research question, which addresses the generalization capacity of the parser to new texts and a new genre. For this purpose, we use the remaining three texts (CRM14, Tafel, and Trappen) in Phase III.

4.1.1. Data preprocessing

For all annotated data, including the auxiliary language corpora, empty categories and coreferential indices are removed. Sentences longer than 100 words are excluded, as they exceed the maximum sequence length supported by *Benepar*. More details of data preprocessing and postprocessing are provided in Appendix A.1. Unlike studies on constituency parsing in modern languages (e.g. Cross and Huang, 2016; Kitaev et al., 2019; Coavoux and Crabbé, 2017), which remove all function tags during training and evaluation, we retain them because they provide crucial syntactic information essential for linguistic analysis.

4.1.2. Evaluation metrics

For **evaluation**, we use *evalb*, a standard metric that compares spans and labels in gold and predicted trees, provided in the release of Kitaev et al. (2019). Following Kulick et al. (2022) and Sapp et al. (2023), we report results with **all function tags** retained (e.g., "NP-SBJ" is treated as an atomic unit).

To this end, we modify the *evalb* source code to preserve function tags during evaluation. Punctuation is excluded from PoS tagging and parsing evaluation, following Kitaev and Klein (2018) and Kitaev et al. (2019). For comparability with previous work in modern languages, Appendix A.2 also reports scores of the best models[6] on each Middle Dutch text with function tags removed (e.g., treating "NP-SBJ" as "NP").

4.1.3. Cross-validation and resampling

Since *Etstoel* is relatively small, in Phases I and II, we apply 10-fold cross-validation to *Etstoel* to enhance the reliability of our results.[7] For each run, eight folds are used for train, one fold serves as the dev set to determine the number of training epochs (early stopping), and one fold is reserved for test. In Phase II, for each auxiliary language, the data are randomly split into 2,000 examples for dev, 2,000 examples for test, and the rest for train. The same splits are used for all experiments. As the main objective is to optimize the Middle Dutch parser, early stopping is used and the number of training epochs is determined based on the *Etstoel* dev set defined in Phase I. All scores reported in Phases I and II for the parser are averaged across 10-fold cross-validation of *Etstoel* test set.

Dataset	Train	Dev	Test	Total
Etstoel	1568	195	195	1958
CRM	10-100-200	n/a	169	369
Tafel	10-100-200	n/a	574	774
Trappen	10-100-200	n/a	751	951

Table 1: Middle Dutch data splits.

In Phase III, for the zero-shot and fine-tuning experiments, the parser is retrained on nine folds of *Etstoel* with the best auxiliary language to obtain the best in-domain model, while the remaining fold is used as the development set for early stopping.[8] For experiments that combine *Etstoel* with other Middle Dutch data (combined data and adversarial training), the same nine folds of *Etstoel* are used for training and the remaining fold for development. As for the three remaining texts (considered as three new domains), since they are much smaller than *Etstoel*, a 10-fold cross-validation is problematic due to limited test data per fold. Therefore, following Sapp et al. (2023), we use 10-resampling instead. In particular, for each text, 200 sentences are taken for training, and the remaining sentences are used for testing, with the sampling process repeated 10 times under each condition. Because of the limited data size, no separate development set is used for the new texts; early stopping is again determined based on the *Etstoel* dev set defined in Phase II.[9] Table 1 shows the split of train/dev/test for each treebank. Scores reported for the parser are averaged across ten resampling of *CRM14*, *Tafel* and *Trappen* in Phase III.

4.2. Parsers

Given the robust performance of *Benepar* in constituency parsing across modern languages (Kitaev et al., 2019) and historical languages (Arnardóttir and Ingason, 2020; Kulick et al., 2022), we choose it for our experiments with Middle Dutch. *Benepar* is a span-based neural parser that assigns a score to every possible span of words with a label of a sentence, and uses a modified version of the CKY algorithm (Gaddy et al., 2018) to combine these span scores and construct the parse tree with the highest overall score. PoS tags are assigned using a separate classifier on top of the encoder output, which is jointly optimized with the span classifier (cf. Kitaev and Klein 2018). We use the publicly released code of Kitaev et al. (2019).[10]

Pretrained embeddings have been shown to improve cross-domain parsing performance (Yang et al., 2022; Fried et al., 2019; Kitaev et al., 2019). Previous studies further indicate that contextualized embeddings trained on data from time periods closer to the target yield better results (Kulick et al., 2022; Grobol et al., 2022). Accordingly, we use the dbmdz BERT embeddings (Devlin et al., 2019), which are trained on historical Dutch.[11] Due to the discrepancy in time and genre of the dbmdz BERT training data (1618-1879, newspapers) and our Middle Dutch data (14th-15th centuries, legal charters and religious texts), we continue to fine-tune the BERT parameters along with parser training, as in the original implementation of *Benepar* (Kitaev et al., 2019). Li et al. (2020, 2022) further show that continued pre-training of BERT with target domain raw text prior to parser training can significantly improve domain adaptation. However, since it relies on substantial unlabeled data and diverse text genres (Gururangan et al., 2020; Li et al., 2020, 2022; Grobol et al., 2022)), we leave it for future work.

For all experiments described in this paper, we trained our Benepar models using a single Nvidia

[6] We report scores for the best Bikel and Benepar models among all our model variants : with/without PoS tag prediction, with/without target domain data, with/without auxiliary languages and with which auxiliary language, etc.

[7] See the suggestions of Gorman and Bedrick (2019).

[8] The auxiliary language data is randomly resplit into 2,000 examples for development and the rest for training.

[9] Models not using *Etstoel* (fine-tuning models in Phase III) are trained for an arbitrary of 50 epochs.

[10] https://github.com/nikitakit/self-attentive-parser

[11] https://huggingface.co/dbmdz/bert-base-historic-dutch-cased

RTX A6000 GPU card. A table detailing the hyperparameters used during this training process can be found in Appendix A.3.

As a baseline model, we use the *Bikel* parser (Bikel, 2004), a statistical lexicalized PCFG parser, as it is currently used for constituency parsing of Middle Dutch. It extends the models of Collins (1996, 1997, 1999) with a more flexible and modular implementation, and enables multilingual parsing. The *Bikel* parser is trained and evaluated using 10-fold cross-validation (Phases I, II and III) on *Etstoel* and 10 resampling runs on the target data (Phase III), following the same data splits as *Benepar*, but without using the dev set, since early stopping is not available. Furthermore, since the *Bikel* parser does not support auxiliary task training or joint training with auxiliary languages, we train it using gold PoS tags only and without any auxiliary language. We also consider using an LLM as another baseline model for parsing Middle Dutch, as large language models are likely to have been exposed to historical Dutch data during pre-training. However, when we experiment with GPT-4.1 under OpenAI on the Etstoel texts, both with and without finetuning, the model often modifies the original Middle Dutch input sentence when producing the output constituency tree. This occurs even when the prompt explicitly instructs the model to keep the sentence identical to the input and the temperature is set to zero. Since faithful preservation of the original sentence is essential for syntactic parsing, this hallucination behavior makes generative LLMs unsuitable for parsing, and we therefore do not include them as a baseline in the present study.

4.3. Phases I: Auxiliary Task

Coavoux and Crabbé (2017) have shown that word-level auxiliary tasks improve the performance of their constituency parser based on a bi-LSTM architecture. Therefore, we explore whether training with the PoS-tag prediction as an auxiliary task improves the self-attentive parser for Middle Dutch. In *Benepar* (Kitaev et al., 2019), two additional randomly initialized self-attention layers are applied to the output of BERT. The parsing module and the optional PoS-tagging module are added on top of these layers as two parallel, independent tasks (cf. Figure 2a, without auxiliary languages).

Results Table 2 shows the result of 10-fold cross-validation of *Bikel* parser and *Benepar* on *Etstoel*. *Benepar* significantly outperforms *Bikel* parser regardless of the presence of the auxiliary task, showing that the self-attentive model with pretrained embeddings is more effective in constituency parsing than statistical models, even when trained with less than 2,000 sentences. Regarding the auxiliary task, *Benepar* achieves comparable results when PoS-tag prediction is added or not. This may be due to the self-attentive encoder already implicitly capturing part-of-speech information during parsing. For the remaining experiments, we choose the version with PoS prediction, as this task remains useful for the construction of linguistic treebanks.

Condition	Parsing F1	Δ
Bikel	71.71 (1.22)	-13.63
Benepar (w/o PoS pred.)	85.34 (1.08)	0.00
Benepar (w/ PoS pred.)	85.48 (1.5)	+0.14

Table 2: Parsing F1 scores on the *Etstoel* test set, evaluated by evalb, averaged across 10-fold cross-validation, and the difference Δ with *Benepar* without PoS prediction. Standard deviations are shown in parentheses.

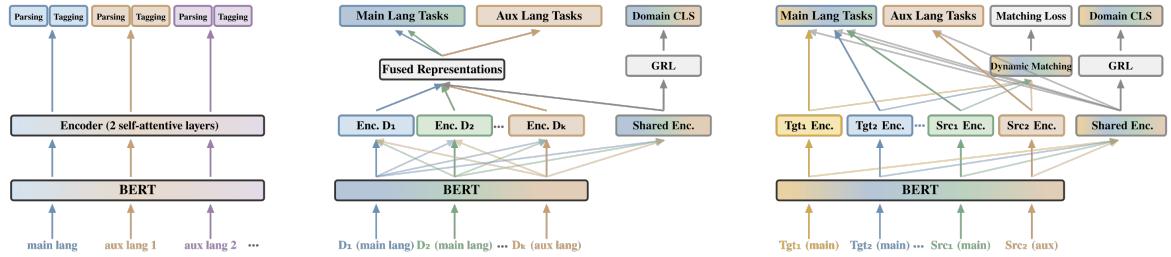
4.4. Phase II: Auxiliary Languages

Kitaev et al. (2019) report that incorporating a high-resource language as an auxiliary task can improve parsing performance on lower-resource languages, and that the size of the auxiliary language dataset has a greater impact than morphological relatedness. Sapp et al. (2023) also show that the related high-resource CHLG improve the parsing of the target low-resourced ENHG. Given the limited size of parsed Middle Dutch data, we investigate whether existing large-scale constituency treebanks in other languages improve Middle Dutch parsing.

Unlike Sapp et al. (2023), who combine source language and target language and use a single parsing module, we assign a separate parsing module to each language and train them jointly, so that Middle Dutch and each auxiliary language maintain distinct tagsets and parameters for parsing, as shown in Figure 2a, following Kitaev et al. (2019).[12] However, instead of using a multilingual BERT as Kitaev et al. (2019), we employ a historical Dutch BERT model (see Section 4.2), as our goal is to maximize Middle Dutch performance rather than build a universal multilingual parser. The tested auxiliary languages are described in Sectoin 3.2, chosen by data sizes and relatedness to Middle Dutch. For each auxiliary language, we treat *Etstoel* as the main language (main lang) and perform bilingual joint training. For languages with multiple historical stages, we additionally conduct multilingual training in which each auxiliary language has a separate PoS-tagging and parsing module (e.g., MHG = aux lang 1, ENHG = aux lang 2, in Figure 2a). The data splits for *Etstoel* and auxiliary languages are detailed in Section 4.1.3, and we apply 10-fold cross-validation as in Phase I.

Results Table 3 summarizes the parsing F1 of

[12]We implement this functionality following Kitaev et al. (2019), as it is missing from their public Github code.



(a) Auxiliary task and languages (b) Fused Representations (c) Multi-source Dynamic Matching

Figure 2: Model architectures. Enc. stands for encoder and GRL for gradient reversal layer.

Aux	ENHG	EModE	MHG	AllHE	CHLG	AllHG	MF	ME	AllHF	OF	HP	Lassy	ModBE	PTB	OE	CTB
# train sents	11.7k	28.4k	6.5k	267.9k	6.3k	18.3k	39.0k	79.5k	110.4k	71.4k	55.4k	31.9k	54.2k	45.2k	105.7k	64.8k
On Aux	78.80	84.17	81.31	82.97	83.30	78.75	86.08	83.16	83.61	83.84	83.75	80.82	86.46	**89.77**	82.73	14.96
	(0.92)	(0.72)	(1.21)	(1.66)	(0.73)	(3.29)	(0.99)	(1.52)	(1.38)	(0.75)	(0.58)	(0.68)	(1.44)	(0.61)	(1.54)	(1.27)
On *Etstoel*	**86.21**	**86.21**	86.15	86.14	86.13	86.12	86.08	86.01	86.00	85.97	85.81	85.78	85.77	85.54	85.51	85.50
	(1.13)	(1.19)	(1.05)	(0.95)	(1.19)	(1.12)	(1.22)	(1.24)	(1.06)	(1.17)	(1.22)	(1.28)	(1.31)	(1.29)	(1.30)	(1.27)
Δ NoAux(85.48)	**+0.73**	**+0.73**	+0.67	+0.66	+0.65	+0.64	+0.60	+0.53	+0.52	+0.49	+0.33	+0.30	+0.29	+0.06	+0.03	+0.01

Table 3: Parsing F1 scores on auxiliary language test sets and the *Etstoel* test set, ordered by F1 difference Δ on *Etstoel* when *Benepar* is trained on *Etstoel* train with *vs.* without each auxiliary language (last row). Averages over 10-fold cross-validation; standard deviations in parentheses.

Etstoel without auxilary language, or with different combination of auxiliary languages. We find that auxiliary languages consistently improve parsing on *Etstoel*, but to varying degrees. The top four auxiliary languages are Early New High German (ENHG), Early Modern English (EModE), Middle High German (MHG) and Middle Low German (CHLG). They are all geographically and temporally closest to Middle Dutch, suggesting that morphological proximity contributes greatly to effective transfer. By contrast, modern languages such as Chinese (CTB), Modern English (PTB and ModBE), and Modern Dutch (Lassy) yield the smallest gains, likely due to differences in the temporal gap from Middle Dutch. Old English (OE) offers only a weak benefit (+0.03), whereas Old French (OF)'s improvement is much larger (+0.49), which could also be explained by temporal proximity: OE ends around the 11th century, while OF last to the 13th century, closer to *Etstoel*'s composition time (15th century). Our findings indicate that temporal and geographical relatedness outweigh corpus size: although ModBE, PTB, and OE are among the largest auxiliary datasets, their impact is minimal, MHG and CHLG, each more than nine times smaller, rank among the most effective. Overall, our results confirm Kitaev et al. (2019) that auxiliary languages enhance target language parsing accuracy, but we find that temporal and geographical proximity play a more decisive role than data size.

4.5. Phase III: Domain adaptation

After obtaining a good model on single source text, we are interested in finding out ways to improve the parser's cross-domain generalization with data of high heterogeneity and limited quantity. We first test our best model on *Etstoel* obtained after Phase II directly on the three target domains (CRM, Tafel and Trappen) without any further adaptation (**0-shot**). We then fine-tune this model with target domain data in a few-shot learning scheme (**fine-tune**) as suggested by Chen et al. (2020). Following Sapp et al. (2023), we also attempt with models retrained with combined data: Etstoel + one or multiple of the target domains using the same auxiliary language that yield the best model in Phase II (**combined**). Additionally, we compare a *Bikel* parser baseline trained with the same combined data (**Bikel combined**). Finally, we apply feature separation techniques with adversarial training (adapted from Sato et al., 2017; Li et al., 2020, 2022) and retrain with combined data and the best auxiliary language (**DA-fs** and **DA-msdm**). The domain adaptation (**DA**) models are detailed below.

4.5.1. Adversarial Feature Separation

We study two methods for low-resource domain adaptation in the adversarial feature separation scheme. As illustrated in Figure 2b and 2c, instead of a single common encoder, a separate encoder is used for each domain plus a shared encoder for all domains. A domain classifier that receives the

Domain	Bikel		Benepar				
	0-shot	combined	0-shot	fine-tune	combined	DA-fs	DA-msdm
CRM	37.32	43.34-58.83-62.91	50.69	**58.89**-**71.97**-74.39	57.84-71.64-74.76	58.36-71.51-**75.45**	49.29-71.50-75.31
Tafel	39.12	45.14-53.25-56.18	65.35	**67.02**-71.97-73.64	66.50-**72.24**-74.16	66.81-72.11-**74.73**	59.27-71.27-74.09
Trappen	44.00	47.63-55.82-59.10	65.91	68.37-**76.87**-79.09	68.35-76.48-78.70	**68.52**-76.08-**79.55**	60.41-75.81-79.06

Table 4: Parsing F1 on target domain test sets. Models are trained with 10-100-200 examples from *every* target domain (except for zero-shot; plus *Etstoel* for data-combined and domain adaptation models). Averages over 10-fold cross-validation.

output of the shared encoder is trained in an adversarial manner with *gradient reversal* to encourage the shared encoder to learn knowledge not specific to any particular domain (Sato et al., 2017; Ganin and Lempitsky, 2015; Goodfellow et al., 2014).

Class-balanced Batching. Our target domain training data are limited in size, especially compared to that of the auxiliary language. To ensure that the stability of adversarial training, for all the **DA** models, we use separate class-balanced batching for the domain classifier so that each batch contains an equal amount of input of from each domain.

Orthogonality Constraints. Following Bousmalis et al. (2016); Li et al. (2020, 2022), we apply *orthogonality constraints* to encourage each private encoder to learn different representations than the shared one. The loss of orthogonality constraints is defined as follows:

$$\mathcal{L}_{\text{ort}} = \frac{1}{N} \sum_{d \in \mathcal{D}} \left\| \left(\mathbf{H}_c\right)^\top \mathbf{H}_p^{(d)} \right\|_F^2 \tag{1}$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm, $\mathcal{D}$ is the set of N domains, $\mathbf{H}_c$ is the shared representations, and $\mathbf{H}_p^{(d)}$ is the domain d private representations. The orthogonality constraints are implemented for all the **DA** models.

Fused Representations. In the shared-private scheme, the parsing module leverages representations from both the shared and private encoders. Sato et al. (2017) use gates to mix representations, but observe harm to performance when target domain data are scarce. This may be due to underfit of the target private encoder and of the target-specific gate parameters. Li et al. (2020) propose to alleviate this problem with *fused target-domain word embeddings*, which uses mixed private representations (from both target and source domain encoders) when the input comes from the low-resource target domain, and keeps the raw private representations if the input comes from the high-resource source domain. The rational behind this is to leverage transferable knowledge from a better optimized encoder of a similar domain trained with sufficient data. However, the ratio of this mixture is treated as a hyperparameter, whose choice can be crucial to adaptation performance.

In light of this, we propose to fuse the outputs of different encoders using a set of learnable coefficients (cf. Figure 2b):

$$\mathbf{H}^{(d')} = \alpha_c^{(d')}\, \mathbf{H}_c + \sum_{d \in \mathcal{D}} \alpha_p^{(d',d)}\, \mathbf{H}_p^{(d)} \tag{2}$$

where $\mathbf{H}^{(d')}$ denotes the fused representations for domain $d' \in \mathcal{D}$ fed into the task networks, $\alpha_c^{(d')}$ and $\alpha_p^{(d',d)}$ are domain-specific coefficients for the shared and private representations respectively, and $\alpha_c^{(d')} + \sum_{d \in \mathcal{D}} \alpha_p^{(d',d)} = 1$. The initial values of these coefficients are set proportionally to data sizes.[13] We implement this feature in the **DA-fs** model.

Multi-source Dynamic Matching. Li et al. (2022) tackle the low-resource private encoder underfit problem from a similar angle. They use a *dynamic matching network* to encourage the target encoder to learn from useful source features. We make simple extension to this network in order to adapt to our multi-domain and multi-source setting.[14] The modified matching loss is defined as follows:

$$\begin{aligned} \mathcal{L}_{\text{mat}} &= \frac{1}{|T||S|} \sum_{i \in T} \sum_{j \in S} \mathcal{L}_{\text{mat}}^{(i,j)}, \\ \mathcal{L}_{\text{mat}}^{(i,j)} &= \frac{1}{KD} \sum_{n,m} W_{i,j}^{n,m} \sum_{d=1}^{D} Q_{i,j,d}^{n,m} \left(f_\theta(t_j^m) - s_i^n\right)_d^2 \end{aligned} \tag{3}$$

where S and T denote the sets of source (teacher) and target (student) domain encoders, respectively. $i \in T$ and $j \in S$ index a target and a source encoder. $W_{i,j}^{n,m}$ and $Q_{i,j,d}^{n,m}$ are layer- and element-level matching weights. $f_\theta(\cdot)$ is a linear transformation. s_i^n and t_j^m denote outputs of the n-th and m-th layers of encoders E_i and E_j. D is the encoders' output dimension, and K the total number of layer pairs. The multi-source domain matching loss is implemented by the **DA-msdm** model.

[13] In practice, we initialize these coefficients based on domain sizes, ensuring that low-resource domains can also rely on shared features while data-rich ones capture more private signals. For domain i, we compute a normalized ratio $r_i = (N_i - \min(N))/(\max(N) - \min(N))$. The coefficients are then set to r_i for the domain-specific representations and $1 - r_i$ for the shared representations, with a fixed bias of -2.0 applied to all other domains.

[14] We refer the reader to Li et al. (2022) for detailed descriptions of the domain matching network design.

4.5.2. Results

Table 4 summarizes the parsing F1 of different models on the three target domain test sets (CRM, Tafel and Trappen), where the Benepar-fine-tune model is fine-tuned on 10, 100 or 200 examples from *every* target domain (thus 30, 300 or 600 examples in total), and plus *Etstoel* for data-combined and domain adaptation models. We find that *Benepar* strongly outperforms *Bikel* parser even in zero-shot conditions. The *Benepar* zero-shot model even outperforms data-combined *Bikel* with 200 examples each when tested on *Tafel* and *Trappen*, or that with 10 examples each when tested on *CRM*. Fine-tuning and data-combined retraining with sentences from the target domain yield comparable results, and the parser begins to show improvement with as few as 10 examples each and surpasses 70 F1 after 100 examples each.

We further examine feature separation techniques for domain adaptation and find that, with around 200 examples each per domain, cross-domain performance exceeds 74.7 F1 across all three new domains, while smaller datasets yield little or no improvement. Having only 10 examples from every target domain is detrimental to paring performance in the case of the model with multi-source dynamic matching (DA-msdm). This is likely because of the lack of training examples in the target domains causing training instabilities of the matching network.

5. Conclusion

We have adapted a transformer-based constituency parser to Middle Dutch and have shown that jointly training with temporally, geographically and typologically closer auxiliary languages yields larger improvements. For new domains, fine-tuning offers a faster alternative to data-combined retraining with similar effectiveness: small gains appear with about 10 examples, and clear improvements begin around 100. Feature separation also helps cross-domain performance but only when at least 200 sentences are available.

Future work includes exploiting unlabeled Middle Dutch data through continued pretraining (cf. Li et al., 2020, 2022; see also Gururangan et al., 2020; Grobol et al., 2022) or semi-supervised learning (cf. Rotman and Reichart, 2019) with auxiliary tasks, deepening exploration of domain adaptation with ongoing development of parsed and unparsed texts, and exploring multilingual objectives for historical languages.

6. Acknowledgements

We thank three anonymous reviewers, Pascal Amsili and Timothée Bernard for their valuable comments. We are also very grateful to Alexandra Simonenko for sharing the manually corrected parsed Middle Dutch data with us. This research has received funding from the European Union (ERC, CAUSALITY, grant number 101042427) as well as the laboratory LLF of Université Paris Cité.

7. Bibliographical References

Þórunn Arnardóttir and Anton Karl Ingason. 2020. A Neural Parsing Pipeline for Icelandic Using the Berkeley Neural Parser. In *Proceedings of CLARIN 2020*.

Emanuele Bastianelli, Andrea Vanzo, and Oliver Lemon. 2020. Encoding Syntactic Constituency Paths for Frame-Semantic Parsing with Graph Convolutional Networks.

Daniel M. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, pages 178–182, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Daniel M. Bikel. 2004. Intricacies of Collins' Parsing Model. *Computational Linguistics*, 30(4):479–511.

Leonard Bloomfield. 1933. *Language*. Holt, Rinehart & Winston, New York.

Hannah Booth, Anne Breitbarth, Aaron Ecay, and Melissa Farasyn. 2020. A Penn-style Treebank of Middle Low German. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 766–775, Marseille, France. European Language Resources Association.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a german corpus. *Research on language and computation*, 2(4):597–620.

Eugene Charniak. 1997. Statistical Techniques for Natural Language Parsing. *AI Magazine*, 18(4):33–43.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Do Kook Choe and Eugene Charniak. 2016. Parsing as Language Modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, Texas. Association for Computational Linguistics.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague.

Maximin Coavoux and Benoît Crabbé. 2017. Multilingual Lexicalized Constituency Parsing with Word-Level Auxiliary Tasks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 331–336, Valencia, Spain. Association for Computational Linguistics.

Michael Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191, Santa Cruz, California, USA. Association for Computational Linguistics.

Michael Collins. 1997. Three Generative, Lexicalised Models for Statistical Parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain. Association for Computational Linguistics.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Benoit Crabbé. 2015. Multilingual discriminative lexicalized phrase structure parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1847–1856, Lisbon, Portugal. Association for Computational Linguistics.

James Cross and Liang Huang. 2016. Span-Based Constituency Parsing with a Structure-Label System and Provably Optimal Dynamic Oracles. In *EMNLP 2016*. arXiv.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Melissa Farasyn, Anne-Sophie Ghyselen, Jacques Van Keymeulen, and Anne Breitbarth. 2022. Challenges in tagging and parsing spoken dialects of Dutch. *Journal of Historical Syntax*, 6(4-11):1–36.

Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. Cross-domain generalization of neural constituency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 323–330, Florence, Italy. Association for Computational Linguistics.

Yao Fu, Chuanqi Tan, Mosha Chen, Songfang Huang, and Fei Huang. 2021. Nested Named Entity Recognition with Partially-Observed TreeCRFs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12839–12847.

David Gaddy, Mitchell Stern, and Dan Klein. 2018. What's Going On in Neural Constituency Parsers? An Analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 999–1010, New Orleans, Louisiana. Association for Computational Linguistics.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Loïc Grobol and Benoit Crabbé. 2021. Analyse en dépendances du français avec des plongements

contextualisés (French dependency parsing with contextualized embeddings). In *Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 106–114, Lille, France.

Loïc Grobol, Mathilde Regnault, Pedro Ortiz Suarez, Benoît Sagot, Laurent Romary, and Benoît Crabbé. 2022. BERTrade: Using Contextual Embeddings to Parse Old French. In *Proceedings of the 13th Language Resources and Evaluation Conference*, Marseille, France. European Language Resources Association.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised Domain Adaptation of Contextualized Embeddings for Sequence Labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

Matthias Hüning and Ulrike Vogl. 2009. Middle Dutch – A short introduction. In André Bouwman and Bart Besamusca, editors, *Of Reynaert the Fox: Text and Facing Translation of the Middle Dutch Beast Epic Van Den Vos Reynaerde*, pages 257–272. Amsterdam University Press, Amsterdam.

Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. 2019. Learning what and where to transfer.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1199, Melbourne, Australia. Association for Computational Linguistics.

Colette Marie-Christine Etienne Van Kerckvoorde. 1993. *An Introduction to Middle Dutch*. Walter de Gruyter.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2016. Frustratingly easy neural domain adaptation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 387–396, Osaka, Japan. The COLING 2016 Organizing Committee.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual Constituency Parsing with Self-Attention and Pre-Training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. arXiv.

Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. arXiv.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Springer International Publishing, Cham.

Seth Kulick, Neville Ryant, and Beatrice Santorini. 2022. Penn-Helsinki Parsed Corpus of Early Modern English: First Parsing Results and Analysis. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 578–593, Seattle, United States. Association for Computational Linguistics.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Ying Li, Shuaike Li, and Min Zhang. 2022. Semi-supervised domain adaptation for dependency parsing with dynamic matching network. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1035–1045, Dublin, Ireland. Association for Computational Linguistics.

Ying Li, Zhenghua Li, and Min Zhang. 2020. Semi-supervised domain adaptation for dependency parsing via improved contextualized word representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3806–3817, Barcelona, Spain (Online). International Committee on Computational Linguistics.

KyungTae Lim, Jay Yoon Lee, Jaime Carbonell, and Thierry Poibeau. 2020. Semi-Supervised Learning on Meta Structure: Multi-Task Tagging and Parsing in Low-Resource Scenarios. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8344–8351.

Gabriel Marzinotto, Géraldine Damnati, Frédéric Béchet, and Benoît Favre. 2019. Robust semantic parsing with adversarial learning for domain generalization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 166–173, Minneapolis, Minnesota. Association for Computational Linguistics.

Ercong Nie, Helmut Schmid, and Hinrich Schütze. 2023. Cross-Lingual Constituency Parsing for Middle High German: A Delexicalized Approach. In *Proceedings of the Ancient Language Processing Workshop*, pages 68–79, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Beth Randall. 2010. CorpusSearch 2.

Eiríkur Rögnvaldsson, Anton Karl Ingason, Einar Freyr Sigurðsson, and Joel Wallenberg. 2012. The Icelandic parsed historical corpus (IcePaHC). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1977–1984, Istanbul, Turkey. European Language Resources Association (ELRA).

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713.

Sebastian Ruder. 2019. *Neural transfer learning for natural language processing*. Ph.D. thesis, NUI Galway.

Beatrice Santorini. 2022. Annotation manual for the Penn Parsed Corpora of Historical English.

Christopher Sapp, Daniel Dakota, and Elliott Evans. 2023. Parsing Early New High German: Benefits and limitations of cross-dialectal training. In *Proceedings of the 21st International Workshop on Treebanks and Linguistic Theories (TLT, GURT/SyntaxFest 2023)*, pages 54–66, Washington, D.C. Association for Computational Linguistics.

Christopher D. Sapp, Elliott Evans, Rex Sprouse, and Daniel Dakota. 2024. Introducing a Parsed Corpus of Historical High German. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9224–9233, Torino, Italia. ELRA and ICCL.

Motoki Sato, Hitoshi Manabe, Hiroshi Noji, and Yuji Matsumoto. 2017. Adversarial training for cross-domain Universal Dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 71–79, Vancouver, Canada. Association for Computational Linguistics.

Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. 2014. Introducing the SPMRL 2014 shared task on parsing morphologically-rich languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 103–109, Dublin, Ireland. Dublin City University.

Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. 2013. Large Scale Syntactic Annotation of Written Dutch: Lassy. In Peter Spyns and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch: Results by the STEVIN Programme*, pages 147–164. Springer, Berlin, Heidelberg.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages.

Bailin Wang and Wei Lu. 2018. Neural Segmental Hypergraphs for Overlapping Mention Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214, Brussels, Belgium. Association for Computational Linguistics.

Yufei Wang, Mark Johnson, Stephen Wan, Yifang Sun, and Wei Wang. 2019. How to Best Use Syntax in Semantic Role Labelling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5338–5343, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in*

Natural Language Processing: System Demonstrations, pages 38–45, Online. Association for Computational Linguistics.

Sen Yang, Leyang Cui, Ruoxi Ning, Di Wu, and Yue Zhang. 2022. Challenges to open-domain constituency parsing. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 112–127, Dublin, Ireland. Association for Computational Linguistics.

Songlin Yang and Kewei Tu. 2022. Bottom-Up Constituency Parsing and Nested Named Entity Recognition with Pointer Networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2403–2416, Dublin, Ireland. Association for Computational Linguistics.

Wenwen Zhang. 2022. Neural Dependency Parsing of Low-resource Languages: A Case Study on Marathi. Master's thesis, Uppsala University.

Junru Zhou and Hai Zhao. 2019. Head-Driven Phrase Structure Grammar Parsing on Penn Treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy. Association for Computational Linguistics.

8. Language Resource References

Bies, Ann and Mott, Justin and Warner, Colin. 2015. *Penn Treebank Revised: English News Text Treebank LDC2015T13*.

Galves, Charlotte and Faria, Pablo. 2017. *Tycho Brahe Parsed Corpus of Historical Portuguese*.

Kroch, Anthony and Santorini, Beatrice. 2010. *Penn Supplement to MCVF*.

Kroch, Anthony and Santorini, Beatrice and Delfs, Lauren. 2004. *Penn-Helsinki Parsed Corpus of Early Modern English (PPCEME)*.

Kroch, Anthony and Santorini, Beatrice and Diertani, C.E.A. 2016. *The Penn Parsed Corpus of Modern British English, 2nd Edition (PPCMBE2)*.

Kroch, Anthony and Taylor, Ann. 2000. *Penn-Helsinki Parsed Corpus of Middle English, Second Edition (PPCME2)*.

Martineau, France and Santorini, Beatrice. 2010. *Modéliser le changement: Les voies de français (MCVF) and Penn-BFM Parsed Corpus of Historical French (PPCHF)*. Original-date: 2021-05-31T01:44:53Z.

Simonenko, Alexandra and Liang, Yiming. *Penn-style Treebank of Middle Dutch*.

Taylor, Ann and Warner, Anthony and Pintzuk, Susan and Beths, Frank. 2003. *The York-TorontoHelsinki Parsed Corpus of Old English Prose (YCOE)*. PID https://penn-historical-corpora.uni-mannheim.de/ycoe/YCOEHomepage.html.

van Kemenade, Ans and Postma, Gertjan. *Etstoel-Drenthe corpus, INPOLDER project*.

van Reenen, Pieter and Mulder, Maaike. *Het Corpus Van Reenen-Mulder van 14e-eeuwse Middelnederlandse oorkonden*.

Xue, Nianwen and Zhang, Xiuhong and Jiang, Zixin and Palmer, Martha and Xia, Fei and Chiou, Fu-Dong and Chang, Meiyu. 2013. *Chinese Treebank 8.0*. Linguistic Data Consortium.

A. Appendix

A.1. Data preprocessing and postprocessing

Preprocessing: All annotated data, including the auxiliary language corpora, undergo the following preprocessing steps: First, sentences containing more than 100 words were removed, as this exceeds the maximal sequence length supported by Benepar. Second, all empty categories (surrounded by * or marked by 0, e.g, indications of null subjects), as well as co-reference indices on trace (marked as _ or = accompanied by a coreferent number to the syntactic/PoS tag) are removed, since they are manually added after automatic parsing. Third, sentences which contain foreign words only (headed by FW) are removed. Fourth, non-linguistic nodes, such as <CODE>, <META> and <ID>, are also removed.

Postprocessing: Kulick et al. (2022) use a modified evalb (Seddah et al., 2014) that does not delete punctuations to avoid mismatch for models that predict PoS tags. We instead use the original evalb as *Benepar* but perform the following postprocessing for parser outputs: for each mismatch of punctuation/non-punctuation tags, replace the parser predicted tag with a special label `*_REPL_*`. This modification does not affect parsing scores, and PoS tagging accuracy is evaluated with punctuations excluded.

A.2. Results without function tags

Table 5 shows results without function tags on different datasets. We compare the parsing and PoS tagging performance of the best models for the *Bikel*

Test Dataset	Model	Parsing F1	Parsing Δ	PoS acc.	PoS Δ
Etstoel	Bikel	77.83 (1.01)	0.00	92.21 (0.29)	0.00
	Benepar (Etstoel + ENHG)	89.54 (1.05)	+11.71	91.53 (0.39)	-0.68
CRM	Bikel (all)	67.12 (0.79)	0.00	93.08 (0.28)	0.00
	Benepar-fs (all + ENHG)	77.96 (0.45)	+10.84	90.70 (0.34)	-2.38
Tafel	Bikel (all)	64.39 (0.91)	0.00	87.35 (0.35)	0.00
	Benepar-fs (all + ENHG)	79.81 (0.61)	+15.42	83.74 (0.40)	-3.61
Trappen	Bikel (all)	67.43 (0.93)	0.00	88.36 (0.72)	0.00
	Benepar-fs (all + ENHG)	84.00 (0.40)	+16.57	88.25 (0.40)	-0.11

Table 5: Parsing F1 and PoS tagging accuracy on test sets, evaluated by evalb (ignoring function tags), averaged across 10-fold cross-validation, and the differences Δs with Bikel (or Bikel-0-shot) parser. *all* stands for training with Etstoel + 200 exp. each from CRM, Tafel and Trappen. Standard deviations are shown in parentheses.

and *Benepar* parsers. For each parser, we choose the best performing (parsing F1) model on each of the test dataset from all experimental setups : with/without PoS tag prediction, with/without target domain data, with/without auxiliary languages and with which auxiliary language, etc.

A.3. Hyperparameters

This appendix provides an overview of the hyperparameters used for training our Benepar models across all experimental phases. Table 6 details the specific configurations for the model architecture, optimization strategy, and various regularization and multi-task learning parameters. These settings were consistently applied unless explicitly stated otherwise in the main text.

Hyperparameter	Value
Architecture	
Pretrained language model	`dbmdz/bert-base-historic-dutch-cased`
Number of self-attention layers	2
Number of attention heads	8
Hidden size	1024
Feed-forward size	2048
Tag projection size	256
Label projection size	256
Optimization	
Base learning rate	1.32e-4
Learning rate warmup steps	160
Step decay factor	0.5
Step decay patience	5 checks
Regularization & Multi-task	
Attention dropout	0.2
Residual dropout	0.2
ReLU dropout	0.1
PoS tagging loss weight	4.66
Domain orthogonality constraint weight (λ_{ort})	0.01
Gradient reversal lambda (λ_{GRL})	0.5

Table 6: Hyperparameters used for training our Benepar models.