Proceedings of the Society for Computation in Linguistics

Volume 6 Article 21

6-1-2023

Parsing "Early English Books Online" for Linguistic Search

Seth Kulick
University of Pennsylvania, skulick@ldc.upenn.edu

Neville Ryant
University of Pennsylvania, nryant@ldc.upenn.edu

Beatrice Santorini
University of Pennsylvania, beatrice@sas.upenn.edu

Parsing "Early English Books Online" for Linguistic Search

Seth Kulick and **Neville Ryant**
Linguistic Data Consortium
University of Pennsylvania
{skulick,nryant}@ldc.upenn.edu

Beatrice Santorini
Department of Linguistics
University of Pennsylvania
beatrice@sas.upenn.edu

Abstract

This work addresses the question of how to evaluate a state-of-the-art parser on Early English Books Online (EEBO), a 1.5-billion-word collection of unannotated text, for utility in linguistic research. Earlier work has trained and evaluated a parser on the 1.7-million-word Penn-Helsinki Parsed Corpus of Early Modern English (PPCEME) and defined a query-based evaluation to score the retrieval of 6 specific sentence types of interest. However, significant differences between EEBO and the manually-annotated PPCEME make it inappropriate to assume that these results will generalize to EEBO. Fortunately, an overlap of source material in PPCEME and EEBO allows us to establish a token alignment between them and to score the POS-tagging on EEBO. We use this alignment together with a more principled version of the query-based evaluation to score the recovery of sentence types on this subset of EEBO, thus allowing us to estimate the increase in error rate on EEBO compared to PPCEME. The increase is largely due to differences in sentence segmentation between the two corpora, pointing the way to further improvements.

1 Introduction

The Penn-Helsinki Parsed Corpus of Early Modern English (PPCEME) (Kroch et al., 2004) consists of over 1.7 million tokens of text from 1500 to 1712, manually annotated for phrase structure. It belongs to a family of treebanks of historical English (Taylor et al., 2003, 2006; Kroch, 2020) and other languages (Wallenberg et al., 2011; Galves et al., 2017; Martineau et al., 2021; Kroch and Santorini, 2021) with a shared annotation philosophy and similar guidelines across languages, which form the basis for reproducible studies of syntactic change (Kroch et al., 2000; Ecay, 2015; Wallenberg, 2016; Galves, 2020; Wallenberg et al., 2021).

While all of these corpora are large for manually annotated corpora, even relatively common phenomena still occur too rarely to support reliable statistical models of how they change over time. We therefore wish to parse and search the much larger corpora that are becoming publicly available, such as the Early English Books Online (EEBO) corpus (Text Creation Partnership, 2019) with its 1.5 billion words of text from 1475 to 1700. However, EEBO's potential as a resource for linguistic research remains unrealized because it is not linguistically annotated and its size renders manual annotation infeasible. Our goal is therefore to parse EEBO automatically.

Kulick et al. (2022a) took first steps in this direction by training and evaluating a constituency parser using the gold trees from PPCEME. This parser achieved a cross-validated evalb score (Sekine and Collins, 2008) of 90.53%, suggesting the feasibility of the larger project of parsing EEBO. In a follow-up paper, Kulick et al. (2022b) directly evaluated the utility of the recovered parse trees for the retrieval of sentence types necessary to study a particular linguistic change in the history of English. Utilizing a novel alternative to evalb, termed "query-based evaluation", the parser was evaluated by specifically scoring the retrieval of these sentence types. The resulting precision scores were promising, warranting further work.

However, Kulick et al. (2022a,b) obtained their results for PPCEME, not EEBO. While both corpora consist of Early Modern English texts, they harbor significant differences, making it inappropriate to assume that results obtained for PPCEME generalize to EEBO.

In this work, we therefore extend the parser evaluation to EEBO itself. An apparently intractable difficulty is the absence of gold parse trees for EEBO. Fortunately, there is some overlap between PPCEME and EEBO; specifically, about 42% of PPCEME consists of source texts also present in EEBO, though possibly based on variant editions that differ in spelling and punctuation. Using an improved language model, we train a parser

Proceedings of the Society for Computation in Linguistics (SCiL) 2023, pages 222-242.
Amherst, Massachusetts, June 15-17, 2023

Missing words or punctuation:

```
haue alway resysted hym , and
haue       resisted       and
```

Tokenization differences:

```
In whom , nat withstandyng   ,
In whom   not with standynge ,
```

Bullet (illegible) character in EEBO:

```
I will not let , openlie to
I will not l•• , openlie to
```

Figure 1: Examples of mismatches in PPCEME (top) and EEBO (bottom) source texts.

on the non-overlap section of PPCEME and then parse both the PPCEME and EEBO versions of the overlap. We create a token alignment between the two overlap versions, which allows us to evaluate the parsed EEBO overlap for part-of-speech (POS) accuracy. We also improve the mechanics of the query-based evaluation from Kulick et al. (2022b) and use that, together with the alignment, to evaluate the parser's performance on the EEBO overlap text.

The rest of the paper is structured as follows. Section 2 discusses some important features of the overlap and the alignment between the two versions. Section 3 presents the parser model, along with results on PPCEME based on evalb, which we include to show improvements due to the new language model. Section 4 discusses the parsing of the EEBO overlap and the POS evaluation. Section 5 describes the queries and the new alignment-mediated scoring method, and Section 6 presents the results. Section 7 summarizes with lessons learned and suggestions for future work.

2 PPCEME-EEBO Overlap

2.1 Overview

PPCEME consists of material from 232 source texts, 42 of which have EEBO counterparts (see Appendix A for details). It might be thought that PPCEME should form a proper subset of EEBO, but this is not the case as while EEBO consists of all English-language material printed before 1700, many texts in PPCEME - notably private letters and editions of minor plays - did not appear in print until after 1700.

Figure 1 illustrates the main differences between the PPCEME and EEBO versions of the overlap.

source	# sents	# tokens	tokens/sent
PPCEME	39,400	805,475	20.44
EEBO	28,378	813,947	28.68

Table 1: Sentence counts and token counts for the PPCEME and EEBO versions of the overlap material.

The first example shows how one version may have tokens that are entirely missing from the other ("alway", "hym", ","). The second shows a typical case of a whitespace tokenization difference - "withstandyng" vs. "with standynge". Both examples also show differences in spelling and punctuation. The third example is a very specific type of spelling difference. Illegible characters in the source material are represented in EEBO by a bullet character. Here, "l••" has two illegible characters, corresponding to "let" in PPCEME.[1]

A further significant difference concerns sentence segmentation. Sentence segmentation in PPCEME was performed manually in accordance with annotation guidelines based on linguistic considerations. This is not the case for EEBO. The lack of standardized punctuation conventions for Early Modern English makes it non-trivial to segment sentences according to modern punctuation conventions, let alone according to PPCEME's guidelines.

Figure 2 gives two examples, each of which illustrates a string of text divided into separate sentences (and therefore trees) in PPCEME. The corresponding nearly identical text in EEBO is not so divided. As is evident, PPCEME sometimes splits on commas (e.g., the comma after *rest*) and colons (e.g., the colon after *Gold*). In contrast, after word tokenization, we split sentences in EEBO automatically on question mark, exclamation mark, and period.[2] (The reason that EEBO has no sentence break after *quills* in the first example is that it has a comma for PPCEME's period.) We refrain from splitting on commas because doing so would massively overgenerate sentence fragments.

As a result, sentences in EEBO tend to be longer than in PPCEME. Table 1 shows the number of tokens, sentences, and mean sentence length for the overlap material. The number of tokens is roughly the same, but EEBO has fewer sentences and so higher mean sentence length. Appendix A breaks

[1] PPCEME contains no bullet characters because any illegible characters were manually resolved in the process of either data entry or annotation.

[2] We do not split on periods in common abbreviations, Roman numerals of the era, and the like.

```
|| His Dame comming home and hearing that her man was gone to bed , tooke that night
but small rest , || and early in the morning hearing him vp at his worke merrily
singing , shee by and by arose , || and in seemely sort attyring her selfe , she
came into the worke-shop , || and sat her downe to make quills .  || Quoth Iohn ,
Good morow Dame , || how do you to day ?  ||

|| No , Nan Winchcombe , I will call her name , plaine Nan :  || what , I was a
woman , when she was sir-reuerence a paltry girle , though now shée goes in her Hood
and Chaine of Gold :  || what care I for her ?  ||
```

Figure 2: Sentence segmentation in PPCEME (indicated by vertical bars). Each of the two corresponding examples in EEBO is treated as a single long sentence.

source	# aligned	# unaligned	%
PPCEME	796,704	8,771	98.9
EEBO		17,243	97.9

Table 2: Number of aligned and unaligned tokens, and percentage of aligned, in PPCEME and EEBO.

down Table 1 by source text, revealing significant differences for some files, and also compares the sentence lengths of the PPCEME and EEBO version of the overlap to that of PPCEME and EEBO as a whole.

2.2 Alignment

The rest of the work relies on having a token-to-token (words and punctuation) alignment between the PPCEME and EEBO versions of the overlap. Both versions required some preparatory work before running our token-alignment algorithm.

For EEBO, we followed the same procedure as detailed in Kulick et al. (2022a,b) in connection with using EEBO for language model training, with sentence segmentation as just described.

In PPCEME, the 42 source texts are generally represented by non-exhaustive samples. Moreover, because of how the corpus was constructed over time, these samples do not always appear in the order in which they appear in the edition (for instance, parts of a play's fourth act might be interleaved with the first act). We therefore prepared a normalized version of the material with the sentences in order, which was then processed further as described for PPCEME in Kulick et al. (2022a,b).

We then aligned each of the 42 texts at the token level with our implementation of the Smith-Waterman algorithm (Smith et al., 1981), using a similarity measure based on Levenshtein distance (Levenshtein et al., 1966). To help anchor the alignment, we lowered the substitution costs for the bullet character (to 0.1) and the relatively common *u*/*v* alternation (to 0.2). We also forced the similarity to equal 1 for consistent cases of alternation between PPCEME and EEBO (e.g. *&c*/*etc.*, *&*/*and*, and *the*/*ye*).

Table 2 summarizes the completeness of the alignment, showing number of aligned and unaligned tokens in PPCEME and EEBO. For example, *alway*, *hym* and the comma in the first example in Figure 1 are unaligned PPCEME tokens. In the second example, *withstandyng* and *standynge* are aligned, so *with* is an unaligned token in EEBO. For additional alignment details, including per-text statistics, consult Appendix B.

3 Model and Evaluation

3.1 Parser Architecture

We use the same parser architecture as Kulick et al. (2022a,b), but with an improved language model. The parser model is based on Kitaev et al. (2019), which represents a constituency tree T as a set of labeled spans (i, j, l), where i and j are a span's beginning and ending positions and l is its label. Each tree is assigned a score $s(T)$, which is decomposed as a sum of per-span scores:

$$s(T) = \sum_{(i,j,l) \in T} s(i, j, l) \qquad (1)$$

The per-span scores $s(i, j, l)$ themselves are assigned using a neural network that takes a sequence of embeddings as input, processes these embeddings using a transformer-based encoder (Vaswani et al., 2017), and produces a span score from an MLP classifier (Stern et al., 2017). The highest-scoring valid tree is then found using a variant of the CKY algorithm. POS tags are recovered using a separate classifier operating on top of the encoder output, which is jointly optimized with the span

section	parser	POS
dev	92.08 (1.6)	98.23 (0.7)
test	91.77 (0.6)	98.37 (0.3)

Table 3: Cross-validation parser and POS results. Each result is the mean for the section (dev or test) over the 8 splits (standard deviation in parentheses). All scores are expressed as percentages.

classifier. For more details, see Kitaev and Klein (2018).

Our implementation is based on version 0.2.0 of the Berkeley Neural Parser[3], with some modifications for using the PPCEME and EEBO data as input.[4] While the earlier work used ELMo embeddings pretrained from scratch on EEBO, here we use RoBERTa embeddings (Liu et al., 2019) with continued pre-training for two epochs on EEBO starting from *roberta-base*.[5] For more details on training and hyperparameters, see Appendix C.

3.2 Cross-Validation Results on PPCEME

We use the same 8-fold split of PPCEME as in Kulick et al. (2022a,b), training each of the 8 models for 50 epochs and using the evalb score on the dev section as our criterion for saving the best model. Table 3 gives our parsing and POS results, combined over the 8 cross-validation splits, as scored by evalb (matching brackets for the parsing score and POS accuracy for the tagging score).[6]

The parser scores are all 1.2% higher (absolute) than the ELMo-based results reported in Kulick et al. (2022b), with the POS results also showing a slight increase (an average of 0.08). Kulick et al. (2022b) point out some differences in annotation style from the Penn Treebank (PTB) (e.g., lack of base NPs) that lead to lower parser scores here than if run on PTB. For details of the cross-validation splits, see Appendix D.

4 Parsing and POS Accuracy for Overlap

At this point, we have the token alignment between the PPCEME and EEBO overlap versions, and we have trained and evaluated on all of PPCEME with cross-validation, showing improved results over earlier work. Our next step is to train the parser in order to evaluate on the overlap versions.

We reserve the overlap for testing and partition the remaining non-overlap PPCEME material into training and dev sections, as set out in Table 4.

section	# files	# tokens	% of split
train	184	1,041,352	54.58
dev	6	60,960	3.20
overlap	42	805,475	42.22
total	232	1,907,787	100.00

Table 4: Split of PPCEME for evaluating on overlap.

	# tokens	parser	POS
PPCEME overlap			
all tokens	805,475	91.64	98.26
EEBO overlap			
aligned tokens	796,704	-	95.17
non-punc only	702,464	-	97.25
only w/ bullet	2,057	-	80.12

Table 5: Parser (evalb f1) and POS (accuracy) scores for PPCEME and EEBO versions of overlap.

4.1 Scoring the PPCEME overlap

Having trained the parser, we now evaluate it on the PPCEME version of the overlap. Since we have gold trees for this material, we can do so with evalb. The top part of Table 5 shows aggregate evalb and POS results. Appendix E gives a breakdown by text, including recall and precision.

The parser score of 91.64% is lower than the cross-validated results in Table 3. This is hardly surprising, since the parser is only being given 55% as much training data.

4.2 Scoring the EEBO overlap

For the EEBO version of the overlap, we have no corresponding gold trees, and so cannot evaluate with evalb.[7] However, we can - for the first time - evaluate POS accuracy on EEBO by taking advantage of the token alignment discussed in Section 2.2. 97.9% (796,704) of the tokens in EEBO are aligned to a corresponding token in PPCEME. For these tokens, we can take the gold tag in EEBO to be that of its PPCEME partner. EEBO tokens

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

[4] These modifications and other relevant software will be made available at https://github.com/skulick/emeparse.

[5] https://huggingface.co/roberta-base

[6] For reasons discussed in Kulick et al. (2022b), we use the modified evalb supplied with the Berkeley parser (Seddah et al., 2014), which does not remove words based on punctuation tags.

[7] But see the conclusion for a possible modification of evalb.

	# tokens				
tag	gold	EEBO	rec	prec	f1
N	93,720	92,513	96.82	95.57	96.19
P	91,175	91,190	98.89	98.91	98.90
,	57,992	71,966	76.91	95.44	85.18
D	62,701	62,440	99.49	99.08	99.29
PRO	52,368	52,204	99.34	99.03	99.19
CONJ	42,478	42,154	99.44	98.68	99.06
ADJ	35,769	35,480	95.93	95.16	95.54
NS	30,937	30,974	96.79	96.91	96.85
ADV	24,804	24,477	96.83	95.56	96.19
VB	22,724	22,718	97.39	97.37	97.38
.	36,415	22,274	88.70	54.26	67.33
NPR	19,277	20,210	88.36	92.64	90.45
PRO$	17,060	17,023	99.37	99.16	99.26
BEP	14,938	14,905	99.14	98.92	99.03
VAN	14,540	14,726	95.43	96.65	96.04
VBP	14,291	14,345	95.88	96.24	96.06
Q	14,044	13,998	98.75	98.43	98.59
MD	13,828	13,709	99.43	98.58	99.00
VBD	13,663	13,653	97.48	97.41	97.44
TO	10,890	10,858	99.54	99.25	99.39
total	796,704	796,704	95.17	95.17	95.17

Table 6: Breakdown by 20 most frequent tags for the 95.17% score in row "aligned tokens" of Table 5. Note that the *total* row includes all POS tags.

without an alignment partner are left out of the scoring.

The results are shown in row "aligned tokens" in Table 5. The score (95.17%) is lower than the corresponding score for the PPCEME overlap (98.26%). Table 6 breaks the score down by tag for the 20 most common tags. Appendix F presents more detailed results along two dimensions, breaking down the bottom (EEBO) part of Table 5 by overlap file and expanding Table 6 to include all tags.

4.3 Punctuation in EEBO

The third most common tag in Table 6, comma, and the 11th most common tag, period, have low scores of 85.18% and 67.33%, respectively. This is because they are often confused, which in turn follows from a combination of PPCEME's POS annotation style with the differences in sentence segmentation in PPCEME and EEBO discussed in Section 2.1. PPCEME tags all tree-final punctuation as period. For example, in the first two lines of Figure 2, the comma after *bed* is tagged as comma, while the one after *rest* is tagged as period. In contrast, the parser assigns comma to both in EEBO - a reasonable error since in the EEBO version, the second comma is not tree-final. Re-evaluating without these two tags (row "non-punc only" in Table 5) raises the accuracy to 97.25%.

Negative declarative sentences	
VERB-NOT-DECL	*They drank not the ale*
DO-NOT-DECL	*They did not drink the ale*
Negative imperatives	
VERB-NOT-IMP	*Drink not the ale*
DO-NOT-IMP	*Do not drink the ale*
Direct questions	
VERB-SBJ	*Drank they the ale?*
DO-SBJ	*Did they drink the ale?*

Table 7: Sentence types retrieved by query searches.

4.4 Tokens with Bullet Characters in EEBO

We were also curious about accuracy on tokens containing a bullet character. As the row "only w/ bullet" shows, the score for such tokens drops to 80.12%, although they are too rare to have a major effect on the overall score. The bullet character is completely missing from the training data. Augmenting that data to randomly include it would likely improve the score on these tokens.

5 Queries and Scoring

5.1 Query Types

Kulick et al. (2022b) focused on six sentence types, which are formulated as queries for tree structures in the CorpusSearch query language (Randall, 2010). We use the same queries here. Table 7 illustrates the three pairs of sentence types retrieved by the queries, along with our labels for them (see Appendix G for a full description of the sentence types). For each pair, the first sentence type is the variant dominant in 1500, and the second the variant dominant by 1700. The leading idea of the overall project is that large datasets like EEBO will eventually allow us to decide between competing conceptual models of the loss of the older variant - specifically, competition (Kroch, 1989; Zimmermann, 2017) versus drift (Karjus, 2020).

We run the queries over three sets of trees - *PPCEME-gold* (the gold trees from the release), *PPCEME-parsed* (the parsed trees of the PPCEME version of the overlap, using the parser trained with the split described in Section 4), and *EEBO-parsed* (the parsed trees of the EEBO version of the overlap, using the same parser). This allows us to ad-

dress the problem outlined in the introduction - determining the accuracy of the query-based retrieval on parsed EEBO text as compared to parsed PPCEME text - by comparing query hits on *EEBO-parsed* and *PPCEME-parsed*, respectively, to query hits on *PPCEME-gold*.

5.2 Query Scoring on PPCEME

For scoring the query retrieval on PPCEME, we can use the same approach as Kulick et al. (2022b) for scoring queries over the PPCEME cross-validation splits. Since we are comparing parsed to gold versions of the same text, the sentence segmentation and tokens are identical, and the comparison can therefore proceed on a tree basis. Each query hit is considered to have a location (tree #, index), where the tree number is the tree it occurs in, and the index is an arbitrary numbering of the number of hits within a tree (usually just 1). Since the trees are in alignment, the matches are those for which the query hits from the gold and parsed trees have the same location, and the recall/precision/f-measure scores follow.

5.3 The Need for a New Method

However, this approach does not extend to scoring *EEBO-parsed* vs. *PPCEME-gold*, since neither the sentence segmentation nor the tokens necessarily match up. Figure 3 illustrates the problem, using the last two segments from the first example in Figure 2. The left side shows two gold PPCEME trees, while the corresponding text on the right comes from one large EEBO tree, due to the different segmentation in EEBO.

The lower PPCEME tree shows a VERB-SBJ query hit covering *how do you to day ?*, and the EEBO tree fragment shows a VERB-SBJ query hit covering *Good morrow Dame , how doe you to day*. (The question mark is not part of the hit in the EEBO version, since there it is outside of the CP-QUE-MAT clause.)

While the text covered is different, both query hits correctly label the sentences they find as VERB-SBJ. But the PPCEME trees are #s 137 and 138 among the PPCEME trees, while the EEBO tree is #98 among the EEBO trees, and so comparison by tree number is not possible.

5.4 Alignment-Mediated Scoring

However, we also have the spans of the constituents from the query hits, as indicated by `<3272,3278>` for the PPCEME tree and `<3001,3009>` for the EEBO tree. These spans refer to the PPCEME or EEBO overlap section as a whole, not to the individual trees. We can use this span information, together with our word alignment, to carry out an *alignment-mediated scoring* (AMS), as follows:

(1) Given a list of m query hits from the gold trees, and n query hits from the parsed trees, we form a $m \times n$ array of scores for each pair of hits. The score for a pair of hits is computed by using the token alignment to convert the *EEBO-parsed* span to a span in *PPCEME-gold*, and then computing a simple span overlap, normalized for length. For example, in Figure 3, `<3001,3009>` in the EEBO tree maps (by the alignment) to `<3268,3277>` on PPCEME, and so the span overlap is computed between the PPCEME hit span `<3272,3278>` and the span mapped from EEBO, `<3268,3277>`. The span overlap score is 0.55, and this becomes the score for this pair of hits. Hits in PPCEME and EEBO that match exactly would have a score of 1.0, and ones with no tokens in common (again, after using the alignment to compare them) would have a score of 0.0.

(2) We treat this as a bipartite graph minimum weight matching problem, where the "weight" for a pair of trees is one minus the span overlap, computed as just described. In this way the "penalty" for the overall mapping is minimized. We filter the results to ensure that all hits have at least one token in common.

For consistency, we compare *PPCEME-parsed* to *PPCEME-gold* in the same way as *EEBO-parsed* to *PPCEME-gold* (that is, using AMS). This also demonstrates the validity of the algorithm, since the results for *PPCEME-parsed* vs. *PPCEME-gold* using AMS are identical to those using the method from Kulick et al. (2022b).[8] Further details of the algorithm are available in Appendix H.

6 AMS Results and Analysis

6.1 Results and Analysis for EEBO

Table 8 presents scores from the query-based evaluation for *PPCEME-parsed* and *EEBO-parsed*, using AMS to produce the latter results. Our goal was to estimate the effect on the query results of

[8] We thus also resolve a lingering doubt from the earlier work. The earlier scoring method allowed a hit in a parsed PPCEME tree to "match" a hit in a completely different part of the corresponding PPCEME gold tree. The current method rules out such spurious matches, since it relies on the actual spans, not just the trees in which they occur.

```
(IP-MAT
    (VBD Quoth)
    (NP-SBJ (NPR Iohn))
    (, ,)
    (QTP (INTJP (ADJ Good)
                (N morow))
         (NP-VOC (N Dame)))
    (. ,))

(CP-QUE-MAT *<3272,3278>|verb-sbj*
    (WADVP-1 (WADV how))
    (IP-SUB
        (ADVP *T*-1)
        (DOP do)
        (NP-SBJ (PRO you))
        (NP-TMP (N (N to)
                   (N day))))
    (. ?))
```

(a) Two gold PPCEME trees

```
[...]
(IP-PPL
    (VBD quoth)
    (NP-SBJ (NPR Iohn))
    (, ,)
    (CP-QUE-MAT *<3001,3009>|verb-sbj*
        (NP-VOC (ADJ Good)
                (N morrow)
                (NP-VOC (N Dame)))
        (, ,)
        (WADVP (WADV how))
        (IP-SUB
            (DOP doe)
            (NP-SBJ (PRO you))
            (NP-TMP (N (N to)
                       (N day)))))))
(. ?))
```

(b) Fragment of one parsed EEBO tree

Figure 3: Example of the query matching problem. (a) shows two gold PPCEME trees with a VERB-SBJ query hit in the lower tree, at span <3272,3278> covering *how do you to day ?*. (b) is a fragment of a larger parsed EEBO tree with a VERB-SBJ query hit at span <3001,3009> covering *Good morrow Dame , how doe you to day*.

	PPCEME-gold	*PPCEME-parsed*				*EEBO-parsed*			
query	# hits	# hits	recall	prec	f1	# hits	recall	prec	f1
Negative declarative sentences									
VERB-NOT-DECL	662	680	95.47	92.94	94.19	634	87.16	91.01	89.04
DO-NOT-DECL	329	318	95.44	98.74	97.06	304	89.97	97.37	93.52
Negative imperative sentences									
VERB-NOT-IMP	148	135	81.76	89.63	85.51	120	71.62	88.33	79.10
DO-NOT-IMP	31	26	80.65	96.15	87.72	25	77.42	96.00	85.71
Questions									
VERB-SBJ	302	266	79.47	90.23	84.51	228	68.54	90.79	78.11
DO-SBJ-ORD	306	282	90.20	97.87	93.88	253	80.72	97.63	88.37

Table 8: AMS results for *PPCEME-parsed* and *EEBO-parsed* versions of overlap, as compared to *PPCEME-gold* trees.

parsing on EEBO instead of PPCEME. This table provides the answer - the f1 scores generally decrease by about 4-6 points. (The score for DO-NOT-IMP, which is less frequent, decreases less.)

Comparing the recall and precision scores reveals that the decrease is largely due to decreases in recall. Precision stays relatively stable, while recall goes down by as much as 10 points (e.g. for VERB-SBJ, from 79.47% to 68.54%). This means that parser errors on EEBO are preventing the queries from finding the structure that is present in the gold PPCEME trees.

Examination of the parser errors suggests that longer sentence length is exacerbating a tendency of the parser (already noted in Kulick et al. (2022b)) to produce nonsensical flat structures with two subjects or two finite verbs (or both). For example, consider the second sentence in the example of sentence segmentation in Figure 2. The last segment *what care I for her ?* is a VERB-SBJ that is missed in the *EEBO-parsed* tree because the parse of the entire sentence *No , Nan Winchcombe ... for her ?* is such a nonsensical structure. Omitting details, the structure of the parse, with the two subjects and two finite verbs, is shown in Figure 4.

corpus	gold	parsed
PPCEME	4	233
EEBO	-	934

Table 9: Number of trees with two subjects, as one example of nonsensical parser error.

Parser error analysis can be complex and tedious (especially here, with the differences in sentence segmentation), but we can facilitate it by extending our use of query-based searches from finding

```
(CP-QUE-MAT
    (INTJ No)
    (NP-VOC (NPR Nan) (NPR Winchcombe))
    (, ,)
    (NP-SBJ (PRO I))
    (MD will)
    (VB call)
    (IP-SMC her name...Nan)
    (, :)
    (INTJP (WPRO what))
    (, ,)
    (NP-SBJ (PRO I))
    (BED was)
    ....
```

Figure 4: Incorrect flat parse on EEBO text.

structures of linguistic interest to finding structures that should never occur, such as clauses with two subjects. Table 9 shows the large increase of trees with such impossible structures in *EEBO-parsed*, although the number in *PPCEME-parsed* is already higher than desired.[9]

6.2 Cross-Validated Results on PPCEME

As pointed out in Kulick et al. (2022b), the parses need not be perfect for query-based search to be useful, since if an error rate can be estimated, it can be factored into the linguistic analysis. We have determined the increase in error rate when querying on EEBO rather than on PPCEME.

We are also interested in determining the error rate when querying on PPCEME. Kulick et al. (2022b) addressed this issue with the cross-validation query-based evaluation on PPCEME. However, that was using an older language model, and while here we presented improved evalb scores (Table 3), the improvements are not guaranteed to carry over to the query-based scores.

This other aspect is not our focus here, but it is part of the overall goal, and so we give some results in Appendix I, with updated cross-validated query results on the current version of the parser, along with some discussion of the impact of using less training data for the overlap split.

7 Conclusion

Exploiting the existence of an overlap between PPCEME and EEBO, we have succeeded in scoring POS tagging on EEBO and extending query-based evaluation to EEBO. Given these results, could the trees and POS-tags of a parsed EEBO be used with confidence? For those wishing to use the POS tags, we have shown that the POS tags can overall be expected to be of high accuracy (with some variation for individual tags, and excepting the punctuation issue discussed in Section 4.2). For the structure-based queries, we now have the needed estimate of the decrease in accuracy on EEBO compared to PPCEME. There are some obvious next steps to improve the parsing and query results on EEBO, and so lessen that decrease in accuracy.

The first priority is to address the problem of sentence segmentation in EEBO. We have shown in this paper why this is an important issue for parsing EEBO, and we can use the overlap to measure the effect more precisely. We will do so by using the token alignment to "fix" the sentence segmentation in the EEBO version of the overlap to be consistent with the sentence segmentation in the PPCEME version of the overlap, thus allowing us to directly measure the query accuracy on EEBO without the distorting effect of the segmentation differences and thus to estimate the latter effect.

Following this step, we see two possibilities for addressing the effects of the differences in sentence segmentation in PPCEME and EEBO. One approach modifies the training data, while leaving the EEBO segmentation as it is, by combining the PPCEME trees used for training when the text has a final comma in the text, thus approximating the EEBO segmentation. The other (preferred) approach would directly modify the EEBO segmentation by using the existing segmentation in PPCEME to train a segmenter for EEBO.

The are different directions to pursue after that point:

Improving the parser architecture. While Section 6 discussed the increase in nonsensical structures from PPCEME to EEBO, there were already too many (233) with PPCEME. It is possible that a parser model that moves away from the span-based approach of the Berkeley neural parser, using well-defined grammatical structures instead, might overcome this problem. In particular we plan to experiment with a Tree Adjoining Grammar (TAG) or related architecture (Kasai et al., 2018). This change of architecture would also allow for the recovery of the empty categories and co-indexing, which will be required as the range of linguistic inquiries expands, with the precise approach used to accomplish this depending on which architecture is chosen.

Another aspect of the parser architecture that

[9]The four occurrences in *PPCEME-gold* are annotation errors that have been corrected for the next release.

should be improved is the recovery of the function tags. As mentioned in Appendix C.2, currently we simply retain the function tags as part of atomic nonterminals for the training and parsing. While this approach works surprisingly well, it is potentially problematic for combinations of nonterminal/function tag that do not appear with frequency in the training data. One possible alternative is to integrate the function tag recovery in the current parser model analogously the POS tagging, as a separate classifier with a joint training loss.

Treebank representation. PPCEME is a phrase-structure treebank, with the associated linguistic queries reflecting that structure, and so it was natural for us to focus on phrase-structure parsing. However, it would be useful to represent PPCEME in a dependency format, so that a dependency parser could be used as well. While it might be possible to adapt one of the phrase-structure-to-dependency converters for use on PPCEME and the parsed EEBO, our preference would be for this to follow from the use of a TAG-like formalism, which is in a sense intermediate between a phrase-structure and dependency representation.

Application to other historical treebanks. As mentioned in the introduction, PPCEME is just one of a series of historical treebanks, which share annotation philosophy and guidelines. In addition to applying this work to those other treebanks, they would in turn serve as an extensive and varied testbed for evaluating the different parser models.

Query retrieval without parsing. An entirely different direction from the work described here, but with the same goal, is to use sentence embeddings derived from token embeddings, as in Arora et al. (2017), to identify the desired sentences directly, without using a parser at all. For example, it might be possible to find EEBO sentences "similar" to a given sentence, akin to an information retrieval system.

Additional types of annotation. The overlap and alignment to PPCEME can be used to evaluate the automatic annotation of other types of annotation on EEBO. For example, if PPCEME were annotated with lemmas for each token, then a lemmatizer on EEBO could be tested on the overlap section by using the existing alignment and treating the PPCEME lemmas for the aligned tokens as the gold lemmas. In addition, research in the last few years on improving word sense disambiguation (Bevilacqua et al., 2021) could be applied to the parsed EEBO, by using example sentences in the Oxford English Dictionary[10] to map each word instance to its sense usage.

Modified evalb. Finally, we stated in Section 4.2 that we cannot run evalb on parsed EEBO files in the absence of gold trees for EEBO. However, AMS opens up the possibility of doing just that. Since evalb scores matching brackets, it can be modified to match brackets using this approach instead of searching for identical spans. Such a modification could then even be adopted for material that already has matching text and sentence segmentation, allowing for a "fuzzy" evalb that can match brackets without an exact match, degrading the score if desired.

Acknowledgments

This work was supported by NSF grant BCS 13-046668, "Annotating and extracting detailed syntactic information from a 1.1-billion-word corpus." We thank the three anonymous reviewers for their insightful comments. We would also like to acknowledge here the tremendous debt, intellectual and in other ways, that this work owes to the late Prof. Anthony Kroch.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations*.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conference on Artificial Intelligence, Inc.

Aaron Ecay. 2015. *A multi-step analysis of the evolution of English do-support*. Ph.D. thesis, University of Pennsylvania.

Ryan Gabbard, Seth Kulick, and Mitchell Marcus. 2006. Fully parsing the Penn Treebank. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 184–191, New York, New York. Association for Computational Linguistics.

Charlotte Galves. 2020. Relaxed V-Second in Classical Portuguese. In Rebecca Woods and Sam Wolfe,

[10] https://www.oed.com

editors, *Rethinking Verb-Second*, pages 368–395. Oxford University Press.

Charlotte Galves, Aroldo Leal de Andrade, and Pablo Faria. 2017. Tycho Brahe Parsed Corpus of Historical Portuguese. http://www.tycho.iel.unicamp.br/~tycho/corpus/texts/psd.zip.

Andres Karjus. 2020. *Competition, selection and communicative need in language change: An investigation using corpora, computational modelling and experimentation*. Ph.D. thesis, University of Edinburgh.

Jungo Kasai, Robert Frank, Pauli Xu, William Merrill, and Owen Rambow. 2018. End-to-end graph-based TAG parsing with neural networks. 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 1181–1194, New Orleans, Louisiana. Association for Computational Linguistics.

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*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

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)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Anthony Kroch. 1989. Reflexes of grammar in patterns of language change. *Language Variation and Change*, 1(3):199–244.

Anthony Kroch. 2020. Penn Parsed Corpora of Historical English. LDC2020T16 Web Download. Philadelphia: Linguistic Data Consortium. Contains Penn-Helsinki Parsed Corpus of Middle English, second edition, Penn-Helsinki Parsed Corpus of Early Modern English, and Penn Parsed Corpus of Modern British English.

Anthony Kroch and Beatrice Santorini. 2021. Penn-BFM Parsed Corpus of Historical French, version 1.0. https://github.com/beatrice57/mcvf-plus-ppchf.

Anthony Kroch, Beatrice Santorini, and Lauren Delfs. 2004. Penn-Helsinki Parsed Corpus of Early Modern English (PPCEME). CD-ROM, first edition, release 3. http://www.ling.upenn.edu/ppche/ppche-release-2016/PPCEME-RELEASE-3.

Anthony Kroch, Ann Taylor, and Donald Ringe. 2000. The Middle English verb-second constraint: A case study in language contact and language change. In Susan Herring, Lene Schoessler, and Peter van Reenen, editors, *Textual parameters in older language*, pages 353–391. Benjamins.

Seth Kulick, Neville Ryant, and Beatrice Santorini. 2022a. 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, Washington. Association for Computational Linguistics.

Seth Kulick, Neville Ryant, and Beatrice Santorini. 2022b. Parsing Early Modern English for linguistic search. In *Proceedings of the Society for Computation in Linguistics 2022*, pages 143–157, online. Association for Computational Linguistics.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

France Martineau, Paul Hirschbühler, Anthony Kroch, and Yves Charles Morin. 2021. MCVF Corpus, parsed, version 2.0. https://github.com/beatrice57/mcvf-plus-ppchf.

Beth Randall. 2010. CorpusSearch 2: a tool for linguistic research. Download site: http://corpussearch.sourceforge.net/CS.html. User guide: https://www.ling.upenn.edu/~beatrice/corpus-ling/CS-users-guide/index.html.

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.

Satoshi Sekine and Michael Collins. 2008. evalb. http://nlp.cs.nyu.edu/evalb/.

Temple F Smith, Michael S Waterman, et al. 1981. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827, Vancouver, Canada. Association for Computational Linguistics.

Ann Taylor, Arja Nurmi, Anthony Warner, Susan Pintzuk, and Terttu Nevalainen. 2006. Parsed Corpus of Early English Correspondence. Distributed by the Oxford Text Archive. Revised corrected version at https://github.com/beatrice57/pceec2.

Ann Taylor, Anthony Warner, Susan Pintzuk, and Frans Beths. 2003. York-Toronto-Helsinki Parsed Corpus of Old English Prose. Distributed by the Oxford Text Archive.

Text Creation Partnership. 2019. Early English Books Online. https://textcreationpartnership.org/tcp-texts/eebo-tcp-early-english-books-online/. Version 2019-04-25.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Joel C. Wallenberg. 2016. Extraposition is disappearing. *Language*, 92(4):e237–e256.

Joel C. Wallenberg, Rachael Bailes, Christine Cuskley, and Anton Karl Ingason. 2021. Smooth signals and syntactic change. *Languages*, 6(2):60.

Joel C. Wallenberg, Anton Karl Ingason, Einar Freyr Sigurdhsson, and Eiríkur Rögnvaldsson. 2011. Icelandic Parsed Historical Corpus (IcePaHC), v0.9. http://www.linguist.is/icelandic_treebank.

Richard Zimmermann. 2017. *Formal and quantitative approaches to the study of syntactic change: Three case studies from the history of English*. Ph.D. thesis, University of Geneva.

A Detailed Overlap Section Statistics

A.1 Sentence lengths by file

Table 10 extends Table 1 from Section 2 to the 42 files shared by PPCEME and EEBO. For each corresponding file (e.g., *armin-e2/A21397*), it details the total number of tokens and mean sentence length for both versions.

A.2 How representative is the overlap section?

In Figure 5 and Table 11 we provide summaries of the sentence length distributions for both PPCEME and EEBO, both overall and for the overlap section alone. From both the summary statistics and kernel density estimates (KDE), it is readily apparent that the PPCEME overlap is very representative of PPCEME overall, showing a nearly identical distribution of sentence lengths. The EEBO overlap shows less of a match to EEBO overall, with the overlap section containing a much higher proportion of extremely short sentences relative to EEBO as a whole.

This striking difference in distributions for EEBO overlap vs EEBO overall is overwhelmingly an artifact of how the overlap correspondence was constructed. As discussed in Appendix B, the EEBO version of the overlap section contains character names in plays that are counted as two-word "sentences". However, when considering all of EEBO, we only consider sentences with EEBO contexts (as indicated by the markup in the EEBO XML files) that are relevant for the query search (<P> indicating prose and <SP/L> indicating verse structure within speech.) The EEBO overall sentence lengths therefore do not include these two-word "sentences". The main point here is that the segmentation problem discussed throughout the main text is not peculiar to the EEBO overlap section. The average sentence length throughout EEBO is greater than that of PPCEME.

B Alignment Details

B.1 Alignment by Source

Table 12 expands on Table 2 by providing full alignment statistics for each text. In addition to raw counts for number of insertions/deletions/substitutions in each text, it also provides a summary statistic for alignment quality – token error rate (TER) – which is defined as:

$$TER = 100 * \frac{\text{\# insert} + \text{\# del} + \text{\# sub}}{\text{\# PPCEME tokens}} \quad (2)$$

where # insert/del/sub are the total count of insertion, deletion, and substitution errors for the text.

B.2 Alignment Algorithm Details

As mentioned in Section 2.2, the sentences of the PPCEME source texts are not always in the same order as in the corresponding EEBO files, and so we first focused on a rough correspondence between the PPCEME and EEBO versions of the overlap, followed by the word alignment. Since the sentences of the EEBO files were in the proper order, we rearranged the PPCEME sentences to match that order. At the same time, some of the meta info in PPCEME, such as character names in plays, was filtered out of the the PPCEME source by the initial preprocessing of PPCEME. As a result, the EEBO overlap has instances of character names that are not present in the PPCEME version of the overlap.

We spot-checked cases of unaligned tokens in both directions, making sure that such cases fell into the categories discussed in the text (e.g., the first two cases in Figure 1), or the character names just discussed. In addition, each pair of aligned tokens has a Levenshtein distance similarity score,

	PPCEME				EEBO			
	name	# sents	# tokens	tokens/sent	name	# sents	# tokens	tokens/sent
0	armin-e2	1,271	18,768	14.77	A21397	358	18,150	50.70
1	asch-e1	496	16,121	32.50	A21975	314	16,010	50.99
2	bacon-e2	480	20,181	42.04	A01516	260	20,209	77.73
3	behn-e3	675	19,335	28.64	A27305	302	19,481	64.51
4	blundev-e2	750	22,619	30.16	A16221	374	22,787	60.93
5	boethpr-e3	1499	32,806	21.89	A28548	1,332	33,176	24.91
6	boylecol-e3	165	7,544	45.72	A28975	78	7,545	96.73
7	brinsley-e2	656	19,710	30.05	A16865	590	19,830	33.61
8	burnetroc-e3	680	21,112	31.05	A30466	356	21,123	59.33
9	clowes-e2	905	22,500	24.86	A19029	427	21,937	51.37
10	coverte-e2	984	20,769	21.11	A19470	446	20,785	46.60
11	deloney-e2	1346	26,738	19.86	A20126	679	27,014	39.78
12	elyot-e1	514	19,157	37.27	A21287	472	19,387	41.07
13	fabyan-e1	507	19,029	37.53	A00525	518	19,023	36.72
14	fisher-e1	466	10,915	23.42	A00771	891	10,918	12.25
15	fitzh-e1	1058	18,813	17.78	A00884	550	19,068	34.67
16	fryer-e3	610	18,970	31.10	A40522	279	19,093	68.43
17	gifford-e2	1230	21,148	17.19	A01716	922	21,642	23.47
18	harman-e1	1115	19,366	17.37	A02657	372	18,026	48.46
19	hooke-e3	539	22,494	41.73	A44323	247	22,464	90.95
20	hooker-a-e2	343	9,025	26.31	A03598	233	9,043	38.81
21	hooker-b-e2	405	8,600	21.23	A03598	258	8,641	33.49
22	hoole-e3	552	21,531	39.01	A44390	364	21,527	59.14
23	jetaylormeas-e3	404	8,682	21.49	A64030	130	8,753	67.33
24	jotaylor-e2	1106	31,202	28.21	A13415	367	31,215	85.05
25	langf-e3	767	18,351	23.93	A49545	355	18,140	51.10
26	latimer-e1	966	17,603	18.22	A05143	698	17,827	25.54
27	markham-e2	253	6,138	24.26	A06913	47	6,192	131.74
28	middlet-e2	2117	19,051	9.00	A07493	3,111	21,624	6.95
29	milton-e3	638	21,307	33.40	A50902	395	21,325	53.99
30	record-e1	1092	23,422	21.45	A10541	620	23,778	38.35
31	shakesp-e2	2332	22,032	9.45	A11954	2,315	24,166	10.44
32	smith-e2	949	18,408	19.40	A12367	457	18,463	40.40
33	stevenso-e1	1512	16,936	11.20	A12969	1,962	17,404	8.87
34	stow-e2	640	17,457	27.28	A13043	353	17,627	49.93
35	turner-e1	581	16,302	28.06	A14053	464	16,320	35.17
36	turnerherb-e1	43	837	19.47	A14059	31	844	27.23
37	tyndnew-e1	2906	39,476	13.58	A68940	1,931	39,654	20.54
38	tyndold-e1	2149	33,901	15.78	A13203	1,209	34,080	28.19
39	vanbr-e3	2081	25,052	12.04	A65075	2,570	27,954	10.88
40	vicary-e1	954	19,510	20.45	A14387	424	19,269	45.45
41	walton-e3	664	12,557	18.91	A67462	317	12,433	39.22
	total	**39,400**	**805,475**	**20.44**		**28,378**	**813,947**	**28.68**

Table 10: Overlap between PPCEME and EEBO, with filename, number of tokens, number of sentences, and mean sentence length.

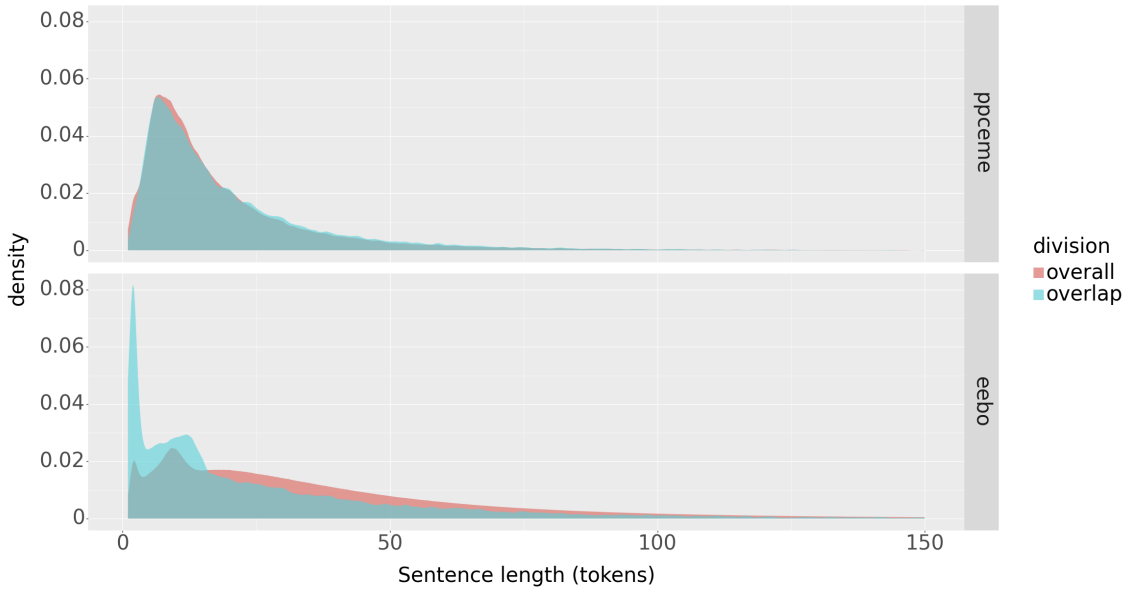

Figure 5: Kernel density estimates (KDE) for sentence length in PPCEME and EEBO. KDEs for the overall corpus and overlap section are plotted on the same figure.

			sentence length								
source	division	# sents	mean	std	min	10%	25%	50%	75%	90%	max
EEBO	overall	33,840,032	41.45	45.57	1	7	14	29	53	88	8,195
EEBO	overlap	28,378	28.68	36.89	1	2	6	16	37	69	562
PPCEME	overall	94,462	20.20	24.77	1	5	8	13	24	41	957
PPCEME	overlap	39,400	20.44	20.01	1	5	8	14	26	43	399

Table 11: Sentence length summary statistics in PPCEME and EEBO. The following statistics are presented for each corpus (both overall and for the overlap section only): mean/standard deviation, min/max, and 10/25/50/75/90-th percentiles.

	# tokens					
text	PPCEME	EEBO	TER	# insertions	# deletions	# substitutions
00-armin-e2	18,768	18,150	23.22	277	895	3,185
01-asch-e1	16,121	16,010	3.60	50	161	370
02-bacon-e2	20,181	20,209	2.16	69	41	326
03-behn-e3	19,335	19,481	19.73	262	116	3,437
04-blundev-e2	22,619	22,787	9.77	285	117	1,807
05-boethpr-e3	32,806	33,176	2.12	411	41	245
06-boylecol-e3	7,544	7,545	1.50	17	16	80
07-brinsley-e2	19,710	19,830	13.28	274	154	2,190
08-burnetroc-e3	21,112	21,123	1.17	44	33	170
09-clowes-e2	22,500	21,937	7.10	108	671	819
10-coverte-e2	20,769	20,785	3.53	40	24	670
11-deloney-e2	26,738	27,014	13.60	436	160	3,040
12-elyot-e1	19,157	19,387	24.97	630	400	3,753
13-fabyan-e1	19,029	19,023	36.82	421	427	6,159
14-fisher-e1	10,915	10,918	15.72	67	64	1,585
15-fitzh-e1	18,813	19,068	7.94	428	173	892
16-fryer-e3	18,970	19,093	2.31	160	37	242
17-gifford-e2	21,148	21,642	4.59	530	36	404
18-harman-e1	19,366	18,026	29.36	178	1518	3,990
19-hooke-e3	22,494	22,464	1.55	47	77	224
20-hooker-a-e2	9,025	9,043	1.97	45	27	106
21-hooker-b-e2	8,600	8,641	2.38	63	22	120
22-hoole-e3	21,531	21,527	2.16	91	95	279
23-jetaylormeas-e3	8,682	8,753	7.79	175	104	397
24-jotaylor-e2	31,202	31,215	3.98	141	128	972
25-langf-e3	18,351	18,140	5.70	115	326	605
26-latimer-e1	17,603	17,827	28.86	393	169	4,519
27-markham-e2	6,138	6,192	7.41	79	25	351
28-middlet-e2	19,051	21,624	17.35	2665	92	548
29-milton-e3	21,307	21,325	0.82	33	15	126
30-record-e1	23,422	23,778	11.32	554	198	1,899
31-shakesp-e2	22,032	24,166	13.41	2259	125	570
32-smith-e2	18,408	18,463	1.51	82	27	169
33-stevenso-e1	16,936	17,404	13.52	763	295	1,231
34-stow-e2	17,457	17,627	5.31	205	35	687
35-turner-e1	16,302	16,320	4.31	133	115	454
36-turnerherb-e1	837	844	10.99	7	0	85
37-tyndnew-e1	39,476	39,654	13.07	258	80	4,821
38-tyndold-e1	33,901	34,080	8.46	300	121	2,448
39-vanbr-e3	25,052	27,954	19.41	3247	345	1,270
40-vicary-e1	19,510	19,269	19.27	204	445	3,111
41-walton-e3	12,557	12,433	21.34	697	821	1,162
total	**805,475**	**813,947**	**10.62**	**17,243**	**8,771**	**59,518**

Table 12: Token alignment statistics for each text. The first two columns indicate the token counts in the PPCEME and EEBO versions of the text. The next four columns provide information about the alignment quality with *# insertions*, *# deletions*, and *# substitutions* indicating the total number of insertion, deletion, and substitution errors in the EEBO text relative to the PPCEME text given the alignment. TER is a summary statistic defined as in eqn. 2

hyperparameter	value
attention_dropout	0.2
batch_size	32
char_lstm_input_dropout	0.2
checks_per_epoch	4
clip_grad_norm	0.0
d_char_emb	64
d_ff	2048
d_kv	64
d_label_hidden	256
d_model	1,024
d_tag_hidden	256
elmo_dropout	0.5
encoder_max_len	512
force_root_constituent	'auto'
learning_rate	5e-05
learning_rate_warmup_steps	160
max_consecutive_decays	3
max_len_dev	0
max_len_train	0
morpho_emb_dropout	0.2
num_heads	8
num_layers	8
predict_tags	True
relu_dropout	0.1
residual_dropout	0.2
step_decay_factor	0.5
step_decay_patience	5
tag_loss_scale	5.0
max_epochs	50

Table 13: Hyperparameters used with the Berkeley Neural Parser.

modified by common and expected cases for character differences, as discussed in Section 2.2. We spot-checked cases where the similarity was below 0.9, which highlighted cases such as those discussed in Section 2.2 (e.g., *&* and *and*). We then treated these as special cases for the similarity metric and redid the alignment, in an iterative process.

C Model and Evaluation

Table 13 shows the hyperparameter settings used in the Berkeley Neural Parser (all default). We added a parameter `max_epochs` for the maximum number of epochs, setting it to 50 for the cross-validation training reported.

C.1 RoBERTa Pretraining

We downloaded the most recent version of English *roberta-base* from Huggingface[11] and continued pre-training for two epochs on EEBO. EEBO was preprocessed using the same steps as described in Kulick et al. (2022a,b), yielding a 1.374 billion token train set and 115K token validation set. We used the `run_mlm` script from Hugging Face with a batch size of 2 on 5 GPUs for an effective batch size of 10. Future work will explore improved performance as a function of larger models and/or additional epochs.

C.2 Function Tags

Function tags are important for us since the queries rely upon them to find the structures of linguistic interest. As in Kulick et al. (2022a,b), we adopted the approach of Gabbard et al. (2006) to function tag recovery. The function tags are retained in preprocessing, and so nonterminals like `NP-SBJ` are treated as atomic units. Since the decision whether to delete is part of preprocessing, this approach does not require modification to the parser.

C.3 Default Flat Parses

Of the 28,378 sentences in the EEBO overlap section, 5 exceeded the 512 subword limit imposed by the language model and the encoder within the parser. For such cases, we modified the parser to output a dummy flat parse, with each token assigned the tag XX.

D Cross-Validation Splits

Table 14 summarizes the composition of the train/dev/test sections across the cross-validation 8 splits; specifically, the total number of files, the total number of tokens, and the percentage of total tokens in each section. Since the partitioning process is performed at the level of PPCEME source files, and these files differ substantially in size, there is some variation in these numbers across the splits. For this reason, we report standard deviations as well as means. The final row ("total") gives numbers for a complete split (i.e., the train/dev/test sections combined); as these are constant across each split, they have a standard deviation of zero. As can be seen, overall the splits attain the target 90-5-5 breakdown; e.g., the train section on average comprises 89.65% of the total tokens with a standard deviation of 0.54%.

The total number of tokens here (1,944,480) is greater than the total number of tokens listed in Table 4 (1,907,787). This is because some sentences were removed from the PPCEME overlap files in

[11] https://huggingface.co/roberta-base

section	# files		# tokens		% of split	
train	205.88	(13.34)	1,743,211.25	(10,441.53)	89.65	(0.54)
dev	12.50	(7.15)	101,000.12	(4,081.82)	5.19	(0.21)
test	13.62	(7.91)	100,268.62	(7,832.66)	5.16	(0.40)
total	232	(0.00)	1,944,480	(0.00)	100	(0.00)

Table 14: Mean number of files and tokens for train/dev/test sections across the 8 cross-validation splits (standard deviations in parentheses). The percentage of tokens in each section is given in column "% of split".

the course of preprocessing.

E Results for PPCEME Overlap

Table 15 breaks down the scores for each of the overlap files in PPCEME. The totals for all files correspond to the number of tokens in Table 4 and the scores in the top part of Table 5.

F Results for EEBO Overlap

Here we expand in two ways on the POS tagging results on EEBO from Section 4.2. First, Table 16 breaks down the results in the bottom part of Table 5 by file. Table 17 shows the complete listing of overall results by tag, the 20 most frequent of which were shown in Table 6. The tag XX occurs in the five overly long sentences mentioned in Section C.3.

G Full Query Details

We wish to identify certain sentence types that allow us to track the rise of auxiliary *do* over the course of Early Modern English. For expository reasons, we present these sentence types in reverse chronological order.[12]

G.1 Sentence Types with Auxiliary *Do*

Modern English is unusual in requiring the auxiliary verb *do* in negative declarative sentences, negative imperatives, and all direct questions (whether positive or negative).

DO-NOT-DECL. In negative declarative sentences, the main verb appears in uninflected form. Such sentences also contain auxiliary *do* in either the present or past tense, and the negative marker *not* appears between the auxiliary and the main verb.

```
(IP-SUB (NP-SBJ (PRO they))
        (DOP do)
        (NEG not)
        (NP-MSR (Q much))
        (VB minde)
        (NP-OB1 (PRO them))
```

The IP in this sentence type (and also its historical counterpart without *do*) can be an independent matrix (MAT) clause or, as here, a subordinate (SUB) clause.

`Do-not-imp`. Negative imperatives are analogous, except for the IMP function tag on IP, and the imperative POS tag (DOI) on the auxiliary.

```
(IP-IMP (PP (P For)
            (NP (NPR$ God's)
                (N sake)))
        (DOI do)
        (NEG not)
        (VB overlay)
        (NP-OB1 (PRO me))
        (PP (P with)
            (NP (ADJ superfluous)
                (N Matter)))
    (. .))
```

`Do-sbj`. Finally, in direct questions, auxiliary *do* precedes the subject instead of following it. This inversion occurs in both positive and negative questions, and so retrieving this sentence type relies on the parser correctly identifying the subject via the SBJ function tag. The annotation guidelines for PPCEME require direct questions to be annotated as CP-QUE-MAT immediately dominating IP-SUB. In this context, IP-SUB is understood as part of the direct question rather than an ordinary subordinate clause.

```
(CP-QUE-MAT (WADVP (WADV How))
    (IP-SUB (DOP do's)
            (NP-SBJ (D this) (N Sute))
            (VB fit)
            (NP-OB1 (PRO me)))
    (NP-VOC (NPR Dauy))
    (. ?))
```

G.2 Sentence Types Without Auxiliary *Do*

We now illustrate the historical precursors of the modern sentence types just discussed. In all 3 old forms, it is the main verb (rather than auxiliary *do*) that appears in a past or present tense

[12] We are concerned only with sentences without modal verbs (*can, will*, etc.), aspectual auxiliaries *have* and *be*, or main verb *be*; sentences containing these elements were not affected by the change.

text	# tokens	recall	prec	f1	pos
00-armin-e2	18,768	91.67	92.49	92.08	98.42
01-asch-e1	16,121	89.42	89.96	89.69	98.57
02-bacon-e2	20,181	90.28	91.06	90.67	99.11
03-behn-e3	19,335	92.61	92.69	92.65	99.24
04-blundev-e2	22,619	88.44	90.76	89.58	98.21
05-boethpr-e3	32,806	95.08	95.27	95.17	99.29
06-boylecol-e3	7,544	90.66	91.53	91.09	98.63
07-brinsley-e2	19,710	89.38	89.98	89.68	98.43
08-burnetroc-e3	21,112	93.70	94.05	93.87	99.30
09-clowes-e2	22,500	90.34	91.13	90.73	98.32
10-coverte-e2	20,769	90.83	91.23	91.03	98.39
11-deloney-e2	26,738	93.10	93.43	93.26	98.58
12-elyot-e1	19,157	90.79	91.73	91.26	98.55
13-fabyan-e1	19,029	89.33	89.82	89.57	97.91
14-fisher-e1	10,915	91.13	91.50	91.31	97.27
15-fitzh-e1	18,813	90.51	90.87	90.69	97.74
16-fryer-e3	18,970	88.83	89.29	89.06	97.83
17-gifford-e2	21,148	93.90	94.36	94.13	98.84
18-harman-e1	19,366	90.90	91.80	91.35	98.01
19-hooke-e3	22,494	88.69	88.97	88.83	98.54
20-hooker-a-e2	9,025	91.00	91.79	91.39	98.67
21-hooker-b-e2	8,600	92.13	92.96	92.54	99.09
22-hoole-e3	21,531	89.79	90.27	90.03	98.37
23-jetaylormeas-e3	8,682	93.01	94.03	93.52	99.14
24-jotaylor-e2	31,202	90.72	91.35	91.03	98.50
25-langf-e3	18,351	90.45	90.90	90.67	98.72
26-latimer-e1	17,603	91.40	92.24	91.82	98.53
27-markham-e2	6,138	90.53	91.17	90.85	98.32
28-middlet-e2	19,051	90.09	91.12	90.60	97.25
29-milton-e3	21,307	88.24	89.11	88.67	99.02
30-record-e1	23,422	89.58	89.61	89.59	94.75
31-shakesp-e2	22,032	91.24	91.72	91.48	97.20
32-smith-e2	18,408	94.70	95.07	94.88	99.19
33-stevenso-e1	16,936	84.78	87.10	85.92	93.43
34-stow-e2	17,457	91.66	91.91	91.78	98.75
35-turner-e1	16,302	89.47	90.10	89.78	98.26
36-turnerherb-e1	837	66.55	70.09	68.27	90.20
37-tyndnew-e1	39,476	96.27	96.61	96.44	98.82
38-tyndold-e1	33,901	93.29	93.56	93.42	98.36
39-vanbr-e3	25,052	94.13	94.24	94.18	98.46
40-vicary-e1	19,510	91.36	92.08	91.72	97.89
41-walton-e3	12,557	92.36	92.42	92.39	98.96
total	**805,475**	**91.35**	**91.94**	**91.64**	**98.26**

Table 15: Breakdown of aggregate evalb and POS results for PPCEME overlap files shown in Table 5.

		aligned		non-punc		bullet	
sec	# not aligned	#	acc	#	acc	#	acc
00-armin-e2	277	17,873	88.27	15,858	91.79	6	66.67
01-asch-e1	50	15,960	97.07	13,250	98.13	2	100.00
02-bacon-e2	69	20,140	97.66	17,696	98.81	52	98.08
03-behn-e3	262	19,219	96.72	16,885	98.74	1	0.00
04-blundev-e2	285	22,502	94.24	20,475	95.64	9	88.89
05-boethpr-e3	411	32,765	97.45	29,068	99.07	2	100.00
06-boylecol-e3	17	7,528	97.48	6,793	98.48	0	0.00
07-brinsley-e2	274	19,556	97.21	17,207	98.01	21	85.71
08-burnetroc-e3	44	21,079	97.46	18,832	99.14	0	0.00
09-clowes-e2	108	21,829	95.74	19,360	98.03	36	91.67
10-coverte-e2	40	20,745	95.86	18,268	98.04	87	83.91
11-deloney-e2	436	26,578	95.79	23,553	98.25	4	100.00
12-elyot-e1	630	18,757	96.83	16,789	97.83	2	100.00
13-fabyan-e1	421	18,602	95.86	17,005	97.39	47	82.98
14-fisher-e1	67	10,851	92.43	9,924	96.37	4	75.00
15-fitzh-e1	428	18,640	94.25	16,134	96.88	5	100.00
16-fryer-e3	160	18,933	95.63	16,626	97.46	10	90.00
17-gifford-e2	530	21,112	96.03	18,571	98.51	6	100.00
18-harman-e1	178	17,848	93.32	16,147	96.64	130	79.23
19-hooke-e3	47	22,417	97.06	19,833	98.34	16	100.00
20-hooker-a-e2	45	8,998	97.18	7,950	98.34	2	100.00
21-hooker-b-e2	63	8,578	97.18	7,513	98.70	12	91.67
22-hoole-e3	91	21,436	97.26	19,128	98.11	18	83.33
23-jetaylormeas-e3	175	8,578	89.83	7,697	92.19	27	22.22
24-jotaylor-e2	141	31,074	95.61	27,332	97.74	401	76.81
25-langf-e3	115	18,025	96.48	16,111	98.41	4	100.00
26-latimer-e1	393	17,434	95.21	15,422	97.22	0	0.00
27-markham-e2	79	6,113	86.52	5,579	87.99	5	80.00
28-middlet-e2	2665	18,959	91.80	16,057	95.39	4	75.00
29-milton-e3	33	21,292	97.24	18,470	98.74	6	100.00
30-record-e1	554	23,224	92.24	20,747	93.41	7	57.14
31-shakesp-e2	2259	21,907	91.40	18,220	95.93	5	60.00
32-smith-e2	82	18,381	96.14	16,035	98.88	2	50.00
33-stevenso-e1	763	16,641	88.40	14,569	90.10	385	64.42
34-stow-e2	205	17,422	96.61	15,386	98.39	54	88.89
35-turner-e1	133	16,187	95.72	14,315	97.88	2	50.00
36-turnerherb-e1	7	837	96.30	747	97.46	0	0.00
37-tyndnew-e1	258	39,396	96.13	34,304	98.36	257	93.00
38-tyndold-e1	300	33,780	95.66	30,471	97.52	241	87.55
39-vanbr-e3	3247	24,707	94.22	20,975	97.24	7	57.14
40-vicary-e1	204	19,065	94.36	16,832	97.08	178	85.39
41-walton-e3	697	11736	92.98	10,330	96.56	0	0.00
total	**17,243**	**796,704**	**95.17**	**702,464**	**97.25**	**2,057**	**80.12**

Table 16: Breakdown of aggregate POS results for PPCEME overlap files from Table 5. "aligned" includes all aligned PPCEME tokens (796,704), "non-punc" excludes punctuation tags, and "bullet" includes only words with a bullet character.

tag	gold	EEBO	rec	prec	f1	tag	gold	EEBO	rec	prec	f1
N	93,720	92,513	96.82	95.57	96.19	ADJR	1,530	1527	93.71	93.53	93.62
P	91,175	91,190	98.89	98.91	98.90	EX	1478	1495	97.26	98.38	97.81
,	57,992	71,966	76.91	95.44	85.18	HV	1382	1368	98.98	97.97	98.47
D	62,701	62,440	99.49	99.08	99.29	INTJ	1378	1457	80.44	85.05	82.68
PRO	52,368	52,204	99.34	99.03	99.19	SUCH	1361	1352	99.70	99.04	99.37
CONJ	42,478	42,154	99.44	98.68	99.06	ADJS	1288	1309	95.19	96.74	95.96
ADJ	35,769	35,480	95.93	95.16	95.54	N$	1217	1238	87.96	89.48	88.72
NS	30,937	30,974	96.79	96.91	96.85	ALSO	1212	1205	99.59	99.01	99.30
ADV	24,804	24,477	96.83	95.56	96.19	NPRS	1177	1186	82.21	82.84	82.52
VB	22,724	22,718	97.39	97.37	97.38	BAG	1163	1161	99.40	99.23	99.31
.	36,415	22,274	88.70	54.26	67.33	WD	1127	1145	95.81	97.34	96.57
NPR	19,277	20,210	88.36	92.64	90.45	QS	989	1001	97.70	98.89	98.29
PRO$	17,060	17,023	99.37	99.16	99.26	DOD	887	884	99.66	99.32	99.49
BEP	14,938	14,905	99.14	98.92	99.03	BEN	730	726	99.72	99.18	99.45
VAN	14,540	14,726	95.43	96.65	96.04	DO	728	740	97.16	98.76	97.96
VBP	14,291	14,345	95.88	96.24	96.06	NPR$	707	695	86.19	84.72	85.45
Q	14,044	13,998	98.75	98.43	98.59	OTHERS	487	506	91.50	95.07	93.25
MD	13,828	13,709	99.43	98.58	99.00	HAG	397	393	98.98	97.98	98.48
VBD	13,663	13,653	97.48	97.41	97.44	WARD	320	323	92.57	93.44	93.00
TO	10,890	10,858	99.54	99.25	99.39	WPRO$	315	310	99.03	97.46	98.24
C	9,071	9,113	97.52	97.97	97.75	DAN	266	266	98.12	98.12	98.12
WPRO	7,934	7,920	99.12	98.94	99.03	WQ	258	259	91.51	91.86	91.68
NUM	6,419	6,473	94.72	95.51	95.11	NS$	254	294	67.69	78.35	72.63
VAG	6,181	6,140	95.70	95.07	95.38	FOR	244	259	92.28	97.95	95.03
BED	6,014	6,001	99.60	99.38	99.49	DON	186	184	97.83	96.77	97.30
NEG	5,720	5,691	99.58	99.07	99.33	ADVS	186	150	95.33	76.88	85.12
BE	5,379	5,361	99.22	98.88	99.05	ELSE	133	138	92.03	95.49	93.73
VBN	4,547	4,586	96.14	96.97	96.55	DOI	108	106	94.34	92.59	93.46
FW	4,637	4,332	91.60	85.57	88.48	HVN	89	91	91.21	93.26	92.22
HVP	4,276	4,265	99.11	98.85	98.98	BEI	83	88	81.82	86.75	84.21
RP	4,194	4,182	95.84	95.57	95.70	DAG	55	44	65.91	52.73	58.59
ADVR	3,930	3,880	97.14	95.90	96.52	HAN	52	56	92.86	100.00	96.30
VBI	3,991	3,733	93.65	87.60	90.52	ONES	52	54	94.44	98.08	96.23
WADV	2,874	2,822	98.02	96.24	97.12	NPRS$	36	33	81.82	75.00	78.26
ONE	2,483	2,478	98.95	98.75	98.85	HVI	27	21	85.71	66.67	75.00
OTHER	2,430	2,441	97.91	98.35	98.13	X	24	80	0.00	0.00	0.00
XX	0	2,242	0.00	0.00	0.00	$	23	23	65.22	65.22	65.22
HVD	2,064	2,051	99.07	98.45	98.76	OTHER$	21	22	81.82	85.71	83.72
DOP	2,028	2,026	99.01	98.92	98.96	"	16	9	0.00	0.00	0.00
FP	1,833	1,847	95.34	96.07	95.71	ONE$	12	13	84.62	91.67	88.00
QR	1,718	1,717	98.66	98.60	98.63	’	9	0	0.00	0.00	0.00
OPAREN	1,690	1,703	97.18	97.93	97.55	OTHERS$	6	4	75.00	50.00	60.00
CPAREN	1,661	1,668	97.24	97.65	97.45	NUM$	3	0	0.00	0.00	0.00
total	**796,704**	**796,704**	**95.17**	**95.17**	**95.17**						

Table 17: Complete breakdown by tag of 95.17% score in row “aligned words” in Table 5, extending Table 6. EEBO tags are mapped to PPCEME (gold) tags using token alignment.

form, and it occupies the same position as auxiliary *do*. Thus, we have negative declarative sentences (VERB-DECL-NOT) like:

```
(IP-SUB (NP-SBJ (PRO I))
        (VBD sent)
        (NEG not)
        (PP (P to)
            (NP (PRO you))))
```

negative imperatives (VERB-NOT-IMP) like:

```
(IP-IMP (VBI let)
        (NEG not)
        (IP-INF (NP-SBJ (D that))
                (VB hurt)
                (NP-OB1 (PRO me)))
    (. .))
```

and questions (VERB-SBJ) like:

```
(CP-QUE-MAT
    (WADVP (WADV When))
    (IP-SUB (VBP comes)
            (NP-SBJ (PRO$ your)
                    (N Taylor))
            (ADVP-DIR (ADV hither)))
    (. ?))
```

G.3 Sample CorpusSearch Query

In order to retrieve the 6 diagnostic sentence types, we formulate queries in CorpusSearch (Randall, 2010), a query language for querying, editing, and coding tree structures. Each query is a sequence of boolean conditions on the parser output. For instance, the following query retrieves direct questions with auxiliary *do* (DO-SBJ).

```
    (CP-QUE-MAT* iDoms IP-SUB*)
AND (IP-SUB* iDoms DOD|DOP)
AND (IP-SUB* iDoms NP-SBJ*)
AND (IP-SUB* iDoms DO|VB)
AND (DOD|DOP precedes NP-SBJ*)
AND (NP-SBJ* precedes DO|VB)
```

The asterisks on the labels allow the query to match tokens with further trailing function tags (say, -SPE to indicate direct speech or -RSP for resumptive subjects). Our formulation of the queries assumes that the parser has correctly constructed the relevant clause boundaries.

H Alignment-Mediated Scoring

For the bipartite graph minimum weight matching problem, we use the scipy implementation https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linear_sum_assignment.html.

The *PPCEME-gold* trees that are compared against are the original "psd" files from the release, because we wanted to avoid any possibility of preprocessing of PPCEME affecting the "gold" results. These ".psd" trees have empty categories and meta data that affects the spans of the query hits on these gold trees. For example, the lower gold tree in Figure 3(a) has a `(ADVP *T*-1)`, and the span for the VERB-SBJ, `<3272,3278>` includes that empty leaf.

As mentioned in Section 2.2, we follow the preprocessing of the PPCEME files as described in Kulick et al. (2022b,a), for parsing the PPCEME files. This preprocessing removes empty categories and meta data, and so even the AMS evaluation of the *PPCEME-parsed* files requires a token alignment, although a trivial one that simply skips over the empty categories in the alignment from *PPCEME-parsed* to *PPCEME-gold*. Likewise, the alignment used for the AMS scoring of *EEBO-parsed* to *PPCEME-gold* is a slightly modified version of the alignment to *PPCEME-parsed* discussed in Section 2.2, that aligns to the *PPCEME-gold* trees.

I Cross-Validated Results on PPCEME

As briefly discussed in Section 6.2, in addition to the evalb scores on the cross-validated PPCEME sections using the the new language model (Table 3), we also have query-based evaluation (using the method in Kulick et al. (2022b)) for the cross-validation splits. These results are shown in Table 18 and are generally increases over the scores reported in Kulick et al. (2022b).

As discussed in Section 4.1, the parser used on the overlap section is trained on less data than each of these cross-validation splits were trained on, yielding a score of 91.64%, which is lower (as expected) than the cross-validated results using more training data (Table 3). In order to obtain a more robust measure of how the loss in training data affects the parser, we redid the cross-validation split with the training section of each split cut to 55% of its original size. These results are in the bottom half of Table 19, in which, for convenience, we repeat the scores from Table 3. The score of 91.64% is within the expected range.

	DEV				EVAL			
query	# hits	recall	prec	f1	# hits	recall	prec	f1
				Negative declarative sentences				
VERB-NOT-DECL	720	93.73 (3.5)	92.95 (3.9)	93.32 (3.5)	655	94.05 (3.3)	93.68 (2.8)	93.79 (1.4)
DO-NOT-DECL	339	96.53 (2.3)	98.05 (2.7)	97.23 (0.7)	405	96.26 (4.4)	98.58 (2.4)	97.34 (2.2)
				Negative imperative sentences				
VERB-NOT-IMP	120	93.08 (8.0)	90.93 (8.5)	91.69 (6.2)	143	78.16 (14.1)	83.91 (12.8)	80.69 (12.6)
DO-NOT-IMP	41	74.34 (45.9)	71.01 (44.2)	72.55 (44.9)	23	80.0 (38.5)	87.5 (35.4)	82.14 (36.4)
				Questions				
VERB-SBJ	387	89.83 (8.3)	94.65 (4.5)	92.06 (5.9)	190	78.68 (10.2)	87.37 (8.8)	82.18 (5.3)
DO-SBJ	564	92.64 (3.9)	99.01 (1.0)	95.67 (1.9)	329	94.36 (7.1)	99.46 (1.5)	96.75 (4.4)

Table 18: Query-based results for the cross-validation dev and test sections.

	DEV		EVAL	
% train	evalb	POS	evalb	POS
100	92.08 (1.6)	98.23 (0.7)	91.77 (0.6)	98.37 (0.3)
55	91.54 (1.7)	98.02 (0.7)	91.35 (0.7)	98.26 (0.4)

Table 19: Cross-validation parser and POS results. Each result is the mean for the relevant section (dev or test) over the 8 splits (standard deviation in parentheses). Results are reported using both full train section of each split and 55% of the train section.