Psycholinguistically motivated Construction-based Tree Adjoining Grammar

Shingo Hattori* **Laura Kallmeyer**** **Rainer Osswald****
*University of Tokyo **Heinrich Heine University Düsseldorf
*shinhattori@g.ecc.u-tokyo.ac.jp
**{kallmeyer,osswald}@phil.hhu.de

Abstract

This paper proposes a formal framework based on Tree Adjoining Grammar (TAG) that aims to incorporate central tenets of Construction Grammar while integrating mechanisms from a psycholinguistically motivated variant of TAG. Central ideas are (i) to give TAG-inspired tree representation to various constructions including schematic constructions like argument structure constructions, (ii) to link schematic constructions that are extensions of each other within a network of constructions, (iii) to make the derivation proceed incrementally, (iv) to allow the prediction of upcoming constructions during derivation and (v) to introduce the *incremental extension* of schematic constructions to larger ones via *extension trees* in a usage-based manner. The final point is the major novel contribution, which can be conceptualized as the on-the-fly traversal of the inheritance links in the network of constructions. Moreover, we present first experiments towards a parser implementation. We report preliminary results of extracting constructions from the Penn Treebank and automatically identifying constructions to be added during incremental parsing, based on a generative language model (GPT-2).

1 Introduction

Theories of construction grammar (Goldberg, 1995, 2003) posit that the building blocks of language are constructions, or form-meaning pairs at various levels of abstraction: not only words but also phrasal or larger patterns, from multi-word expressions and collocations to syntactic patterns like argument structures. In this approach, those constructions are combined to form representations linked to sentences in a manner constrained by semantic or pragmatic compatibilities as well as usage. It is further hypothesized that these constructions are cognitively organized as a network, whose links represent inheritance relations.

Despite the strong concern with cognitive plausibility in construction grammar, psycholinguistic evaluation of its tenets seems to be done mainly on qualitative predictions (Bencini and Goldberg, 2000; Perek, 2025), while more quantitative evaluation with psycholinguistic data has been attempted for other grammar formalisms (Roark et al., 2009; Padó, 2007; Konieczny, 1996; Stanojević et al., 2023; Brennan et al., 2016). This is not surprising, given that the existing formalized variants of construction grammar (Bergen and Chang, 2005; Steels, 2017; Boas and Sag, 2012) and studies of computational extraction of constructions (Dunn, 2017) appear to lack broad coverage and psycholinguistically plausible parsers.

In this regard, Tree Adjoining Grammar (TAG, Joshi et al., 1975) seems to be a promising framework to formalize and implement construction grammar, as has been suggested in Kallmeyer and Osswald (2013) and Lichte and Kallmeyer (2017) among others. Moreover, there is a psycholinguistically motivated variant of TAG with an incremental broad-coverage parser (Psycholinguistically motivated TAG, PLTAG, Demberg et al., 2013; Demberg-Winterfors, 2010). PLTAG, however, does not take into account all the key tenets of construction grammar.

Thus, we will develop Psycholinguistically motivated Construction-based TAG, or PLCxTAG, a formalization of construction grammar inspired by PLTAG. In addition, we will present a preliminary implementation of the framework leveraging a neural language model (LM) as a proof of concept, including a lexicon automatically extracted from the Penn Treebank (Marcus et al., 1993, PTB) and a broad-coverage supertagger based on a decoder LM (GPT-2, Radford et al., 2019), which will comprise the core of an incremental parser for psycholinguistic evaluation to be conducted in future work.

Proceedings of the Second International Workshop on Construction Grammars and NLP, pages 130–143

2 Related work

2.1 Construction grammar

Key tenets of construction grammar. The focus of our approach is on the following five key tenets of construction grammar (Goldberg, 2003). First, the grammar is viewed as the *composition of constructions*, which are form-meaning pairs stored in memory. This requires that phrasal or larger patterns can be directly associated with meaning. Also, the constructions are combined depending on the semantic compatibility among them, rejecting the autonomy of syntax. Second, *schematic constructions* such as argument structure constructions (Goldberg, 1995) are also memorized. These capture the regularities traditionally described in syntax. Third, construction grammar is generally *surface-oriented* (Goldberg, 2003). Great emphasis is placed on the surface generalization, without resorting to assumptions about deep structure from which surface structures might be derived. Also, phonologically null elements like traces, PRO or null function heads are avoided. Fourth, the theories of construction grammar often take a *usage-based approach* (Langacker, 1987; Bybee, 2010; Tomasello, 2005), which postulates that specific usages are memorized according to their frequencies, and more general constructions like schematic constructions arise in a bottom-up manner. Finally, the lexicon of constructions is postulated to be structured as a *network of constructions* (Diessel, 2023), where constructions are connected based on inheritance relations among them, where more specific constructions "inherit" information from abstract constructions. This network captures how schematic constructions, such as argument structure constructions, productively license quite rare but grammatical uses, e.g. "sneeze the foam off the cappuccino" or "kick Bob a ball" (Goldberg, 1995).

Formalizations of construction grammar. There are three major computational theories of construction grammar: Embodied Construction Grammar (ECG, Bergen and Chang, 2005; Chang, 2008; Feldman, 2022; Bryant, 2008), Fluid Construction Grammar (FCG, Steels, 2017; Beuls and Van Eecke, 2023) and Sign-Based Construction Grammar (SBCG, Boas and Sag, 2012). All of them have parser implementations, but no incremental parser exists for FCG nor SBCG to our knowledge, though Müller (2017) suggests that existing incremental parsers for Head-driven Phrase Structure Grammar (e.g. Konieczny, 1996) could be adapted to SBCG. ECG does have a psycholinguistically motivated incremental parser, "constructional analyzer" (Bryant, 2008), but the scalability of the parser appears limited due to the need of manually writing the grammar and defining parameters for some phenomena of interest (Bryant, 2008, p. 156).

There have also been attempts to extract constructions automatically with a view to making the study of constructions scalable and not limited to a handful of constructions selected by linguists (Dunn, 2017). Still, it is by no means obvious how these constructions can be combined to form actual sentence representations.

2.2 Modeling human sentence processing

Properties of human sentence processing. Accumulating studies in psycholinguistics have not only identified various psycholinguistic phenomena, such as garden path, indicating the preferences of certain structures over others, but also demonstrated three general principles of human sentence processing (Demberg and Keller, 2019). First, the parse is built *incrementally*, updated for every new word (Konieczny, 2000; Tanenhaus et al., 1995). Second, it is known that the mismatch of subject and reflexive pronoun affects the reading time even before the VP is completed with the second PP object, suggestive of *connected syntactic structure* facilitating such agreement (Sturt and Lombardo, 2005). Finally, parsing proceeds based on *predictions*, e.g. by anticipating the argument of a verb before encountering it (DeLong et al., 2005; Kamide et al., 2003; Staub and Clifton, 2006).

PLTAG. PLTAG is a psycholinguistically motivated variant of TAG (Demberg et al., 2013; Demberg-Winterfors, 2010), which is designed to satisfy the three properties described above.

There are crucial innovations to maintain incrementality and connectedness during derivation, such as the prediction-verification scheme. Moreover, the grammar has been automatically extracted from the PTB, and a broad-coverage parser was implemented based on it, which was then evaluated on reading time data.

Yet, there is some room for exploring alternative formalizations, and more importantly, PLTAG does not satisfy some key tenets of construction grammar, e.g., there is no network of constructions

and null elements are used extensively (though this latter property is not inherent to the PLTAG formalism, i.e., it would be straightforward to choose a grammar without null elements).

3 Formal framework

Our formalization is guided by the principles of linguistic and psycholinguistic plausibility. Conditions for *linguistic plausibility* consist of the five tenets of construction grammar: (i) Grammar as the composition of constructions, (ii) Schematic constructions, (iii) Surface-oriented approach, (iv) Usage-based approach and (v) Network of constructions. As conditions for *psycholinguistic plausibility*, three properties of human sentence processing are chosen: (a) incremental, (b) connected and (c) predictive.

Our formalization incorporates on the one hand aspects of *constructions and their composition*, and on the other hand aspects of incremental processing that lead to *incremental extension of constructions* (along the network of constructions) and additional *prediction of upcoming constructions*.

CxTAG: Constructions and their composition. Our starting point is the use of (lexicalized) TAG (LTAG) and frame semantics for modeling constructions along the lines of Kallmeyer and Osswald (2013) and Lichte and Kallmeyer (2017). In that approach, the elementary trees of TAG are paired with frame-semantic representations (formalized as extended attribute value structures) to *elementary constructions* (or *lexicalized constructions*), in which specific nodes of the tree can be linked to specific components of the semantic frame. The tree-combining operations *substitution* and *adjunction* go along with the unification of the associated frames. (In our case, *substitution* and *sister adjunction* are used.[1]) In the present paper, we keep the semantic side of constructions largely aside since our primary focus is on the formal aspects of incremental syntactic processing as well as on the extraction of the form aspect of constructions from treebanks. Note, however, that in the ongoing parsing implementation (Section 5), semantics is implicitly covered both at the lexical as well as at the constructional level via the embeddings learned in the model.

[1] *Substitution* consists of inserting a tree at a non-terminal leaf, i.e., filling an argument slot. *Sister adjunction* merges the root of the adjoining tree with an internal node, thereby introducing additional subtrees below that internal node.

Elementary trees are partial constituent trees such that each tree has at least one leaf representing the head word, called a *lexical anchor*, and that all of the anchor's projections and arguments are localized in the same tree where arguments are represented as leaves that are substitution nodes; see the tree for 'gave' in Fig. 1 for illustration.[2]

Our theoretical judgment of what qualifies as arguments or adjuncts is more or less in line with standard LTAG analysis (XTAG Research Group, 1998): The subject and the objects of verbs and the noun phrase in prepositional phrases are arguments, while determiners, adjectives, adverbs, auxiliary verbs, semi-auxiliary verbs (e.g. "used to"), copula verbs, raising verbs, complementizers and the infinitive marker "to" are adjuncts.

Elementary constructions also cover multiword expressions, collocations, and frequently co-occurring patterns, motivated by usage-based postulates. In these cases, the corresponding elementary trees can have multiple anchors, called *co-anchors*.

Schematic constructions such as argument structure constructions, however, are not to be represented by LTAG elementary trees since they are unlexicalized and lack a lexical anchor. Therefore, in order to represent them, we employ unlexicalized counterparts of elementary trees known as *supertags* in the TAG literature (Bangalore and Joshi, 2010). The parent node of a removed lexical element in a supertag is called an *anchor node* and is usually marked with a $\diamond$.

The network of constructions and the 'extend' operation. Within the (L)TAG framework, more complex constructions can be derived from simpler ones in a strictly compositional manner by general tree operations such as substitution and adjunction. How different elementary constructions are related to each other and, in particular, how certain constructions can be part of certain other constructions, are not expressed by tree operations but at a different level of grammatical description, often called the *metagrammar* (Kallmeyer and Osswald, 2013; Lichte and Kallmeyer, 2017). The metagrammar allows the specification of trees (and frames) by means of expressive constraint languages (Crabbé et al., 2013; Lichte and Petitjean, 2015).

The resulting set of schematic constructions can

[2] The specific categories and trees used in this paper are to some degree influenced by the PTB (Marcus et al., 1993) employed in Section 5. Notice, however, that the formalization presented here is general and compatible with other constituency formats.

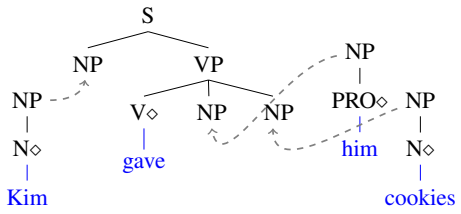

Figure 1: Composition of elementary trees for (1-a); the black trees represent schematic constructions (i.e., 'supertags'); dashed arrows represent tree operations.

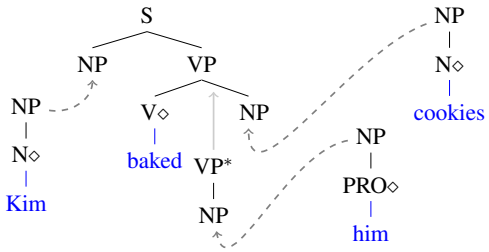

Figure 2: Elementary trees and schematic constructions for (1-b); the solid gray arrow indicates extension by a benefactive NP resulting in the ditransitive construction.

be seen as a *network of constructions*, whose relations indicate specialization ("inheritance") but also more complex types of embeddings of substructures. The described division of labor between general operations on elementary constructions and the more advanced "off-line" specification of elementary constructions and their interrelation by means of constraints have conceptual and practical advantages. The approach falls short, however, if we are to study in which way the network of constructions can guide incremental language processing.

In order to overcome this problem, we propose an additional "operation" *extend*, which mimics standard tree operations, mostly adjunction, but in effect realizes the move from one construction to another, usually more extended construction, which (at least on the semantic side) is typically non-compositional. We refer to the modified formalism as *Construction-based Tree Adjoining Grammar (CxTAG)*.

Schematic, i.e., lexically unanchored constructions are instantiated by lexicalized constructions for words in context. Therefore, instead of treating elementary constructions as part of the lexicon, we further assume here that for a given lexical element w_i, depending on its left context LC_i (comprising $w_1 \ldots w_i$ and any syntactic, semantic and pragmatic structures built so far), schematic constructions $t_i^{\diamond}$ are chosen with a certain probability $P(t_i^{\diamond}|LC_i)$, and also extensions $t_{i,j}^{e}$ of previously chosen constructions $t_j^{\diamond}$ $(j < i)$ to more specific ones occur with a certain probability $P(t_{i,j}^{e}|LC_i)$.[3] This is why trees assigned to each word in the figures contain anchor nodes with $\diamond$.

Consider the examples in (1) for illustration, whose derivations are shown in Figs. 1 and 2.

(1) a. Kim gave him cookies
b. Kim baked him cookies

Both sentences are assumed to give rise to the same syntactic trees except for the lexical head verb. Their derivations differ, however: We may assume that the verb 'gave' generally selects a ditransitive argument structure construction with much higher probability than a transitive construction. The ditransitive construction then provides substitution sites for the two remaining NP arguments. The verb 'baked', by comparison, would select a transitive argument structure construction with higher probability by default as is most likely, and a benefactive NP tree is added by means of extend, which in turn gives rise to a structure that matches an existing construction, namely the benefactive ditransitive construction. In this case, we call the added NP construction (the benefactive NP) an *extension tree* and say that the transitive construction has been extended to the benefactive ditransitive construction.

Note that in general extension trees could also add another co-anchor to extend constructions to those representing multi-word expressions. From the perspective of usage-based approach, extension trees can be seen as secondary generalizations that emerge through the comparison of existing schematic constructions, which are themselves generalizations from instantiations, along with the formation of the network, and they are often interpretable as constructions themselves.[4]

PLCxTAG: Incremental extension of constructions and prediction of upcoming constructions. In the following, we extend the CxTAG formalism in the spirit of PLTAG (Demberg et al., 2013). So far, the order of the derivation steps in CxTAG is

[3] In our implementation, the probabilities for schematic constructions are estimated via fine-tuning a GPT-2 model towards predicting them, see Section 5.

[4] We avoided referring to extension trees as constructions categorically, since extension trees may not always qualify as independent constructions from a semantic viewpoint.

not restricted, but in order to achieve psycholinguistic plausibility, we extend the formalism towards allowing derivations that build connected parses incrementally. This not only imposes constraints on how syntactic operations can be applied but also requires additional mechanisms and operations.

At the same time, our formalism departs from PLTAG in that it aims to capture the key tenets of construction grammar. Crucially, the extension of schematic constructions is an integral part of the incremental derivation: For each word, a supertag representing some schematic construction is added given the context up to it, and it can be extended later to match the appropriate construction by the end of the sentence in a way described in CxTAG. This *incremental extension* might be conceptualized as the traversal of inheritance links during the derivation.

The overall idea of our psycholinguistically motivated modification of CxTAG is that at each word, we add a new elementary tree and at most one extension tree via substitution or sister adjunction, where the operation can be in both directions (i.e., the already derived tree added to the new one by substitution/adjunction or vice versa). These derivation steps are restricted in such a way as to add material only to the right of the rightmost lexical node in the already derived tree. As an example, Fig. 3 shows the sequence of derived trees we obtain with such a derivation when combining the constructions from Fig. 2. The (orange) tree fragment representing the supertag added at 'baked' is extended to a larger supertag at 'him'. The words (in green) above the $\rightsquigarrow$ arrows indicate the next word, whose processing triggers the next derivation step.

Such an incremental connected derivation is, however, not always possible: When the elementary tree of a word should be combined with a node in an elementary tree of a future word, it is impossible to create a connected partial parse. For example, in "John often smiles", the supertags for words up to "often" cannot be combined without the S and VP nodes from the supertag for "smiles" (see Fig. 4).

To remedy this, we employ a restricted version of the prediction-verification scheme proposed in PLTAG (Demberg et al., 2013; Demberg-Winterfors, 2010) and a new scheme, delay, to compensate for the restriction.

The prediction-verification scheme consists of two steps: *prediction* and later *verification*. In prediction, for each word, an additional supertag is optionally selected as a *prediction tree*, such that it contains all the nodes missing at that point but necessary for a connected partial parse. We assume here that the prediction trees are chosen probabilistically, depending on the left context.[5] They are attached via substitution or sister adjunction to the partial parse while keeping track of the fact that they are only predicted. At a later stage, such a prediction tree has to be verified by a matching supertag that is anchored by an actual word. Note that a supertag used to verify a prediction tree will not be attached to the partial parse via substitution or adjunction. Instead, the nodes from the prediction tree have to be mapped to corresponding nodes in the verifying supertag in such a way that labeling and structural relations are preserved. A sample derivation is given in Fig. 5. The prediction tree is the upper-left tree (depicted in gray) and the mapping performed in the verification operation is indicated by dotted arrows. The red numbers at the arrows indicate the order of derivation operations.

Prediction trees can be extended to larger su-

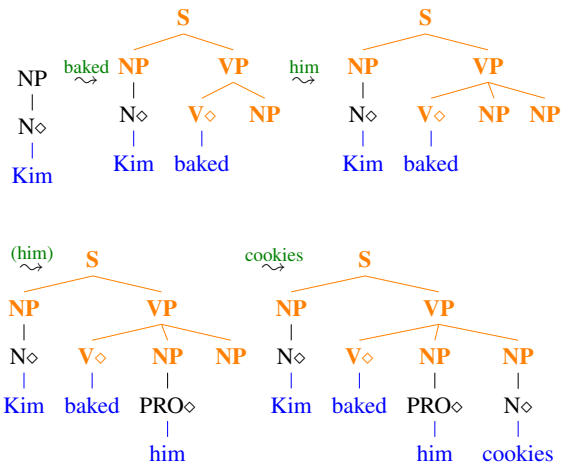

Figure 3: Incremental and connected derivation for (1-b): Incremental extension for inheritance

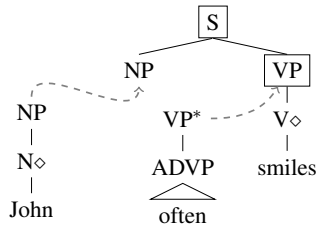

Figure 4: Intervening nodes come from the supertag for a subsequent word

[5]Estimated by the second classification head of the fine-tuned GPT-2 model in our implementation, see Section 5.

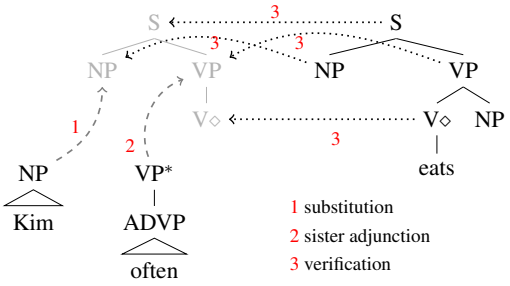

Figure 5: PLCxTAG derivation of 'Kim often eats ...'

pertags due to verification, in cases where the two trees are not isomorphic. For instance, to attach an adverb after a subject NP, as in Fig. 5, it is enough to use an intransitive supertag as the prediction tree, even if the following verb is actually transitive. In this case, the prediction tree is extended during verification due to the verb's transitive supertag.

For prediction-verification, it is an open question how to configure the granularity of predictions: Even though we decided to use supertags as prediction trees, one could also create a separate lexicon of tree fragments that only contain the necessary nodes, as in PLTAG (Demberg et al., 2013; Demberg-Winterfors, 2010).

On the other hand, we forbid adding several prediction trees in a row, following PLTAG. This means that if the nodes needed for a connected partial parse come from multiple supertags, one prediction tree is not enough in our framework. In the example in Fig. 6, the boxed AP and NP nodes are both necessary in order to combine the supertag for 'very' with the partial parse.

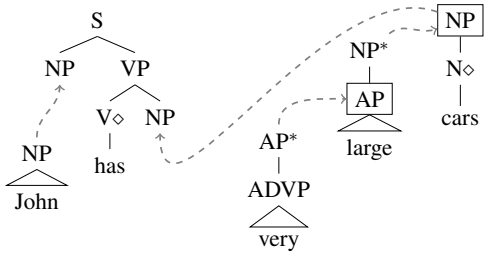

Figure 6: Nodes from multiple supertags

To address such cases, we decided to relax the incrementality condition and allow the delayed attachment of a word's supertag: The creation of a connected partial parse is suspended, waiting for necessary nodes to appear in supertags assigned to subsequent words. To be more precise, we attach the supertag in question to the supertag for the next word first, which in turn will be combined with the partial parse. If needed, we might allow further delays. Our hypothesis is that most actually occurring cases are covered with a maximal delay of 1, based on the inspection of the data from the PTB used in Section 5 below.

The resulting extension of CxTAG is called *Psycholinguistically motivated CxTAG (PLCxTAG)*.

4 Sample derivations involving various constructions

For further illustrations, let us look at a few more interesting examples. It should be noted that the derivations presented below are not prescriptive, and PLCxTAG can be employed to represent alternative analyses.

Argument structure constructions without co-anchors. Let us first consider examples of argument structure constructions: caused motion construction and resultative construction.

(2) a. Kim kicked the ball over the fence
b. Kim sneezed the foam off the cappuccino
c. Kim painted the barn red
d. Kim kicked his feet sore

Derivations for (2-a) and (2-b) are given in Fig. 7–8. The red numbers indicate the order of derivation steps. Dashed arrows indicate substitutions and sister adjunctions that are standard combinations of elementary trees. In contrast, solid gray arrows indicate operations that extend an already chosen elementary tree to a larger one such as the caused motion construction or the resultative construction with extension trees. For the sake of readability, some of the sub-derivations are omitted, i.e., only their result is displayed.

In Fig. 7, the transitive tree selected for 'kick' is extended to the caused motion construction by adding a path PP.

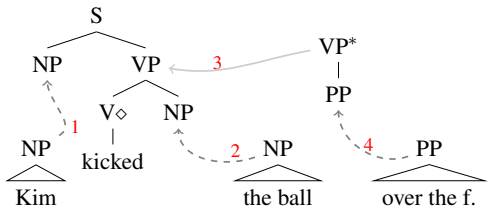

Figure 7: PLCxTAG derivation of (2-a)

The derivation of (2-b) extends an intransitive tree (the 'sneezed' supertag) to the caused motion

construction where slots for both mover (NP) and path (PP) are added. The derivation for resulta-

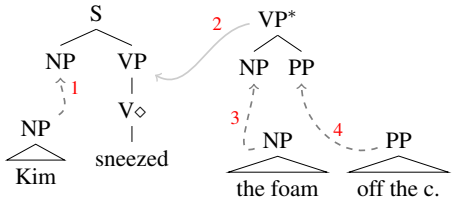

Figure 8: PLCxTAG derivation of (2-b)

tive constructions such as (2-c) and (2-d) would look very similar to the two derivations in Fig. 7 (where the object NP is already introduced with the verb) and Fig. 8 (where the object NP is introduced via the extension), except that the result is an AP. Semantically, caused motion and resultative constructions differ as a matter of course.

Constructions with co-anchors. In the following, we will discuss analysis options for two examples involving co-anchors, (3-a) and (3-b).

(3) a. Kim elbowed his way through the crowd
b. Kim kicked the bucket

The respective complete elementary trees for the two verbal construction would be as in Fig. 9. Note, however, that (3-a) is not restricted to a single verb while 'kick the bucket' is a fixed idiomatic expression. Concerning the latter, it can also have a literal meaning, but our analysis here is about the idiomatic meaning, to which we assign an independent elementary tree with co-anchors.

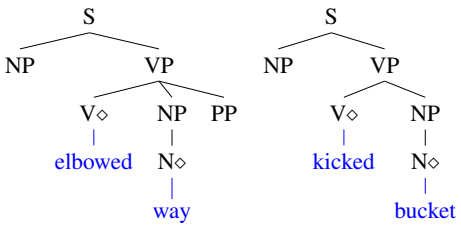

Figure 9: Complete elementary trees for the multi-anchored constructions in (3-a) and (3-b)

If we assume a strictly incremental derivation with prediction trees whenever words cannot be connected yet, we could choose an analysis as in Fig. 10. Step 5 in this case is special since it not only verifies the predicted NP tree but also reanalyzes its substitution (operation 3 in this derivation) as a substitution that is an extension. This latter is something that is not yet covered by the above definition of PLCxTAG but that could be added.

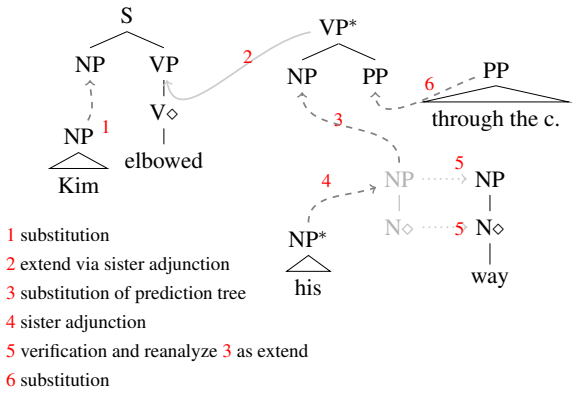

Figure 10: PLCxTAG derivation of (3-a) with verifcation and reanalyze as extension

The difficulty here comes from the fact that 'his' has to be attached before seeing 'way', a difficulty that could be avoided with a delay for this attachment. In general, it might be justified to adopt a delay for all cases of functional operator attachment. If we do this, we can actually adopt an analysis as in Fig. 11, where the extension tree anchored at 'way' extends the verbal tree.

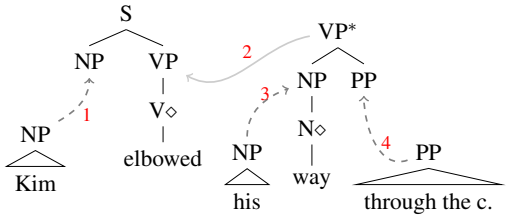

Figure 11: PLCxTAG derivation of (3-a) with delayed attachment of 'his'

Similarly, for (3-b), we could predict an NP tree at 'the', followed by a verification by a tree anchored by 'bucket', thereby reanalyzing the substitution of the prediction tree as an extension. Assuming, however, that the attachment of function words can be delayed, this complication is not needed. The corresponding derivation is given in Fig. 12.

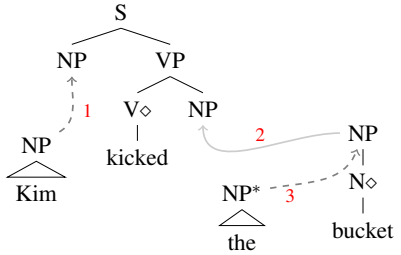

Figure 12: PLCxTAG derivation of (3-b) with delayed attachment for 'the'

5 Implementation

In this section, we will present preliminary results from the ongoing implementation of a PLCxTAG parser. The details of the parser architecture are still in development, but the core components are to be the lexicon, the (k-best) supertagger and a parallel parsing scheme with beam search. As the current stage of the implementation, we present the preliminary lexicon extraction and a supertagger.

Lexicon extraction. To extract the lexicon automatically, we used the Sections 00–24 from the Wall Street Journal portion of the Penn Treebank (Taylor et al., 2003). The extraction procedure is similar to those previously used for LTAG extraction (Xia et al., 2000; Chiang, 2000; Demberg et al., 2013): Trees in the PTB were preprocessed to suit our linguistic analysis and nodes were marked as head, argument and adjunct, using a modified version of the head and argument/adjunct rules from Collins (1999, 1997).

The preprocessing of the trees consists of five steps, three of which were conducted before marking the nodes, and the remaining two were performed afterwards.

Firstly, to be surface-oriented, (a) we deleted all null elements (Bies et al., 1995), including traces and PRO. Secondly, (b) we collapsed unary branches that appear due to the previous step, while retaining those which are already present in the original PTB. Thirdly, (c) we relabeled the part of speech tags of auxiliaries 'have', 'be' and 'do' as AUX, which are originally labeled as full verb, because all auxiliaries including those should be labeled as adjunct.

Then, we annotated each node of the trees according to the head/argument/adjunct rules described in Appendix A, making use of the tags including function tags.

At this point, (d) we reduced the tagset by removing function tags and merging some of the tags used in the PTB (cf. Appendix B). This reduced the number of tags from 71 to 36, which would in turn reduce the number of supertags and thus potentially improve the efficiency of the supertagger training as well as the performance of the resulting model. Then, (e) we collapsed the tree branches if the label of the parent node is identical to that of the head child node and no other children nodes are labeled as argument. This is because in CxTAG sister adjunction is used to attach adjuncts directly to the head phrase, without introducing new branches.

Fig. 13 illustrates the procedure with an example from the PTB. First, (a) the * (PRO) under -NONE- is removed, and then (b) the resulting unary branch from S to VP is collapsed. According to the marking rules, all nodes (except the root) are labeled as H(ead), C(omplement for argument) and A(djunct). Finally, (d) the tagset is reduced, where function tags like -SBJ are removed, NNP is modified along with other subcategories of noun to N(oun) and VB and VBD are merged into V(erb). Then, (e) the VP above TO and its head child labeled as VP are collapsed, since the other child is an adjunct. The result is the middle tree in Fig. 13.

Then elementary trees were extracted based on the annotation in a bottom-up fashion. Basically, the tree is to be split at the nodes labeled as C or containing children labeled as A (cf. the third tree in Fig. 13).

The elementary trees in this version are without co-anchors, excluding some well-known constructions like way-construction. Also, extension trees and hence the network of constructions are not covered yet. For those, we would need further extraction procedures to combine or decompose supertags obtained so far, depending on the statistics of the entire treebank.

In addition to supertags, we also extracted a sequence of prediction trees for each sentence in the data that the parser has to predict when processing it by computing the connection path (Demberg et al., 2013; Demberg-Winterfors, 2010) to check for each pair of adjacent words if there are some intervening nodes belonging to supertags to be anchored by subsequent words. At this point, the delay mechanism is not implemented yet, limiting the coverage to 33466 sentences out of 49208.

Our current lexicon extraction on the PTB yields 2663 different supertags, out of which 1293 are also used as prediction trees.

Supertagging. The supertagger is implemented via fine-tuning GPT-2 using multi-task learning with two classification heads, returning a pair of prediction tree (possibly none) and supertag for each word. We trained the model on Sections 02–21 for five epochs and evaluated it on Section 23. The data consists of a sequence of pairs of prediction tree/none and supertag. For more details see Appendix C.

The per-word accuracy of the supertagger after training is 0.91 and 0.79 for prediction trees

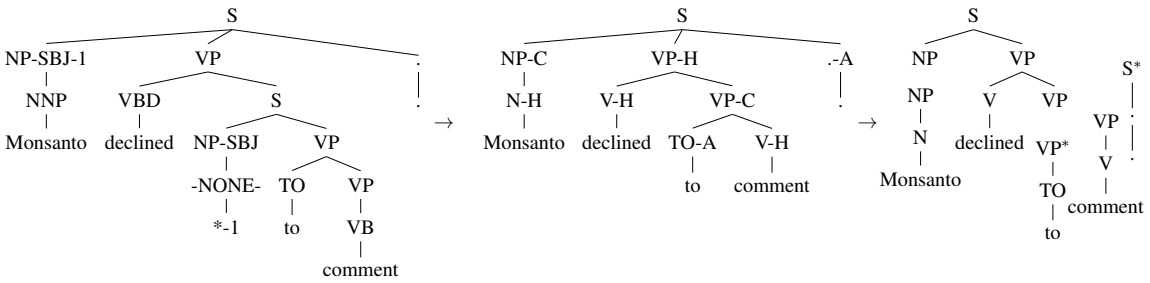

Figure 13: Sample extraction of supertags from a PTB tree

and supertags, respectively. Note that the accuracy for prediction trees looks better than it actually is, since for most words, the prediction is none, i.e., always predicting none could already yield an accuracy of about 0.80. For supertags, our accuracy is below the state-of-the-art for LTAG supertagging (for instance, Bladier et al., 2019, achieved 0.81 on French, which is usually harder than English), but not comparable to standard supertagging results because (for the sake of psycholinguistic plausibility) we employed a generative incremental LM as basis while Bladier et al. (2019) used a bidirectional model. Overall, the scores we achieved with these first experiments are quite promising.

6 Discussion and Conclusion

Summary. In this paper, we presented an alternative formalization of construction grammar guided by linguistic and psycholinguistic plausibility.

In particular, incremental extension based on frequencies of constructions is a novel way to formalize the inheritance and underspecification under usage-based tenets: The selection of schematic constructions is distributed over multiple words, facilitated by the the network of constructions and reflecting the predictability of constructions at each point in the incremental derivation.

Also, the results of our preliminary implementation of PLCxTAG, extracted lexicon and the supertagger, serve as a proof of concept. We are hopeful that future implementation of PLCxTAG will pave the way for quantitative psycholinguistic evaluation of the tenets of construction grammar.

Future directions. We are currently building a PLCxTAG parser which we will use to quantify the processing difficulty in a way similar to (Demberg et al., 2013; Demberg-Winterfors, 2010) for comparison with reading time data.

To this end, we are in the process of modifying the extraction and supertagging implementation to include extension trees and a delay mechanism, as well as designing the parallel parsing scheme that decides how to combine the trees returned by the supertagger. Concerning extension trees, the idea is to start with the supertags extracted in the way proposed here and train our supertagger on it. Based on the predicted supertags, we will then decompose some of the extracted gold supertags into smaller supertags and extension trees.

For psycholinguistic evaluation, we plan to use a corpus annotated with reading time data such as that presented in Frank et al. (2013) and evaluate along the lines of Mielczarek et al. (2025).

In addition, there are some aspects of the formalism that might require further discussion and improvement. For instance, we have yet to see which of the strategies sketched in Section 5 works better for constructions with co-anchors. In this context, the evaluation on psycholinguistic data will be taken into consideration. Also, there are some syntactic phenomena beyond the current formalization. For example, the use of only substitution and sister adjunction restricts the generative capacity of PLCxTAG in such a way that phenomena of long-distance dependencies cannot be adequatly treated. Second, we did not explicitly model semantic representation of constructions. Note, however, that our supertagger produces semantic representations of lexical anchors and, implicitly in its activation vectors, also of schematic constructions.

Finally, we are planning to extend PLCxTAG to other languages, in particular German and French, where we already have experience with TAG-based supertag extraction (Bladier et al., 2019). Ideally, in the long run, we would like to apply the framework also to a typologically broader set of languages such as Japanese.

Acknowledgements

We would like to thank three anonymous reviewers for their valuable and helpful feedback.

References

Srinivas Bangalore and Aravind K. Joshi, editors. 2010. *Supertagging: Using Complex Lexical Descriptions in Natural Language Processing*. MIT Press, Cambridge, MA.

Giulia M L Bencini and Adele E Goldberg. 2000. The contribution of argument structure constructions to sentence meaning. *J. Mem. Lang.*, 43(4):640–651.

Benjamin K Bergen and Nancy Chang. 2005. Embodied construction grammar in simulation-based language understanding. In *Constructional Approaches to Language*, pages 147–190. John Benjamins Publishing Company, Amsterdam.

Katrien Beuls and Paul Van Eecke. 2023. Fluid construction grammar: State of the art and future outlook. In *Proceedings of the First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)*, pages 41–50, Washington, D.C. Association for Computational Linguistics.

Ann Bies, Mark Ferguson, Karen Katz, and Robert MacIntyre. 1995. Bracketing guidelines for treebank II style Penn treebank project. Technical report, Linguistic Data Consortium.

Tatiana Bladier, Jakub Waszczuk, Laura Kallmeyer, and Jörg Janke. 2019. From partial neural graph-based LTAG parsing towards full parsing. *Computational Linguistics in the Netherlands Journal*, 9:3–26.

Hans C. Boas and Ivan Sag, editors. 2012. *Sign-Based Construction Grammar*. CSLI Publications, Stanford.

Jonathan R Brennan, Edward P Stabler, Sarah E Van Wagenen, Wen-Ming Luh, and John T Hale. 2016. Abstract linguistic structure correlates with temporal activity during naturalistic comprehension. *Brain Lang.*, 157-158:81–94.

John E. Bryant. 2008. *Best-Fit Constructional Analysis*. Ph.D. thesis, University of California at Berkeley.

Joan Bybee. 2010. *Language, Usage and Cognition*. Cambridge University Press, Cambridge.

Nancy Chang. 2008. *Constructing grammar: A computational model of the emergence of early constructions*. Ph.D. thesis, University of California at Berkeley.

David Chiang. 2000. Statistical parsing with an automatically-extracted Tree Adjoining Grammar. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 456–463, Hong Kong. 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 Processing*. Ph.D. thesis, University of Pennsylvania.

Benoit Crabbé, Denys Duchier, Claire Gardent, Joseph Le Roux, and Yannick Parmentier. 2013. XMG: eXtensible MetaGrammar. *Computational Linguistics*, 39(3):591–629.

Katherine A DeLong, T Urbach, and M Kutas. 2005. Probabilistic word pre-activation during language comprehension inferred from electrical brain activity. *Nat. Neurosci.*, 8:1117–1121.

Vera Demberg and Frank Keller. 2019. Cognitive models of syntax and sentence processing. In Peter Hagoort, editor, *Human Language: From Genes and Brains to Behavior*, pages 293–312. The MIT Press.

Vera Demberg, Frank Keller, and Alexander Koller. 2013. Incremental, predictive parsing with psycholinguistically motivated Tree-Adjoining Grammar. *Computational Linguistics*, 39(4):1025–1066.

Vera Demberg-Winterfors. 2010. *A Broad-Coverage Modelof Prediction in Human Sentence Processing*. Ph.D. thesis, University of Edinburgh.

Holger Diessel. 2023. *The Constructicon: Taxonomies and Networks*. Elements in Construction Grammar. Cambridge University Press, Cambridge.

Jonathan Dunn. 2017. Computational learning of construction grammars. *Language and Cognition*, 9(2):254–292.

Jerome A Feldman. 2022. Advances in embodied construction grammar. In *Benjamins Current Topics*, pages 147–167. John Benjamins Publishing Company, Amsterdam.

Stefan L. Frank, Irene Fernandez Monsalve, Robin L. Thompson, and Gabrielle Vigliocco. 2013. Reading time data for evaluating broad-coverage models of English sentence processing. *Behavior Research Methods*, 45(4):1182–1190.

Adele E Goldberg. 1995. *Constructions: A Construction Grammar Approach to Argument Structure*. University of Chicago Press.

Adele E Goldberg. 2003. Constructions: a new theoretical approach to language. *Trends Cogn. Sci.*, 7(5):219–224.

Aravind K Joshi, Leon S Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *J. Comput. Syst. Sci.*, 10(1):136–163.

Laura Kallmeyer and Rainer Osswald. 2013. Syntax-driven semantic frame composition in Lexicalized Tree Adjoining Grammars. *Journal of Language Modelling*, 1(2):267–330.

Yuki Kamide, Gerry T M Altmann, and Sarah L Haywood. 2003. The time-course of prediction in incremental sentence processing: Evidence from anticipatory eye movements. *J. Mem. Lang.*, 49(1):133–156.

Lars Konieczny. 1996. *Human sentence processing: a semantics-oriented parsing approach*. Ph.D. thesis, University of Freiburg.

Lars Konieczny. 2000. Locality and parsing complexity. *Journal of Psycholinguistic Research*, 29(6):627–645.

Ronald W. Langacker. 1987. *Foundations of Cognitive Grammar: Volume I: Theoretical Prerequisites*. Stanford University Press, Stanford, CA.

Timm Lichte and Laura Kallmeyer. 2017. Tree-Adjoining Grammar: A tree-based constructionist grammar framework for natural language understanding. In *The AAAI 2017 Spring Symposium on computational construction grammar and natural language understanding*, pages 205–212, Stanford, CA.

Timm Lichte and Simon Petitjean. 2015. Implementing semantic frames as typed feature structures with XMG. *Journal of Language Modelling*, 3(1):185–228.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330. Special Issue on Using Large Corpora: II.

Lukas Mielczarek, Timothée Bernard, Laura Kallmeyer, Katharina Spalek, and Benoit Crabbé. 2025. Modelling expectation-based and memory-based predictors of human reading times with syntax-guided attention. In *Proceedings of BriGAP-2*, Düsseldorf, Germany. Association for Computational Linguistics.

Stefan Müller. 2017. Head-driven phrase structure grammar, sign-based construction grammar, and fluid construction grammar: Commonalities and differences. *Constructions and Frames*, 9(1):139–173.

Ulrike Padó. 2007. *The integration of syntax and semantic plausibility in a wide-coverage model of human sentence processing*. Ph.D. thesis, Saarland University.

Florent Perek. 2025. Behavioral evidence and experimental methods. In Mirjam Fried and Kiki Nikiforidou, editors, *The Cambridge Handbook of Construction Grammar*, Cambridge Handbooks in Language and Linguistics, page 196–219. Cambridge University Press.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Brian Roark, Asaf Bachrach, Carlos Cardenas, and Christophe Pallier. 2009. Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 324–333, Singapore. Association for Computational Linguistics.

Miloš Stanojević, Jonathan R Brennan, Donald Dunagan, Mark Steedman, and John T Hale. 2023. Modeling structure-building in the brain with CCG parsing and large language models. *Cogn. Sci.*, 47(7):e13312.

Adrian Staub and Charles Clifton, Jr. 2006. Syntactic prediction in language comprehension: evidence from either...or. *J. Exp. Psychol. Learn. Mem. Cogn.*, 32(2):425–436.

Luc Steels. 2017. Basics of fluid construction grammar. *Constructions and Frames*, 9(2):178–225.

Patrick Sturt and Vincenzo Lombardo. 2005. Processing coordinated structures: incrementality and connectedness. *Cogn. Sci.*, 29(2):291–305.

Michael K. Tanenhaus, Michael J. Spivey-Knowlton, Kathleen M. Eberhard, and Julie C. Sedivy. 1995. Integration of visual and linguistic information in spoken language comprehension. *Science*, 268(5217):1632–1634.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn Treebank: An overview. In Anne Abeillé, editor, *Treebanks. Building and Using Parsed Corpora*, pages 5–22. Springer, New York.

Michael Tomasello. 2005. *Constructing a Language. A Usage-Based Theory of Language Acquisition*. Harvard University Press, Cambridge, MA.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M Rush. 2019. HuggingFace's transformers: State-of-the-art natural language processing. *arXiv [cs.CL]*.

Fei Xia, Martha Palmer, and Aravind Joshi. 2000. A uniform method of grammar extraction and its applications. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics* -, Morristown, NJ, USA. Association for Computational Linguistics.

XTAG Research Group. 1998. A Lexicalized Tree Adjoining Grammar for English. Technical report, University of Pennsylvania, Institute for Research in Cognitive Science.

A Appendix: Annotation of nodes as head, argument and adjunct

As is the case in previous attempts to extract LTAG from the PTB, we exploited the original PTB tags including function tags to mark nodes of trees as head, argument and adjunct.

A.1 Head rules

For the identification of heads, we followed the general procedure described in Collins (1999), where two head percolation tables are used, one for most tags and another for NP.

Still, we have modified both tables (cf. Tables 1 and 2). In particular, three major changes were made to the table for most tags.

Firstly, MD, TO and IN have been removed from the head candidates, since modal auxiliaries, the infinitive marker "to" and complementizers (labeled as IN along with prepositions) are to be adjuncts.

Secondly, -PRD is added as the candidate, since it indicates the existence of accompanying copulative verb like 'be' or 'seem'. In those cases, these verbs are to be adjuncts, even though they are labeled as full verb. That is why -PRD is placed higher in priority than tags for full verbs.

Thirdly, WHNP, WHPP, WHADVP, WHADJP and DT are removed from the candidate list for SBAR.

A.2 Argument/adjunct rules

After annotating the heads, we marked the remaining nodes as either argument or adjunct. Our rules for arguments and adjuncts are inspired by Collins (1997), but there are important changes to the original procedure.

Collins (1997) marks only the following as argument, while marking all else as adjunct:

(a) 1. NP/SBAR/S under S
2. NP/SBAR/S/VP under VP
3. S under SBAR
if without any of the following function tags: -ADV, -VOC, -BNF, -DIR, -EXT, -LOC, -MNR, -TMP, -CLR and -PRP

(b) the first child following the head under PP

This procedure, however, is highly problematic for numerous cases of coordination (especially when no CC or CONJP is involved) and for PP nodes with three or more children, as is exemplified in Fig. 14. In the left-hand side example, two

Parent	From	Priority list
ADJP	L	NNS QP NN $ ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB
ADVP	R	RB RBR RBS FW ADVP TO CD JJR JJ IN NP JJS NN
CONJP	R	CC RB IN
FRAG	R	
INTJ	L	
LST	R	LS :
NAC	L	NN NNS NNP NNPS NP NAC EX $ CD QP PRP VBG JJ JJS JJR ADJP FW
PP	R	IN TO VBG VBN RP FW
PRN	L	
PRT	R	RP
QP	L	$ IN NNS NN JJ RB DT CD QP JJR JJS
RRC	R	-PRD VP NP ADVP ADJP PP
S	L	-PRD VP S SBAR ADJP UCP NP
SBAR	L	S SQ SINV SBAR FRAG
SBARQ	L	SQ S SINV SBARQ FRAG
SINV	L	-PRD VBZ VBD VBP VB VP S SINV ADJP NP
SQ	L	-PRD VBZ VBD VBP VB VP SQ
UCP	R	
VP	L	-PRD VBD VBN VBZ VB VBG VBP VP ADJP NN NNS NP
WHADJP	L	CC WRB JJ ADJP
WHADVP	R	CC WRB
WHNP	L	WDT WP WP$ WHADJP WHPP WHNP
WHPP	R	IN TO FW

Table 1: Head table for most phrasal tags, the 2nd column gives the search order (starting from L(eft) or R(ight))

From	Candidate list
R	NN
L	NNP NNPS
R	NNS NX JJR PRP
L	NP
R	$ ADJP PRN
R	CD
R	JJ JJS RB QP

Table 2: Head table for parent tag NP

S children are coordinated by a semicolon labeled as :, where the first S is already labeled as head due to the head rule described in Table 1. In this case, the latter S would be marked as argument of the former, which it is not. The second example shows an instance where the annotator placed D and N immediately below PP without intermediate NP, resulting in D being marked as argument and N as adjunct.

Figure 14: Problematic examples from the PTB.

Therefore, for (a), when candidate nodes are coordinated, we only choose the left-most one, and for (b), we decided to use finer-grained conditions, depending on the number of PP's children.

In addition to these, we decided to mark all the nodes with some function tags like -SBJ as argument.

The resulting rules are:

(a) 1. NP/SBAR/S under S
2. NP/SBAR/S/VP under VP
3. S under SBAR

if without any of the following function tags: -ADV, -VOC, -BNF, -DIR, -EXT, -LOC, -MNR, -TMP, -CLR and -PRP

&

i. if not coordinated

or

ii. if the left-most coordinated element

(b) 1. the non-head child under a PP with two children
2. the first NP child under a PP with three or more children

(c) nodes with one of the following function tags: -DTV, -BNF, -LGS, -PUT, -SBJ, -CLF and -CLR

B Appendix: Tagset reduction

Tagset reduction was done by collapsing tags according to Tables 3 and 4.

Original tags	Reduced tag
JJ JJR JJS	A
RB RBR RBS WRB	Adv
DT PDT WDT PRP$ WP$	D
CD NN NNS NNP NNPS PRP WP EX $ #	N
AUX MD VB VBP VBZ VBN VBD VBG	V
Other POS tags	(unchanged)

Table 3: Tagset reduction for POS tags

Original tags	Reduced tag
ADJP WHADJP	AP
ADVP WHADVP	ADVP
NP NAC NX QP WHNP	NP
PP WHPP	PP
S SQ SBAR SBARQ SINV	S
Other phrasal tags	(unchanged)

Table 4: Tagset reduction for phrasal tags

C Appendix: Training of the supertagger

C.1 Model architecture

We modified GPT2PreTrainedModel (Radford et al., 2019) from the transformers library (Wolf et al., 2019) by adding the second linear classification head. The overall loss function was the mean of two cross entropy functions, one for each classifier.

C.2 Hyperparameters used in the training

We used the Trainer class from transformers library to train the model. See Table 5 for values chosen for the hyperparameters used in the training.

hyperparameter	value
learning rate	2e-05
number of epochs	5
weight decay	0.01
train batch size	8
evaluation batch size	8
seed	42
betas for ADAMW	(0.9,0.999)
epsilon for ADAMW	1e-08

Table 5: Hyperparameters of supertagging