From Dependency to CCG to Incremental CCG: Approaches for Flexible Word Order in Turkish

Özge Bakay[1,a], Oğuz Kerem Yıldız[2,b], Rajesh Bhatt[1,c], Brian Dillon[1,d], Olcay Taner Yıldız[2,e]
[1]University of Massachusetts Amherst, [2]Özyeğin University
[a]obakay@umass.edu, [b]oguzkeremyildiz@gmail.com,
[c]bhatt@umass.edu, [d]bwdillon@umass.edu, [e]olcay.yildiz@ozyegin.edu.tr

Abstract

Combinatory Categorial Grammar (CCG), a lexicalized formalism known for its flexible constituency, is well-suited for modeling head-final languages with flexible word order like Turkish. Building on Kuzgun et al. (2023), we first develop a Turkish CCG lexicon by automatically inducing categories from a dependency treebank. By leveraging standard and extended operations tailored to Turkish syntax, our parser achieves a robust coverage of 92.5%. Furthermore, we introduce the first (partially) incremental, left-to-right CCG parser for Turkish, designed to facilitate the immediate integration of words into the evolving representation. Finally, we present an example experiment showing that CCG parsers can model psycholinguistic evidence for extra processing costs associated with arguments in non-canonical positions, via the frequency of order-reversing operations. These findings provide evidence that CCG offers a cognitively plausible framework for modeling real-time processing in languages like Turkish.

1 Introduction

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism that maps syntactic categories to functional semantic types, allowing for a transparent interface between form and meaning (Steedman, 2000). Its incremental variants are particularly useful for modeling the word-by-word human sentence processing and real-time computational tasks (Stanojević et al., 2023). In the context of Turkish, previous research has developed CCG lexicons using both morphemic, i.e., categories for sub-word units (Bozşahin, 2002; Çakıcı, 2008), and lexicalist approaches, i.e., categories at the word level (Çakıcı, 2008; Hoffman, 1995; Kuzgun et al., 2023). CCG seems particularly well-suited for Turkish, a language with flexible word order, because its flexible constituency enables connected representations across varying structures.

CCGbanks are commonly created by converting treebanks into CCG derivations (e.g., English: Hockenmaier and Steedman 2007, Italian: Bos et al. 2009, Japanese: Uematsu et al. 2015) or by automatically inducing CCG lexicons using Universal Dependency (UD) relations (e.g., Hindi: Ambati et al. 2018, Telugu: Kumari and Rao 2015, Turkish: Çakıcı 2008; Kuzgun et al. 2023). These induction algorithms generate robust, lexicalized grammars suitable for statistical parsing.

In this work, we refine Kuzgun et al.'s (2023) algorithm for inducing a Turkish CCG lexicon from a UD standardized dataset. We modify their induction algorithm to ensure stricter alignment with the CCG framework and capture Turkish word order. We then present a novel methodology for creating connected, grammatical CCG derivations from this lexicon using both standard and extended CCG operations. Furthermore, we present, to our knowledge, the first incremental CCG parsing algorithm for Turkish, i.e., a parser that is forced to operate in a left-to-right, word-by-word fashion. Lastly, we suggest that CCG parsers may serve as cognitive models of real-time sentence processing. As an example, we model the empirical observation for an increased processing difficulty with arguments in non-canonical positions using our incremental CCG parser.

The rest of this paper is structured as follows: Section 2 provides an overview of the CCG framework; Section 3 outlines the properties of Turkish word order and details our approach in handling some cases related to those; Section 4 presents the current study, including a critical review of Kuzgun et al.'s (2023) CCG lexicon, the refinements that we make on it, the standard and incremental CCG parsers that we develop and their performance, and an experiment to exemplify how CCG parsers can be used to model psycholinguistic data; Section 5 summarizes the key findings.

2 Combinatory Categorial Grammar

CCG is a mildly-context-sensitive formalism that represents grammatical structures by combining basic linguistic categories (Steedman, 2000). Unlike traditional phrase structure grammars, CCG employs a lexicalized approach, using a small set of syntactic and semantic types along with combinatory rules to generate complex structures. A key feature of CCG is that semantic representation is directly built from surface syntactic constituents, without relying on intermediate-level representations. Each word in a sentence is assigned a syntactic category, and a constrained set of operations specify how adjacent categories are merged to form larger units. CCG is especially well-suited for modeling languages with flexible word order such as Turkish as it allows constituents to combine in varying orders and maintain a connected representation.

CCG categories are specified in the lexicon and may be either atomic (e.g., NP, for an argument) or complex (e.g., S\NP, for a function). Functions define a functor-argument relationship, with the argument given on the right, the resulting category on the left, and the direction of the argument encoded via a forward (/) or backward slash (\).

Table 1 lists the standard CCG operations for merging categories, and Table 2 illustrates these operations through derivations of Turkish sentences. While application and composition define functor-argument and functor-functor relations, type-raising turns an argument into a function that seeks the category it serves as an argument for (e.g., subject NP turned into S/(S\NP)). This operation is mainly adopted for case-marked arguments as it captures the role of morphological case in determining the syntactic position and the head of an argument. While the two crossed composition operations in Table 1 are *order-reversing*, i.e., combining functions in a different direction than their default, all other operations are *order-preserving*.

Beyond the standard CCG framework, there is a growing interest in developing *incremental* CCG parsers to better capture the dynamics of real-time, word-by-word language processing (Ambati et al., 2015; Demberg, 2012; Hefny et al., 2011; Stanojević and Steedman, 2019). Humans interpret linguistic input incrementally by building partial syntactic and semantic representations word-by-word without waiting till the end of a phrase or sentence (Marslen-Wilson, 1973). CCG is particularly suitable to model human incremental processing due to its flexible constituency structure and powerful combinators enabling connected representations (Demberg, 2012; Ozaki et al., 2024; Stanojević et al., 2023). Furthermore, due to their efficiency in rapid linguistic processing, incremental parsers are appealing for real-time computational applications such as speech recognition (Chelba and Jelinek, 2000), dialogue systems (Stoness et al., 2004) and machine translation (Schwartz et al., 2011).

3 Word order in Turkish and its consequences for CCG

Turkish is an agglutinative, head-final language characterized by highly flexible word order. It is also a pro-drop language, allowing the main arguments to be omitted. When both the subject and object arguments are overt, all six permutations of subject (S), object (O) and verb (V) are grammatically permissible: SOV, SVO, OSV, OVS, VSO, and VOS (Erguvanlı, 1984; Göksel and Kerslake, 2005; Kornfilt, 2003; Kural, 1997; Özge et al., 2019).

Despite this flexibility, these orders occur with varying frequencies in naturalistic data, with SOV as the most frequent one (e.g., Slobin and Bever,

Rules	Category 1	Category 2	Notation	Result
Forward type raising	X		$\Rightarrow_{>T}$	T/(T\X)
Backward type raising	X		$\Rightarrow_{<T}$	T\(T/X)
Forward application	X/Y	Y	$\Rightarrow_{>}$	X
Backward application	Y	X\Y	$\Rightarrow_{<}$	X
Forward (serial) composition	X/Y	Y/Z	$\Rightarrow_{>B}$	X/Z
Backward (serial) composition	Y\Z	X\Y	$\Rightarrow_{<B}$	X\Z
Forward crossed composition	X/Y	Y\Z	$\Rightarrow_{>Bx}$	X\Z
Backward crossed composition	Y/Z	X\Y	$\Rightarrow_{<Bx}$	X/Z
Coordination	X conj	X	$\Rightarrow_{\emptyset}$	X

Table 1: Standard CCG operations (from Steedman, 2000).

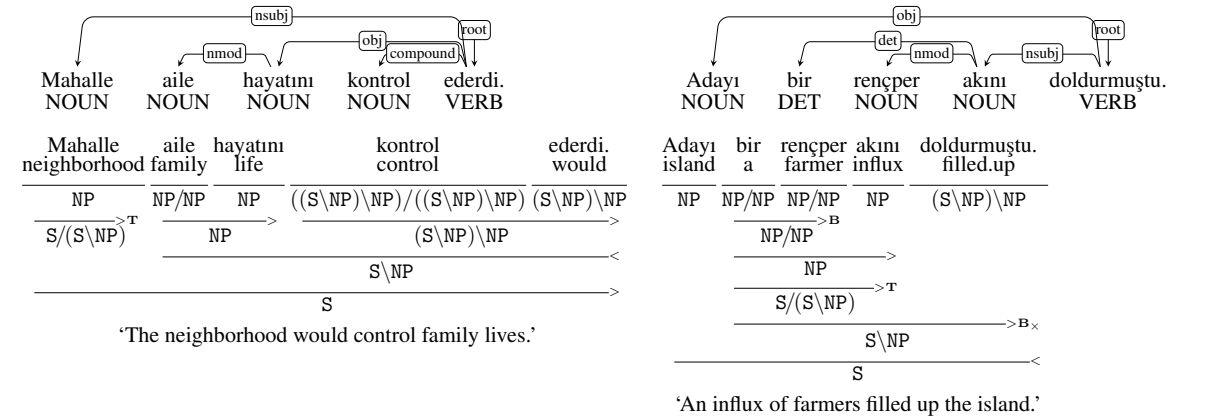


Table 2: CCG derivations with standard operations for Turkish sentences in canonical word order (adapted from the KeNet corpus), along with their UD relations.

1982). SOV is also considered the canonical or unmarked word order in Turkish.

The word order flexibility extends beyond the primary arguments of a clause. Word order variation is also observed with adjuncts, which can appear in multiple preverbal and postverbal positions, as can be seen with the adverb *dün* ('yesterday') in (1a). Furthermore, Turkish allows for long-distance scrambling where constituents of embedded clauses are positioned outside of their original clause boundaries. This is illustrated in (1b), where the embedded subject *Berna* appears in the main clause.

(1) a. (Dün) ben (dün) Ali-ye (dün)
yesterday I Ali-DAT
rastla-dı-m (dün).
run.into-PST-1SG
'I ran into Ali yesterday.'

b. Berna-nın$_i$ Aylin [_$_i$ o kitab-ı
Berna-GEN Aylin that book-ACC
beğen-diğ-in-i] duy-muş.
like-PST-3SG.POSS-ACC hear-PST
'Aylin heard that Berna liked that book.'

The flexible word order of Turkish poses significant challenges for CCG parsing, as the CCG formalism traditionally defines categories based on a fixed linear position and employs direction-sensitive combinatory operations. Below we outline our methodology for addressing two word-order-related challenges in developing a Turkish CCGbank.

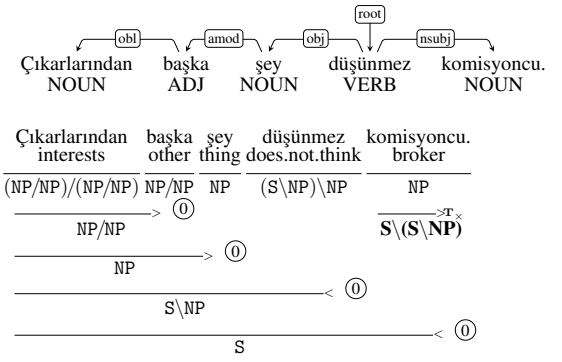
3.1 *Extraposition* for postverbal arguments

As noted above, the main arguments of a Turkish verb, namely subjects and objects, can appear in various positions, both preverbally and postverbally. Sentences with preverbal arguments, such as those exhibiting SOV and OSV orders, can be parsed using standard CCG operations (see the examples in Table 2). To comply with the verb-final structure of Turkish, in these examples, we define the category for transitive verbs as (S\NP)\NP, indicating that both the subject and object canonically appear to the left of the verb, though their relative internal order remains flexible (Bozşahin, 2000, 2002).

In contrast, sentences containing postverbal arguments, such as Example 1 in Table 3, cannot be parsed with standard CCG operations alone. In this example, once standard operations are applied, the verb phrase (S\NP) and the postverbal subject (NP) would remain unattached. This is due to the head-final nature of Turkish: Because arguments precede the verb in canonical SOV order, the verb's category is defined to seek its arguments from the left.

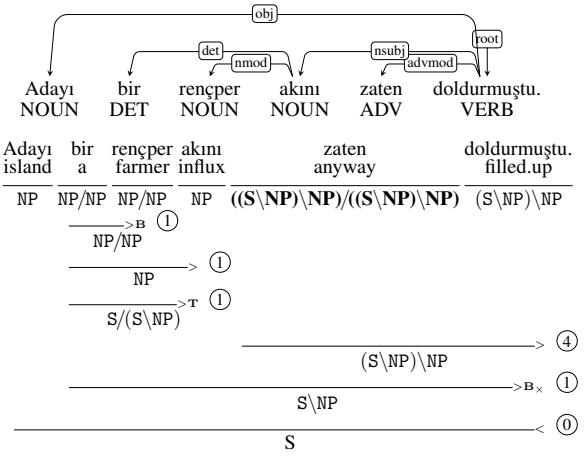
Two solutions have been proposed for this problem. One is Multiset-CCG introduced by Hoffman (1995), which eliminates directionality of arguments by not specifying their order with respect to the verb. In this approach, a transitive verb with two arguments is assigned the multiset category S|{NP, NP}, where '|' denotes an undirected slash, meaning the argument can appear on either side. While this allows postverbal arguments to be

1. Extraposition



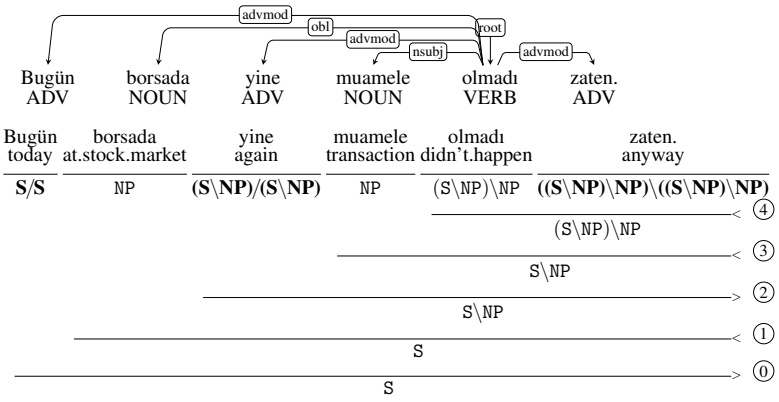
'The broker doesn't think of anything but his interests.'

3. Partial incrementality



'An influx of farmers filled up the island anyway.'

2. Flexible modifiers



'There wasn't a transaction at the stock market today anyway.'

Table 3: CCG derivations with extended operations for example Turkish sentences (adapted from the KeNet corpus), along with their UD relations. Derivations exemplify the *extraposition rule* in Example 1, flexible modifiers (given in bold) in Examples 2 and 3 and partial incrementality in Example 3. Circled numbers show the 'distance' from a strictly incremental merging operation, as defined for the *incrementality score* in Section 4.5.

parsed with standard operations, it fails to capture the head-finality of Turkish. Moreover, the lack of a direction for arguments would lead to a failure in type-raising operations which require a fixed subcategorization order (Bozşahin, 2002, 2019).

A second solution, proposed by Bozşahin (2000, 2002), relies on the assumption that Turkish lexicon encodes the two basic word orders of SOV and OSV. According to Bozşahin, parsing of sentences with postverbal arguments requires an additional rule, i.e., *forward extraposition*, as shown in (2). This rule type-raises postverbal constituents and reverses their direction relative to the verb.

(2) $X \Rightarrow_{>Tx} S\backslash(S\backslash X)$.

In our CCG parsers we use the extraposition rule to merge postverbal arguments, which is exemplified for the merging of the postverbal subject, *komisyoncu* 'broker', in Example 1 in Table 3.

3.2 Flexible CCG categories for modifiers

Modifiers may appear in different positions in Turkish, as illustrated in (1a) above. To account for this flexibility, we adopted a dynamic approach to category assignment due to two issues, i.e., direction of modifiers relative to their head and variations in the syntactic category being modified.

First, Turkish modifiers may precede or follow their heads. This is exemplified with the adverb *zaten* 'anyway' in Table 3, which precede its modi-

fied head in Example 2, and follows it in Example 3. Following Kuzgun et al. (2023), we derived the categories of modifiers from their heads to ensure that modifiers can be merged via functional application (e.g., X/X for a head of category X). Additionally, we dynamically determined the direction of the slash by assessing the modifier's linear position relative to its head, using a forward slash (/) for pre-head modifiers and a backward slash (\) for those are post-head.

Second, modifiers may intervene between a head and its various arguments, resulting in different syntactic categories being modified. For instance, in Example 2 in Table 3, the adverb *bugün* 'today' modifies the entire clause, including the subject (NSUBJ) and the oblique object (OBL). In contrast, the adverb *yine* 'again' in the same sentence modifies the verb phrase with the subject. To capture this, we assigned CCG categories based on the specific constituent being modified in a given sentence. For example, *bugün* is assigned a category that modifies the full sentence (i.e., S/S), whereas *yine* is assigned a category that modifies a partial verb phrase (i.e., (S\NP)/(S\NP)).

4 Present Study

4.1 The data

In the present study, we use two datasets to develop a CCGbank for Turkish. Our primary dataset, used for the development of induced categories and the evaluation of CCG derivations, is the KeNet WordNet corpus (Bakay et al., 2019a,b, 2021). As a secondary dataset, we use the Turkish Penn Treebank (Kuzgun et al., 2020). The Penn Treebank corpus was used to independently validate our induction algorithm and was not used during the development phase of the CCGbank itself.

The KeNet data consists of 18,687 sentences sourced from the Turkish National Dictionary, where they were provided as examples to support dictionary entries. The dataset primarily includes sentences from novels and daily speech, with some lines from poetry, though the exact distribution of sentences by genre is unavailable.

The Turkish Penn Treebank includes 9,560 sentences, which were Turkish translations of the original Penn Treebank sentences in English (Marcus et al., 1993). Manual annotations of dependency relations in the UD framework are available for both datasets (de Marneffe et al., 2021).

4.2 Kuzgun et al.'s (2023) Turkish CCG lexicon

Kuzgun et al. (2023) developed a CCG lexicon for Turkish by automatically inducing CCG categories to each word token using part-of-speech, dependency tag and head-dependent relations within the UD framework (Kuzgun et al., 2020). Kuzgun et al.'s work, as well as ours, followed a lexicalist instead of a morphemic approach because UD syntactic relations are defined on lexemes.

Their process proceeded linearly from the first token in a sentence. They automatically assigned CCG categories to each token by directly mapping dependency relations to functional CCG categories, e.g., X/X for modifiers with a head of category X. They also encoded the head-final structure of Turkish by using backward slashes (\) to introduce dependents in a head's category (e.g., S\NP for a verb that canonically follows its NP argument) and forward slashes (/) for dependents (e.g., NP/NP for a modifier that canonically precedes its head, NP). Because complex categories often contain unresolved variables (marked as X), the algorithm required multiple iterations to fully identify every element within a constituent. This iterative cycle continued until a complete category was assigned to every token in the sentence (for further details, see Kuzgun et al., 2023, pp. 5-7).

One significant limitation in Kuzgun et al. is the absence of a parsing algorithm to evaluate whether the categories in their lexicon can form complete CCG derivations. To assess the viability of their categories, we parsed the sentences in Kuzgun et al.'s lexicon using our CCG parsing algorithm (see Section 4.4 below). We found that only $\sim$5% of the sentences could be successfully derived, indicating that the induced categories were often incompatible with each other. This poor performance underscores the need for a more robust approach to CCG category assignment, which we address next.

4.3 Improving the Turkish CCG lexicon

Following Kuzgun et al. (2023), we induced CCG categories using the UD framework and POS information. Unlike the left-to-right algorithm in Kuzgun et al., we first identified the root of the dependency tree and assigned it a base category. Subsequently, we traversed the tree recursively to identify its dependents and their respective subdependents. This recursive process ensured that lexical categories reflect head-dependent relations.

Consider Example 1 in Table 3. In this example, first, the verb *does.not.think* was identified as the verbal root and assigned the category S. Then, this root was changed to the category (S\NP)\NP, as it has two dependent arguments, i.e., an object and a subject, marked with OBJ and NSUBJ, respectively. This was followed with assigning the NP category to the subject and object arguments. Then, the modifier of the object marked with AMOD was assigned the category NP/NP. Finally, the dependent of this modifier marked with OBL was assigned the category (NP/NP)/(NP/NP).

To enhance parsing coverage, we have implemented several systematic refinements to the lexicon established by Kuzgun et al. (2023). These modifications are mostly done to ensure that the induced categories accurately reflect the underlying dependency structures of Turkish. A comparative summary of these enhancements is provided below (for a comprehensive mapping of UD relations to CCG categories, see Appendix A).

Hierarchical Parentheses: We move from flat CCG categories to those with parentheses to reflect the hierarchical representation within constituents and to ensure an unambiguous order of operations (e.g., S/S\NP changed into S/(S\NP)).

Recursive Argument Count: We move from a static lookup of dependency labels for a limited number of arguments (i.e., NSUBJ, OBJ) to a recursive function, which wraps a head's category (typically S or NP) in \NP layers for every argument(-like) NP dependent on it (i.e., COMP, CSUBJ, IOBJ, NSUBJ, OBJ, OBL). This allows the head's category to grow dynamically based on the number of its dependents in a given sentence.

Flexible Categories for Modifiers: The previous induction algorithm assigned categories to various modifier(-like) relations (i.e., ACL, ADVCL, ADVMOD, AMOD, CASE, COMPOUND, CLF, CC, DET, FLAT, NMOD, NUMMOD, REPARANDUM) by duplicating the head's category using fixed forward slashes (e.g., NP/NP for an NP head). To account for flexible ordering of modifiers within a sentence, the current version dynamically determines the slash direction (/ or \) depending on the position of the modifier relative to its head in a given sentence. It also uses a function to strip the argument(-like) NP(s) from the head's category before copying, to remove those that appear in between the modifier and its head. This allows us to reflect the category syntactically modified (see Section 3.2 for details). While the earlier version used type-raising for certain head-modifier relations (i.e., CASE, MARK, our version treats these relations similar to modifiers.

Clausal Head Differentiation: While the previous version started with assigning the default S category to all heads (i.e., verbal and nominalized), the current version checks the category of the head and assigns NP as base to nominalized ones (i.e., XCOMP, CCOMP) and S to all others.

Generalized NP: While the earlier version used nominative case marking in the NP category for subjects and their modifiers (e.g., NP[nom]), this was inconsistent and limited to a single case marking. In our version, we remove [nom] to have a generalized NP category.

Removal of Punctuation: While the previous version assigned categories to punctuation, we remove punctuation before category assignment to avoid generating excessively complex categories.

Table 4 lists the ten most frequent categories in our CCG lexicon, which mostly associated with nominals (e.g., NP), modifiers (e.g., (NP/NP), (S/S)) and verbs (e.g., (S\NP), ((S\NP)\NP)) (also see Appendix B for the top 3 most frequenct CCG categories per word form).

CCG tag	Count	Frequency %
1. NP	41,490	27.99
2. NP/NP	26,198	17.67
3. S\NP	10,146	6.84
4. NP\NP	8,283	5.59
5. (NP/NP)/(NP/NP)	6,958	4.69
6. (S\NP)\NP	6,873	4.64
7. S/S	6,292	4.24
8. (S\NP)/(S\NP)	5,104	3.44
9. S	4,166	2.81
10. (S/S)/(S/S)	2,579	1.74
Total	118,089	79.65

Table 4: The most frequent 10 CCG categories in the KeNet dataset, with their count and frequency (%). Total CCG categories assigned = 148,249.

4.4 The Standard CCG Parser

Next we developed a novel *Standard CCG* parsing algorithm to derive sentences with the improved categories, using standard CCG operations and the extraposition rule. This algorithm operates by iteratively identifying and merging adjacent categories, systematically evaluating all possible combinatory sequences to explore every valid derivation path. Standard CCG parser successfully derived 17,280 out of 18,687 sentences (92.5%),

showing a substantial improvement over previous Turkish CCGbanks: Kuzgun et al.'s (2023) lexicon achieved only ∼5% coverage when using our Standard CCG parser, while Çakıcı's (2008) lexemic lexicon reached 65−69%. Our Turkish CCGbank's coverage is now comparable to those developed with similar induction algorithms for other languages, such as Hindi (96%; Ambati et al., 2018) and Telugu (99%; Kumari and Rao, 2015).

4.5 The Incremental CCG Parser

Standard CCG parsers, such as the one described above, rely on an exhaustive search to parse a sentence. However, it is a well-established fact that human language comprehension proceeds incrementally (Marslen-Wilson, 1973). That is, as comprehenders receive linguistic input sequentially from left to right, they construct partial representations step by step. This incrementality is observed crosslinguistically. For example, psycholinguistic evidence from head-final, SOV languages such as Turkish and Japanese demonstrates that human comprehenders do not wait for the final verb to integrate preceding words; instead, they interpret them incrementally and leverage various sources of information from them to generate real-time predictions (e.g., Bakay et al., 2026; Kamide et al., 2003; Özge et al., 2019).

In a CCG framework, an incremental parser attempts to combine its existing partial representation with the immediately adjacent constituent to its right at each parsing step. While strictly incremental parsers aim to adjoin categories sequentially from the leftmost constituent onward, such connected derivations cannot always be achieved within a standard CCG framework. This is exemplified in Example 3 in Table 3. To increase incrementality, various approaches have been proposed including *tree rotation* to incrementally convert left-branching to right-branching structures (Stanojević and Steedman, 2019), *revealing* to reconfigure the required information from an internal constituent (Ambati et al., 2015) or specialized operations to incrementally integrate categories that would otherwise be merged with a delay (Demberg, 2012).

As a first step towards creating a cognitively more plausible CCG parser for Turkish, we developed an incremental CCG parsing algorithm by strictly forcing combinatory operations to proceed in a left-to-right fashion. This constraint encourages the parser to integrate words into the active constituent representation at the earliest possible choice point. Operating via standard CCG combinators and the specialized extraposition rule, our incremental parser successfully derives 15,223 out of 18,687 sentences (81.4%) in the KeNet corpus.

To quantify the degree of incrementality of a derivation, we created an *incrementality score* based on the position of each combinatory operation relative to the leftmost constituent (for other examples, see Stanojević and Steedman, 2019). In standard CCG, a sentence of length n requires $n-1$ combinatory operations to achieve a full derivation. For each merging operation j, we identify the index of the leftmost constituent being merged, denoted as C_j, which represents the constituent's current position in an active list of constituents that shifts leftward as operations are performed (rather than original word indices in the string). The difference between this position and the leftmost possible index ($C_j - 1$) quantifies the 'distance' from a strictly incremental merge. We sum these differences for each merge and normalize this sum by the maximum possible sum of differences for a sentence of that length. This is shown in Equation (3).

$$\text{Incrementality score} = \frac{\sum_{j=1}^{n-1}(C_j - 1)}{\frac{(n-1)(n-2)}{2}} \quad (3)$$

Under this metric, a strictly incremental derivation, whereby every merge involves the constituent at index 1, results in a score of 0. Conversely, a fully non-incremental derivation yields a score of 1, reflecting the maximum delay in integrating the leftmost partial representation. For instance, the incrementality scores for the derivations in Examples 1, 2 and 3 in Table 3 are 0, 1 and 0.4, respectively. The mean incrementality score for the successfully derived sentences in our corpus is 0.182, suggesting that our *incremental* parser is only partially, rather than strictly, incremental.

Lastly, to test the generalizability of our parsing algorithms, we applied it to an external corpus that was not referred to during our CCGbank creation, i.e., the Turkish Penn Treebank (Kuzgun et al., 2020). Our standard and incremental parsing algorithms achieved satisfactory coverage (89.04% and 80.35% respectively), providing evidence for the generalizability of our approach.

4.6 Experiment: Modeling processing difficulty with non-canonical word orders using CCG parsers

Recent studies show that CCG-based parsing metrics improve predictions of incremental processing difficulty in self-paced reading (Ozaki et al., 2024) and neuroimaging studies (Brennan et al., 2016; Stanojević et al., 2023). Here we extend their use to predictions of whole-sentence processing difficulty. Specifically, we investigate whether they can predict differences in processing sentences in canonical vs. non-canonical word orders.

There is compelling evidence that noncanonical word orders are harder to process than canonical ones (verb-medial/verb-initial vs. verb-final in Turkish) (Sekerina 2003; for Turkish, see Kahraman and Hirose 2018; Kuribayashi 2009; Özge et al. 2013). Here we use CCG-based metrics to model these empirical findings on existing human data. To approximate the processing cost of non-canonical word orders, we quantify the proportions of order-reversing operations (i.e., crossed composition, extraposition) relative to the total number of operations within each derivation. This approach is based on the assumption that order-reversing operations reflect increased cognitive load, as they alter the canonical directionality of arguments or functions defined in the lexicon.

As Table 5 demonstrates, order-reversing operations occur less frequently in canonical, verb-final configurations, but increase substantially for non-canonical, verb-initial, and verb-medial word orders. Furthermore, canonical verb-final orders exhibit substantially lower incrementality scores than their non-canonical counterparts. Note that Table 5 does not include VSO and VOS word orders, as they are significantly scarcer in the corpus.

Word Order	N of Sentences	% of order-reversing in Standard CCG	% of order-reversing in Incremental CCG	Mean incrementality score
SV	4,276	.012	.012	.283
OV	3,845	.015	.013	.236
SOV	1,783	.01	.009	.375
OSV	179	.007	.006	.334
VS	149	.185	.179	.154
VO	125	.171	.165	.197
SVO	59	.127	.122	.178
OVS	33	.152	.129	.161

Table 5: Number of sentences, mean % of order-reversing operations and mean incrementality scores for the word orders in the Turkish KeNet corpus (total N=10,449).

To statistically evaluate these patterns, we compared the percentages of order-reversing operations and incrementality scores of non-canonical word orders against their canonical baselines, analyzing structures with one argument (SV) and two arguments (SOV) separately.

Across both the standard and incremental parsers, non-canonical word orders yielded significantly higher proportions of order-reversing operations than their canonical counterparts (t's $\geq$ 24.38), with the exceptions of OV and OSV in the incremental parser (t = 1.10 and t = -1.13, respectively). This indicates that a CCG-based metric that incorporates order-reversing operations can successfully model the processing difficulty associated with non-canonical word orders.

Notably, incrementality scores–where lower scores indicate increased incrementality–were significantly lower for non-canonical word orders compared to their canonical baselines (t's $\geq$ 2.40). The decreased incrementality with canonical word orders is likely due to words that precede or appear at a distance from their syntactic heads. Since preverbal arguments or adjuncts are not type-raised in the current parser, they remain unintegrated until their corresponding head is processed.

5 Discussion

In this work, we presented a comprehensive framework for developing a Turkish CCGbank, where CCG categories are induced from underlying dependency relations and evaluated using both standard and incremental CCG parsers. Our methodology specifically addresses the structural challenges inherent to Turkish, a head-final language characterized by highly flexible word order. By refining the category induction pipeline introduced by Kuzgun et al. (2023), we substantially improved the parsing coverage of the Standard CCG parser to 92.5%, marking the highest coverage reported for Turkish CCG parsing to date. Furthermore, we introduced the first incremental, left-to-right Turkish CCG parser. While this parser remains partially incremental at its current stage, the architectural approaches developed here provide a foundation that can be adapted to other free-word-order languages.

Beyond grammar development, this study establishes a bridge between computational parsing efficiency and human sentence processing. We demonstrated that CCG-based parsing metrics–specifically those operationalizing the proportion

of order-reversing combinatory operations–can effectively capture the increased processing costs associated with non-canonical word orders in human behavioral data. These findings suggest that CCG-derived metrics can serve as a viable proxy for the cognitive load induced by scrambling, offering promising new avenues for modeling real-time, human-like language processing within the CCG framework.

Finally, we introduced a novel incrementality score designed to quantify the degree of incrementality achieved within a derivation. Due to the structural constraints of our current incremental parser, this metric predicts lower incrementality scores for canonical, verb-final constructions. However, extensive psycholinguistic evidence demonstrates that human processing of canonical word orders in Turkish is incremental (Bakay et al., 2026; Özge et al., 2019). Consequently, a crucial next step for this parser is to incorporate mechanisms that facilitate the immediate, incremental integration of preverbal elements, such as arguments and adjuncts, before the final verb is encountered.

Limitations

While this work is a significant step towards developing a Turkish CCGbank, several limitations remain. First, our use of a generalized NP category does not reflect the rich morphological case system of Turkish. Empirical work shows that case markers are immediately used by human comprehenders to generate predictions (Kamide et al., 2003; Özge et al., 2019). Including case features in CCG categories could potentially enhance the parser's predictive accuracy and would be better to model the real-time processing in human data.

Second, while our methodology significantly improves parsing coverage for a Turkish CCG parser, our evaluation primarily measures the lexicon's viability rather than its structural accuracy. Furthermore, the current study does not quantify the degree of syntactic ambiguity; thus, it remains to be systematically verified whether the correct analysis is present and prioritized among all possible parser outputs. While our random manual checks suggest that our analyses are reliable, future work should employ more rigorous assessment of precision and ambiguity resolution, potentially via evaluations against a gold-standard dependencies in the original dependency treebank (see Ambati et al., 2018).

Third, our parser does not achieve full incrementality, because not every constituent is immediately integrated into the structure upon being encountered. This is evidenced by the less incremental derivations for verb-final word orders as shown in Table 5. Although we have generated an algorithm which forced constituent merging in a left-to-right fashion, further adjustments are needed for better incrementality. For example, while the current parser uses type-raising for subject arguments only, extending this rule to all preverbal argument-like NPs as suggested by Bozşahin (2019), or even to adjuncts, could significantly improve incrementality in head-final languages.

Fourth, while we used CCG parsing to model sentence-level processing difficulty, future research could implement these metrics at the word level. This would reveal whether such metrics can predict the location of processing difficulty as observed in human data, as demonstrated for English (Ozaki et al., 2024; Stanojević et al., 2023).

References

Bharat Ram Ambati, Tejaswini Deoskar, Mark Johnson, and Mark Steedman. 2015. An incremental algorithm for transition-based CCG parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 53–63, Denver, Colorado. Association for Computational Linguistics.

Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. 2018. Hindi CCGbank: A CCG treebank from the Hindi dependency treebank. *Language Resources and Evaluation*, 52(1):67–100.

Özge Bakay, Faruk Akkuş, and Brian Dillon. 2026. Hierarchical relations guide memory retrieval in sentence comprehension: Evidence from a local anaphor in Turkish. *Journal of Memory and Language*, 148:104747.

Özge Bakay, Özlem Ergelen, Elif Sarmış, Selin Yıldırım, Bilge Nas Arıcan, Atilla Kocabalcıoğlu, Merve Özçelik, Ezgi Sanıyar, Oğuzhan Kuyrukçu, Begüm Avar, and Olcay Taner Yıldız. 2021. Turkish WordNet KeNet. In *Proceedings of the 11th Global Wordnet Conference*, pages 166–174, University of South Africa (UNISA). Global Wordnet Association.

Özge Bakay, Özlem Ergelen, and Olcay Taner Yldz. 2019a. Integrating Turkish Wordnet KeNet to Princeton WordNet: The case of one-to-many correspondences. In *5th Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–5.

Özge Bakay, Özlem Ergelen, and Olcay Taner Yldz. 2019b. Problems caused by semantic drift in word-

net synset construction. In *4th International Conference on Computer Science and Engineering (UBMK)*, pages 1–5.

Johan Bos, Cristina Bosco, Alessandro Mazzei, and 1 others. 2009. Converting a dependency treebank to a categorial grammar treebank for Italian. In *Proceedings of the Eight international workshop on treebanks and linguistic theories (TLT8)*, pages 27–38.

Cem Bozşahin. 2000. Gapping and word order in Turkish. In *Proceedings of the 10th International Conference on Turkish Linguistics*, pages 58–66.

Cem Bozşahin. 2002. The combinatory morphemic lexicon. *Computational Linguistics*, 28(2):145–186.

Cem Bozşahin. 2019. Command and Order by Type Substitution: Another Way to Look at Word Order. In A. Sumru Özsoy, editor, *Word Order in Turkish*, pages 179–216. Springer Nature Switzerland.

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 and Language*, 157:81–94.

Ruket Çakıcı. 2008. *Wide-coverage parsing for Turkish*. Ph.D. thesis, University of Edinburgh.

Ciprian Chelba and Frederick Jelinek. 2000. Structured language modeling. *Computer Speech & Language*, 14(4):283–332.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Vera Demberg. 2012. Incremental derivations in CCG. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 11)*, pages 198–206.

Eser Emine Erguvanlı. 1984. *The function of word order in Turkish grammar*. University of California Press, Berkeley.

Aslı Göksel and Celia Kerslake. 2005. *Turkish: A comprehensive grammar*. Routledge, New York.

Ahmed Hefny, Hany Hassan, and Mohamed Bahgat. 2011. Incremental Combinatory Categorial Grammar and its derivations. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 96–108.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Beryl Ann Hoffman. 1995. *The computational analysis of the syntax and interpretation of 'free' word order in Turkish*. Ph.D. thesis, University of Pennsylvania.

Barış Kahraman and Yuki Hirose. 2018. Online comprehension of SOV and OSV sentences in Turkish with a supporting context. In *The Proceedings of 10th Workshop on Altaic Formal Linguistics*, pages 1–7.

Yuki Kamide, Gerry TM Altmann, and Sarah L Haywood. 2003. The time-course of prediction in incremental sentence processing: Evidence from anticipatory eye movements. *Journal of Memory and language*, 49(1):133–156.

Jaklin Kornfilt. 2003. Scrambling, subscrambling, and case in Turkish. In Simin Karimi, editor, *Word order and scrambling*, pages 125–155. Blackwell, Oxford.

B Venkata Seshu Kumari and Ramisetty Rajeshwara Rao. 2015. Improving Telugu dependency parsing using Combinatory Categorial Grammar supertags. In *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, volume 14, pages 1–10.

Murat Kural. 1997. Postverbal constituents in Turkish and the linear correspondence axiom. *Linguistic Inquiry*, pages 498–519.

Yu Kuribayashi. 2009. Structure and description of southwestern Turkic. In *CSEL Series 16*, Kyushu University.

Aslı Kuzgun, Neslihan Cesur, Bilge Nas Arcan, Merve Özçelik, Büşra Marşan, Neslihan Kara, Deniz Baran Aslan, and Olcay Taner Yıldız. 2020. On building the largest and cross-linguistic Turkish dependency corpus. In *Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–6.

Aslı Kuzgun, Oğuz Kerem Yıldız, and Olcay Taner Yıldız. 2023. A CCGbank for Turkish: From dependency to CCG. In *Proceedings of the 12th Global Wordnet Conference*, pages 205–213.

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

William Marslen-Wilson. 1973. Linguistic structure and speech shadowing at very short latencies. *Nature*, 244:522–523.

Satoru Ozaki, Aniello De Santo, Tal Lizen, and Brian Dillon. 2024. CCG parsing effort and surprisal jointly predict RT but underpredict garden-path effects. In *Society for Computation in Linguistics (SCiL 2024)*.

Duygu Özge, Aylin Küntay, and Jesse Snedeker. 2019. Why wait for the verb? Turkish speaking children use case markers for incremental language comprehension. *Cognition*, 183:152–180.

Duygu Özge, Theodoros Marinis, Deniz Zeyrek, and Umut Özge. 2013. Object-first orders do not pose a challenge during processing for Turkish speakers. In

Proceedings of the 8th Workshop on Altaic Formal Linguistics, pages 269–280.

Lane Schwartz, Chris Callison-Burch, William Schuler, and Stephen Wu. 2011. Incremental syntactic language models for phrase-based translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics.*, pages 620–631.

Irina A Sekerina. 2003. Scrambling and processing: Dependencies, complexity, and constraints. In S. Karimi, editor, *Word order and scrambling*, pages 301–324. Wiley Online Library, Malden, MA: Blackwell.

Dan I. Slobin and Thomas G. Bever. 1982. Children use canonical sentence schemas: A crosslinguistic study of word order and inflections. *Cognition*, 12(3):229–265.

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. *Cognitive science*, 47(7):e13312.

Miloš Stanojević and Mark Steedman. 2019. Ccg parsing algorithm with incremental tree rotation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 228–239.

Mark Steedman. 2000. *The syntactic process*. MIT press, Cambridge, MA.

Scott C Stoness, Joel Tetreault, and James Allen. 2004. Incremental parsing with reference interaction. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 18–25.

Sumire Uematsu, Takuya Matsuzaki, Hiroki Hanaoka, Yusuke Miyao, and Hideki Mima. 2015. Integrating multiple dependency corpora for inducing wide-coverage Japanese CCG resources. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 14(1):1–24.

Appendix A: UD-to-CCG inductions

UD relations	CCG categories for dependent	CCG categories for head
COMP, CSUBJ, IOBJ, NSUBJ, OBJ, OBL	NP	$X\backslash NP_i$; i = N of NPs
XCOMP, CCOMP	if null, set to NP	$NP\backslash NP_i$, $S\backslash NP_i$
ACL, ADVCL, ADVMOD, AMOD, CASE, COMPOUND, CLF, CC, DET, FLAT, NMOD, NUMMOD, REPARANDUM	$(X/NP_{i-k})/(X/NP_{i-k})$, $(X/NP_{i-k})\backslash(X/NP_{i-k})$, $(X\backslash NP_{i-k})/(X\backslash NP_{i-k})$, $(X\backslash NP_{i-k})\backslash(X\backslash NP_{i-k})$; k = N of NPs in the phrase modified	$(X/NP_i)/(X/NP_i)$, $(X/NP_i)\backslash(X/NP_i)$, $(X\backslash NP_i)/(X\backslash NP_i)$, $(X\backslash NP_i)\backslash(X\backslash NP_i)$
MARK, DISCOURSE, AUX, LIST, GOESWITH, FIXED, CONJ	X/X, X\X	X
ORPHAN, APPOS	NP/NP, NP\NP	if null, set to NP
VOCATIVE	S/S, S\S	if null, set to S
DISLOCATED, DEP	NP/S, NP\S	if null, set to S

Table 6: Summary of UD-to-CCG inductions in generating our CCG Lexicon.

Appendix B: Top 3 most frequent CCG categories for each Part-of-Speech (POS) tag

POS tag	Rank 1 Category	Count	%	Rank 2 Category	Count	%	Rank 3 Category	Count	%
ADJ	(NP/NP)	8,446	63.8	((NP/NP)/(NP/NP))	1,971	14.9	((NP/NP)\NP)	1,557	11.7
ADP	(NP\NP)	1,718	61.2	((S/S)\(S/S))	443	15.8	(((S\NP)/(S\NP))\((S\NP)/(S\NP)))	265	9.4
ADV	(S/S)	1,731	31.7	((S\NP)/(S\NP))	1,691	31.0	(NP/NP)	849	15.5
AUX	((S\NP)\(S\NP))	305	43.1	(((S\NP)\NP)\((S\NP)\NP))	128	18.1	(S\S)	111	15.7
CCONJ	(NP\NP)	1,097	41.4	(((NP/NP)\(NP/NP))/((NP/NP)\(NP/NP)))	408	15.4	(((S\NP)\(S\NP))/((S\NP)\(S\NP)))	328	12.4
DET	(NP/NP)	4,772	71.6	((NP/NP)/(NP/NP))	1,100	16.5	(S/S)	281	4.2
INTJ	(S/S)	68	21.4	((S\NP)/(S\NP))	54	17.0	(S\S)	26	8.2
NOUN	NP	34,955	71.6	(NP/NP)	9,582	19.6	(NP\NP)	4,286	8.8
NUM	(NP/NP)	792	50.1	((NP/NP)/(NP/NP))	462	29.2	(((NP/NP)/(NP/NP))/((NP/NP)/(NP/NP)))	141	8.9
PRON	NP	3,514	77.9	(NP/NP)	846	18.8	((NP/NP)/(NP/NP))	150	3.3
PROPN	NP	1,217	57.3	(NP/NP)	498	23.5	(NP\NP)	208	9.8
SCONJ	((S/S)\(S/S))	104	37.1	((S\NP)\(S\NP))	70	25.0	(((S\NP)\NP)\((S\NP)\NP))	46	16.4
VERB	(S\NP)	7,351	44.2	((S\NP)\NP)	6,141	36.9	S	1,915	11.5
X	(NP/NP)	42	45.7	NP	29	31.5	((S\NP)/(S\NP))	21	22.8

Table 7: Top three most frequent CCG categories per POS tag in the KeNet corpus with counts and relative proportions within each POS tag.