Learning to generate and `corr-` uh I mean *repair* language in real-time

Arash Eshghi*† Arash Ashrafzadeh*
*Heriot-Watt University, Edinburgh, United Kingdom
†AlanaAI, Edinburgh, United Kingdom
`{a.eshghi, aa2070}@hw.ac.uk`

Abstract

In conversation, speakers produce language *incrementally*, word by word, while continuously monitoring the appropriateness of their own contribution in the dynamically unfolding context of the conversation; and this often leads them to repair their own utterance on the fly. This real-time language processing capacity is furthermore crucial to the development of fluent and natural conversational AI. In this paper, we use a previously learned Dynamic Syntax grammar and the CHILDES corpus to develop, train and evaluate a probabilistic model for incremental generation where input to the model is a purely *semantic generation goal concept* in Type Theory with Records (TTR)[1]. We show that the model's output exactly matches the gold candidate in 78% of cases with a ROUGE-l score of 0.86. We further do a zero-shot evaluation of the ability of the same model to generate *self-repairs* when the generation goal changes mid-utterance. Automatic evaluation shows that the model can generate self-repairs correctly in 85% of cases. A small human evaluation confirms the naturalness and grammaticality of the generated self-repairs. Overall, these results further highlight the generalisation power of grammar-based models and lay the foundations for more controllable, and naturally interactive conversational AI systems.

1 Introduction

People process language incrementally, in real-time (see Crocker et al. (2000); Ferreira (1996); Kempson et al. (2016) among many others), i.e. both language understanding and generation proceed on a word by word rather than a sentence by sentence, or utterance by utterance basis. This real-time processing capacity underpins participant coordination in conversation (Gregoromichelaki et al., 2012, 2020) and leads to many characteristic phenomena such as split-utterances (Poesio and Rieser, 2010; Purver et al., 2009), mid-utterance feedback in the form of backchannels (Heldner et al., 2013) or clarification requests (Healey et al., 2011; Howes and Eshghi, 2021), hesitations, self-repairs (Schegloff et al., 1977) and more.

Language generation – our focus here – is just as incremental as language understanding: speakers normally do not have a fully formed conceptualisation or plan of what they want to say before they start articulating, and conceptualisation needs only to be one step ahead of generation or articulation (Guhe, 2007; Levelt, 1989). This is possible because speakers are able to continuously monitor the syntax, semantics, and the pragmatic appropriateness of their own contribution (Levelt, 1989) in the fast, dynamically evolving context of the conversation. In turn this allows them to pivot or correct themselves on the fly if needed, e.g. because they misarticulate a word, get feedback from their interlocutors (Goodwin, 1981), or else the generation goal changes due to the dynamics of the environment.

Real-time language processing is likewise crucial in designing dialogue systems that are more responsive, more naturally interactive (Skantze and Hjalmarsson, 2010; Aist et al., 2006), and are more accessible to people with memory impairments (Addlesee et al., 2019; Addlesee and Damonte, 2023; Nasreen et al., 2021). Despite this importance, relative to turn-based systems, it has received little attention from the wider NLP community; perhaps because it has deep implications for the architecture of such systems (Schlangen and Skantze, 2009; Skantze and Schlangen, 2009; Kennington et al., 2014), which make them much harder to build and maintain.

In this paper, we extend the work of Purver and Kempson (2004); Hough and Purver (2012);

[1]All relevant code, models, and data are available at `https://bitbucket.org/dylandialoguesystem/dsttr/src/dsttr_arash_a/`

Hough (2015), who lay the theoretical foundations for incremental generation and later the processing of self-repairs in Dynamic Syntax (Kempson et al., 2001, 2016, Sec. 2.3). For the first time, we develop a probabilistic model for incremental generation (Sec. 3) that conditions next word selection on the current incrementally unfolding context of the conversation, and also on features of a *purely semantic generation goal concept*, expressed as a Record Type (RT) in Type Theory with Records (Cooper, 2012; Cooper and Ginzburg, 2015). The model is trained and evaluated on part of the CHILDES corpus (MacWhinney, 2000) using an extant grammar that was learned by Eshghi et al. (2013) from the same data. Results show that in the best case, the model output matches the gold generation test candidate in 83% of cases (Sec. 4.2). We then go on to experiment with and evaluate the ability of the same model to generate self-repairs in a zero-shot setting in the face of *revisions to the goal concept RT* under various conditions (Sec 4.3): viz. for forward-looking and backward-looking repair and at different distances from the reparandum. Automatic evaluation shows that it can generate self-repairs correctly in 85% of cases. A small human evaluation confirms the overall naturalness and grammaticality of the generated repairs. Overall, these results further highlight the generalisation power of grammar-based models (see also Mao et al. (2021); Eshghi et al. (2017) and lay the foundations for more controllable, and naturally interactive conversational AI systems.

2 Dynamic Syntax and Type Theory with Records (DS-TTR)

Dynamic Syntax (DS, Kempson et al., 2016; Cann et al., 2005; Kempson et al., 2001) is a process-oriented grammar formalism that captures the real-time, incremental nature of the dual processes of linguistic comprehension and production, on a word by word or token by token basis. It models the time-linear construction of *semantic* representations (i.e. *interpretations*) as progressively more linguistic input is parsed or generated. DS is idiosyncratic in that it does not recognise an independent level of structure over words: on this view syntax is sets of constraints on the incremental processing of semantic information.

The output of parsing any given string of words is thus a *semantic tree* representing its predicate-argument structure (see Fig. 1). DS trees are always binary branching, with argument nodes conventionally on the right and functor nodes to the left; tree nodes correspond to terms in the lambda calculus, decorated with labels expressing their semantic type (e.g. $Ty(e)$) and formulae – here as record types of Type Theory with Records (TTR, see Sec. 2.1 below); and beta-reduction determines the type and formula at a mother node from those at its daughters (Fig. 1). These trees can be *partial*, containing unsatisfied *requirements* potentially for any element (e.g. $?Ty(e)$, a requirement for future development to $Ty(e)$), and contain a *pointer*, $\diamond$, labelling the node currently under development.

Grammaticality is defined as parsability in a context: the successful incremental word-by-word construction of a tree with no outstanding requirements (a *complete* tree) using all information given by the words in a string. We can also distinguish *potential grammaticality* (a successful sequence of steps up to a given point, although the tree is not complete and may have outstanding requirements) from *ungrammaticality* (no possible sequence of steps up to a given point).

Fig. 1 shows "John arrives", parsed incrementally, starting with the axiom tree with one node ($?Ty(t)$), and ending with a complete tree. The intermediate steps show the effects of: (i) DS Computational Actions (e.g. COMPLETION which moves the pointer up and out of a complete node or ANTICIPATION which moves the pointer down from the root to its functor daughter.) which are language-general and apply without any lexical input whenever their preconditions are met; and (ii) Lexical Actions which correspond to words and are triggered when a word is parsed.

Context In DS, context, required for processing various forms of context-dependency – including pronouns, VP-ellipsis, and short answers, as well as self-repair – is the parse search Directed Acyclic Graph (DAG), and as such, is also process-oriented. Edges correspond to DS actions – both Computational and Lexical Actions – and nodes correspond to semantic trees after the application of each action (Sato, 2011; Eshghi et al., 2012; Kempson et al., 2015). Here, we take a coarser-grained view of the DAG with edges corresponding to words (sequences of computational actions followed by a single lexical action) rather than single actions, and we drop abandoned parse

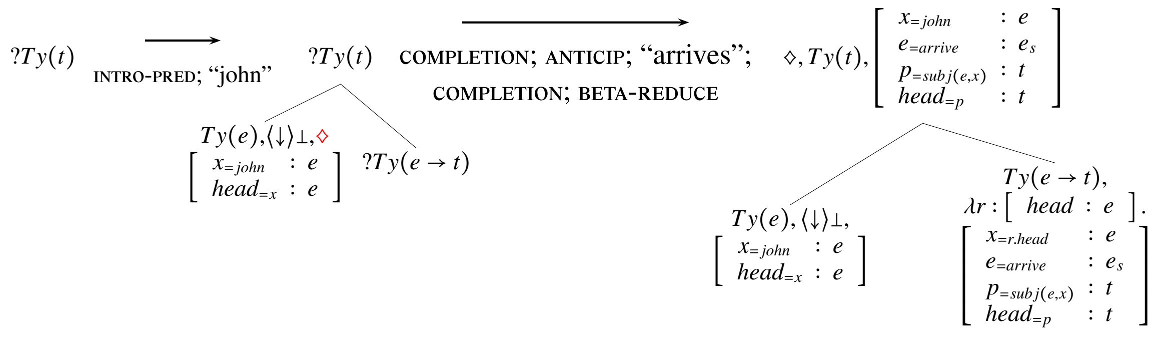

Figure 1: Incremental parsing in DS-TTR: *"John arrives"*

paths (see Eshghi et al., 2015; Howes and Eshghi, 2021, for details) – Fig. 4 shows an example.

2.1 Type Theory with Records (TTR)

Dynamic Syntax is currently integrated with TTR (Cooper, 2012, 2005) as the semantic formalism in which meaning representations are couched (Eshghi et al., 2012; Purver et al., 2011, 2010)[2].

TTR is an extension of standard type theory, and has been shown to be useful in contextual and semantic modelling in dialogue (see e.g. Ginzburg, 2012; Fernández, 2006; Purver et al., 2010, among many others), as well as the integration of perceptual and linguistic semantics (Larsson, 2013; Dobnik et al., 2012; Yu et al., 2017). With its rich notions of underspecification and subtyping, TTR has proved crucial for DS research in the incremental specification of content (Purver et al., 2011; Hough, 2015); specification of a richer notion of dialogue context (Purver et al., 2010); models of DS grammar learning (Eshghi et al., 2013); and models for learning dialogue systems from data (Eshghi et al., 2017; Kalatzis et al., 2016; Eshghi and Lemon, 2014).

In TTR, logical forms are specified as *record types*, which are sequences of *fields* of the form $\left[\ l : T\ \right]$ containing a label l and a type T. Record types can be witnessed (i.e. judged true) by *records* of that type, where a record is a sequence of label-value pairs $\left[\ l = v\ \right]$. We say that $\left[\ l = v\ \right]$ is of type $\left[\ l : T\ \right]$ just in case v is of type T. Fields can be *manifest*, i.e. given a singleton type e.g. $\left[\ l : T_a\ \right]$ where T_a is the type of which only a is a member; here, we write this as $\left[\ l_{=a} : T\ \right]$. Fields can also be *dependent* on fields preceding them (i.e. higher) in the record type (see Fig. 2).

[2]DS models the structural growth of representations and is agnostic to the formalism for semantic representation. As such, it has also been combined with RDF (Addlesee and Eshghi, 2021) and with vector-space representations (Purver et al., 2021)

$$R_1 : \begin{bmatrix} l_1 & : & T_1 \\ l_{2=a} & : & T_2 \\ l_{3=p(l_2)} & : & T_3 \end{bmatrix} \quad R_2 : \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_{2'} \end{bmatrix} \quad R_3 : []$$

Figure 2: Example TTR record types

The standard subtype relation $\sqsubseteq$ can be defined for record types: $R_1 \sqsubseteq R_2$ if for all fields $\left[\ l : T_2\ \right]$ in R_2, R_1 contains $\left[\ l : T_1\ \right]$ where $T_1 \sqsubseteq T_2$. In Fig. 2, $R_1 \sqsubseteq R_2$ if $T_2 \sqsubseteq T_{2'}$, and both R_1 and R_2 are subtypes of R_3. This subtyping relation allows semantic information to be incrementally specified, i.e. record types can be indefinitely extended with more information and/or constraints.

Additionally, Larsson (2010) defines the meet ($\bigwedge$) operation of two (or more) RTs as the union of their fields; the equivalent of conjunction in FoL; see figure 3 for an example. We will need this below (Sec.3) where we define our probabilistic model.

$$\begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \end{bmatrix} \bigwedge \begin{bmatrix} l_2 & : & T_2 \\ l_3 & : & T_3 \end{bmatrix} = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \\ l_3 & : & T_3 \end{bmatrix}$$

Figure 3: Example of merge operation between two RTs

2.2 Generation in DS-TTR

As alluded to in the introduction, to handle typical incremental phenomena in dialogue such as split utterances, interruptive clarification requests or self-repair, any generation model must be as incremental as interpretation: full syntactic and semantic information should be available after generating every word with continual access to the incrementally unfolding context of the conversation (Hough and Purver, 2012; Eshghi et al., 2015).

In generation, there is an extra requirement on models, namely *representational interchangability* (Eshghi et al., 2011): parsing and generation should employ the same mechanisms and use the same kind of representation so that parsing can pick up where generation left off, and vice versa.

DS-TTR can meet these requirements, because generation employs exactly the same mechanisms as in parsing (Purver and Kempson, 2004) with the simple addition of a *subsumption check* against a *generation goal concept*, expressed as a Record Type (RT) in TTR (see Sec. 2.1); and where this goal concept can be partial (does not need to correspond to a complete sentence), and need only to be one step ahead of the generated utterance so far. This ease of matching incrementality in both generation and parsing is not matched by other models aiming to reflect incrementality in the dialogue model while adopting relatively conservative grammar frameworks, some matching syntactic requirements but without incremental semantics (Skantze and Hjalmarsson, 2010), others matching incremental growth of semantic input but leaving the incrementality of structural growth unaddressed (Guhe, 2007).

As such, generation involves *lexical search* whereby at every step, words from the lexicon are test-parsed in order to find words that (i) are parsable in the current context; and (ii) the resulting TTR semantics of the current DS tree subsumes or is monotonically extendable the generation goal. The subsumption relation is the inverse of the subtype relation defined above (see Sec. 2.1; i.e. $R_1 \mathit{subsumes}\, R_2$ iff $R_2 \sqsubseteq R_1$).

Without a probabilistic model for word selection at each step of generation, this process is effectively brute-force, computationally very inefficient, and therefore simply impractical, especially with large lexicons. This is the shortcoming that we address here for the first time by conditioning word selection on the generation goal RT. This involves learning, through Maximum Likelihood Estimation from data, $P(w|T,R_g)$, where w ranges over the lexicon, T is the current DS tree including its maximal semantics, and R_g is the generation goal. This parametrisation is described in full below in Sec. 3.

2.3 Processing Self-repair in DS-TTR

In this section, we briefly introduce the DS model of self-repair from (Hough and Purver, 2012): there are two types of self-repair that are addressed: *backward-looking repair* (aka. overt repair), where the repair involves a local, and partial restart of the reparandum, as in (1) and forward-looking repair (aka. covert repair) where the repair is simply a local extension, i.e. a further specification of the reparandum as in (2).

(1) "Sure enough ten minutes later the bell r-the doorbell rang" *(Schegloff et al., 1977)*

(2) "I-I mean the-he-they, y'know the guy, the the pathologist, looks at the tissue in the microscope…" *(Schegloff et al., 1977)*

In the model set out above, a backward-looking repair arises due to an online revision of a generation goal RT, whereby the new goal is not a subtype of the one the speaker (or the dialogue manager) had initially set out to realise. We model this via backtracking along the incrementally available context DAG as set out above. More specifically, repair is invoked if there is no possible DAG extension after the test-parsing and subsumption check stage of generation (resulting in no candidate succeeding word edge).

The repair procedure proceeds by restarting generation from the last realised (generated) word edge. It continues backtracking by one DAG vertex at a time until the root record type of the current partial tree is a subtype of the new goal concept. Generation then proceeds as usual by extending the DAG from that vertex. The word edges backtracked over are not removed, but are simply marked as repaired (see also Eshghi et al. (2015) for a fuller account), following the principle that the revision process is on the public conversational record and hence should still be accessible for later anaphoric reference (see Fig. 4).

Forward-looking repairs on the other hand, i.e. *extensions*, where the repair effects an "afterthought" are also dealt with straightforwardly by the model. The DS-TTR parser simply treats these as monotonic extensions of the current tree, resulting in subtype extension of the root TTR record type. Thus, a change in goal concept during generation will not always put demands on the system to backtrack, such as in generating the fragment after the pause in "I go to Paris … from London". Backtracking only operates at a semantics-syntax mismatch where the revised goal concept is no longer a subtype of the root record type for the (sub-)utterance so far realised, as in Figure 4.

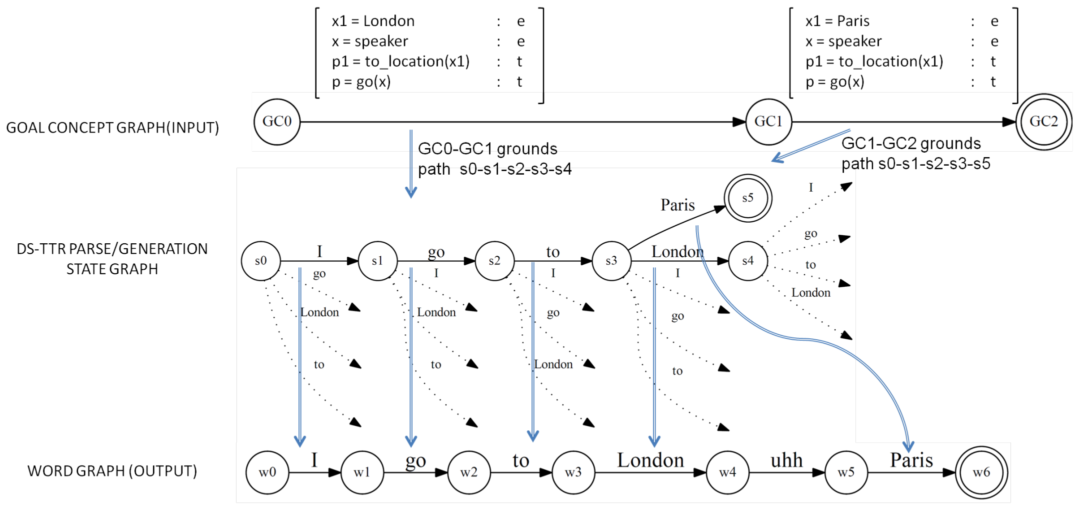
x1 = London : e
x = speaker : e
p1 = to_location(x1) : t
p = go(x) : t
x1 = Paris : e
x = speaker : e
p1 = to_location(x1) : t
p = go(x) : t
GOAL CONCEPT GRAPH(INPUT)
GC0
GC1
GC2
GC0-GC1 grounds path s0-s1-s2-s3-s4
GC1-GC2 grounds path s0-s1-s2-s3-s5
DS-TTR PARSE/GENERATION STATE GRAPH
s0 I s1 go s2 to s3 London s4
Paris s5
WORD GRAPH (OUTPUT)
w0 I w1 go w2 to w3 London w4 uhh w5 Paris w6

Figure 4: Incremental DS-TTR generation of a self-repair upon change of goal concept. Type-matched record types are double-circled nodes and edges indicating failed paths are dotted.

3 Probabilistic Model of Generation

In this section, we follow on from Sec. 2.3 above and describe the probabilistic model that we have developed for incremental probabilistic generation. First we describe the model itself, its parameters, and how these are estimated from data. Then we describe how the model is used at inference time to generate.

Model and Parameter Estimation As noted, generation in Dynamic Syntax is defined in terms of parsing. Specifically, it proceeds via lexical search, i.e. test-parsing (all) words from the lexicon while checking for *subsumption* against the *goal concept*: a record type (RT) in TTR; henceforth R_g. Words that parse successfully with a resulting (partial) semantics that subsume the goal concept are successfully generated. This process goes on until the semantics of the generated sentence equals the goal. This process is highly inefficient and impractical for larger lexicons.

On a high level, we solve this problem by building a probabilistic model which conditions the probability of generating the next word, w, on: (i) R_{cur}: the semantics of the generated utterance thus far; (ii) R_g, the goal concept; and (iii) the current DS tree (henceforth T_{cur}). We condition on (i) to allow the model to keep track of the semantics of what's already been generated, i.e. the left semantic context of generation; on (ii) to aid the model in selecting words that contribute the correct semantic increments to approach the goal concept; and on (iii) to capture the syntactic constraints on what words can grammatically follow. In sum, we need to compute $P(w|T_{cur}, R_{cur}, R_g)$ for all the words w in the lexicon.

As you will see below, we learn to generate by parsing, and therefore we the use Bayes rule in Eq. 3 to cast probabilistic generation roughly in probabilistic parsing terms:

$$\underbrace{P(w|T_{cur}, R_{cur}, R_g)}_{\text{probabilistic generation}} \overset{\text{Bayes Rule}}{=} \frac{\overbrace{P(T_{cur}, R_{cur}, R_g|w)}^{\text{probabilistic parsing}} P(w)}{P(T_{cur}, R_{cur}, R_g)} \tag{3}$$

On the right hand side of Eq. 3, $P(w)$ is the prior probability of w, which we obtain from the frequency of w in our training data; and $P(T_{cur}, R_{cur}, R_g)$ a normalisation constant which we do not need to estimate.

We learn $P(T_{cur}, R_{cur}, R_g|w)$ from gold data in the form of $\langle Utt = \langle w_1, \ldots, w_N \rangle, R_g \rangle$, where Utt is the utterance to be generated, and R_g is its gold semantics. To do this, we use the DS parser to parse Utt yielding a parse path (see e.g. Fig. 4) that starts with the DS axiom tree (empty tree) to the tree whose semantics is R_g together with all the DS trees produced after parsing each w_i in between; viz. a sequence $S_p = \{\langle T_1, w_1 \rangle, \ldots, \langle T_N, w_N \rangle\}$, where T_i are the DS trees in the context of which the w_i's were parsed. This sequence constitutes the observations from which we estimate $P(T_{cur}, R_{cur}, R_g|w)$ by Maximum Likelihood Estimation (MLE).

T_{cur}, R_{cur} and R_g are all composed of many individual features, and as a whole, would be observed very rarely. Therefore, for generalisation, we need to decompose them and compute the probability of the whole as the conjunction (product) of the probabilities of their individual atomic features.

For T_{cur} we follow Eshghi et al. (2013) and consider only one feature of T_{cur}: that of the type of the pointed node, or a requirement for a type (e.g. $Ty(e)$, $?Ty(e \rightarrow t)$, etc) – call this Ty_p. This simplifies the model considerably, but has the downside of not capturing all grammatical constraints (e.g. *person* constraints in English verbs will not be captured this way), and leading to some overgeneration.

We also simplify the model by conditioning on the semantics that *remains to be generated* – call it R_{inc} – rather than conditioning on both R_{cur} and R_g. We can compute R_{inc} each time through the well-defined *record type subtraction* operation in TTR where: $R_{inc} = R_g \backslash R_{cur}$.

With these simplifications, what we need to estimate by MLE from each sequences S_p (see above) is: $P(Ty_p, R_{inc}|w)$.

As noted, for any generalisation at all, R_{inc} now needs to be decomposed into its individual atomic features so that we can compute the probability of each of these features individually, rather than that of R_{inc} as a whole. We decompose R_{inc} as follows: $R_{inc} = \bigwedge_k(R_k)$, where $\bigwedge$ is the TTR equivalent of the conjunction operation in FoL (see above, Sec. 2.1); and each R_k is potentially *dependent* on R_j where $j < k$.

Using the probabilistic variant of TTR (Cooper et al., 2013, 2014), we can use the chain rule to then derive Eq. 4:

$$P(\bigwedge_k R_k|w) = \Pi_k P(R_k|w, R_1 \bigwedge \ldots \bigwedge R_{k-1}) \quad (4)$$

This then allows us to express the probability of a complex record type in terms of the product of its potentially *dependent*, atomic supertypes. This, finally, puts us in a position to compute $P(Ty_p, R_{inc}|w)$ as follows:

$$P(R_{inc}, Ty_p|w) \overset{independence}{=} P(R_{inc}|w) \cdot P(Ty_p|w)$$
$$\overset{decompose\ R_{inc}}{=} P(\bigwedge_k R_k|w) \cdot P(Ty_p|w)$$

We implement the above procedure by constructing a 2D conditional count table where the rows are the words, and the columns are all the atomic semantic features observed during learning by parsing: effectively the result of decomposing all the R_g's in our data; this, in addition to all the Ty_p features we have observed on all the DS trees encountered in the S_p sequences above. Then, each time we observe an atomic semantic feature of R_{inc}, say, R_k, in the context of a word, w, we increment the cell (R_k, w) by 1. After learning, we normalise the columns of the table to obtain all $P(F|w)$ where F ranges over all semantic features and pointed node types, and w over all words in the lexicon.

Inference At inference time, we need to estimate $P(w|T_{cur}, R_{cur}, R_g)$: a probability distribution over all the words in the lexicon, given the current context of generation, T_{cur} including the current semantics so far generated, R_{cur}, and the goal concept, R_g. Given the above we take the following steps to *populate a beam* for generating the next word: (i) compute $R_{inc} = R_g \backslash R_{cur}$; (ii) compute all the atomic semantic features, R_k – the headings in the columns in our conditional probability table – that R_{inc} triggers or 'turns on'. This can be done by checking whether $R_{inc} \sqsubseteq R_k$; (iii) compute the single Ty_p (type of pointed node) feature by observing the type of the pointed node on T_{cur}; (iv) for each row (i.e. each word) take the product (or sum of log probabilities) of all the column features thus triggered in steps (ii) and (iii); (v) sort the words in the lexicon by their probability from (iv) and have the top N fill the beam of size N.

Once the beam is thus populated, we use the DS grammar to parse each word in the beam in turn; upon success, that is, if the word is parsable, and the resulting semantics subsumes the goal concept, R_g, we move on to generate the next word incrementally until we reach the goal concept, that is, until $R_g \sqsubseteq R_{cur} \wedge R_{cur} \sqsubseteq R_g$.

Repair mechanism The DS repair mechanism, i.e. that of backtrack and parse / generate (see above Sec. 2.3), is triggered when none of the words in the beam successfully generate; either because neither are parsable, or else their resulting semantics don't subsume R_g (because it may have been revised). When triggered, the model backtracks over the context DAG path (see above), and, following the same inference process, attempts to (re-)populate the beam and generate from there. Backtracking continues until generation is successful, with the model having generated the interregnum (e.g. "I mean", "sorry I mean", "uh", "no", etc.) before it generates the first repair word.

Generation continues normally from that point until the (potentially new) goal concept is reached.

4 Evaluation

4.1 Data

The data to train and test our model comes from the Eve section of the CHILDES corpus (MacWhinney, 2000). This section was annotated with logical forms (LF) by Kwiatkowski et al. (2012). The LFs were then converted to TTR record types (RT) by Eshghi et al. (2013). This dataset consists of utterances towards children from parents, therefore the sentences have a relatively simple structure than adult language. We will use it in the shape of ⟨Utterance, Goal Concept⟩ pairs to train and test our model.

For training our generator, we test-parsed the dataset using two versions of the grammar learned by Eshghi et al. (2013): the grammar containing the top 1 hypothesis and the grammar containing the top 3. This resulted in two subsets of the data that could be parsed and in which the produced RT semantics matched the gold semantics exactly. Let's call these `top-1` and `top-3` respectively. We report their relevant statistics in Table 1.

dataset	total samples	total words	mode length	max length	type / token ratio
top-1	729	2152	3	7	18.08
top-3	1361	4194	3	7	21.96

Table 1: Filtered Dataset Statistics

However, even as the top-3 grammar from Eshghi et al. (2013) gives wider parsing coverage, it included many erroneously learned lexical actions. We therefore decided to carry out our experiments below on the `top-1` dataset filtered using the top-1 grammar. This is at the expense of not generating sentences that we'd otherwise be able to generate since the overall distribution of the two datasets are similar. Therefore, the results we report below are more conservative (i.e. lower) than those we'd have been able to achieve if we'd manually cleaned up the top-3 grammar and applied it to learning and generation.

4.2 Model Evaluation

We evaluate our generation model on the `top-1` set in two ways: (i) standard evaluation of generation without any mid-generation revisions to the goal; (ii) we evaluate the capability of the same model to generalise to cases where the goal concept is revised mid-generation, i.e. to cases where the model needs to produce *self-repairs*.

Standard evaluation For this, we report percentage of exact match (EM), ROUGE-1, Rouge-2, and ROUGE-l between the gold sentences in the dataset and the output sentences from the model. On the training set, we could observe that out of 656 training samples, we can generate 597 utterances (91.01%) whose semantics exactly matches the generation goal concept; 416 of these fully match the gold sentence, yielding an EM score of 0.6341 (meaning 63.41% of the output sentences fully match the gold sentences). For the test set, out of 73 total samples, 64 sentences were generated fully to the goal concept (87.67%), and 46 of these (63.01%) completely matched the gold sentence in the dataset. Among the outputs not fully match by the gold sentences a large portion of them are very close to an exact match. For example the generated sample, "what is that", where the gold sentence is "what's that": such samples were not counted initially among the exact matches. We then took these to be exact matches and recomputed evaluation scores. The final results are summarised in Table. 2.

	EM	ROUGE-1	ROUGE-2	ROUGE-l
Train	0.84	0.94	0.71	0.92
Test	0.78	0.88	0.67	0.86

Table 2: Evaluation results for generation without any goal concept revisions

4.3 Generating self-repairs: a zero-shot evaluation

To evaluate the ability of the model to generate self-repairs in a zero shot setting, we generate a dataset of *semantic revisions* to the goal concept using the original `top-1` data. We use the Stanford POS tagger to automatically generate a set of revisions, where each revision is a tuple, $\langle R_g, index, R_r, Utt_r, forward\rangle$: R_g: is the original goal concept; *index*: is the position along the generation path where the revision takes place; R_r: is the revised goal; Utt_r: is the result of replacing a single word in the *original* gold utterance with a word from our data of the same POS – R_r now corresponds to the (revised goal) semantics of Utt_r; and, finally: $forward$: is either true or false, marking whether the revised semantic material has already been contributed before *index* or not; if true, we would expect a *forward-*

looking self-repair, and otherwise a *backward-looking* one (see Sec. 2.3 above). We derive these revision tuples for every utterance in the dataset with length greater than 4, and on the following Parts of Speech: {NOUN, ADJ, PROPN, ADP, ADV}. These tuples therefore give us 4 experimental conditions, across two binary factors: (i) locality: is the point at which the revision is made strictly local to the repairandum; or does it have a distance of more than 1; (ii) Is the revision after or before the corresponding semantic contributions have been made?

We then run the revisions through the model and evaluate the output automatically as follows: we use a simple rule-based algorithm to 'clean out' the self-repair from the model output, and compare this to the revised utterance, Utt_r. For this comparison, we only report EM – see Table 3. We observed 641 of the generatable revisions in total are an exact match.

	forward-looking	backward-looking
local	0.93	0.89
distant	0.73	0.82

Table 3: EM for zero-shot evaluation of repairs

Since we do not have gold data for self-repairs, we did a small human evaluation on the model output: the authors each independently annotated a subset of 30 examples, assigning scores on a Likert scale from 1 to 3 for: (a) grammaticality of the self-repairs; and (b) their human-likness or naturalness, which initially led to a low agreement. They then met to discuss the disagreements in order to iron out the differences between the criteria they had applied. They then continued to annotate 70 additional system outputs. This led to a Krippendorff's alpha score of 0.88 for grammaticality and 0.82 for naturalness, demonstrating very high agreement. To then report the average scores given by the human annotators, the lower score was chosen when there was a disagreement, resulting in 2.72 and 2.28 mean scores for grammaticality and naturalness respectively, confirming the quality of the generated output.

5 Discussion

During the error analysis we observed the following error patterns: In the standard evaluation of generation, there were 199 instances where the model had fully generated to the goal concept, while the generated output did not match the gold utterance. Many were cases where the model had generated a statement instead of a question or vice versa (e.g. "I may see them" is generated over "may I see them"). In a few cases, the generated output was ungrammatical with the wrong word order: both of these are caused by the original grammar from Eshghi et al. (2013) overgenerating – this is acknowledged by the authors, and it is due to the fact that their induced grammar did not capture the full set of syntactic constraints present in their data. This is in turn because they were only conditioning their search on the type of the pointed node, like we do here. Inducing the full set of syntactic constraints was left to future work, as it is here.

5.1 Limitations

Our evaluation in this paper has at least two important **limitations**:

(1) We evaluate our incremental generation model on a small, and relatively simple dataset (leading to high ROUGE scores because of the little variation in data and relative similarity between training and testing sets) due to the fact that we currently do not have access to a wider coverage grammar. However, this was a conscious choice on the authors' part: we used a learned grammar to induce our probabilistic generation model and evaluated it on exactly the same dataset from which the grammar was learned (Eshghi et al., 2013). This was deemed to be methodologically both sounder and cleaner than, say, use of a manually constructed grammar. We also believe that the probabilistic model we have contributed here will generalise to larger, more complex datasets when wider-coverage grammars becomes available. We leave this for future work.

(2) Perhaps more importantly, we have no comparative evaluation, and this in a climate where neural NLG has seen astonishing advances in the work on Transformer-based (large) Language Models. To carry out this comparative evaluation, we need to integrate our model with a downstream, and, ideally, multimodal dialogue task (see e.g. Yu et al. (2016, 2017) for how DS-TTR can be integrated within a visually grounded task). This requires substantial further work which is our next step.

5.2 Why a grammar-based approach?

It might reasonably be asked why we are using a grammar-based approach in the age of Large Language Models (LLM) such as GPT-4 and a large number of other, open source models following. These models are astonishing few-shot learners, and have recently achieved great successes that few thought possible (e.g. in open-domain dialogue, conversational question answering, essay writing, summerisation, translation etc), and are changing the human world in ways that we have not yet had time to grasp.

Nevertheless, for the moment, the fact remains that: (a) these models are extremely costly to train and run due their sheer size and the amount of resources (data, compute power, energy) needed to train them; it's also been demonstrated, time and again, that they have poor compositional generalisation properties (see Pantazopoulos et al. (2022); Nikolaus et al. (2019) among others), which explains much of their characteristic data inefficiency; (b) they are very difficult to *control* and/or adapt while often producing factually incorrect statements, commonly referred to as hallucinations (Rashkin et al., 2021; Dziri et al., 2022) using very convincing language – this extends to confident prediction of erroneous actions or plans in multi-modal, embodied settings; (d) they are very hard to sufficiently *verify*, making them unsuitable for use in safety-critical domains such as healthcare; (e) particularly important for us here, unlike recurrent models such as RNNs and LSTMs, standard Transformer-based neural architectures (Vaswani et al., 2017) are not properly incremental – even the auto-regressive variants such as GPT – in the sense that they process word sequences as whole, rather than word by word; they can be run under an 'incremental interface' (Madureira and Schlangen, 2020; Rohanian and Hough, 2021) where input is reprocessed from the beginning with every new token, but even then, they exhibit poor incremental performance with unstable output compared to e.g. LSTMs (Madureira and Schlangen, 2020). Interesting recent work has explored using Linear Transformers (Katharopoulos et al., 2020) with recurrent memory to properly incrementalise LMs (Kahardipraja et al., 2021a), but this work is as yet in its infancy, and we do not yet know of any work that integrates LMs end to end within a real-time, incremental dialogue system.

On the other hand, grammar-based approaches have the advantage of being highly controllable and transparent; but crucially, they incorporate the very large wealth of linguistic knowledge that has arisen from decades of linguistics and semantics research. This knowledge has been demonstrated to be a very effective source of inductive bias in grammar-based models which in turn translates to remarkable generalisation potential, and thus also data efficiency (see e.g. Mao et al. (2021) for a CCG-based multi-modal model, and Eshghi et al. (2017) for a DS-TTR-based one) – see Eshghi et al. (2022) for an extended discussion. One common criticism is that grammar-based models are brittle. This is often true, but we do not believe this to be a fundamental property, and think that specific grammars of a language are adaptable and learnable from interaction. But much work remains to be done to demonstrate this property.

For these reasons, we believe that grammar-based approaches hold promises that are as yet unfulfilled, and are therefore still worth exploring in parallel to the much needed work on making LM architectures and training regimes more incremental (see Kahardipraja et al. (2021b, 2023)).

6 Conclusion

We developed the first semantic, probabilistic model of real-time language generation using the Dynamic Syntax framework. The results show that the model performs well, even though we evaluated it only on a small dataset. We also demonstrated the zero-shot generalisation ability of the model to generate self-repairs where none were observed during training. To our knowledge, this is the first model capable of reacting to real-time changes to the generation goal by generating suitable self-corrections. This ability is essential in dialogue systems in highly dynamic contexts or environments. Our generation model can be seamlessly integrated into incremental dialogue system architectures (e.g. based on Schlangen and Skantze (2009)). This work further highlights the generalisation power of grammar-based approaches, and lays the foundations for creating conversational AI systems that are controllable, data-efficient, more naturally interactive, and more accessible to people with cognitive impairments.

Acknowledgements

We are very grateful to Tanvi Dinkar and Julian Hough for some of the ideas in this paper and subsequent discussion. We would also like to thank the SemDial reviewers whose constructive critique lead to further changes and elaboration.

References

Angus Addlesee and Marco Damonte. 2023. Understanding and answering incomplete questions. In *Proceedings of the 5th Conference on Conversational User Interfaces*.

Angus Addlesee and Arash Eshghi. 2021. Incremental graph-based semantics and reasoning for conversational AI. In *Proceedings of the Reasoning and Interaction Conference (ReInAct 2021)*, pages 1–7, Gothenburg, Sweden. Association for Computational Linguistics.

Angus Addlesee, Ioannis Konstas, and Arash Eshghi. 2019. Current challenges in spoken dialogue systems and why they are critical for healthcare applications.

G.S. Aist, J. Allen, E. Campana, L. Galescu, C.A. Gomez Gallo, S. Stoness, M. Swift, and M. Tanenhaus. 2006. Software architectures for incremental understanding of human speech. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, Pittsburgh.

Ronnie Cann, Ruth Kempson, and Lutz Marten. 2005. *The Dynamics of Language*. Elsevier, Oxford.

Robin Cooper. 2005. Records and record types in semantic theory. *Journal of Logic and Computation*, 15(2):99–112.

Robin Cooper. 2012. Type theory and semantics in flux. In Ruth Kempson, Nicholas Asher, and Tim Fernando, editors, *Handbook of the Philosophy of Science*, volume 14: Philosophy of Linguistics, pages 271–323. North Holland.

Robin Cooper, Simon Dobnik, Shalom Lappin, and Staffan Larsson. 2013. Probabilistic Type Theory and Natural Language Semantics. *Unpublished Manuscript, University of Gothenburg and King's College London*.

Robin Cooper, Simon Dobnik, Shalom Lappin, and Staffan Larsson. 2014. A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS)*, Gothenburg, Sweden. Association for Computational Linguistics.

Robin Cooper and Jonathan Ginzburg. 2015. Type theory with records for natural language semantics. *The Handbook of Contemporary Semantic Theory*, pages 375–407.

Matthew Crocker, Martin Pickering, and Charles Clifton, editors. 2000. *Architectures and Mechanisms in Sentence Comprehension*. Cambridge University Press.

Simon Dobnik, Robin Cooper, and Staffan Larsson. 2012. Modelling language, action, and perception in type theory with records. In *Proceedings of the 7th International Workshop on Constraint Solving and Language Processing (CSLP?Äô12)*, pages 51–63.

Nouha Dziri, Sivan Milton, Mo Yu, Osmar Zaiane, and Siva Reddy. 2022. On the origin of hallucinations in conversational models: Is it the datasets or the models? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5271–5285, Seattle, United States. Association for Computational Linguistics.

A. Eshghi, C. Howes, E. Gregoromichelaki, J. Hough, and M. Purver. 2015. Feedback in conversation as incremental semantic update. In *Proceedings of the 11th International Conference on Computational Semantics (IWCS 2015)*, London, UK. Association for Computational Linguisitics.

A. Eshghi, M. Purver, and Julian Hough. 2011. Dylan: Parser for dynamic syntax. Technical report, Queen Mary University of London.

Arash Eshghi, Eleni Gregoromichelaki, and Christine Howes. 2022. *Action Coordination and Learning in Dialogue*. CSLI.

Arash Eshghi, Julian Hough, and Matthew Purver. 2013. Incremental grammar induction from child-directed dialogue utterances. In *Proceedings of the 4th Annual Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, pages 94–103, Sofia, Bulgaria. Association for Computational Linguistics.

Arash Eshghi, Julian Hough, Matthew Purver, Ruth Kempson, and Eleni Gregoromichelaki. 2012. Conversational interactions: Capturing dialogue dynamics. In S. Larsson and L. Borin, editors, *From Quantification to Conversation: Festschrift for Robin Cooper on the occasion of his 65th birthday*, volume 19 of *Tributes*, pages 325–349. College Publications, London.

Arash Eshghi and Oliver Lemon. 2014. How domain-general can we be? Learning incremental dialogue systems without dialogue acts. In *Proceedings of Semdial 2014 (DialWatt)*.

Arash Eshghi, Igor Shalyminov, and Oliver Lemon. 2017. Bootstrapping incremental dialogue systems from minimal data: linguistic knowledge or machine learning? In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Raquel Fernández. 2006. *Non-Sentential Utterances in Dialogue: Classification, Resolution and Use*. Ph.D. thesis, King's College London, University of London.

Victor Ferreira. 1996. Is it better to give than to donate? Syntactic flexibility in language production. *Journal of Memory and Language*, 35:724–755.

Jonathan Ginzburg. 2012. *The Interactive Stance: Meaning for Conversation*. Oxford University Press.

C. Goodwin. 1981. *Conversational organization: Interaction between speakers and hearers*. Academic Press, New York.

Eleni Gregoromichelaki, Ronnie Cann, and Ruth Kempson. 2012. Language as tools for interaction: Grammar and the dynamics of ellipsis resolution. *The Linguistic Review*, 29(4):563–584.

Eleni Gregoromichelaki, Gregory James Mills, Christine Howes, Arash Eshghi, Stergios Chatzikyriakidis, Matthew Purver, Ruth Kempson, Ronnie Cann, and Patrick G. T. Healey. 2020. Completability vs (in)completeness. *Acta Linguistica Hafniensia*, 52(2):260–284.

Markus Guhe. 2007. *Incremental Conceptualization for Language Production*. NJ: Lawrence Erlbaum Associates.

P. G. T. Healey, Arash Eshghi, Christine Howes, and Matthew Purver. 2011. Making a contribution: Processing clarification requests in dialogue. In *Proceedings of the 21st Annual Meeting of the Society for Text and Discourse*, Poitiers.

Mattias Heldner, Anna Hjalmarsson, and Jens Edlund. 2013. Backchannel relevance spaces. In *Nordic Prosody: Proceedings of XIth Conference, Tartu 2012*, pages 137–146.

Julian Hough. 2015. *Modelling Incremental Self-Repair Processing in Dialogue*. Ph.D. thesis, Queen Mary University of London.

Julian Hough and Matthew Purver. 2012. Processing self-repairs in an incremental type-theoretic dialogue system. In *Proceedings of the 16th SemDial Workshop on the Semantics and Pragmatics of Dialogue (SeineDial)*, pages 136–144, Paris, France.

Christine Howes and Arash Eshghi. 2021. Feedback relevance spaces: Interactional constraints on processing contexts in dynamic syntax. *Journal of Logic, Language and Information*, 30(2):331–362.

Patrick Kahardipraja, Brielen Madureira, and David Schlangen. 2021a. Towards incremental transformers: An empirical analysis of transformer models for incremental nlu. pages 1178–1189. Association for Computational Linguistics.

Patrick Kahardipraja, Brielen Madureira, and David Schlangen. 2021b. Towards incremental transformers: An empirical analysis of transformer models for incremental NLU. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1178–1189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Patrick Kahardipraja, Brielen Madureira, and David Schlangen. 2023. TAPIR: Learning adaptive revision for incremental natural language understanding with a two-pass model. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4173–4197, Toronto, Canada. Association for Computational Linguistics.

Dimitrios Kalatzis, Arash Eshghi, and Oliver Lemon. 2016. Bootstrapping incremental dialogue systems: using linguistic knowledge to learn from minimal data. In *Proceedings of the NIPS 2016 workshop on Learning Methods for Dialogue*, Barcelona.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention.

Ruth Kempson, Ronnie Cann, Arash Eshghi, Eleni Gregoromichelaki, and Matthew Purver. 2015. Ellipsis. In Shalom Lappin and Chris Fox, editors, *The Handbook of Contemporary Semantics*. Wiley-Blackwell.

Ruth Kempson, Ronnie Cann, Eleni Gregoromichelaki, and Stergios Chatzikiriakidis. 2016. Language as mechanisms for interaction. *Theoretical Linguistics*, 42(3-4):203–275.

Ruth Kempson, Wilfried Meyer-Viol, and Dov Gabbay. 2001. *Dynamic Syntax: The Flow of Language Understanding*. Wiley-Blackwell.

Casey Kennington, Spyros Kousidis, and David Schlangen. 2014. InproTKs: A Toolkit for Incremental Situated Processing. In *Proceedings of SIGdial 2014: Short Papers*, pages 84–88.

Tom Kwiatkowski, Sharon Goldwater, Luke Zettlemoyer, and Mark Steedman. 2012. A probabilistic model of syntactic and semantic acquisition from child-directed utterances and their meanings.

Staffan Larsson. 2010. Accommodating innovative meaning in dialogue. *Proc. of Londial, SemDial Workshop*, pages 83–90.

Staffan Larsson. 2013. Formal semantics for perceptual classification. *Journal of logic and computation*.

W.J.M. Levelt. 1989. *Speaking: From Intention to Articulation*. MIT Press.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk*, third edition. Lawrence Erlbaum Associates, Mahwah, New Jersey.

Brielen Madureira and David Schlangen. 2020. Incremental processing in the age of non-incremental encoders: An empirical assessment of bidirectional models for incremental nlu.

Jiayuan Mao, Freda H. Shi, Jiajun Wu, Roger P. Levy, and Joshua B. Tenenbaum. 2021. Grammar-based grounded lexicon learning. In *Advances in Neural Information Processing Systems*.

Shamila Nasreen, Morteza Rohanian, Julian Hough, and Matthew Purver. 2021. Alzheimers dementia recognition from spontaneous speech using disfluency and interactional features. *Frontiers in Computer Science*, 3.

Mitja Nikolaus, Mostafa Abdou, Matthew Lamm, Rahul Aralikatte, and Desmond Elliott. 2019. Compositional generalization in image captioning. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 87–98, Hong Kong, China. Association for Computational Linguistics.

George Pantazopoulos, Alessandro Suglia, and Arash Eshghi. 2022. Combine to describe: Evaluating compositional generalization in image captioning. pages 115–131. Association for Computational Linguistics.

Massimo Poesio and Hannes Rieser. 2010. Completions, coordination, and alignment in dialogue. *Dialogue and Discourse*, 1:1–89.

Matthew Purver, Arash Eshghi, and Julian Hough. 2011. Incremental semantic construction in a dialogue system. In *Proceedings of the 9th International Conference on Computational Semantics*, pages 365–369, Oxford, UK.

Matthew Purver, Eleni Gregoromichelaki, Wilfried Meyer-Viol, and Ronnie Cann. 2010. Splitting the 'I's and crossing the 'You's: Context, speech acts and grammar. In *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, pages 43–50, Poznań. Polish Society for Cognitive Science.

Matthew Purver, Christine Howes, Eleni Gregoromichelaki, and Patrick G. T. Healey. 2009. Split utterances in dialogue: A corpus study. In *Proceedings of the 10th Annual SIGDIAL Meeting on Discourse and Dialogue (SIGDIAL 2009 Conference)*, pages 262–271, London, UK. Association for Computational Linguistics.

Matthew Purver and Ruth Kempson. 2004. Incremental context-based generation for dialogue. In *Proceedings of the 3rd International Conference on Natural Language Generation (INLG04)*, number 3123 in Lecture Notes in Artifical Intelligence, pages 151–160, Brockenhurst, UK. Springer.

Matthew Purver, Mehrnoosh Sadrzadeh, Ruth Kempson, Gijs Wijnholds, and Julian Hough. 2021. Incremental composition in distributional semantics. *Journal of Logic, Language and Information*, 30(2):379–406.

Hannah Rashkin, David Reitter, Gaurav Singh Tomar, and Dipanjan Das. 2021. Increasing faithfulness in knowledge-grounded dialogue with controllable features. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 704–718, Online. Association for Computational Linguistics.

Morteza Rohanian and Julian Hough. 2021. Reframing incremental deep language models for dialogue processing with multi-task learning.

Yo Sato. 2011. Local ambiguity, search strategies and parsing in Dynamic Syntax. In E. Gregoromichelaki, R. Kempson, and C. Howes, editors, *The Dynamics of Lexical Interfaces*. CSLI Publications.

E.A. Schegloff, Gail Jefferson, and Harvey Sacks. 1977. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382.

David Schlangen and Gabriel Skantze. 2009. A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 710–718, Athens, Greece. Association for Computational Linguistics.

Gabriel Skantze and Anna Hjalmarsson. 2010. Towards incremental speech generation in dialogue systems. In *Proceedings of the SIGDIAL 2010 Conference*, pages 1–8, Tokyo, Japan. Association for Computational Linguistics.

Gabriel Skantze and David Schlangen. 2009. Incremental dialogue processing in a micro-domain. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 745–753, Athens, Greece. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. volume 2017-December.

Yanchao Yu, Arash Eshghi, and Oliver Lemon. 2016. Training an adaptive dialogue policy for interactive learning of visually grounded word meanings. In *Proceedings of SIGDIAL 2016, 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 339–349, Los Angeles.

Yanchao Yu, Arash Eshghi, and Oliver Lemon. 2017. Learning how to learn: an adaptive dialogue agent for incrementally learning visually grounded word meanings. In *Proceedings of the First Workshop on Language Grounding for Robotics*, pages 1–10. ACL.