In: *Proceedings of the 7th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EpiRob)*, pages 215–220, Lisbon, Portugal - Sep 2017.

Syntactic Reanalysis in Language Models for Speech Recognition

Johannes Twiefel[*], Xavier Hinaut[*†‡§] and Stefan Wermter[*]
[*]Knowledge Technology Group, Department of Computer Science, University of Hamburg
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany {twiefel, hinaut, wermter}@informatik.uni-hamburg.de
[†]Inria Bordeaux Sud-Ouest, Talence, France xavier.hinaut@inria.fr
[‡]LaBRI, UMR 5800, CNRS, Bordeaux INP, Université de Bordeaux, Talence, France
[§]Institut des Maladies Neurodégénératives, UMR 5293, CNRS, Université de Bordeaux, Bordeaux, France

***Abstract*—State-of-the-art speech recognition systems steadily increase their performance using different variants of deep neural networks and postprocess the results by employing N-gram statistical models trained on a large amount of data coming from the general-purpose domain. While achieving an excellent performance regarding Word Error Rate (17.343% on our Human-Robot Interaction data set), state-of-the-art systems generate hypotheses that are grammatically incorrect in 57.316% of the cases. Moreover, if employed in a restricted domain (e.g. Human-Robot Interaction), around 50% of the hypotheses contain out-of-domain words. The latter are confused with similarly pronounced in-domain words and cannot be interpreted by a domain-specific inference system.**

The state-of-the-art speech recognition systems lack a mechanism that addresses the syntactic correctness of hypotheses. We propose a system that can detect and repair grammatically incorrect or infrequent sentence forms. It is inspired by a computational neuroscience model that we developed previously. The current system is still a proof-of-concept version of a future neurobiologically more plausible neural network model. Hence, the resulting system postprocesses sentence hypotheses of state-of-the-art speech recognition systems, producing in-domain words in 100% of the cases, syntactically and grammatically correct hypotheses in 90.319% of the cases. Moreover, it reduces the Word Error Rate to 11.038%.

I. Introduction

State-of-the-art automatic speech recognition (ASR) systems like Google's Search by Voice [1], [2], [3] or Baidu's Deep Speech 2 [4] are based on deep neural networks (DNN), which require huge amounts of data and processing power. Statistical higher order n-gram models are also trained on large amounts of text data from the general-purpose domain to perform language modeling or postprocessing of the hypotheses.

Speech is an important modality in Human-Robot Interaction (HRI) and used to communicate with the agent. The mentioned state-of-the-art ASR systems achieve impressive performance on benchmark data sets regarding ASR metrics like Word Error Rate (WER). For HRI, the requirement for an ASR hypothesis is not only that it has a low WER, but also that it can be understood and interpreted by the robot. This interpretation can be performed using thematic role labeling systems [5], [6]. If the syntax is violated or out-of-domain words occur, the ASR hypotheses could possibly not be interpreted. Often, HRI consists of a specific application scenario, where the robot is inside a restricted domain and does not need to have knowledge about the world outside of the domain. For example, a kitchen robot needs to understand utterances related to a kitchen scenario, like `'cook me a meal'` but is not expected to understand sentences like `'score a goal'` which belong to a football domain.

As we showed already in previous work, domain knowledge helps in outperforming state-of-the-art general-purpose models in ASR benchmark tasks [7]. If the sentences to be possibly uttered by the users are known in advance or could be generated by a given grammar, the system achieved better performance regarding the WER metric. Additionally, the hypotheses produced by our system possess a grammatically and syntactically correct form. This correctness should assist the interpretation of those hypotheses by thematic role labeling systems [5]. If the concrete sentences for a restricted domain or a grammar are not available, we also use a statistical n-gram model to generate the hypotheses, like Google and Baidu did. N-gram models are only able to help with correcting acoustic recognition errors in the local context of a word and not in the global context (the whole sentence). As statistical models rely on probabilities, their benefit is to be able to generate combinations of words which are not covered by the training data. This leads to the dilemma that word combinations can be generated which violate the grammar and syntax of the given language.

In this work, we propose a method which combines the advantages of a grammar with the ones of a statistical n-gram model to reduce the risk of generating incorrect hypotheses and is also able to generalize to non-existent word combinations in the training data. The model is inspired by the model of Hinaut and Dominey [5][8] which exploits the "constructions" (templates of sentences) hypothesis about children language acquisition [9].

II. Related Work

In this section, we shortly present the mechanisms of state-of-the-art speech recognition systems and our previous work on phonemic postprocessing to improve the performance of ASR systems in a restricted domain.

©2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. DOI: not available yet.

A. State-of-the-Art Speech Recognition

State-of-the-art speech recognition systems like Google's Search by Voice [1], [2], [3] or Baidu's Deep Speech 2 [4] employ a deep Long Short-Term Memory (LSTM) [10] together with Connectionist Temporal Classification (CTC) [11]. LSTMs are able to learn long-range dependencies inside a sequence, compared to N-gram Hidden Markov Models (HMM) which are only able to model the local context. Frames of audio data are taken as input and trained to produce e.g. phones, letters or words. The problem of the different timescales between acoustic frames and labels is solved using CTC, which introduces a *blank* label to fill the gaps and performs an alignment between the two sequences. The outputs are postprocessed by a 5-gram statistical language model both for Google's and Baidu's ASR.

B. Phonemic Postprocessing

Our preliminary work [7] suggests that domain knowledge helps in improving the results of DNN-based speech recognition by postprocessing them using this knowledge. Traditional speech recognition commonly used before the success of *Deep Learning* [12] in general consists of an acoustic model, which generates phonemes from acoustic data and a language model that generates words based on a grammar or a statistical n-gram model. The phoneme representation of these words are then scored against the phoneme sequences generated by the acoustic model to produce a probabilistic word sequence hypothesis. As described in section II-A, state-of-the-art ASR can also generate an intermediate representation like phonemes. However, the acoustic model and the language model can also both be included in a large DNN and phonemes are not necessary in this case as words are being generated directly.

The acoustic models used for traditional speech recognition are based on Mel Frequency Cepstral Coefficients (MFCCs) [13] which are used to extract human speech from the audio signal. The acoustic model is trained on MFCC features derived from speech and the corresponding phoneme sequences.

A phoneme is the smallest meaning distinguishing unit to express a word. Compared to text, a character would be the smallest meaning distinguishing unit. Phonemes can be uttered using different phones or speech sounds, which makes phonemes a superclass of phones. Comparing this to characters again, a character can be expressed using different fonts. By this definition, a phoneme is speaker-independent, making it a suitable intermediate representation that can be used for scoring. We hypothesize that a phoneme is also spoken the same way independently from its domain, meaning phonemes are spoken the same way in a kitchen, football or Human-Robot Interaction context, which would make acoustic modeling domain-independent, and acoustic models could be transferred from one domain to another.

Another hypothesis is that language models are domain-dependent, as the training data for a model should follow the same distribution as the data of the environment it is used in, and this is only true if a general-purpose model is used in domains which are a subset of a general-purpose domain. If a general-purpose language model is used only inside a specific domain that does not follow the same distribution as the general-purpose domain, the language model is not the optimal one for this domain.

For this reason, we proposed a unified ASR system that consists of a large and well-trained DNN-based domain-independent acoustic model combined with a domain-specific language model. Due to the nature of DNNs to require large amounts of data and the lack of this data in small domains, we recommended to use traditional language modeling like statistical n-grams models and grammars.

One of the approaches are based on the traditional open-source ASR system Sphinx-4 [14], which uses HMMs for language and acoustic modeling and a Viterbi decoder to find the best word sequence hypothesis. As the acoustic model relies on MFCCs and is trained on a limited amount of labeled acoustic data compared to the massive amount of data companies like Google are able to generate and process, the acoustic model is the weakness of the Sphinx-4 system. The scoring is performed on the phoneme level, which offers the possibility to remove the acoustic model from Sphinx and feed in a phoneme sequence directly. Instead of training our own domain-independent acoustic model, we employ the massive acoustic models of e.g. Google by delegating the acoustic processing to Google's Search by Voice. The new unified system is called *DOCKS* [7] and supports language models in the form of grammars (*DOCKS Grammar*) or statistical bigram models (*DOCKS Bigram*).

Google's hypothesis for the reference text `'addressed mail'` could be something similar to `'a dressed male'` which is completely incorrect on the word level. On the phoneme level both grapheme sequences can be represented as `'AH D R EH S T M EY L'`. We employ the trainable grapheme-to-phoneme converter SequiturG2P [15] and train it on CMUdict 0.7a [1] to be able to generate a phoneme sequence for any grapheme sequence coming from Google's ASR. These phoneme sequences are then fed to our postprocessing system. We proved that this principle works better than using the given acoustic models of Sphinx-4 [7].

Another approach contained in the *DOCKS* system is called *DOCKS Sentencelist*. If a list of all possible sentences that can be uttered in a restricted domain is known beforehand, this restricted but robust approach can be used. The approach is based on the Levenshtein distance [16], which is a standard method to calculate a distance score between two sequences a and b, with i and j being the recursively processed indices of the sequences:

$$\mathcal{L}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \mathcal{L}_{a,b}(i-1,j) + 1 \\ \mathcal{L}_{a,b}(i,j-1) + 1 \\ \mathcal{L}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases} \tag{1}$$

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

We convert the 10 best hypotheses from Google's ASR to phoneme sequences and do the same for the list of expectable sentences. Then, a normalized Levenshtein distance is calculated over all 10 best phoneme sequences (H) against all phoneme sequences of the sentence list (S):

$$\lambda \ = \ \text{argmin} \ \mathcal{L}_{h_k,s_l}(|h_k|, |s_l|) \tag{2}$$

where $\mathcal{L}$ is the Levenshtein distance. The confidence value was computed as

$$\gamma^A = \max(0, 1 - \frac{\mathcal{L}_{h_k,s_l}(|h_k|, |s_l|)}{|s_l|}) \tag{3}$$

with $h_k \in H$ (set of the 10 best hypotheses) and $s_l \in S$ (set of reference sentences) both in phonemic representation. As this approach is the most restricted one, it performed best (WER around 0%) if all spoken sentences are known in advance [7].

III. Approach

This work is inspired by the work of Hinaut and Dominey [5], who proposed a neural computational model for thematic role labeling from incoming grammatical constructions [17], while our approach does not employ neural networks. One hypothesis for language acquisition is that children learn "templates of sentences" [9] (i.e. grammatical constructions [17]). The θRARes model proposed by Hinaut and Dominey [5] learns to assign θ-roles (thematic roles) to the semantic words (SW) or content words in a sentence. E.g. the sentence `'put the pyramid on the cube'` is mapped to the predicate `'put(pyramid, cube)'`. This mapping is called a grammatical construction [9]. The sentence is preprocessed by removing all semantic words from the sequence and replacing them by the wildcard token `X`: `'X the X on the X'`. In this work, we call these structures like `'X the X on the X'` a *Sentence Template* (ST), which is adapted from Hinaut and Dominey [5]. These STs are used for our language model.

The system learns to assign the slots of the predicate to the SWs of the sequence and requires to determine if a word is a SW and has to be replaced by the wildcard token. As SWs are an open class, which means that new words can be added to this class, it is not trivial to determine if a word is a SW or not. Instead, non-semantic words, which are function words, are identified, as they belong to a closed class meaning that there is only a finite number. All words that do not belong to the closed class of function words are considered to be SWs. As function words belong to a closed class, they are known beforehand even for new domains.

The idea of our work is to be able to perform domain-restricted language modeling while also making use of general-purpose language models. The text output coming from Google's ASR is not ignored like in previous approaches (see Sec. II-B) where only the phonemic representation was processed. This approach also handles the syntactic structure in the form of STs. We consider function words to be domain independent, which makes STs consist of domain-independent elements. Instead of only processing the phonemic representation of a hypothesis, we are now able to exploit the benefits of a general-purpose language model.

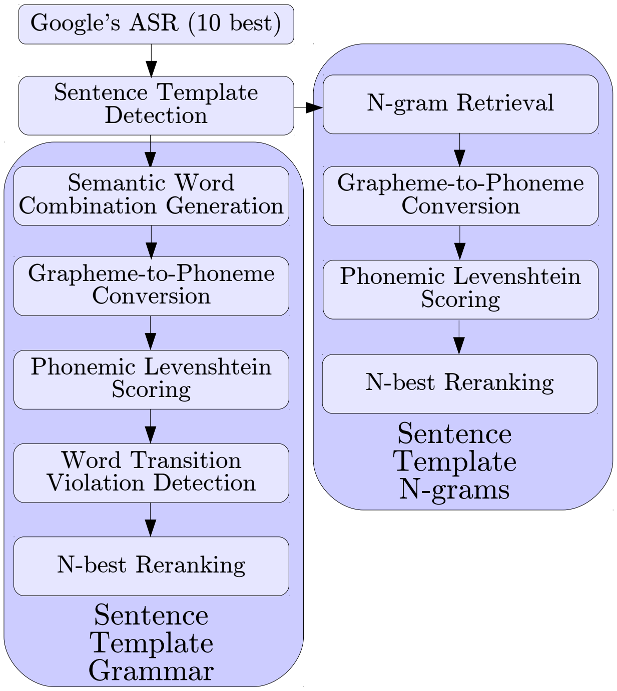
Google's ASR (10 best)
Sentence Template Detection
Semantic Word Combination Generation
Grapheme-to-Phoneme Conversion
Phonemic Levenshtein Scoring
Word Transition Violation Detection
N-best Reranking
Sentence Template Grammar
N-gram Retrieval
Grapheme-to-Phoneme Conversion
Phonemic Levenshtein Scoring
N-best Reranking
Sentence Template N-grams

Fig. 1. Architecture. This figure shows the processing pipeline of our system. First, the system tries to use the *Sentence Template Grammar*. If the confidence for individual words or subsequences is below a threshold, this part of the hypothesis is postprocessed by the *Sentence Template N-gram* module.

Figure 1 depicts the processing pipeline of our system. We take the text hypothesis coming from Google's speech recognition and build the ST out of it. The training data that we possess for our restricted domain consists of grammatically and syntactically correct sentences. As the hypothesis coming from Google may be grammatically or syntactically incorrect, we try to find the closest sentence producible from the training data. To be able to generate variations that are not covered by the training data, we do not match the concrete sentence (like in Sec. II-B), but only its ST against the ST of the training sentences using the normalized Levenshtein distance mentioned in section II-B. This produces a ranked list of STs closest to the one that was generated by Google's ASR.

Inspired by the work of Hinaut and Dominey [5] we interpret the SWs of the training data as terminal words (non-replaceable words) in a grammar, and the wildcard slots as the non-terminals to be replaced. We call these sets of possible SWs *Terminal Bags* (TB; e.g. $t0$, $t1$ below). This way, the model is able to generate variations of sentences for the same ST that may not be contained literally in the training data. The following example clarifies the principle. The training data consists of the sentences:

- `'put the pyramid on the cube'`
- `'move the prism on the block'`
- `'move the prism to the left'`

The ST for the first two sentences is `'X the X on the X'` and `'X the X to the X'` for the third one. The training is performed by generating a grammar:

- ⟨s0⟩ = ⟨t0⟩ the ⟨t1⟩ on the ⟨t2⟩
- ⟨s1⟩ = ⟨t3⟩ the ⟨t4⟩ to the ⟨t5⟩
- ⟨t0⟩ = put | move
- ⟨t1⟩ = pyramid | prism
- ⟨t2⟩ = cube | block
- ⟨t3⟩ = move
- ⟨t4⟩ = prism
- ⟨t5⟩ = left

This grammar is able to generate, e.g., the sentence 'put the prism on the cube', which is not present in the training data. Processing a speech utterance is performed by generating a hypothesis using e.g. Google's ASR. In this example, the reference text is 'put the prism on the cube', Google may produce the hypothesis 'pull the pistol on the cube'. This hypothesis is transformed to its ST, which is 'X the X on the X'. We employ the normalized Levenshtein distance (see eq. 3) to calculate the best matching ST from the training data, which is in this case also 'X the X on the X'. Then, the SWs are matched against each other to find the best matching word from the TB. This means that we calculate the normalized Levenshtein distance of 'pull' against 'put' and 'pull' against 'move' on phoneme level for the first SW gap. We do this for all SW gaps and get the most probable sequence 'put the prism on the cube'.

In some cases, the hypothesis coming from Google cannot be converted to a correct ST, e.g. 'put **them** prism on the cube'. In this case, we calculate the best matching ST and generate possible combinations for the incorrect part of the sequence, e.g. 'put the pyramid', 'put the prism', 'move the prism', 'move the pyramid'. This guarantees a correct ST which can be interpreted by a θ-role assignment model (e.g. [5]).

As the normalized Levenshtein distance can be used as a confidence for each word, words that the system is uncertain about can be identified. This information can be used to repair a hypothesis in a second step in case some of the words possess a low confidence (threshold 0.5).

For long and nested sentences the correct hypothesis cannot be constructed due to the lack of training data, meaning not enough samples for a specific ST and a lack of possible SWs. This means, the TBs do not contain many words and possibly not the word that was uttered. We use a threshold for each word and repair all words that were recognized with a confidence lower than this threshold (0.5). For this, we train the system by collecting all n-grams inside the training data up to 10-grams. For example, if we have the input sentence 'put the yellow prism on the cube', we possibly could only generate 'put the blue prism on the cube', as the word 'yellow' was for example never used in this SW gap. The function words 'the' and 'on' can be used as "anchors" to retrieve n-grams starting with 'the' and ending with 'on'. Then, another scoring process is started using the normalized Levenshtein distance to calculate the best matching n-gram from the training data. This way, we can generalize from other sentences and STs in the training data.

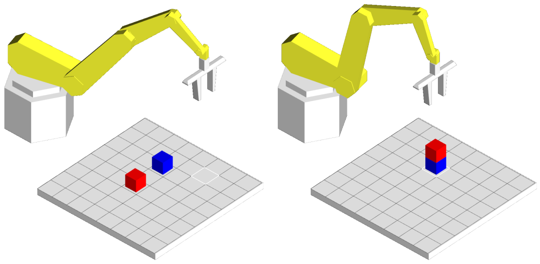

Fig. 2. Example from the corpus. An example for a board scene before (left board) and after (right board) the command 'Move the red brick on top of the blue brick'.

To reduce the number of incorrect sentences, we check each generated hypothesis for non-existing word transitions (bigrams) in the training data and, if occurring, we can drop that hypothesis. Also, we perform the postprocessing for the 10 best hypotheses coming from Google, and afterwards sort our generated hypotheses by average word confidence.

IV. Experiments

To evaluate our system we chose the *Train Robots* dataset [18]. It consists of instructions directed at a robot arm that can move objects around like boxes and pyramids in a discrete world and is inspired by SHRDLU [19]. The text corpus contains 2500 training sentences and 909 test sentences, which are linguistically rich, including ellipses, anaphoric references, multi-word spatial expressions and lexical disambiguation. The dataset has a "before" and "after" scene for each of the commands, Fig. 2 shows an example. The sentences are e.g. 'Move the red brick on top of the blue brick' or 'Pick up the blue block from the top of the green block and put it down on the blue block which lies next to another green block'.

The original corpus was created via crowd-sourcing and contains a lot of grammatical and syntactic errors, which would make a model learn incorrect syntax. A prerequisite for our approach to work is that the system learns a correct syntax of the language. It could be possible to learn a correct syntax from data that contains partially incorrect syntax, as incorrect syntax could be identified via outlier or anomaly detection, which could possibly be integrated into an extension of this approach.

For restricted domains like the *Train Robots* scenario we believe that there is not enough data available to be able to identify incorrect syntax in an unsupervised way. That is why we corrected grammatical errors in the training data to make sure the system learns only correct syntax. We recorded audio data for the 909 test sentences with 9 different non-native speakers, each one uttering 101 sentences from the test set.

We compare the performance of our system with Google's ASR, *DOCKS Bigram* and *DOCKS Sentencelist* and measure Word Error Rate (WER) and Sentence Error Rate (SER), which is the rate of incorrect hypotheses (ones that contain at least one error). Also, we calculate the number of

TABLE I
PERFORMANCE REGARDING CONVENTIONAL ASR METRICS ON THE *Train Robots* DATA SET.

Approach	Word Error Rate	Sentence Error Rate
Google	17.3%	77.7%
DOCKS Bigram	24.2%	76.9%
DOCKS Sentencelist	32.2%	81.7%
ST Grammar	22.6%	71.2%
ST N-gram	**11.0%**	**53.355%**

grammatically and syntactically correct sentences. The 2500 training sentences are used to train *DOCKS Bigram*, *DOCKS Sentencelist* and our models. For *DOCKS Sentencelist* the list of expectable sentences is not given in this scenario, but the training sentences are used to build the sentence list.

Table I shows the results of our experiments. Google's Search by Voice ASR achieves a WER of 17.343%, which is better than *DOCKS Bigram* (24.338%) and shows that the corpus is too challenging to improve the results with *Bigram* postprocessing. When using *Sentencelist* postprocessing, the WER rises to 32.174%, which shows that the corpus contains so many variations that the intersection between training data and test data is not large enough to make this approach usable for this data set. By performing the first postprocessing step (*ST Grammar*), the WER also increases while the SER decreases, which means that more sentences are recognized correctly but fewer words in total. The second postprocessing step (*ST N-gram*) reduces the WER to 11.038% and the SER to 53.355%, which is a large boost in performance regarding conventional ASR benchmark metrics.

For the second evaluation, we measure the number of syntactically and grammatically correct sentences produced by the baseline approach (Google) and our best approach. The *ST N-gram* approach produced 90.319% syntactically and grammatically correct sentences and no incorrect word transitions. When incorrect syntax was produced, there were incorrect verbs inside a correct phrase structure in most of the cases, e.g. `'hold the yellow cube on top of the single blue cube'`, which would be correct if `'hold'` was replaced by `'move'`. A known ST was created in 100% of the cases. The hypotheses produced by Google's Search by Voice were syntactically and grammatically correct for only 42.684% of the test sentences (according to the subjective measure of an expert), while we even ignored errors concerning wrong capitalization, which would lead to almost 0% syntactically and grammatically correct hypotheses. Also, more than 50% of the sentences contained words that did not belong to the vocabulary of the domain like `'Musa red pyramid for sale spec'`, because words were confused with similarly pronounced out-of-domain words. The original sentence here was `'move the red pyramid to the left edge'`.

V. DISCUSSION AND CONCLUSION

In the current robotic application, our domain-dependent postprocessor clearly outperforms Google's ASR regarding conventional metrics like WER and SER. The model is able to generalize for unseen sentences and is producing syntactically correct hypotheses in 9 out of 10 cases. The increase in WER and decrease in SER for the *ST Grammar* is caused by the lack of training data in the domain, as long sentences (and their ST) only have few samples and though not all combinations of SWs can be produced. For shorter sentences, where more examples are available, the system is performing much better. The *ST N-Gram* approach does not have this kind of problems, as SW gaps are filled with n-grams coming from other training sentences. The main contribution of this model is the possibility to generate syntactically correct sentences without having to hand-craft a grammar, instead, the grammar is learned by our approach in an unsupervised way. Also a list of all possible sentences that can be uttered is not necessary any more.

One disadvantage is that sometimes syntactically incorrect sentences are produced by the *ST N-gram* model like `'hold the yellow cube on top of the single blue cube'`, which is caused by the local context of the n-gram. If the training data of the *ST Grammar* approach was augmented, these cases would not occur for that method, as such cases would not be present in the training data. The system could be used for interactive data augmentation by generating new training sentences using the *ST N-gram* model and adding those sentences to the training data of the *ST Grammar* if validated by a human teacher. We interpret this model as a function that is necessary to produce syntactically correct sentences and is not present in other systems.

The θ-RARes model of Hinaut and Dominey [5] is also based on STs and can be used to generate predicates containing the thematic roles of a hypothesis. As most of the sentences are syntactically correct and all of them possess a correct ST, the model is also able to understand incorrect hypotheses which are expected to be rejected completely by other systems. In previous work [6], we investigated the problems of integrating inconsistent multimodal input (coming from speech and vision), where the θ-RARes model was also employed. The results showed that 50% of the spoken input coming from the user was inconsistent with the input coming from the vision module. For example, the user was describing a position of an object and confusing left and right. In that work, we showed that feedback generated from incorrect input is useful for the user to interact with an agent.

In future work, we plan to connect the θ-RARes model to our system to be able to interpret both correct and incorrect sentences and provide feedback if necessary instead of only rejecting incorrect input. Also, we plan to integrate a visual component which is able to interpret the generated predicates if possible or otherwise provide suggestions to change specific words inside a hypothesis to make it plausible and consistent with the visual context. We believe that this idea cannot be implemented using the raw results coming from Google's ASR or other comparable ASR systems, as they often contain out-of-domain words or are syntactically incorrect.

MacGregor et al. [20] and Visser et al. [21] state that lexical-semantic categorization is necessary to interpret words and cannot be performed for invented words. Given the Google hypothesis `'Musa red pyramid for sale spec'`, our system "does not know" words like `'Musa'`, `'sale'` or `'spec'`, as they are not contained in the training data. These words are treated as out-of-domain words and cannot be interpreted by the system. These cases do not occur in the hypotheses generated by our system, as no out-of-domain words can be produced, which guarantees a lexical-semantic categorization. Additionally, errors like `'hold the yellow cube on top of the single blue cube'` can be prevented using the θ-RARes model, as such a predicate was never contained in the training data and could be rejected.

Diaz and McCarthy [22] found out that function words cause activity in different brain areas compared to SWs, which supports our hypothesis that a speech recognition system should treat function words differently than SWs, as our system does. In general, the P600 Event-Related-Potential (ERP) that could be recorded from the brain is thought to be an indication for a syntactic reanalysis and repair mechanism [23]. Therefore we think syntactic postprocessing mechanism is necessary for an ASR system.

Also, new words can easily be added to our language model by declaring a new word to be equivalent to an existing word given the context. E.g., the unknown word `'cuboid'` can be declared to be used in the same context as the known word `'box'`. The word `'cuboid'` is then added to all *Terminal Bags* of the *ST Grammar* where the word `'box'` is present. Also, all n-grams of the *ST N-gram* model containing the word `'box'` are taken, the word `'box'` is replaced by `'cuboid'` and the new n-gram is added to the training data. This process is performed automatically after only providing the grapheme representation of the new word `'cuboid'`.

The proposed method could also help with improving performance in other HRI disciplines, like child ASR [24]. The system was running in realtime on a computer (Intel(R) Core(TM) i5-4590 CPU @ 3.30GH, 16GB RAM) which makes it also interesting for other HRI tasks.

Acknowledgment

We would like to thank Tobias Hinz for correcting the grammatical errors in the data, and all speakers for their participation. We gratefully acknowledge partial support for the project " LingoRob - Learning Language in Developmental Robots" by Campus France PHC Procope Project 37857TF and by DAAD Project 57317821 in the Förderprogramm "Projektbezogener Personenaustausch Frankreich".

References

[1] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[2] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *Interspeech*, 2014, pp. 338–342.

[3] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4580–4584.

[4] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[5] X. Hinaut and P. F. Dominey, "Real-time parallel processing of grammatical structure in the fronto-striatal system: a recurrent network simulation study using reservoir computing," *PloS one*, vol. 8, no. 2, p. e52946, 2013.

[6] J. Twiefel, X. Hinaut, M. Borghetti, E. Strahl, and S. Wermter, "Using natural language feedback in a neuro-inspired integrated multimodal robotic architecture," in *25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2016, pp. 52–57.

[7] J. Twiefel, T. Baumann, S. Heinrich, and S. Wermter, "Improving domain-independent cloud-based speech recognition with domain-dependent phonetic post-processing," in *Twenty-Eighth AAAI. Québec City, Canada*, 2014, pp. 1529–1535.

[8] X. Hinaut, M. Petit, G. Pointeau, and P. F. Dominey, "Exploring the acquisition and production of grammatical constructions through human-robot interaction with echo state networks," *Frontiers in Neurorobotics*, vol. 8, 2014.

[9] M. Tomasello, *Constructing a language: A usage based approach to language acquisition*. Cambridge, MA: Harvard University Press, 2003.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] P. Mermelstein, "Distance measures for speech recognition – psychological and instrumental," in *Joint Workshop on Pattern Recognition and Artificial Intelligence*, 1976, pp. 374–388.

[14] P. Lamere, P. Kwok, W. Walker, E. Gouvea, R. Singh, and P. Wolf, "Design of the cmu sphinx-4 decoder," in *8th European Conference on Speech Communication and Technology (EUROSPEECH)*. ISCA, 9 2003, pp. 1181–1184.

[15] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[16] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics – Doklady*, vol. 10, no. 8, pp. 707–710, 2 1966.

[17] A. Goldberg, *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, 1995.

[18] K. Dukes, "Train robots: A dataset for natural language human-robot spatial interaction through verbal commands," in *ICSR. Embodied Communication of Goals and Intentions Workshop, Bristol, UK*, 2013.

[19] T. Winograd, "Understanding natural language," *Cognitive psychology*, vol. 3, no. 1, pp. 1–191, 1972.

[20] L. J. MacGregor, F. Pulvermüller, M. Van Casteren, and Y. Shtyrov, "Ultra-rapid access to words in the brain," *Nature Communications*, vol. 3, p. 711, 2012.

[21] M. Visser, E. Jefferies, and M. L. Ralph, "Semantic processing in the anterior temporal lobes: a meta-analysis of the functional neuroimaging literature," *Journal of cognitive neuroscience*, vol. 22, no. 6, pp. 1083–1094, 2010.

[22] M. T. Diaz and G. McCarthy, "A comparison of brain activity evoked by single content and function words: an fmri investigation of implicit word processing," *Brain research*, vol. 1282, pp. 38–49, 2009.

[23] A. D. Friederici, "The brain basis of language processing: from structure to function," *Physiological reviews*, vol. 91, no. 4, pp. 1357–1392, 2011.

[24] J. Kennedy, S. Lemaignan, C. Montassier, P. Lavalade, B. Irfan, F. Papadopoulos, E. Senft, and T. Belpaeme, "Child speech recognition in human-robot interaction: Evaluations and recommendations," in *12th ACM International Conference on Human-Robot Interaction*, 2017.