Interspeech 2018
2-6 September 2018, Hyderabad

An Optimization Based Approach for Solving Spoken CALL Shared Task

Mohammad Ateeq,Abualsoud Hanani, Aziz Qaroush

Birzeit University, Palestine
{mateeq, ahanani, aqaroush}@birzeit.edu

Abstract

In this paper, we are describing our developed systems for the 2018 SLaTE CALL Shared Task on grammatical and linguistic assessment of English spoken by German-speaking Swiss teenagers. The English spoken response is converted to text using baseline English DNN-HMM ASR trained on the shared task training data and another two commercial ASRs (Google and Microsoft Bing). The produced transcription is assessed in terms of language and meaning errors. In this work, we focused on the text-processing component. Grammatical errors are detected using English grammar checker, part of speech analysis and extracting incorrect bi-grams from grammatically incorrect responses. Errors related to the meaning are detected using novel approaches which measure the similarity between the given response and stored set of reference responses.

The outputs of several systems have been fused together into one overall system, where the fusion weights and parameters are tuned using genetic algorithm. The best result on the 2018 shared task test dataset is D-score of 14.41, which was achieved by the fused system and the optimized set of incorrect bi-grams.

Index Terms: speech recognition, human-computer interaction, optimization, linguistic assessment.

1. Introduction

Over time, the introduction of Computer-Assisted Language Learning (CALL) models is a pioneer factor in development of speech and language technology especially after integrating Automatic Speech Recognition (ASR) as one of the components. CALL system can better help improve language skill of the L2 learners. To date, most of the common speech-based CALL systems focus on the pronunciation quality of the L2 language. A good and well-documented example of these systems is the EduSpeak system [1] which plays the student a recorded sentence, asks them to imitate it, and then rates them on the accuracy of their imitation, giving advice if appropriate on how to improve pronunciation or prosody. There is no doubt that this is useful, but does not give the student a real opportunity to practice spoken language skills.

Rayner et. al. in [2] took this a further step by building a speech-based CALL system by which students can interact and response to the systems prompts. This system prompts the student in his/her L1 language indicating in an indirect way what he/she is supposed to say in the L2 language. Then, the system automatically assesses the spoken response, based the grammar and linguistic, and provides a feedback.

Five of the participants in the 2017 CALL shared task [3, 4, 5, 6, 7] presented their systems in the Slate 2017 workshop [1] which was held directly after the Interspeech 2107.They introduced different ideas for improving to the baseline system at both the ASR and the text processing stages.

[1] http://www.slate2017.org/

Mengjie Qian et al. [3] improved the ASR WER to 9.27 compared with 14 for the baseline ASR. In addition to the development data provided by the shared task, they used AMI [8] and German PF-STAR [9] datasets to train a hybrid DNN-HMM ASR system. In addition, they enhanced the grammar in the text processing phase by including additional responses that were extracted from the transcriptions of the shared task.

Axtmann et .al. [4] developed their system for the spoken call shared task based on a pipeline of predefined rules. They followed different rules [10] that cover German mispronunciations in English. The best result reported was D score of 4.79.

Magooda and Litman [5] enhanced the baseline-system by extracting syntactic and semantic features from the transcripts to measure the inconsistency on the language level and on the meaning, respectively. They extracted 10 features to detect if the response is linguistically correct or not. They used NLTK English spell checker [11] to detect the spelling mistakes. Stanford part of speech tagger [12] was used to measure how real the response is. Two classification techniques were used to evaluate the effectiveness of the extracted features: K-nearest neighbor (KNN) [13] and Support Vector Machines (SVM) [14]. Their system achieved the third rank in the 2017 CALL shared task competition and the best D-score was 3.047.

The performance of pre-existing automated content scoring system was investigated in [6]. This system extracted various features from word n-gram and used them to accept or reject the responses. Support vector regression was used to train several models to evaluate different sets of proposed features. The experiments showed that the features based on the similarity between the user response and the possible responses are more effective for this task. The system achieved a D score of 4.353 using 2017 shared task test data.

A deep learning based approach was proposed by [7] to evaluate the grammar and semantic errors of the user response. Different types of features were extracted from the response: language models features, features were extracted using sentence-embedding approach [15], two features to measure the similarity between the the user response and all possible responses were extracted using word-embedding approach [16] and other different grammar features. Deep Neural Network (DNN) was used to evaluate the features. The experiments showed that the D score increased to 4.37 when the proposed method was evaluated on the test set.

In this paper, we describe our systems and results for the 2018 CALL shared task. The general structure of the baseline system consists of English ASR followed by a text processing component which assesses the user response and decide if it is correct or incorrect in terms of grammar and linguistic meaning. In all of our presented systems, we used the baseline ASR system provided by the shared task organizers which achieved the highest WER (9.27) in the 2017 CALL shared task developed and submitted by the University of Birmingham team [3]. In the text processing unit, various approaches were used for

10.21437/Interspeech.2018-1328

estimating similarity between the user response and the stored correct responses.

2. The 2018 CALL shared task

2.1. Shared task overview

CALL shared task 2018 is the second edition of the original CALL shared task which was organized last year [2]. The results of the first edition of the task were presented at the SLaTE workshop in August 2017, with the highest D score of 4.766 [3]. The training data of the second edition was released in October 2017 and test data in March 2018. In addition, the annotation of each utterance contains a prompt, a transcription by human experts, a meaning evaluation (correct/incorrect), and an overall evaluation (correct/incorrect) done by human experts. The basic assessment method performs speech recognition on spoken response and then accepts it if the recognized text matches at least one of the reference responses corresponding to the given prompt [17]. The system accepts responses which are grammatically and linguistically correct and rejects those incorrect either in grammar or meaning according to the judgments of a panel of human experts [18].

CALL shared task 2018 is similar to the original one, in which it consists of two versions; the first one focuses on the effect of speech recognizer and the other focuses on the effect of text processing at the overall system performance. Consequently, the input of the speech-processing version consists of an identifier, a German text prompt, and a speech file containing an English language response. For the text-processing version, there is an extra text string representing the text obtained from a baseline ASR system [3] on the speech file. In addition, it provides the reference texts corresponding to each prompt.

2.2. Data description

The data provided for the first and second edition of this CALL shared task were collected from different German speaking schools in 2014 and 2015. The speakers are German-speaking Swiss students ranging in age from 12 to 15 years. Each participant was asked to response verbally in English to a given German text prompt. Three native English language experts judged each response as accepted or rejected in terms of both language grammar and meaning. For each prompt, a number of possible accepted responses were added by the experts and used as a correct reference responses. The data was divided into training and testing. The training set contains 6698 utterances in the second edition and the testing set contains 1000 utterances. The gender, age, English proficiency and motivation of the participants made balanced in the training and testing subsets. The recording environment is not perfect due to the background noises in schools.

2.3. Evaluation metric

The annotators labeled each data item according to its linguistic correctness and its meaning. The correctness of the vocabulary and grammar were used for the linguistic assessment. On the other hand, the meaning was judged according to the context of the given prompt. Consequently, the response is rejected if either the linguistic or the meaning is incorrect, and accepted when both are correct. However, it makes more 'sense' for the system to accept meaningful responses with language mistakes, than to accept linguistically correct with meaningless responses.

[2]http://regulus.unige.ch/spokencallsharedtask

Thus, for each prompt, the system's output (correct or incorrect) falls into one of the following four categories compared with the language and meaning gold standards provided by the annotators:

1. **Correct Reject (CR)**: It represents the number of utterances where the system rejects students response which is incorrect in term of meaning or language.
2. **Correct Accept (CA)**: It represents the number of utterances where the system accepts students response which is correct in meaning and it has no linguistic error.
3. **False Reject (FR)**: It represents the number of utterances where the system rejects students response which is correct in meaning and it has no linguistic error.
4. **False Accept (FA)** is defined by $FA = PFA + k : GFA$, where PFA Plain False Accept represent the number of utterances where the students response is correct in meaning but has a linguistic error, and the system accept it, and GFA represent the number of incorrect responses in terms of meaning or linguistic where the system accept it. The parameter k represents a weighting factor that makes gross false accepts relatively more important which is set to 3.

According to the above four categories, the performance of the overall system is calculated by the Differential (D) score [18] which is mathematically defined by the following equation:

$$D = \frac{CR/(CR+FA)}{FR/(FR+CA)} \quad (1)$$

3. Proposed system

Our proposed system is a rule-based model which mainly depends on the similarity between the user response and the set of reference responses, where the similarity is calculated in two ways using the Cosine similarity between plain texts and using the Jaccard similarity based on Part Of Speech (POS) tagging. In addition, the proposed system employed two new approaches to enhance the overall performance.

3.1. Pre-processing

Before computing similarity, the output of ASR is first cleaned for further processing. In this stage, abbreviations in the transcript text are expanded (e.g 'I'd' to 'I would) and some duplicated words are removed.

3.2. Cosine similarity

Cosine Similarity (CS) measure is used to compute the similarity between the user response and each reference response in the grammar file. Formally, given a user response UR and its possible references PR = [PR_1,PR_2, . . . ,PR_N] where N is the number of all possible responses for the corresponding prompt. Cosine similarity is calculated as:

$$CS(UR, PR_i) = \frac{\sum_{n=1}^{m} URw_i.PRw_i}{\sqrt{\sum_{n=1}^{m} URw_i^2}.\sqrt{\sum_{n=1}^{m} PRw_i^2}} \quad (2)$$

Where, UR=[URw_1, . . . ,URw_m] represents the m-dimensional vector for the user response UR and PR_i=[PRw_1, . . . ,PRw_m] represents the vector for the ith possible response PR. Note that, all of these vectors are computed using bag-of-words model of all distinct terms occurred in

the set of all possible responses PR. Then each vector is multiplied by a weighting vector Tw = [Tw_1,Tw_2, . . . ,Tw_m], where m is the number of distinct terms and Tw_i is the weight of corresponding term T_i calculated as:

$$Tw_i = TF_i * \frac{n_i}{N} \quad (3)$$

where TF_i is the frequency of a term T_i in response r_i $\in$ {UR,PR$_i$ }, N is the number of all reference responses in PR and n_i is the number of possible responses containing term T_i. Weighting vector in equation3 puts more weight on the term that occurs more frequently in the corresponding possible responses.

3.3. Part-of-Speech (POS) level similarity

In this type of similarity, a sequence is created for each user response by converting each possible response for that user response into its corresponding POS (Part of Speech) level. Formally, let $r_i \in$ {UR,PR$_i$ } = [t_1, t_2, . . . , t_m] represents all the terms occurred in a certain response. Thus, the POS-level list for r_i can be represented by POSTAG = [p_1, p_2, . . . , p_m], where the p_i is the part of speech for the term t_i in the response r_i. Also, let PR = [PR_1,PR_2, . . . ,PR_N] represents a collection of all possible responses for a certain user response, POS_{PRi} is the POS-level list for possible response PR_i and POS_{UR} is the POS-level list for user response. Therefore, the similarity between POS_{UR} and POS_{PRi} is estimated by Jaccard score (JS), as shown in the following equation:

$$JS(POS_{UR}, POS_{PRi}) = \frac{POS_{UR} \bigcap POS_{PRi}}{POS_{UR} \bigcup POS_{PRi}} \quad (4)$$

However, Jaccard score does not take into consideration the order of the elements in the two sets. So, the Edit Distance (ED) metric is employed to measure the similarity between POS_{UR} and POS_{PRi} sets. The edit distance is defined as the minimum number of (add/delete/replace) operations required to transform POS_{UR} set into POS_{PRi} set. The following formula is used to compute and normalize the edit distance (NED) between two sets:

$$NED = \frac{POS_{UR} \bigcap POS_{PRi}}{POS_{UR} \bigcap POS_{PRi} + ED} \quad (5)$$

3.4. Basic rule-based judgment method

The shared task basic system was used as a baseline system for all of our proposed systems. The proposed systems take a final decision about the given response (correct or incorrect), by passing audio transcription given by ASR through a sequence of stages and rules as shown in the following:

Rule1: Extract the grammar errors using a Python checker tool [3]. Therefore, if a grammar errors is found, the system rejects the response at this stage.

Rule2: If the response has no grammatically errors, the system converts each possible response in Grammar XML file (i.e. reference responses) into its corresponding POS_{PRi} set. Similarly, it converts the user response into POS_{UR} set. Then, it computes the Jaccard coefficient and the normalized ED between POS_{PRi} and POS_{UR} sets. Therefore, If the maximum value of the Jaccard measure is less than an experimentally predefined threshold (JAC_{TH}) and the maximum value of the normalizedED is less than an experimentally threshold (NED_{TH}) , the system rejects the response (i.e. the response is incorrect).

[3]https://pypi.python.org/pypi/grammar-check/1.3.1

Rule3: If the conditions in step 2 and step 3 above are not satisfied, the system computes the cosine similarity between the student response and each response in the reference responses. The system takes the maximum value and compares it with an experimentally threshold (COS_{TH}) to decide if the response is correct or not. This threshold value is practically tuned on the enrollment data.

3.5. Fusion of multiple systems

To handle some of the errors caused by ASR system, an additional two well-known ASRs were used to process the user response including; Google ASR and Microsoft Bing ASR. Therefore, each user response is converted into text using these ASRs in addition to the SLaTE2018 ASR to get three transcriptions, $TEXT_{GOOGLE}$,$TEXT_{BING}$ and $TEXT_{SLaTE2018}$. Table 1 shows the transcriptions of three examples recognized by the three mentioned ASRs. It is clear that GOOGLE and BING ASRs are more accurate than the baseline ASR in the first example. On the other hand, the baseline ASR performs better in the second example.

Table 1: *Examples for different recognized texts.*

Recognized Text	**ASR**
from italy	True Transcription
i'm from italy	SLaTE2018
from italy	GOOGLE
from italy	BING
I want to leave at Tuesday	True Transcription
I want to leave at Tuesday	SLaTE2018
I want to leave on Tuesday	GOOGLE
I want to leave at two today	BING

Algorithm 1 describes how these three recognized texts are combined to make the final decision. The algorithm starts by computing CS, JS, and NED scores for each ASR transcript for each reference response and selects the maximum score. We adopt weighted linear sum of each ASR score, such that $Score_i = W_{i1} * Score_{iASR1} + W_{i2} * Score_{iASR2} + W_{i3} * Score_{iASR3}$. All wights ($W_1$,$W_2$,...,$W_9$), and thresholds ($JAC_{TH}$, NED_{TH} and COS_{TH}) were optimized using the genetic algorithm. The function "pythonGrammarCheck" returns 1 if grammar checker tool detects an error in the response. The configurations of the genetic algorithm is as follow: the chromosomes are represented by a list of length 12, where the first 9 positions 1_{th} to 9_{th} hold the weights W_1 to W_9, while the remaining positions (10_{th},11_{th} and 12_{th}) were used to hold JAC_{TH}, NED_{TH} and COS_{TH}, respectively. The chromosome was initialized by 12 random numbers between 0 and 1 taking into account three constrains: (I) $W_1 + W_2 + W_3 = 1$, (II) $W_4 + W_5 + W_6 = 1$ and (III) $W_7 + W_8 + W_9 = 1$. For mutation operator, we choose a randomly position in the chromosome and set its value to new random number between 0 and 1. Two point crossover operator were used. The crossover probability and mutation probability were set to 0.7 and 0.3, respectively. The D score of 6698 training samples was used to decide how chromosome fitness will be evaluated during the iterations. In each training example, algorithm 1 was used to accept/reject the response. However D score was set to 0 when the system rejects less than 25% of all incorrect responses. Algorithm 1 provide a different decision when the genes in a specified chromosome fluctuate through iterations, because the wights W_1 to

W_9 and all thresholds are inputs to this algorithm. Finally, a tournament selection was used, and the algorithm was executed for 1000 generation with population size equal 100.

Algorithm 1: Classification of the response based on the weights and thresholds

Input RecognizedTranscripts= [$TEXT_{GOOGLE}$,$TEXT_{BING}$,$TEXT_{SLaTE2018}$], W_1 to W_9, JAC_{TH}, NED_{TH},COS_{TH};
Initialize JacScoreList=[], NEDScoreList=[], CosScoreList=[], pythonCheckList=[];
while *Text= RecognizedTranscripts.getElement* **do**
  while *Possible Response= getPossibleResponse* **do**
    CosineValues.add(CS(UR, PR_i).;
    Jaccards.add(JS(POS_{UR},POS_{PRi});
    NormalizedEDs.add(NED(POS_{UR},POS_{PRi});
  end
  JacScoreList.add(MAX(Jaccards));
  NEDScoreList.add(MAX(NormalizedEDs));
  CosScoreList.add(MAX(CosineValues));
  pythonCheckList.add(pythonGrammarCheck(Text))
end
CosScore = W_1 * CosScoreList[0] + W_2 * CosScoreList[1] + W_3 * CosScoreList[2] ;
JacScore=W_4 * jacScoreList[0] + W_5 * jacScoreList[1] + W_6 * jacScoreList[2];
NEDScore = W_7 * NEDScoreList[0] + W_8 * NEDScoreList[1] + W_9 * NEDScoreList[2];
if *sum(pythonCheckList)* $\geqslant 2$ **then**
  decision=reject;
else if *JacScore*$< JAC_{TH}$ *and NEDScore*$< NED_{TH}$ **then**
  decision=reject;
else if *CosScore* $< COS_{TH}$ **then**
  decision=reject;
else
  decision=accept;
end
Output decision

3.6. Extract a list of incorrect bi-grams

In this approach, a list of all incorrect bi-gram tokens are extracted from the user response that has one or more language errors (Language="incorrect and meaning = "correct"). Similarly, the bi-gram tokens of all of the corresponding reference responses are extracted. Therefore, if there is a bi-gram in the grammatically incorrect response which does not exists in any reference response bi-grams, we add it to a new list called "incorrectBigrams". On the other hand, if a correct user response includes a bi-gram that is found in "incorrectBigrams" list, we remove it from the list. Indeed, not all bi-gram tokens in the "incorrectBigrams" list cause a linguistic errors because the student may correct his response. For example, the student response "can I pay with credit card ah post card sorry" is classified as correct and it has "card sorry" bi-gram which is not exist in any reference response.

Genetic algorithm was used as an optimization technique to refine the "incorrectBigrams" list. Formally, let Bi=[BI_1, . . . ,BI_m] represents all bi-grams in "incorrectBigrams" list, where m is the number of bi-gram tokens in the list. The chromosome modeled as binary list [C_1, . . . ,C_m]. In this representation, the gene C_i equals zero if removing the bi-gram BI_i leads to increase the D score of training examples. Otherwise, C_i equals one. To compute the D score of 6698 training examples, the weights W_1 to W_9, and thresholds (JAC_{TH}, NED_{TH} and COS_{TH}) were fixed and computed as described in section 3.5. Two point crossover operators were used. For mutation, we flip the value of a randomly chosen gene in the chromosome. The crossover probability and mutation probability were set to 0.7 and 0.3 respectively. The algorithm was executed for 1000 generation with population size equal 100. To add the refined list in the judgment procedure, the student response is rejected when at least two recognized texts contain a bi-gram in the refined list.

4. Results and discussion

In this section, we present the results of our three systems: system1 which uses baseline ASR and applies the basic rules that descried in section 3.4 (system1 in table2), system2 which applies the fusion technique described in section 3.5 (system2 in table 2) and system3 which adds the "incorrect bigrams" list described in section 3.6 (system3 in table 2). The second edition of the CALL shared task had 18 submissions. The performance of our proposed systems is reported in table 2 with a comparison with the best 5 results. All results were evaluated using the 2018 test data which was released in Feb. 2018.

Table 2: *Results of the three proposed systems, where IRej =rejections on incorrect responses and CRej=rejections on correct responses.*

System	IRej	CRej	D score
Baseline	0.777	0.145	5.343
System1	0.673	0.111	6.079
III	0.364	0.033	10.909
GGG	0.381	0.033	11.424
KKK	0.399	0.033	11.965
System2	0.26	0.02	12.12
HHH	0.342	0.025	13.492
System3	0.33	0.02	14.41
LLL	0.305	0.016	19.088

As shown in the table 2, the performance of system2 is improved to 12.12 (D score) compared to 6.08 of the system1. This is because of the fusion weights (W_1 to W_9) and thresholds (JAC_{TH}, NED_{TH} and COS_{TH}) optimization with the genetic algorithm. Also, the results show the improvement (D score of 14.41) when adding the refined list of incorrect bigrams to system2. As expected, this list did not affect the rejections on correct responses (CRej), but it improved the D score by increasing IRej (rejections on incorrect responses).

5. Conclusions

In this paper, an optimization-based approaches to assess the English spoken responses released by the 2018 CALL shared task. Three English ASRs were used in the proposed systems. They were fused together, where the fusion weights were tuned using the genetic algorithm. A list of bi-gram tokens was extracted from grammatically incorrect responses and then refined using the genetic algorithm. The best performance of 14.4 (D score) was achieved by the fused system and the optimized set of incorrect bi-grams.

6. References

[1] H. Franco, H. Bratt, R. Rossier, V. Rao Gadde, E. Shriberg, V. Abrash, and K. Precoda, "Eduspeak®: A speech recognition and pronunciation scoring toolkit for computer-aided language learning applications," *Language Testing*, vol. 27, no. 3, pp. 401–418, 2010.

[2] E. Rayner, P. Bouillon, and J. Gerlach, "Evaluating appropriateness of system responses in a spoken call game," 2012.

[3] M. Qian, X. Wei, P. Jancovic, and M. Russell, "The university of birmingham 2017 slate call shared task systems," in *Proceedings of the Seventh SLaTE Workshop, Stockholm, Sweden*, 2017.

[4] N. Axtmann, C. Mehret, and K. Berkling, "The csu-k rule-based pipeline system for spoken call shared task."

[5] A. Magooda and D. Litman, "Syntactic and semantic features for human like judgement in spoken call," in *Proc. 7th ISCA Workshop on Speech and Language Technology in Education*, pp. 109–114.

[6] K. Evanini, M. Mulholland, E. Tsuprun, and Y. Qian, "Using an automated content scoring system for spoken call responses: The ets submission for the spoken call challenge."

[7] Y. R. Oh, H.-B. Jeon, H. J. Song, B. O. Kang, Y.-K. Lee, J.-G. Park, and Y.-K. Lee, "Deep-learning based automatic spontaneous speech assessment in a data-driven approach for the 2017 slate call shared challenge," in *Proc. 7th ISCA Workshop on Speech and Language Technology in Education*, pp. 103–108.

[8] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal *et al.*, "The ami meeting corpus: A pre-announcement," in *International Workshop on Machine Learning for Multimodal Interaction*. Springer, 2005, pp. 28–39.

[9] A. Batliner, M. Blomberg, S. D'Arcy, D. Elenius, D. Giuliani, M. Gerosa, C. Hacker, M. Russell, S. Steidl, and M. Wong, "The pf_star children's speech corpus," in *Ninth European Conference on Speech Communication and Technology*, 2005.

[10] E. Atwell, P. Howarth, and D. Souter, "The isle corpus: Italian and german spoken learner's english," *ICAME Journal: International Computer Archive of Modern and Medieval English Journal*, vol. 27, pp. 5–18, 2003.

[11] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[12] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003, pp. 173–180.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[14] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[15] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward, "Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 4, pp. 694–707, 2016.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[17] E. Rayner, C. Baur, C. Chua, and N. Tsourakis, "Supervised learning of response grammars in a spoken call system," 2015.

[18] C. Baur, J. Gerlach, E. Rayner, M. Russell, and H. Strik, "A shared task for spoken call?" 2016.