Proceedings of the 18th Conference on Computer Science and Intelligence Systems pp. 1325–1328
DOI: 10.15439/2023F9627
ISSN 2300-5963 ACSIS, Vol. 35

Boosting conversational AI correctness by accounting for ASR errors using a sequence to sequence model

Szymon Jadczak
Adam Mickiewicz University in Poznań
Faculty of Mathematics and Computer Science
Email: szyjad@st.amu.edu.pl

Rafał Jaworski
0000-0001-8827-3318
Adam Mickiewicz University in Poznań
Faculty of Mathematics and Computer Science
Email: rjawor@amu.edu.pl

***Abstract*—This paper describes the winning submission to the challenge CAICCAIC: Center for Artificial Intelligence Challenge on Conversational AI Correctness. The aim of the challenge was to design a mechanism of natural language understanding capable of interpreting user prompts. The prompts were the output of an automatic speech recognition system and therefore contained errors. In this scenario, it was necessary to apply techniques of accounting for these errors. As per the results of the challenge, the most effective technique proved to be an original use of a sequence to sequence model. The key idea was the concatenation of labels before passing them to the model for training and prediction.**

I. Problem formulation

AUTOMATIC Speech Recognition (ASR) systems are immensely popular in today's world. They find use in assistant applications for phones, cars or at home. The ASR techniques have been perfected over many years to achieve maximum available output quality. However, it is not always possible to recognize speech with perfect accuracy due to numerous factors, such as:

- external noise,
- individual voice features (prosody),
- ambiguity of spoken language

and many others. Problems with the accuracy of ASR may also arise from using imperfect models which produce sub-optimal output quality.

In such scenarios it is well justified to apply an error proof natural language understanding (NLU) module on the results of ASR. The goal of that module is the conversion of the text output of ASR into semantically meaningful objects. Typically, NLU modules operate on ASR output which is assumed to be correct. If ASR makes an error, NLU is not necessarily expected to interpret the output of ASR correctly. In this challenge, however, the text input to NLU is noisy.

This simulates the real-life data which is typically presented to ASR systems in the form of user commands. Being able to counter the challenges of processing this data allows for the creation of more robust and usable voice command interpreters. This research has potentially very high impact on the experience of the users of those systems who are often left frustrated by the ASR module not functioning properly. This frustration not only lowers the user's satisfaction but often causes the user to refrain from using the system completely. Proper handling of ASR errors can enable the usage of voice commands more frequently and in more different scenarios.

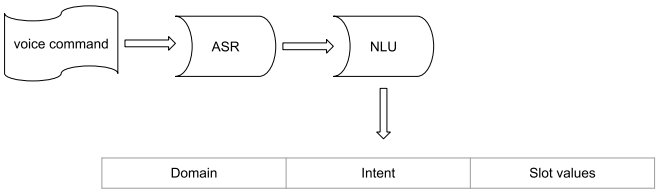

Fig. 1. The usage of the ASR system

The CAICCAIC: Centre for Artificial Intelligence Challenge on Conversational AI Correctness [1] was organized to spark the research on ASR error correction. The challenge was published on the Gonito.net platform [2]. The data set consisted of utterances of user commands annotated with the following data:

- Domain
- Intent label
- Slot values

The usage of the ASR system is presented in Figure 1 (source of the figure: [1]).

Domain is a general indication of the environment that the user is interacting with. For instance, this may be the name of the voice operated appliance, such as *Airconditioner*.

Intent is the specific action the user would like to see accomplished by using a voice command. The intent in the data set is given as a string label representing a specific function of the appliance. In the domain of an air conditioner, the example intent label is *SetTemperatureToValueOnDevice*.

Slot values are pieces of specific information being passed along with the voice command. Exemplary slot values are presented below:

```
{'device_name': 'reception room',
'value': '82 degrees fahrenheit'}
```

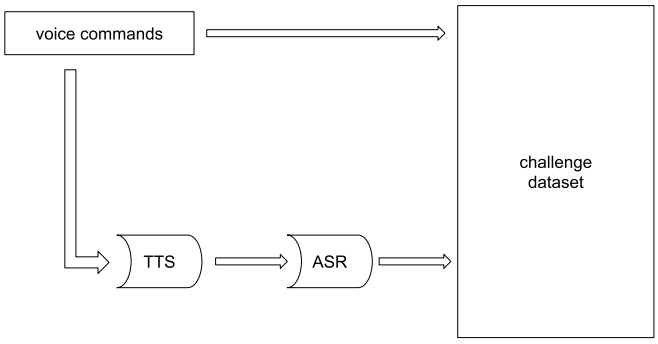

Fig. 2. Challenge data preparation

TABLE I
STATISTICS OF THE DATA

Language	Set	Utterances	Mean length
English	test	3344	9.95
English	train	13022	9.34
English	valid	3633	9.28
Spanish	test	3520	13.21
Spanish	train	15043	13.37
Spanish	valid	3546	13.15
Polish	test	3494	8.93
Polish	train	12753	8.97
Polish	valid	3498	9.02

Some of the annotated user commands in the data set of the challenge were intentionally distorted by the task organizers. However, in order not to let the participants of the challenge train NLU modules on those distortions, majority of user commands were left intact.

Distortion was achieved by first feeding the original utterances into a Text-To-Speech (TTS) system in order to obtain their audio versions. These, in turn, were converted back to text with the means of an ASR system. Since both TTS and ASR are prone to errors, the final text output was distorted. The utterances were prepared according to the scheme presented in Figure 2 (source of the figure: [1]).

The data was prepared in English, Spanish and Polish. Annotated utterances were split into train, test and validation sets. The statistics of the data are presented in Table I.

II. RELATED RESEARCH

The problem of identifying the domain and intent can be seen as classification of distorted data. Such problems were typically approached with the use of statistical methods. One such solution is described in [3]. This work was dealing with the problem of thematic classification of texts. The texts were articles from a collection of digital libraries and were output of an OCR mechanism. Noisy data was not corrected but instead classified as is with the use of Latent Dirichlet Allocation.

The same text data was also used in another research - automatic prediction of the year of publication of the article. This was organized as the RetroC challenge [4] on the aforementioned Gonito.net platform. Winning submissions to this challenge also did not venture to correct OCR errors but instead performed the classification on the raw text data.

The problem of filling the slots, on the other hand, requires not only classification but deeper understanding of spoken language. The article [5] describes the challenges of this task and lists current solutions. Among the main challenges is the nature of ASR errors. These errors are significantly different than those observed in text (typing, spelling or grammar mistakes). In ASR whole words and phrases are substituted with fragments sounding similarly but carrying completely different meaning.

According to the authors [5], majority of researchers approaching the problem of interpreting noisy ASR output focus on correcting the errors first with the use of text correction tools, such as in [6]. The corrected ASR output is then interpreted using a NLU module. However, in recent years a new trend is observed. Experiments with direct understanding of spoken language (SLU - Spoken Language Understanding) have yielded impressive results (see for instance [7]).

The approach assumed in the CAICCAIC challenge, however, is based only on text processing. This has the following advantages over SLU:

- independence of the ASR module,
- not requiring specialistic speech-to-meaning data sets,
- ability to take full advantage of recent advances in text modelling and generation.

Among natural language processing techniques known to operate well on text containing errors we can mention character-based neural networks. The paper [10] presents research on error correction using character-based attention architecture. Thanks to operating on the character level, the solution is able to deal with out-of-vocabulary words.

III. SOLUTION

This section describes the author's solution to the problem formulated by the CAICCAIC challenge which was evaluated as the winning submission.

The classic solution to this problem would involve training three separate classifiers – for domain, intent and slot values. The problem of identifying the domain is relatively the easiest. It can be viewed as a classification problem with few classes (there were not many distinct domain labels in the data set). Such problem could have been solved with classic text feature extraction methods (TF-IDF) and statistical classification mechanisms, such as SVM. Since the domain is heavily dependent on some specific keywords in the user prompt (e.g. temperature - air conditioner, event - calendar etc.)

Similarly, the intent classification could probably also be approached this way. Here, however, the spectrum of possible values of the intent label is wider. Moreover, the intent is not necessarily well correlated with specific words in the user prompt. This is due to the fact that a single intent can be expressed in many ways by the user. Consider the following example: the intent of checking the current temperature on

an air conditioner can be expressed in the following ways (examples taken from the training set of the challenge):

- how many fahrenheits degrees are on my cooling system
- show me the temperature on the playroom thermostat
- give me temperature on my air conditioning

This makes the problem of classifying the intent label more difficult than the classification of the domain.

Moreover, the problem of filling the slot values is even more challenging. In this case it is not sufficient to guess a value from a narrow set. This problem consists in interpreting the user prompt and extracting the most important piece of information. This problem should rather be approached using text generation techniques.

This the exact idea behind the author's solution to the whole problem. In the rise of sequence to sequence language models the problem was approached with this exact technique.

Instead of training three separate models, only one is trained. During training, the expected output was concatenated into one sequence. Thus, the model was trained with the sequences in the format presented in Table II. The output of the model was then split into domain, intent and slot values.

In order to take full advantage of a sequence to sequence model it was necessary to use one created with sizeable training data. This was done in hope of achieving the best possible results especially in the task of filling the slot values. Apart from that, the model had to be multilingual as the data set contained sentences in English, Spanish and Polish. The language of the prompt was in fact annotated in the data but author's solution was aimed at providing a language-independent NLU module. The use of a large-scale model was also motivated by the fact that the input is sometimes distorted. Such models are known to deal well with the task of text generation even for noisy prompts.

At first, the FLAN-T5 [8] model was used. It was observed, however, that the results it renders fall below expectations for a specific technical reason. For Polish prompts, the outputs rendered by FLAN-T5 had erroneous Polish character encoding. This caused a significant and unnecessary drop in the quality measures of the CAICCAIC challenge. This motivated the switch to the Facebook mBART [9] model which yielded much better results altogether.

IV. Evaluation of the solution

As all submissions to the CAICCAIC challenge, the author's challenge was evaluated according to a detailed procedure described in [1]. The metric used to rank the submissions was Exact Match Accuracy (EMA), i.e. "the percentage of utterance-level predictions in which domain, intent, and all the slots are correct". Apart from that, the following additional metrics were reported for each submission:

- Domain accuracy (the percentage of utterances with correct domain prediction)
- Intent accuracy (the percentage of utterances with the correct intent prediction)
- Slot Word Recognition Rate (Word Recognition Rate calculated on slot annotations which is the percentage of correctly annotated slot values).

The evaluation scores for top five submissions are presented in Table III.

V. Error analysis

This section presents the analysis of some of the errors that the author's solution has committed. An error is counted when as per the Exact Match Accuracy metric. This means that error is reported when any of the labels is predicted incorrectly by the system.

A. Example: cold/weather

Command:

```
it is too cold in here
```

Expected output:

```
Airconditioner ChangeTemperature {}
```

System's output:

```
Weather OpenWeather {}
```

This example shows incorrect attribution of domain, intent and slot values. This error is caused by the confusion related to the word "cold" which can be associated with both air-conditioning and weather.

B. Example: temperature

Command:

```
20 degrees celsius would be ideal
temperature because it is too cold in here
```

Expected output:

```
Airconditioner SetTemperatureToValue
{'value': '20 degrees celsius'}
```

System's output:

```
Weather SetTemperatureToValue
{'value': '20 degrees celsius'}
```

This example shows incorrect attribution of the domain only. This is caused by the association of the word "temperature" with weather instead of air-conditioning. Based on this and the previous error example it is possible to conclude that the author's solution could benefit from a separate model to predict the domain only. Such model would have higher probability of predicting the domain correctly in these cases and this information could be used to affect the predictions of intent and slot values.

C. Example: expected result is not the only correct

Command:

```
give me information about my events
```

Expected output:

```
Calendar CheckCalendarOnDate {}
```

System's output:

```
Calendar OpenCalendar {}
```

The system's output is also acceptable.

TABLE II
EXAMPLE DATA FOR A TEXT2TEXT MODEL TRAINING

Input	Output
change the maximum temperature on my thermostat	Airconditioner ChangeTemperature {}
check the temperature on the keeping room sensor	Airconditioner GetTemperatureFromDevice {'device_name': 'keeping room'}

TABLE III
TOP FIVE SUBMISSIONS IN THE CAICCAIC CHALLENGE

Description	Slot WRR	Intent	Domain	EMA
author	0.87	0.92	0.96	0.75
flanT5-large	0.80	0.92	0.97	0.69
baseline	0.75	0.95	0.98	0.68
flanT5-large	0.77	0.90	0.95	0.67
flanT5-large	0.74	0.82	0.93	0.57

D. Example: minor mistake

Command:

```
update me when an appointment
in the calendar in location thornton begins
```

Expected output:

```
Calendar NotifyOnEventInLocation
{'location': 'thornton'}
```

System's output:

```
alendar NotifyOnEventInLocation
{'location': 'tornton'}
```

This error is a minor phonetic mistake.

VI. CONCLUSIONS

In general, the results achieved by the first five submissions prove the hypothesis that the problem of domain classification was relatively the easiest, intent label classification slightly more difficult and slot filling significantly more challenging than the first two problems. Also, the FLAN-T5 language model was a popular choice among the participants. This followed from the fact that the idea behind the Gonito.net platform is full collaboration between participants. Taking the solution of another participant, improving it even slightly and then uploading it as one's own is not only not forbidden but encouraged.

It can be observed that the author's solution scored significantly better in the most important EMA metric and on par with the best solutions in all other metrics. Best results in intent and domain classification individually were achieved by the baseline provided by the organizers of the challenge.

A good idea for future experiments would be a combination of solutions from the baseline and from the winning submission. Some errors identified during the error analysis process could also be corrected.

REFERENCES

[1] M. Kubis, P. Skórzewski, M. Sowański, T. Ziętkiewicz, "CAICCAIC: Centre for Artificial Intelligence Challenge on Conversational AI Correctness", *Proceedings of FedCSIS 2023*, 2023

[2] F. Graliński, R. Jaworski, Ł. Borchmann and P. Wierzchoń, "Gonito.net - Open Platform for Research Competition, Cooperation and Reproducibility" *Proceedings of the 4REAL Workshop: Workshop on Research Results Reproducibility and Resources Citation in Science and Technology of Language / Branco António, Calzolari Nicoletta* Paris, France, European Language Resources, 2016, pp. 13–20

[3] F. Graliński, R. Jaworski, Ł. Borchmann and P. Wierzchoń, "A semi-automatic method for thematic classification of documents in a large text corpus" *Mambrini Francesco, Passarotti Marco, Sporleder Caroline : Proceedings of the Workshop on Corpus-Based Research in the Humanities (CRH)*, Warszawa, Institute of Computer Science, Polish Academy of Sciences, 2015, pp. 13–21

[4] F. Graliński, Ł. Borchmann, R. Jaworski and P. Wierzchoń, "The RetroC challenge: How to guess the publication year of a text?" *Anatonacopoulos Apostolos (red.): DATeCH 2017: Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage,*New York, Association for Computing Machinery, 2017

[5] M. Faruqui and D. Hakkani-Tür, "Revisiting the Boundary between ASR and NLU in the Age of Conversational Dialog Systems". *Computational Linguistics* vol. 48 (1), 2022, pp. 221–232

[6] J. Lichtarge, C. Alberti, S. Kumar, N. Shazeer, N. Parmar and S. Tong. "Corpora generation for grammatical error correction." *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* Volume 1, 2019, pp. 3291–3301. https://doi.org/10.18653/v1/N19-1333

[7] L. Velikovich, I. Williams, J. Scheiner, P. Aleksic, P. Moreno, and M. Riley. 2018. "Semantic lattice processing in contextual automatic speech recognition for Google Assistant." *Proceedings of Interspeech*, 2018, pp. 2222–2226.

[8] H. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W Fedus et al. "Scaling Instruction-Finetuned Language Models", https://arxiv.org/pdf/2210.11416.pdf, 2022

[9] Y. Tang, C. Tran, X. Li, P. Chen, N. Goyal, V. Chaudhary et al., "Multilingual Translation with Extensible Multilingual Pretraining and Finetuning", https://arxiv.org/abs/2008.00401, 2022

[10] Z. Xie, A. Avati, N. Arivazhagan, D. Jurafsky, A. Ng, "Neural Language Correction with Character-Based Attention. ", https://arxiv.org/abs/1603.09727, 2016