Interspeech 2018
2-6 September 2018, Hyderabad

A Robust Context-Dependent Speech-to-Speech Phraselator Toolkit for Alexa

Manny Rayner[1], *Nikos Tsourakis*[1], *Jan Stanek*[2]

[1]University of Geneva, FTI/TIM, Switzerland [2]University of South Australia, Adelaide, Australia

Emmanuel.Rayner@unige.ch, Nikolaos.Tsourakis@unige.ch, Jan.Stanek@unisa.edu.au

Abstract

We present an open source toolkit for creating robust speech-to-speech phraselators, suitable for medical and other safety-critical domains, that can be hosted on the Amazon Alexa platform. Supported functionality includes context-dependent translation of incomplete utterances. We describe a preliminary evaluation on an English medical examination grammar.

Index Terms: Alexa, speech translation, medical applications, robustness, context-dependence

1. Motivation and background

A speech-to-speech phraselator is a speech-enabled system which contains a repertoire of phrases, each associated with translations in one or more target languages. The user speaks, and the phraselator attempts to find the best match between what is said and one of the phrases in its repertoire, showing the user what it has understood. If the user approves, the phraselator speaks the translation in the selected target language. For safety-critical domains, in particular medical domains, phraselators have not been rendered obsolete by general speech translation systems like Google Translate (GT). GT is known to be seriously inaccurate in medical situations; experiments carried out by our own and other groups suggest that it mistranslates 30–40% of all utterances [1, 2]. The problem is not so much the error rate as the fact that the only feedback given to the source language user, the recognition result, is misleading, since correctly recognised utterances can often be mistranslated. For these reasons, doctors are sceptical about systems like GT and more interested in phraselators, which are constructed to give completely reliable feedback.

In previous papers, [1, 3, 4], we have presented BabelDr, a speech-to-speech phraselator for medical domains currently being developed in a collaboration between Geneva University's Faculty of Translation and Interpretation and HUG, the University Hospital. BabelDr achieves good performance, with a semantic error rate of around 20% on unseen speech data for a repertoire of about two thousand phrases, but uses a complex architecture including proprietary components not easy to obtain or install. In the present paper, we describe a port of the phraselator to the popular Alexa platform, which otherwise uses only open source material. The core of the system is an efficient robust parser which uses a combination of tf-idf and dynamic programming to find the best match between output from a large-vocabulary speech recogniser and the defined set of phrases. Translation rules are written in the same Synchronous Context Free Grammar (SCFG) notation as in the previous version and compiled into tables which can be uploaded to Alexa. A novelty is that the robust matching method has been extended so that it also takes preceding dialogue context into account, making it possible to respond to incomplete utterances.

The rest of the paper is organised as follows. Section 2 describes the phraselator functionality, focusing on new material that has been added over the last year, Section 3 describes the Alexa-based version, and Section 4 presents a preliminary evaluation using an English medical examination grammar.

```
Utterance
Source have you $had_or_felt \
       $stomach_pain $$for_period
Source has it ( hurt | been hurting ) \
       $$for_period
Target/english Have you had  \
       stomach pains $$for_period ?
Target/french avez-vous mal au ventre \
       $$for_period ?
EndUtterance

Phrase
PhraseId $had_or_felt
Source ( had | felt | experienced | \
         been having | been feeling | \
         been experiencing )
EndPhrase

TrLex $$for_period \
      source="for a long time" \
      english="for a long time" \
      french="depuis longtemps"

TrLex $$for_period \
      source="for a few hours" \
      english="for a few hours" \
      french="depuis plusieurs heures"
```

Figure 1: *Sample translation rules for the question-schema "Have you had stomach pain for [period]?".*

2. Functionality

The intention is that a set of phraselator rules will be structured as a "flat" SCFG grammar, where there is one top-level rule for each phrase. A rule will typically specify many ways to speak the phrase, and will include nonterminals for common sub-phrases. These nonterminals can be of two types. "Plain nonterminals", written with a single $ sign, specify permitted variations on the source side; "translation nonterminals", written with a double $ sign, specify SCFG rules which associate a set of source phrases with a phrase in each target language. Figure 1 shows an example of a top-level rule and some associated rules for non-terminals.

At runtime, recogniser output is matched against grammar rules using the efficient bottom-up dynamic programming algorithm described in [4]. Two main new features have been added compared to the version from the previous paper. First, a sim-

ple mechanism is included to handle out-of-vocabulary (OOV) words: this finds the closest in-grammar word on a character-based edit-distance metric, substituting it for the OOV word if the proportion of letters in common is greater than a user-defined threshold. The method is made efficient by including a preprocessing phase which uses a cosine distance method to create a shortlist.

Second, and more interestingly, the robust matching method has been generalised to allow translation of incomplete utterances. The top-level matching function can optionally take input both from the recogniser output string and from a second string, which represents the context; the most straightforward way to define the "context string" is to let it be the matched string from the preceding utterance. When the matching algorithm takes a word W from the context string, it multiplies the match score for W by a user-defined discounting factor $k_{context}$; after all matching has completed, an additional term is added to the global score which consists of the word edit distance between the new match string and the context string, multiplied by a second user-defined constant $k_{parallel}$.

Despite the extreme simplicity of the method, we have been surprised to find that it works quite well; initial testing on artificial spoken test suites gives error rates for incomplete utterances about 1.5 times higher than for complete utterances. The increase is partly explained by the fact that speech recognition is considerably less accurate on incomplete utterances, with Word Error Rate increased by about the same factor. The scores are not sensitive to the settings of $k_{context}$ and $k_{parallel}$, as long as $k_{parallel}$ is small compared to 1, and $k_{context}$ is small enough that words taken from the context always receive lower scores than words taken from the current sentence. This work is fully described in a paper currently under submission.

3. A phraselator toolkit for Alexa

The toolkit which is the main subject of this paper supports rapid construction and deployment of speech-to-speech phraselators on Alexa, using the basic architecture described in the previous section. It consists of the following components:

Grammar compiler A script which converts an SCFG grammar description into the following outputs: i) Tables, encoded as three files of Python dictionaries in pickled gzipped form, which encode data for robust parsing, translation, and looking up recorded audio translations; ii) A corpus produced by random sampling of the grammar. This is used to adapt Alexa recognition to the domain, which makes a large difference to recognition quality; iii) A spreadsheet listing the target language audio files that need to be recorded. These can be created using a third-party TTS engine, e.g. Amazon Polly.

Robust parser A piece of Python code which uses the tables produced by the grammar compiler to find an n-best list of closest matches between a ⟨String, Context⟩ pair and the grammar rules. There are two entry points: a "load" function and a "match" function.

Offline testing tool A piece of Python code which processes a spreadsheet of utterances using a currently loaded compiled grammar. Each line consists of an utterance and a gold standard annotation. Output is formatted as another spreadsheet.

Sample Alexa phraselator app A "lambda-function", implemented in Python, and a JSON "interaction model" definition, which together give a minimal example of the top-level Alexa content needed to specify a generic phraselator app. The app listens for a phrase, finds the best match against the grammar, and speaks the back-translation. If the user then says "proceed", it speaks the translation using a recorded audio file.

Sample grammars Two sample grammars. The first is a toy grammar with a handful of greeting phrases; the second is the English to French medical examination grammar from [3], which contains rules for 651 top-level phrases.

Documentation There is online documentation [5].

The above resources can be used in a straightforward way to build and deploy a high-performance Alexa phraselator app. The lambda-function loads the tables produced by the compiler at init time: the highly optimised nature of Python's `pickle` package means that loading is fast, typically a few seconds.

4. Preliminary evaluation

We would like to evaluate our large French BabelDr grammar on Alexa and compare performance against the version of [4], but this was not possible for the current paper; Amazon only began supporting French language apps on June 13 2018. As an interim measure, we carried out a simple evaluation using the smaller English BabelDr grammar from [3]. We created a corpus of 160 English BabelDr sentences, of which 110 were in grammar coverage and 50 not in grammar coverage (this ratio of in-coverage to out-of-coverage is similar to the one in [4]). We tagged each out-of-coverage example with the distance in words to the closest in-coverage sentence. We then asked five subjects, four English native speakers and one fluent non-native[1], to read each sentence in the corpus twice to the app in a natural voice, noting down backtranslations as "fully correct", "partially correct" and "incorrect", yielding a total of 1600 examples. For in-coverage utterances, 95.5% were fully correct and 2.2% partially correct; for utterances that were 1 or 2 words out-of-grammar, the scores were 72% fully correct and 6% partially correct; for utterances that were 3 or more words out-of-grammar, 62% were fully correct and 13.5% partially correct. With the obvious provisos, these results seem to us quite promising, and not worse than [4].

5. References

[1] P. Bouillon, J. Gerlach, H. Spechbach, N. Tsourakis, and S. Halimi, "BabelDr vs Google Translate: a user study at Geneva University Hospitals (HUG)," in *Proceedings of EAMT 2017*, Prague, Czech Republic, 2017.

[2] S. Patil and P. Davies, "Use of Google Translate in medical communication: evaluation of accuracy," *BMJ*, vol. 349, p. g7392, 2014.

[3] F. Ahmed *et al.*, "A robust medical speech-to-speech/speech-to-sign phraselator," in *Proceedings of Interspeech 2017*, Stockholm, Sweden, 2017.

[4] M. Rayner, N. Tsourakis, and J. Gerlach, "Lightweight spoken utterance classification with CFG, tf-idf and dynamic programming," in *International Conference on Statistical Language and Speech Processing*. Springer, 2017, pp. 143–154.

[5] M. Rayner and N. Tsourakis, *Building Alexa Apps with the Regulus Lite Speech2Speech Platform*, https://www.issco.unige.ch/en/research/projects/Speech2SpeechDocAlexa/build/html/index.html, 2018, online documentation.

[1] We would like to thank Cathy Chua, Yasmin Gomes and Yolanda Gomes for kind assistance with data collection.