An Application for Building a Polish Telephone Speech Corpus

Bartosz Ziółko[1,2], Piotr Żelasko[1], Ireneusz Gawlik[1,3], Tomasz Pędzimąż[1,2], Tomasz Jadczyk[2]
[1]AGH University of Science and Technology, Poland,
Department of Computer Science, Electronics and Telecommunications,
al. Mickiewicza 30, Kraków, Poland, www.dsp.agh.edu.pl
[2]Techmo, Kraków, Poland, techmo.pl
[3]Grupa Allegro Sp. z o.o., Poland
bartosz.ziolko@techmo.pl, pzelasko@agh.edu.pl, igawlik@live.com, {pedzimaz, jadczyk}@techmo.pl

Abstract

The paper presents our approach towards building a tool for speech corpus collection of a specific domain content. We describe our iterative approach to the development of this tool, with focus on the most problematic issues at each working stage. Our latest version synchronizes VoIP call management and recording with a web application providing content. The tool was already used and applied for Polish to gather 63 hours of automatically annotated recordings across several domains. Amongst them, we obtained a continuous speech corpus designed with an emphasis on optimal phonetic diversification in relation to the phonetically balanced National Corpus of Polish. We evaluate the usefulness of this data against the GlobalPhone corpus in the task of training an acoustic model for a telephone speech ASR system and show that the model trained on our balanced corpus achieves significantly lower WER in two grammar-based speech recognition tasks - street names and public transport routes numbers.

Keywords: speech recognition, language resources, corpus recording tools

1. Introduction

Speech corpora acquisition is a significant problem in voice applications development, however, its magnitude is often under-appreciated. In this paper, we address this problem by exploring different means of how speech recordings can be collected. The process we describe is iterative - as we will show, at each iteration the previous approach is scrutinised and improved upon, which allows for acquisition of the recordings on a larger scale than possible before, while improving user experience, and thus the quality of the data collected.

Specifically, we describe a tool designed to improve our Automatic Speech Recognition (ASR) system performance, specialised for conversations over telephone - SARMATA (Ziółko et al., 2015). To that end, we focus on improving the training corpus by collecting more and more data, as Polish is still quite limited in the availability of speech corpora regarding their sizes and diversity. Our main assumptions were:

- training data for telephone ASR should be recorded through a telecommunication channel in order to condition the ASR on the channel effects met in a production environment,
- speech content should be as diversified as possible,
- it is faster and more cost efficient to produce speech for a given transcription than to annotate existing recordings.

Our ASR is based on a Deep Neural Network (DNN) architecture (Ziółko et al., 2015). At the moment, we are able to train the DNN with approx. 4 million parameters on as little as around 100 hours of speech. However, the de facto standard in the industry is to use sizeable training corpora, which total durations are measured in hundreds or even thousands of hours (Amodei and others, December 2015; Xiong and others, January 2017). Therefore, we expect to achieve better model generalisation and word error rates (WER) with more data, and thus aim in training SARMATA on at least 1000 hours of speech. To achieve this goal without sacrificing quality of the data, we chose to record sentences from National Corpus of Polish (NCP) (Przepiórkowski et al., 2012) and parts of Polish Wikipedia, choosing phrase subsets with optimal phonetic diversification. This process is described in details in section 3.1.

2. Early Development Stage

The present discussion has focused on the collection of a large quantity of continuous speech recordings. However, our first attempts at recorded speech collection were focused on a much simpler case, i.e. isolated digits, keywords and commands recognition, as described in (Żelasko et al., 2016). During that time, we used a free VoIP softphone application to call speakers whom we wanted to record, and manually annotated the recordings. Not surprisingly, this approach did not yield a considerable amount of data, and was not satisfactory in the long term.

In preparation for the next iteration, in which we wanted to record about 1200 different street names, we established that the recordings have to be automatically annotated for our approach to be viable. To this end, we developed a simple C++ VoIP call recorder application, which could be called by a person to be recorded. At the beginning of the call, the user listened to a pre-recorded invitation prompt with instructions and asked to prepare the list of street names he wants to record and select appropriate number on the DTMF keyboard. The street names were separated into 12 disjoint sets, each sized at about a hundred utterances, and so each key on the keyboard could be assigned to a different phrase set. Upon selecting the phrase set, the user had to read the entire list in correct order, and wait for a beep between each street name. The recording

was then automatically cut at the moments where beep was played and annotated according to the ordering of the items in a given set.
We need to bear in mind, that the timing was crucial in this process, as long waiting times between beeps can quickly become exhausting for the speaker, yet on the other hand, giving too short an interval for reading led to prematurely and incorrectly cut recordings. We observed that setting the time interval between beeps as constant was found unsatisfactory, because some street names could be very long, e.g. "Osiemnastego Bielskiego Batalionu Desantowo Szturmowego" (eng. 18th Bielski assault and landing battalion), while others are very short, e.g. "Agawy" (eng. "Agave street"), which in turn led user waiting for a very long time between each beep. We solved the problem by developing a heuristic for predicting how much time T should pass between beeps, given by a linear function of the number n of non-whitespace characters in the utterance. In practice, we found that using an offset of 1.5 seconds and adding approx. 300 milliseconds per syllable allowed most of our speakers to read at an optimal pace. We approximated the number of syllables as half the number of characters in an utterance. The relation is given by

$$T = 1500 \text{ ms} + n * 150 \text{ ms} . \quad (1)$$

Using this application, we gathered a total of several hours of streets names recordings. However, it was still far from perfect and had several problems. Firstly, we still needed to manually verify that the data was properly cut and annotated, and fix the mistakes. Secondly, the process was confusing for some speakers, causing them to wait for a few beeps, which introduced offsets in some of the annotations, which were later corrected. Also, due to the simplicity of the application, only a single speaker could be recorded at a time. Seeing as the process of data acquisition was still largely manual (including the selection of phrase set and addition of new recordings to the corpus), we designed and implemented a more robust solution.

3. The Application

The first major version of our tool consists of a front-end web application and a back-end server application. The front-end part is capable of displaying a previously prepared sentence set to the user, and establishing a phone call between the server and the users phone, enabling the user to dictate the phrases over the phone.
Each user, upon entering the website, is led through the following steps:

- Webform, allowing to enter users nickname, telephone number and gender. The user also chooses the phrase set to be recorded (see Fig. 1). The user may also access this screen through a specially crafted URL with pre-filled data, so that he does not have to fill any field.
- Initial invitation screen, with instructions related to the recording process. (see Fig. 2).
- Terms of usage screen, with necessary agreement statement. That screen includes a "Call me" button (see Fig. 3).
- Screen, which informs about ongoing call. User is presented with the phone number used to call them. (see Fig. 4).
- After the call is answered, the screen automatically changes to display the phrases to be read (see Fig. 5). To advance to the next phrase, the user can click on the "Next" button or press space bar. The recorded utterance is stored on the machine, where the back-end server application is being run.

Many speakers misspoke sentence or react emotionally to read texts. This generated additional work on the verification of the quality of recordings and as a result reduced the amount of received recordings. Most speakers were aware of committed mistakes. In order to solve this issue, we gave the speakers an option to repeat the phrase or to go back to the previous one. Both actions result in overriding previously recorded phrase on the server.
In order not to burden our users with the payment for the recording session, we decided to take advantage of VoIP service providers, allowing us to call the speaker directly. Due to that, the costs of the calls were covered from the corpus maker side and much lower due to a large scale payment plan.
Back-end part of the application was implemented in C++. In order to be able to connect with most of the Polish VoIP service providers, we chose Session Initiation Protocol (SIP) as our call handling protocol, and Real-Time Transport Protocol (RTP) for transfer of audio packets. The audio packets encoding varies for each call session, as it is negotiated between call endpoints, and in our case is one of: G.722 HD, G.711u, G.711a, G.729, G.726-32, G.722, iLBC or GSM - however, it is possible that audio packets are transcoded at a node between call endpoints, which results in a diverse range of channel modifications in the obtained recordings. Multiple recording sessions are allowed, by means of thread-per-session approach. Front-end content is served statically, and communication between back-end and front-end parts relies on XHR calls. REST interface, needed for communication with application front-end part, is implemented with the use of Mongoose, open source, embedded C++ HTTP server library (Hammel, 2010).
Front-end part is implemented as a Single Page Application (SPA), using AngularJS 1.4 library (Jain et al., 2014). We took into consideration unexpected events, such as call failures. Persistent state exchange from back-end to front-end is implemented using AJAX polling technique. Use of such technique allows us to detect closing of browser window, and in effect ending the call with server side event. To avoid the abuse of the service, a lack of interaction with front-end part also results in ending the call.

3.1. Speech diversification

It has been shown that phonetically balanced training corpora tend to improve the efficacy of the resulting ASR system (Uraga and Gamboa, 2004) (Wang, 1998). In order to acquire this property, we extracted optimal phonetically diversified phrase sets from the NCP and Polish pages of

Figure 1: The application starts with collecting information about the speaker (from the top): the nickname, phone number, gender and the choice of a phrase set. All the information can be also inserted by a link to speed up the process.

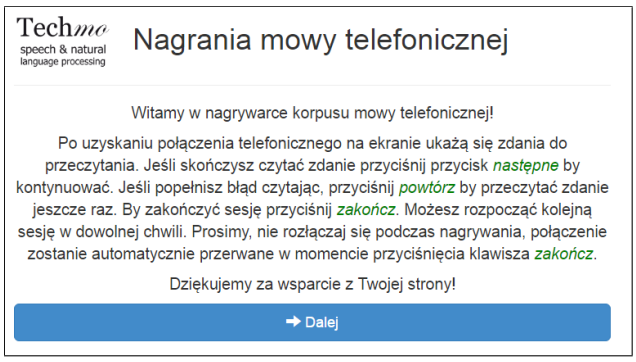

Figure 2: After user fills the questionnaire, a short explanation of how the application works and what is expected from the speaker is displayed.

Wikipedia. As such, a relatively small subcorpus of selected phrases had retained phonetic distribution which is close to the distribution of the whole corpus. We have decided to implement two approaches for records gathering.
In the first one, we created a huge corpus of randomly selected phrases, and let multiple users iterate over following sentences for as long as they were willing to. When multiple speakers were recording, each one has received a sentence that was enqueued in a circular phrase-list. That approach, however, did not guarantee that the built corpora would be balanced both in terms of voice diversity as well as phonetically balanced for each speaker.
In the second approach, we prepared phonetically balanced phrase subsets of fixed size by selection of specific sentences. However, simple approach based on random subsets selection of feasible size (less than 10 minutes of continuous speech) did not retain expected phonetic properties.

Figure 3: The user is informed that the recordings may be further processed, demonstrated and sold as part of a corpus, without revealing information about the users identity. In order to proceed, the user has to agree.

Figure 4: Information that the user is being called by the server (after he agrees to the terms of service).

To overcome this issue, we developed a subset selection heuristic algorithm (see Algorithm 1), allowing us to prepare diverse, phonetically balanced phrase subsets for each speaker.
In the proposed approach, we transformed each NCP sentence into a bag of phonemes (BOP) representation, so that each sentence was represented by a fixed length phoneme count vector. Generated phrase subset was initialized with a single, randomly selected sentence. Then, we iteratively expanded the subset with new sentences. The choice of the new sentence is based on phonetic similarity, so that the BOP of each selected sentence closely matches the difference in phoneme distribution between NCP and current state of the subset, by means of cosine measure. Additional constraint on sentence length was applied, reducing bias towards short sentences.

3.2. Issues encountered during corpus collection

We encountered some issues, which resulted in some parts of our acquired data to be unnatural. The first one was a result of wrong gender inflection in sentences displayed to speakers, i.e. males were sometimes given female sentences and the other way around. The second source of unnatural recordings was unusually large number of swear

Figure 5: The main working screen consists of a display of the phrase to be read (here 'Stare Miasto'), information about the next phrase (here 'Grzegorzki') and steering buttons - from the left (previous, repeat and next) and the large red one - finish.

Algorithm 1 NCP subset selection procedure

```
function SELECTNCPSUBSET(corpus, subsetSize)
    BOPSentences ← TOBOP(corpus)
    Subset ← EMPTYSET()
    Initial ← RANDOM(BOPSentences)
    INSERT(Subset, Initial)
    while SIZE(Subset) < subsetSize do
        Expected ← PHONETICDISTDIFF(Subset, BOPSentences)
        Selected ← LOOKUP(BOPSentences, Expected)
        INSERT(Subset, Selected)
        REMOVE(BOPSentences, Selected)
    end while
end function
```

words in NCP. Two speakers actually stopped participation in the recordings, because they refused to read such phrases. Some other speakers encountered problems while reading unusual words, especially archaic ones.

A possible field for improvement is automatic progress to a next phrase. In our application it was arranged by a click of a button while it could be arranged automatically by silence detection and (or) on-line recognition. Also, we did not provide automatic check of the content by application of ASR.

The application was created for collection of a relatively small corpus. Due to that it did not provide an efficient user and content control tool. For our goals it was not necessary, but it could prove problematic in a large, speaker-diversified corpus acquisition scenario.

We did not control the environment of the speakers we were recording. Even though we asked them to record in silent conditions some of the recording where done with background noise.

We also considered recording with a separate WebRTC channel through a PC microphone, in parallel to telephone call. This would have provided us with the same speech recording, transferred via different media. At the time, we considered it to be too complex and in the first version we decided not to do it.

We plan to address those issues in the next iteration of our tool.

4. Results

4.1. Collected resources

We managed to gather several corpora using various versions of the tool. Most notably, we collected a general-purpose corpus of continuous speech from the NCP and Wikipedia phrase sets, which consists of 15600 utterances, uttered by a total of 163 speakers (63 female and 100 male). This is more than 25 hours of continuous speech.

Other collected corpora are domain-specific. The Commands corpora consists of utterances which often appear in Interactive Voice Response (IVR) scenarios, such as basic commands (e.g. "*go back*"), numbers and dates, as well as more specific vocabulary, such as names of medical specializations. The *Bus* corpus is a collection of recordings of the bus line numbers in Cracow. The only large collection we gathered before application 1.0 is the *Streets* corpus, as mentioned in section 2.. Details about those corpora are presented in table 1.

Corpus	Speakers	Utterances	Duration	Size[MB]
Continuous	163	15685	25:05	2860
Commands	72	10846	09:05	1166
Bus	130	17991	17:21	1969
Streets	47	10507	12:09	1380
Total	434	55029	63:40	7375

Table 1: Detailed information about sets of collected recordings. The duration is given in [hh:mm] format.

	GlobalPhone	Continuous	Both
Streets	50.41%	5.21%	3.59%
Bus	51.97%	11.47%	8.29%

Table 2: Word error rates (WER) of the SARMATA ASR system trained exclusively on: I. GlobalPhone; II. Continuous corpus; III. Sum of GlobalPhone and Continuous. The test corpora are the Streets and Bus corpora.

4.2. Speech recognition

In order to ascertain the quality of gathered recordings, we incorporated them into training of our SARMATA ASR (Ziółko et al., 2015). The baseline system was trained using the Polish training subset of the GlobalPhone corpus (Schultz, 2002). Second variant of the system was trained on the *Continuous* corpus, and finally the third variant was trained on both of these corpora. For evaluation, we used previously mentioned *Bus* and *Streets* corpora. The results are shown in table 2.

Training our system on the *Continuous* corpus resulted in a major improvement of the word error rate (WER) during evaluation. We attribute this result to several factors. Firstly, our corpus consists only of the telephone speech, and GlobalPhone, contrary to its name, does not. This results in a mismatch between the spectral characteristics of the training and evaluation recordings in case of a system trained on the GlobalPhone. Another reason is the larger amount of data available in the *Continuous* corpus - 25 hours versus GlobalPhone's 15 hours. Finally, our recordings are of higher quality and generally not as noisy as those encountered in GlobalPhone.

Combining both GlobalPhone and *Continuous* corpora introduced additional diversity in the recordings, further improving the ASR acoustic model generalisation and reducing WER.

5. Conclusions

We presented an iterative process of designing a tool for automated corpus collection. Throughout its development, we observed and addressed subsequent issues resulting from automating the annotation process. Current version of the tool provides a user-friendly approach towards recording continuous speech. As a result of our work, we were able to obtain 63 hours of quality recordings, which we use both for training of the acoustic models and for ASR evaluation. We observed a significant WER improvement with regard to the baseline system evaluated on telephonic speech.

6. Acknowledgments

This project has received funding from the European Union's Horizon2020 research and innovation programme under grant agreement No 761349.

7. Bibliographical References

Amodei, D. et al. (December 2015). Deep speech 2: End-to-end speech recognition in english and mandarin. *arXiv*.

Hammel, M. (2010). Mongoose: an embeddable web server in c. *Linux Journal*, 2010.

Jain, N., Mangal, P., and Mehta, D. (2014). Angularjs: A modern mvc framework in javascript. *Journal of Global Research in Computer Science*, 5(12).

Wang, H.-m. (1998). Statistical analysis of mandarin acoustic units and automatic extraction of phonetically rich sentences based upon a very large chinese text corpus. *Computational Linguistics*, 3(2):93–114.

Xiong, W. et al. (January 2017). The microsoft 2016 conversational speech recognition system. *arXiv*.

Ziółko, B., Jadczyk, T., Skurzok, D., Żelasko, P., Gałka, J., Pędzimąż, T., Gawlik, I., and Pałka, S. (2015). SARMATA 2.0 automatic Polish language speech recognition system. *Interspeech, Dresden*.

8. Language Resource References

Przepiórkowski, A., Bańko, M., Górski, R., and Lewandowska-Tomaszczyk, B. (2012). *Narodowy Korpus Jezyka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.

Schultz, T. (2002). Globalphone: A multilingual speech and text database developed at Karlsruhe University. In *Proceedings of the ICSLP*, pages 345–348.

Uraga, E. and Gamboa, C. (2004). Voxmex speech database: design of a phonetically balanced corpus. In *Fourth International Conference on Language Resources and Evaluation*.

Żelasko, P., Ziółko, B., Jadczyk, T., and Skurzok, D. (2016). AGH corpus of Polish speech. *Language Resources and Evaluation*, 50:585–601.