Pronunciation Learning for Named-Entities through Crowd-Sourcing

Attapol T. Rutherford[†] * *Fuchun Peng*[‡] *Françoise Beaufays*[‡]

[†]Brandeis University, Waltham, MA, USA
[‡]Google Inc., Mountain View, CA, USA

Abstract

Obtaining good pronunciations for named-entities poses a challenge for automated speech recognition because named-entities are diverse in nature and origin, and new entities come up every day. In this paper, we investigate the feasibility of learning named-entity pronunciations using crowd-sourcing. By collecting audio samples from non-linguistic-expert speakers with Mechanical Turk and learning from them, we can quickly derive pronunciations that are more accurate in speech recognition tests than manual pronunciations generated by linguistic experts. Compared to traditional approaches of generating pronunciations, this new approach proves to be cheap, fast, and quite accurate.

1. Introduction

The pronunciation lexicon is a key component in a speech recognition system. If the pronunciation for a word is wrong, the recognizer will systematically misrecognize that word, which can result in a very frustrating user experience. This is particularly noticeable in mobile applications that let users search for and retrieve information [1, 2]. Named-entities are frequently requested in such applications, and these often include complicated words for which the pronunciation models are least accurate.

One reason for this weakness is that pronunciation dictionaries are traditionally developed by human experts who write word pronunciations by hand. While this process typically results in high quality entries, it is slow, expensive, and hard to scale to all the words that make up local entities, mobile application names, artist names, brand names, and all other entities that can be queried by users, in all languages.

If a named-entity is not in the expert-generated dictionary, its pronunciation is typically created by an automated pronunciation engine such as a grapheme-to-phoneme (G2P) converter [3]. However, the quality of G2P pronunciations highly depends on how similar the word in question is to the engine's training data. Thus, the conversion typically works well on words that are very regular, but is less satisfactory for words with greater orthographic variation, which unfortunately is often the case for named-entities.

In our experience, both manual pronunciation generation and G2P conversion tend to quickly break down on uncommon words. For example, we observed that trained linguists may slow down by a factor of five when presented with infrequent words from tail search queries compared to processing common words. They often also struggle with words of foreign origin: should they follow the phonology of the target language or that of the original language? Should they take the frequency of the word into account? Moreover, when we tried to equip our linguists with acoustic samples of the words for which we needed pronunciations, the work slowed down even more: random people let themselves be influenced by various factors when they speak a word and don't think about it too much, trained linguists instead naturally try to rationalize their decisions, which makes the process extremely slow. On the other hand, G2P algorithms generalize from their training data: they typically do not take into account morphology or etymology and might for instance pronounce the Chinese name "Fuchun" as "f ah ch ih n", which is far from the correct and intuitive pronunciation "f u ch u n".

The challenge remains: is it possible to generate pronunciations quickly, yet at least as accurately as human experts? In this work, we explore crowd-sourcing as a potential solution to this problem.

Crowd-sourcing sites such as Amazon Mechanical Turk (MTurk) open new possibilities for generating speech and language data in a scalable and cost-effective fashion. Within recent years, MTurk has gained more attention from the speech and natural language processing community as it allows developers to collect fresh data in large amounts and with high frequency [4]. MTurk provides a convenient platform for anyone (referred to as a *Turker*) to complete tasks online and be compensated for their time. In practice, the quality of the data obtained through this channel may vary and needs to be verified before the data can be utilized [5].

Here we propose that pronunciations can be learned from the audio data generated by non-linguistic-expert speakers recruited through Mechanical Turk. Striking a balance between automatic and manual solutions to pronunciation learning, we cheaply crowd-source the data acquisition and then automatically extract best fitting pronunciations from the audio data. This hybrid approach significantly enhances the performance of the speech recognizer and proves to be a feasible and scalable way to learn pronunciations, both for common and less-common words not covered in the pronunciation lexicon.

It should be added that our focus is *not* on learning pronunciations for specific subgroups of users, such as people with some accent, or on specific conditions, e.g. fast speech vs. careful speech; our focus is on learning generic pronunciations that will, broadly-speaking, improve the experience of our users. We are thus making the implicit assumption that the wide pool of Turkers used to collect the acoustic data roughly matches the population who will be consuming the learned pronunciations.

2. Related Work

Learning pronunciations from audio data is not a new concept [6, 7], and it has recently seen more advancement. For example, McGraw et al. [8] learn pronunciation with a mixture model and update pronunciation weights with the Expectation-

*Worked on this project during his internship at Google.

Maximization (EM) algorithm. Similarly, Lu et al. [9] use a seed lexicon and an iterative optimization method for updating weights, finding Viterbi approximation to outperform full EM optimization. Using the Viterbi approximation is equivalent to performing a forced alignment of the acoustic data to their corresponding transcript. Li et al. [10] incorporate audio into G2P learning. Lei et al. [11] use audio data to expand a Chinese pronunciation lexicon.

Laurent et al. [12, 13] use an iterative approach to extract phone and transcript alignment for entities. The basic concept behind our paper is similar: extracting pronunciations from phone alignments derived from audio data. Instead of extracting the subsection of audio data aligned to entities, we align the whole audio with the whole transcript.

Crowd-sourcing is a very active field of research which spans across many subdisciplines and applications such as natural language processing, computer vision, and speech recognition. In particular, several papers investigate the feasibility of crowd-sourcing acoustic data collection. Mechanical Turk, for example, has been used to collect speech samples to narrate Wikipedia articles for vision-impaired users [14], to create speech corpora for specific domains [15], and to collect data for low resource languages [16].

As far as we know, this is the first publication on investigating the feasibility of learning named-entity pronunciations to improve a state-of-the-art automated speech recognition system through crowd-sourcing.

3. Pronunciation Learning Algorithm

In speech recognition, we wish to find the word sequence W^* that maximizes the likelihood of the acoustic observations, $\mathcal{X}$

$$W^* = \arg\max_i P(W_i|\mathcal{X}) \quad (1)$$

$$= \arg\max_i P(\mathcal{X}|W_i)P(W_i). \quad (2)$$

Assuming that different phone sequences S_j^i can underlie the same word sequence W_i (pronunciation dictionaries often allow multiple pronunciations per word), and with the Viterbi approximation, Eq. 2 becomes

$$W^* = \arg\max_{i,j} P(\mathcal{X}, S_j^i|W_i)P(S_j^i) \quad (3)$$

$$= \arg\max_{i,j} P(\mathcal{X}|S_j^i)P(S_j^i|W_i)P(W_i). \quad (4)$$

For the purpose of pronunciation learning, we instead assume that the word sequence W_i corresponding to the acoustic sequence $\mathcal{X}$ is given, but multiple pronunciations are available. We wish to find the pronunciation sequence S^* that maximizes the likelihood of the acoustic data, under the assumption of the given word sequence:

$$S^* = \arg\max_j P(\mathcal{X}|S_j^i)P(S_j^i|W_i)P(W_i) \quad (5)$$

$$= \arg\max_j P(\mathcal{X}|S_j^i)P(S_j^i|W_i) \quad (6)$$

$$= \arg\max_j P(\mathcal{X}|S_j^i). \quad (7)$$

where $P(S_j^i|W_i)$ can be dropped if we assume equal priors on the pronunciation sequences.

If several training audio sequences, $\mathcal{X}_k$, are available for a given word sequence W_i, a pronunciation S_k^* can be derived for each training sample ($\mathcal{X}_k$, W_i). The pronunciation distribution so obtained can be reduced to one or more chosen pronunciations by a voting scheme: pick the few most commonly chosen pronunciations S_k^*, or by summing over the audio samples: $\sum_k \max_j P(\mathcal{X}_k|S_j^i)$.

The pronunciation learning pipeline we propose consists of the following steps:

1. Collect audio samples for each word sequence:
   For each word or phrase for which we wish to learn a pronunciation, we elicit speech utterances from a crowd-sourcing platform such as Mechanical Turk.
2. Generate pronunciation candidates for each word:
   For each word in the batch of words or phrases we wish to learn, we use a standard G2P algorithm [3] to generate a number of candidate pronunciations. We found 20 pronunciations to be a good number.
3. Extract best-fitting pronunciations:
   Based on Eq. 7, we force-align all the text/audio pairs we have collected, using a dictionary that has been augmented with all the candidate pronunciations generated in the previous step.
4. Select pronunciations:
   Aggregating across all the training phrases, we obtain the distribution of pronunciations that best fits each word. We then select one or more pronunciations from the distribution and introduce them to the recognition dictionary. We experimented with a few options to select pronunciations, and found that just chosing the best learned pronunciation was about just as good as other reasonable choices.

We call this method FG2P, for force-align grapheme-to-phoneme converion. This algorithm is extremely simple and scales easily. Audio data can be acquired quickly and cheaply. Mechanical Turk, for example, can yield 10 utterances for each of 1000 English queries within hours at the cost of a few cents per utterance. Rapid data acquisition makes it possible to update the pronunciation dictionary on a daily basis. The approach also scales to the many languages for which the crowd-sourcing platform provides coverage.

4. Experimental Setup

We used Google's Voice Search production recognition engine as the speech recognizer to perform the force-alignment process described in Eq. 7 and to perform all of the experiments reported in this paper. This is a standard large-vocabulary state-of-the-art speech recognizer with Deep Neural Network (DNN) acoustic models [17], a Finite State Transducer (FST) decoder [18], and a standard 5-gram language model trained on a variety of text corpora. The language model is optimized for the type of traffic experimented with in the paper, so that out-of-vocabulary (OOV) words are quite infrequent and are not a limiting factor to recognition accuracy. Experiments were conducted in American English.

4.1. Evaluation Metrics

In addition to word error rate (WER) and sentence accuracy (SACC) measurements, we evaluated the impact of pronunci-

ation variants by performing side-by-side (SxS) tests [1]. In these tests, two recognition engines are contrasted, one without the learned pronunciations and one with them. The two engines are used to re-decode speech data extracted from Voice Search anonymized logs. Queries where the two recognition results differ are evaluated by human raters and classified into one of four categories:

- non-sense: the transcript is a complete non-sense.
- unsable: the transcript is sensical but does not correspond to the audio.
- usable: the transcript contains only small errors.
- exact: the transcript matches the spoken audio exactly.

A positive experiment will show decreasing fractions of non-sense and unusable queries and increasing fractions of usable and exact queries.

5. Experiments on Popular Entertainment Entities

In these experiments, we created four entertainment-related data sets by selecting the 1,000 most downloaded entries from Google Play Store for each of the following categories: *artist names*, *song titles*, *television show titles*, and *movie titles*. These entries contain popular proper names or potentially new words that might not be accounted for by a standard pronunciation dictionary. We acquired ten utterances for each entry from ten different English speakers on the MTurk platform. Seven of the utterances of the same transcription were used for pronunciation learning, and the other three were used for testing.

5.1. Configurations

We established two baselines. The first one, referred to as *Lexicon + G2P*, uses the production pronunciation engine, where each word is either listed explicitly in an expert-generated lexicon file, or is given a single G2P-generated pronunciation. The second baseline instead, referred to as *Lexicon + Manual*, augments the production lexicon file with manually derived pronunciations created by expert linguists for the relevant words. These two baselines are compared to the following two new pronunciation configurations: *FG2P*, where each word is paired with the most frequent learned pronunciation, and *Manual + FG2P*, which include both learned and manual pronunciations. Language model tokens that are not part of the experiment receive baseline *Lexicon + G2P* pronunciations.

5.2. Results

Table 1 summarizes the results of different systems on the four test sets. Several observations can be made. First, between the two baselines, we can see that *Lexicon + Manual* strongly outperforms *Lexicon + G2P*. This indicates that indeed the G2P engine doesn't perform well on these named-entities. Second, FG2P outperforms *Lexicon + Manual* across all of the data sets. The relative WER improvement for the artist, movie, song, and TV show data sets are 3.1%, 1.6%, 5.0%, and 1.4%, respectively. This suggests that FG2P provides a good substitute to manual annotation efforts. Part of the reason for this is that the algorithm benefits from access to audio samples while the expert linguists do not, for speed reasons as discussed in the introduction. For example, FG2P learned for the word "Bexar", a Texas county, the (apparently correct) pronunciation "b eh r", instead of the linguistically expected "b eh k s ao r". Third, combining manual and FG2P pronunciations brings further improvements, as one might expect from combining two high-quality but imperfect, independent, data sources.

We further notice that the performance on the *artist* data set benefits the most from the FG2P pronunciations. Comparing *FG2P* to *Lexicon + G2P*, the relative WER reduction for the artist data is 11.3% compared to 5.4%, 5.1%, and 6.5% for the *movie*, *song*, and *TV show* data sets, respectively. The artist names indeed are the most difficult to recognize among the four categories because they include foreign names (e.g. Avicii, Wolfgang Amadeus) and unusual names (e.g. Flo Rida). This suggests that the recognition performance on proper names can be substantially improved by our approach.

We also compared manual and FG2P pronunciations in a SxS test. The result, illustrated in Fig. 1, shows a clear shift from "usable" to "exact" queries, when contrasting human pronunciation (Model A) to FG2P pronunciations (Model B). This indicates that while the human pronunciations were likely of good quality, the learned pronuciations resulted in recognition results that were prefered by human raters. It should be noted that the pool of human raters had no likely overlap with the pool of Turkers used to collect the audio data, so the assessment from the SxS raters provides a completely independent judgement on the impact of the algorithm, and this on actual Voice Search queries, not on crowd-sourced data.

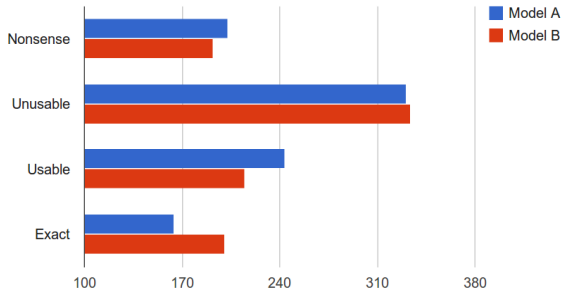

Figure 1: SxS test comparing recognition results using manual pronunciations (Model A) and FG2P pronunciations (Model B).

5.3. Pronunciation Candidates

To get a sense of how many pronunciation candidates need to be considered for FG2P selection, we measured the "rank" (G2P candidate index) of each learned pronunciation, and plotted the rank distribution over all learned pronunciations in Fig. 2. As expected, pronunciations of low rank are more often chosen, but some high-rank pronunciations are also selected.

6. Experiments on Rare Local Entities

In this experiment instead, we focused on Local entities such as business names and street names. We extract 14,000 anonymized Google Maps typed search queries that did not occur frequently in speech recognition logs. Unlike the most downloaded entertainment data sets from the previous experiments, the entities in this set often contain rare words, such as "Mekeni Filipino restaurant" and "Ehukai Beach, Oahu".

Next, we selected 5,000 words from the above data set for which there was no manual pronunciation in the lexicon files.

[1] https://code.google.com/p/sxse/

	Artist names	Movie titles	Song titles	TV Show titles
(1) Lexicon + G2P	28.2 / 68.2	13.0 / 78.0	11.8 / 81.4	15.4 / 76.5
(2) Lexicon + Manual	25.8 / 70.8	12.5 / 78.8	11.8 / 81.4	14.6 / 77.3
(3) FG2P	25.0 / 73.2	12.3 / 79.0	11.2 / 82.7	14.4 / 77.7
(4) Manual + FG2P	24.3 / 73.1	12.1 / 79.0	11.1 / 82.8	13.9 / 78.1

Table 1: WER/SACC results of four systems: (1) Lexicon + G2P, (2) Lexicon + Manual, (3) FG2P, (4) Manual + FG2P.

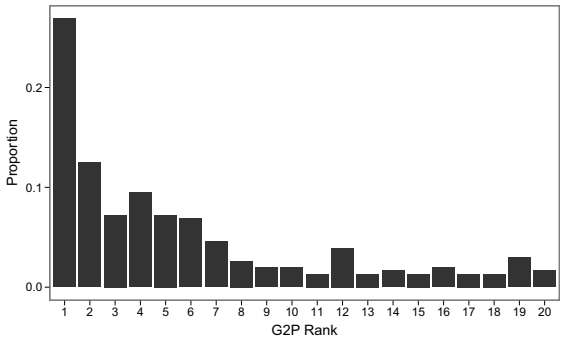

Figure 2: Distribution of ranks of learned pronunciations.

We collected ten audio samples for each word and learned pronunciations from these samples. These were compared to the baseline *G2P* pronunciations in a SxS test. Thus, no manual pronunciations were created for these words, and the SxS test illustrates the benefit we might expect in production from the proposed technique.

Figure 3 compares learned pronunciations (Model B) to G2P baseline pronunciations (Model A).

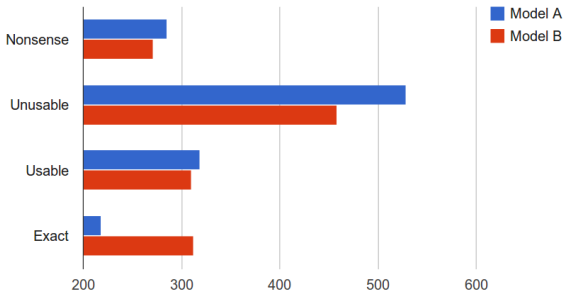

Figure 3: SxS for Local queries, comparing G2P pronunciations (Model A) and FG2P pronunciations (Model B).

We observe a large reduction in Unusable queries with Model B (learned pronunciations), and a large increase in Exact queries (fluctuations in the other two categories are within the noise level).

7. Discussion

The first experiment, using entertainment-related data sets, showed that pronunciations for difficult yet fairly common words can be learned from crowd-sourced audio samples. Test sets consisting of equally crowd-sourced samples showed a substantial WER decrease.

There is however some element of artificiallity to this experiment. First, one might object that "of course" everybody knows how to pronounce the names of celebrities, so the MTurk training data is likely of good quality. In practice, this is not completely true, not every Turker is well-versed in American pop culture, but since we select only the best of up to 7 learned pronunciations, this wisdom of the crowd rounds out imperfect knowledge, and the data indeed turn out to be of good quality. Second, the test set is well matched to the training set: if some less-known name is commonly mispronounced in the training set, it will likely be similarly mispronounced in the test set, and the learned pronunciation will contribute to decreasing the test WER, even though a knowledgeable user who pronounces the word "correctly" and differently from the possibly ignorant Turker will still be left without a proper pronunciation to match his query.

While these objections are valid, the SxS experiments conducted on production logs, i.e. data collected from "real" Voice Search queries still exhibit a clear preference for the learned pronunciations, indicating that the Turkers, on average at least, agree with Google Voice Search users for these names.

This latest point is resonated more strongly in the Local Entity experiment. Here, we focused on unusual queries for which we know from usage patterns that the speech recognition engine is struggling. It may therefore be expected that the names in the data set are difficult, and that perhaps some local knowledge is needed to know how to speak them. One such example in the US is "Houston, Texas" versus "Houston Street, NY" where the two "Houston"'s are pronounced differently. Nevertheless, the SxS experiments again showed a strong preference for the pronunciations learned from crowd-sourcing.

Another limitation of the proposed technique is that it relies on G2P pronunciations. We showed that with 20 selected G2P candidates, the algorithm still found matching candidates towards the end of the list. Perhaps in some cases the best candidate might not be on the list. A quantitative evaluation of the learned pronunciations did not reveal this as a main limitation. Instead, the opposite problem is more frequent: a fairly low-rank G2P pronunciation is selected because it matches the audio well, even though it presents characteristics that a trained linguist would disagree with, such as missing final consonants, extra final consonants, slight vowel shifts, etc. Nonetheless, on average, the proposed technique proves to be an efficient way to improve the accuracy of a large, state-of-the-art, recognition system on difficult Voice Search queries.

8. Conclusion

We proposed an approach to learn pronunciations for named-entities from crowd-sourced data, and demonstrated that this approach can help improve speech recognition accuracy for such entities. It allows us to cheaply and frequently refresh the pronunciation dictionary of our production recognizer and to incorporate new words as needed to support new data. The pronunciations learned in this fashion appear to be more accurate than manually produced pronunciations. The approach is also highly scalable to any language for which the crowd-sourcing service is available.

9. References

[1] F. Peng, S. Roy, B. Shahshahani, and F. Beaufays, "Search results based n-best hypothesis rescoring with maximum entropy classification," in *Proceedings of ASRU*, 2013.

[2] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope, "Google Search by Voice: A case study," *Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics*, 2010.

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

[4] C. Callison-Burch and M. Dredze, "Creating speech and language data with amazon's mechanical turk," in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, ser. CSLDAMT '10, 2010.

[5] R. Snow, B. O'Connor, D. Jurafsky, and A. Ng, "Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks," in *Proceedings of EMNLP*, 2008, pp. 254–263.

[6] M. Weintraub, E. Fosler, C. Galles, Y. hung Kao, S. Khudanpur, M. Saraclar, and S. Wegmann, "Automatic learning of word pronunciation from data," in *JHU/CLSP Workshop Project Report*, 1996.

[7] F. Beaufays, A. Sankar, S. Williams, and M. Weintraub, "Learning linguistically valid pronunciations from acoustic data," in *Proceedings of EuroSpeech*, 2003.

[8] I. McGraw, I. Badr, and J. Glass, "Learning lexicons form speech using a pronunciation mixture model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, 2013.

[9] L. Lu, A. Ghoshal, and S. Renals, "Acoustic data-driven pronunciation lexicon for large vocabulary speech recognition," in *In proceedings of ASRU*, 2013.

[10] X. Li, A. Gunawardana, and A. Acero, "Adapting grapheme-to-phoneme conversion for name recognition," in *Proceedings of ASRU*, 2007.

[11] X. Lei, W. Wang, and A. Stolcke, "Data-driven lexicon expansion for mandarin broadcast news and conversation speech recognition," in *Proc. ICASSP*, 2009.

[12] A. Laurent, L. Mans, T. Merlin, S. Meignier, Y. Esteve, and P. Deleglise, "Iterative filtering of phonetic transcriptions of proper nouns," in *Proceedings of ICASSP*, 2009.

[13] A. Laurent, S. Meignier, T. Merlin, and P. Delglise, "Acoustics-based phonetic transcription method for proper nouns," in *Proceedings of InterSpeech*, 2010.

[14] S. Novotney and C. Callison-Burch, "Shared task: crowdsourced accessibility elicitation of wikipedia articles," in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, 2010, pp. 41–44.

[15] I. McGraw, C.-y. Lee, I. L. Hetherington, S. Seneff, and J. Glass, "Collecting voices from the cloud," in *LREC*, 2010.

[16] H. Gelas, Besacier, and F. L. Pellegrino, "Developments of swahili resources for an automatic speech recognition system," in *SLTU-Workshop on Spoken Language Technologies for Under-Resourced Languages*, 2012.

[17] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Proceedings of ICASSP*, 2013.

[18] M. Mohri, F. C. N. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 69–88, 2002.