Interspeech 2018
2-6 September 2018, Hyderabad

Sequence-to-Sequence Neural Network Model with 2D Attention for Learning Japanese Pitch Accents

Antoine Bruguier[1], *Heiga Zen*[2], *Arkady Arkhangorodsky*[3]

[1] Google, Mountain View, USA [2] Google, London, UK
[3] University of Toronto, Tronto, Canada

{tonybruguier,heigazen}@google.com, arkady.arkhangorodsky@mail.utoronto.ca

Abstract

Many Japanese text-to-speech (TTS) systems use word-level pitch accents as one of their prosodic features. Combination of a pronunciation dictionary including lexical pitch accents and a statistical model representing the word accent sandhi is often used to predict pitch accents from a text. However, using human transcribers to build the dictionary and training data for the model is tedious and expensive. This paper proposes a neural pitch accent recognition model. This model combines the information from audio, and its transcription (word sequence in hiragana characters) via two-dimensional attention and outputs word-level pitch accents. Experimental results show a reduction in the word pitch accent prediction error rate over that with text only. It lowers the load of human annotators when building a pronunciation dictionary. As the approach is general, it can be used to do pronunciation learning in other languages as well.

Index Terms: Japanese text-to-speech; pitch accent; pronunciation learning; attention; neural network.

1. Introduction

Typical TTS systems consist of a number of sub components, such as sentence segmentation, word segmentation, text normalization, grapheme to phoneme conversion, prosody prediction, and speech waveform generation. Predicting prosodic features from a raw text is one of the most important and difficult parts, as no information about prosody is usually available in the raw text.

In the case of Japanese, the control of fundamental frequency (F_0) movement is critical to achieve natural-sounding synthetic speech. Standard Japanese (i.e., Tokyo-dialect) is known as a pitch-accented language. The accent is characterized as a downstep from high-pitch to low-pitch; the F_0 of a word rises until it reaches a mora with downstep, then drops abruptly. The position of the mora with downstep is called pitch accent type (nucleus). Here are examples of lexical pitch accent types of Japanese place names.

- 大阪 (おおさか; Ōsaka) → type = 0 (i.e., no downstep)
- 奈良 (な↓ら; Nara) → type = 1
- 徳島 (とく↓しま; Tokushima) → type = 2

The correct rendering of pitch accent is essential to having a pleasant and accurate TTS system in Japanese. Indeed, some words are pitch accent homophones: they are pronounced with the same phonemes, but the pitch accent is different and the words have distinctive meaning. For example, the words 箸 (chopsticks, type = 1) and 橋 (bridge, type = 0) have the same hiragana writing はし but have different pitch accents. Japanese is mostly written using Kanji characters, Hiragana characters, and Katakana characters. Both Hiragana and Katakana characters are an accurate representation of the phonemes of a word and word written in Kanji can be transformed into either Hiragana or Katakana characters. However, none of these three written forms have a marker for the pitch accent. In fact, many native Japanese speakers are unaware that their language has pitch accent, but if a TTS system renders a word with the correct sequence of phonemes but an incorrect pitch accent, it will immediately sound incorrect to a native listener.

For a given text, the location of the pitch accent nucleus (downstep) needs to be predicted for each accentual phrase as well as the boundaries of breath groups and accentual phrases. An accentual phrase is often composed of a few words, typically a content word followed by a function word. Although all the content words (and some function words) have their own accent type as their lexical attribute, the accent nucleus of an accentual phrase often shifts due to the word accent sandhi. In the following example, pitch accent types change when two words are concatenated to form a compound word.

- 音響 (おんきょう; Onkyō) → type = 0
- 学会 (がっかい; Gakkai) → type = 0
- 音響学会 (おんきょうが↓っかい; Onkyōgakkai) → type = 5

Since the position of the pitch accent in a word varies depending on context [1], having a dictionary may not be sufficient. This accent shift has to be correctly predicted in TTS conversion. Many Japanese TTS systems adopted rules developed by Sagisaka et al. [2, 3]. There have also been statistical model-based approaches to learn the word accent sandhi, such as N-gram language models [4] and conditional random fields (CRFs) [5].

Both rule-based and statistical model-based approaches rely on the pronunciation dictionary which includes lexical pitch accents. The statistical model-based approach further requires training data, which is a set of pitch accent annotated sentences. However, building the dictionary and the training data is tedious and expensive; building the dictionary typically requires professional linguists, and annotating pitch accents of Japanese sentences requires human transcribers who speak the Tokyo dialect of Japanese and are familiar with the concept of Japanese pitch accents. For these reasons, the publicly available linguistic resources of Japanese pitch accents is limited (e.g., Unidic [6]).

This paper proposes a neural pitch accent recognition model. This model combines the information from audio and its transcription (word sequence in Hiragana characters) via two-dimensional attention and outputs word-level pitch accents. It lowers the load of human annotators when building a pronunciation dictionary and pitch accent sandhi model.

The rest of this paper is organized as follows. Section 2 describes the task. Sections 3 and 4 explain the proposed ap-

10.21437/Interspeech.2018-1381

proach. Section 5 discusses the relationship to pronunciation learning. Section 6 shows experimental results. The final section gives concluding remarks.

2. Description of the Task

The goals of the task is to 1) build a pronunciation dictionary where the representation of the pronunciation makes explicit both the phonemes and the pitch accent and 2) annotate pitch accents given audio and its phonetic transcription (without pitch accents) to be used for training pitch accent sandhi models.

One way to build such a dictionary and annotate data is to train human native speakers of Japanese and ask them to transcribe a large quantity of words and annotate audio. This is usually a tedious and expensive task. It is quite challenging to scale to a large dictionary. It would also require the native speakers to know the correct pitch accent in standard Japanese (Tokyo dialect) for words in the tail-end of the distribution.

One way to improve the accuracy of the human transcribing the pitch accent is to provide them with an example of audio. If they can listen to the example of the audio, then they will know what the correct pitch accent is. Thus, providing the example improves the accuracy of the human transcription. However, it does not reduce the tediousness of the task.

Therefore, this paper propose to build a machine learning system that performs the same task as the expert human transcribers. The system takes as input the Hiragana representation (phonetic transcription) of what is being said, and the audio waveform of the voice. It then produces the same sequence of Hiragana interspersed with pitch accent marks. For example, an input of the system would be:

- るな?に?よると?すぐ?に?おこるから?しんばい?
- audio waveform

The question mark is the indicator that a pitch accent must be predicted. The target sequence is:

- るな1に0よると2すぐ1に0おこるから2しんばい0

where the numbers indicate the pitch accent types of their preceding word in Hiragana.

Thus, such a system would perform the same task as human annotators. This paper will show that, while the accuracy of the system is not 100%, its performance is high enough that it greatly reduces the tediousness of the task for humans, and thus increases the throughput.

As a comparison, we measure the performance of a model that does not use the audio and tries to predict the target sequence only from the input text. If we can show that the model that uses the audio waveforms outperforms the model that does not use them, then we will have built a model that truly learns pronunciations from audio.

3. Input Sequences

The proposed model has two input sequences, one for transcription and another is for audio. The first input is the sequence of T input graphemes (Hiragana), $\boldsymbol{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$. Each grapheme is represented as a one-hot vector $\boldsymbol{x}_t = [\tilde{x}_{1,t}, \ldots, \tilde{x}_{N,t}]$ where N denotes the total number of possible graphemes.

The second input is acoustic features derived from the input audio waveform. The acoustic features are a sequence of M-dimensional mel-filterbank outputs, $\boldsymbol{y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_U\}$ where U denotes the length of acoustic features.

Each of the input sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ is the input of an encoder long short-term memory (LSTM) [7] model. The encoder LSTM models output the two sequences of encoded inputs, $\{\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_T\}$ and $\{\bar{\boldsymbol{y}}_1, \ldots, \bar{\boldsymbol{y}}_U\}$, where $\bar{\boldsymbol{x}}_t$ and $\bar{\boldsymbol{y}}_t$ correspond to the I and J-dimensional encoded grapheme and acoustic features, respectively. We used the standard LSTM model. It is hypothesized that the LSTM model that computes $\bar{\boldsymbol{x}}$ can extract representation which can predict pitch accents of given graphemes. It is also hypothesized that the LSTM model that computes $\bar{\boldsymbol{y}}_t$ is able to extract representation which can predict both phonemes and F_0. While the acoustic features do not explicitly include F_0, this LSTM model has access to the mel-filterbank outputs, which has enough information to discover F_0 [8] (or whatever signal is a better predictor of F_0). Then these two streams must be combined to predict the pitch accents.

If only the sequence $\bar{\boldsymbol{x}}$ derived from the graphemes is available, an attention-based sequence-to-sequence model [9, 10] would be a natural choice. Indeed these models have been used in grapheme-to-phoneme prediction [11]. It would be able to focus its attention on the context of the word whenever it has to predict pitch accent markers. However, there are two sequences of varying length T and U, and a new way to combine them is required.

4. Two-Dimensional Attention

One way to combine the two sequences would be to force-align the audio to the text. This way, we assign a grapheme to each of the audio frames (or a blank if no voice is occurring). This preprocessing step would then allow us to feed a single sequence into our model that predicts the pitch accent.

This approach has two main disadvantages. The first disadvantage is that we would need to first train a force-alignment model. While this can be derived from a traditional automatic speech recognition (ASR) system, it requires building and training one. Another disadvantage is that the force-alignment model and the pitch accent prediction model would be separately optimized. This would mean, in particular, that the force-align model may not be optimal for the final task of pitch accent prediction.

Instead, we propose to extend the traditional attention-based sequence-to-sequence model to use a two-dimensional attention. With this two-dimensional attention, the model will be free to do the force-align task as best as it sees fits and will have at all times access to whichever part of the text and audio that it deems useful to the task. This will be achieved by building an attention matrix that can be indexed by a weight matrix $w_{u,t}$ with the two time-dimension indices.

While 2D attention has been used for images (e.g. [12]), the problem here is slightly different. The speech can occur at a different rate from one example to another (depending on how fast a person speaks) and the onset and offset of it may not be at the beginning and end of the audio. In essence, it is like having rectangular pixels for an image. We could force-align the sequences, but then we lose the end-to-end joint advantage of training a single model.

As described above, the two inputs used to build an attention matrix are:

- $\bar{x}_{i,t}$ derived form the T graphemes (t from 1 to T) and encoded on I dimensions (i from 1 to I).
- $\bar{y}_{j,u}$ derived form the U audio frames (u from 1 to U) and encoded on J dimensions (j from 1 to J).

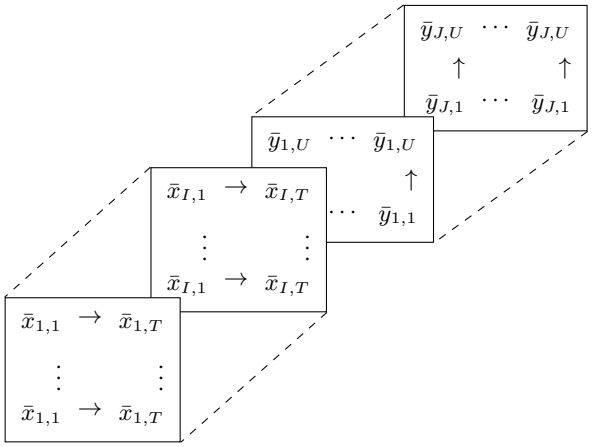

Figure 1: *Encoding of the 2D attention*

We then build the input attention matrix $A_{k,u,t}$ of dimensions $I + J$, T, and U (figure 1). Its coefficients are:

- $A_{k,u,t} = \bar{x}_{k,u}$ if $k \leq I$
- $A_{k,u,t} = \bar{y}_{k-I,t}$ if $k > I$

Note that the input attention matrix contains repeated information. Along one dimension, the steps increase with audio time, and along another, the steps increase with grapheme time. It is up to the attention network to decide how to align the two steps. In the equations above, for example if $k \leq I$, the variable t is not used on the right-hand side of the equation. This allows for the model to decide separately what part of the grapheme and what part of the audio to attend to.

The input of the decoder is the dot product between the input attention matrix $A_{k,u,t}$ and weights $w_{u,t}$:

$$h_k = \sum_{u,t} w_{u,t}.A_{k,u,t}$$

We then operate a recurrent decoding, starting with a special `<sos>` symbol, predicting phonemes until a special `<eos>` symbol is predicted [13].

5. Relation to pronunciation learning

The proposed approach is closely related to the task of pronunciation learning [14]. The task of pronunciation learning is to combine audio and graphemes to attach a sequence of phonemes to words.

Even though the model that we use has the same symbols for the input graphemes and the target, there is no a-priori restriction to using a different set of symbols for the target. We could, for example, use English audio as input, along with input graphemes transcribed using the Latin alphabet, and have a sequence of X-Sampa phonemes as targets.

Previous attempts at pronunciation learning used decisions trees [15], expectation maximization [16], hidden Markov models [17], discriminative training [18], or finite-state transducers [19]. Here, we present an end-to-end approach that trains in a single run. The model does not need prior force-alignment.

This approach has the additional advantage that the weight given to the graphemes and the audio does not need to be fixed. In [19], it was shown that tuning the relative importance of the graphemes versus the audio yields large accuracy gains. However, the optimal weight is likely to be speaker, task, and context dependent. An end-to-end model has a better chance of capturing these variations than a single hyperparameter.

Finally, another advantage is that the model is jointly trained. Previous attempts typically reuse an acoustic model and/or a grapheme-to-phoneme models. These models are trained with a different quality metric than pronunciation learning accuracy. Here we train for the intended usage.

Our approach, however, requires data that is typically not available. To be used effectively, we would need to have a large data set where the following information is available:

- audio waveform
- human-transcribed text
- human-transcribed phoneme sequence

The previous approaches have different models, and thus do not require these data sets. For example, if a previous approach requires an acoustic model, then it can be trained using a data set that does not have human-transcribed phoneme sequences; only audio and graphemes.

However, the problem of pitch accent prediction from audio is well-suited for designing an end-to-end pronunciation learning model. First, the data is available to us. Second, it solves a real world problem for which we previously didn't have any direct solution. Third, the task is simpler because the input graphemes and the target are very close to each other, and thus we conceivably have a chance of building a predictive model with a relatively small amount of data.

6. Experiment

In order to perform our experiments, we collected audio recordings of about 19,000 sentences. These segments are of studio quality, and are used to build our TTS voice systems. For this reason, they also had been annotated by expert linguists so that in addition to text transcription, the data also had pitch accent markers. The data came from two speakers.

While this data set does not contain a full variety of how native voices render pitch accent, it is nevertheless very useful. Our practical goal for building such pitch accent models is to accelerate and simplify future annotation of voice building data, and thus lower its cost.

We split the data in two subsets with 90% for training and 10% for testing.

Our goal was to prove that the addition of the audio data provided additional information that the model could use. Thus, we trained two models on the same data. In the base model, we only used the graphemes and tried to predict the graphemes with the pitch markers. Since there is only a single input stream, we used a 1D attention. The test model had access to the audio, and thus used a 2D attention on both the text and the audio, as described in section 4. If the test model does indeed use the audio information, then we should see a reduction in the number of errors made.

We used standard encoding neural networks. For both models, the grapheme encoder had three bidirectional LSTM layers, with 256 units each. The test model had an additional audio encoder, consisting of six bidirectional LSTM layers, each with 64 units. In both models, we used dropout [20] with a keep probability $p = 0.10$ to reduce over-fitting.

We also used a standard decoding neural network. For both models, the decoder used a dot-product attention and was followed by a 3-layer DNN with 256 units per layer. In short,

the only difference between the train and the test models was whether an audio encoder was present and whether the attention needed to be 2D.

Using graphemes only	19%
With graphemes and audio	16%

Table 1: *Error rate of the predictions*

We see in Table 1 a reduction of the prediction error when using the audio in addition to the text. Thus, in our case, we were able to reduce the number of pitch accent prediction errors by about 15% relative to adding a new audio encoder and using a 2D attention to combine the information from the grapheme and audio.

7. Conclusion

We presented an extension of attention-based neural networks that allow for using *two* input sequences whose lengths vary from example to example. We then applied the new model to the problem of learning the pitch accent of Japanese words by using as input the audio waveform and the transcript written using hiragana characters. On a data set of about 1,900 examples, we reduced the prediction error by 15% (relative).

The method we proposed is, however, general. It can be readily expanded to the more general problem of pronunciation learning where we learn the complete sequence of phonemes. The new model naturally learns to force-align input sequences, and how to weight information from graphemes and audio. In addition, it is trained with the objective function that is precisely the phoneme accuracy, as opposed to previous attempts that relied on models trained independently and with objective functions not designed for pronunciation learning.

8. References

[1] N. Minematsu, R. Kuroiwa, K. Hirose, and M. Watanabe, "CRF-based statistical learning of japanese accent sandhi for developing japanese text-to-speech synthesis systems," *ICSA*, no. 6, 2007.

[2] Y. Sagisaka and H. Sato, "Accentuation rules for japanese text-to-speech conversion," *Review of the Electrical Communication Laboratories*, 1984.

[3] ——, "Accentuation rules for japanese word concatenation," *Transactions of IECE*, 1983.

[4] T. Nagano, S. Mori, and M. Nishimura, "An N-gram-based approach to phoneme and accent estimation for TTS," *Trans. IPSJ*, vol. 47, no. 6, 2006.

[5] M. Suzuki, R. Kuroiwa, K. Innami, S. Kobayashi, S. Shimizu, N. Minematsu, and K. Hirose, "Accent sandhi estimation of tokyo dialect of Japanese using conditional random fields," *IEICE Trans.*, vol. E100-D, no. 4, pp. 655–661, 2017.

[6] "Unidic," http://unidic.ninjal.ac.jp/.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] X. Shao and B. Milner, "Pitch prediction from MFCC vectors for speech reconstruction," in *Proc. ICASSP*, 2004, pp. 97–100.

[9] S. Toshniwal and K. Livescu, "Read, attend and pronounce: An attention-based approach for grapheme-to-phoneme conversion," *Interspeech*, 2016.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *NIPS*, 2018.

[11] B. Milde, C. Schmidt, and J. Köhler, "Multitask sequence-to-sequence models for grapheme-to-phoneme conversion," *Interspeech*, 2017.

[12] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[13] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *CoRR*, 2015.

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

[15] B. Byrne, M. Finke, S. Khudanpur, J. Mcdonough, H. Nock, M. Riley, M. Saraclar, C. Wooters, and G. Zavaliagkos, "Pronunciation modelling for conversational speech recognition: A status report from ws97," in *Fifth LVCSR Summer Workshop, Johns Hopkins University*, 1997.

[16] I. Badr, "Pronunciation learning for automatic speech recognition," Ph.D. dissertation, Massachusetts Institute of Technology, 2011, http://hdl.handle.net/1721.1/66022.

[17] R. Rasipuram, M. Razavi, and M. Magimai-Doss, "Integrated pronunciation learning for automatic speech recognition using probabilistic lexicon modeling," in *ICASSP*, 2015.

[18] O. Vinyals, L. Deng, D. Yu, and A. Acero, "Discriminative pronunciation learning using phonetic decoder and minimum-classification-error criterion," in *ICASSP*, 2009.

[19] A. Bruguier, D. Gnanapragasam, L. Johnson, K. Rao, and F. Beaufays, "Pronunciation learning with rnn-transducers," in *Interspeech*, 2017.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," in *Journal of Machine Learning Research*, 2014.