AN ATTENTION BASED DEEP NEURAL NETWORK FOR AUTOMATIC LEXICAL STRESS DETECTION

Tian Xia, Xianfeng Rui, Chien-Lin Huang, Iek Heng Chu, Shaojun Wang, Mei Han

{SummerRainET2008, huntrui, chiccocl, iekhengchu0217, swang.usa, hanmei613}@gmail.com

ABSTRACT

Lexical stress detection is one of important tasks in self-directed language learning application. We address this task by leveraging two successful attention techniques in natural language processing, **inner attention** and **self-attention**. First, combined with LSTM to model time-series features, inner attention could extract most important information and then convert length-varying input into a fixed-length feature vector; Second, self-attention intrinsically support as input words with *different number of syllables* to model contexture information. Besides, our model is straightforward to expand to include hand-crafted features to improve performance, and can be applied to similar tasks, such as pitch accent detector. Experiments on LibriSpeech, TedLium and a third self-recored datasets show the high performance of our proposed attention based neural network.

***Index Terms*—** stress, detection, attention, neural network

1. INTRODUCTION

Nowadays self-directed language learning has become more and more popular, and computer-aided pronunciation training therefore has drawn considerable attention. The research in this direction is focusing on mispronunciation detection and diagnosis problems by using speech recognition related technologies. These problems can be addressed in segmental level and suprasegmental level. For example, segmental level work involving phones and words [1, 2], suprasegmental level involving lexical stress [3, 4, 5] and pitch accent [3, 6], etc. This paper targets at lexical stress detection problem, which is one of most important factors for evaluating proficiency.

Lexical stress has a relation with the prominent syllables of a word. As pointed in the work of [7], in many cases the position of syllable carries important information to disambiguate word semantics. e.g., "subject" vs. "sub'ject", "permit" vs. "per'mit". Once we detect lexical stress for a word, and compare with its typical lexical stress pattern from dictionary, we could determine whether stress pronunciation is correct.

There are a line of related work. Such as the work in [8] uses Gaussian mixture models; Zhao et al. [9] adopted SVMs to identify the vowels of L2 English speech carrying primary stress or not. Other similar studies using SVMs to detect lexical stress of English speech with Taiwanese accent were reported in Wang et al. [10], Chen and Wang [11], etc. Ferrer et al. [12] introduced a system for lexical stress detection using both prosodic (pitch, energy and duration) and spectral MFCC features. Results showed that the MFCC helped improve the experimental performance. The error rates of the system for L1-English and L1-Japanese data were 11% and 20% respectively.

Another important work in [7] uses multi-distribution deep neural networks (MD-DNN), which are constructed by stacking up multiple Restricted Boltzmann Machines (RBMs). The work from Li et al. work uses several hand-crafted syllable-based features with length fixed. The main difference with our model is we do not hand-craft features, but have the neural network learn itself. Moreover, our model does not require the input to be of 5-syllable[1], and prevent any approximation of input.

2. ATTENTION BASED NEURAL NETWORK

2.1. Time-series features

Intuitively, pitch highly correlates with stress. Instead of extracting mere one highest frequency, we extract several highest frequencies in each frame, which empirically contribute to the performance moderately..

As stressed syllable usually exhibits higher energy than its neighboring ones, thus energy is extracted in the frame level.

Besides, Mel Frequency Cepstral Coefficient (MFCC) features are found to benefit stress detection ask in [12]. Hence we adopt MFCC here, with Delta and Delta-Delta information included as well. Empirically, we found large dimension of MFCC improves the final performance. Thus we prefer large dimensions.

All the three kinds of features above are normalized in the word level and in each feature dimension. Our normalization strategy has two parts. First, lineally scale features into the range of minimum and maximum, and second, subtract by the mean value. Adopting either part separately hurts the performance.

[1]They would fill short words to 5-syllable with special values, usually 0s.

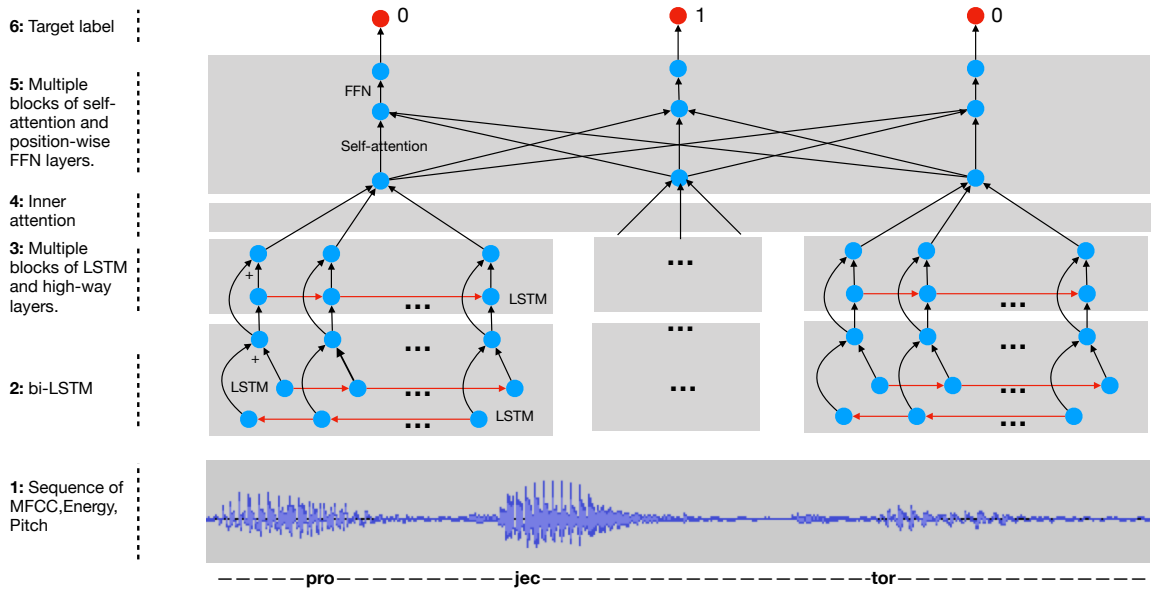

Fig. 1. Our neural network structure. For example, first, the word "**projector**" is partitioned into three syllables by linguistic rules, "pro", "jec" and "tor". Second, each syllable, represented as concatenation of several time-series features in the frame level, e.g., MFCC, pitch and energy, is encoded by the LSTM blocks, and then be converted into a fixed-length feature vector by inner attention Third, all syllable-representing feature vectors are interacting with each other via **self-attention**, and finally be trained to fit their final labels. Note that, all LSTM models are sharing the same parameters, and so are all position-wise Feed-Forward-Networks (FFN).

2.2. Syllable encoding module

The logic level 2, 3, 4 in Fig. 1 demonstrate the internal structure of syllable encoding module. It consists of one bidirectional LSTM, several blocks of uni-directional LSTM and residual edge, and one inner attention layer [13].

Based on the statistics of syllable duration, we limit the maximum LSTM steps as 50, corresponding to 500 ms duration. This setting is the same through logic level 2 and 3. In logic level 2, two frame-level LSTMs runs from different directions and element-wisely sum together to enrich both the left and right context for each frame state. This is also the common doing when applying LSTM into a sequential data.

The neural network structure in logic level 3 is alike in that of Google translation model [14]. It is made of multiple identical blocks, each of which has a uni-directional LSTM and element-wisely add its input into its output via a residual edge.

In the logic level 4, inner-attention part, which can be interpreted as a special weighted-pooling strategy, the processing is a bit tricky. Because the durations of syllables vary a lot, it is beneficial to only weight those real frame information, and ignore those filled frame information due to the maximum LSTMS steps or maximum frame number[2]. See Equ. 1.

$$\alpha_i = \begin{cases} \text{softmax}\{f(\mathbf{S_i}, \mathbf{H})\} & i \in [0, \text{syllable\_length}) \\ 0 & i \in [\text{syllable\_length}, 50) \end{cases} \quad (1)$$

$$\mathbf{S} = \sum \alpha_i \cdot \mathbf{S_i} \quad (2)$$

where $\mathbf{S_i}$ is the state vector of LSTM, corresponding to each speech frame. $\mathbf{H}$ is a global and trainable vector, shared by each syllable.The function f defines how to compute the importance of each state vector by its real content. The simplest definition is the inner product.

2.3. Syllable interaction module

Pronunciation stress kinds of reflects the focused information of a whole word, thus, it is natural to consider contextual information. The work in [7] also take advantage of context of a syllable, but they considers fixed four-syllable contextual information. In comparison, we take a self-attention [15, 16] based network, which digests words with different number of syllables as input. There is no need to expand input by filling empty positions.

The logic level 5 includes two parts, $\mathcal{O}(n^2)$ operations of self-attention, and $\mathcal{O}(n)$ operations of position-wise feed forward network.

In the self-attention part, we adopt more efficient bi-linear formula than inner product for the attention weight $\alpha_{i,j}$, and

[2]Based on statistics of syllable durations, we just set maximum frame number as 50.

the matrix $\mathbf{M}$ is a globally trainable parameter. There are other alternatives to calculate the weight $\alpha_{i,j}$, such as more powerful multi-head attention in BERT model [15, 16]. It is not difficult to replace with it, however, we adopt the bi-linear formula due to two reasons, simple to implement and focused on the whole network structure itself.

$$\alpha_{i,j} = \text{softmax}\{\mathbf{S_i}^T \mathbf{M} \mathbf{S_j}\} \tag{3}$$

$$\mathbf{S_i} = \sum_j \alpha_{i,j} \cdot \mathbf{S_j} \tag{4}$$

In the position-wise feed forward network part, we use the definition in [16]. It is actually equivalent to two dense networks, with the first one having the relu activation function and the second one not.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b \tag{5}$$

2.4. Target label

Generally, we assign scores of 1, 0.5, 0 [3] as our target labels for primary stress, secondary stress, and no stress respectively.We then use $l1$-norm to convert these label scores into a probability distribution, and use in our cross-entropy based loss function. We should note that one word might have more than one primary stress, thus problem is a not multi-class problem, but a multi-label problem.

$$\mathcal{L} = -\sum_{syl} p_{\text{syl}}^{\text{label}} \log P_{syl}^{\text{o}} \tag{6}$$

where $p_{\text{syl}}^{\text{label}}$ is the normalized target label probability of some syllable, and P_{syl}^{o} is the corresponding output probability from the self-attention blocks.

3. EXPERIMENTS

3.1. Datasets

As there lack public datasets with enough wrong stress pronunciations, we adopt two public datasets, which is usually used for ASR task, to evaluate the performance of detected **true stress** pronunciations; we also recored a third dataset with one half of wrong stress pronunciation, to evaluate the performance of detected **wrong stress** pronunciations.

In practical application, users are more concerned about the correctness of the primary stress, so we do not predict and measure the **secondary stress**, though our model is capable.

Besides, our model is end-to-end based to directly optimize the stress prediction in the **word-level**, thus, we do not report measures of syllable-level stress prediction. All reported results in the experiments are F-values, which balances the precision rate and recall rate.

[3] It is only based on experiences, and might not the optimal.

1. LibriSpeech [4] dataset, and we use 360 hours clean read English speech for training, 50 hours for testing.

2. TedLium dataset, which is talk set with a variety of speakers and topics. We use 400 hours for training, 50 hours for testing.

3. dictionary based dataset. We have 10 teammates record 2000 vocabularies, most of which have 3 syllable and 4 syllables [5], and pronuciate each word for three time. Totally we have about 6000 word based samples, and a half is from female speakers.

3.2. Features and systems

All features used in our systems and baseline systems are based on syllable. Here for our syllable feature extraction, we use the Maximal Onset Principle (Pulgram, 1970) to extract syllables from the phoneme sequence decoded from our automatic speech recognizer.

Four kinds of features are considered in baseline systems, and they are duration, energy, pitches, and MFCC. Since the absolute duration of the same syllable within one word can vary from people to people, we actually measures the relative duration of each syllable within one word. Regarding the other features, we process in the same way. They are extracted in the frame level, and normalized at the word boundary to compute its relative values. We chose 25% percentile,50% percentile, 75% percentile, minimum and maximum value within the syllable window. Regarding MFCC, we observe higher dimension benefits better results. Therefore, we set the dimension of MFCC to 40. Plus additional delta and delta-delta information, we actually use 120 dimension MFCC related features.

1. The first baseline system is a SVM based system which is similar to the one in [9]. We use Gaussian kernel and default parameter settings.

2. The second baseline baseline system is gradient-boosting tree based, sharing exact features with the first baseline. There are some tunable parameters, such as tree depth, leave number. We report the best result.

3. Our attention based network model is capable of direct processing time series features, thus we take energy, pitches and MFCC in all frames, but we do not use the duration feature. The model is implemented in Tensorflow, and the optimizer is Adam with default hyper parameters. The learning rate is 1e-3. We found 10 epoch of train is enough to reach good performance.

[4] http://www.openslr.org/12

[5] 2-syllable words are much easier. See Table 1.

Dataset	Model	*#syllable*=all	2	3	4	5
LibriSpeech	SVM	0.80	0.822	0.783	0.730	0.652
	Boosting	0.85	0.874	0.834	0.784	0.708
	Ours(summer)	**0.95**	0.9619	0.9138	0.8304	0.7232
TedLium	SVM	0.812	0.834	0.779	0.739	0.661
	Boosting	0.862	0.888	0.831	0.813	0.732
	Ours	**0.951**	0.9669	0.9438	0.8804	0.7832
dictionary	SVM	0.69	0.712	0.682	0.682	0.643
	Boosting	0.726	0.734	0.721	0.712	0.654
	Ours	**0.788**	0.821	0.777	0.762	0.723

Table 1. In LibriSpeech and TedLium datasets have its own training and testing data, while as dictionary-data is limited, we just take the best model trained in LibriSpeech data. Our model adopts the best configuration. More details can be found in Table 2 and 3.

3.3. Results

We first look at the performances of all models on the whole datasets in Table 1. Boosting tree based training performs consistently better than SVM based, because boosting tree has more controllable parameters to tune in practice. More, tree-based models have no requirement in the data normalization, and it is hence more robust. Our attention based network model performs significantly better than boosting tree based. This is our model theoretically digests time-series features, and they contains more information than the input feeding into SVM and boosting tree based .

In LibriSpeech and TedLium datasets, our model has much better ability to sufficiently fit the data. However, this advantages moderately decrease in the self-recorded dictionary dataset. We consider there are two reasons. The first one is different data distribution. The dictionary dataset is from Chinese accent speakers, while LibriSpeech data is American English, and TedLium is mixed with a variety of speakers with different accent. The second one is, both LibriSpeech and TedLium datasets are supposed to not include wrong stress pronunciations. This might result in sort of memorizing vocabularies, as well as their stresses, with any feature set. We could not exclude this possibility, as it is hard to create a dataset with enough wrong stresses to verify it. Anyway, our model still reveals better performance than baselines comparatively. To construct a large dataset with wrong stresses to improve our model is also our future work.

As the distribution of syllable number of words varies in datasets, we further have a statistics of model performances on words with different syllable number. In Table 1, generally, our model on short words, which has no more than 3 syllables, performs quite well with at least 0.90 F-values. On longer words, all models perform worse, and our model still holds advance over other two baselines.

In Table 2, we display the performances from different number of LSTM blocks. Besides testing on the LibriSpeech data, we also apply the best model into dictionary dataset for testing. We observed that more LSTM layers strongly contributes to the final performance. However, when the layer number reaches 6, the performance decreases. Besides, more LSTM blocks make the training significantly slower.

#LSTM	1	2	3	4	5	6
LibriSpeech	0.92	0.928	0.939	0.944	**0.951**	0.948
dictionary	0.743	0.751	0.760	0.768	**0.77**	0.764

Table 2. We show the performances from different block numbers of LSTM in the logic layer 3 in Figure 1, where the self-attention block number is fixed to 1. See Table 3.

#self-att	0	1	2
LibriSpeech	0.941	**0.951**	0.929
dictionary	0.743	**0.77**	0.760

Table 3. We show the performances from different number of self-attention blocks in the logic layer 5 in Figure 1, where the LSTM block number is fixed to 5.

In Table 3, we surprisingly find more self-attention layers are not helpful to this task, which kind of differs from that in machine translation experiments [16]. We conjecture our task is comparatively simpler than machine translation task, and more self-attention layers lead to overfitting. We also tried other position-wise activation function in Equ. 5, such as one layer linear network with relu or other activations, and they are not helpful.

4. CONCLUSION

In this work, we successfully applied two widely used attention techniques into lexical stress detection problem. Our proposed model directly takes time series features as input that it could fully explore input information; and the network structure instrinstically support words with different number of syllables, without expanding short words, to reduce input approximation. Experiments also demonstrate the improvements over baselines systems.

5. REFERENCES

[1] Kun Li, Xiaojun Qian, Shiyin Kang, Pengfei Liu, and Helen Meng, "Integrating acoustic and state-transition models for free phone recognition in l2 english speech using multi-distribution deep neural networks.," in *SLaTE*. Citeseer, 2015, pp. 119–124.

[2] Kun Li, Xiaojun Qian, and Helen Meng, "Mispronunciation detection and diagnosis in l2 english speech using multidistribution deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 193–207, 2016.

[3] Kun Li, Shuang Zhang, Mingxing Li, Wai-Kit Lo, and Helen Meng, "Prominence model for prosodic features in automatic lexical stress and pitch accent detection," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[4] Kun Li and Helen Meng, "Perceptually-motivated assessment of automatically detected lexical stress in l2 learners' speech," in *2012 8th International Symposium on Chinese Spoken Language Processing*. IEEE, 2012, pp. 179–183.

[5] Kun Li, Xiaojun Qian, Shiyin Kang, and Helen Meng, "Lexical stress detection for l2 english speech using deep belief networks.," in *Interspeech*, 2013, pp. 1811–1815.

[6] Junhong Zhao, Wei-Qiang Zhang, Hua Yuan, Michael T Johnson, Jia Liu, and Shanhong Xia, "Exploiting contextual information for prosodic event detection using auto-context," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2013, no. 1, pp. 30, 2013.

[7] Kun Li, Shaoguang Mao, Xu Li, Zhiyong Wu, and Helen Meng, "Automatic lexical stress and pitch accent detection for l2 english speech using multi-distribution deep neural networks," *Speech Communication*, vol. 96, pp. 28–36, 2018.

[8] Joseph Tepperman and Shrikanth Narayanan, "Automatic syllable stress detection using prosodic features for pronunciation evaluation of language learners," in *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.* IEEE, 2005, vol. 1, pp. I–937.

[9] Junhong Zhao, Hua Yuan, Jia Liu, and S Xia, "Automatic lexical stress detection using acoustic features for computer assisted language learning," *Proc. APSIPA ASC*, pp. 247–251, 2011.

[10] Jhing-Fa Wang, Gung-Ming Chang, Jia-Ching Wang, and Shun-Chieh Lin, "Stress detection based on multiclass probabilistic support vector machines for accented english speech," in *2009 WRI World Congress on Computer Science and Information Engineering*. IEEE, 2009, vol. 7, pp. 346–350.

[11] Jin-Yu Chen and Lan Wang, "Automatic lexical stress detection for chinese learners' of english," in *2010 7th International Symposium on Chinese Spoken Language Processing*. IEEE, 2010, pp. 407–411.

[12] Luciana Ferrer, Harry Bratt, Colleen Richey, Horacio Franco, Victor Abrash, and Kristin Precoda, "Classification of lexical stress using spectral and prosodic features for computer-assisted language learning systems," *Speech Communication*, vol. 69, pp. 31–45, 2015.

[13] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang, "Learning natural language inference using bidirectional lstm model and inner-attention," *arXiv preprint arXiv:1605.09090*, 2016.

[14] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.