INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Predicting Pronunciations with Syllabification and Stress with Recurrent Neural Networks

Daan van Esch, Mason Chua, Kanishka Rao

Google Inc., USA
{dvanesch,aftran,kanishkarao}@google.com

Abstract

Word pronunciations, consisting of phoneme sequences and the associated syllabification and stress patterns, are vital for both speech recognition and text-to-speech (TTS) systems. For speech recognition phoneme sequences for words may be learned from audio data. We train recurrent neural network (RNN) based models to predict the syllabification and stress pattern for such pronunciations making them usable for TTS. We find these RNN models significantly outperform naive rule-based models for almost all languages we tested. Further, we find additional improvements to the stress prediction model by using the spelling as features in addition to the phoneme sequence. Finally, we train a single RNN model to predict the phoneme sequence, syllabification and stress for a given word. For several languages, this single RNN outperforms similar models trained specifically for either phoneme sequence or stress prediction. We report an exhaustive comparison of these approaches for twenty languages.

Index Terms: LSTM, pronunciation, syllabification, stress

1. Introduction

Knowing how words are pronounced is an essential ingredient in any state-of-the-art automatic speech recognition (ASR) or text-to-speech (TTS) system. For ASR, typically a manually curated pronunciation dictionary is used along with a grapheme-to-phoneme (G2P) model learned from data to cover any missing words. In most ASR systems, pronunciations consist of phoneme sequences, e.g. "foo"→ /f u/ (in the X-SAMPA phoneset).

For high-quality speech synthesis, the phoneme sequence also needs to have the correct syllabification and, in languages with stress, the correct stress placement (Taylor [1]). For example, the raw phoneme sequence of "hello" is /h E l oU/, but including syllable boundaries (indicated throughout using a dot) and stress information (indicated throughout using 0 for no stress, 1 for primary stress and 2 for secondary stress) would yield /h E 2 . l oU 1/.

2. Grapheme-to-phoneme Prediction

Knowing how words are pronounced is a challenging task, especially for irregular languages like English. Even the most sophisticated grapheme-to-phoneme (G2P) models only correctly predict the pronunciation for about 80% [2] of the words. Improving G2P models has been an active area of research, and most models fall into two categories: joint-sequence n-gram models [3, 4, 5] and sequence-to-sequence models [2]. N-gram based translation models are usally implemented as a weighted finite-state transducer (WFST) [6, 7]. G2P can also be approached as a sequence labeling problem using statistical techniques like conditional random fields (CRF) [8, 9].

Neural network approaches [10, 11] have also been proposed for G2P problems. Recurrent neural networks in particular have been shown to be effective for this sequence transcription task [2]. The state-of-the-art models are hybrids: joint n-gram models have been combined with CRF models (Wu et al. [12]), decision trees (Hahn et al. [13]) and RNNs (Rao et al. [2]).

Pronunciations can also be also learned directly from sample audio recordings; see e.g. Rutherford et al. [14] and Kou et al. [15]. This approach is useful for increasing coverage in domains with large and growing lexicons, such as geographical entities. These audio-driven pronunciations have a higher quality than an automated G2P model, but they cannot be reused in TTS systems without some way to to mark their syllable boundaries and stress placements.

Sproat et al. [16] show that models can be trained to predict stress placement given spelling, following Dou et al. [17] who report numbers on both stand-alone stress prediction as well as joint phoneme and stress prediction accuracy. Dou et al. [17] report numbers over four European languages and use the CELEX2 [18] lexicon. We instead use an internal human-curated lexicon that has more loan words, personal names and place names than CELEX2, since accurate performance on those types of words is critical to our application. Place names, for example, make up 3% of CELEX2's Dutch lexicon, which has about 120,000 words, but 8% of ours, which has about 435,000 words.

In the rest of this paper, we will explore using maximal-onset syllabification and pronunciation-based stress prediction. We will also report the accuracy of an approach using both spelling and syllabified pronunciation as input. Finally, we will report the accuracy of a single RNN that maps graphemes to syllabified, stress-marked phoneme sequences.

3. Syllabification and Stress Prediction

3.1. Rule-based Syllabification

Initially, we used a rule-based syllabifier to insert syllable boundaries into phoneme sequences using a finite-state transducer that implements maximal onset syllabification (Taylor [1]). Each language has a hand-coded inventory of consonants, vowels and allowed syllable onsets (Taylor [19]). The transducer accepts all sequences but has a preference for maximally long valid onsets before each vowel, resulting in maximal-onset syllabification when composed with a phoneme sequence that did not previously have syllable boundaries marked, e.g. a pronunciation learned from audio as described in section 2.

http://dx.doi.org/10.21437/Interspeech.2016-1419

3.2. LSTM-Based Stress Prediction Given Phonemes

Given a syllabified pronunciation, we still need to determine its stress pattern before it can be used for TTS. Initially, we used a pattern-based approach that uses the most frequent stress pattern for a given number of syllables in our human-curated lexicon. For example, given a sample lexicon with ten two-syllable words, if six of them have primary stress on the first syllable, we would assume that any two-syllable word has primary stress on its first syllable. This is a naive, low-accuracy approach.

Having established a baseline, we decided to use a long short-term memory (LSTM) recurrent neural network (RNN). RNNs use self-loops to retain information from previous inputs, making them able to take context within a sequence into account. An LSTM is a type of RNN that solves the vanishing gradient problem and can learn long-context behavior [20]. LSTMs have been particularly effective [2] for the G2P task. With an LSTM, the G2P task can be a sequence transcription task that converts a word's grapheme sequence into its phoneme sequence.

Similarly, we can train an LSTM to add stress markers to a syllabified pronunciation. The input sequence is the syllabified phoneme sequence decorated with the reserved symbol ⟨stress⟩ wherever we want a stress marker to appear. By convention in our system, there is a stress directly after the vowel in each syllable. The LSTM is trained to label each ⟨stress⟩ symbol with a number representing the syllable's level of stress. For example, primary and secondary stress are represented by 1 and 2. Symbols in other positions are labeled with E. Table 1 shows the input and target sequence for a sample pronunciation. The input and output symbols are encoded as one-hot vectors.

Input	o	⟨stress⟩	.	l	e	⟨stress⟩	.	o	⟨stress⟩
Output	E	1	E	E	E	2	E	E	2

Table 1: An example input and output from Spanish.

3.2.1. LSTM with Constraints

In some cases, the LSTM might fail to learn hard constraints imposed by the phonotactics of the language or the conventions of our notation. Given the input /h E ⟨stress⟩ . l oU ⟨stress⟩/ ("hello"), for example, the LSTM might assign secondary stress to both syllables, violating the expectations of our TTS system. The model could make this mistake even if no training item does.

We rectify this problem by encoding rules about the stress patterns as a finite state transducer (FST) that includes all valid stress patterns. We also represent the output of the LSTM (which are the stress posteriors) as a weighted FST. We then intersect both FSTs and pick the most probable path as the final stress pattern. In this way, we select the valid stress pattern that is the most probable according to the LSTM.

Several rules can be enforced at once during inference of the LSTM. For example, we could require exactly one primary stress to be present and disallow word-final stress if required by the language.

Table 2 shows the effect of the single-primary-stress constraint in three languages. Constraints were not used for the rest of the metrics in this paper, but we expect they would further improve accuracy.

Language	LSTM	LSTM+Constraints
da_dk	87.6	90.3
en_us	90.9	93.5
ru_ru	87.0	93.9

Table 2: Word-level accuracies for stress prediction with and without the constraint that all words must have exactly one syllable with primary stress. The test/train set split used here differs slightly from the set used in experiments below.

3.3. Spelling Features in LSTM-Based Stress Prediction

We further improve upon the accuracy of the LSTM-based stress prediction by adding the word's spelling as an input sequence parallel to the pronunciation. The LSTM now receives a second input from a one-hot representation of the current spelling symbol at each time step. Since this architecture depends on parallel sequences being the same length, the shorter sequence is padded at the beginning with a reserved symbol. Table 3 shows the inputs and target sequence for a sample pronunciation.

The inputs have no information about the alignment between the spelling and pronunciation sequences. Rather, the network must learn to propagate evidence about stress from the spelling sequence forwards or backwards in time. In languages like Greek where stress is usually marked with vowel diacritics, for example, the network must learn an implicit alignment model between written vowels and their corresponding stress markers.

Spelling	⟨pad⟩	⟨pad⟩	χ	ι	ω	τ	ώ	ν
Pronunciation	xj	o	⟨stress⟩	.	t	o	⟨stress⟩	n
Output	E	E	0	E	E	E	1	E

Table 3: An example data point from Greek. The written vowels ι and ω are not aligned with the stress marker positions in the output sequence, and due to the presence of a diphthong there are more written vowels than stress marker positions.

4. Experimental Set-Up Across Languages

4.1. Data

The training and test data sets for each language were derived from our internal human-curated pronunciation lexicons. For most languages, the lexicon has loan and foreign words, including words written in foreign alphabets. Processing and evaluation do not distinguish native and foreign words or symbols.

aakkosjärjestyksen
aa 1 k . k o 0 s . j ae 2 r . j e 0 s . t y 0 k . s e 0 n

Table 4: A lexical entry in Finnish, containing a word's spelling followed by its phonemic transcription. Numbers represent stress levels. Dots separate syllables.

For table 6, the lexicons were sorted by spelling, and every tenth item was reserved for testing. The same training and test sets were used for all tasks, with the exception of the constrained decoding comparison in table 2, which uses a different train-test split.

To generate an input sequence for the grapheme-to-phoneme models, the spelling is lowercased and split into graphemes. The output sequence is a phoneme sequence that can contain syllable boundaries and stress markers. The input and ouutput sequences are padded until they are twice the length of the input sequence.

To derive a training item for the stress models, the pronunciation undergoes two rewrites using Thrax grammars ([20]). For the input sequence, the stress markers are removed, and the symbol "⟨stress⟩" is inserted after each syllable nucleus. The output sequence is produced by the further step of replacing all stress markers with their corresponding stress value from the lexical entry, then replacing all non-stress symbols with "E". Spelling, if used by the model, is split into individual symbols in the same way as for the grapheme-to-phoneme model, except it is not lowercased. All three sequences are then padded to the same length. See table 5 for an example.

Graphemes	⟨pad⟩⟨pad⟩⟨pad⟩⟨pad⟩⟨pad⟩⟨pad⟩⟨pad⟩ ⟨pad⟩⟨pad⟩⟨pad⟩ a a k k o s j ä r j e s t y k s e n
Phonemes	aa ⟨stress⟩ k . k o ⟨stress⟩ s . j ae ⟨stress⟩ r . j e ⟨stress⟩ s . t y ⟨stress⟩ k . s e ⟨stress⟩ n
Output	E 1 E E E E 0 E E E E 2 E E E E 0 E E E E 0 E E E E 0 E

Table 5: An example training item in Finnish for the spelling-aware stress model. The non-spelling-aware stress model's sequences are identical except without the presence of the spelling or the necessity for padding.

After an output sequence has been decoded, the stress markers in the input sequence are replaced by the symbols at their corresponding positions in the output sequence. For example, if the LSTM maps "d i ⟨stress⟩ . n ⟨stress⟩" to "E E 1 E E E 0", the predicted pronunciation is "d i 1 . n er 0".

4.2. Training

All network weights are randomly initialized in $[-0.01, 0.01]$ and trained with a learning rate of 0.00005 with distributed asynchronous gradient descent using the DistBelief parameter server [21] across 200 replicas. Training was stopped after the best test set accuracy had not changed for at least three hours.

4.2.1. Stress and Syllabification Prediction

For models that only predict stress, we use a bidirectional LSTM [2] whose forward and backward layers each have 128 memory cells. An input layer contains a one-hot vector encoding of the current pronunciation symbol. For the spelling-aware model, a second input sequence contains a one-hot vector encoding of the current spelling symbol. In both models, the input layers are fully connected to the LSTM layers. The models are trained using a cross-entropy loss.

4.2.2. Pronunciation, Stress and Syllabification Prediction

For models that predict pronunciations, we use 3 stacked bidirectional LSTMs whose forward and backward layers each have 64 memory cells. The input layer is fully connected to the first pair of LSTM layers. Following [2], these models are trained with the CTC objective function [22] since the alignment between the input spelling and phoneme sequence is unknown. We improve accuracy by using dropout [22], dropping the LSTM output activations with probabilities of 0.25, 0.30 and 0.35 in stacked order from input to output.

5. Results Across Languages

5.1. Grapheme Features in Stress Prediction

In almost all languages, the stress-predicting LSTM improves over the pattern-based baseline. The exceptions were languages whose stress patterns were regular enough for the pattern-based approach to work well, e.g. the first syllable is always stressed. Adding grapheme features to the stress-predicting LSTM increases word-level accuracy for all languages except Hindi, where it makes little difference (see columns 1a and 1b in table 6 and figure 1).

The biggest improvements were seen in languages whose writing systems explicitly mark some or all stressed syllables, like Russian, Greek, Spanish, Italian and Portuguese. In Greek, whose writing system most consistently marks stressed vowels among the languages evaluated, the grapheme-aware model makes less than 2% of the amount of errors as the non-grapheme-aware model. Russian, Spanish, Italian and Portuguese saw error rate reductions in the 50-60% range. While accuracy in Russian is slightly lower than Sproat et al. [16] showed, our approach does not require any language-specific feature engineering.

5.2. Grapheme-to-pronunciation Prediction

Since stress, syllables and phonemes are all part of a pronunciation, we wanted to know if predicting them jointly creates a model with better stress-prediction abilities than even the spelling-aware stress predictor. We also wanted to know if having to also predict stress and syllabification causes the network to be better at predicting phonemes, similarly to work done for non-RNN approaches described by e.g. Dou et al. [17]. The answer to both questions is that it depends on the language.

When used as a stress predictor (table 6 column 1c), the grapheme-to-pronunciation LSTM outperforms this paper's best stress-only model (column 1b) for Spanish, Italian, Turkish and Ukrainian. The rest of the languages got worse, with Hindi and English regressing the most. We hypothesize that the amount of improvement correlates with orthographic transparency and that facts about a language's writing system and phonotactics can create a state of affairs where learning to predict phonemes creates intermediate representations that are useful for stress and syllable prediction, but hard for the stress-only architecture to learn.

When used as a phoneme predictor (column 2b or figure 2), the grapheme-to-pronunciation LSTM outperforms the same architecture trained only on phoneme sequences for all languages except Greek (where it stays the same), Czech, Finnish, Hindi and Ukrainian. This time the set of languages that were improved includes both ends of the orthrographic transparency spectrum from English to Spanish.

Column 3 reports the joint grapheme-to-pronnunciation model's accuracy. For comparison to an existing task, we ran the same model and evaluation process on syllabified, stress-marked CELEX2 Dutch data, which has 106,495 usable training items and 12,141 usable test items. The word-level accuracy was 82.2%, compared to the 65.7% when using our data set.

6. Conclusions

We showed that RNNs are capable of predicting stress placement accurately for large test sets across twenty languages, containing words commonly used in industry applications of speech systems, e.g. personal names and geographical names

Language	Training items	Test items	0. Training set accuracy for pattern-based stress	1a. Stress LSTM given phonemes	1b. Stress LSTM given phonemes and graphemes	1c. Implicit stress model in LSTM-G2P	2a. LSTM that predicts phonemes only	2b. Implicit phoneme model in LSTM-G2P	3. LSTM-G2P
cs_cz	82,434	9,161	99.9	100	100	98.3	90.1	89.9	89.6
da_dk	230,487	25,627	58.7	89.7	90.9	84.8	60.8	63.9	58.7
de_de	618,740	68,757	83.5	88.9	91.2	88.3	79.4	81.0	73.8
el_gr	106,221	11,806	45.8	65.2	99.6	98.1	97.2	97.2	96.8
en_gb	340,265	37,814	88.3	90.8	93.1	75.1	61.9	64.2	59.5
en_us	619,716	69,220	74.6	88	89.8	68.4	50.8	53.2	48.4
es_es	410,180	45,575	69.1	79.5	88.6	95.5	75.0	77.3	76.7
es_us	390,305	43,371	70.8	84.0	91.3	95.7	76.8	78.5	77.6
fi_fi	97,935	10,877	52.4	95.8	96.4	96.3	93.9	93.6	91.6
hi_in	98,755	10,975	74.9	87.6	87.3	72.2	60.3	59.7	49.7
hu_hu	112,643	12,514	99.9	99.8	99.8	98.0	90.4	91.0	88.9
it_it	796,603	88,552	76.7	92.6	95.9	97.2	82.1	85.0	84.6
nb_no	122,836	13,646	41.8	73.2	76.0	71.9	58.4	69.0	54.7
nl_nl	391,476	43,506	72.8	81.7	84.4	82.7	74.1	75.8	65.7
pl_pl	229,353	25,484	98.0	97.3	97.8	96.5	92.2	92.5	91.0
pt_br	207,782	25,043	71.1	89.0	94.0	86.3	75.6	76.5	72.9
ru_ru	1,669,183	185,686	45.3	87.4	93.6	85.0	76.3	79.4	73.3
sv_se	144,705	16,085	45.0	80.9	84.9	82.1	72.3	74.1	66.4
tr_tr	94,866	10,541	68.4	80.6	84.1	88.2	89.0	89.4	80.8
uk_ua	95,735	10,648	55.0	70.0	73.2	83.0	98.5	98.1	80.7

Table 6: Corpus sizes and word-level accuracies as percentages. Column 0 is the pattern-based model's accuracy on its training set. Columns 1a and 1b report the word-level accuracies of the spelling-unaware and spelling-aware stress-predicting LSTMs. Column 1c reports the accuracy of the grapheme-to-pronunciation LSTM when used as a stress predictor. Columns 2a and 2b compare the accuracies of the grapheme-to-pronunciation LSTM with and without the removal of stress and syllable information from the corpus. Column 3 reports the full accuracy of the grapheme-to-pronunciation LSTM. Columns are comparable if they are not separated by a double vertical line.

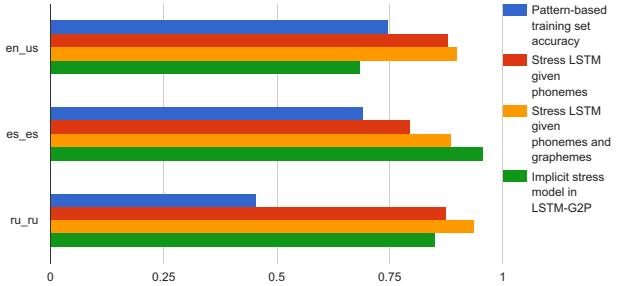

Figure 1: Stress model accuracies for selected languages.

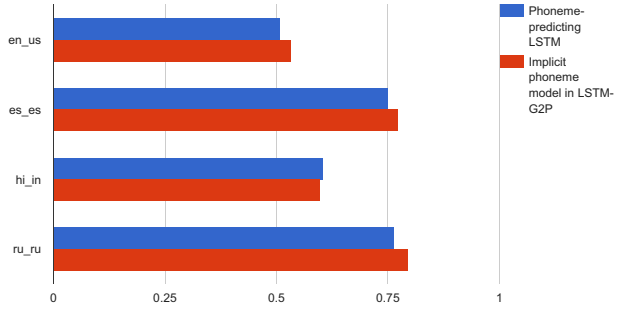

Figure 2: Phoneme model accuracies for selected languages.

in addition to regular words. These RNNs did not require any language-specific tuning. In almost all languages, using both graphemes and phonemes as input to the stress prediction models improves accuracy over using graphemes alone. In some languages, further gains were achieved by using the implicit stress models learned when training RNNs to jointly predict phoneme sequences, syllabification and stress.

7. Acknowledgements

The authors would like to thank Fuchun Peng for advice on pronunciation and stress modeling, and Françoise Beaufays for inspiring the end-to-end models. We would also like to thank Anna Garbier, Jonas Fromseier Mortensen, Jeremy O'Brien and Aly Pitts for their help in setting up the Thrax grammars across many languages, as well as the many linguists who tirelessly helped curate our pronunciation lexicons.

8. References

[1] P. Taylor, *Text-to-Speech Synthesis*. Cambridge: Cambridge University Press, 2009, ch. 7, pp. 184–189.

[2] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks," in *Proceedings of ICASSP*, 2015.

[3] S. Deligne, F. Yvon, and F. Bimbot, "Variable-length sequence matching for phonetic transcription using 2-level multigrams," in *Proceedings of the Conference of the European Speech Communication Association (EuroSpeech)*, 1995, pp. 2243–2246.

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

[5] L. Galescu and J. F. Allen, "Pronunciation of proper names with a joint n-gram model for bi-directional grapheme-to-phoneme conversion," in *Proceedings of InterSpeech*, 2002.

[6] J. Novak, P. Dixon, N. Minematsu, K. Hirose, C. Hori, and H. Kashioka, "Improving WFST-based G2P conversion with alignment constraints and RNNLM N-best rescoring," in *Proceedings of InterSpeech*, 2012.

[7] J. R. Novak, N. Minematu, and K. Hirose, "Failure transitions for joint n-gram models and G2P conversion," in *Proceedings of InterSpeech*, 2013.

[8] D. Wang and S. King, "Letter-to-sound pronunciation prediction using conditional random fields," *IEEE Signal Processing Letters*, vol. 18 (2), pp. 122 – 125, 2011.

[9] P. Lehnen, A. Allauzen, T. Lavergne, F. Yvon, S. Hahn, and H. Ney, "Structure learning in hidden conditional random fields for grapheme-to-phoneme conversion," in *Proceedings of InterSpeech*, 2013.

[10] E. B. Bilcu, "Text-to-phoneme mapping using neural networks," Ph.D. dissertation, Tampere University of Technology, 2008.

[11] T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce english text," *Complex systems*, vol. 1, no. 1, pp. 145–168, 1987.

[12] K. Wu, C. Allauzen, K. Hall, M. Riley, and B. Roark, "Encoding linear models as weighted finite-state transducers," in *Proceedings of InterSpeech*, 2014.

[13] S. Hahn, P. Vozila, and M. Bisani, "Comparison of grapheme-to-phoneme methods on large pronunciation dictionaries and LVCSR tasks," in *Proceedings of InterSpeech*, 2012.

[14] A. Rutherford, F. Peng, and F. Beaufays, "Pronunciation learning for named-entities through crowd-sourcing," in *Proceedings of InterSpeech*, 2014.

[15] Z. Kou, D. Stanton, F. Peng, F. Beaufays, and T. Strohman, "Fix it where it fails: Pronunciation learning by mining error corrections from speech logs," in *Proceedings of ICASSP*, 2015.

[16] R. Sproat and K. Hall, "Applications of maximum entropy rankers to problems in spoken language processing," in *Interspeech 2014*, 2014.

[17] Q. Dou, S. Bergsma, S. Jiampojamarn, and G. Kondrak, "A ranking approach to stress prediction for letter-to-phoneme conversion," in *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*. Suntec, Singapore: Association for Computational Linguistics, August 2009, pp. 118–126.

[18] R. Baayen, R. Piepenbrock, and L. Gulikers, "Celex2 ldc96l14," *Web Download. Philadelphia: Linguistic Data Consortium*, 1995.

[19] P. Taylor, *Text-to-Speech Synthesis*. Cambridge: Cambridge University Press, 2009, ch. 8, pp. 192–210.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9(8), p. 1735–1780, 1997.

[21] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large scale distributed deep networks," in *NIPS*, 2012.

[22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of ICML*, 2006.