Combining Multiple-Type Input Units using Recurrent Neural Network for LVCSR Language Modeling

Vataya Chunwijitra, Ananlada Chotimongkol, Chai Wutiwiwatchai

NECTEC, National Science and Technology Development Agency (NSTDA),
112 Pahonyothin Road, Pathumthani, 12120, Thailand
{vataya.chunwijitra, ananlada.chotimongkol, chai.wutiwiwatchai} @nectec.or.th

Abstract

In this paper, we investigate the use of a Recurrent Neural Network (RNN) in combining hybrid input types, namely word and pseudo-morpheme (PM) for Thai LVCSR language modeling. Similar to other neural network frameworks, there is no restriction on RNN input types. To exploit this advantage, the input vector of a proposed hybrid RNN language model (RNNLM) is a concatenated vector of word and PM vectors. After the first-pass decoding with an n-gram LM, a word-based lattice is expanded to include the corresponding PMs of each word. The hybrid RNNLM is then used to re-score the hybrid lattice in the second-pass decoding. We tested our hybrid RNNLM on two recognition tasks: broadcast news transcription and mobile speech-to-speech translation. The proposed model achieved better recognition performance than a baseline word-based RNNLM as hybrid input types provide more flexible unit choices for language model re-scoring. The computational complexity of a full-hybrid RNNLM can be reduced by limiting the input vector to include only frequent words and PMs. In a reduced-hybrid RNNLM, the size of the input vector can be reduced by half which can considerably save both training and decoding time without affecting recognition accuracy.

Index Terms: Recurrent neural network language model, Pseudo-morpheme, hybrid language model, LVCSR

1. Introduction

The vocabulary of any active language continues to grow as new words are introduced everyday. This poses a challenge on building a language model (LM) for a large vocabulary continuous speech recognition (LVCSR) system. As the vocabulary size grows, an unrestricted or open-vocabulary LM requires a larger amount of resources, e.g. memory and computational time, for both training and decoding. A hybrid LM of word and sub-word units has been shown to be resource efficient for an open-domain LVCSR system as sub-lexical units can be combined to form new words and, thus, reduce vocabulary size. In [1–3], a hybrid LM has been shown to alleviate the problem of OOV words in English, German and Turkish respectively. Moreover, different levels of linguistic information from multiple input types in a hybrid LM can be combined to better predict word probability. In [4, 5], characters were combined with words to add another type of constraints in Chinese hybrid LMs.

Since there is neither inflection nor derivative in Thai, a syllable-based unit called pseudo-morpheme (PM) was used as a sub-lexical unit in a hybrid LM instead of a morpheme-based unit as in morphologically rich languages. The hybrid word-PM model could reduce the OOV rate while still achieved a comparable recognition error rate to a word-based model with smaller lexicon size [6]. Furthermore, according to Thai writing rules, PM is more deterministic when compared with word and has been shown to help alleviate a word segmentation problem [7].

Besides a choice of a suitable sub-lexical unit which largely depends on the characteristic of each language, a unit combination method is another issue that has to be considered in a hybrid LM. In [6], words and sub-words were combined using a traditional n-gram model where different types of units were treated equally in a statistical LM. In [4], word and character LMs were combined using Weighted Finite State Transducer (WFST) composition operation to form a multi-level LM. A hybrid WFST LM achieved higher recognition accuracy than a hypothesis based combination method using ROVER [8]. A neural network (NN) was used in [5] to model a hybrid word-character LM for Mandarin and achieved better recognition results than a traditional n-gram model. The key advantage of and NNLN over an n-gram LM is its ability to learn a distributed representation of words in continuous space [9].

In this paper, we investigate the use of a Recurrent Neural Network (RNN) in combining words and pseudo-morphemes for Thai LVCSR language modeling. RNN is chosen from its ability to model longer history not only n – 1 previous words through recurrent connections between its hidden layer and input layer as detailed in [10]. Therefore, it can capture long context patterns instead of fixed length context as in an NNLN or an n-gram LM. Moreover, RNN can naturally model multiple input types in a hybrid LM as there is no restriction on its input types. In our proposed hybrid RNNLM, we use a hybrid vector of words and PMs as an input vector. Unlike [4, 5], the output of our hybrid RNNLM is also a hybrid word-subword output. To be able to output both words and PMs, a word N-best list obtained from the first-pass decoding with an n-gram LM is expanded to include the corresponding PMs of each word. The RNNLM is then applied in the second-pass re-scoring. RNNLMs could be directly applied to the n-best lattice with more complex algorithms such as RNNLM lattice re-scoring [11] and a cache-based RNNLM which can be used straightaway in the first-pass decoding [12]. Besides the application of RNN, we also explore several hybrid input representations to optimize both recognition accuracy and computational time. In addition to a full-hybrid RNNLM which takes both a word sequence and a PM sequence as its input, two variations of reduced-hybrid RNNLMs are proposed to decrease computational complexity. By using two types of units, the vocabulary size of the full-hybrid RNNLM could be twice the size of the word-based RNNLM. In the first reduced-hybrid RNNLM variation, the vocabulary size is reduced to be equal to the size of the word-based RNNLM vocabulary by including only frequent words and PMs. In another variation, a mixed of word and PM
The architecture of the proposed RNNLM is demonstrated in Sect. 3.1. Model structure

sequence is used as an input instead.

This paper is organized as follows: Sect. 2 explains the characteristics of Thai text together with a pseudo-morpheme (PM), a sub-lexical unit in Thai. Sect. 3 describes our proposed hybrid RNNLM framework for combining different input types. Recognition results of the proposed word-PM RNNLM on two Thai LVCSR tasks are discussed in Sect. 4. We finally conclude our work and discuss future directions in Sect. 5.

2. Thai lexical units

Thai is a non-segmented script language, i.e. there is no boundary marker between words while boundary markers on phrase and sentence levels are ambiguous as space is used to mark both phrase and sentence boundaries. At a level of word, there is neither inflection nor derivative in Thai. Thus, another type of sub-lexical units should be used instead of morpheme. As a letter in Thai is a phonogram which roughly represents a phoneme or combination of phonemes, Thai word could be segmented into a set of syllable-like units. A basic Thai textual syllable consists of four components, initial consonant, vowel, final consonant, and tone, as shown in Fig. 1. The corresponding phoneme (or phonemes) of each component in IPA is also illustrated.

A syllable-like unit called pseudo-morpheme (PM) was used as a sub-lexical unit for Thai in a hybrid language model [6]. The word ‘ความมั่นคง’ /khwāam-mān-khōŋ/ (stability), in Fig. 1, consists of three PMs: ‘ความ’ /khwāam/, ‘มั่น’ /mān/, and ‘คง’ /khōŋ/. Some textual syllables may have multiple initial consonants, vowel forms or final consonants while some syllables may have components omitted. Nonetheless, according to Thai writing rules, PM is more deterministic when compared with word. Given a word or a string of text, PMs can be determined quite accurately with an automatic segmentation tool [13].

3. A hybrid recurrent neural network language model

3.1. Model structure

The architecture of the proposed RNNLM is demonstrated in Fig. 2. In this work, we employ a standard class-based RNNLM [14]. Each word is assigned to exactly one class based on its frequency in training data. A class can be considered as a frequency bin. The network is represented by three layers (input layer, hidden layer and output layer) and corresponding weight matrices (matrix $\mathbf{U}$ between the input layer and the hidden layer and matrices $\mathbf{V}$ and $\mathbf{Z}$ between the hidden layer and the output layer). Unlike a conventional word-based RNNLM, the input vector $x(t)$ of the hybrid RNNLM is formed by concatenating a hybrid vector $h(t)$, instead of a word vector $w(t)$, with a vector $s(t-1)$ as represented by the following equations:

$$h(t) = [w(t)^T p(t)^T]^T$$

$$x(t) = [h(t)^T s(t-1)^T]^T$$

where $h(t)$ is a concatenated vector of a word vector $w(t)$ and a PM vector $p(t)$, and $s(t-1)$ is the output from the hidden layer at time $t-1$. Using the hybrid vector, the hybrid RNNLM can simultaneously integrate multiple input types in its input layer. The hidden layer employs a sigmoid activation function:

$$s_j(t) = f\left(\sum_i x_i(t) u_{ij}\right), \quad f(a) = \frac{1}{1 + e^{-a}}. \quad (3)$$

where $u_{ij}$ is an element in matrix $\mathbf{U}$, and $j$ is an index to hidden neurons in the hidden layer. The output layer is divided into two parts: the first part outputs the probability distribution over all classes $c(t)$ while the second part outputs the probability distribution over the hybrid units $b(t)$ that belong to a specific class, the one that contains the predicted hybrid unit:

$$c_m(t) = g\left(\sum_j s_j(t) z_{mj}\right)$$

$$b_k(t) = g\left(\sum_j s_j(t) v_{kj}\right), \quad (5)$$

where $z_{mj}$ and $v_{kj}$ are an element in matrix $\mathbf{Z}$ and $\mathbf{V}$ respectively. To ensure that all output values are between 0 and 1, and their summation is equal to 1, the output layer employs a softmax activation function:

$$g(a_m) = \frac{e^{a_m}}{\sum_k e^{a_k}}. \quad (6)$$

Finally, the probability of a predicted hybrid unit $h(i)$ is then computed as

$$P(h(i|s(t))) = P(h(i|c_i,s(t))) P(c_i|s(t)), \quad (7)$$

where $h_i$ is an index of the hybrid unit and $c_i$ is its class.

3.2. A hybrid input representation for LM training

In this section, we discuss an input representation for hybrid RNNLM training data and a vocabulary list. As a hybrid RNNLM can take two types of input units, the training text consists of two sets: a word sequence set and a PM sequence set. Both data sets share the same content, but have different segmentations. Fig. 3 illustrates various segmentation types of the same text utterance. The same text in (a) is segmented into a sequence of words in (b) and a sequence of PMs in (c). A space is used to illustrate the boundary between units.

After specifying training data, a vocabulary list is constructed. Typically, not every word found in the training data is included in the vocabulary list as low frequency words could be typos. In practice, only the top-$N$ most frequent words are included. For a hybrid RNNLM, each input type has its own
set of vocabulary. Let \( N \) be the size of word vocabulary and \( M \) be the size of PM vocabulary. The vocabulary size of a hybrid word-PM RNNLM is \( N + M \) which could be twice the size of the word-based RNNLM. The size of the vector \( h(t) \) in Equation 2 is equal the vocabulary size. The hybrid RNNLM which uses a full vocabulary of both word and PM similar to [5], or a full-hybrid RNNLM, may suffer from the cost of computational complexity. Moreover, as our hybrid RNNLM also has a hybrid output, the output layer \( b(t) \) has the same dimensionality as \( h(t) \). As the output layer contains one neuron for each word or PM in the vocabulary, it may be infeasible to train the model with large vocabulary size.

To decrease computational complexity of the full-hybrid RNNLM (H-F), two variations of reduced-hybrid RNNLMs are proposed. In the first variation (H-R1), the vocabulary size is reduced to be equal to the size of the word-based RNNLM vocabulary (\( N \)) by including only frequent words and PMs. Let \( N' \) and \( M' \) be the size of the top-\( N' \) most frequent words and the top-\( M' \) most frequent PMs respectively, \( N' + M' = N \) in H-R1. In the second variation (H-R2), a hybrid word-PM sequence is used as an input instead of two separate word and PM sequences. Its vocabulary size is also limited to \( N \). The slight difference from H-R1 is the composition of the vocabulary. Since an input text is a mixed of words and PMs, the top-\( N' \) most frequent words are determined first and kept as word units in the vocabulary. Next, the less frequent words are segmented into PMs. The top-\( M' \) most frequent PMs from this list are then added into the vocabulary where \( N'' + M'' = N \). Fig. 3 (d) illustrates a hybrid sequence where a frequent word "แนวโน้ม" is kept as a word while an infrequent word "ใช้อำนาจ" is segmented into three PMs: "ใช้" and "อำนาจ". We note that, for H-R2, the list of PMs and their frequency come from the less frequent words not all words in the training data. In H-R1 and H-R2, the size of the vector \( h(t) \) is also decreased. The reduction in computational time comparing with the full-hybrid RNNLM is discussed in Sect. 4.3.

3.3. Hybrid lattice decoding and re-scoring

To perform automatic speech recognition with an RNNLM, we employ a two-pass decoding scheme. In the first pass, a decoder uses an acoustic model and a 3-gram LM to generate multiple recognition hypotheses which can be compactly represented in a data structure called word lattice. An N-best word list is then extracted from the word lattice as shown in Fig. 4 (a). In the second pass, an RNNLM is applied to re-score the hypotheses in the N-best list obtained from the first step. To be able to output both words and PMs, the word-based hypotheses from the first pass has to be expanded to include the corresponding PMs of each word before re-scoring. For the full-hybrid RNNLM which takes both a word sequence and a PM sequence as its input, every word in the hypotheses are split into PMs as shown in Fig. 4 (b). The acoustic score and LM score of each PM are calculated by the linear distribution of its corresponding word scores. The reduced-hybrid H-R1 which takes the same input representation as H-F but use a reduced vocabulary set also uses the word-PM N-best list shown in Fig. 4 (b). For the reduced-hybrid H-R2 where frequent words are represented as word units while infrequent words are represented by PMs, a hybrid N-best list illustrated in Fig. 4 (c) is used in the second-pass re-scoring. In the second-pass re-scoring, a new LM score is obtained by interpolating the probability from RNNLM with the first-pass n-gram LM score. Then, the N-best hybrid list with the new LM scores is reconstructed into a hybrid lattice. Finally, the new best hypothesis is chosen based on the re-scored score.

4. Experiments

We evaluated our proposed hybrid RNNLMs on two recognition tasks: broadcast news transcription and speech-to-speech translation (S2S). Training and test data along with the experimental conditions are described in Sect. 4.1. Recognition performance and run-time efficiency are reported in Sect. 4.2 and 4.3.

4.1. Experimental conditions

LM training data contain 9.4M words from three text corpora, BEST [15], LOTUS-BN [16], and HIT-BTEC [17]. As these corpora cover variety of domains, e.g. law, news, and travel, with the vocabulary size of 143K, they are good resources for training a hybrid LM for open-vocabulary LVCSR. Five variations of RNNLMs were investigated: word-based (W), PM-based (PM), full-hybrid (H-F), and two variations of reduced-hybrid (H-R1 and H-R2). The PM RNNLM was trained in the same fashion as the conventional word RNNLM except that the input unit is PM instead of word. In all experiments, the class-based RNNLMs were trained with 2 iterations of Back-propagation through time (BPTT) [18], 400 hidden neurons, and 400 classes. It has been shown in [6] that with this size of training data the vocabulary size of approximately 35K is optimal for word-based LM. Hence the vocabulary size of the word RNNLM (\( N \)) is set to 35K. For fair comparison, the vocabulary size of the PM RNNLM (\( M \)) is also set to 35K. For H-F, the vocabulary size is \( N + M \) which is 70K as detailed in Sect. 3.2. For H-R1, by including only words that occur more than 3 times, the number of words (\( N' \)) is 30K which makes the number of PMs (\( M' \)) becomes 5K. The same sizes of word and PM units are used in H-R2. RNNLMs were trained with the RNNLM toolkit [19] while n-gram LMs with modified Kneser-Ney smoothing were built using kaldi LM toolkit [20]. In RNNLM training, 1% or 10K words were excluded from the training set for validation.
testing. The size of the word-based N-best list obtained from the first-pass decoding is set to be at most 100 hypotheses for each utterance. In the second-pass re-scoring, RNNLMs were interpolated with the 4-gram LM using a weight of 0.25.

Acoustic model training data comprises of 224 hours of speech from LOTUS [21], LOTUS-BN [16], and VoiceTra4U. VoiceTra4U is a speech translation application in sport and travel domains developed under the Universal Speech Translation Advanced Research (U-STAR) project (http://www.ustar-consortium.com/). 22 hours of speech were recorded on mobile devices in real environment. We used the Kaldi Speech Recognition Toolkit [20] to first train a conventional GMM-based acoustic model, then applied the Minimum Phone Error (MPE) discriminative training technique described in [22]. Each frame of speech data was converted into a sequence of 39 dimensional feature vectors of 12 MFCCs augmented with log energy, their first and second derivatives. We used a frame length of 25 ms. with 10 ms. window shift each time. Features from a context window of 3 frames to the left and right are also included. A Linear Discriminate Analysis (LDA) was also applied to the feature space to reduce feature dimensions to 40.

We evaluated our approach with two different recognition tasks: broadcast news transcription (BN), S2S application VoiceTra4U (VT). The BN test set consists of 3,140 utterances of 2 male and 1 female speakers taken from the LOTUS-BN evaluation test set (ET). The VT test set consists of 1,916 utterances of VoiceTra4U not included in the training set.

### 4.2. Recognition performance

Since multiple types of units, i.e. word and PM, are used in the hybrid LM, we measure the recognition error rate based on the smaller unit, PM. PM Error Rate (PER) was reported in all experiments. Table 1 shows PER results from first pass decoding using word-based 3-gram LM (3gr) and second pass re-scoring with word-based 4-gram LM (4gr) and various RNNLMs.

As expected, all the second-pass re-scoring results were better than the first-pass results. When compared RNNLMs with a 4-gram LM in the second-pass re-scoring, all RNNLMs obtained better recognition results than the 4-gram LM. Among various RNNLMs, H-F achieved the lowest PER in all test sets. When compared with a conventional word-based RNNLM (W), the best proposed hybrid RNNLM obtained 1.68% relative PER reduction on average. As many applications rely on a word unit, we string together multiple-type output units into an utterance and then re-segment it into words using a word segmentation tool. The best hybrid RNNLM, a full-hybrid, obtained 1.01% WER reduction when compared with a conventional word-based RNNLM. From result analysis, we found that the word-based RNNLM sometimes made mistake by choosing a long word or a compound word when its can be acoustically confused with correct words in an input utterance. A hybrid word-PM RNNLM, on the other hand, has more flexible unit choices as it can output both word and sub-word units, and thus can avoid this kind of mistakes. We also compared the proposed hybrid RNNLM with a hybrid n-gram LM in [6]. With the same acoustic model, the PER of the hybrid n-gram LM is 19.70% on average. With the use of RNN for combining multiple-type input units, 3.60% relative PER reduction can be obtained.

### 4.3. Computational efficiency

In this section, we analyzed the computational time of the RNNLMs. Table 2 shows the RNNLMs training time, and the second-pass decoding time on a PC with 98GB of memory and 24 cores 2.67 GHz CPU. Since H-F and H-R1 used both word and PM sequences as an input when trained the models, they used much longer training time than other types of RNNLMs. Their decoding times are almost twice when compared with other models. H-R2 which takes a single hybrid sequence of words and PMs as its input has the lowest training and decoding times among all RNNLM variations. In terms of recognition performance, when compared with H-F which obtained the best recognition result, H-R2 achieved almost the same PER. Both the training and decoding time can be saved by more than 50% in H-R2 without affecting recognition accuracy. When compared with the 4-gram LM, which has about 6 minutes training time and 17 minutes decoding time, all RNNLMs have much longer training time but have slightly faster decoding time while also achieve better recognition results. The 4-gram LM has longer decoding time due to its larger vocabulary size.

### 5. Conclusions

We proposed a hybrid RNNLM framework for modeling multiple-type input units, namely word and sub-word units. Several modifications are made to a standard RNNLM framework to model multiple-type input units in the proposed hybrid RNNLM. A concatenated vector of word and pseudo-morpheme vectors, or a hybrid vector, is used as an input vector instead of a word vector. Since the output of our proposed hybrid RNNLM is also a hybrid word-subword output, a word N-best list from the first-pass decoding has to be expanded into a hybrid N-best list for the second-pass re-scoring with a hybrid RNNLM. Several hybrid input representations were also explored to optimize both recognition accuracy and computational time.

We tested our hybrid RNNLMs on two Thai LVCSR tasks: broadcast news transcription and speech-to-speech translation. The best proposed hybrid RNNLM, a full-hybrid, obtained 1.68% relative PER reduction and 1.01% WER reduction when compared with a conventional word-based RNNLM as hybrid input types provide more flexible unit choices for LM re-scoring. Nevertheless, a full-hybrid RNNLM which takes both a word sequence and a PM sequence as its input requires a lot of computation. The computational complexity of a full-hybrid RNNLM can be reduced by using a hybrid word-PM sequence as an input instead. With the proposed reduced-Hybrid RNNLM, both the training and decoding time can be saved by more than 50% without effecting recognition accuracy. In the future, we plan to also use a hybrid n-gram model in the first-pass decoding to explore the benefit of a hybrid word lattice.

---

**Table 1: PM error rate of first pass decoding and second pass re-scoring.**

<table>
<thead>
<tr>
<th>Tasks</th>
<th>First-pass (3gr)</th>
<th>Second-pass RNNLMs</th>
<th>W</th>
<th>PM</th>
<th>H-F</th>
<th>H-R1</th>
<th>H-R2</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Word</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#PM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BN</td>
<td>19.03</td>
<td>20.06</td>
<td>19.18</td>
<td>19.18</td>
<td>18.79</td>
<td>18.83</td>
<td>18.82</td>
</tr>
<tr>
<td>VT</td>
<td>20.07</td>
<td>19.49</td>
<td>19.44</td>
<td>19.94</td>
<td>19.18</td>
<td>19.18</td>
<td>19.18</td>
</tr>
<tr>
<td>AVG</td>
<td>19.30</td>
<td>19.78</td>
<td>19.71</td>
<td>19.56</td>
<td>18.89</td>
<td>19.01</td>
<td>19.00</td>
</tr>
</tbody>
</table>

**Table 2: Training and decoding time (h=hour, m=minute, s=second).**

<table>
<thead>
<tr>
<th>RNNLMs</th>
<th>Training</th>
<th>Decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>33h12m</td>
<td>08h01s</td>
</tr>
<tr>
<td>PM</td>
<td>35h30m</td>
<td>09m18s</td>
</tr>
<tr>
<td>H-F</td>
<td>75h50m</td>
<td>15m35s</td>
</tr>
<tr>
<td>H-R1</td>
<td>60h02m</td>
<td>15m11s</td>
</tr>
<tr>
<td>H-R2</td>
<td>28h02m</td>
<td>07m43s</td>
</tr>
</tbody>
</table>
6. References


