Online Sentence Segmentation for Simultaneous Interpretation using Multi-Shifted Recurrent Neural Network

Xiaolin Wang Masao Utiyama Eiichiro Sumita
Advanced Translation Research and Development Promotion Center
National Institute of Information and Communications Technology, Japan
{xiaolin.wang,mutiyama,eiichiro.sumita}@nict.go.jp

Abstract

This paper is devoted to developing a recurrent neural network (RNN) solution for segmenting the unpunctuated transcripts generated by automatic speech recognition for simultaneous interpretation. RNNs are effective in capturing long-distance dependencies and straightforward for online decoding. Thus, they are ideal for the task compared to the conventional n-gram language model (LM) based approaches and recent neural machine translation based approaches. This paper proposes a multi-shifted RNN to address the trade-off between accuracy and latency, which is one of the key characteristics of the task. Experiments show that our proposed method improves the segmentation accuracy measured in F_1 by 21.1% while maintains approximately the same latency, and reduces the BLEU loss to the oracle segmentation by 28.6%, when compared to a strong baseline of the RNN LM-based method. Our online sentence segmentation toolkit is open-sourced [1] to promote the field.

1 Introduction

Simultaneous interpretation (SI) is to translate one spoken language into another spoken language in real time. Automated SI typically requires integrating two fundamental natural language processing technologies – automatic speech recognition (ASR) and machine translation (MT). Both technologies have become quite capable after half a century's intensive study, but one problem makes it difficult for them to work together – the raw transcripts generated by ASR contains no segmentation (see Table 1 for an example), while MT expects segmented sentences as input.

even cats were watching this video cats were watching other cats watch this video but what 's important here is the creativity that it inspired amongst this techie geeky internet culture there were remixes someone made an old timey version and then it went international there were remixes someone made an old timey version

Table 1: Illustration of Input for Sentence Segmentation

Online sentence segmentation smoothly bridges the gap between ASR and MT through segmenting the transcripts generated by ASR engines into sentences in real time. As a matter of fact, the task is non-trivial. The example presented in Table 1 is extracted from a TED talk [2], which is used in the experiments of this paper. Readers may find the raw sequence of words difficult to read. However, the readability is greatly improved once it is segmented as follows,

- even cats were watching this video
- cats were watching other cats watch this video
- but what 's important here is the creativity that it inspired amongst this techie geeky internet culture
- there were remixes
- someone made an old timey version

[1] https://github.com/arthurxlw/cytonNss

[2] https://www.ted.com/

- and then it went international
- there were remixes someone made an old timey version

Therefore, sentence segmentation is a meaningful natural language processing task. Correctly segmenting an ASR transcript requires a certain level of understanding the content.

This paper proposes a multi-shifted RNN to approach the problem of online sentence segmentation, which shifts target signals by multiple durations of time as illustrated by Table 2. This design emphasizes two central elements of the task – accuracy and latency. Usually, predicting a sentence boundary immediately after a last input word is not wise. Instead, waiting and checking a few words to make sure that a new sentence has started can raise the accuracy at the cost of latency. Shifting the target signals n time stamps right implements the idea of waiting and checking more words, but the optimal n varies on different textual contexts. Therefore, the proposed network learns multiple shifted target signals during training, and maintains multiple pathway of trading latency with accuracy during test. Experimental results demonstrate the effectiveness of our proposed method.

The contributions of this paper include,

- proposing a multi-shifted RNN for online sentence segmentation;
- achieving competitive performance on a real-world corpus;
- releasing the source code for reproducibility.

The rest of the paper is organized as follows. Section 2 reviews a baseline n-gram LM-based method which serves as a foundation of our method. Section 3 describes our method from the aspects of training, decoding and tuning. Section 4 presents the experiments. Section 5 compares our method with some related works. Section 6 concludes this paper with a description on future works.

2 Baseline: N-gram LM-based Method

N-gram LMs are used to segment unpunctuated transcripts by Stolcke et al. (1996; 1998) and Wang et al. (2016). They view sentence boundaries as hidden events occurring between the input words, and use n-gram LMs to compute the likelihood of the input words with or without sentence boundaries. Among them, the work of Wang et al.(2016) is the most related to this paper, because it addresses segmenting in an online manner for SI. Suppose an input sequence of words is $\cdots$, w_{t-1}, w_t, w_{t+1}, $\cdots$. The following two hypotheses are considered,

- *Hypothesis I*: there is no sentence boundary after the word w_t, which assumes that the underlying input remains the same as $\cdots$, w_{t-1}, w_t, w_{t+1}, $\cdots$.
- *Hypothesis II*: there is a sentence boundary after the word w_t, which assumes that the underlying input is $\cdots$, w_{t-1}, w_t, $\langle/s\rangle$, $\langle s\rangle$, w_{t+1}, $\cdots$.

The segmentation is predicted by comparing the probabilities of the two sequences as,

$$\begin{aligned} s_t &= \frac{P_t^{\langle \mathrm{II}\rangle}}{P_t^{\langle \mathrm{I}\rangle}} \\ &= p(\langle/s\rangle|w_{t-o+2}^t) \cdot \frac{p(w_{t+1}|\langle s\rangle)}{p(w_{t+1}|w_{t-o-2}^t)} \\ &\quad \cdot \prod_{k=t+2}^{t-o+1} \frac{p(w_k|w_{t+1}^{k-1}, \langle s\rangle)}{p(w_k|w_{k-o+1}^{k-1})} \end{aligned} \tag{1}$$

where o is the order of a n-gram LM, and s_t is the confidence score of placing a sentence boundary after w_t. The left hand of the formula has one item for $\langle s\rangle$, w_{t+1}, ..., w_{t+o-1}, respectively. Theoretically, the o-1 future words w_{t+1}, ..., w_{t+o-1} are required when predicting the segmentation for the time stamp t. Empirically, it is found that 1 or 2 future words is enough for accuracy while having the merit of low latency.

N-gram LM-based methods are effective. However, they have two shortages. First, n-gram LMs cannot capture the long-distance dependencies required by the task, as the length of a sentence is typically larger than the order of n-gram LMs. Second, they are generative methods as the prediction is made by comparing the generative probability of two sequences. The accuracy of generative methods is known to be lower than that of discriminative methods. In the paper, we explore using RNN LM (Mikolov et al., 2010) to extend the n-gram LM-based method to address the first issue. This method turns out to be quite effective and serves as a strong baseline in this paper, though it does not address the second issue. Our

Time Stamp	1	2	3	4	5	6	7	8	9	10	11	12	13	...
Input	i	'd	like	some	tea	and	cake†	that	will	be	a	very	nice	...
Target	0	0	0	0	0	0	1	0	0	0	0	0	0	...
Shift by 1	0	0	0	0	0	0	0	1	0	0	0	0	0	...
Shift by 2	0	0	0	0	0	0	0	0	1	0	0	0	0	...
Shift by 3	0	0	0	0	0	0	0	0	0	1	0	0	0	...
Shift by 4	0	0	0	0	0	0	0	0	0	0	1	0	0	...

Table 2: Illustration of Multi-Shifted Target Signals for Sentence Segmentation. The input is a sequence of words. The target signals are 0's and 1's where 1 means a sentence boundary after the current time stamp. The last four rows shift the target signals by 1 to 4 time units. † Suppose the sentence ends here.

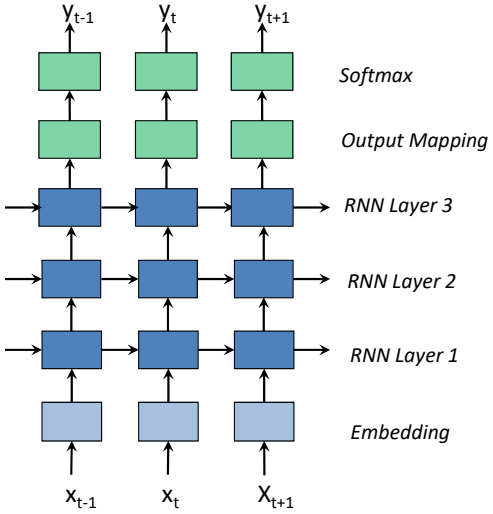

Figure 1: Network Architecture of Multi-shifted RNN Sentence Segmentor

proposed method addresses both issues; thus, it achieves even higher accuracy.

3 Our Method

3.1 Network Architecture

A network architecture inspired by RNN LM is adopted (illustrated by Figure 1). The network works in an online manner by taking one word x_t at each time stamp t as input, and outputting y_t for sentence segmentation.

The output y_t is an $(m+1)$-dimensional vector ($y_t^{\langle 1\rangle}$, $y_t^{\langle 2\rangle}$, ..., $y_t^{\langle m\rangle}$, $y_t^{\langle m+1\rangle}$), where $y_t^{\langle k\rangle}$ ($1 \leqslant k \leqslant m$) presented the confidence of putting a sentence boundary after the k-th word before the time stamp t, while $y_t^{\langle m+1\rangle}$ is imposed by the softmax layer to sum up the probabilities to one. To be precise,

- $y_t^{\langle 1\rangle}$ indicated segmenting after w_{t-1} ;
- $y_t^{\langle 2\rangle}$ indicated segmenting after w_{t-2} ;
- ...
- $y_t^{\langle m\rangle}$ indicated segmenting after w_{t-m} ;
- $y_t^{\langle m+1\rangle}$ equals to $1 - y_t^{\langle 1\rangle} - y_t^{\langle 2\rangle} \ldots - y_t^{\langle m\rangle}$.

In contrast to LM-based methods, this design removes the use of a fixed number of future words. It enables the network to predict a sentence boundary flexibly to time stamps.

3.2 Training

The proposed network is trained on the samples extracted from neighboring sentences, and the training target is to match the output y_t with the oracle segmentation signals. The following two paragraphs explain these two aspects in details.

3.2.1 Extracting Training Samples

Suppose $\mathbb{S} = (S_1, S_2, \ldots)$ is a sequence of sentences which are taken from continuous text. In other words, S_{i+1} is the succeeding sentence of S_i.

Suppose $S_i = (w_1^i, w_2^i, \ldots, w_{n_i}^i)$ where w_t^i ($1 \leqslant t \leqslant n_i$) are the n_i words in the sentence.

One training sample (X_i , n_i) is extracted from (S_i, S_{i+1}) as (illustrated by Figure 2),

$$x_t = \begin{cases} w_t^i & 1 \leqslant t \leqslant n_i \\ w_{t-n_i}^{i+1} & n_i + 1 \leqslant t \leqslant n_i + m \end{cases} \quad (2)$$

where $X_i = (x_1, x_2, \ldots, x_{n_i+m})$ is a sequence of input words.

3.2.2 Training Criterion

The desired value of y_t is formulated as,

$$y_t^{\langle k\rangle} \doteq \begin{cases} 1 & 1 \leqslant t \leqslant n_i, k = m+1 \\ 1 & n_i + 1 \leqslant t \leqslant n_i + m, k = t - n_i \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Therefore, minimizing the cross entropy between y_t and the desired value is taken as the train-

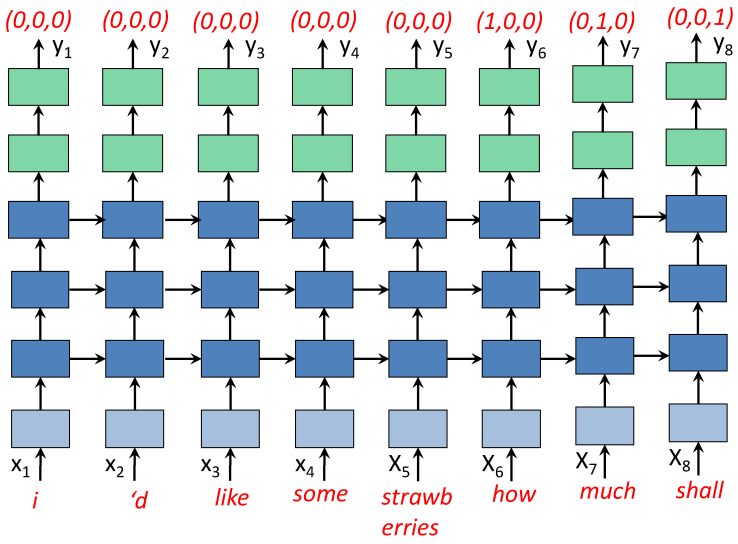

Figure 2: Unrolling a Sample on Multi-shifted RNN Sentence Segmentor. The ASR transcript is "i 'd like some strawberries how much does it cost" where the first sentence ends after "strawberries". Note that y_t's last dimension $y_t^{\langle 4 \rangle} = 1 - y_t^{\langle 1 \rangle} - y_t^{\langle 2 \rangle} - y_t^{\langle 3 \rangle}$ is omitted for simplicity.

ing criterion,

$$\mathbb{E}(\mathbb{S}) = - \underset{(X_i, n_i)}{E} \left(\sum_{t=1}^{n_i} log\, y_t^{\langle m+1 \rangle} + \sum_{t=n_i+1}^{n_i+m} log\, y_t^{\langle t-n_i \rangle} \right) \quad (4)$$

Note that the equation 4 treats each dimension of the output y_t separately. Other sophisticated training criteria that encourage the cooperation among different dimensions have been tried, such as

$$\mathbb{E}(\mathbb{S}) = - \underset{(X_i, n_i)}{E} \left(\sum_{t=1}^{n_i} log\, y_t^{\langle m+1 \rangle} + \max_{t=n_i+1}^{n_i+m} log\, y_t^{\langle t-n_i \rangle} \right) \quad (5)$$

which requires only one of the output to be 1 if the corresponding position is a sentence boundary. However, decrease of segmentation accuracy is observed from this kind of training criteria. We suspect that these criteria introduce dependency among the different dimensions, which reduces the robustness of the method and eventually harms the performance. Therefore, the idea has been avoided.

3.3 Decoding

Decoding on the proposed network is to infer the position of sentence boundaries from a sequence of real-number vectors y_t. The decoding method should be both simple enough to cause no additional latency, and effective enough to achieve competitive accuracy. Therefore, the threshold-latency hybrid decoding strategy proposed by Wang et al. (2016) is extended for the proposed network (illustrated by Figure 3).

The extended decoding strategy uses an m-dimensional threshold vector θ=($\theta^{\langle 1 \rangle}$, $\theta^{\langle 2 \rangle}$, ..., $\theta^{\langle m \rangle}$) to deal with the m-dimensional output y_t. The strategy works as, for each time stamp t,

1. if $y_t^{\langle k \rangle}$ exceeds $\theta^{\langle k \rangle}$ ($k = m, m-1, \ldots, 1$), set $\hat{t} = t - k$ and go to 3;
2. if the buffered input exceed the maximum length, find $argmax_{t',k}(y_{t'}^{\langle k \rangle} - \theta^{\langle k \rangle})$, set $\hat{t} = t' - k$ and go to 3;
3. predict a sentence boundary after $\hat{t}$, and restart the decoding from $\hat{t} + 1$.

The method of tuning θ is described in Section 3.4.

3.4 Tuning

This subsection first defines an empirical score to measure the overall performance of online sentence segmentation, which serves as a target for tuning; then presents an algorithm to search for the optimal threshold vector to maximize the score.

3.4.1 Performance Measurement

An F_1 score calculated on the base of sentences is adopted to measure the accuracy of sentence segmentation. According to our observation, SI

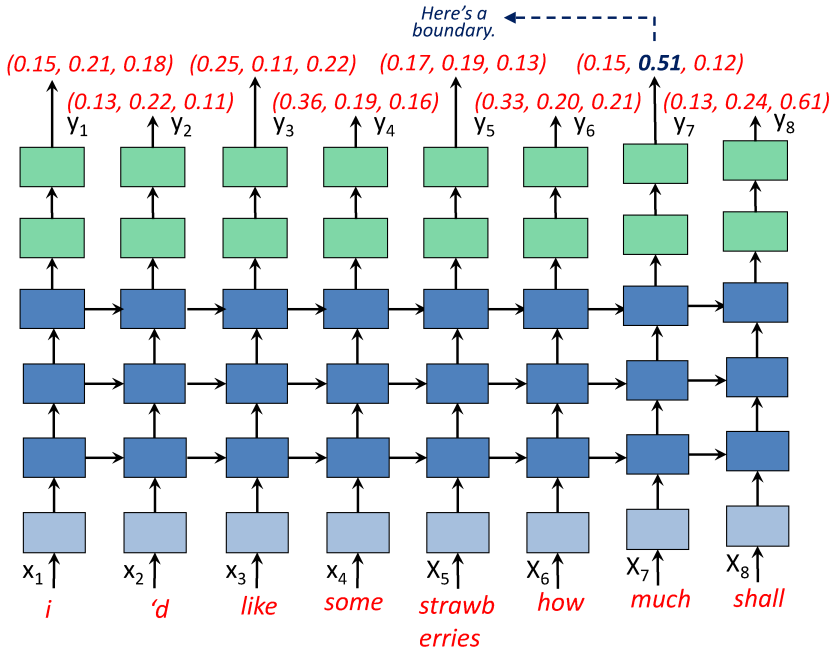

Figure 3: Decoding a Sample on Multi-shifted RNN Sentence Segmentor. Suppose the threshold vector is $(0.40, 0.50, 0.60)$. $y_7^{\langle 2 \rangle}$ is the first value that exceeds the corresponding threshold $\theta^{\langle 2 \rangle}$. This correctly predicts a sentence boundary after the time stamp 5. Note that y_t's last dimension $y_t^{\langle 4 \rangle} = 1 - y_t^{\langle 1 \rangle} - y_t^{\langle 2 \rangle} - y_t^{\langle 3 \rangle}$ is omitted for simplicity.

users often judge the performance based on sentences – how many predicted sentences are correct and how many oracle sentences are recalled. The F_1 score summarizes the precision and recall through calculating the harmonic mean as,

$$F_1 = 2 \frac{precision \cdot recall}{precision + recall}. \quad (6)$$

The latency of sentence segmentation is measured as the average distance per word between the time stamp when a word is input to the segmentor, and the time stamp when this word is output as part of a sentence. Please see Section 4.2 on calculating the latency of the oracle segmentation for an example.

An empirical score is proposed to summarize accuracy and latency, calculated as

$$score = F_1 - \alpha \cdot latency, \quad (7)$$

The trade-off existed because a segmentor could either trade latency for accuracy by waiting for more input words to re-evaluate a prediction, or trade accuracy for latency by predicting boldly without waiting for more evidence brought by input words. The trade-off ratio α is set to 0.01 in this paper according to our observation on SI users and our test on practical sentence segmentors. Note that this ratio can be changed to fit practical applications without the need to revise the proposed method.

3.4.2 Tuning Algorithm

Manually tuning the threshold vector θ for the proposed network is unfeasible as it has m dimensions. Therefore, we propose to use a heuristic greedy search to maximize the score on a develop set, presented in Algorithm 1. The algorithm increases the efficiency by,

- prioritizing the threshold vectors whose parent have achieved high scores;
- pruning the search space by the heuristic that the $\theta^{\langle k \rangle}$ ($k = 1 \ldots m$) should be in descending order.

The intuition for the second point is that a higher threshold should be given to the value derived from fewer future words, because the evidence under that circumstance is weaker.

4 Experiments

4.1 Experimental Setting

The corpora from the shared task in the international workshop on spoken language translation (IWSLT 2015) are used as the experimental corpora (Cettolo et al., 2015)[3]. The task is to translate

[3]https://wit3.fbk.eu/mt.php?release=2015-01

Algorithm 1 Tuning Threshold Vector

Require: θ_0 ▷ a seed threshold vector
Require: $\mathbb{D}$ ▷ a development set
Require: μ ▷ a search step on threshold
Require: ν ▷ a margin on score
$\Theta \leftarrow [\theta_0 : 0]$ ▷ a sorted list of threshold vectors descending on the scores of their parents
$s^* \leftarrow -\infty$ ▷ the best score
$\theta^* \leftarrow \theta_0$ ▷ the best threshold vector
$dict \leftarrow \{\}$ ▷ a dictionary of visited threshold vectors
for θ in the beginning of Θ **do**
    remove θ from Θ
    if θ' not in $dict$ **then**
        $dict \leftarrow dict \cup \{\theta\}$
        $s \leftarrow$ decode $\mathbb{D}$ using θ and evaluate
        if $s \geqslant s^* - \nu$ **then**
            if $s > s^*$ **then**
                $s^* \leftarrow s$
                $\theta^* \leftarrow \theta$
            end if
            for k in 1 to m **do**
                $\theta' \leftarrow$ increase/decrease $\theta^{\langle k \rangle}$ by μ
                if $\theta'^{\langle k-1 \rangle} \geqslant \theta'^{\langle k \rangle} \geqslant \theta'^{\langle k+1 \rangle}$ **and** $0.0 \leqslant \theta'^{\langle k \rangle} \leqslant 1.0$ **then**
                    $\Theta \leftarrow \Theta \cup [\theta' : s]$
                end if
            end for
        end if
    end if
end for
return θ^*

English TED talks into Chinese. Table 3 presents the statistics of the corpora. The news commentary corpora (Tiedemann, 2012)[4] and a subset of the OpenSubtitles corpora (Lison and Tiedemann, 2016)[5] are used to scale up the in-domain training set in order to achieve higher performance.

The corpora are pre-processed using standard procedures for MT. The English text is tokenized using the toolkit released with the Europarl corpus (Koehn, 2005) and converted to lower case. The Chinese text is tokenized into Chinese characters and English words using the tool of *splitUTF8Characters.pl* from the NIST Open Machine Translation 2008 Evaluation [6]

[4]http://opus.nlpl.eu/News-Commentary.php
[5]http://opus.nlpl.eu/OpenSubtitles2016.php
[6]ftp://jaguar.ncsl.nist.gov/mt/resources/

Two operations are applied in order to simulate the transcripts generated by ASR following the setting in (Wang et al., 2016) and (Cho et al., 2017). First, because ASR engines normally do not produce punctuation, punctuation is removed from the text. Second, because ASR engines split output based on long pauses, and each of the output contains multiple sentences; every 10 neighboring sentences in the development and test set are concatenated to form an input for sentence segmentation.

Two baselines are used in the experiments. The first baseline is the n-gram LM-based method proposed by Wang et al. (2016). The toolkit of SRILM (Stolcke, 2002)[7] is used to build n-gram LMs with Kneser-Ney Smoothing and an order of 6.

The second baseline is an extension of the first one by replacing the n-gram LM with an RNN LM. The settings of RNN LM follow the large LSTM setting used by Zaremba et al. (2014) which consists of two layers of 1500 LSTM units (Hochreiter and Schmidhuber, 1997), and a vocabulary size of 10K. A dropout of 0.65 is applied to the non-recurrent connections.

The proposed neural network adopts three layers of 512 LSTM units, and an input vocabulary size of 20K according to our pilot experiments. The output dimension m is 6. A dropout of 0.50 is applied to the non-recurrent connections. Larger networks have been tried in our experiments, but no significant improvement has been observed.

Both the proposed network and RNN LM are trained using SGD with a start learning rate of 1.0. The cross-entropy on the development set is measured after each epoch. When the development cross-entropy stops decreasing, the learning rate starts to decay by 0.5 per epoch. The training terminates when no improvement is made during 3 continuous attempts of decaying learning rates.

The numbers of future words for the two baseline methods are enumerated from 1 to 6, and the decoding thresholds are tuned by a grid search from -1.6 to 1.6 with a step of 0.2. The decoding threshold vector for the proposed method is tuned by Algorithm 1 with θ_0 = (0.9, 0.8, 0.7, 0.6, 0.5, 0.4), μ = 0.1, and ν = 0.04 . The maximum sentence length is set to 40 for all the methods, which covers approximately 95% development and test sentences.

[7]http://www.speech.sri.com/projects/srilm/

Corpus	Sentences	Src. Tokens	Trg. Tokens
IWSLT-Train	209,491	4,270,869	6,050,169
News Commentary	223,153	5,689,117	5,660,789
OpenSubtitle(subset)†	1,000,000	8,682,476	1,047,208
Dev (test2010 test2011)	2,815	55,426	83,317
Test (test2012 test2013)	2,658	52,766	74,822

Table 3: Experimental Corpora. † The subset consists of the first one million sentence pairs.

The software is implemented using C++ and NVIDIA's GPU-accelerated libraries. The experiments are run on a workstation equipped with an Intel Xeon CPU E5-2630 and a GPU Quadro M4000.

4.2 Evaluation after Training on Standard Set

The three methods – two baselines and the proposed method – first learn their models on the source side of the standard training set (Table 3). The n-gram LM-based method learns a 6-ordered n-gram LM whose perplexity on the development set is 148.17. The RNN LM-based method learns an RNN LM with a development perplexity of 62.93. The proposed method learns a network model with a development cross entropy of 0.441. After that, each method tunes its decoding parameters on the development set to maximize the score (the equation 6). In the end, each method decodes the test set using its learned method and tuned parameters. The evaluation of the results is presented in Table 4.

The proposed method outperforms the stronger baseline of the RNN LM-based method by 18.8% on the measurement of score, which is quite large. The improvement is caused by the rise of the measurement of accuracy – F_1 – which is improved by 13.5%, and the stableness of the latency which is only enlarged by 3.4%. This result indicates that the architecture of the proposed network suits the task better than that of RNN LM. In addition, the RNN LM-based method outperforms the n-gram LM-based method by 67.7%. This confirms our expectation that RNN can model a sentence better than n-gram as it can capture long-distance dependencies.

The table also presents the latency of the oracle segmentation which assumes that every sentence is submitted to MT engines as soon as it ends. Suppose the i-th sentence has l_i words, the average latency per word would be $\frac{\sum_i l_i \cdot (l_i - 1)/2}{\sum_i l_i}$. On the experimental test set in, the latency of the oracle segmentation is 8.126, and the latency of the proposed method is 12.386. This approximately means a delay of 4.2 words per sentence, which is acceptable in a real-world environment.

4.3 Evaluation after Adapting Models Trained on Scaled-up Set

Luong et al. (2015) and Cho et al. (2016) show that large-scale out-domain training data and model adaption can effective improve the quality of NMT models. They first train models on the union set of in-domain and out-domain data, and then adapt the models by resuming training on in-domain data only. Inspired by their work, we scale up the standard training set to pursuit better performance for sentence segmentation (see Table 3 for details) .

Through scaling up training set and model adaptation, the development perplexity of the RNN LM is reduced by 8.06% (from 62.93 to 57.86), and the development cross entropy of the model learned by the proposed method decreases by 0.082 (from 0.441 to 0.359).

The n-gram LM is adapted by linear interpretation. The mixture weight is tuned to minimize the development perplexity, whose value turns out to be 0.7. The development perplexity of the n-gram LM is reduced by 8.25% (from 148.16 to 135.93)

Each method again tunes its decoding parameters, and then decodes the test set as described in Section 4.2. Table 5 summarizes the results, and compares them with the previous ones on the standard training set. The performance of all three methods is found to be improved, while the proposed method achieves the largest improvement.

The detailed comparison between the two results (the last row in Table 5) shows that all the individual performance measurements have been improved. Moreover, the optimal thresholds generally get lower. This clearly indicates that the quality of the trained model has been improved, which is quite impressive. The same effects also

Methods	Parameters		Performance				
	n_f	thresh.	Precision	Recall	F_1	Latency	Score
Oracle			1.000	1.000	1.000	8.126	0.9187
n-gram LM	1	-0.6	0.1402	0.2432	0.1779	8.3410	0.0945
	2	-0.6	**0.1862**	**0.3087**	**0.2323**	**9.6480**	**0.1358**
	3	-0.6	0.1928	0.3005	0.2349	11.2520	0.1224
	4	-0.6	0.1944	0.2993	0.2357	12.2930	0.1128
	5	-0.6	0.1935	0.2959	0.2340	13.2410	0.1016
	6	-0.6	0.1927	0.2937	0.2327	14.1570	0.0912
RNN LM	1	-0.8	0.2686	0.3213	0.2926	10.3503	0.1891
	2	-0.6	**0.3289**	**0.3683**	**0.3475**	**11.9733**	**0.2277**
	3	-0.8	0.3255	0.3743	0.3482	12.7531	0.2207
	4	-0.8	0.3372	0.3845	0.3593	13.8317	0.2210
	5	-0.8	0.3342	0.3822	0.3566	14.8643	0.2080
	6	-0.8	0.3256	0.3740	0.3481	15.7449	0.1907
Proposed	1 – 6	(...)‡	**0.3583**	**0.4387**	**0.3945**	**12.3863**	**0.2706**
Improve†			8.9%	19.1%	13.5%	-3.4%	18.8%

Table 4: Performance after Training on Standard Set. † Improvement versus the stronger baseline of RNN LM. ‡ The optimal threshold vector is (1.0, 0.8, 0.8, 0.5, 0.5, 0.3).

Methods	Parameters		Performance				
	n_f	thresh.	Precision	Recall	F_1	Latency	Score
n-gram LM	1	-0.6	0.1349	0.2541	0.1762	7.6290	0.1000
	2	-0.4	**0.2054**	**0.3163**	**0.2490**	**10.3310**	**0.1457** (+0.0099)†
	3	-0.4	0.2125	0.3148	0.2537	11.6760	0.1369
	4	-0.4	0.2129	0.3129	0.2534	12.7040	0.1264
	5	-0.4	0.2125	0.3099	0.2521	13.6660	0.1154
	6	-0.4	0.2120	0.3080	0.2512	14.5780	0.1054
RNN LM	1	-1.0	0.2574	0.3269	0.2880	9.7292	0.1907
	2	-1.0	**0.3205**	**0.3894**	**0.3516**	**11.2249**	**0.2394** (+0.0117)†
	3	-0.8	0.3383	0.3856	0.3604	12.8106	0.2323
	4	-1.0	0.3315	0.3894	0.3581	13.6455	0.2217
	5	-1.0	0.3302	0.3871	0.3564	14.7268	0.2092
	6	-1.0	0.3295	0.3845	0.3549	15.7642	0.1972
Proposed	1–6	(...)‡	**0.3959**	**0.4605**	**0.4257**	**12.1118**	**0.3046** (+0.0340) †
Imp. vs. RNN LM			23.5%	18.3%	21.1%	-7.9%	27.2%
Imp. vs. standard†			10.5%	5.0 %	7.9%	2.2%	12.6%

Table 5: Segmentation Performance after Adapting the Models Trained on Scaled-up Set. † Compared to the best score of each method on the standard training set. ‡ The optimal threshold vector is (0.9, 0.8, 0.5, 0.5, 0.5, 0.4)

happen on the RNN LM-based method. Therefore, adapting neural network models through resuming training is a very effective technique.

4.4 Evaluation of End-to-end Translation Quality

The best segmentations of each method, which are listed in Table 5 in bold font, are post-processed to recover case and punctuation, and then piped into an English-to-Chinese NMT engine. The post-processing is conducted by a monotone phrase-based statistical MT system, which is trained to translate lower-cased unpunctuated sentences to cased punctuated sentences. Moses toolkit (Koehn et al., 2007) is used. The NMT engine is an implementation of attention-based encoder-decoder proposed by Bahdanau et al. (2014) and Luong et al. (2015), and the model is trained and tuned on an

Methods	BLEU	Loss[†]
Oracle	19.73	
n-gram LM	18.98	0.75
RNN LM	19.38	0.35
Proposed	**19.48**	**0.25** (-28.6%)[‡]

Table 6: Evaluation of End-to-end Translation Quality. [†] Compared to the BLEU of the oracle sentence segmentation. [‡] Compared to the stronger baseline of RNN LM.

in-house parallel corpus of approximately 21 million sentence pairs from various domains.

The translations are evaluated following the official guidelines of IWSLT 2015. The translations are aligned to reference sentences through edit distance (Matusov et al., 2005). BLEU is calculated on cased tokens including Chinese characters and English words. Table 6 presents the results.

The results show that the proposed method achieves the highest BLEU, which is lower than that of the oracle segmentation only by 0.25. The improvement compared to the stronger baseline of the RNN LM-based method is 0.10 BLEU point, or 28.6% calculated by 0.10 / 0.35.

5 Related Works

Segmenting the unpunctuated transcripts generated by ASR have attracted attentions from many researchers. A large variety of methods have been proposed.

Conditional random fields (CRFs) are used to approach the problem. Hassan et al. (2014) did a thorough treatment of this problem in 2014. However, CRFs have been outperformed by neural networks recently.

MT systems are used to approach the problem by Cho et al.(2015), Ha et al. (2015), Kzai et al. (2015), Cho et al. (2017), Pham et al. (2016), Klejch et al. (2016; 2017) and Przybysz et al. (2016). This approach builds MT systems to translate unpunctuated text into punctuated text which contains full stop marks as sentence boundaries. The drawback of this approach is that MT systems normally expect complete sequences as input, which prevents them from working in an online manner. Cho et al. (2015; 2017) address the issue using sliding windows. A fixed-length subsequence of words are extracted from the stream of words, and then feed into MT systems. The shortage of this method is that the dependencies outside the sliding windows are ignored, which will decrease the accuracy. In contrast, our RNN-based method performs incremental decoding from the beginning of sentences, so it can capture all the dependencies within a whole sentence.

Pauses, or precisely the duration of silence between two spoken words, which can be captured by ASR engines, are used to predict sentence boundaries by Fügen et al. (2007) and Bangalore et al. (2012). However, studies on human interpreters reveal that segmenting merely by pauses is insufficient, as human speakers might not pause between sentences. The mean proportion of silence-based chunking by interpreters is 6.6% when the source is English, 10% when it is French, and 17.1% when it is German (Venuti, 2012). Therefore, this paper focuses on using linguistic information. Nevertheless, pauses can be directly integrated into our proposed method to boost performance.

There are several segmentation methods that target at splitting an input sentence into smaller pieces for simultaneous interpretation, such as Yarmohammadi et al. (2013), Oda et al. (2014), and Fujita et al. (2013). However, these methods often assume that ASR transcripts have already been segmented into sentences, which is the task addressed by this paper. Therefore, our method is orthogonal to these methods, and it is possible to pipeline our proposed method with them.

6 Conclusion

In this paper, a multi-shifted RNN is proposed to solve the problem of segmenting the unpunctuated ASR transcripts for SI. The multi-shifted RNN addresses the trade-off between accuracy and latency which are the two central elements of the problem. The experiments show that the proposed method greatly outperforms an n-gram LM-based method and an RNN LM-based method on accuracy, latency and end-to-end BLEU, under both a standard training set and a scaled-up training set.

Acknowledgement

This work was partially conducted under the program "Promotion of Global Communications Plan: Research, Development, and Social Demonstration of Multilingual Speech Translation Technology" of the Ministry of Internal Affairs and Communications (MIC), Japan.

References

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Proceedings of the 3rd International Conference on Learning Representations.*, pages 1–15.

Bangalore, Srinivas, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 437–445. Association for Computational Linguistics.

Cettolo, Mauro, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2015. The IWSLT 2015 evaluation campaign. In *IWSLT 2015, International Workshop on Spoken Language Translation*.

Cho, Eunah, Jan Niehues, Kevin Kilgour, and Alex Waibel. 2015. Punctuation insertion for real-time spoken language translation. In *Proceedings of the Eleventh International Workshop on Spoken Language Translation*.

Cho, Eunah, Jan Niehues, Thanh-Le Ha, Matthias Sperber, Mohammed Mediani, and Alex Waibel. 2016. Adaptation and combination of NMT systems: The KIT translation systems for IWSLT 2016. In *Proceedings of the 13th International Workshop on Spoken Language Translation (IWSLT 2016)*.

Cho, Eunah, Jan Niehues, and Alex Waibel. 2017. Nmt-based segmentation and punctuation insertion for real-time spoken language translation. *Proc. Interspeech 2017*, pages 2645–2649.

Fügen, Christian, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21(4):209–252.

Fujita, Tomoki, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Simple, lexicalized choice of translation timing for simultaneous speech translation. In *INTERSPEECH*, pages 3487–3491.

Ha, Thanh-Le, Jan Niehues, Eunah Cho, Mohammed Mediani, and Alex Waibel. 2015. The KIT translation systems for IWSLT 2015. In *Proceedings of the twelveth International Workshop on Spoken Language Translation (IWSLT), Da Nang, Veitnam*, pages 62–69.

Hassan, Hany, Lee Schwartz, Dilek Hakkani-Tür, and Gokhan Tur. 2014. Segmentation and disfluency removal for conversational speech translation. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Hochreiter, Sepp and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Klejch, Ondřej, Peter Bell, and Steve Renals. 2016. Punctuated transcription of multi-genre broadcasts using acoustic and lexical approaches. In *Spoken Language Technology Workshop (SLT), 2016 IEEE*, pages 433–440. IEEE.

Klejch, Ondřej, Peter Bell, and Steve Renals. 2017. Sequence-to-sequence models for punctuated transcription combining lexical and acoustic features. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 5700–5704. IEEE.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*, volume 5, pages 79–86.

Kzai, Michaeel, Brian Thompson, Elizabeth Salesky, Timothy Anderson, Grant Erdmann, Eric Hansen, Brian Ore, Katherine Young, Jeremy Gwinnup, Michael Hutt, and Christina May. 2015. The MITLL-AFRL IWSLT 2015 systems. In *Proceedings of the twelveth International Workshop on Spoken Language Translation (IWSLT), Da Nang, Veitnam*, pages 23–30.

Lison, Pierre and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles.

Luong, Minh-Thang and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 76–79.

Luong, Thang, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Matusov, Evgeny, Gregor Leusch, Oliver Bender, Hermann Ney, et al. 2005. Evaluating machine translation output with automatic sentence segmentation. In *IWSLT*, pages 138–144. Citeseer.

Mikolov, Tomáš, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association.*

Oda, Yusuke, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *ACL (2)*, pages 551–556.

Pham, Ngoc-Quan, Matthias Sperber, Elizabeth Salesky, Thanh-Le Ha, Jan Niehues, and Alex Waibel. 2016. KIT's Multilingual Neural Machine Translation systems for IWSLT 2017. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*, pages 42–47.

Przybysz, Pawel, Marcin Chochowski, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. The Samsung and University of Edinburghs submission to IWSLT17. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*, pages 23–28.

Stolcke, Andreas and Elizabeth Shriberg. 1996. Automatic linguistic segmentation of conversational speech. In *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, volume 2, pages 1005–1008. IEEE.

Stolcke, Andreas, Elizabeth Shriberg, Rebecca A Bates, Mari Ostendorf, Dilek Hakkani, Madelaine Plauche, Gökhan Tür, and Yu Lu. 1998. Automatic detection of sentence boundaries and disfluencies based on recognized words. In *Proceedings of 5th International Conference on Spoken Language Processing*, pages 2247–2250.

Stolcke, Andreas. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*.

Tiedemann, Jörg. 2012. Parallel Data, Tools and Interfaces in OPUS. In *LREC*, volume 2012, pages 2214–2218.

Venuti, Lawrence. 2012. *The translation studies reader*. Routledge.

Wang, Xiaolin, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. 2016. An efficient and effective online sentence segmenter for simultaneous interpretation. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 139–148, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Yarmohammadi, Mahsa, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *IJCNLP*, pages 1032–1036.

Zaremba, Wojciech, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.