Dynamic Sentence Boundary Detection for Simultaneous Translation

Ruiqing Zhang
Baidu, Inc., / Beijing, China
zhangruiqing01@baidu.com

Chuanqiang Zhang
Baidu, Inc., / Beijing, China
zhangchuanqiang@baidu.com

Abstract

Simultaneous Translation is a great challenge in which translation starts before the source sentence finished. Most studies take transcription as input and focus on balancing translation quality and latency for each sentence. However, most ASR systems can not provide accurate sentence boundaries in realtime. Thus it is a key problem to segment sentences for the word streaming before translation. In this paper, we propose a novel method for sentence boundary detection that takes it as a multi-class classification task under the end-to-end pre-training framework. Experiments show significant improvements both in terms of translation quality and latency.

1 Introduction

Simultaneous Translation aims to translate the speech of a source language into a target language as quickly as possible without interrupting the speaker. Typically, a simultaneous translation system is comprised of an auto-speech-recognition (ASR) model and a machine translation (MT) model. The ASR model transforms the audio signal into the text of source language and the MT model translates the source text into the target language.

Recent studies on simultaneous translation (Cho and Esipova, 2016; Ma et al., 2019; Arivazhagan et al., 2019) focus on the trade-off between translation quality and latency. They explore a policy that determines when to begin translating with the input of a stream of transcription. However, there is a gap between transcription and ASR that some ASR model doesn't provide punctuations or cannot provide accurate punctuation in realtime, while the transcription is always well-formed. See Figure 1 for illustration. Without sentence boundaries, the state-of-the-art *wait-k* model takes insufficient text as input and produces an incorrect translation.

Therefore, sentence boundary detection (or sentence segmentation) [1] plays an important role to narrow the gap between the ASR and transcription. A good segmentation will not only improve translation quality but also reduce latency.

Studies of sentence segmentation falls into one of the following two bins:

- The strategy performs segmentation from a speech perspective. Fügen et al. (2007) and Bangalore et al. (2012) used prosodic pauses in speech recognition as segmentation boundaries. This method is effective in dialogue scenarios, with clear silence during the conversation. However, it does not work well in long speech audio, such as lecture scenarios. According to Venuti (2012), silence-based chunking accounts for only 6.6%, 10%, and 17.1% in English, French, and German, respectively. Indicating that in most cases, it cannot effectively detect boundaries for streaming words.

- The strategy takes segmentation as a standard text processing problem. The studies considered the problem as classification or sequence labeling, based on SVM, (Sridhar et al., 2013) conditional random filed (CRFs) (Lu and Ng, 2010; Wang et al., 2012; Ueffing et al., 2013). Other researches utilized language model, either based on N-gram (Wang et al., 2016) or recurrent neural network (RNN)(Tilk and Alumäe, 2015).

In this paper, we use classification to solve the problem of sentence segmentation from the perspective of text. Instead of predicting a sentence boundary for a certain position, we propose a multi-position boundary prediction approach. Specifically, for a source text $\mathbf{x} = \{x_1, ..., x_T\}$, we calculate the probability of predicting sentence boundary

[1] We use both terms interchangeably in this paper.

Proceedings of the 1st Workshop on Automatic Simultaneous Translation, pages 1–9
July 10, 2020. ©2020 Association for Computational Linguistics

src	One	of	two	things	is	going	to	happen	.	Either	**it**	**'s**	**going**	**to**	...
Reference	Eines von zwei Dingen wird passieren.									Entweder **wird** ...					
wait3				Eines	von	zwei	Dingen	wird	passieren.				Entweder	**wird**	...
src without boundary	One	of	two	things	is	going	to	happen	either	**it**	**'s**	**going**	**to**	...	
wait3				Eines	von	zwei	Dingen	wird	passieren	entweder	**es**	**ist**	**geht**	**dass**	...

Figure 1: An English-to-German example that translates from a streaming source with and without sentence boundaries. We take the wait-K model (Ma et al., 2019) for illustration, K=3 here. The *wait3* model first performs three READ (wait) action at the beginning of each sentence (as shown in blue), and then alternating one READ with one WRITE action in the following steps. Given the input source without sentence boundaries (in the $4^{th} line$), the *wait3* model (in the $5^{th} line$) doesn't take the three READ action at the beginning of following sentences. Therefore, the English phrase "it's going to", which should have been translated as "wird", produced a meaningless translation "es ist geht dass" with limited context during *wait3* model inference.

after x_t , $t = T, T-1, ..., T-M$. Thus the latency of translation can be controlled within $L+M$ words, where L is the length of the sentence. Inspired by the recent pre-training techniques (Devlin et al., 2019; Sun et al., 2019) that successfully used in many NLP tasks, we used a pre-trained model for initialization and fine-tune the model on the source side of the sentence. Overall, the contributions are as follows:

- We propose a novel sentence segmentation method based on pre-trained language representations, which have been successfully used in various NLP tasks.
- Our method dynamically predicts the boundary at multiple locations, rather than a specific location, achieving high accuracy with low latency.

2 Background

Recent studies show that the pre-training and fine-tuning framework achieves significant improvements in various NLP tasks. Generally, a model is first pre-trained on large unlabeled data. After that, on the fine-tuning step, the model is initialized by the parameters obtained by the pre-training step and fine-tuned using labeled data for specific tasks.

Devlin et al. (2019) proposed a generalized framework BERT, to learn language representations based on a deep Transformer (Vaswani et al., 2017) encoder. Rather than traditionally train a language model from-left-to-right or from-right-to-left, they proposed a masked language model (MLM) that randomly replace some tokens in a sequence by a placeholder (mask) and trained the model to predict the original tokens. They also pre-train the model for the next sentence prediction (NSP) task that is to predict whether a sentence is the subsequent sentence of the first sentence. Sun et al. (2019) proposed a pre-training framework ERNIE, by integrating more knowledge. Rather than masking single tokens, they proposed to mask a group of words on different levels, such as entities, phrases, etc. The model achieves state-of-the-art performances on many NLP tasks.

In this paper, we train our model under the ERNIE framework.

3 Our Method

Given a streaming input $\mathbf{x} = \{x_1, ..., x_t, ..., x_T\}$, the task of sentence segmentation is to determine whether $x_t \in \mathbf{x}$ is the end of a sentence. Thus the task can be considered as a classification problem, that is $p(y_t|\mathbf{x}, \theta)$, where $y_t \in \{0, 1\}$. However, in simultaneous translation scenario, the latency is unacceptable if we take the full source text as contextual information. Thus we should limit the context size and make a decision dynamically.

As the input is a word streaming, the sentence boundary detection problem can be transformed as, whether there exists a sentence boundary until the current word x_t. Thus we can use the word streaming as a context to make a prediction. We propose a multi-class classification model to predict the probability of a few words before x_t as sentence boundaries (Section 3.1). We use the ERNIE framework to first pre-train a language representation and then fine-tune it to sentence boundary detection (Section 3.2). We also propose a dynamic voted inference strategy (Section 3.3).

3.1 The Model

For a streaming input $\mathbf{x} = \{x_1, ..., x_t\}$, our goal is to detect whether there is a sentence boundary

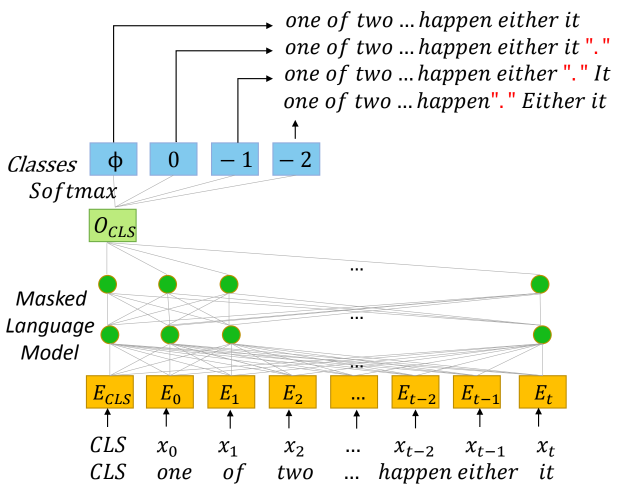


Figure 2: Illustration of the dynamic classification model. $M = 2$ means there are 4 classes. We use ERNIE to train a classifier. Class ϕ means that there is no sentence boundary in the stream till now. Class $-m$ $m = 0, 1, 2$ means that x_{t-m} is the end of a sentence and we then put a period after it.

till the current word x_t from last sentence boundary. Rather than a binary classification that detects whether x_t is a sentence boundary, we propose a multi-class method. The classes are as follows:

$$y = \begin{cases} \phi, & \text{no sentence boundary detected} \\ 0, & x_t \quad \text{is the end of a sentence} \\ -1, & x_{t-1} \quad \text{is the end of a sentence} \\ \dots & \\ -M, & x_{t-M} \quad \text{is the end of a sentence} \end{cases}$$

where M is the maximum offset size to the current state. Thus, we have $M + 2$ classes.

See Figure 2 for illustration. We set $M = 2$, indicating that the model predicts 4 classes for the input stream. If the output class is ϕ, meaning that the model does not detect any sentence boundary. Thus the model will continue receiving new words. If the output class is 0, indicating that the current word x_t is the end of a sentence and we put a period after the word. Similarly, class $-m$ denotes to add a sentence boundary after x_{t-m}. While a sentence boundary is detected, the sentence will be extracted from the stream and sent to the MT system as an input for translation. The sentence detection then continues from x_{t-m+1}.

Each time our system receives a new word x_t, the classifier predicts probabilities for the last $M+1$ words as sentence boundaries. If the output class is ϕ, the classifier receives a new word x_{t+1}, and recompute the probabilities for x_{t+1}, x_t, x_{t-1}, ..., x_{t-M+1}. Generally, more contextual information will help the classifier improve the precision (Section 4.5).

3.2 Training Objective

Our training data is extracted from paragraphs. Question marks, exclamation marks, and semicolons are mapped to periods and all other punctuation symbols are removed from the corpora. Then for every two adjacent sentences in a paragraph, we concatenate them to form a long sequence, **x**. We record the position of the period as r and then remove the period from the sequence.

For $x = (x_1, x_2, ..., x_N)$ with N words, we generate $r + M$ samples for $t = 1, 2, ..., (r + M)$, in the form of $< (x_1, ..., x_t), y_t >$, where y_t is the label that:

$$y_t = \left\{ \begin{array}{lll} \phi, & \textit{if} & t < r \\ -(t-r), & \textit{if} & t \in [r, r+M] \end{array} \right\} \qquad (1)$$

Note that if the length of the second sentence is less than M, we concatenate subsequent sentences until $r + M$ samples are collected. Then we define the loss function as follows:

$$\begin{aligned} \mathcal{J}(\theta) = \sum_{(\mathbf{x},r)\in D} log(\sum_{t=1}^{r-1} p(y_t = \phi | x_{\leq t}; \theta) \\ + \sum_{t=r}^{r+M} p(y_t = -(t-r)) | x_{\leq t}; \theta)) \end{aligned} \qquad (2)$$

where D is the dataset that contains pairs of concatenated sentences **x** and its corresponding position of the removed periods r. M is a hyperparameter denotes the number of waiting words.

Note that our method differs from previous work in the manner of classification. Sridhar et al. (2013) predicts whether a word x_t labeled as the end of a sentence or not by a binary classification:

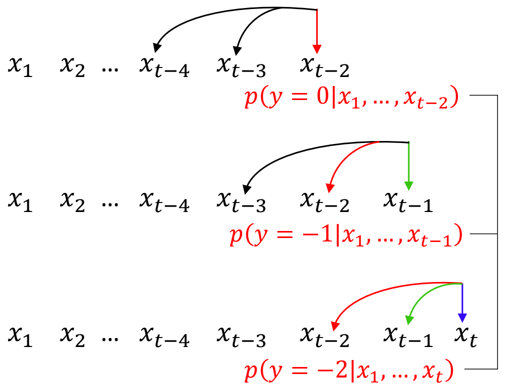
$$p(y_t = 0 | x_{t-2}^{t+2}) + p(y_t = 1 | x_{t-2}^{t+2}) = 1 \qquad (3)$$

where $y_t = 0$ means x_t is not the end of a sentence and $y_t = 1$ means x_t is the end. x_{t-2}^{t+2} denotes 5 words $x_{t-2}, x_{t-1}, ..., x_{t+2}$.

Some other language-model based work (Wang et al., 2016) calculates probabilities over all words in the vocabulary including the period:

$$\sum_{w \in V \cup \text{“.”}} p(y_t = w | x_{\leq t}) = 1 \qquad (4)$$

and decides whether x_t is a sentence boundary by comparing the probability of $y_t =$“.” and $y_t = x_{t+1}$.



x_1 x_2 ... x_{t-4} x_{t-3} x_{t-2}
$p(y=0|x_1,...,x_{t-2})$
x_1 x_2 ... x_{t-4} x_{t-3} x_{t-2} x_{t-1}
$p(y=-1|x_1,...,x_{t-1})$
x_1 x_2 ... x_{t-4} x_{t-3} x_{t-2} x_{t-1} x_t
$p(y=-2|x_1,...,x_t)$

Figure 3: Our voting algorithm for online prediction with M equals to 2. Input the stream text till x_t, the overall probability of add a sentence boundary after x_{t-2} is averaged by the $M+1$ probabilities in red, while for x_{t-1} (in green) and x_t (in blue), the number of deterministic probability is less than $M+1$.

The performance of these methods is limited by incomplete semantics, without considering global boundary detection. In our methods, we leverage more future words and restrict classes globally:

$$p(y_t = \phi|x_{\leq t}) + \sum_{m=0}^{M} p(y_t = -m|x_{\leq t}) = 1 \quad (5)$$

The restriction is motivated that in a lecture scenario, where a sentence could not be very short that contains only 1 or 2 words. Thus, the probability distribution prohibits that adjacent words to be the end of sentences at the same time.

3.3 Dynamic Inference

At inference time, we predict sentence boundaries sequentially with a dynamic voting strategy. Each time a new word x_t is received, we predict the probability of $M+1$ classes as shown in the bottom of Figure 3, then calculate if the probability of previous $M+1$ positions $(x_{t-M}, x_{t-M+1}, x_t)$ is larger then a threshold θ_{Th}. If yes, we add a sentence boundary at the corresponding position. Otherwise, we continue to receive new words.

Note that the probability is adopted as the voted probability. While the probability of adding a sentence boundary after x_{t-M} has $M+1$ probabilities to calculate the average, the number of probabilities to determine whether it is a sentence boundary at subsequent positions is less than $M+1$. Here we use the voted average of existing probabilities. Specifically, to judge whether $x_{t'}$ is a sentence boundary, it needs $t-t'+1$ probabilities:

$$\frac{1}{t-t'+1}\sum_{m=0}^{t-t'} p(y=-m|x_1,...,x_{t+m}) \quad (6)$$

where $t' \in [t-M, t]$.

If more than one sentence boundary probabilities for $x_{t-M}, ..., x_t$ exceeds the threshold θ_{Th} at the same time, we choose the front-most position as a sentence boundary. This is consistent with our training process, that is, if there is a sample of two or more sentence boundaries, we ignore the following and label the class y_t according to the first boundary. This is because we generate samples with each period in the original paragraph as depicted in Section 3.2. From another point of view, the strategy can also compensate for some incorrect suppression of adjacent boundaries, thereby improving online prediction accuracy.

	Dataset	Sentences	Tokens/s
Train	WMT 14	4.4M	23.22
	IWSLT 14	0.19M	20.26
Test	IWSLT 2010-2014	7040	19.03

Table 1: Experimental Corpora without punctuation. Token/s denotes the number of tokens per sentence in English.

4 Experiment

Experiments are conducted on English-German (En-De) simultaneous translation. We evaluate 1) the F-score[2] of sentence boundary detection and 2) case-sensitive tokenized 4-gram BLEU (Papineni et al., 2002) as the final translation effect of the segmented sentences. To reduce the impact of the ASR system, we use the transcription without punctuation in both training and evaluation.

The datasets used in our experiments are listed in Table 1. We use two parallel corpus from machine translation task: WMT 14[3] and IWSLT 14 [4]. WMT 14 is a text translation corpus including 4.4M sentences, mainly on news and web sources. And IWSLT 14 is a speech translation corpus of TED lectures with transcribed text and corresponding translation. Here we only use the text part in it, containing 0.19M sentences in the training set.

[2]harmonic average of the precision and recall
[3]http://www.statmt.org/wmt14/translation-task.html
[4]https://wit3.fbk.eu/

Method	Hyperparameter	F-score	BLEU	avgCW	maxCW
Oracle	NA	1.0	22.76	NA	NA
N-gram	N=5, $\theta_{Th} = e^{0.0}$	0.46	17.83	**6.64**	56
N-gram	N=5, $\theta_{Th} = e^{2.0}$	0.48	19.20	13.43	161
T-LSTM	d=256	0.55	20.46	10.14	53
dynamic-force	$\theta_l = 40, \theta_{Th} = 0.5$	**0.74**	**22.01**	14.43	**40**
dynamic-base	$\theta_{Th} = 0.5$	**0.74**	21.93	14.58	50

Table 2: Segmentation Performance trained on IWSLT2014. All methods are conducted with future words M equals to 1.

We train the machine translation model on WMT 14 with the base version of the Transformer model (Vaswani et al., 2017), achieving a BLEU score of 27.2 on newstest2014. And our sentence boundary detection model is trained on the source transcription of IWSLT 14 unless otherwise specified (Section 4.3). To evaluate the system performance, we merge the IWSLT test set of 4 years (2010-2014) to construct a big test set of 7040 sentences. The overall statistics of our dataset is shown in Table 1.

We evaluate our model and two existing methods listed below:

- *dynamic-base* is our proposed method that detect sentence boundaries dynamically using a multi-class classification.

- *dynamic-force* adds a constraint on *dynamic-base*. In order to keep in line with (Wang et al., 2016), we add a constraint that sentence should be force segmented if longer than θ_l.

- *N-gram* is the method using an N-gram language model to compare the probability of adding vs. not adding a boundary at x_t after receiving $x_{t-N+1}, ..., x_t$. We implement according to (Wang et al., 2016).

- *T-LSTM* uses a RNN-based classification model with two classes. We implement a unidirectional RNN and perform training according to (Tilk and Alumäe, 2015)[5].

Our classifier in *dynamic-base* and *dynamic-force* is trained under ERNIE base framework. We use the released [6] parameters obtained at pre-training step as initialization. In the fine-tuning stage, we use a learning rate of $2e^{-5}$.

[5]we only keep the two classes of *period* and ϕ in this work

[6]https://github.com/PaddlePaddle/ERNIE

4.1 Overall Results

Table 2 reports the results of source sentence segmentation on En-De translation, where the latency is measured by Consecutive Wait (CW) (Gu et al., 2017), the number of words between two translate actions. To eliminate the impact of the different policies in simultaneous translation, we only execute translation at the end of each sentence. Therefore, the CW here denotes the sentence length L plus the number of future words M. We calculate its average and maximum value as "avgCW" and "maxCW", respectively. Better performance expect high F-score, BLEU, and low latency (CW). The translation effect obtained by using the groundtruth period as the sentence segmentation is shown in the first line of *Oracle*.

The *N-gram* method calculate the probability of add (p_{add}) and not add (p_{not}) period at each position, and decide whether to chunk by comparing whether p_{add}/p_{not} exceeds θ_{Th}. The N-gram method without threshold tuning (with $\theta_{Th} = e^{0.0}$) divides sentences into small pieces, achieving the lowest average latency of 6.64. However, the F-score of segmentation is very low because of the incomplete essence of the n-gram feature. Notable, the precision and recall differs much ($precision = 0.33, recall = 0.78$) in this setup. Therefore, we need to choose a better threshold by grid search (Wang et al., 2016). With θ_{Th} equals to $e^{2.0}$, the F-score of N-gram method increased a little bit ($0.46 \rightarrow 0.48$), with a more balanced precision and recall ($precision = 0.51, recall = 0.48$). However, the max latency runs out of control, resulting in a maximum of 161 words in a sentence. We also tried to shorten the latency of the N-gram method by force segmentation (Wang et al., 2016), but the result was very poor ($precision = 0.33, recall = 0.40$).

The *T-LSTM* method with the hidden size of 256 performs better than *N-gram*, but the F-score

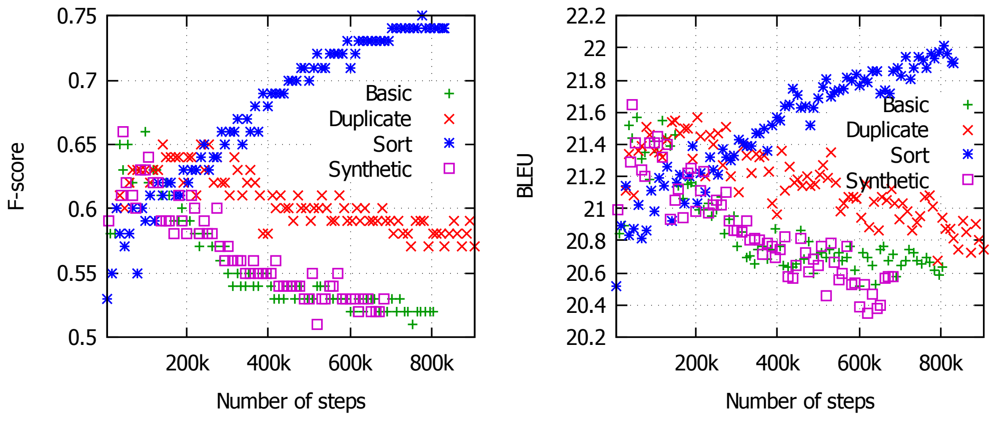


Figure 4: Performance evaluated on IWSLT14 testset for different training sample building strategies.

and BLEU is still limited. On the contrary, our *dynamic*-based approaches with $M = 1$ achieve the best F-score at 0.74 and the final translation is very close to the result of *Oracle*. In particular, the precision and recall reached about 0.72 and 0.77 in both *dynamic-force* and *dynamic-base*, respectively. Accurate sentence segmentation brings better performance in translation, bringing an improvement of 1.55 over *T-LSTM*. Moreover, our approach is not inferior in terms of latency. Both average latency and max latency is controlled at a relatively low level.

It is interesting to note that, *dynamic-force* performs better than *dynamic-base*, in terms of latency and BLEU. This suggests the effectiveness of the force segmentation strategy, that is, select the chunking location with a sentence length limitation will not affect the accuracy of segmentation, and would enhance the translation effect.

4.2 Magic in Data Processing

According to Section 3.2, the order between sentences of original corpora would affect the generation of training samples. In this section, we investigate the effect of various data reordering strategies.

A basic method is to use the original sentence order of speech corpora, denote as ***Basic***. However, the samples generated is limited, which makes the model easy to over-fit. To overcome this problem, we adopt two methods to expand data scale: 1) ***Duplicate*** the original data multiple times or 2) Add ***Synthetic*** adjacent sentences, through randomly selecting two sentences from the corpora. These two methods greatly expand the total amount of data, but the gain to the model is uncertain. As an alternative, we explore a ***Sort*** method, to sort sentences according to alphabetic order.

The performance of the four training data organization methods is shown in Figure 4, all built on IWSLT2014 and conducted under the setup of $M = 1$ and $\theta_l = 40$. It is clear that *Basic*, *Duplicate* and *Synthetic* are all involved in the problem of over-fitting. They quickly achieved their best results and then gradually declined. Surprisingly, the *Sort* approach is prominent in both segmentation accuracy and translation performance. This may be due to the following reasons: 1) Sentence classification is not a difficult task, especially when $M = 1$ for 3-class classification ($y \in [\phi, 0, -1]$), making the task easy to over-fit. 2) Compared with *Basic*, *Duplicate* is more abundant in the sample combination in batch training, but there is no essential difference between the two methods. 3) *Synthetic* hardly profits our model, because the synthesized data may be very simple due to random selection. 4) *Sort* may simulate difficult cases in real scenes and train them pertinently, bringing it a poor performance at start but not prone to over-fit. There are many samples with identical head and tail words in the sorted data, such as: "*and it gives me a lot of hope* || *and ...*" and "*that means there's literally thousands of new ideas* || *that ...* ". Even human beings find it difficult to determine whether the words before || is sentence boundaries of these samples. In *Basic*, *Duplicate* and *Synthetic* methods, such samples are usually submerged in a large quantity of simple samples. However, the data organization mode of *Sort* greatly strengthens the model's ability to learn these difficult samples.

There is no need to worry that the *Sort* method cannot cover simple samples. Because we sort by rows in source file, and some of the rows contain multiple sentences (an average of 1.01 sentences

Method	F-score	BLEU	avgCW	maxCW
N-gram	0.48	19.58	15.60	156
T-LSTM	0.56	20.77	15.65	51
dyn-force	**0.68**	**21.48**	**15.53**	**40**
dyn-base	**0.68**	21.40	16.08	46

Table 3: Segmentation Performance trained on WMT14. All methods are conducted with future words M equals to 1. *N-gram* uses grid-search to get the best hyperparamters. *dyn* is short for *dynamic* and *dynamic-force* adopts $\theta_l = 40$.

θ_l	F-score	BLEU	avgCW
10	0.40	16.27	**5.85**
20	0.58	20.34	9.74
40	**0.74**	**22.01**	14.43
80	0.73	21.60	15.15

Table 4: Segmentation Performance of *dynamic-force* trained on IWSLT2014. All methods are conducted with future words M equals to 1.

per row), which are in real speech order. We argue that these sentences are sufficient to model the classification of simple samples, based on the rapid overfit performance of the other three methods.

4.3 Out-of-Domain vs. In-Domain

Next, we turn to the question that how does the domain of training corpus affects results. With the test set unchanged, we compare the sentence boundary detections model trained on out-of-domain corpora WMT 14 and in-domain corpora IWSLT 14, respectively.

As mentioned before, WMT 14 is a larger text translation corpus mainly on news and web sources. But the test set comes from IWSLT, which contains transcriptions of TED lectures of various directions. Intuitively, larger dataset provides more diverse samples, but due to domain changes, it does not necessarily lead to improvements in accuracy.

The performance of various models trained on WMT14 is shown in Table 3. *Dynamic-force* also achieves the best translation performance with a relatively small latency on average and limited the max latency within 40 words. However, it underperforms the same model trained on IWSLT2014 (as shown in Table 2), demonstrating its sensitivity to the training domain.

On the contrary, *N-gram* and *T-LSTM* is hardly affected. For *N-gram*, one possible reason is the before mentioned weakness of the N-gram: segmentation depends on only N previous words, which is more steady compared to the whole sentence, thus eliminating the perturbation of whole sentence brought by the domain variation. For *T-LSTM*, it even improves a little compared with its in-domain performance. This may be due to the lack of training samples. 0.19M sentences of IWSLT2014 is insufficient to fit the parameters of *T-LSTM*. Thus the model would benefit from increasing the corpus size. However, our method needs less data in training because our model has been pre-trained. Based on a powerful representation, we need only a small amount of training data in fine-tuning, which is best aligned with the test set in the domain.

4.4 Length of window θ_l

Next, we discuss the effect of changing θ. The performance of *dynamic-force* with varying θ_l is shown in Table 4. Smaller θ_l brings shorter latency, as well as worse performance. The effect is extremely poor with $\theta_l = 10$. There are two possible reasons: 1) Constraint sentence length less than θ_l is too harsh under small θ_l, 2) The discrepancy between the unrestricted training and length-restricted testing causes the poor effect.

We first focus on the second possible reason. While the difference between *dynamic-base* and *dynamic-force* is only in prediction, we want to know whether we can achieve better results by controlling the length of training samples. Accordingly, we only use the samples shorter than a fixed value: θ_l in training phrase. At inference time, we use both *dynamic-force* with the same sentence length constraint θ_l and *dynamic-base* to predict sentence boundaries. As elaborated in Figure 5, For each pair of curves with a same θ_l, *dynamic-force* and *dynamic-base* present similar performance. This demonstrates the main reason for the poor performance with small θ_l is not the training-testing discrepancy but lies in the first reason that the force constraint is too harsh.

Moreover, it is interesting to find that the performance of $\theta_l = 80$ is similar with $\theta_l = 40$ at the beginning but falls a little during training. This probably because the setup with $\theta_l = 40$ can filter some inaccurate cases, as the average number of words in IWSLT2014 training set is 20.26.

4.5 Number of Future Words M

We investigate whether can we achieve better performance with more or less future words. We experiment with M from 0 to 5. The result is shown

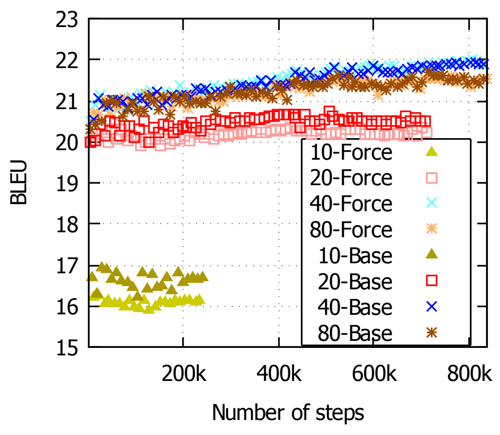


Figure 5: Translation performance on IWSLT2014 testset. "θ_l-Force" denotes to set the sentence length threshold to θ_l in both training sample generation and prediction. "θ_l-Base" is to set this constraint only in training samples generation process.

M	F-score	BLEU	avgCW
0	0.66	21.54	13.23
1	0.74	22.01	14.43
2	0.77	22.23	15.24
3	0.79	22.23	16.52
4	0.80	22.29	17.15

Table 5: Segmentation Performance of *dynamic-force* trained on IWSLT2014. All methods are conducted with $\theta_l = 40$.

in Table 5. Reducing M to zero means that do not refer to any future words in prediction. This degrades performance a lot, proving the effectiveness of adding future words in prediction. Increase M from 1 to 2 also promote the performance in both sentence boundary detection f-score and the system BLEU. However, as more future words added (increase M to 3 and 4), the improvement becomes less obvious.

5 Related Work

Sentence boundary detection has been explored for years, but the majority of these work focuses on offline punctuation restoration, instead of applied in simultaneous translation. Existing work can be divided into two classes according to the model input.

5.1 N-gram based methods

Some work takes a fixed size of words as input. Focus on utilizing a limited size of the streaming input, they predict the probability of putting a boundary at a specific position x_t by a N-gram language model (Wang et al., 2016) or a classification model (Sridhar et al., 2013; Yarmohammadi et al., 2013). The language-model based method make decision depends on N words $(x_{t-N+2}, ..., x_{t+1})$ and compares its probability with $(x_{t-N+2}, ..., x_t, \text{"."})$. The classification model takes features of N words around x_t and classifies to two classes denoting x_t is a sentence boundary or not. The main deficiency of this method is that the dependencies outside the input window are lost, resulting in low accuracy.

5.2 Whole sentence-based methods

Some other work focuses on restoring punctuation and capitalization using the whole sentence. To improve the sentence boundary classification accuracy, some work upgrade the N-gram input to variable-length input by using recurrent neural network (RNN) (Tilk and Alumäe, 2015; Salloum et al., 2017). Some other work takes punctuation restoration as a sequence labeling problem and investigates using Conditional Random Fields (CRFs) (Lu and Ng, 2010; Wang et al., 2012; Ueffing et al., 2013). Peitz et al. (2011) and Cho et al. (2012) treats this problem as a machine translation task, training to translate non-punctuated transcription into punctuated text. However, all these methods utilize the whole sentence information, which is not fit for the simultaneous translation scenario. Moreover, the translation model based methods require multiple steps of decoding, making it unsuitable for online prediction.

6 Conclusion

In this paper, we propose an online sentence boundary detection approach. With the input of streaming words, our model predicts the probability of multiple positions rather than a certain position. By adding this adjacent position constraint and using dynamic prediction, our method achieves higher accuracy with lower latency.

We also incorporate the pre-trained technique, ERNIE to implement our classification model. The empirical results on IWSLT2014 demonstrate that our approach achieves significant improvements of 0.19 F-score on sentence segmentation and 1.55 BLEU points compared with the language-model based methods.

References

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Srinivas Bangalore, 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*. Association for Computational Linguistics.

Eunah Cho, Jan Niehues, and Alex Waibel. 2012. Segmentation and punctuation prediction in speech language translation using a monolingual translation system. In *International Workshop on Spoken Language Translation (IWSLT) 2012*.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*.

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

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor OK Li. 2017. Learning to translate in real-time with neural machine translation.

Wei Lu and Hwee Tou Ng. 2010. Better punctuation prediction with dynamic conditional random fields. In *Proceedings of the 2010 conference on empirical methods in natural language processing*.

Mingbo Ma, Liang Huang, Hao Xiong, Kaibo Liu, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, and Haifeng Wang. 2019. STACL: simultaneous translation with integrated anticipation and controllable latency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics.

Stephan Peitz, Markus Freitag, Arne Mauser, and Hermann Ney. 2011. Modeling punctuation prediction as machine translation. In *International Workshop on Spoken Language Translation (IWSLT) 2011*.

Wael Salloum, Gregory Finley, Erik Edwards, Mark Miller, and David Suendermann-Oeft. 2017. Deep learning for punctuation restoration in medical reports. In *BioNLP 2017*.

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ottokar Tilk and Tanel Alumäe. 2015. Lstm for punctuation restoration in speech transcripts. In *Sixteenth annual conference of the international speech communication association*.

Nicola Ueffing, Maximilian Bisani, and Paul Vozila. 2013. Improved models for automatic punctuation prediction for spoken and written text. In *Interspeech*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *31st Conference on Neural Information Processing Systems (NIPS 2017)*.

L. Venuti. 2012. The translation studies reader.

Xiaolin Wang, 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)*.

Xuancong Wang, Hwee Tou Ng, and Khe Chai Sim. 2012. Dynamic conditional random fields for joint sentence boundary and punctuation prediction. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*.