Optimizing Segmentation Strategies for Simultaneous Speech Translation

Yusuke Oda **Graham Neubig** **Sakriani Sakti** **Tomoki Toda** **Satoshi Nakamura**
Graduate School of Information Science
Nara Institute of Science and Technology
Takayama, Ikoma, Nara 630-0192, Japan
{oda.yusuke.on9, neubig, ssakti, tomoki, s-nakamura}@is.naist.jp

Abstract

In this paper, we propose new algorithms for learning segmentation strategies for simultaneous speech translation. In contrast to previously proposed heuristic methods, our method finds a segmentation that directly maximizes the performance of the machine translation system. We describe two methods based on greedy search and dynamic programming that search for the optimal segmentation strategy. An experimental evaluation finds that our algorithm is able to segment the input two to three times more frequently than conventional methods in terms of number of words, while maintaining the same score of automatic evaluation.[1]

1 Introduction

The performance of speech translation systems has greatly improved in the past several years, and these systems are starting to find wide use in a number of applications. Simultaneous speech translation, which translates speech from the source language into the target language in real time, is one example of such an application. When translating dialogue, the length of each utterance will usually be short, so the system can simply start the translation process when it detects the end of an utterance. However, in the case of lectures, for example, there is often no obvious boundary between utterances. Thus, translation systems require a method of deciding the timing at which to start the translation process. Using estimated ends of sentences as the timing with which to start translation, in the same way as a normal text translation, is a straightforward solution to this problem (Matusov et al., 2006). However, this approach impairs the simultaneity of translation because the system needs to wait too long until the appearance of a estimated sentence boundary. For this reason, segmentation strategies, which separate the input at appropriate positions other than end of the sentence, have been studied.

A number of segmentation strategies for simultaneous speech translation have been proposed in recent years. Fügen et al. (2007) and Bangalore et al. (2012) propose using prosodic pauses in speech recognition to denote segmentation boundaries, but this method strongly depends on characteristics of the speech, such as the speed of speaking. There is also research on methods that depend on linguistic or non-linguistic heuristics over recognized text (Rangarajan Sridhar et al., 2013), and it was found that a method that predicts the location of commas or periods achieves the highest performance. Methods have also been proposed using the phrase table (Yarmohammadi et al., 2013) or the right probability (RP) of phrases (Fujita et al., 2013), which indicates whether a phrase reordering occurs or not.

However, each of the previously mentioned methods decides the segmentation on the basis of heuristics, so the impact of each segmentation strategy on translation performance is not directly considered. In addition, the mean number of words in the translation unit, which strongly affects the delay of translation, cannot be directly controlled by these methods.[2]

In this paper, we propose new segmentation algorithms that directly optimize translation performance given the mean number of words in the translation unit. Our approaches find appropriate segmentation boundaries incrementally using greedy search and dynamic programming. Each boundary is selected to explicitly maximize trans-

[1] The implementation is available at http://odaemon.com/docs/codes/greedyseg.html.

[2] The method using RP can decide relative frequency of segmentation by changing a parameter, but guessing the length of a translation unit from this parameter is not trivial.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 551–556,
Baltimore, Maryland, USA, June 23-25 2014.

lation accuracy as measured by BLEU or another evaluation measure.

We evaluate our methods on a speech translation task, and we confirm that our approaches can achieve translation units two to three times as fine-grained as other methods, while maintaining the same accuracy.

2 Optimization Framework

Our methods use the outputs of an existing machine translation system to learn a segmentation strategy. We define $\mathcal{F} = \{\boldsymbol{f}_j : 1 \leq j \leq N\}$, $\mathcal{E} = \{\boldsymbol{e}_j : 1 \leq j \leq N\}$ as a parallel corpus of source and target language sentences used to train the segmentation strategy. N represents the number of sentences in the corpus. In this work, we consider sub-sentential segmentation, where the input is already separated into sentences, and we want to further segment these sentences into shorter units. In an actual speech translation system, these sentence boundaries can be estimated automatically using a method like the period estimation mentioned in Rangarajan Sridhar et al. (2013). We also assume the machine translation system is defined by a function $MT(\boldsymbol{f})$ that takes a string of source words $\boldsymbol{f}$ as an argument and returns the translation result $\hat{\boldsymbol{e}}$.[3]

We will introduce individual methods in the following sections, but all follow the general framework shown below:

1. Decide the mean number of words μ and the machine translation evaluation measure EV as parameters of algorithm. We can use an automatic evaluation measure such as BLEU (Papineni et al., 2002) as EV. Then, we calculate the number of sub-sentential segmentation boundaries K that we will need to insert into $\mathcal{F}$ to achieve an average segment length μ:

$$K := \max\left(0, \left\lfloor \frac{\sum_{\boldsymbol{f}\in\mathcal{F}} |\boldsymbol{f}|}{\mu} \right\rfloor - N\right). \quad (1)$$

2. Define $\mathcal{S}$ as a set of positions in $\mathcal{F}$ in which we will insert segmentation boundaries. For example, if we will segment the first sentence after the third word and the third sentence after the fifth word, then $\mathcal{S} = \{\langle 1, 3\rangle, \langle 3, 5\rangle\}$.

[3]In this work, we do not use the history of the language model mentioned in Bangalore et al. (2012). Considering this information improves the MT performance and we plan to include this in our approach in future work.

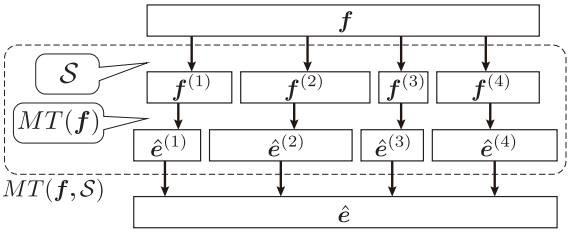

Figure 1: Concatenated translation $MT(\boldsymbol{f}, \mathcal{S})$.

Based on this representation, choose K segmentation boundaries in $\mathcal{F}$ to make the set $\mathcal{S}^*$ that maximizes an evaluation function ω as below:

$$\mathcal{S}^* := \underset{\mathcal{S}\in\{\mathcal{S}':|\mathcal{S}'|=K\}}{\arg\max} \omega(\mathcal{S}; \mathcal{F}, \mathcal{E}, EV, MT). \quad (2)$$

In this work, we define ω as the sum of the evaluation measure for each parallel sentence pair $\langle \boldsymbol{f}_j, \boldsymbol{e}_j \rangle$:

$$\omega(\mathcal{S}) := \sum_{j=1}^{N} EV(MT(\boldsymbol{f}_j, \mathcal{S}), \boldsymbol{e}_j), \quad (3)$$

where $MT(\boldsymbol{f}, \mathcal{S})$ represents the concatenation of all partial translations $\{MT(\boldsymbol{f}^{(n)})\}$ given the segments $\mathcal{S}$ as shown in Figure 1.

Equation (3) indicates that we assume all parallel sentences to be independent of each other, and the evaluation measure is calculated for each sentence separately. This locality assumption eases efficient implementation of our algorithm, and can be realized using a sentence-level evaluation measure such as BLEU+1 (Lin and Och, 2004).

3. Make a segmentation model $M_{\mathcal{S}^*}$ by treating the obtained segmentation boundaries $\mathcal{S}^*$ as positive labels, all other positions as negative labels, and training a classifier to distinguish between them. This classifier is used to detect segmentation boundaries at test time.

Steps 1. and 3. of the above procedure are trivial. In contrast, choosing a good segmentation according to Equation (2) is difficult and the focus of the rest of this paper. In order to exactly solve Equation (2), we must perform brute-force search over all possible segmentations unless we make some assumptions about the relation between the ω yielded by different segmentations. However, the number of possible segmentations is exponentially large, so brute-force search is obviously intractable. In the following sections, we propose 2

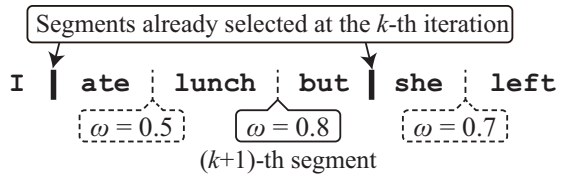

Figure 2: Example of greedy search.

Algorithm 1 *Greedy* segmentation search

$\mathcal{S}^* \leftarrow \varnothing$
for $k = 1$ **to** K **do**
$$\mathcal{S}^* \leftarrow \mathcal{S}^* \cup \left\{ \arg\max_{s \notin \mathcal{S}^*} \omega(\mathcal{S}^* \cup \{s\}) \right\}$$
end for
return $\mathcal{S}^*$

methods that approximately search for a solution to Equation (2).

2.1 Greedy Search

Our first approximation is a greedy algorithm that selects segmentation boundaries one-by-one. In this method, k already-selected boundaries are left unchanged when deciding the $(k+1)$-th boundary. We find the unselected boundary that maximizes ω and add it to $\mathcal{S}$:

$$\mathcal{S}_{k+1} = \mathcal{S}_k \cup \left\{ \arg\max_{s \notin \mathcal{S}_k} \omega(\mathcal{S}_k \cup \{s\}) \right\}. \quad (4)$$

Figure 2 shows an example of this process for a single sentence, and Algorithm 1 shows the algorithm for calculating K boundaries.

2.2 Greedy Search with Feature Grouping and Dynamic Programming

The method described in the previous section finds segments that achieve high translation performance for the training data. However, because the translation system MT and evaluation measure EV are both complex, the evaluation function ω includes a certain amount of noise. As a result, the greedy algorithm that uses only ω may find a segmentation that achieves high translation performance in the training data by chance. However, these segmentations will not generalize, reducing the performance for other data.

We can assume that this problem can be solved by selecting more consistent segmentations of the training data. To achieve this, we introduce a constraint that all positions that have similar characteristics must be selected at the same time. Specifically, we first group all positions in the source

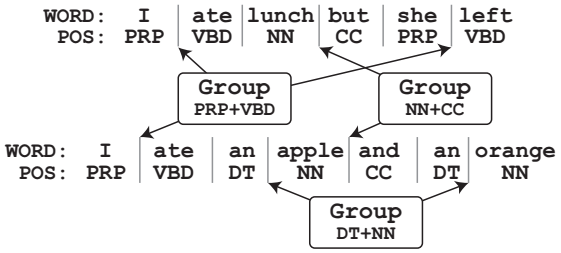

Figure 3: Grouping segments by POS bigrams.

sentences using features of the position, and introduce a constraint that all positions with identical features must be selected at the same time. Figure 3 shows an example of how this grouping works when we use the POS bigram surrounding each potential boundary as our feature set.

By introducing this constraint, we can expect that features which have good performance overall will be selected, while features that have relatively bad performance will not be selected even if good performance is obtained when segmenting at a specific location. In addition, because all positions can be classified as either segmented or not by evaluating whether the corresponding feature is in the learned feature set or not, it is not necessary to train an additional classifier for the segmentation model when using this algorithm. In other words, this constraint conducts a kind of feature selection for greedy search.

In contrast to Algorithm 1, which only selected one segmentation boundary at once, in our new setting there are multiple positions selected at one time. Thus, we need to update our search algorithm to handle this setting. To do so, we use dynamic programming (DP) together with greedy search. Algorithm 2 shows our *Greedy+DP* search algorithm. Here, $c(\phi; \mathcal{F})$ represents the number of appearances of ϕ in the set of source sentences $\mathcal{F}$, and $\mathcal{S}(\mathcal{F}, \Phi)$ represents the set of segments defined by both $\mathcal{F}$ and the set of features Φ.

The outer loop of the algorithm, like *Greedy*, iterates over all $\mathcal{S}$ of size 1 to K. The inner loop examines all features that appear exactly j times in $\mathcal{F}$, and measures the effect of adding them to the best segmentation with $(k - j)$ boundaries.

2.3 Regularization by Feature Count

Even after we apply grouping by features, it is likely that noise will still remain in the less frequently-seen features. To avoid this problem, we introduce regularization into the *Greedy+DP* algorithm, with the evaluation function ω rewrit-

Algorithm 2 *Greedy+DP* segmentation search

$\Phi_0 \leftarrow \varnothing$
for $k = 1$ **to** K **do**
  for $j = 0$ **to** $k-1$ **do**
    $\Phi' \leftarrow \{\phi : c(\phi; \mathcal{F}) = k - j \wedge \phi \not\in \Phi_j\}$
    $\Phi_{k,j} \leftarrow \Phi_j \cup \left\{ \arg\max_{\phi \in \Phi'} \omega(\mathcal{S}(\mathcal{F}, \Phi_j \cup \{\phi\})) \right\}$
  end for
  $\Phi_k \leftarrow \arg\max_{\Phi \in \{\Phi_{k,j} : 0 \le j < k\}} \omega(\mathcal{S}(\mathcal{F}, \Phi))$
end for
return $\mathcal{S}(\mathcal{F}, \Phi_K)$

ten as below:

$$\omega_\alpha(\Phi) := \omega(\mathcal{S}(\mathcal{F}, \Phi)) - \alpha|\Phi|. \quad (5)$$

The coefficient α is the strength of the regularization with regards to the number of selected features. A larger α will result in a larger penalty against adding new features into the model. As a result, the *Greedy+DP* algorithm will value frequently appearing features. Note that the method described in the previous section is equal to the case of $\alpha = 0$ in this section.

2.4 Implementation Details

Our *Greedy* and *Greedy+DP* search algorithms are completely described in Algorithms 1 and 2. However, these algorithms require a large amount of computation and simple implementations of them are too slow to finish in realistic time. Because the heaviest parts of the algorithm are the calculation of MT and EV, we can greatly improve efficiency by memoizing the results of these functions, only recalculating on new input.

3 Experiments

3.1 Experimental Settings

We evaluated the performance of our segmentation strategies by applying them to English-German and English-Japanese TED speech translation data from WIT3 (Cettolo et al., 2012). For English-German, we used the TED data and splits from the IWSLT2013 evaluation campaign (Cettolo et al., 2013), as well as 1M sentences selected from the out-of-domain training data using the method of Duh et al. (2013). For English-Japanese, we used TED data and the dictionary entries and sentences from EIJIRO.[4] Table 1 shows summaries of the datasets we used.

[4] http://eowp.alc.co.jp/info2/

f-e	Type	#words f	#words e
En-De	Train MT	21.8M	20.3M
	Train Seg.	424k	390k
	Test	27.6k	25.4k
En-Ja	Train MT	13.7M	19.7M
	Train Seg.	401k	550k
	Test	8.20k	11.9k

Table 1: Size of MT training, segmentation training and testing datasets.

We use the Stanford POS Tagger (Toutanova et al., 2003) to tokenize and POS tag English and German sentences, and KyTea (Neubig et al., 2011) to tokenize Japanese sentences. A phrase-based machine translation (PBMT) system learned by Moses (Koehn et al., 2007) is used as the translation system MT. We use BLEU+1 as the evaluation measure EV in the proposed method. The results on the test data are evaluated by BLEU and RIBES (Isozaki et al., 2010), which is an evaluation measure more sensitive to global reordering than BLEU.

We evaluated our algorithm and two conventional methods listed below:

Greedy is our first method that uses simple greedy search and a linear SVM (using surrounding word/POS 1, 2 and 3-grams as features) to learn the segmentation model.

Greedy+DP is the algorithm that introduces grouping the positions in the source sentence by POS bigrams.

Punct-Predict is the method using predicted positions of punctuation (Rangarajan Sridhar et al., 2013).

RP is the method using right probability (Fujita et al., 2013).

3.2 Results and Discussion

Figures 4 and 5 show the results of evaluation for each segmentation strategy measured by BLEU and RIBES respectively. The horizontal axis is the mean number of words in the generated translation units. This value is proportional to the delay experienced during simultaneous speech translation (Rangarajan Sridhar et al., 2013) and thus a smaller value is desirable.

RP, *Greedy*, and *Greedy+DP* methods have multiple results in these graphs because these methods have a parameter that controls segmentation frequency. We move this parameter from no segmentation (sentence-based translation) to

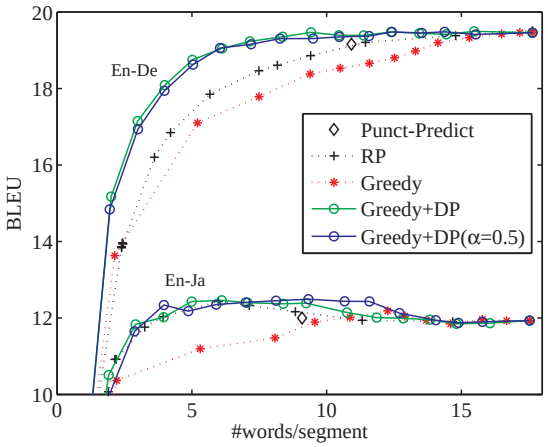

Figure 4: BLEU score of test set.

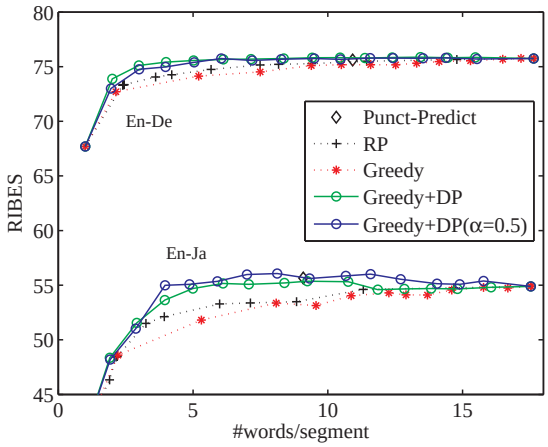

Figure 5: RIBES score of test set.

segmenting every possible boundary (word-based translation) and evaluate the results.

First, focusing on the *Greedy* method, we can see that it underperforms the other methods. This is a result of over-fitting as will be described in detail later. In contrast, the proposed *Greedy+DP* method shows high performance compared to the other methods. Especially, the result of BLEU on the English-German and the RIBES on both language pairs show higher performance than *RP* at all speed settings. *Punct-Predict* does not have an adjustable parameter, so we can only show one point. We can see that *Greedy+DP* can begin translation about two to three times faster than *Punct-Predict* while maintaining the same performance.

Figure 6 shows the BLEU on the training data. From this figure, it is clear that *Greedy* achieves much higher performance than *Greedy+DP*. From this result, we can see that the *Greedy* algorithm is choosing a segmentation that achieves high accuracy on the training data but does not generalize to the test data. In contrast, the grouping constraint in the *Greedy+DP* algorithm is effectively suppressing this overfitting.

The mean number of words μ can be decided independently from other information, but a configuration of μ affects tradeoff relation between translation accuracy and simultaneity. For example, smaller μ makes faster translation speed but it also makes less translation accuracy. Basically, we should choose μ by considering this tradeoff.

4 Conclusion and Future Work

We proposed new algorithms for learning a segmentation strategy in simultaneous speech translation. Our algorithms directly optimize the performance of a machine translation system according to an evaluation measure, and are calculated by greedy search and dynamic programming. Experiments show our *Greedy+DP* method effectively separates the source sentence into smaller units while maintaining translation performance.

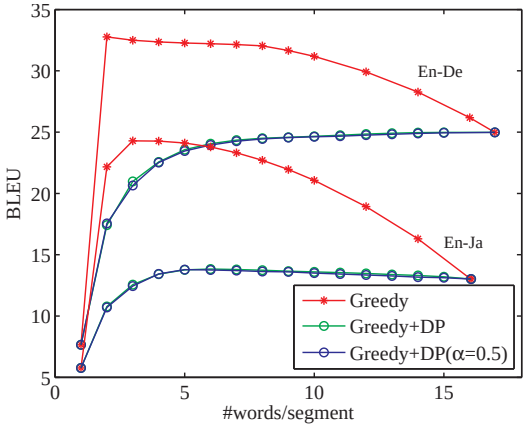

Figure 6: BLEU score of training set.

With regards to future work, it has been noted that translation performance can be improved by considering the previously translated segment when calculating LM probabilities (Rangarajan Sridhar et al., 2013). We would like to expand our method to this framework, although incorporation of context-sensitive translations is not trivial. In addition, the *Greedy+DP* algorithm uses only one feature per a position in this paper. Using a variety of features is also possible, so we plan to examine expansions of our algorithm to multiple overlapping features in future work.

Acknowledgements

Part of this work was supported by JSPS KAKENHI Grant Number 24240032.

References

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proc. NAACL HLT*, pages 437–445.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proc. EAMT*, pages 261–268.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th iwslt evaluation campaign. In *Proc. IWSLT*.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proc. ACL*, pages 678–683.

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

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

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proc. EMNLP*, pages 944–952.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL*, pages 177–180.

Chin-Yew Lin and Franz Josef Och. 2004. Orange: A method for evaluating automatic evaluation metrics for machine translation. In *Proc. COLING*.

Evgeny Matusov, Arne Mauser, and Hermann Ney. 2006. Automatic sentence segmentation and punctuation prediction for spoken language translation. In *Proc. IWSLT*, pages 158–165.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proc. NAACL HLT*, pages 529–533.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proc. NAACL HLT*, pages 230–238.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. NAACL*, pages 173–180.

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