Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 18 No. 23 (2024) | OPEN ACCESS

https://doi.org/10.3991/ijim.v18i23.52879

PAPER

Optimization and Performance Analysis of Real-time Speech Translation Systems Based on Mobile Technology

Ning Yang(✉)

School of Education, Xi'an Fanyi University, Xi'an, China

yn@xafy.edu.cn

ABSTRACT

As globalization deepens and mobile technology rapidly advances, the demand for cross-linguistic communication has been steadily increasing, making real-time speech translation systems a research focus. However, given the limited computational capacity and storage space of mobile devices, optimizing system performance while maintaining translation quality has become a critical challenge. Current optimization approaches for real-time speech translation systems primarily focus on improvements to model architectures and hardware acceleration, often neglecting a systematic study of model compression. This is particularly evident when handling real-time data, where achieving both high efficiency and translation accuracy remains difficult. To address these challenges, a model compression method based on the connectionist temporal classification (CTC) criterion was proposed, along with an in-depth study of parameter compression tailored for mobile applications. The research focuses on two key areas: first, model compression techniques based on the CTC criterion were explored to enhance the efficiency of real-time speech translation; second, parameter compression methods were investigated to significantly reduce resource consumption in mobile applications while preserving translation quality. The aim of this study is to improve the performance and user experience of real-time speech translation systems on mobile devices.

KEYWORDS

real-time speech translation, mobile technology, model compression, connectionist temporal classification (CTC), criterion, performance optimization, parameter compression

1 INTRODUCTION

With the acceleration of globalization and the rapid advancement of mobile technologies, the demand for cross-linguistic communication has grown significantly, leading to the emergence of real-time speech translation systems [1–4]. These systems enable instant communication across various contexts, greatly facilitating

Yang, N. (2024). Optimization and Performance Analysis of Real-time Speech Translation Systems Based on Mobile Technology. *International Journal of Interactive Mobile Technologies (iJIM)*, 18(23), pp. 57–71. https://doi.org/10.3991/ijim.v18i23.52879

Article submitted 2024-08-14. Revision uploaded 2024-09-27. Final acceptance 2024-10-03.

international exchanges and collaboration [5, 6]. The widespread adoption of mobile devices has brought real-time speech translation systems based on mobile technologies into the spotlight of research. However, mobile devices are limited in both computational power and storage capacity, raising the critical challenge of optimizing system performance while ensuring translation quality.

The optimization of real-time speech translation systems is not only of academic importance but also has broad practical implications [7–10]. Efficiently optimized systems can operate across a wide range of mobile devices, enhancing the user experience and expanding their application scope [11–15]. In sectors such as business, tourism, education, and healthcare, real-time speech translation technology has the potential to break down language barriers, offering instant and accurate translation services that foster the development of various industries.

Despite some progress made in the optimization of real-time speech translation systems, several gaps remain. Existing approaches are predominantly focused on improving model architectures and leveraging hardware acceleration, with relatively little attention given to systematic studies of model compression [16–18]. Furthermore, current compression methods often encounter performance bottlenecks when handling real-time data, making it difficult to achieve both efficient operation and high translation accuracy [19–21]. This challenge is particularly pronounced on mobile devices, where maintaining efficiency and accuracy under resource constraints remains an unresolved issue.

In response to these challenges, this study proposes a novel model compression method for real-time speech translation based on the connectionist temporal classification (CTC) criterion, alongside an in-depth investigation into parameter compression tailored for mobile applications. The research focuses on two primary areas: firstly, exploring CTC-based model compression techniques to enhance the efficiency of real-time speech translation; and secondly, examining parameter compression methods designed to significantly reduce resource consumption in mobile applications while preserving translation quality. This study aims to provide new insights and approaches for optimizing real-time speech translation systems, ultimately improving their performance and user experience on mobile devices. The research carries both theoretical and practical significance.

2 CTC CRITERION FOR MODEL COMPRESSION IN REAL-TIME SPEECH TRANSLATION

In modern society, the demand for cross-linguistic communication continues to rise, and the application scenarios for real-time speech translation systems have become increasingly diverse. As mobile devices become more ubiquitous, it has become crucial for these systems to operate efficiently on mobile platforms. Figure 1 illustrates the framework of a real-time speech translation system. However, mobile devices are typically constrained by limited computational capacity, storage space, and battery life, making the achievement of efficient, real-time speech translation under such resource limitations a significant challenge. To address this challenge, the compression of model parameters in speech recognition systems, specifically for continuous real-time speech sequence classification, is critical.

Real-time performance is one of the core features of speech translation systems. Users expect immediate translation feedback when using such systems to facilitate smooth cross-linguistic communication. To achieve this on mobile devices, speech recognition models must be capable of processing input speech signals quickly and

generating translation outputs in real time. However, modern speech recognition models often contain a large number of parameters, which consume significant computational resources and memory during operation. This results in slower system response times, making it difficult to meet real-time requirements. Therefore, compressing model parameters can significantly reduce computational and storage demands, thereby enhancing the system's ability to process data in real time.

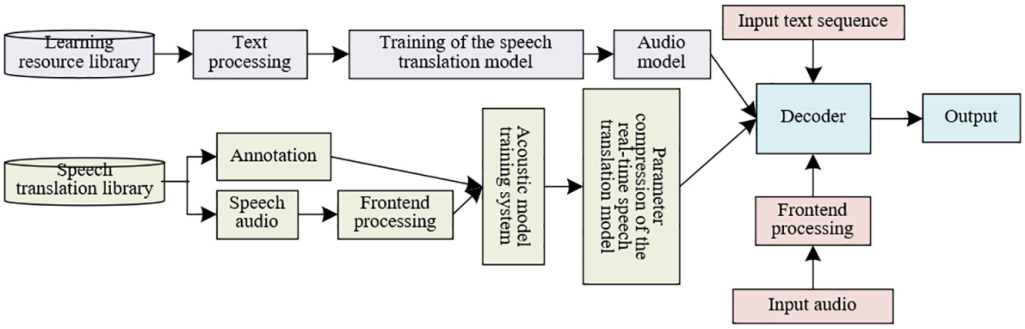

Fig. 1. Framework of the real-time speech translation system

The limited hardware resources of mobile devices, particularly in terms of storage capacity and computational power, fall significantly short when compared to server-grade equipment. Running large-scale speech recognition models on mobile platforms can lead to memory shortages and computational bottlenecks, negatively impacting both translation quality and user experience. Parameter compression techniques offer an effective means to reduce model storage requirements and computational complexity, enabling efficient operation on resource-constrained mobile devices. This ensures the stability and reliability of the system.

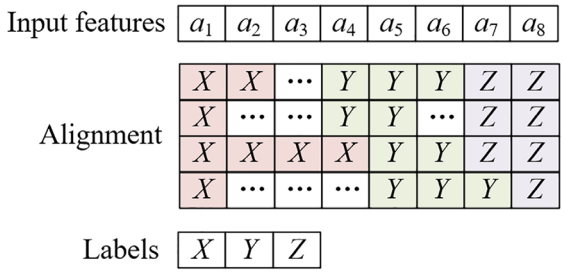

Fig. 2. CTC alignment diagram

The CTC criterion has demonstrated outstanding performance in handling speech recognition tasks. The CTC criterion efficiently manages input sequences of varying lengths, making it well-suited to different speech signals and inherently advantageous for real-time speech translation applications. However, even models based on the CTC criterion face challenges related to resource limitations and real-time performance if the number of parameters is too large. Compressing the parameters of models that utilize the CTC criterion allows the system to retain its efficient handling of variable-length speech sequences while significantly reducing model size and further enhancing real-time translation performance. Figure 2 provides an illustration of the CTC alignment.

In speech translation tasks, the input sequence consists of feature sequences derived from speech signals, where the length of the input sequence is typically much greater than the corresponding annotated text sequence. For example, a few

seconds of speech may contain hundreds of feature frames, while the corresponding text annotation may consist of only a few dozen characters. Traditional models, such as bidirectional long short-term memory (BiLSTM) networks, generally require a one-to-one correspondence between the input and output sequences, which is unrealistic in speech recognition. The CTC criterion addresses this alignment issue by introducing a special "blank" label that allows the network to make an output at each time step. This enables the length of the annotated sequence to be shorter than the input sequence, thereby solving the alignment problem. This flexibility makes CTC highly suitable for speech recognition and translation tasks. Moreover, in conventional sequence modelling methods, complex alignment algorithms are required to align input sequences with output sequences. CTC, by contrast, directly optimizes the conditional probability of the annotated sequence, thus avoiding the need for an explicit alignment process. This not only simplifies the model training process but also improves training efficiency. Consequently, in resource-limited environments such as mobile devices, the CTC criterion reduces both computational complexity and memory usage, allowing real-time speech translation systems to operate more efficiently.

In practical speech translation tasks, the input is a sequence of speech features, which is generally much longer than the annotated text sequence. CTC networks include a softmax layer with one additional unit beyond the set of characters, denoted as L, where one unit represents the probability of observing a "blank" output, and the remaining $|M|$ units represent the probability of observing the corresponding label at a given time. The outputs from these units represent the probability of all possible ways of aligning the annotated sequence with the input sequence. For an input sequence a of length S, a BiLSTM defined as $V_q(E^l)S \mapsto (E^l)S$ with input dimension l, output dimension v, and connection weight vector q was applied. The output sequence is defined as $b = V_q(a)$, where b_j^s denotes the activation of unit j at time step s, representing the probability of observing label j at time s. This defines the probability distribution of the sequence set M'^S of length T on the character set $M' = M \cup \{BLANK\}$. Through this probability distribution, CTC directly optimizes the conditional probability of the annotated sequence without the need for explicit alignment between the input and output sequences. This not only enhances the training efficiency of the model but also reduces computational complexity, making it particularly suitable for resource-constrained mobile devices. Given a path τ in the set of character sequences M'^S, the probability distribution is expressed as follows:

$$o(\tau \mid a) = \prod_{s=1}^{S} b_{\tau_s}^s, \forall \tau \in M'^S \tag{1}$$

The conditional probability of a given annotated sequence m, defined by the many-to-one mapping relationship α, is expressed as follows:

$$o(m \mid a) = \sum_{\tau \in \alpha^{-1}(1)} o(\tau \mid a) \tag{2}$$

The output formula of the classifier g is given as follows:

$$g(a) = \underset{1 \in M^{\leq S}}{\operatorname{argmax}}\, o(m \mid s) \tag{3}$$

In real-time speech translation systems based on mobile technology, the decoding process is a critical step in converting speech to text. To optimize this process, a

multi-stage decoding strategy was adopted in this study, combining the advantages of best-path decoding and prefix-search decoding. For instance, in the initial stage, best-path decoding was used to generate results quickly. As user interaction continued, prefix-search decoding was gradually applied to refine the output. This strategy ensures that real-time performance is maintained while improving translation accuracy. Best-path decoding assumes that the most probable annotated sequence corresponds to the output label with the highest probability at each time step. The specific steps are as follows: a) For each time step, the label with the highest probability was selected; b) The selected labels were concatenated to form an initial output sequence; and c) Blank labels and consecutive repeated labels were removed to yield the final annotated sequence. The calculation is expressed as follows:

$$g(a) \approx \alpha(\tau^*) WHERE\, \tau^* = \arg\max_{\tau} o\left(\tau \mid a\right) \tag{4}$$

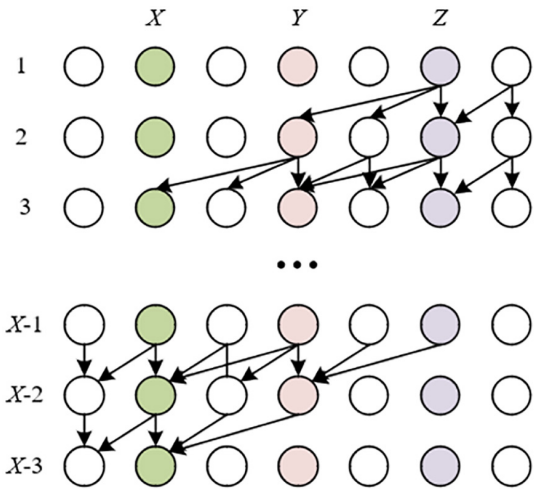

Fig. 3. Application of the forward-backward algorithm to annotating translated speech sequences

Prefix search decoding employs dynamic programming to consider all possible prefixes of the annotated sequences, progressively expanding and calculating the probability of each prefix. The specific steps are as follows: a) The prefix set was initialized, with the initial probability of each prefix set to 1; b) At each time step, the prefix set was updated by calculating the probabilities for all possible new prefixes that can be generated by extending each existing prefix to the next time step; c) Identical prefixes were merged, and their probabilities were accumulated; and d) The prefixes with the highest probabilities were retained, while prefixes with lower probabilities were discarded. These steps are repeated until all time steps can be processed. Figure 3 provides an example of the forward-backward algorithm applied to annotating translated speech sequences.

To calculate the conditional probability of a specific annotated sequence, the forward-backward algorithm is incorporated into the CTC framework. This algorithm increases computational efficiency by iteratively calculating the probabilities of partial paths, breaking down the complex global computation into a series of simpler local computations. In CTC, the annotated sequence is modified to include blank characters. For example, the original sequence "*HELLO*" is transformed into "*#H#E#L#L#O#*", where "#" represents a blank character. These blank characters allow the model to transition between different characters more flexibly, thus aligning input and output sequences.

The forward-backward algorithm calculates the conditional probability of the entire annotated sequence by evaluating the partial path probabilities at each time step. Specifically, the forward algorithm computes the cumulative probability of all possible paths ending with the current character at each time step, while the backward algorithm calculates the cumulative probability of all possible paths starting from the current character at each time step. By combining these two partial path probabilities, the conditional probability of the entire annotated sequence can be obtained. Specifically, for a given sequence w of length e, the notations $w_{1:o}$ and $w_{e-o:e}$ represent the first o characters and the last o characters of w, respectively. For an annotated sequence m, the total probability of sequence $m_{1:t}$ at time s is defined as the forward variable $\beta_s(t)$, and can be calculated as follows:

$$\beta_s(t) = \sum_{\tau:\alpha(\tau_{1:s})=1_{1:t}} \prod_{t'=1}^{s} b_{\tau_{s'}}^{s'} \tag{5}$$

The forward variable was initialised as follows:

$$\beta_1(1) = b_y^1 \tag{6}$$

$$\beta_1(2) = b_{1_1}^1 \tag{7}$$

$$\beta_1(t) = 0, \forall t > 2 \tag{8}$$

The recursive relationship was defined as follows:

$$\beta_s(t) = \begin{cases} \bar{\beta}_s(t) b_{1_t'}^s & IF\ m_t' = y\ OR\ m_{t-2}' = m_t' \\ \left(\bar{\beta}_s(t) + \beta_{s-1}(t-2)\right) b_{1_t'}^s & OTHERWISE \end{cases} \tag{9}$$

Where,

$$\bar{\beta}_s(t) = \beta_{s-1}(t) + \beta_{s-1}(t-1) \tag{10}$$

The probability of the annotated sequence m is the sum of the probabilities of m', as expressed in the following formula:

$$o(m \mid a) = \beta_S\left(\left|m'\right|\right) + \beta_S\left(\left|m'\right| - 1\right) \tag{11}$$

Similarly, the backward variable $\alpha_s(t)$ represents the total probability of the sequence $m_{t:|1|}$ at time s, which is expressed as:

$$\alpha_s(t) = \sum_{\tau:\alpha(\tau_{s:S})=1_{t:|1|}} \prod_{s'=s}^{S} b_{\tau_{s'}}^{s'} \tag{12}$$

The initialization was defined as follows:

$$\alpha_S\left(\left|m'\right|\right) = b_y^S \tag{13}$$

$$\alpha_S\left(\left|m'\right| - 1\right) = b_{1_{|1|}}^S \tag{14}$$

$$\alpha_S(t) = 0, \forall t < \left|m'\right| - 1 \tag{15}$$

The recursive relation can be expressed as follows:

$$\alpha_s(t) = \begin{cases} \bar{\alpha}_s(t) b^s_{1'_t} & IF\ m'_t = y\ OR\ m'_{t+2} = m'_t \\ \left(\bar{\alpha}_s(t) + \alpha_{s+1}(t+2)\right) b^s_{1'_s} & OTHERWISE \end{cases} \tag{16}$$

Where,

$$\bar{\alpha}_s(t) = \alpha_{s+1}(t) + \alpha_{s+1}(t+1) \tag{17}$$

Optimizing the model that combines BiLSTM and the CTC criterion is crucial when implementing real-time speech translation systems based on mobile technology. In defining the objective function, the maximum likelihood criterion was employed, where the objective function *O* is defined as the negative log-likelihood of correctly annotating the entire training set. The purpose of this objective function is to minimize the negative log likelihood of the target annotated sequence, thereby improving the model's accuracy. To compute the gradient of the objective function, the forward and backward algorithms are combined. The forward algorithm calculates the probability from the beginning to the current time point, while the backward algorithm computes the probability from the current time point to the end. By combining these two algorithms, the probability of the entire sequence can be obtained. The formula for the objective function is as follows:

$$P = -LN\left(\prod_{(a,c)\in T} o(c \mid a)\right) = -\sum_{(a,c)\in T} LNo(c \mid a) \tag{18}$$

Gradient computation is another essential step. To obtain the gradient of the objective function, derivatives with respect to the network output *P* for some training samples must be computed:

$$\frac{\partial P}{\partial b^s_j} = -\frac{\partial LNo(c \mid a)}{\partial b^s_j} = -\frac{1}{o(c \mid a)} \frac{\partial o(c \mid a)}{\partial b^s_j} \tag{19}$$

Furthermore, the forward-backward algorithm was applied to calculate the expression above:

$$\beta_s(t)\alpha_s(t) = \sum_{\tau \in \alpha^1(c):\tau_s = c'_t} b^s_{c'_t} \prod_{s=1}^{S} b^s_{\tau_s} \tag{20}$$

Substituting into Equation (1), the result is as follows:

$$\frac{\beta_s(t)\alpha_s(t)}{b^s_{c'_t}} = \sum_{\tau \in \alpha^{-1}(c):\tau_s = c'_t} o(\tau \mid a) \tag{21}$$

By combining with Equation (2), the following expression can be obtained:

$$o(c \mid a) = \sum_{t=1}^{|c'|} \frac{\beta_s(t)\alpha_s(t)}{b^s_{c'_t}} \tag{22}$$

To derive with respect to b_j^s, it is necessary to consider the paths passing through label j at time s. The set of positions where j appears is denoted as $LA(c,j)=\left\{t:c_t'=j\right\}$. The derivative of b_j^s with respect to $lLN(o(c|a))$ is then expressed as:

$$\frac{\partial LN\left(o\left(z|a\right)\right)}{\partial b_j^s}=\frac{1}{o\left(z|a\right)}\frac{\partial o\left(z|a\right)}{\partial b_j^s}=\frac{1}{o\left(z|a\right)}\frac{1}{b_j^{s2}}\sum_{t\in LA(c|j)}\beta_s(t)\alpha_s(t) \tag{23}$$

Backpropagation is a critical step in the optimization process. The calculated gradient was propagated back through the softmax layer. First, the derivative of the input x_j^s to the softmax layer was computed. Using the chain rule of the softmax function, the derivative of the objective function P with respect to the input x_j^s to the softmax layer was determined. Next, the standard time-domain backpropagation algorithm was applied, propagating the gradient from the softmax layer back to the BiLSTM layer to update the network weights. In this way, errors were propagated through each layer, and the parameters were adjusted, continuously improving the model's performance. The specific derivative formula is as follows:

$$\frac{\partial P}{\partial x_j^s}=b_j^s\frac{1}{o\left(c|a\right)b_j^s}\sum_{t\in LA(c,j)}\beta_s(t)\alpha_s(t) \tag{24}$$

The derivative result expressed above was utilised for training the network model.

3 PARAMETER COMPRESSION FOR REAL-TIME SPEECH TRANSLATION MODELS IN MOBILE APPLICATIONS

To provide high-quality translation services within the constraints of limited computational resources and battery life, compressing model parameters is essential. This section primarily focuses on compressing the parameters of the BiLSTM-CTC model to meet the requirements of real-time processing and low power consumption in mobile applications. The BiLSTM-CTC model was selected as the foundational architecture for speech translation due to its ability to handle variable-length input sequences and its use of BiLSTM layers to capture contextual information, which is crucial for both speech recognition and translation tasks. The preprocessing steps involve extracting features from the speech signal and feeding the feature sequence into the BiLSTM network. During the training process, the CTC loss function was employed to align the input and output sequences, addressing the alignment issue found in traditional sequence-to-sequence models. The model was trained to learn how to map input speech feature sequences to the target text sequences while optimizing the network parameters. For a single-layer BiLSTM, assuming the weight connection matrix for modelling forward information is denoted by $\vec{Q}_*$, and the weight connection matrix for modelling backward information is represented by $\bar{Q}_*$, the calculation formula is as follows:

$$\vec{u}_s=\delta\left(\vec{Q}_{au}a_s+\vec{Q}_{lu}\vec{al}_{s-1}+\vec{Q}_{zu}\vec{xz}_{s-1}+\vec{y}_u\right) \tag{25}$$

$$\bar{u}_s=\delta\left(\bar{Q}_{au}a_s+\bar{Q}_{lu}\bar{al}_{s-1}+\bar{Q}_{zu}\bar{xz}_{s-1}+\bar{y}_u\right) \tag{26}$$

$$\vec{d}_s=\delta\left(\vec{Q}_{ad}a_s+\vec{Q}_{ld}\vec{l}_{s-1}+\vec{Q}_{zd}\vec{z}_{s-1}+\vec{y}_d\right) \tag{27}$$

$$\overleftarrow{d}_s = \delta\left(\overleftarrow{Q}_{ad} a_s + \overleftarrow{Q}_{ld} \overleftarrow{l}_{s-1} + \overleftarrow{Q}_{zd} \overleftarrow{z}_{s-1} + \overleftarrow{y}_d\right) \tag{28}$$

$$\vec{z}_s = d_s \Phi \vec{z}_{s-1} + \vec{u} s \Phi h\left(\vec{Q}_{az} a_s + \vec{Q}_{lz} \vec{l}_{s-1} + \vec{y}_z\right) \tag{29}$$

$$\overleftarrow{z}_s = d_s \Phi \overleftarrow{z}_{s-1} + \overleftarrow{u} s \Phi h\left(\overleftarrow{Q}_{az} a_s + \overleftarrow{Q}_{lz} \overleftarrow{l}_{s-1} + \overleftarrow{y}_z\right) \tag{30}$$

$$\vec{p}_s = \delta\left(\vec{Q}_{ap} a_s + \vec{Q}_{lp} \vec{l}_{s-1} + \vec{Q}_{zp} \vec{z}_s + \vec{y}_p\right) \tag{31}$$

$$\overleftarrow{p}_s = \delta\left(\overleftarrow{Q}_{ap} a_s + \overleftarrow{Q}_{lp} \overleftarrow{l}_{s-1} + \overleftarrow{Q}_{zp} \overleftarrow{z}_s + \overleftarrow{y}_p\right) \tag{32}$$

$$\vec{l}_s = \vec{p}_s \Phi g\left(\vec{z}_s\right) \tag{33}$$

$$\overleftarrow{l}_s = \overleftarrow{p}_s \Phi g\left(\overleftarrow{z}_s\right) \tag{34}$$

$$b_s = \varphi\left(\vec{Q}_{lb} \vec{l}_s + \overleftarrow{Q}_{lb} \overleftarrow{l}_s + y_b\right) \tag{35}$$

To compress the model, a smoothing gate mechanism was introduced to evaluate and prune less important memory units. In the BiLSTM network, each memory unit is controlled by multiple gates. By analysing the outputs of these control gates, such as the input gate, forget gate, and output gate, the importance of each memory unit can be assessed. Specifically, the smoothing gate information was computed for each control gate, which helps in determining which memory units contribute less to the final output. Both the forward and backward memory units were evaluated for their importance. By utilizing the smoothing gate information from control gates, such as the activation values of the input and forget gates, the importance of each memory unit was further quantified. Memory units that were assessed to be of lower importance were pruned, reducing the overall parameter count of the model. This pruning process was conducted dynamically during training to ensure that the pruned model still maintains high translation accuracy. The following equations provide the update formulas for the smoothing gate information in the hidden layer of the forward sublayer:

$$\vec{i}_u = \beta \vec{u}_u + \alpha \vec{u}_s, \vec{u}_u = \vec{i}_u \tag{36}$$

$$\vec{i}_d = \beta \vec{u}_d + \alpha \vec{f}_s, \vec{u}_d = \vec{i}_d \tag{37}$$

$$\vec{i}_u = \beta \vec{u}_u + \alpha \vec{p}_s, \vec{u}_p = \vec{i}_p \tag{38}$$

Assuming the initial value of the smoothing gate information is denoted by u_s, the corresponding update formulas for the backward sublayer are as follows:

$$\overleftarrow{i}_u = \beta \overleftarrow{u}_u + \alpha \overleftarrow{u}_s, \overleftarrow{u}_u = \overleftarrow{i}_u \tag{39}$$

$$\overleftarrow{i}_d = \beta \overleftarrow{u}_d + \alpha \overleftarrow{d}_s, \overleftarrow{u}_d = \overleftarrow{i}_d \tag{40}$$

$$\overleftarrow{i}_u = \beta \overleftarrow{u}_u + \alpha \overleftarrow{p}_s, \overleftarrow{u}_p = \overleftarrow{i}_p \tag{41}$$

When information from two or three control gates is used simultaneously, the corresponding smoothing gate update formula is given by the following equations:

$$\vec{i}_{IF} = \beta \vec{u}_{IF} + \alpha\left(\vec{u}_s + \vec{d}_s\right)/2, \vec{u}_{IF} = \vec{i}_{IF} \tag{42}$$

$$\overleftarrow{i}_{IF} = \beta \overleftarrow{u}_{IF} + \alpha\left(\overleftarrow{u}_s + \overleftarrow{d}_s\right)/2, \overleftarrow{u}_{IF} = \overleftarrow{i}_{IF} \tag{43}$$

After pruning, the model's parameter count was significantly reduced. Next, the compressed model was further optimized by retraining or fine-tuning, ensuring that its performance is restored or improved. This step guarantees that the model, while reducing computational resource consumption, does not suffer a significant loss in translation quality. The optimized BiLSTM-CTC model was then deployed on mobile devices. During deployment, considerations must be made for the real-time inference of the model and resource consumption. By leveraging the hardware acceleration capabilities of mobile devices and combining them with an efficient inference framework, highly efficient real-time speech translation can be achieved.

4 EXPERIMENTAL RESULTS AND DISCUSSION

As shown in Table 1, there is a significant difference in parameter complexity and performance between the baseline model and the model that incorporates the input, forget, and output gates. Specifically, the baseline model contains 10.89 million (M) parameters and requires 1.10×10^7 multiplication operations, whereas the model incorporating smoothing gates reduces the parameter count to 6.36 million and the number of multiplication operations to 6.36×10^6. Despite the substantial reduction in parameters and computation, the translation performance, as measured by word error rate (WER), shows minimal change. The baseline model's WER is 14.25%/14.18%, while the optimized model achieves a WER of 14.22%/14.09%. This indicates that the introduction of smoothing gates (input gate, forget gate, and output gate) reduced the model's parameters by approximately 42% and the multiplication operations by around 42%, a particularly important improvement for mobile applications. In mobile devices, computational and storage resources are often limited. Thus, the reduction in parameters and computational complexity can significantly improve the efficiency of real-time speech translation systems, extend battery life, and reduce processing latency. Despite the substantial decrease in model parameters and operations, translation performance (as measured by WER) did not degrade significantly. This demonstrates that the model with smoothing gates can achieve compression while maintaining translation quality. For mobile applications, where user experience relies heavily on high-quality translation results, the ability to reduce resource consumption without compromising performance is of significant practical value.

Table 1. Parameter complexity of the compressed model for real-time speech translation in mobile environments

Smoothing Gate	Model Parameters	Multiplication Operations	WER
Baseline	10.89 M	1.10E7	14.25%/14.18%
Input gate, forget gate, and output gate	6.36 M	6.36E6	14.22%/14.09%

As seen in Table 2, there are significant differences in parameter count and translation performance (WER%) across models with different architectures. The model in Experiment 2 achieved a parameter reduction of over 50%, from 44.67 M to

21.45 M, with only a slight performance drop (WER 14.5%) compared to the baseline model. Similarly, the model in Experiment 4 demonstrated a considerable parameter reduction (28.46 M) while maintaining a WER of 14.3%, close to the baseline performance. In contrast, the model in Experiment 3, although also reducing the parameter count (37.98 M), achieved a WER of 14.2%, slightly outperforming the baseline. Despite the significant parameter reduction in Experiments 2 and 4, their translation performance remained close to that of the baseline model, indicating that adjusting the model architecture—such as reducing the number of nodes per hidden layer—can effectively lower model complexity while preserving translation quality. This finding is particularly important for real-time speech translation systems on mobile devices, where computational and storage resources are limited.

Table 2. Structure and parameter count of compressed models for real-time speech translation in mobile environments

Experiment ID	Number of Nodes per Hidden Layer	Parameter Count	WER% (tg)
1	6 hidden layers, 2,048 nodes per layer, using the ReLU activation function	44.67 M	14.4
2	1180-1022-1343-1222-1347-1349	21.45 M	14.5
3	1811-1885-1851-1825-1872-1844	37.98 M	14.2
4	1467-1552-1634-1550-1652-1644	28.46 M	14.3

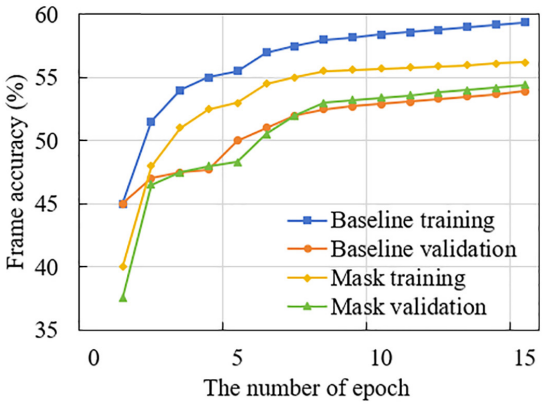

Fig. 4. Frame accuracy on the training and validation sets

The data in Figure 4, which includes baseline training, baseline validation, mask training, and mask validation, illustrate the changes in frame accuracy on the training and validation sets as the number of training epochs increase. It can be observed that both the training and validation accuracy of the masked model progressively improve, eventually matching or even surpassing the baseline model in later epochs. This demonstrates that the model employing the masking strategy is capable of achieving performance similar to or better than the baseline model after sufficient training. This is particularly critical for mobile applications, as the masking strategy is often utilized for model compression and optimization, thereby reducing the demand for computational and storage resources. Both the baseline and masked models show good convergence in terms of frame accuracy on the training and validation sets. By the 15th epoch, the training accuracy of the baseline model reaches 59.4, while the masked model's training accuracy is slightly lower at 56.2. However, the masked model exhibits higher accuracy on the validation set, indicating stronger generalization ability in practical applications. This confirms the

effectiveness of the masking strategy in mobile environments, as it can significantly reduce model complexity while maintaining high translation quality.

Table 3. ACC of compressed models for real-time speech translation in mobile environments

	Configuration	Number of Filters	ACC% (Single-Pass Decoding)	ACC% (Two-Pass Decoding)
Baseline	–	13,568	91.12	91.88
Threshold of 0.15	Continue training after pruning with an unchanged learning rate.	9,362 (69%)	90.52	91.87
	Continue training after pruning with doubled learning rate.		91.12	91.28
Threshold of 0.20	Continue training after pruning with an unchanged learning rate.	8,875 (65.4%)	91.25	91.68
	Continue training after pruning with doubled learning rate.		91.02	91.75
init	–		90.79	91.65

Pruning techniques are particularly crucial for mobile devices, as they reduce the demand for hardware resources. The data in Table 3 show that by applying pruning strategies with thresholds of 0.15 or 0.20, the number of filters in the model is significantly reduced (by 31% and 34.6%, respectively), yet the impact on model accuracy is negligible. The accuracy of the pruned models for both single-pass and two-pass decoding remains similar to, or even slightly higher than that of the baseline model. When the models are retrained after pruning, doubling the learning rate shows a positive effect on performance. For the pruning strategy with a threshold of 0.15, the accuracy of single-pass decoding improves from 90.52% to 91.12%, while two-pass decoding accuracy rises from 91.87% to 91.28%. For the threshold of 0.20, although the single-pass decoding accuracy slightly decreases with a doubled learning rate, the two-pass decoding performance remains stable. These findings suggest that an appropriate adjustment of the learning rate can enhance the performance of pruned models, bringing them close to or even above baseline levels. With the 0.20 threshold pruning strategy, the number of filters was reduced by 34.6%, and the accuracy for single-pass decoding exceeded that of the baseline model, reaching 91.25%. This demonstrates that pruning strategies can effectively eliminate redundant parameters while maintaining or even improving model performance. Considering the resource constraints of mobile devices, this pruning technique offers a significant reduction in model complexity without a notable loss in accuracy, making it highly suitable for mobile environments.

Table 4. ACC of different compressed models for real-time speech translation systems

	Parameter Count	ACC% (Single-Pass Decoding)	ACC% (Two-Pass Decoding)
Baseline	46.87 M	91.12	91.87
MobileNet	31.26 M	90.87	91.75
	10.45 M	90.36	91.35
Proposed model	28.89 M	91.48	91.89

Table 4 summarizes the parameter counts and corresponding accuracy (ACC%) of different compressed models in real-time speech translation systems. The results indicate that while various compression methods reduce the parameter count, their impact on model accuracy varies. The baseline model has the largest parameter count but also achieves the highest accuracy. MobileNet, a lightweight network architecture, reduces computational complexity by lowering the parameter count. At 31.26 M and 10.45 M parameters, the accuracy for both single-pass and two-pass decoding shows a slight decrease compared to the baseline model, though the overall performance remains close. The compression method proposed in this study shows a clear advantage. With a parameter count reduced to 28.89 M, the accuracy for single-pass decoding is 91.48%, and for two-pass decoding, it reaches 91.89%. These results not only exceed the accuracy of the baseline model but also significantly reduce the parameter count. This demonstrates that the proposed method achieves effective model compression while maintaining high accuracy, making it particularly suitable for resource-constrained mobile device applications.

5 CONCLUSION

This study successfully addressed the challenges of efficiency and accuracy in real-time speech translation models for mobile applications by employing a compression method based on the CTC criterion. Compared to traditional large-scale models, the proposed method reduced the parameter count by approximately 38% while maintaining high-quality translation output (with ACC reaching 91.48% and 91.89%). This demonstrates that the approach holds significant practical value for resource-constrained real-time speech translation applications on mobile devices. By reducing the model parameters, computational and storage resource demands were significantly lowered, enabling complex speech translation tasks to be effectively executed in resource-limited environments such as mobile devices. Furthermore, the CTC-based model compression method successfully minimized redundant parameters without compromising accuracy, providing a new optimization strategy for future speech translation systems.

This study offers substantial practical application value, particularly in the feasibility of mobile device deployment and the development of efficient model compression techniques. By significantly reducing model parameters, this method enables efficient real-time speech translation on mobile devices, drastically reducing computational and storage demands. This ensures that complex translation tasks can be performed smoothly in constrained environments. The CTC-based compression approach maximizes the reduction of redundant parameters while preserving translation quality, offering a promising avenue for future optimizations with broad application potential.

6 REFERENCES

[1] M. Kabát, “Possible translation problems, their causes, and solutions in agile localization of software,” *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 17, no. 1, pp. 129–140, 2023. https://doi.org/10.3991/ijim.v17i01.36367

[2] X. Lei, “Real-time translation of English speech through speech feature extraction,” *Artificial Life and Robotics*, vol. 29, pp. 410–415, 2024. https://doi.org/10.1007/s10015-024-00951-w

[3] L. M. Rababah, N. Al-Khawaldeh, and M. A. Rababah, "Mobile-assisted listening instructions with Jordanian audio materials: A pathway to EFL proficiency," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 17, no. 21, pp. 129–144, 2023. https://doi.org/10.3991/ijim.v17i21.42789

[4] X. Ke, "English synchronous real-time translation method based on reinforcement learning," *Wireless Networks*, vol. 30, pp. 4167–4179, 2024. https://doi.org/10.1007/s11276-022-02910-4

[5] Z. Kozhirbayev, "Enhancing neural machine translation with fine-tuned mBART50 pre-trained model: An examination with low-resource translation pairs," *Ingénierie des Systèmes d'Information*, vol. 29, no. 3, pp. 831–838, 2024. https://doi.org/10.18280/isi.290304

[6] H. K. Chen, "Cognitive perspectives on English learning methods: Efficiency and achievements under task-based instruction," *Education Science and Management*, vol. 1, no. 2, pp. 86–100, 2023. https://doi.org/10.56578/esm010203

[7] S. Novitasari, S. Sakti, and S. Nakamura, "Neural incremental speech recognition toward real-time machine speech translation," *IEICE Transactions on Information and Systems*, vol. E104.D, no. 12, pp. 2195–2208, 2021. https://doi.org/10.1587/transinf.2021EDP7014

[8] V. R. Tripathi, H. K. Jha, M. Popli, P. Shah, and G. Desai, "Clinic, community, and in-between: The influence of space on real-time translation of medical expertise by frontline healthcare professionals in marginal tribal communities," *Journal of Professions and Organization*, vol. 8, no. 3, pp. 273–294, 2021. https://doi.org/10.1093/jpo/joab012

[9] A. Kumar, A. Pratap, and A. K. Singh, "Generative adversarial neural machine translation for phonetic languages via reinforcement learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 1, pp. 190–199, 2022. https://doi.org/10.1109/TETCI.2022.3209394

[10] A. Bilas, "Psycholinguistic aspect of phonetic and orthographic means of French colloquial speech in Ukrainian translation," *Psycholinguistics*, vol. 27, no. 2, pp. 71–89, 2020. https://doi.org/10.31470/2309-1797-2020-27-2-71-89

[11] X. Feng, Z. Feng, W. Zhao, B. Qin, and T. Liu, "Enhanced neural machine translation by joint decoding with word and POS-tagging sequences," *Mobile Networks and Applications*, vol. 25, pp. 1722–1728, 2020. https://doi.org/10.1007/s11036-020-01582-8

[12] H. Alotaibi and D. Salamah, "The impact of translation apps on translation students' performance," *Education and Information Technologies*, vol. 28, pp. 10709–10729, 2023. https://doi.org/10.1007/s10639-023-11578-y

[13] Z. X. Tan, Z. Y. Yang, and Y. Liu, "Dynamic multi-branch layers for on-device neural machine translation," *IEEE-ACM Transactions on Audio Speech and Language Processing*, vol. 30, pp. 958–967, 2022. https://doi.org/10.1109/TASLP.2022.3153257

[14] M. Li, J. Pang, F. Yue, F. Liu, J. Wang, and J. Tan, "Enhancing dynamic binary translation in mobile computing by leveraging polyhedral optimization," *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, pp. 1–12, 2021. https://doi.org/10.1155/2021/6611867

[15] M. Jonathan and E. Rakun, "Translating SIBI (Sign System for Indonesian Gesture) gesture-to-text in real-time using a mobile device," *Journal of ICT Research & Applications*, vol. 16, no. 3, pp. 259–280, 2022. https://doi.org/10.5614/itbj.ict.res.appl.2022.16.3.5

[16] A. Panayiotou *et al.*, "The perceptions of translation apps for everyday health care in healthcare workers and older people: A multi-method study," *Journal of Clinical Nursing*, vol. 29, nos. 17–18, pp. 3516–3526, 2020. https://doi.org/10.1111/jocn.15390

[17] F. Wang, Q. Wang, and C. Du, "WeChat-based interactive translation mobile teaching model," *Mobile Information Systems*, vol. 2021, no. 1, pp. 1–9, 2021. https://doi.org/10.1155/2021/7054016

[18] E. Sappey-Marinier *et al.*, "No difference in patellar position between mobile-bearing and fixed-bearing total knee arthroplasty for medial osteoarthritis: A prospective randomized study," *Knee Surgery, Sports Traumatology, Arthroscopy*, vol. 28, pp. 1542–1550, 2020. https://doi.org/10.1007/s00167-019-05565-5

[19] S. Sviķe, "Mobile apps as language-learning tools: Challenges, problems and solutions of specialised lexicography," *AILA Review*, vol. 34, no. 1, pp. 19–36, 2021. https://doi.org/10.1075/aila.20006.svi

[20] Y. Wang, "Artificial Intelligence technologies in college English translation teaching," *Journal of Psycholinguistic Research*, vol. 52, pp. 1525–1544, 2023. https://doi.org/10.1007/s10936-023-09960-5

[21] T. Tsushima *et al.*, "Mobile bearing shows larger rollback motion than fixed bearing in total knee arthroplasty using a medial stabilising technique with a navigation system," *Journal of Experimental Orthopaedics*, vol. 11, no. 3, pp. 1–9, 2024. https://doi.org/10.1002/jeo2.12053

7 AUTHOR

Ning Yang did Master's degree from Northwest University. Now she works in School of Education, Xi'an Fanyi University. Her research interests include English Translation and Interpreting and English Education (E-mail: yn@xafy.edu.cn).