Code-switching Mediated Sentence-level Semantic Learning

Shuai Zhang[1], Jiangyan Yi[1], Zhengqi Wen[1], Jianhua Tao [1*], Feihu Che [1*], Jinyang Wu[1], Ruibo Fu[2]

[1]Department of Automation & BNRist, Tsinghua University
[2]Institute of Automation, Chinese Academy of Sciences
{zhang_shuai, yijy, zqwen, jhtao, qkr, wu-jy23}@mail.tsinghua.edu.cn, ruibo.fu@nlpr.ia.ac.cn

Abstract

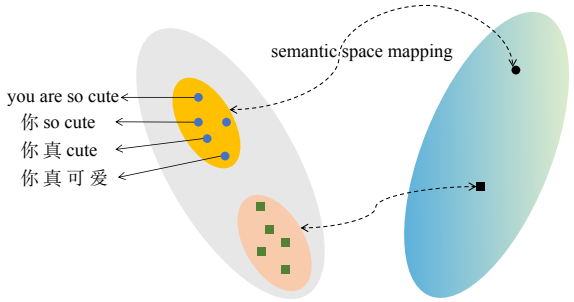
Code-switching is a linguistic phenomenon in which different languages are used interactively during conversation. It poses significant performance challenges to natural language processing (NLP) tasks due to the often monolingual nature of the underlying system. We focus on sentence-level semantic associations between the different code-switching expressions. And we propose an innovative task-free semantic learning method based on the semantic property. Specifically, there are many different ways of languages switching for a sentence with the same meaning. We refine this into a semantic computational method by designing the loss of semantic invariant constraint during the model optimization. In this work, we conduct thorough experiments on speech recognition, speech translation, and language modeling tasks. The experimental results fully demonstrate that the proposed method can widely improve the performance of code-switching related tasks.

Introduction

Code-switching is a common linguistic phenomenon in which several languages are used interactively during conversation (Poplack 1981). The number of multilingual speakers far outnumbers monolingual speakers in the worldwide population (Tucker 2003; Winata et al. 2021). Code-switching expressions are widely used in a variety of scenarios, including and not limited to daily conversations, classroom teaching, conferences, social media, etc. It is a strong incentive to develop technologies that can handle code-switching efficiently. However, progress in this area has been limited, primarily since code-switching typically occurs during informal expressions, such as spoken language, where real-time data collection is difficult (Sitaram et al. 2019; Jose et al. 2020; Doruz et al. 2021). The significant increase of smart speech devices has alleviated this problem, while new technologies are developed to fulfill users' needs for multilingual interaction.

Code-switching computation research focuses on two main areas: speech processing and text processing. Automatic speech recognition (ASR) and speech synthesis are the most widely researched tasks in speech processing (Graves et al. 2006; Graves, Mohamed, and Hinton 2013; Jia et al. 2019; Sperber et al. 2019; Bérard et al. 2016), and the most interesting tasks for researchers in text processing are language identification, sentiment analysis, and language modeling, with some other research in text classification, question answering, and sequence labeling (Solorio et al. 2014; Molina et al. 2019; Patra, Das, and Das 2018; Qin et al. 2020; Zheng et al. 2021). This strong correlation between task and venue shows that the speech-processing and text-processing communities remain somewhat fragmented and tend to work in isolation from each other. Such research isolation limits the study of code-switching and hinders the ability to draw more generalized task-independent insights (Napoli et al. 2014).

This study aims to explore the semantic association across code-switching expressions and develop a novel generalized semantic learning method for various tasks. Specifically, the process of code-switching occurs with a certain degree of randomness, influenced by linguistic rules, social psychology, and other factors (Poplack 1981). There are multiple legitimate textual candidates for each position in a sentence, leading to many different expression forms of the same



(a) code-switching formal space (b) sentence-level semantic space

Figure 1: Schematic illustration of the correspondence between code-switching expressions and semantics. (a) represents the code-switching formal space, where each point represents a language mixing modality and similar expressions are distributed in adjacent regions. (b) represents the sentence-level semantic space, each dot denotes a sentence semantic, with correspondences to the many different forms of the left diagram.

*: corresponding author

sentence-level semantic. As shown in Figure 1, each dot in the left graph represents a code-switching formal, and multiple dots together correspond to the same semantic in the right semantic space. We refine this into an explicit semantic computational method by designing the loss of semantic invariant metrics across sentences. To verify the effectiveness and generalization of the proposed method, we conduct experiments on typical research tasks in the fields of speech and text processing, including ASR, automatic speech translation (AST), and language modeling. We also explore the capability boundaries of the proposed method by examining the generative capabilities of large language models (LLM) for code-switching. The results indicate a significant improvement in performance for each task. Additionally, we demonstrate the effectiveness of the proposed method in semantic modeling through visual qualitative analysis of the samples. This indicates that the methodology presented in this paper can provide a novel perspectives on the study of code-switching computational methods, which can benefit a wide range of related research tasks.

Our main contributions are summarized as follows:

- We utilize the semantic properties of code-switching to achieve a task-free approach to semantic learning, with implications for related research in both speech and text domains.
- We use code-switching as a mediator to design task-related prompts for efficient unified modeling of ASR and AST tasks.
- Detailed experimental results on a variety of different tasks with careful analysis prove that our method significantly outperforms the baseline model and some existing methods in terms of performance and semantic modeling ability.

Related Work

ASR refers to the transcription of code-switched speech into corresponding text, and AST refers to the direct translation of speech into another language. Recently, the end-to-end model has attracted attention in the two fields for its extremely simplified architecture without complicated pipeline systems (Graves et al. 2006; Graves, Mohamed, and Hinton 2013; Jia et al. 2019; Sperber et al. 2019; Bérard et al. 2016). Some work applies multi-task learning to train AST and ASR task jointly (Weiss et al. 2017; Anastasopoulos and Chiang 2018; Berard et al. 2018; Vydana et al. 2021; Nakayama et al. 2019). Some work uses semantic information to improve the quality of AST (Dong et al. 2021a,b). The semantic information usually comes from two aspects, one is pre-trained models (Dong et al. 2021a), such as BERT (Kenton and Toutanova 2019), and the other is from acoustic features (Dong et al. 2021a). However, these methods do not establish semantic associations between ASR and AST, making it difficult to achieve efficient unified modeling and limiting further performance improvements.

Semantic Information for Code-switching

Most research on extracting semantic information for code-switching involves transforming linguistic theories into computable forms to improve code-switching-related tasks (Qin et al. 2020; Zheng et al. 2021; Li and Fung 2013, 2014). Another approach is to use code-switching as a data augmentation method to enhance the performance of the multilingual model (Qin et al. 2020). However, this approach does not explicitly model the semantic relationships between code-switching expressions. (Zheng et al. 2021) notes semantic associations, however, it is still considered as a data augmentation method to enhance text tasks that do not involve cross-modal code-switching task. It lacks systematic observation and validation of code-switching research.

Methodology

Semantic Invariance Constraint

We illustrate the principle of semantic invariance constraint based on ASR and AST tasks. As shown in Figure 2, for a code-switching speech input, three kinds of texts are constructed, corresponding to English AST task, code-switching ASR task, and Chinese AST task. Despite the different format, the three target text clearly express the same semantic. This work extracts semantic representations and implements semantic constraints by measuring the invariance between semantic.

Two methods are employed to extract sentence-level semantic vector, one is performing an average pooling operation on the decoder contextual vectors to obtain the corresponding semantic representation. Another is to add a special symbol [CLS] in the sentence and integrate the semantic information of the whole sentence through the attention mechanism. For example, $<ENG>$ *you are so cute* $[CLS]$.

After obtaining the sentence-level semantic representation of different tasks, we measure the semantic distance between tasks according to the semantic invariance. In order not to lose generality, we describe our approach using the case of CS and the two corresponding monolinguals. The semantic invariance loss can be expressed as follows,

$$\mathcal{L}_{sil}(\theta) = \mathcal{D}(\theta; \mathbf{s}_A, \mathbf{s}_E) + \mathcal{D}(\theta; \mathbf{s}_A, \mathbf{s}_C) + \mathcal{D}(\theta; \mathbf{s}_C, \mathbf{s}_E) \quad (1)$$

where $\mathcal{L}_{sil}$ refers to the total semantic invariance loss, θ refers to the model parameter, $\mathcal{D}$ refers to the distance calculator between semantic representation, $\mathbf{s}_A$,$\mathbf{s}_E$,$\mathbf{s}_C$ refer to semantic vectors of ASR, English AST, Chinese AST respectively.

Model Details

Problem Formulation The data used in this paper contain speech-transcription-translations quadruples, denoted as $S = (\mathbf{x}, \mathbf{z}, \mathbf{e}, \mathbf{c})$. Specially, $\mathbf{x} = (x_1, ..., x_{T_x})$, $\mathbf{z} = (z_1, ..., z_{T_z})$, $\mathbf{e} = (e_1, ..., e_{T_e})$, $\mathbf{c} = (c_1, ..., c_{T_c})$ represent the acoustic features sequence, the corresponding transcription, the translation of English and the translation of Chinese respectively. And the T_x is the frame number of the speech sequence. The T_z, T_e, T_c are the lengths of the above three target sequences. The goal is to model the all three target sequences simultaneously $(\mathbf{z}, \mathbf{e}, \mathbf{c})$ based on the acoustic features $\mathbf{x}$.

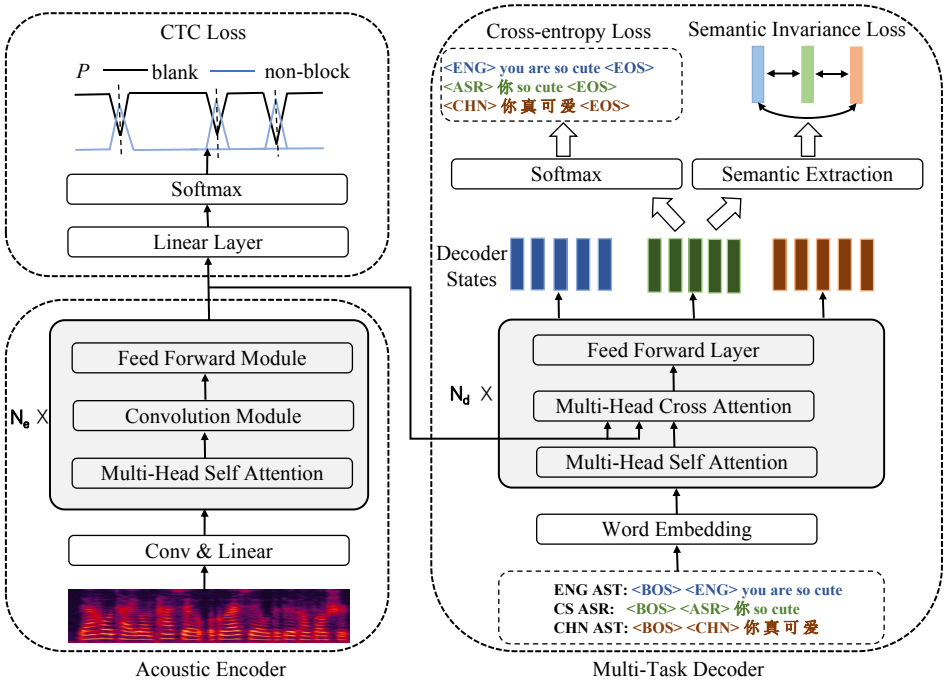


Figure 2: The model architecture of unified ASR and AST task learning. The left part is the acoustic encoder which takes acoustic features as input. Its main component is a convolution enhanced transformer structure for efficient encoding of acoustic feature. The right part is the multi-task decoder. It receives text input for the three tasks and extracts information from the acoustic encoder while modeling ASR and AST.

Model Components In this section, we illustrate the structure of our model and how it deal with three different tasks simultaneously. As shown in Figure 2, the overall architecture of the model consists of two modules: a) an acoustic encoder network that encodes the speech features sequence into a high-level hidden representation; b) a multi-mask decoder receives text input for the three tasks and extracts information from the acoustic encoder while modeling ASR and AST. One can freely choose the structure of the encoder and decoder, such as transformer network, recurrent neural network, convolution network, and so on. We adopt transformer as the backbone network. It is now the state-of-the-art model in the translation task, and it also shows excellent performance in the ASR field. For details of the model, please refer to (Gulati et al. 2020).

Acoustic Encoder. The acoustic encoder receives the input of low-level acoustic features and outputs the high-level hidden representation. It is based on the conformer, a convolution-augmented transformer structure. Since the number of acoustic feature frames is much larger than the length of the corresponding text, the down-sampling technique is essential. We adopt the 2D CNN layer to produce the down-sampled acoustic hidden representation. After a linear layer, the positional encoding is used to attend relative positions. Then a stack of N_e conformer blocks is used to get the final encoded representation. Each conformer module mainly includes three modules, which are multi-head self-attention module, convolution module, and feed-forward module in sequence. Compared with the classic transformer structure, it adds a convolution module to extract local information in acoustic encoding.

Multi-Task Decoder. For the decoder, a learnable word embedding and positional encoding are applied to the target sequence. Then a stack of N_d decoder blocks is subsequent. The decoder mainly consists of three parts: multi-head self-attention, multi-head cross-attention, and feed forward network. The multi-head self-attention is used to encode multi-task input text to obtain high-dimensional encoding representation. The multi-head cross-attention takes the high-dimensional representation as the query vectors and performs cross-attention computation on the output vectors of the acoustic encoder to get the contextual vectors. For the self-attention, the query, key, and value are the target text embedding. For the multi-head cross attention, the key and value come from the encoder outputs and the query comes from the previous sub-block outputs. The feed-forward network performs further encoding on the context vectors, followed by dimensional transformation and softmax to get the final decoder output.

Loss Function

The loss function consists of three parts, including connectionist temporal classification (CTC) loss (Graves et al. 2006), cross-entropy loss, and semantic invariance loss. The cross-entropy loss is the sum of the losses for the three tasks.

$$\mathcal{L}_{ce}(\theta; \mathbf{x}, \mathbf{z}, \mathbf{e}, \mathbf{c}) = \mathcal{L}_{ce}(\theta; \mathbf{x}, \mathbf{z}) + \mathcal{L}_{ce}(\theta; \mathbf{x}, \mathbf{e}) + \mathcal{L}_{ce}(\theta; \mathbf{x}, \mathbf{c}) \quad (2)$$

In this paper, we use Kullback-Leibler divergence (KL) and mean squared error (MSE) to measure the similarity between semantic vectors. Therefore, the overall loss function for end-to-end multi-task training is the weighted sum for the above three parts:

$$\mathcal{L}_{all}(\theta) = \alpha\mathcal{L}_{ce}(\theta) + (1-\alpha)\mathcal{L}_{CTC}(\theta) + \beta\mathcal{L}_{sil}(\theta) \quad (3)$$

where the α is hyper-parameters to balance the cross entropy loss $\mathcal{L}_{ce}(\theta)$ and the CTC loss $\mathcal{L}_{CTC}(\theta)$. The hyper-parameter β is used to adjust the weight of the semantic invariance loss $\mathcal{L}_{sil}(\theta)$ in the total loss.

CTC Auxiliary Module CTC is an alignment-free object function for sequence-to-sequence modeling. It counts all possible output sequence forms corresponding to the input sequence based on the idea of dynamic programming. CTC loss is often used as an auxiliary loss for speech translation tasks.

The loss function directly maximizes the probabilities of the correct label.

$$P(\mathbf{z}|\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{z})} P(\pi|\mathbf{x}) = \sum_{\pi} \prod_{t=1}^{T} P(\pi_t|\mathbf{x}) \quad (4)$$

where $\mathbf{T}$ is frame length of input sequence and $\mathbf{B}$ is a many-to-one mapping $\mathbf{B} : \mathbf{Z} \cup \{blank\} \rightarrow \mathbf{Z}$. $\mathbf{Z}$ is the label unit set. $\mathbf{B}$ indicates the label sequence $\mathbf{y}$ and its corresponding set of CTC paths π. The mapping is by inserting an $\mathbf{blank}$ between each label unit in $\mathbf{y}$. $P(\pi_t|\mathbf{x})$ is estimated from the neural network taking the feature sequence $\mathbf{x}$ as the input. With the conditional independent assumption, $P(\pi|\mathbf{x})$ can be decomposed into a product of posterior $P(\pi_t|\mathbf{x})$ in each frame $\mathbf{t}$. Finally, the CTC loss used in this work is defined as

$$\mathcal{L}_{CTC}(\theta; x, z) = -logP(\mathbf{z}|\mathbf{x}) \quad (5)$$

A linear layer is used to transform the output of the acoustic encoder to the appropriate dimension, and then a softmax layer is used for probability normalization. The computation of the CTC loss is performed using the transformed sequence. After training, the probability values of non-blank units are concentrated in a few spikes, as shown in Figure 2. In this work, we only compute the CTC loss for the task of speech recognition.

Training and Inference

During the training process, the ASR and AST tasks are carried out simultaneously. A batch of training data is randomly sampled, consisting of acoustic features and their corresponding target texts. The forward calculation process is completed for each target text in the batch, followed by uniform gradient back-propagation and parameter updates. To differentiate between tasks, a task ID is added to the beginning of each target text. As shown in Figure 2, the task IDs $<ASR>, <ENG>, <CHN>$ refer to the ASR task, English translation task, and Chinese translation task. These IDs are essential for the unified training of different tasks, as they can bias the same model for different tasks.

data	split	type	hours	language
ASRU 2019	Train	CS	200	Mandarin-English
	Dev	CS	20	
	Test	CS	20	
Fisher	Train	CS	13.28	English-Spanish
		Mono	157.3	
	Dev	Mono	1.45	
	Test	CS	1.63	

Table 1: Code-switching audio data distribution information in each dataset. CS stands for code-switch and Mono for monolingual.

In the inference process, only these three task IDs need to be provided to decode the three target texts simultaneously. The decoding process is made via auto-regressive forms which is same as ordinary end-to-end ASR.

Code-switching Capabilities for LLM The powerful generative capabilities of LLM have recently been impressive, and we utilize LLM to explore the boundaries of the validity of our approach. Specifically, we fine-tune the LLM using code-switching text data, adding semantic invariant loss in the process. The fine-tune data passes through the text corresponding to the speech data used in this work. Specifically, the input is monolingual text and the output is code-switching text. The quality of the code-switching data generated by the LLM was evaluated to judge the effectiveness of our method.

Experiments

Data

We conduct our experiments on three popular publicly available datasets, including the ASRU 2019 Mandarin-English code-switching challenge dataset (Shi, Feng, and Xie 2020), Fisher dataset (Cieri, Miller, and Walker 2004) and TED English-Chinese dataset (Liu et al. 2019). The ASRU 2019 dataset is designed for code-switching ASR task. Although the Fisher dataset is not a code-switching focused dataset, it contains a large amount of (annotated) code-switching utterances. Fortunately, the dataset has a corresponding annotated English translation. The Fisher data consists of three evaluation sets (Dev/Dev2/Test) that together contain approximately a thousand instances of code-switching with corresponding translations in monolingual English. We therefore combined all the code-switching data from the three evaluation sets as a test set. Statistical information on the code-switching dataset is shown in Table 1. However the first two datasets are designed for ASR task and are less often used for AST tasks. Therefore to better compare with other methods on the AST, we conduct experiments on the public TED English-Chinese speech translation dataset. It contains 528 hours of English speech and corresponding annotated Chinese translations.

Data Preprocessing

In this paper, the input acoustic features of the encoder network are a 40-dimensional filter bank with 25ms windowing and 10ms frameshift, which are extended with mean and variance normalization. For all ASR transcription, we remove punctuation and lowercase all English words to keep more consistent with the output of ASR. For the ASRU2019 code-switching challenge dataset, we first use Llama 3 70B to get the corresponding Chinese translation text and then perform a sample-by-sample manual check for corrections. For the Chinese translation, we segment the sentence into characters. We keep about 3500 characters as the modeling units. For the Fisher data, the sentences are encoded using the BPE method, with a shared vocabulary of 2000 subwords. For the TED English-Chinese dataset, the processing method of ASR transcription and translation text is similar to the previous method.

Experimental Results

Evaluation Metrics

For the code-switching ASR task, we use a mix error rate (MER) to evaluate the experimental results of our methods. The MER is defined as the word error rate (WER) for English and the character error rate (CER) for Mandarin. For the English ASR task, the WER is used as the evaluation index. For the Chinese and English translation tasks, we report case-insensitive BLEU (Papineni et al. 2002) scores BLEU_{ci} and character-level BLEU scores BLEU_{cl} respectively.

Experimental Details

All of the models are implemented based on transformer architecture. For the input acoustic features, two $3*3$ 2D CNN down-sampling layers with stride 2 are used. The dimension of the subsequent linear layer is 512. Relative position encoding is used to model position information. The attention dimensions of the encoder and decoder are both 512 and the number of the head is 4. The dimension of position-wise feed-forward networks is 1024. The number of acoustic encoder blocks and decoder blocks are 12 and 6 respectively. To avoid over-fitting, the unified label smoothing technique is used, and the parameter is set to 0.1. SpecAugment with frequency masking (F=30, mF=2) and time masking (T=40, mT=2) is used to improve the performance of the models (Park et al. 2019). Meanwhile, we set the residual dropout as 0.1, where the residual dropout is applied to each sub-block before adding the residual information. We use Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.998, \epsilon = 1e^{-8}$ on 4 NVIDIA A100 GPUs. The batch size is set to 128 during the training process. The learning rate is set by a warm-up strategy. We perform decoding using beam search with a beam size of 10.

Hyper-parameter Selection

There are two hyper-parameters in Equation (4), which are used to balance the weights of the cross-entropy loss, the CTC loss, and the semantic invariance loss. First, reasonable hyper-parameters are determined based on the ASRU 2019 dataset for subsequent experiments. As shown in Table 2, when α is set to 0.7 and β is set to 0.1, both ASR and AST tasks can achieve satisfactory results. All subsequent experiments use these parameter settings.

α	β	ASRU 2019 Dev	
		MER(↓)	BLEU_{cl}(↑)
0.7	0.1	**10.55**	**24.71**
0.7	0.05	10.87	23.85
0.7	0.01	10.96	23.70
0.8	0.1	11.01	24.05
0.8	0.05	11.32	24.11
0.8	0.01	10.79	24.14

Table 2: Effects of hyper-parameters in loss.

model	SEM	SCM	WER(↓)	BLEU_{ci}(↑)
Pretrained	–	–	30.21	25.31
Multi-task	–	–	30.57	25.83
(weller et al)	–	–	30.00	25.60
proposed	$[CLS]$	KL	**28.33**	26.88
	$[CLS]$	MSE	28.72	**27.02**
	ave_pool	KL	28.51	26.85
	ave_pool	MSE	28.49	26.46

Table 3: Results of ASR and AST on Fisher test set. The semantic extraction method and the similarity calculation metric are abbreviated SEM and SCM, respectively.

Favorable Effects on Code-switching ASR

The method is first evaluated on the imbalanced code-switching dataset. Two baselines are used: the first pretrains the AST model using ASR data and then finetunes the model on AST data. The second baseline is a multi-task learning model where the ASR and AST models are jointly trained with independent decoders and a shared acoustic encoder. There are two methods for semantic extraction: the $[CLS]$ method, which is similar to using the BERT model for classification, and the average pooling method $ave\_pooling$. For the semantic similarity calculation, we use KL and MSE. The results show that there is not much difference between the different calculation methods. To be precise, the performance is relatively better when using the $[CLS]$ and the MSE at the same time. Based on these two methods, we conduct the following ASR experiments.

Upon further analysis of the experimental results presented in Table 4, it can be observed that the ASR and AST tasks mutually reinforce each other. This mutual promotion can be attributed to the fact that both tasks share the same semantic space and our unified modeling approach better satisfies this condition than pre-training and multi-task training. Subsequent ablation experiments demonstrate that our method continues to outperform several baseline models even after removing the semantic invariance loss. This shows that our implicit semantic modeling scheme can enhance the performance of both ASR and AST at the same time.

To enhance the method's credibility, we compare the per-

model	SEM	SCM	Dev		Test	
			MER(↓)	BLEU$_{cl}$(↑)	MER(↓)	BLEU$_{cl}$(↑)
Pretrained	–	–	11.53	76.39	11.25	77.11
Multi-task	–	–	11.31	78.73	11.01	78.98
(Lu et al. 2020)	–	–	–	–	11.84	-
(Zhang et al. 2021b)	–	–	11.21	–	10.51	–
(Zhang et al. 2021a)	–	–	12.67	–	11.94	–
(Yan et al. 2021)	–	–	–	–	11.1	–
proposed	$[CLS]$	KL	10.76	**81.72**	10.53	82.31
	$[CLS]$	MSE	**10.55**	81.42	**10.37**	**82.61**
	ave_pool	KL	10.91	81.21	10.61	82.43
	ave_pool	MSE	10.78	81.30	10.51	82.52

Table 4: Results of ASR and AST on ASRU2019 code-switching test and dev sets. Unless otherwise noted, Dev and Test in all tables below belong to this dataset. The semantic extraction method and the similarity calculation metric are abbreviated SEM and SCM, respectively.

model	SEM	SCM	Enc Pretrain	Dec Pretrain	WER(↓)	BLEU$_{cl}$(↑)
Transformer+pretrain (Liu et al. 2019)	–	–			–	16.80
+ knowledge distillation (Liu et al. 2019)	–	–			–	19.55
Multi-task+pretrain (Inaguma et al. 2019)	–	–		✗	–	20.45
Interactive decoding(Liu et al. 2020)	–	–	✗	✗	13.38	**21.68**
COSTT without pretraining(Dong et al. 2021a)	–	–	✗	✗	–	21.12
proposed methods	$[CLS]$	KL	✗	✗	11.35	**22.11**
	$[CLS]$	MSE	✗	✗	**11.19**	21.50
	ave_pool	KL	✗	✗	12.12	21.32
	ave_pool	MSE	✗	✗	12.05	21.36

Table 5: Results of ASR and AST on TED English-Chinese test set.

formance of code-switching ASR with other existing research results. This comparison is conducted under the same training data conditions, using only code-switching data from the dataset. The results in Table 3 and Table 4 demonstrate that our method outperforms others under the same training data conditions. As shown in Table 5, we achieve the state-of-the-art recognition performance on the TED dataset.

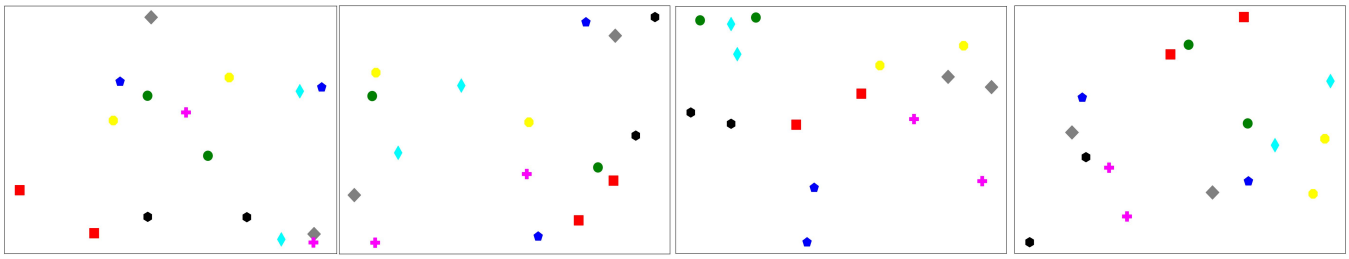
Favorable Effects on Code-switching AST

The semantic enhancement for speech translation is even more obvious. As shown in Table 4, our method has a significant improvement over the strong baseline. And the high BLEU$_{cl}$ score for the Chinese translation task is due to the dominance of Chinese data. This indicates that the task of Chinese translation is easier, which is closely related to Chinese speech recognition. The consistency improvement of our approach can be seen in Table 3. The difference is that the metrics are a bit lower compared to Table 4, which is due to the more balanced language distribution of this dataset, which enhances the difficulty of the speech translation task. Overall, our proposed method outperforms the baseline model in all metrics.

As the above two code-switching datasets are intended for ASR, it is not possible to compare the performance of AST with other existing methods. Therefore, we conducted experiments on the TED English-Chinese dataset, which is designed for AST. Table 5 presents a comparison with existing studies. Only a few research works provide error rate metrics for ASR, and we achieve a relative performance improvement of 16.37%. Our method achieves better performance than other methods on the AST and the experimental results demonstrate its effectiveness.

Code-switching Generation of LLM

We evaluate the proposed method using two LLM at different scales, llama2-7b and llama2-13b (Touvron and et al 2023). The LLM is fine-tuned using code-switching text data, with the addition of semantic invariant loss. One hundred pieces of code-switching data were generated based on the one hundred monolingual data prompts provided. The validity of the data was determined through manual evaluation. Table 7 demonstrates that the LLM's ability to generate legitimate code-switching data is weak. However, fine-tuning the code-switching data can effectively improve this ability. Our method is equally effective on the LLM.



(a) pretrained model (b) Multi-task model (c) Our model (d) No code-switching

Figure 3: Sample visualizations of different methods. From left to right the pretrained model, the multi-task model, and our model. No code-switching refers to samples from TED English-Chinese dataset.

Metrics	Dev of ASRU2019	Test of ASRU2019	Test of Fisher	Test set of TED
MER(↓)	10.55/10.91	10.37/10.77	28.72/29.33	11.19/12.32
BLEU_{ci}(↑)	–/–	–/–	27.02/25.56	–/–
BLEU_{cl}(↑)	81.42/80.37	82.61/80.81	–/–	21.50/21.07

Table 6: Ablation experimental results of semantic invariance loss. (with/without)

model	llama2-7B	llama2-13B
No finetune	23%	44%
supervised finetune	28%	53%
proposed	33%	59%

Table 7: Qualification rate (%) of the code-switching data generated by LLM.

Effect of Semantic Invariance Loss

To assess the impact of semantic invariance loss in our approach, we conducted ablation experiments to analyze the results. Table 6 presents the experimental outcomes with and without semantic invariance loss. Overall, the inclusion of semantic invariance loss is beneficial for both ASR and AST tasks. Notably, this loss has a greater impact on AST than ASR, possibly due to the greater importance of semantic information in AST tasks. Furthermore, our proposed unified modeling approach achieves superior performance compared to baseline methods, even without the semantic invariance loss. The results obtained from the TED English-Chinese dataset also demonstrate highly competitive performance when compared to other existing methods.

Semantic Visualization

To demonstrate more intuitively the semantic modeling capabilities of our method, we visualize word embedding representations in different languages. We use the t-SNE toolkit (Van der Maaten and Hinton 2008) to realize the dimension reduction operation of word embedding. Obviously, it can be seen that the semantic distribution of the pre-trained model is very chaotic due to the lack of semantic modeling constraints. The semantic distribution of the multitask model is relatively regular, but most of the word pairs are still far apart. In Figure 3(c), we can intuitively observe that the distance between the synonym pairs is closer. Our method can effectively learn semantic information by sharing the semantic space and losing the semantic invariance of multi-tasks.

To explore the role of Chinese-English code-switching data in semantic modeling, we select word pairs from the TED English-Chinese dataset, which does not contain code-switching data, and visualize them in Figure 3(d). It can be observed that their semantic distribution is relatively regular, but the distribution between synonym pairs is more scattered compared to Figure 3(c). This suggests that code-switching data plays a facilitating role in semantic modeling. This may be due to the co-occurrence of Chinese and English in the same sentence in the code-switching data. This co-occurrence makes the code-switching data closer to the Chinese and English monolingual data and acts as an intermediate bridge connecting the monolingual data.

Conclusion

In this paper, we focus on exploring the sentence-level semantic associations between different code-switching expressions. We propose a task-free semantic learning method based on this analysis. The model can learn the common semantic information from different tasks by sharing semantic space. We refine this into a semantic computational method by designing the loss of semantic invariant metrics across sentences. Experiments are conducted on tasks such as language modeling, ASR, and AST. The results indicate a significant improvement in performance for each task. This suggests that semantic constraint is a widely applicable method in the context of code-switching.

Acknowledgments

This work is supported by the NationalKey Research Development Plan of China (No.2023YFC3305903), the National Natural Science Foundation of China (NSFC) (No.62101553, No.62322120, No.62306316, No.U21B2010, No. 62206278).

References

Anastasopoulos, A.; and Chiang, D. 2018. Leveraging translations for speech transcription in low-resource settings. 1279–1283.

Berard, A.; Besacier, L.; Kocabiyikoglu, A. C.; and Pietquin, O. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6224–6228. IEEE.

Bérard, A.; Pietquin, O.; Servan, C.; and Besacier, L. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation.

Cieri, C.; Miller, D.; and Walker, K. 2004. The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. In Lino, M. T.; Xavier, M. F.; Ferreira, F.; Costa, R.; and Silva, R., eds., *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*. Lisbon, Portugal: European Language Resources Association (ELRA).

Dong, Q.; Wang, M.; Zhou, H.; Xu, S.; Xu, B.; and Li, L. 2021a. Consecutive decoding for speech-to-text translation. In *The Thirty-fifth AAAI Conference on Artificial Intelligence, AAAI*.

Dong, Q.; Ye, R.; Wang, M.; Zhou, H.; Xu, S.; Xu, B.; and Li, L. 2021b. "Listen, Understand and Translate": Triple Supervision Decouples End-to-end Speech-to-text Translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 12749–12759.

Doruz, A. S.; Sitaram, S.; Bullock, B. E.; and Toribio, A. J. 2021. A Survey of Code-switching: Linguistic and Social Perspectives for Language Technologies. In *Meeting of the Association for Computational Linguistics*.

Graves, A.; Fernandez, S.; Gomez, F.; and Schmidhuber, J. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, 369–376. ACM.

Graves, A.; Mohamed, A.-r.; and Hinton, G. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, 6645–6649. IEEE.

Gulati, A.; Qin, J.; Chiu, C.-C.; Parmar, N.; Zhang, Y.; Yu, J.; Han, W.; Wang, S.; Zhang, Z.; Wu, Y.; et al. 2020. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*.

Inaguma, H.; Duh, K.; Kawahara, T.; and Watanabe, S. 2019. Multilingual end-to-end speech translation. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 570–577. IEEE.

Jia, Y.; Johnson, M.; Macherey, W.; Weiss, R. J.; Cao, Y.; Chiu, C.-C.; Ari, N.; Laurenzo, S.; and Wu, Y. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7180–7184. IEEE.

Jose, N.; Chakravarthi, B. R.; Suryawanshi, S.; Sherly, E.; and Mccrae, J. P. 2020. A survey of current datasets for code-switching research.

Kenton, J. D. M.-W. C.; and Toutanova, L. K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*, 4171–4186.

Li, Y.; and Fung, P. 2013. Improved mixed language speech recognition using asymmetric acoustic model and language model with code-switch inversion constraints. In *IEEE*.

Li, Y.; and Fung, P. 2014. Code switch language modeling with Functional Head Constraint. In *IEEE International Conference on Acoustics*.

Liu, Y.; Xiong, H.; He, Z.; Zhang, J.; Wu, H.; Wang, H.; and Zong, C. 2019. End-to-end speech translation with knowledge distillation.

Liu, Y.; Zhang, J.; Xiong, H.; Zhou, L.; He, Z.; Wu, H.; Wang, H.; and Zong, C. 2020. Synchronous speech recognition and speech-to-text translation with interactive decoding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 8417–8424.

Lu, Y.; Huang, M.; Li, H.; Guo, J.; and Qian, Y. 2020. Bi-Encoder Transformer Network for Mandarin-English Code-Switching Speech Recognition Using Mixture of Experts.

Molina, G.; Alghamdi, F.; Ghoneim, M.; Hawwari, A.; Rey-Villamizar, N.; Diab, M.; and Solorio, T. 2019. Overview for the Second Shared Task on Language Identification in Code-Switched Data. *arXiv e-prints*.

Nakayama, S.; Kano, T.; Tjandra, A.; Sakti, S.; and Nakamura, S. 2019. Recognition and translation of code-switching speech utterances. In *2019 22nd Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, 1–6. IEEE.

Napoli, M. D.; Smith, C. J.; Hopkins, S. J.; Popa-Wagner, A.; and Slevin, M. 2014. Neuroinflammation and Immune Regulation in Ischemic Stroke: Identification of New Pharmacological Targets.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Park, D. S.; Chan, W.; Zhang, Y.; Chiu, C.; Zoph, B.; Cubuk, E. D.; and Le, Q. V. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Annual Conference of the International Speech Communication Association*, 2613–2617.

Patra, B. G.; Das, D.; and Das, A. 2018. Sentiment Analysis of Code-Mixed Indian Languages: An Overview of SAIL_Code-Mixed Shared Task @ICON-2017.

Poplack, S. 1981. *Syntactic structure and the social function of codeswitching*. Latino Language and Communicative Behaviour.

Qin, L.; Ni, M.; Zhang, Y.; and Che, W. 2020. CoSDA-ML: Multi-Lingual Code-Switching Data Augmentation for Zero-Shot Cross-Lingual NLP.

Shi, X.; Feng, Q.; and Xie, L. 2020. The ASRU 2019 Mandarin-English Code-Switching Speech Recognition Challenge: Open Datasets, Tracks, Methods and Results. *WSTCSMC 2020*, 71.

Sitaram, S.; Chandu, K. R.; Rallabandi, S. K.; and Black, A. W. 2019. A Survey of Code-switched Speech and Language Processing.

Solorio, T.; Blair, E.; Maharjan, S.; Bethard, S.; and Fung, P. 2014. Overview for the First Shared Task on Language Identification in Code-Switched Data.

Sperber, M.; Neubig, G.; Niehues, J.; and Waibel, A. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7: 313–325.

Touvron, H.; and et al, L. M. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Tucker, G. R. 2003. A Global Perspective on Bilingualism and Bilingual Education: Implications for New Jersey Educators.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Vydana, H. K.; Karafiat, M.; Zmolikova, K.; Burget, L.; and Cernocky, H. 2021. Jointly trained transformers models for spoken language translation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7513–7517. IEEE.

Weiss, R. J.; Chorowski, J.; Jaitly, N.; Wu, Y.; and Chen, Z. 2017. Sequence-to-sequence models can directly translate foreign speech. *arXiv preprint arXiv:1703.08581*.

Winata, G. I.; Cahyawijaya, S.; Liu, Z.; Lin, Z.; and Fung, P. 2021. Are Multilingual Models Effective in Code-Switching?

Yan, B.; Zhang, C.; Yu, M.; Zhang, S.-X.; Dalmia, S.; Berrebbi, D.; Weng, C.; Watanabe, S.; and Yu, D. 2021. Joint Modeling of Code-Switched and Monolingual ASR via Conditional Factorization. *arXiv preprint arXiv:2111.15016*.

Zhang, S.; Yi, J.; Tian, Z.; Bai, Y.; Tao, J.; Liu, X.; and Wen, Z. 2021a. End-to-End Spelling Correction Conditioned on Acoustic Feature for Code-Switching Speech Recognition. *Proc. Interspeech 2021*, 266–270.

Zhang, S.; Yi, J.; Tian, Z.; Bai, Y.; Tao, J.; et al. 2021b. Decoupling pronunciation and language for end-to-end code-switching automatic speech recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6249–6253. IEEE.

Zheng, B.; Dong, L.; Huang, S.; Wang, W.; Chi, Z.; Singhal, S.; Che, W.; Liu, T.; Song, X.; and Wei, F. 2021. Consistency Regularization for Cross-Lingual Fine-Tuning.