Aligner^2: Enhancing Joint Multiple Intent Detection and Slot Filling via Adjustive and Forced Cross-Task Alignment

Zhihong Zhu, Xuxin Cheng, Yaowei Li, Hongxiang Li, Yuexian Zou*

School of Electronic and Computer Engineering, Peking University
{zhihongzhu, chengxx, ywl, lihongxiang}@stu.pku.edu.cn
zouyx@pku.edu.cn

Abstract

Multi-intent spoken language understanding (SLU) has garnered growing attention due to its ability to handle multiple intent utterances, which closely mirrors practical scenarios. Unlike traditional SLU, each intent in multi-intent SLU corresponds to its designated scope for slots, which occurs in certain fragments within the utterance. As a result, establishing precise scope alignment to mitigate noise impact emerges as a key challenge in multi-intent SLU. More seriously, they lack alignment between the predictions of the two sub-tasks due to task-independent decoding, resulting in a limitation on the overall performance. To address these challenges, we propose a novel framework termed Aligner^2 for multi-intent SLU, which contains an Adjustive Cross-task Aligner (ACA) and a Forced Cross-task Aligner (FCA). ACA utilizes the information conveyed by joint label embeddings to accurately align the scope of intent and corresponding slots, before the interaction of the two subtasks. FCA introduces reinforcement learning, to enforce the alignment of the task-specific hidden states after the interaction, which is explicitly guided by the prediction. Extensive experiments on two public multi-intent SLU datasets demonstrate the superiority of our Aligner^2 over state-of-the-art methods. More encouragingly, the proposed method Aligner^2 can be easily integrated into existing multi-intent SLU frameworks, to further boost performance.

1 Introduction

Spoken Language Understanding (SLU) plays a critical role in task-oriented dialogue systems (Tur and De Mori 2011; Qin et al. 2021c). A typical SLU task mainly includes two subtasks: intent detection to identify users' intents and slot filling to extract semantic constituents from the user's query. In a more practical scenario, users usually express more than one intent in an utterance (Gangadharaiah and Narayanaswamy 2019; Cheng et al. 2023a). Thus, multi-intent SLU is derived and has attracted extensive attention. A simple example of multi-intent SLU is shown in Figure 1.

One of the key challenges in multi-intent SLU is accurately aligning both the user's intent and slots simultaneously within complex utterances. Generally speaking, a user's utterance consists of multiple sub-utterances, with each sub-utterance conveying a corresponding intent. The slots are consequently embedded within certain tokens of each sub-utterance, pertaining solely to the intent of that particular sub-utterance and remaining unrelated to others. Take the example in Figure 1, the intent `AddtoPlaylist` of the previous sub-utterance is related to the token `David axelrod` and `futuors hits`. And the intent of the second sub-utterance `PlayMusic` is mainly originated from token `latin` and `zvooq`. Thus, introducing irrelevant intents to the current sub-utterance and introducing irrelevant tokens in the current utterance for the intent will impact the performance of SLU. To conclude, how to accurately align the scope between each intent and its corresponding slots becomes a major challenge.

Figure 1: A typical example of multi-intent SLU, where `B/I-a` denotes `B/I-artist`, `B/I-p` denotes `B/I-playlist`, `B-g` denotes `B-genre`, and `B-s` denotes `B-service`. Different colors represent different intents and the scope of their corresponding slots.

Another challenge in multi-intent SLU is the lack of alignment between the correct predictions of the two subtasks. To be specific, most existing models decode the hidden states of the two subtasks independently without leveraging the correlations between them, which leads to the misalignment of the correct predictions of the two subtasks. As shown in Figure 2, the F1 score of slot filling and the accuracy of intent detection may increase or decrease asynchronously. Overall accuracy is a more important metric and it denotes the ratio of the utterances for which both intents and slots are predicted correctly in an utterance. Due to the lack of alignment, overall accuracy on utterance-level semantic frame parsing is much worse than these two subtasks' single metrics.

To solve the above challenges, in this paper, we propose a novel method Aligner^2 to enhance multi-intent SLU. Our Aligner^2 contains an adjustive cross-task aligner (ACA) before the interaction of the two subtasks, and a forced cross-task aligner (FCA) after the interaction of the two subtasks. The ACA is proposed to align the scope information for the first challenge. Technically, we introduce joint label embeddings as a bridge to align the semantic scope. Intent-specific

*Corresponding author.

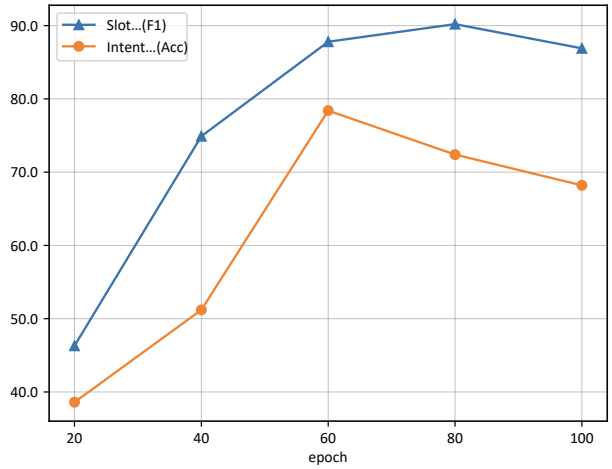

Figure 2: An example illustrates the potential misalignment between correct predictions in intent detection and slot filling. As the F1 score for slot filling increases, there may be a decrease in the accuracy of intent detection, thereby limiting the overall accuracy of the utterance.

and Slot-specific hidden states are represented as the combinations of label embeddings shared between tasks. By this means, the joint label embeddings serve as a bridge to connect the two subtasks, the semantic scope of which then can be aligned by the dual-task inter-dependencies conveyed in the learned label embeddings. ACA can be conceptualized as an adjustment that occurs before the interaction of two hidden states, aligned through the joint label embeddings. For the second challenge, we introduce FCA based on reinforcement learning (RL), which can guide the learning process of cross-task alignment with appropriate supervision from carefully designed rewards. Concretely, we leverage the signals from SLU metrics, *i.e.*, overall accuracy on utterance-level semantic frame parsing, to guide the two subtasks' learning to have a direct target of learning outcome for SLU. FCA can be considered as another alignment after the interaction of two hidden states, which is forced by the predicted results. In a nutshell, our contributions are summarized as follows:

- We propose an adjustive cross-task aligner to accurately align the scope of the intent and its corresponding slots in multi-intent SLU utterances, which relies on information conveyed by joint label embeddings.
- We propose a forced cross-task aligner based on reinforcement learning to enforce the alignment of the hidden states of the two subtasks after interactions, which is directly guided by the predicted results.
- Extensive experiments demonstrate the superiority of our model over cutting-edge methods. More encouragingly, the proposed Aligner[2] can be easily integrated into existing SLU frameworks, to further improve performance.

2 Related Work

In this section, we describe the related works: 1) Spoken Language Understanding; 2) Reinforcement Learning.

Spoken Langugae Understanding Since intent detection and slot filling are highly correlated in SLU, a bunch of models (Wu et al. 2020; Qin et al. 2021a; Wu, Harris, and Zhao 2021; Wang et al. 2021a; Chen et al. 2022a; Zhu et al. 2023a,b,c; Cheng et al. 2023b) have been proposed to jointly tackle the two subtasks. However, they focus on single-intent utterances, which may not be practical in real-world scenarios, where an utterance usually expresses multiple intents.

To this end, Kim, Ryu, and Lee (2017) begin to explore multiple intent detection. Gangadharaiah and Narayanaswamy (2019) first employed a multi-task framework to tackle the multiple intent detection and slot filling jointly. With the increasing popularity of graph neural networks in various NLP tasks (Hu et al. 2019; Wang et al. 2021b; Chen et al. 2022b; Zhang, Zhou, and Wang 2022), multi-intent SLU systems also leveraged GNNs to model the cross-task interactions. Qin et al. (2020) and Qin et al. (2021b) proposed graph interaction networks to model implicit correlations between intent labels and slot tokens. Chen, Zhou, and Zou (2022) reformulated multi-intent detection as a weakly supervised task and designed a self-distillation mechanism to circularly refresh the pipeline. Song et al. (2022) built a global graph to leverage the intent-slot co-occurrence, enhancing the SLU performance. Xing and Tsang (2022a) implemented a two-stage framework achieving mutual guidance between intents and slots. Cheng, Yang, and Jia (2023) proposed a model to improve the intent detecion and mitigate the error propagation problem. Xing and Tsang (2022b) further exploited label typologies and relations, obtaining state-of-the-art results. However, most existing works neglect the cross-task misalignment between the two subtasks. In this work, we propose ACA before the interaction of the two subtasks and FCA after the interaction of the two subtasks to boost SLU performance.

Reinforcement Learning Many NLP tasks have been solved by reinforcement learning (RL) techniques, such as question answering (Xiong, Zhong, and Socher 2018; Lu et al. 2022), dialogue generation (Li et al. 2016), machine translation (Shao et al. 2019; Wu et al. 2018), essay scoring (Wang et al. 2018b) and sentiment transfer (Xu et al. 2018). RL is generally used for auxiliary tasks, such as ensuring the output is properly formatted (Zhong, Xiong, and Socher 2017). In SLU, Wang et al. (2018a) applied RL to learn the wrong labeled slots with or without user's feedback; Rao et al. (2021) proposed a reinforcement framework to improve automatic speech recognition robustness. In this work, we apply reinforcement learning to alleviate the misalignment between the correct predictions of the two subtasks in multi-intent SLU.

3 Problem Formulation

In this section, we introduce the problem formulation for the multi-Intent SLU task. Given an input utterance $\boldsymbol{U} = (u_1, u_2, \cdots, u_n)$, where n denotes the length of the utterance $\boldsymbol{U}$. **Multiple intent detection** is treated as a multi-label classification task. Therein, the token-level predicted intent sequence is denoted as $\boldsymbol{y}'^I = (y'^I_1, y'^I_2, \cdots, y'^I_n)$. The utterance-level predicted intent sequence $\boldsymbol{y}^I$ is obtained by the token-level intent voting method (Qin et al. 2021b). **Slot**

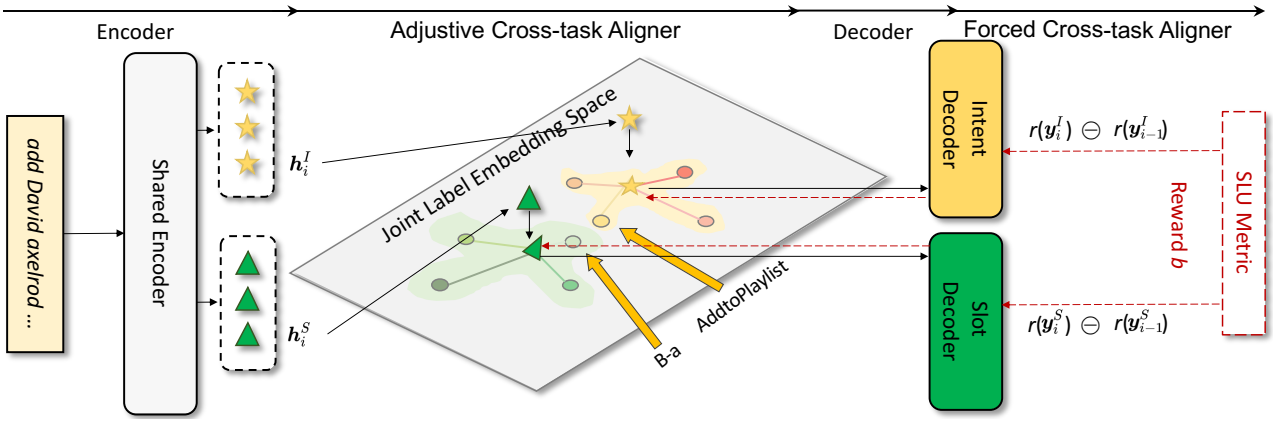

Figure 3: The overall framework of proposed Aligner^2, which contains encoder and decoder (§4.1), adjustive cross-task aligner (§4.2) and forced cross-task aligner (§4.3). For brevity, we omit the interaction between the two subtasks in most existing works, which occurs after the adjustive cross-task aligner and before the decoder.

filling is a sequence labeling task, where the predicted slot sequence is denoted as $\boldsymbol{y}^S = (y_1^S, y_2^S, \cdots, y_n^S)$. For simplicity, we use $\boldsymbol{y}$ to denote the union of $\boldsymbol{y}^I$ and $\boldsymbol{y}^S$.

4 Aligner^2

In this section, we describe the details of our proposed method Aligner^2. The overall framework of Aligner^2 is illustrated in Figure 3. To begin with, we first introduce the encoder and decoder structure of multi-intent SLU.

4.1 Encoder and Decoder Structure

Given $\boldsymbol{U}$, the input hidden state $\boldsymbol{H}$ can be obtained by a shared utterance encoder, *i.e.*, self-attentive encoder (Qin et al. 2020, 2021b) (please refer to §5 for experimental details on pre-trained encoders (Chen et al. 2022c)). Subsequently, $\boldsymbol{H}$ is fed to two different BiLSTMs to obtain intent-specific state $\boldsymbol{H}^I$ and slot-specific state $\boldsymbol{H}^S$ for intent detection and slot filling task, respectively. Typically, these two hidden states undergo a series of interactions, denoted as $\boldsymbol{H}'^I$ and $\boldsymbol{H}'^S$ for intent detection and slot filling after interaction, respectively. The two of them then pass through their respective decoders to get the predicted results $\boldsymbol{y}^I$ and $\boldsymbol{y}^S$ mentioned in §3. Eventually, a joint training scheme is adopted to optimize intent detection and slot filling simultaneously.

Joint Training Due to the interaction between the two subtasks, joint models are widely used to consider the two subtasks and update parameters. Formally, the multiple intent detection objective is defined as:

$$\mathrm{CE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \hat{\boldsymbol{y}} \log(\boldsymbol{y}) + (1 - \hat{\boldsymbol{y}}) \log(1 - \boldsymbol{y}), \tag{1}$$

$$\mathcal{L}_I = -\sum_{i=1}^{n} \sum_{j=1}^{N_I} \mathrm{CE}(\hat{y}_i^{[j,I]}, y_i^{[j,I]}), \tag{2}$$

where N_I denotes the number of intent labels, $\hat{y}_i^{[j,I]}$ denotes the gold intent and $y_i^{[j,I]}$ denotes the predicted intent.

Similarly, the slot filling objective is defined as:

$$\mathcal{L}_S = -\sum_{i=1}^{n} \sum_{j=1}^{N_S} \hat{y}_i^{[j,S]} \log(y_i^{[j,S]}), \tag{3}$$

where N_S denotes the number of slot labels, $\hat{y}_i^{[j,S]}$ denotes the gold slot and $y_i^{[j,S]}$ denotes the predicted slot.

The final joint objective is formulated as:

$$\mathcal{L} = \alpha \mathcal{L}_I + \beta \mathcal{L}_S, \tag{4}$$

where α and β are trade-off hyper-parameters.

4.2 Adjustive Cross-task Aligner

Before the interaction of these two task-specific hidden states, we propose Adjustive Cross-task Aligner (ACA) to align the scope of the intent and its corresponding slot. Note that the significant differences between Xing and Tsang (2022a); Zhu et al. (2023b) and ours are: ACA transforms task-specific representations before interaction, whereas their approaches employ it in the decoding stage; Their label encoding requires a series of intricate technologies such as pre-training and GCN, whereas our label encoding can be initialized randomly, yet still result in substantial improvements. To conclude, ACA can be treated as an adjustment that occurs before the interaction, aligned through the joint label embeddings.

Specifically, the task-specific hidden state is first projected into a joint label embedding space, and then the dot products are conducted between it and all label embeddings of a specific task to obtain the correlation score vector. Finally, we reintegrate the correlation score vector into the task-specific hidden state to achieve scope adjustment. Since labels convey certain semantics, the intents and slots associated with them will be close in the label embedding space. In this way, the joint label embeddings serve as a bridge that connects the two subtasks, and the beneficial correlative information conveyed in the learned label embeddings can align the scope for the two subtasks. Formally, the intent-specific hidden state $\boldsymbol{H}^I$ is first projected into a joint label embedding space via an MLP

layer. Then, the correlation score vector $\boldsymbol{C}^I$ is calculated via dot products between it and all intent label embeddings. This process can be formulated as:

$$\boldsymbol{c}_i^I = (\boldsymbol{W}_I(\text{LeakyReLU}(\boldsymbol{W}_h^I \boldsymbol{h}_i^I + \boldsymbol{b}_h^I)) + \boldsymbol{b}_I)\boldsymbol{E}^I, \quad (5)$$

where $\boldsymbol{c}_i^I$ is a N_I-dimensional vector; $\boldsymbol{E}^I$ is the sample-specific intent label embedding matrix; $\boldsymbol{W}_*$ and $\boldsymbol{b}_*$ are model parameters. Finally, we obtain the adjusted intent-specific hidden state $\hat{\boldsymbol{h}}_i^I$ by: $\hat{\boldsymbol{h}}_i^I = \boldsymbol{c}_i^I \boldsymbol{E}^I$. Similar to Eq.5, the slot correlation score vector $\boldsymbol{c}_i^S$ is obtained by:

$$\boldsymbol{c}_i^S = (\boldsymbol{W}_S(\text{LeakyReLU}(\boldsymbol{W}_h^S \boldsymbol{h}_i^S + \boldsymbol{b}_h^S)) + \boldsymbol{b}_S)\boldsymbol{E}^S, \quad (6)$$

where $\boldsymbol{c}_i^S$ is a N_S-dimensional vector; $\boldsymbol{E}^S$ is the sample-specific slot label embedding matrix; $\boldsymbol{W}_*$ and $\boldsymbol{b}_*$ are model parameters. The adjusted slot-specific hidden state $\hat{\boldsymbol{h}}_i^S$ can be obtained by: $\hat{\boldsymbol{h}}_i^S = \boldsymbol{c}_i^S \boldsymbol{E}^S$.

Now, we can utilize the adjusted task-specific hidden states $\hat{\boldsymbol{H}}^I$ and $\hat{\boldsymbol{H}}^S$ instead of $\boldsymbol{H}^I$ and $\boldsymbol{H}^S$ to perform interaction.

4.3 Forced Cross-task Aligner

Although ACA provides a "soft" mechanism to facilitate the linking between intent and slot hidden states, there still exists misalignment between the correct prediction of the two subtasks, due to the independent decoding process. To this end, we propose Forced Cross-task Aligner (FCA) based on RL to provide appropriate supervision from SLU metrics to search for better alignment between hidden states from the two subtasks after the interaction. In doing so, we treat the SLU model as the **agent** that interacts with an external **environment** (intent-specific and slot-specific hidden states). Therefore, all parameters of our approach, θ, define a **policy** p_θ that results in an action (*i.e.*, the prediction of the intent and slot). During training, the **agent** uses a **reward** r based on SLU metrics, *e.g.*, intent accuracy, slot F1 and overall accuracy, *etc.*, where the **reward** r_t for the **action** at epoch t is the improvement on the SLU metric by predicting the next epoch of intent and slot $\boldsymbol{y}_t$, which is formally expressed by:

$$r_t = r(\boldsymbol{y}_t) - r(\boldsymbol{y}_{t\text{-}1}), \quad (7)$$

where $\boldsymbol{y}_t$ and $\boldsymbol{y}_{t\text{-}1}$ denote the union of predicted intent and slot at t and t-1 epoch, respectively. Therefore, the entire reward R of predicting $\boldsymbol{y}$ is the sum of r_t:

$$R = \sum_{t=1}^{T} r(\boldsymbol{y}_t) - r(\boldsymbol{y}_{t\text{-}1}) = r(\boldsymbol{y}). \quad (8)$$

Then the model is trained to maximize the expected reward $\mathbb{E}_{\boldsymbol{y}\sim p_\theta}[r(\boldsymbol{y})]$. Based on $\mathbb{E}_{\boldsymbol{y}\sim p_\theta}[r(\boldsymbol{y})]$, the loss of our proposed FCA is defined as:

$$\mathcal{L}_{fca}(\theta) = -\mathbb{E}_{\boldsymbol{y}\sim p_\theta}[r(\boldsymbol{y})] \quad (9)$$

with the gradient of $\mathcal{L}(\theta)_{fca}$ for θ computed using the REINFORCE algorithm (Williams 1992) via

$$\nabla_\theta \mathcal{L}_{fca}(\theta) = -\mathbb{E}_{\boldsymbol{y}\sim p_\theta}[r(\boldsymbol{y}) \nabla_\theta \log p_\theta(\boldsymbol{y})]. \quad (10)$$

Then, we approximate the expectation (*i.e.*, the expected gradient) through a single Monte-Carlo sample $\boldsymbol{y}$ from p_θ:

$$\nabla_\theta \mathcal{L}_{fca}(\theta) \approx -r(\boldsymbol{y}) \nabla_\theta \log p_\theta(\boldsymbol{y}). \quad (11)$$

However, the gradient estimated from the above process is of high variance. To maintain the stability of the RL, we follow Rennie et al. (2017) to reduce such variance by introducing a reference reward $\boldsymbol{b}$. which is normally a constant (*i.e.*, a reference reward value) obtained from higher rewards by sampling all possible actions. Note that the introduction of $\boldsymbol{b}$ does not change the expected gradient (Eq.10), which can be proved by:

$$\begin{aligned} \mathbb{E}_{\boldsymbol{y}\sim p_\theta}[\boldsymbol{b} \nabla_\theta \log p_\theta(\boldsymbol{y})] &= \boldsymbol{b} \sum_{\boldsymbol{y}} \nabla_\theta \log p_\theta(\boldsymbol{y}) \\ &= \boldsymbol{b} \nabla_\theta \sum_{\boldsymbol{y}} \log p_\theta(\boldsymbol{y}) \\ &= \boldsymbol{b} \nabla_\theta 1 \\ &= 0. \end{aligned} \quad (12)$$

Therefore, Eq.10 is formalized as:

$$\nabla_\theta \mathcal{L}_{fca}(\theta) = -\mathbb{E}_{\boldsymbol{y}\sim p_\theta}[(r(\boldsymbol{y}) - \boldsymbol{b}) \nabla_\theta \log p_\theta(\boldsymbol{y})] \quad (13)$$

with the expected gradient approximated by

$$\nabla_\theta \mathcal{L}_{fca}(\theta) \approx -(r(\boldsymbol{y}) - \boldsymbol{b}) \nabla_\theta \log p_\theta(\boldsymbol{y}). \quad (14)$$

Note that any actions that return higher $r(\boldsymbol{y})$ than $\boldsymbol{b}$ drive the following learning process to take as more such actions as possible.

5 Experiments

In this section, we first introduce the benchmarks we used for experiments, as well as the metrics, base models and settings that we tested. Then, we present the results and analyses of our proposed Aligner[2].

5.1 Experiment Setup

Datasets We conduct experiments on two publicly multi-intent SLU datasets[1]: MixATIS and MixSNIPS (Hemphill, Godfrey, and Doddington 1990; Coucke et al. 2018; Qin et al. 2020). In MixATIS, the split of the train/dev/test set is 13,162/756/828 (utterances). In MixSNIPS, the split of the train/dev/test set is 39,776/2,198/2,199 (utterances). The data statistics are shown in Table 1.

Dataset	MixATIS	MixSNIPS
Vocabulary Size	722	11241
Intent categories	17	6
Slot categories	116	71
Training set size	13,162	39,776
Validation set size	756	2,198
Test set size	828	2,199

Table 1: Dataset statistics.

Metrics We evaluate the performance of models on the widely-used multi-intent SLU metrics (Goo et al. 2018; Cheng et al. 2023c; Zhuang, Cheng, and Zou 2023), *i.e.*, accuracy for intent-detection, F1 score for slot filling, and overall accuracy for the sentence-level semantic frame parsing. In particular, overall accuracy denotes the ratio of utterances whose intents and slots are all correctly predicted.

[1]https://github.com/LooperXX/AGIF

Method	MixATIS			MixSNIPS		
	Slot (F1)↑	Intent (Acc)↑	Overall (Acc)↑	Slot (F1)↑	Intent (Acc)↑	Overall (Acc)↑
AGIF (Qin et al. 2020)	86.8	74.2	41.0	94.3	94.9	74.1
with Aligner^2	**87.4**	**74.6**	**41.7**	**94.6**	**95.2**	**74.5**
GL-GIN (Qin et al. 2021b)	88.3	76.3	43.5	94.9	95.6	75.4
with Aligner^2	**88.7**	**76.6**	**44.2**	**95.3**	**95.9**	**76.0**
SDJN (Chen, Zhou, and Zou 2022)	88.2	77.1	44.6	94.4	96.5	75.7
with Aligner^2	**88.7**	**77.4**	**45.1**	**94.8**	**96.9**	**76.2**
GISC (Song et al. 2022)	88.5	75.0	48.2	95.0	95.5	75.9
with Aligner^2	**89.1**	**75.7**	**49.0**	**95.6**	**95.8**	**76.5**
Co-guiding Net (Xing and Tsang 2022a)	89.8	79.1	51.3	95.1	97.7	77.5
with Aligner^2	**90.4**	**79.6**	**51.9**	**95.6**	**98.0**	**77.9**
ReLa-Net (Xing and Tsang 2022a)	90.1	78.5	52.2	94.7	97.6	76.1
with Aligner^2	**90.6**	**78.7**	**52.8**	**95.3**	**98.1**	**76.5**
SSRAN (Cheng, Yang, and Jia 2023)	89.2	77.6	48.7	95.7	98.3	77.4
with Aligner^2	**89.7**	**77.9**	**49.4**	**96.0**	**98.6**	**77.8**

Table 2: Performance (%) on MixATIS and MixSNIPS datasets. We conducted 5 runs with different seeds for all experiments, the t-tests indicate that $p < 0.01$. As we can see, all the baseline models with significantly different structures enjoy a comfortable improvement with our proposed Aligner^2.

Baselines In our experiments, we choose seven recent multi-intent SLU models with different interaction structures as baselines, *i.e.*, (1) **AGIF** (Qin et al. 2020) utilized an adaptive graph interaction framework to capture the fine-grained multi-intent information for slot filling. (2) **GL-GIN** (Qin et al. 2021b) proposed a global-local graph interaction network to achieve decoding in a non-autoregressive way. (3) **SDJN** (Chen, Zhou, and Zou 2022) reformulated multi-intent detection as a weakly supervised task. (4) **GISC** (Song et al. 2022) built a global graph based on inter-label statistical dependency. (5) **Co-guiding Net** (Xing and Tsang 2022a) implemented a two-stage framework achieving mutual guidance between intents and slots. (6) **ReLa-Net** (Xing and Tsang 2022b) exploited label typologies and relations, which is the recent state-of-the-art model. (7) **SSRAN** (Cheng, Yang, and Jia 2023) devised a scope-sensitive result attention network based on Transformer to utilize the bidirectional interaction between results and mitigate the error propagation problem.

In detail, to demonstrate the effectiveness of our method, we compare the performance of these models with and without the proposed Aligner^2.

Implementation Details We implemented all the baselines used in our experiments using PyTorch (Paszke et al. 2019) (ver. 1.10.1)[2] one 1 Nvidia V100. We run the baselines also on the same computing environment, using the configuration file they provided. We train our model using the initial loss function for 30 epochs to regularize the action space before the RL is applied. Afterward, we start RL using the Adam optimizer (Kingma and Ba 2014). For each dataset, we try all combinations of the hyper-parameters and use the one achieving the highest overall accuracy on the validation sets.

5.2 Main Results

The experimental results on MixATIS and MixSNIPS datasets are reported in Table 2. From the results, we can conclude that (1) Our proposed Aligner^2 can consistently boost all baselines across all metrics, with a relative overall accuracy improvement of $0.5\% \sim 0.8\%$, $0.4\% \sim 0.6\%$ on MixATIS and MixSNIPS, respectively. These results verify the effectiveness of our model intuitively. (2) Notably, the overall accuracy improvement on MixATIS is more significant than on MixSNIPS. We observe that MixATIS has a relatively smaller sample size but a larger number of intents and slots. This characteristic makes it challenging for conventional methods to effectively align the two subtasks. In contrast, our model achieves an adjusted alignment before interaction and a forced alignment after interaction. Aligner^2 facilitates improved coordination and unification of the hidden states of the two subtasks, thereby boosting SLU performance. (3) To justify whether the improvement is statistically significant, we conducted t-tests between our results and the second-best results and found that all the $p < 0.01$. This also validates the superiority of Aligner^2 over existing methods.

5.3 Ablation and Analysis

In this section, we do extensive ablation analyses to verify the effectiveness of ACA and FCA in detail.

Effect of Adjustive Cross-task Aligner We first remove our ACA component to verify its effectiveness, which is shown in setting (b) of **ReLa-Net** and **SSRAN** in Table 3, respectively. Without ACA, the performance drops by $\sim$0.2% slot F1, $\sim$0.3% intent accuracy, and $\sim$0.3% overall accuracy on MixATIS and $\sim$0.3% slot F1, $\sim$0.1% intent accuracy and $\sim$0.2% overall accuracy on MixSNIPS. This phenomenon indicates that the scope alignment is beneficial for multi-intent SLU, especially on the slot filling task.

Effect of Forced Cross-task Aligner To verify the effect of FCA, we further remove our FCA component, which is referred to as setting (a) of **ReLa-Net** and **SSRAN** in Table 3, respectively. Without FCA, the overall accuracy drops by $\sim$0.5% and $\sim$0.3%, demonstrating the significance of alignment after interaction. Since the FCA and ACA can improve the performance from different alignment stages,

[2]https://github.com/pytorch/pytorch/

Method	Aligner^2		MixATIS			MixSNIPS		
	ACA	FCA	Slot (F1)↑	Intent (Acc)↑	Overall (Acc)↑	Slot (F1)↑	Intent (Acc)↑	Overall (Acc)↑
ReLa-Net	✗	✗	90.1	78.5	52.2	94.7	97.6	76.1
(a)	✓	✗	90.3	78.6	52.4	94.8	97.8	76.2
(b)	✗	✓	90.4	78.7	52.6	94.9	98.0	76.3
Full Model	✓	✓	**90.6**	**78.7**	**52.8**	**95.3**	**98.1**	**76.5**
SSRAN	✗	✗	89.2	77.6	48.7	95.7	98.3	77.4
(a)	✓	✗	89.4	77.6	48.9	95.8	98.4	77.5
(b)	✗	✓	89.5	77.8	49.1	95.8	98.5	77.7
Full Model	✓	✓	**89.7**	**77.9**	**49.4**	**96.0**	**98.6**	**77.8**

Table 3: Ablation study of our approach, which includes two state-of-the-art baseline models. "ACA": Adjustive Cross-task Aligner. "FCA": Forced Cross-task Aligner. "Full Model": the baseline model with Aligner^2.

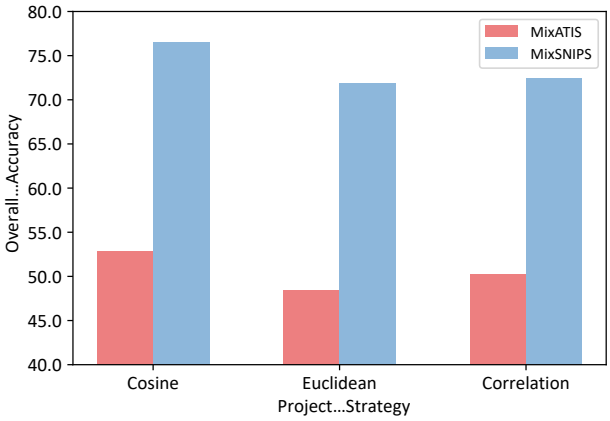

Figure 4: Effect of project strategy (distance) in Adjustive Cross-task Aligner on ReLa-Net on two datasets.

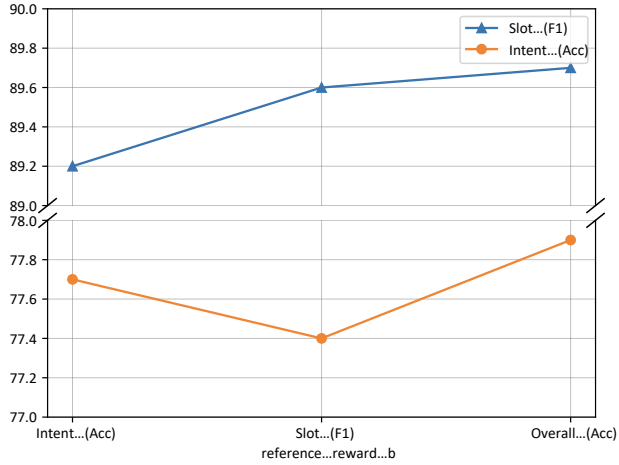

Figure 5: Effect of reference value in forced cross-task aligner on SSRAN on the MixATIS dataset.

combing them can lead to the most prominent improvement across all metrics (*cf.* Full Model), with up to 52.8% and 76.5% overall accuracy in terms of **ReLa-Net**; 49.4% and 77.8% overall accuracy in terms of **SSRAN**.

Note that compared to ACA, FCA leads to a more significant enhancement in the model's performance. We attribute this to the fact that FCA integrates prediction-level cues and employs reinforcement learning for "hard" alignment, whereas ACA utilizes joint label embeddings for "soft" alignment before interaction.

Effect of Project Strategy in ACA In our experiments, we employ dot product operation to calculate the cosine distance between task-specific hidden states and label embeddings in ACA. Furthermore, we delve into the effects of various projection strategies in Figure 4. Experiments on ReLa-Net reveal that our ACA module is minimally influenced by the projection strategy, with the simple Cosine distance strategy outperforming the other two strategies.

Effect of Reference Value in FCA In FCA, we naturally selected the overall accuracy as the reference value $\boldsymbol{b}$, and we substituted this value to observe its impact on the FCA module. As shown in Figure 6, opting for the slot as the reference value yields excellent slot performance, yet results in poor intent performance, thereby exacerbating the misalignment between the two subtasks. Akin observations emerge when the intent is selected as the reference value. Hence, the appropriate approach is to employ a more comprehensive metric encompassing both tasks, *i.e.*, the overall accuracy.

Generalization to single-intent SLU An intuitive consideration is whether our method has the generalization ability. To further verify the generalization ability of Aligner^2, we also select three single-intent SLU baselines with distinct architectures, *i.e.*, **LR-Transformer** (Cheng, Jia, and Yang 2021), **Co-Interactive** (Qin et al. 2021a) and **HAN** (Chen et al. 2022a). As shown in Table 4, we can observe that upon integrating our model, all three baselines achieve consistent and notable enhancements, thereby validating the generalization ability of our Aligner^2.

Method	ATIS	SNIPS
LR-Transformer	87.2	88.4
with Aligner^2	**87.8**	**88.9**
Co-Interactive	90.3	87.4
with Aligner^2	**90.6**	**87.7**
HAN	91.8	88.7
with Aligner^2	**92.4**	**89.2**

Table 4: Overall performance on ATIS and SNIPS datasets.

Utterance	list	la	and	also	how	many	canadian	airlines	flights	use	aircraft	dh8
Intent(Baseline)	atis_quantity											
Intent(Ours)	atis_quantity, atis_city											
Slot(Baseline)	O	B-city_name	O	O	O	O	B-airline_name	I-airline_name	O	O	O	O
Slot(Ours)	O	B-city_name	O	O	O	O	B-airline_name	I-airline_name	O	O	O	B-aircraft_code

(a)

Utterance	how	many	canadian	airlines	international	flights	use	j31
Intent(Baseline)	atis_quantity							
Intent(Ours)	atis_quantity							
Slot(Baseline)	O	O	B-airline_name	I-airline_name	I-airline_name	O	O	B-airline_name
Slot(Ours)	O	O	B-airline_name	I-airline_name	I-airline_name	O	O	B-aircraft_code

(b)

Figure 6: Case study of our model compared to previous models in achieving scope alignment and prediction alignment. Correct predictions are marked in green while errors are marked in red.

Comparison with PLMs Following previous works, we conduct experiments to evaluate the ability of our model to combine pre-trained language models by replacing our encoder with **RoBERTa** (Liu et al. 2019), **BERT** (Kenton and Toutanova 2019) and **XLNet** (Yang et al. 2019), respectively.

The experiment results are shown in Table 5, where we can observe that: (1) Our proposed Aligner^2 benefits from all three pre-trained models, which verifies the compatibility between our method and the pre-trained model. (2) Aligner^2 outperforms **ReLa-Net** and **SSRAN** when utilizing **RoBERTa**, which further verifies the effectiveness of our Aligner^2 on multi-intent SLU systems.

Method	MixATIS	MixSNIPS
RoBERTa†	49.7	80.2
ReLa-Net + RoBERTa‡	54.4	83.8
Aligner^2 + RoBERTa‡	**55.6**	**85.1**
BERT†	51.6	83.0
ReLa-Net + BERT ‡	53.9	84.8
Aligner^2 + BERT‡	**55.7**	**85.6**
XLNet†	52.1	84.8
ReLa-Net + XLNet ‡	54.4	85.9
Aligner^2 + XLNet‡	**56.2**	**86.8**

Table 5: Overall performance on MixATIS and MixSNIPS datasets. "†": results from Cheng, Yang, and Jia (2023). "‡": results by our implementation.

5.4 Qualitative Analysis

We provide two cases from the MixATIS dataset to compare the results generated by the best baseline **ReLa-Net** and our method. In Figure 5(a), the utterance consists of two sub-utterances. Due to lack of scope alignment, **ReLa-Net** misses the intent `atis_city` corresponding to `la` slot and misses token `dh8` corresponding to intent `atis_quantity`. In Figure 5(b), despite the sentence having only one intent, **ReLa-Net** incorrectly predicts the `j31` slot as `B-airline_name`. This error arises due to the absence of alignment in the decoding process between the two subtasks. In contrast, our proposed Aligner^2 can perform scope and prediction alignment and predict them correctly.

6 Conclusion and Future Work

We presented Aligner^2 for multi-intent SLU, which seamlessly unifies the scope alignment before interaction and prediction alignment after interaction. Extensive experiments and analyses on two public SLU datasets demonstrate the superiority of our Aligner^2 over state-of-the-art methods. In the future, we will explore more techniques for mining and exploiting the alignment between the two subtasks.

Acknowledgements

This paper was partially supported by Shenzhen Science & Technology Research Program (No:GXWD20201231165807007-20200814115301001) and NSFC (No: 62176008).

References

Chen, D.; Huang, Z.; Wu, X.; Ge, S.; and Zou, Y. 2022a. Towards Joint Intent Detection and Slot Filling via Higher-order Attention. In *IJCAI*.

Chen, H.; Zhai, Z.; Feng, F.; Li, R.; and Wang, X. 2022b. Enhanced Multi-Channel Graph Convolutional Network for Aspect Sentiment Triplet Extraction. In *ACL*.

Chen, L.; Chen, N.; Zou, Y.; Wang, Y.; and Sun, X. 2022c. A Transformer-based Threshold-Free Framework for Multi-Intent NLU. In *COLING*.

Chen, L.; Zhou, P.; and Zou, Y. 2022. Joint Multiple Intent Detection and Slot Filling Via Self-Distillation. In *ICASSP*.

Cheng, L.; Jia, W.; and Yang, W. 2021. An Effective Non-Autoregressive Model for Spoken Language Understanding. In *CIKM*.

Cheng, L.; Yang, W.; and Jia, W. 2023. A Scope Sensitive and Result Attentive Model for Multi-Intent Spoken Language Understanding. In *AAAI*.

Cheng, X.; Xu, W.; Zhu, Z.; Li, H.; and Zou, Y. 2023a. Towards Spoken Language Understanding via Multi-level Multi-grained Contrastive Learning. In *CIKM*.

Cheng, X.; Zhu, Z.; Cao, B.; Ye, Q.; and Zou, Y. 2023b. MRRL: Modifying the Reference via Reinforcement Learning for Non-Autoregressive Joint Multiple Intent Detection and Slot Filling. In *EMNLP*.

Cheng, X.; Zhu, Z.; Li, H.; Li, Y.; Zhuang, X.; and Zou, Y. 2023c. Towards Multi-Intent Spoken Language Understanding via Hierarchical Attention and Optimal Transport. In *AAAI*.

Coucke, A.; Saade, A.; Ball, A.; Bluche, T.; Caulier, A.; Leroy, D.; Doumouro, C.; Gisselbrecht, T.; Caltagirone, F.; Lavril, T.; Primet, M.; and Dureau, J. 2018. Snips Voice Platform: an embedded Spoken Language Understanding system for private-by-design voice interfaces. *CoRR*.

Gangadharaiah, R.; and Narayanaswamy, B. 2019. Joint Multiple Intent Detection and Slot Labeling for Goal-Oriented Dialog. In *NAACL*.

Goo, C.; Gao, G.; Hsu, Y.; Huo, C.; Chen, T.; Hsu, K.; and Chen, Y. 2018. Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. In *NAACL*.

Hemphill, C. T.; Godfrey, J. J.; and Doddington, G. R. 1990. The ATIS Spoken Language Systems Pilot Corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, USA, June 24-27, 1990*.

Hu, L.; Yang, T.; Shi, C.; Ji, H.; and Li, X. 2019. Heterogeneous Graph Attention Networks for Semi-supervised Short Text Classification. In *EMNLP*.

Kenton, J. D. M.-W. C.; and Toutanova, L. K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *AACL*.

Kim, B.; Ryu, S.; and Lee, G. G. 2017. Two-stage multi-intent detection for spoken language understanding. *Multim. Tools Appl.*

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv*.

Li, J.; Monroe, W.; Ritter, A.; Galley, M.; Gao, J.; and Jurafsky, D. 2016. Deep reinforcement learning for dialogue generation. *arXiv*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv*.

Lu, J.; Ye, X.; Ren, Y.; and Yang, Y. 2022. Good, better, best: Textual distractors generation for multiple-choice visual question answering via reinforcement learning. In *CVPR*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E. Z.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*.

Qin, L.; Liu, T.; Che, W.; Kang, B.; Zhao, S.; and Liu, T. 2021a. A Co-Interactive Transformer for Joint Slot Filling and Intent Detection. In *ICASSP*.

Qin, L.; Wei, F.; Xie, T.; Xu, X.; Che, W.; and Liu, T. 2021b. GL-GIN: Fast and Accurate Non-Autoregressive Model for Joint Multiple Intent Detection and Slot Filling. In *ACL*.

Qin, L.; Xie, T.; Che, W.; and Liu, T. 2021c. A Survey on Spoken Language Understanding: Recent Advances and New Frontiers. In *IJCAI*.

Qin, L.; Xu, X.; Che, W.; and Liu, T. 2020. Towards Fine-Grained Transfer: An Adaptive Graph-Interactive Framework for Joint Multiple Intent Detection and Slot Filling. In *EMNLP*.

Rao, M.; Dheram, P.; Tiwari, G.; Raju, A.; Droppo, J.; Rastrow, A.; and Stolcke, A. 2021. Do as i mean, not as i say: Sequence loss training for spoken language understanding. In *ICASSP*.

Rennie, S. J.; Marcheret, E.; Mroueh, Y.; Ross, J.; and Goel, V. 2017. Self-critical sequence training for image captioning. In *CVPR*.

Shao, C.; Feng, Y.; Zhang, J.; Meng, F.; Chen, X.; and Zhou, J. 2019. Retrieving sequential information for non-autoregressive neural machine translation. *arXiv*.

Song, M.; Yu, B.; Li, Q.; Wang, Y.; Liu, T.; and Xu, H. 2022. Enhancing Joint Multiple Intent Detection and Slot Filling with Global Intent-Slot Co-occurrence. In *EMNLP*.

Tur, G.; and De Mori, R. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Wang, J.; Wei, K.; Radfar, M.; Zhang, W.; and Chung, C. 2021a. Encoding Syntactic Knowledge in Transformer Encoder for Intent Detection and Slot Filling. In *AAAI*.

Wang, Y.; Patel, A.; Shen, Y.; and Jin, H. 2018a. A Deep Reinforcement Learning Based Multimodal Coaching Model (DCM) for Slot Filling in Spoken Language Understanding (SLU). In *Interspeech*.

Wang, Y.; Wei, Z.; Zhou, Y.; and Huang, X.-J. 2018b. Automatic essay scoring incorporating rating schema via reinforcement learning. In *EMNLP*.

Wang, Z.; Liu, X.; Yang, P.; Liu, S.; and Wang, Z. 2021b. Cross-lingual Text Classification with Heterogeneous Graph Neural Network. In *ACL*.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*.

Wu, D.; Ding, L.; Lu, F.; and Xie, J. 2020. SlotRefine: A Fast Non-Autoregressive Model for Joint Intent Detection and Slot Filling. In *EMNLP*.

Wu, J.; Harris, I. G.; and Zhao, H. 2021. Spoken Language Understanding for Task-oriented Dialogue Systems with Augmented Memory Networks. In *NAACL*.

Wu, L.; Tian, F.; Qin, T.; Lai, J.; and Liu, T.-Y. 2018. A study of reinforcement learning for neural machine translation. *arXiv*.

Xing, B.; and Tsang, I. W. 2022a. Co-guiding Net: Achieving Mutual Guidances between Multiple Intent Detection and Slot Filling via Heterogeneous Semantics-Label Graphs. In *EMNLP*.

Xing, B.; and Tsang, I. W. 2022b. Group is better than individual: Exploiting Label Topologies and Label Relations for Joint Multiple Intent Detection and Slot Filling. In *EMNLP*.

Xiong, C.; Zhong, V.; and Socher, R. 2018. DCN+: Mixed Objective And Deep Residual Coattention for Question Answering. In *ICLR*.

Xu, J.; Sun, X.; Zeng, Q.; Ren, X.; Zhang, X.; Wang, H.; and Li, W. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. *arXiv*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R. R.; and Le, Q. V. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *NeurIPS*.

Zhang, Z.; Zhou, Z.; and Wang, Y. 2022. SSEGCN: Syntactic and Semantic Enhanced Graph Convolutional Network for Aspect-based Sentiment Analysis. In *NAACL*.

Zhong, V.; Xiong, C.; and Socher, R. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv*.

Zhu, Z.; Cheng, X.; Huang, Z.; Chen, D.; and Zou, Y. 2023a. Enhancing Code-Switching for Cross-lingual SLU: A Unified View of Semantic and Grammatical Coherence. In *EMNLP*.

Zhu, Z.; Cheng, X.; Huang, Z.; Chen, D.; and Zou, Y. 2023b. Towards Unified Spoken Language Understanding Decoding via Label-aware Compact Linguistics Representations. In *ACL*.

Zhu, Z.; Xu, W.; Cheng, X.; Song, T.; and Zou, Y. 2023c. A Dynamic Graph Interactive Framework with Label-Semantic Injection for Spoken Language Understanding. In *ICASSP*.

Zhuang, X.; Cheng, X.; and Zou, Y. 2023. Towards Explainable Joint Models via Information Theory for Multiple Intent Detection and Slot Filling. In *AAAI*.