ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230538

Joint Multiple Intent Detection and Slot Filling with Supervised Contrastive Learning and Self-Distillation

Nguyen Anh Tu, Hoang Thi Thu Uyen, Tu Minh Phuong and Ngo Xuan Bach[*]

Department of Computer Science,
Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
{anhtunguyen446,thuuyenptit}@gmail.com;{phuongtm,bachnx}@ptit.edu.vn

Abstract. Multiple intent detection and slot filling are two fundamental and crucial tasks in spoken language understanding. Motivated by the fact that the two tasks are closely related, joint models that can detect intents and extract slots simultaneously are preferred to individual models that perform each task independently. The accuracy of a joint model depends heavily on the ability of the model to transfer information between the two tasks so that the result of one task can correct the result of the other. In addition, since a joint model has multiple outputs, how to train the model effectively is also challenging. In this paper, we present a method for multiple intent detection and slot filling by addressing these challenges. First, we propose a bidirectional joint model that explicitly employs intent information to recognize slots and slot features to detect intents. Second, we introduce a novel method for training the proposed joint model using supervised contrastive learning and self-distillation. Experimental results on two benchmark datasets MixATIS and MixSNIPS show that our method outperforms state-of-the-art models in both tasks. The results also demonstrate the contributions of both bidirectional design and the training method to the accuracy improvement. Our source code is available at https://github.com/anhtunguyen98/BiSLU.

1 Introduction

Spoken language understanding (SLU) is a core component of task-oriented dialogue systems - an important class of natural language processing (NLP) applications. With the aim of capturing the semantics of user utterances, SLU consists of two main tasks: 1) multiple *intent detection*, which identifies the intents or desires of the user in utterances; and 2) *slot filling*, which extracts slots that provide necessary information to fulfill those intents [10]. Figure 1 shows an annotated sample from the MixATIS corpus [25], a benchmark dataset widely used in the SLU research community. Given the user utterance "*Show the cheapest round trip tickets and airlines fly from atlanta to washington DC*", an SLU component will detect two intents, i.e., airfare and airline, and extract five slots, i.e., "cheapest" (cost_relative), "round trip" (round_trip), "atlanta" (fromloc.city_name), "Washington" (toloc.city_name), and "DC" (toloc.state_code).

Traditional approaches to multiple intent detection and slot filling consider them as independent problems, namely semantic classification and sequence labelling, and use a separate model for each task. Those approaches ignore the fact that intents and slots are related to each other. For example, intent airfare often requires some specific types of slots such as fromloc.city_name and toloc.city_name. At the same time, slots fromloc.city_name and toloc.city_name tend to occur in an utterance with intent airfare or flight. To utilize such relationships between intents and slots, recent approaches rely on joint models that can detect intents and extract slots simultaneously. Thanks to recent advancements in deep learning, various deep neural network-based joint models have been developed and achieved state-of-the-art results in benchmark SLU datasets [1, 3, 9, 27].

There are several challenges when designing a successful joint model. First, the accuracy of such a model depends heavily on the way information is transferred between the two tasks. The model should be designed so that the result of one task can be used to correct or improve the result of the other, and vice versa. Existing joint models, however, are often unidirectional. Other models transfer information between two tasks implicitly. Those models cannot fully exploit the relationships between the two tasks to get the improved results. In this work, we propose a bidirectional joint model that explicitly utilizes intent information to extract slots and slot features to detect intents (BiSLU). Given an utterance, our model employs a language model-based encoder to generate its representation and intermediate (soft) intents, which are utilized to extract slots with a biaffine classifier. The model then uses slot features as well as the utterance representation to predict the final intents.

Another challenge with a joint model is how to train the model effectively. Such a model typically has several outputs, and the information is circulated between the intent and slot components, making the selection of training objectives non-trivial. Traditionally, intent detection and slot filling are cast as supervised learning problems. In this paper, we propose a training framework that includes also contrastive learning and self-distillation. Contrastive learning is a powerful technique for various tasks in different learning scenarios, including self-supervised learning and supervised learning [4, 14, 17, 33]. Contrastive learning methods aim to produce better representations of data by maximizing agreement between a data sample (a.k.a. the "anchor") and its different augmentations (or views), while maximizing disagreement between the anchor and negative samples, e.g. other samples from the same batch. Following the success of contrastive learning in computer vision, several studies have investigated contrastive learning for NLP tasks [12, 26, 32]. A key advantage of contrastive learning is that machine learning models in different settings can be trained effectively by integrating suitable contrastive losses. In this paper, we propose a method to generate augmenta-

[*] Corresponding Author

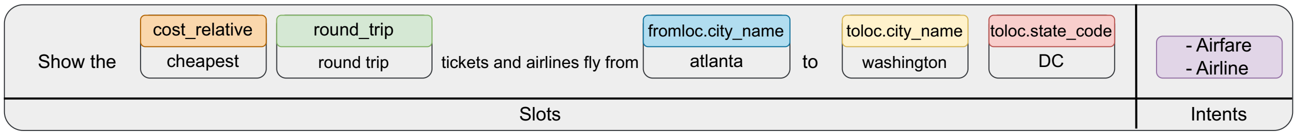

Figure 1. An example from the MixATIS dataset.

tions for the intent detection and slot filling task and integrate the contrastive loss with the original classification loss.

The final component of our method is self-distillation. Here, we use self-distillation to transfer knowledge from the final intents to the intermediate intents, which leads to an improvement in the joint model's performance. With contrastive learning and self-distillation, we train our proposed model with a joint loss function consisting of five components: intent loss, slot loss, contrastive intent loss, contrastive slot loss, and self-distillation loss between the intermediate intents and the final intents.

We verify the effectiveness of the overall method and the contribution of each component in two benchmark datasets: MixATIS and MixSNIPS [25]. The model achieves a new state-of-the-art in joint multiple intent detection and slot filling with relative error reductions ranging from 3% to 22%. The experiments also demonstrate the contribution of contrastive learning and self-distillation components on the accuracy of the final model.

Our main contributions are summarized below:

1. We propose a bidirectional joint model for multiple intent detection and slot filling (BiSLU).
2. We introduce a novel method for training the proposed model effectively using supervised contrastive learning and self-distillation.
3. We empirically show the efficacy of BiSLU as well as the proposed training method on two benchmark datasets MixATIS and MixSNIPS.

In the following, we first review related work on joint multiple intent detection and slot filling in Section 2. We next introduce our proposed method in Section 3, including the bidirectional joint model and the training method. We then describe experimental results and analyses in Section 4. Finally, we conclude the paper in Section 5.

2 Related Work

The interdependence between intent detection and slot filling has been a subject of considerable academic interest, and various models that address both tasks simultaneously within a single framework have been proposed [13, 22, 23]. These models, however, exhibit a significant limitation in their intent detection modules, as they are restricted to single-intent utterances, which may be insufficient for real-world applications where multi-intent utterances are prevalent.

Several studies have been conducted for detecting multiple intents of utterances. Gangadharaiah and Narayanaswamy [10] pioneered a model that concurrently addresses multiple intent identification and slot filling tasks using an attention-based network. Qin et al. [25] proposed an adaptive graph interactive framework (AGIF) that leverages a fine-grained approach to integrate multi-intent information into slot filling. Chen et al. [2] presented a novel self-distillation joint model (SDJN) for multi-intent detection and slot filling. This approach involves mutual intent and slot data sharing for cyclical optimization and employs self-distillation by considering the decoder slots as soft labels for the initial decoder slots.

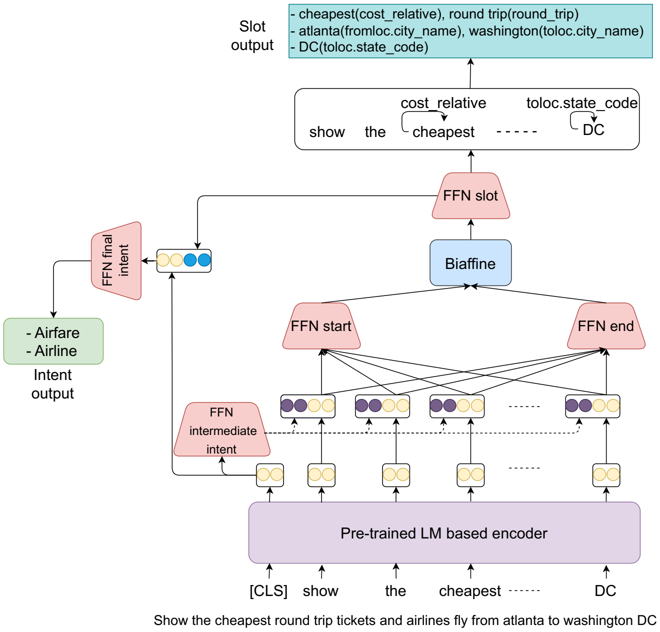

Figure 2. The architecture of the proposed model.

In recent years, contrastive learning has gained attraction, particularly for self-supervised representation learning, resulting in state-of-the-art performance in unsupervised training of deep image models [4]. Khosla et al. [17] extended contrastive learning to supervised setting, which allows us to leverage label information. In supervised contrastive learning, samples with the same label as the anchor in the batch are considered as positive samples, and the rest is regarded as negative ones. To work with multi-label data, Zhang et al. [33] introduced a multi-label contrastive learning framework, which has been shown to be effective in various computer vision tasks. Contrastive learning has been also applied successfully in various NLP tasks, including text classification, sentence embedding, and question answering [16, 18, 28, 30, 31].

Compared to previous studies, we also develop a joint model that deals with the both tasks simultaneously. However, our method has several significant differences: 1) we introduce a novel architecture for bidirectional joint multiple intent detection and slot filling; 2) we employ supervised contrastive learning and self-distillation to train the proposed joint model effectively. To the best of our knowledge, this is the first attempt to incorporate contrastive learning into SLU. Furthermore, we use a new self-distillation technique, which is different from the one described in Chen et al. [2]. As shown in experiments, our method is superior to state-of-the-art multi-intent SLU methods.

3 Our Method

We first present our bidirectional joint model for SLU (BiSLU) (Section 3.1). We then describe our training method with supervised contrastive learning (Section 3.2), self-distillation (Section 3.3), and a joint training procedure (Section 3.4).

3.1 Bidirectional Joint Model

The architecture of our proposed joint model is illustrated in Figure 2, consisting of four components: encoder, intermediate intent detection, slot classifier, and final intent detection. Below we describe each component in detail.

3.1.1 Encoder

Given an input utterance consisting of n words $w_1 w_2 \dots w_n$, we prepend a special classification token [CLS], represented by w_0, to form the input sequence $w_0 w_1 w_2 \dots w_n$. Our encoder utilizes a BERT-base model [8] to generate contextualized word embeddings:

$$\mathbf{c}i = BERT\_Encoder(w_{0:n}, i),$$

where $\mathbf{c}_i \in \mathbb{R}^d$ $(1 \leq i \leq n)$ corresponds to the embedding of word w_i, $\mathbf{c}_0$ represents the embedding of the token [CLS] that signifies the entire utterance, and d indicates the embedding size. It is important to note that the encoder segments words into sub-words for efficient rare word modeling. The contextual embedding of a word is obtained by summing up the embeddings of its constituent sub-words.

3.1.2 Intermediate Intent Detection & Word Representations

We feed the embedding $\mathbf{c}_0$ into a feed-forward neural network (FFN) to predict intermediate intents:

$$\mathbf{p} = Sigmoid(FFN^{Intermediate\_Intent}(\mathbf{c}_0)),$$

where $\mathbf{p} \in \mathbb{R}^l$ is a score vector of each intent probability, and l denotes the number of intents.

To create a word representation, the intermediate intent vector $\mathbf{p}$ is concatenated with each contextualized embedding $\mathbf{c}_i$ as follows:

$$\mathbf{v}_i = \mathbf{p} \oplus \mathbf{c}_i,$$

where $\mathbf{v}_i \in \mathbb{R}^{l+d}$ is the final representation of the i^{th} word, and $\oplus$ denotes the concatenation operation.

3.1.3 Slot Classifier

To extract slots, we employ a biaffine classifier, which provides a global view of the input sequence and, therefore, is effective for sequence tagging and related tasks [19]. We use two feed-forward networks, FFN^{start} and FFN^{end}, to create different representations for the start/end position of slots. The outputs of FFN^{start} and FFN^{end} at position i are denoted by $\mathbf{g}_i^{start}$ and $\mathbf{g}_i^{end}$, respectively:

$$\mathbf{g}_i^{start} = FFN^{start}(\mathbf{v}_i),$$
$$\mathbf{g}_i^{end} = FFN^{end}(\mathbf{v}_i).$$

For each start-end candidate slot (i, j) $1 \leq i \leq j \leq n$, we apply the biaffine classifier:

$$\begin{aligned}\mathbf{z}_{i,j} &= \text{Biaffine}(\mathbf{g}_i^{start}, \mathbf{g}_j^{end}) \\ &= (\mathbf{g}_i^{start})^\top \mathbf{U} \mathbf{g}_j^{end} + \mathbf{W}(\mathbf{g}_i^{start} \oplus \mathbf{g}_j^{end}) + \mathbf{b},\end{aligned}$$

where $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{b}$ are a $k \times s \times k$ tensor, a $s \times 2k$ matrix, and a bias vector, respectively; k denotes the dimension of the output layers of two FFNs; and s is the dimension of vector $\mathbf{z}_{i,j}$, which represents the segment feature going through a slot classifier FFN^{slot}:

$$\mathbf{r}_{i,j} = FFN^{slot}(\mathbf{z}_{i,j})$$

where $\mathbf{r}_{i,j} \in \mathbb{R}^{c+1}$, and c denotes the number of slot labels ($c + 1$ because we add a special label for non-slot segments). Finally, vector $\mathbf{r}_{i,j}$ is fed into a softmax function to produce the probability scores:

$$\mathbf{q}_{i,j}(t) = \frac{\exp(\mathbf{r}_{i,j}(t))}{\sum_{t'=1}^{c+1} \exp(\mathbf{r}_{i,j}(t'))}.$$

The slot label of segment (i, j) can be determined as:

$$\arg\max_t \mathbf{q}_{i,j}(t).$$

Among overlapping predicted slots, if any, we keep only the slot with the highest score and discard the rest.

3.1.4 Final Intent Detection

We create a soft slot vector from the output matrix of the slot classifier:

$$\mathbf{h} = Softmax(\sum_{i=1}^{n} \sum_{j=i}^{n} \mathbf{r}_{i,j})$$

The output vector $\mathbf{h} \in \mathbb{R}^{c+1}$ is concatenated with the representation vector of the token [CLS] ($\mathbf{c}_0$) to form the input $\mathbf{x} \in \mathbb{R}^{d+c+1}$ to predict the final intent:

$$\mathbf{x} = \mathbf{c}_0 \oplus \mathbf{h},$$

$$\mathbf{p}_{final\_intent} = Sigmoid(FFN^{Final\_Intent}(\mathbf{x})),$$

where FFN^{Final_Intent} is a feed forward neural network, and $\mathbf{p}_{final\_intent} \in \mathbb{R}^l$ is the score vector of final intents. Recall that l and c are the number of intent labels and the number of slot labels, respectively. To obtain the final intents, we apply a threshold $0 < t_I < 1$ and select all intents m $(1 \leq m \leq l)$ whose probability is greater than t_I.

3.2 Intent and Slot Contrastive Learning

Supervised contrastive learning (SCL) has achieved remarkable success in computer vision and NLP [17, 33, 32, 26], aiming to maximize similarities between instances from the same class while minimizing similarities between instances from different classes. In general, supervised contrastive learning consists of two steps:

1. **Positive/negative sample construction**. Given an anchor, i.e., a training sample in a mini-batch, generates/selects positive and negative samples for the anchor.
2. **Loss function design**. Builds an appropriate loss function for generated positive/negative samples.

The idea of our SCL method for joint multiple intent detection and slot filling is shown in Figure 3, on the left-hand side. Below we describe those two steps in detail.

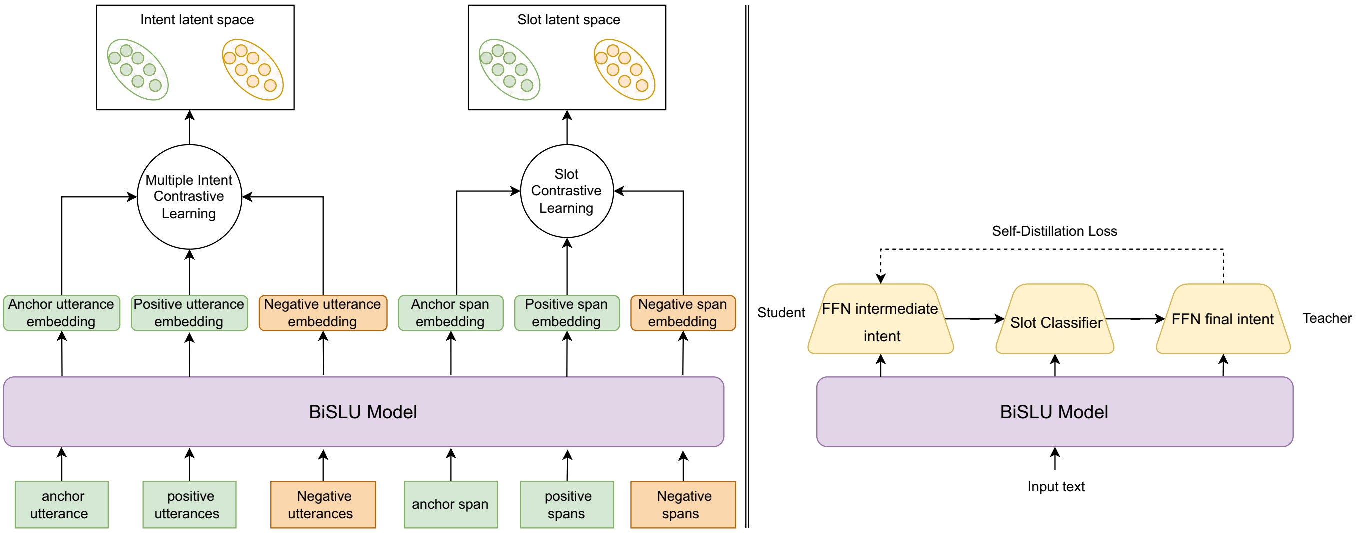

Figure 3. Supervised contrastive learning (left) and self-distillation (right) for joint multiple intent detection and slot filling.

3.2.1 *Positive/Negative Sample Construction*

- **Positive samples:** To create positive samples for a given anchor sample $x = w_1 w_2 \ldots w_n$, we generate positive representations for both the utterance and spans as follows:
  - **Positive utterances**: For a given anchor utterance, we encode the input utterance V times by applying different dropout rates in the encoder. Each of these V encodings will generate a slightly different representation of the input utterance, resulting in a multi-viewed mini-batch. All utterance representations in the multi-viewed mini-batch, (including augmented utterance representations and other utterances which have the same label as the anchor) are considered as positive samples. A positive pair for the utterance will be formed by pairing the anchor utterance representation $\mathbf{c}^{cls}$, with a corresponding positive sample denoted by $\mathbf{c}_p^{cls}$.
  - **Positive spans**: Similarly, we also encode the input utterance V times. For each encoded utterance, we extract span representations based on the output of the biaffine layer and the start/end indices of the spans. The span representation of the anchor span is $\mathbf{z}_{i,j}$, where (i, j) are the word indices in the input utterance and $1 \leq i \leq j \leq n$. All span representations within a multi-viewed mini-batch are regarded as positive spans for the anchor span. We form a positive pair for the span by pairing the anchor span representation $\mathbf{z}_{i,j}$ with a corresponding positive sample represented by $\mathbf{z}_{p(i,j)}$.
- **Negative samples:** For a given anchor utterance, any instance in the multi-viewed mini-batch that has a different intent class from the original sample is selected as a negative sample. Similarly, for a given anchor span, any instance in the multi-viewed mini-batch that has a different span class from the original sample is selected as a negative sample.

3.2.2 *Slot Contrastive Loss*

Let $P(i, j)$ and $A(i, j)$ denote the set of all positive samples and the set of all positive and negative samples corresponding to span (i, j), and I is the collection of all the training spans, we define a supervised contrastive loss for slots as follows:

$$\mathcal{L}_{sf\_scl} = \sum_{(i,j)\in I} \frac{-1}{|P(i,j)|} \sum_{p\in P(i,j)} \log \frac{f\left(\mathbf{z}_{i,j}, \mathbf{z}_{p(i,j)}\right)}{\sum_{k\in A(i,j)} f\left(\mathbf{z}_{i,j}, \mathbf{z}_{k(i,j)}\right)}$$

where $f\left(\mathbf{z}_{i,j}, \mathbf{z}_{p(i,j)}\right) = \exp(\mathbf{z}_{i,j} \cdot \mathbf{z}_{p(i,j)}/\tau)$ calculates the similarity between $\mathbf{z}_{i,j}$ and $\mathbf{z}_{p(i,j)}$, and τ denotes the temperature, a scalar to stabilize the calculation. Recall that $\mathbf{z}_{i,j}$ and $\mathbf{z}_{p(i,j)}$ are vector representations of span (i, j) of the anchor sample and a positive sample, respectively, created from the output matrix of the biaffine layer. Here, i and j are indices, $1 \leq i \leq j \leq n$, and n denotes the length of an input utterance.

3.2.3 *Multiple Intent Contrastive Loss*

Although supervised contrastive learning can distinguish between multiple positive pairs, it is only designed for single labels. Following [33], we define M as the set of all intent labels, and $m \in M$ is a label in the label set. The loss for a pair of the anchor utterance, indexed by i, and a positive utterance of label m can be defined as follows:

$$L^{\text{pair}}\left(i, p_m^i\right) = \log \frac{f\left(\mathbf{c}_i^{cls}, \mathbf{c}_{p_m^i}^{cls}\right)}{\sum_{k\in A(i)} f\left(\mathbf{c}_{k_m^i}^{cls}, \mathbf{c}_i^{cls}\right)}$$

where $\mathbf{c}_i^{cls}$ and $\mathbf{c}_{p_m^i}^{cls}$ are the representation vector of the token [CLS] of the anchor sample and positive samples, respectively; and $A(i)$ denotes the set of all the positive and negative samples corresponding to sample i. The multi-label contrastive loss for intents can then be defined as follows:

$$\mathcal{L}_{id\_scl} = \sum_{m\in M} \frac{1}{|M|} \sum_{i\in I} \frac{-\lambda_m}{|P_m(i)|} \sum_{p_m\in P_m} L^{\text{pair}}\left(i, p_m^i\right)$$

where $\lambda_m = F(m)$ is a controlling parameter that applies a fixed penalty to each label; F is a scale function with m (i.e., exp or pow); and P_m denotes the set of all positive samples of the anchor utterance indexed by i.

3.3 Self-Distillation

Recall that our joint model conveys information bi-directionally: the intermediate intent probabilities serve as part of the input for the slot filling layer, and the predicted slot labels are then utilized to determine the final intents. Consequently, the quality of the intermediate intent detection module significantly impacts on the slot filling and final intent detection modules. We propose a self-knowledge distillation method within the joint training model, where the teacher model is the final intent detection layer, and the student model is the intermediate intent detection layer. Our idea is shown in Figure 3, on the right-hand side. We compute the representative distance by leveraging the hidden states of both the final intent detection and intermediate intent detection layers. During the training process, we aim to minimize this representative distance between the two hidden states ($\mathbf{h}_{intermediate\_intent}$ and $\mathbf{h}_{final\_intent}$) based on the logits-based distillation approach.

Conventional logits-based distillation approaches typically minimize the Kullback-Leibler (KL) divergence between the predicted probabilities, that is, the logits after applying the softmax function, of the teacher and student models. However, directly applying this method to multi-label learning (MLL) scenarios is challenging due to the inherent assumption that the predicted probabilities of all classes should sum to one, an assumption that is seldom valid in MLL cases. To address this limitation, we draw inspiration from the one-versus-all reduction concept and propose a multi-label distillation loss. This approach decomposes the original multi-label task into several binary classification problems and aims to minimize the divergence between the binary predicted probabilities of the teacher and student models. The formal definition of self-distillation loss is as follows:

$$\mathbf{h}_S = \mathbf{h}_{intermediate\_intent} = FFN^{Intermediate\_Intent}(\mathbf{c}_0),$$

$$\mathbf{h}_T = \mathbf{h}_{final\_intent} = FFN^{Final\_Intent}(\mathbf{x}),$$

$$\mathbf{p}_S = Sigmoid(\mathbf{h}_S),$$

$$\mathbf{p}_T = Sigmoid(\mathbf{h}_T),$$

$$\mathcal{L}_{sd} = KL(\mathbf{p}_S||\mathbf{p}_T) + KL(1-\mathbf{p}_S||1-\mathbf{p}_T)$$

where $KL(P||Q)$ denotes the Kullback-Leibler divergence of P from Q, two discrete probability distributions defined on the same sample space $\mathcal{X}$:

$$KL(P||Q) = \sum_{x\in\mathcal{X}} P(x) log(\frac{P(x)}{Q(x)}).$$

3.4 Joint Training

We define a joint loss function $\mathcal{L}$ of our bidirectional model as the weighted sum of the intent detection loss $\mathcal{L}_{id}$, the slot filling loss $\mathcal{L}_{sf}$, the slot contrastive loss $\mathcal{L}_{sf\_scl}$, the intent contrastive loss $\mathcal{L}_{id\_scl}$, and the self-distillation loss $\mathcal{L}_{sd}$:

$$\mathcal{L} = \lambda_1\mathcal{L}_{id} + \lambda_2\mathcal{L}_{sf} + \lambda_3\mathcal{L}_{id\_scl} + \lambda_4\mathcal{L}_{sf\_scl} + \lambda_5\mathcal{L}_{sd}$$

where $\mathcal{L}_{id}$ and $\mathcal{L}_{sf}$ use the standard cross-entropy loss; and $0 \leq \lambda_i \leq 1$ ($\sum_{i=1}^{5} \lambda_i = 1$) are hyperparameters, which are tuned using a development set in experiments.

4 Experiments

4.1 Data and Evaluation Methods

We performed experiments using two benchmarks multi-intent SLU datasets, namely MixATIS and MixSNIPS [25]. They are the multiple intent versions of the ATIS dataset [15] and the SNIPS dataset [6], which were created for single intent detection and slot filling. Both multi-intent SLU datasets exhibit a distribution of sentences containing 1-3 intents at proportions of [0.3, 0.5, 0.2]. In the MixATIS dataset, there are 13,162 training utterances, 759 validation utterances, and 828 testing utterances. Meanwhile, the MixSNIPS dataset consists of 39,776 training utterances, 2,198 validation utterances, and 2,199 testing utterances.

We evaluated the performance of different joint models for multiple intent detection and slot filling using three conventional metrics as in [10]: intent accuracy, slot F_1 score, and sentence-level semantic frame accuracy.

4.2 Models to Compare

We first conducted experiments to compare our proposed method with state-of-the-art multi-intent SLU models. For those models that were originally designed to address single-intent detection, including Attention BiRNN [20], Slot-Gated [11], Bi-Model [29], SF-ID [13] and Stack-Proagation [22], we cited the results reported by Qin et al. [25]. The comprehensive list of all the baseline models employed for the comparison is provided below:

- Attention BiRNN [20] : an alignment-based RNN model for joint slot filling and intent detection.
- Slot-Gated [11]: a unidirectional model that uses attention-based BiLSTMs with a slot-gated mechanism to leverage an intent context vector for improving slot filling.
- Bi-Model [29]: a bidirectional model for single intent detection and slot filling.
- SF-ID [13]: a bidirectional model with SF and ID subnets.
- Stack-Propagation [22]: a unidirectional model with stack-propagation which can directly use the intent information as an input for slot filling.
- Joint Multiple ID-SF [10]: a multi-task framework with a slot-gated mechanism for multiple intent detection and slot filling.
- AGIF [25]: an adaptive interaction network to achieve fine-grained multi-intent information integration.
- GL-GIN [24]: a non-autoregressive approach for joint multiple intent detection and slot filling.
- DGM [9]: a dynamic graph model for joint multiple intent detection and slot filling.
- SDJN [2]: a joint model for multi-intent SLU with self-distillation for slots.
- GISCo [27]: a graph neural network based on the global intent-slot co-occurrence to model the interaction between the two tasks.
- SSRAN [5]: a scope sensitive and result attentive model for multi-intent SLU based on Transformer.
- SLIM [1]: a multi-intent SLU framework that uses a slot-intent classifier to learn the many-to-one mapping between slots and intents based on BERT.
- TFMN [3]: a transformer-based threshold-free multi-intent SLU model.

Table 1. Performance comparison on the MixATIS and MixSNIPS datasets. The best values for each column are shown in bold

Model	**MixATIS**			**MixSNIPS**		
	Intent	**Slot**	**Sent**	**Intent**	**Slot**	**Sent**
Attention BiRNN [20]	74.6	86.4	39.1	95.4	89.4	59.5
Slot-Gated [11]	63.9	87.7	35.5	94.6	87.9	55.4
Bi-Model [29]	70.3	83.9	34.4	95.6	90.7	63.4
SF-ID [13]	66.2	87.4	34.9	95.0	90.6	59.9
Stack-Propagation [22]	72.1	87.8	40.1	96.0	94.2	72.9
Joint Multiple ID-SF [10]	73.4	84.6	36.1	95.1	90.6	62.9
AGIF [25]	74.4	86.7	40.8	95.1	94.2	74.2
GL-GIN [24]	76.3	88.3	43.5	95.6	94.9	75.4
DGM [9]	76.7	88.7	47.1	96.7	94.7	78.0
SDJN [2]	77.1	88.2	44.6	96.5	94.4	75.7
GISCo [27]	75.0	88.5	48.2	95.5	95.0	75.9
SSRAN [5]	77.9	89.4	48.9	**98.4**	95.8	77.5
SLIM [1]	78.3	88.5	47.6	97.2	96.5	84.0
TFMN [3]	79.8	88.0	50.2	97.7	96.4	84.7
Our model (BiSLU)	**81.5**	**89.4**	**51.5**	97.8	**97.2**	**85.4**

Table 2. Experimental results of different variants of our model. The best values for each column are shown in bold

Model (Variant)	**MixATIS**			**MixSNIPS**		
	Intent	**Slot**	**Sent**	**Intent**	**Slot**	**Sent**
Using intermediate intents	77.8	87.9	47.4	96.9	96.4	83.3
Intermediate intent removal	78.2	88.1	47.9	96.3	96.5	84.5
Intent-slot attention	79.1	88.5	48.7	96.9	96.2	84.7
BiSLU Softmax	78.9	87.5	47.3	95.8	95.5	83.7
BiSLU CRFs	78.7	87.6	47.6	96.8	95.9	84.4
BiSLU Biaffine	**79.5**	**88.8**	**49.2**	**97.3**	**96.5**	**84.8**

4.3 Experimental Setup

In this work, we developed models utilizing the PyTorch framework in conjunction with the HuggingFace library[1]. We employed the BERT base model[2] as pre-trained language models for our experimental purposes. Throughout all experiments, we established the maximum sequence length at 100, while setting the dimensions of the output layers of the feed-forward start/end networks and biaffine networks, k and s, to 300 and 200, respectively.

We trained the models using the AdamW optimizer [21], with default values for epsilon and weight decay in PyTorch (i.e., 1e-8). In order to ascertain the optimal hyper-parameters, we performed a grid search on the validation set, adjusting the AdamW initial learning rate within the range of {1e-5, 2e5, 3e-5, 4e-5, 5e-5}, the batch size within the range {8, 16, 32}, and the mixture weight λ_i within the range {0.05, 0.10, 0.15, . . . , 0.80}. We tuned the intent thresshold t_I and the number of views V within the ranges {0.3, 0.4, . . . , 0.8} and {1,2,3,4,5}, respectively.

For each model, we trained for 30 epochs, subsequently evaluating intent accuracy, slot F_1 score, and sentence-level semantic frame accuracy on the validation set after each epoch. The model version that yielded the highest sentence-level semantic frame accuracy was ultimately selected to apply to the test set.

4.4 Experimental Results

Table 1 shows the performance of joint models for multiple intent detection and slot filling on the MixATIS and MixSNIPS datasets. The highest values in each column are highlighted in bold. It is evident from the table that our proposed model outperformed all the other models across all evaluation metrics on both datasets, except for the intent accuracy on the MixSNIPS dataset, where our model achieved the second-best result. Specifically, our model achieved 81.5% intent accuracy, a slot F_1 score of 89.4%, and 51.5% sentence-level semantic frame accuracy on the MixATIS dataset. Compared with the second-best model, i.e., TMFN, our model improved 1.7% (8.4% error rate reduction), 1.4% (11.7% error rate reduction), and 1.3% (2.6% error rate reduction), respectively. On the MixSNIPS dataset, our model obtained 97.8% intent accuracy, a slot F_1 score of 97.2%, and 85.4% sentence-level semantic frame accuracy, which improved 0.1% (4.3% error rate reduction), 0.8% (22.2% error rate reduction) and 0.7% (4.6% error rate reduction) compared with TMFN, respectively.

4.5 Effects of Design Solutions

Next, we conducted experiments with six variants of our model to evaluate the effects of the model's design.

1. **Using intermediate intents**. We used the intermediate intents as the final intents.
2. **Intermediate intent removal**. We removed the intermediate intents from the model architecture.
3. **Intent-slot attention**. Instead of directly using the intermediate intent probabilities, we fed them into an intent-slot attention layer as in [7].
4. **BiSLU CRFs**. We casted the slot filling task as a sequence labeling problem with conditional random fields (CRFs).
5. **BiSLU Softmax**. We used the softmax function instead of using CRFs as in the previous variant.
6. **BiSLU Biaffine**. Our proposed model with the biaffine classifier.

Experimental results of the six variants on MixATIS and MixSNIPS datasets are shown in Table 2. Our proposed model BiSLU with the biaffine classifier achieved the best results on both datasets, demonstrating the reasonableness of our model architecture. The performance degradation of other variants also confirmed the impor-

[1] https://huggingface.co

[2] https://huggingface.co/bert-base-uncased

Table 3. Experimental results with different joint loss functions

Model	MixATIS			MixSNIPS		
	Intent	Slot	Sent	Intent	Slot	Sent
BiSLU (without SCL and self-distillation)	79.5	88.8	49.2	97.3	96.5	84.8
BiSLU + intent SCL	80.5	89.0	50.6	97.5	96.7	85.0
BiSLU + slot SCL	80.4	89.2	50.3	97.4	96.9	85.1
BiSLU + both SCL	80.9	89.1	51.1	97.5	97.0	85.3
BiSLU + both SCL + self-distillation	**81.5**	**89.4**	**51.5**	**97.8**	**97.2**	**85.4**

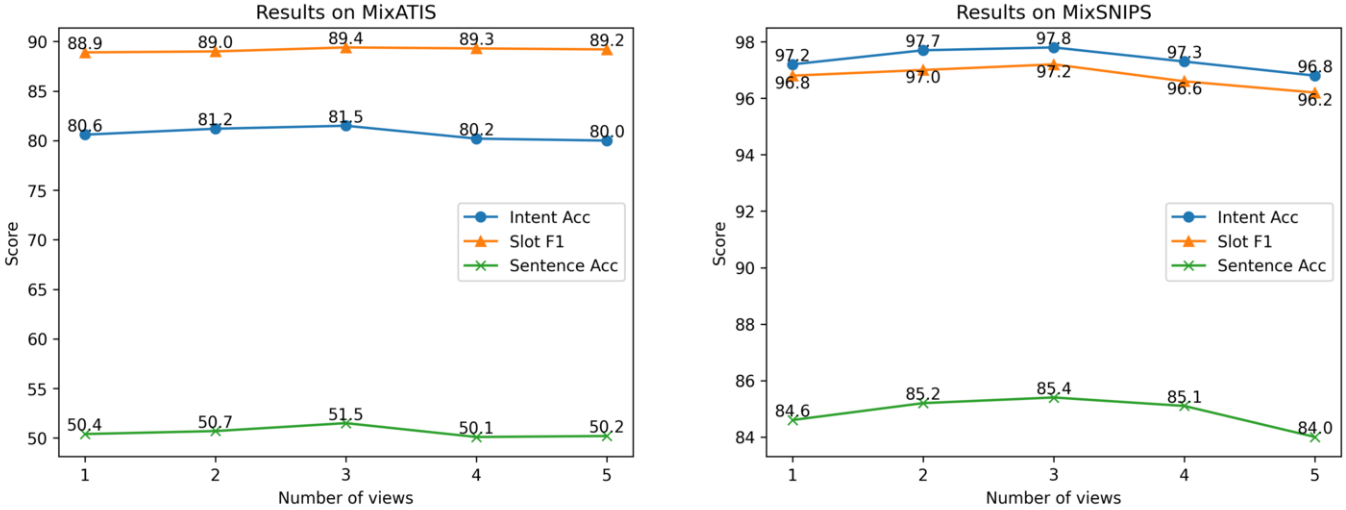

Figure 4. Effects of the number of views.

tance of the intermediate intents, the slot classifier, and the final intents.

To evaluate the efficacy of the proposed training method as well as the contribution of each type of loss, we investigated five variants of the training method:

1. BiSLU without SCL and self-distillation: $\mathcal{L} = \lambda_1 \mathcal{L}_{id} + \lambda_2 \mathcal{L}_{sf}$
2. BiSLU with SCL for intents: $\mathcal{L} = \lambda_1 \mathcal{L}_{id} + \lambda_2 \mathcal{L}_{sf} + \lambda_3 \mathcal{L}_{id\_scl}$
3. BiSLU with SCL for slots:$\mathcal{L} = \lambda_1 \mathcal{L}_{id} + \lambda_2 \mathcal{L}_{sf} + \lambda_4 \mathcal{L}_{sf\_scl}$
4. BiSLU with SCL for both intents and slots:$\mathcal{L} = \lambda_1 \mathcal{L}_{id} + \lambda_2 \mathcal{L}_{sf} + \lambda_3 \mathcal{L}_{id\_scl} + \lambda_4 \mathcal{L}_{sf\_scl}$
5. BiSLU with SCL for both intents and slots and self-distillation as well (our full method).

The results shown in Table 3 demonstrated that incorporating both intent and slot contrastive losses significantly improved the model's performance. The BiSLU model with both contrastive losses achieved 80.9% intent accuracy, 89.1% in the slot F_1 score, and 51.1% sentence-level semantic frame accuracy on the MixATIS dataset. Likewise, on the MixSNIPS dataset, the model got 97.5% intent accuracy, 97.0% in the slot F_1 score, and 85.3% sentence-level semantic frame accuracy. The most significant performance enhancement was observed when using the joint loss function with all five types of losses: the intent loss, the slot loss, the intent contrastive loss, the slot contrastive loss, and the self-distillation loss. These findings suggested that incorporating supervised contrastive learning for both intent and slot and self-distillation led to a more robust and accurate model for multi-intent spoken language understanding.

4.6 *Effects of the Number of Views*

In contrastive learning, the procedure of generating positive and negative samples is essential and has a considerable effect on the performance of the learning models. To investigate the effect of the data augmentation method on the performance of our proposed model, we conducted experiments with a different number of views V. Figure 4 shows the experimental results of our joint models on the MixATIS and the MixSNIPS datasets, with the number of views varying from 1 to 5. The results demonstrated that our model was relatively stable and achieved the best results with $V = 3$ on the both datasets.

5 Conclusion

We presented in this paper a bidirectional joint model for multiple intent detection and slot filling, two fundamental tasks in spoken language understanding. By predicting intermediate (soft) intents first, then slots, and the final intents, our model allows information to be transferred between the two tasks explicitly. We also introduced a novel training framework that includes supervised contrastive learning and self-distillation. Using a joint loss function consisting of different types of losses, i.e., intent loss, slot loss, intent contrastive loss, slot contrastive loss, and self-distillation loss between the intermediate and the final intents, the proposed model can be trained effectively. We empirically showed that our model outperformed state-of-the-art methods in both identifying intents and extracting slots on two benchmark datasets. Experimental results also demonstrated: 1) the reasonableness of our model's design with the intermediate and the final intents as well as the using of a biaffine classifier for extracting slots; 2) the contribution of the contrastive learning and self-distillation components on the performance of the proposed model.

Acknowledgement

This work is supported by Posts and Telecommunications Institute of Technology, Hanoi, Vietnam.

References

[1] Fengyu Cai, Wanhao Zhou, Fei Mi, and Boi Faltings, 'Slim: Explicit slot-intent mapping with bert for joint multi-intent detection and slot filling', in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7607–7611. IEEE, (2022).

[2] Lisong Chen, Peilin Zhou, and Yuexian Zou, 'Joint multiple intent detection and slot filling via self-distillation', in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7612–7616. IEEE, (2022).

[3] Lisung Chen, Nuo Chen, Yuexian Zou, Yong Wang, and Xinzhong Sun, 'A transformer-based threshold-free framework for multi-intent NLU', in *Proceedings of the International Conference on Computational Linguistics*, pp. 7187–7192, Gyeongju, Republic of Korea, (October 2022). International Committee on Computational Linguistics.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, 'A simple framework for contrastive learning of visual representations', in *Proceedings of the International Conference on Machine Learning*, pp. 1597–1607. PMLR, (2020).

[5] Lizhi Cheng, Wenmian Yang, and Weijia Jia, 'A scope sensitive and result attentive model for multi-intent spoken language understanding', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 12691–12699, (2023).

[6] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau, 'Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces', *arXiv preprint arXiv:1805.10190v3*, (2018).

[7] Mai Hoang Dao, Thinh Hung Truong, and Dat Quoc Nguyen, 'Intent detection and slot filling for vietnamese', in *Proceedings of Interspeech*, (2021).

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 'BERT: Pre-training of deep bidirectional transformers for language understanding', in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 4171–4186, (June 2019).

[9] Zeyuan Ding, Zhihao Yang, Hongfei Lin, and Jian Wang, 'Focus on interaction: A novel dynamic graph model for joint multiple intent detection and slot filling.', in *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, pp. 3801–3807, (2021).

[10] Rashmi Gangadharaiah and Balakrishnan Narayanaswamy, 'Joint multiple intent detection and slot labeling for goal-oriented dialog', in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 564–569, (2019).

[11] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen, 'Slot-gated modeling for joint slot filling and intent prediction', in *Proceedings of the North American Chapter of the Association for Computational Linguistics(NAACL), Volume 2 (Short Papers)*, pp. 753–757, (2018).

[12] Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov, 'Supervised contrastive learning for pre-trained language model fine-tuning', in *Proceedings of the International Conference on Learning Representations*, (2021).

[13] E Haihong, Peiqing Niu, Zhongfu Chen, and Meina Song, 'A novel bi-directional interrelated model for joint intent detection and slot filling', in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5467–5471, (2019).

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, 'Momentum contrast for unsupervised visual representation learning', in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, (2020).

[15] Charles T. Hemphill, John J. Godfrey, and George R. Doddington, 'The ATIS spoken language systems pilot corpus', in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*, (1990).

[16] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih, 'Dense passage retrieval for open-domain question answering', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, (2020).

[17] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan, 'Supervised contrastive learning', *Proceedings of the Advances in Neural Information Processing Systems*, **33**, 18661–18673, (2020).

[18] Taeuk Kim, Kang Min Yoo, and Sang-goo Lee, 'Self-guided contrastive learning for BERT sentence representations', in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2528–2540, Online, (August 2021). Association for Computational Linguistics.

[19] Y. Li, Z. Li, M. Zhang, R. Wang, S. Li, and L. Si, 'Self-attentive biaffine dependency parsing', in *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, pp. 5067–5073, (2019).

[20] Bing Liu and Ian Lane, 'Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling', in *Proceedings of Interspeech*, pp. 685–689, (2016).

[21] Ilya Loshchilov and Frank Hutter, 'Decoupled weight decay regularization', in *Proceedings of the International Conference on Learning Representations (ICLR)*, (2019).

[22] Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu, 'A stack-propagation framework with token-level intent detection for spoken language understanding', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2078–2087, (2019).

[23] Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu, 'A co-interactive transformer for joint slot filling and intent detection', in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8193–8197. IEEE, (2021).

[24] Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu, 'GL-GIN: Fast and accurate non-autoregressive model for joint multiple intent detection and slot filling', in *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021) (Volume 1: Long Papers)*, pp. 178–188, (2021).

[25] Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu, 'AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling', in *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1807–1816, (2020).

[26] Nils Rethmeier and Isabelle Augenstein, 'A primer on contrastive pre-training in language processing: Methods, lessons learned, and perspectives', *ACM Computing Surveys*, **55**(10), (2023).

[27] Mengxiao Song, Bowen Yu, Li Quangang, Wang Yubin, Tingwen Liu, and Hongbo Xu, 'Enhancing joint multiple intent detection and slot filling with global intent-slot co-occurrence', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 7967–7977, (2022).

[28] Varsha Suresh and Desmond Ong, 'Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 4381–4394, (2021).

[29] Yu Wang, Yilin Shen, and Hongxia Jin, 'A bi-model based rnn semantic frame parsing model for intent detection and slot filling', in *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 309–314, (2018).

[30] Chenyu You, Nuo Chen, and Yuexian Zou, 'Self-supervised contrastive cross-modality representation learning for spoken question answering', in *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 28–39, Punta Cana, Dominican Republic, (November 2021). Association for Computational Linguistics.

[31] Dejiao Zhang, Shang-Wen Li, Wei Xiao, Henghui Zhu, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang, 'Pairwise supervised contrastive learning of sentence representations', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 5786–5798, (2021).

[32] Rui Zhang, Yangfeng Ji, Yue Zhang, and Rebecca J. Passonneau, 'Contrastive data and learning for natural language processing', in *Proceedings of the North American Chapter of the Association for Computational Linguistics, Tutorial Abstract*, pp. 39–47, (2022).

[33] Shu Zhang, Ran Xu, Caiming Xiong, and Chetan Ramaiah, 'Use all the labels: A hierarchical multi-label contrastive learning framework', in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 16660–16669, (2022).