Span-Based Semantic Role Labeling with Contrastive Learning

Anonymous ACL submission

Abstract

Contrastive learning is widely recognized for its ability to understand the relationships between data and map them into a high-dimensional feature space. In this study, we apply this technique to semantic role labeling, constructing a model that effectively captures the relationships between spans and labels and determines spans accurately. Our model integrates the characteristics of both a conventional span-based model, which predicts spans for labels, and a model that is comparable to state-of-the-art, which predicts labels for spans. In our experiments, we apply these models to NPCMJ-PT, a Japanese corpus that is annotated with semantic role labels and has about 52,500 entries. The semantic roles are defined with 32 types of labels such as Arg0, Arg1 and ArgM-LOC, which are similar to PropBank. The experimental results show that our model outperforms the conventional span-based models, achieving a highest F1 score of 81.2.

1 Introduction

Semantic role labeling (SRL) is a form of shallow semantic parsing whose goal is to discover the predicate-argument structure of each predicate in a given input sentence. Given a sentence, for each target predicate all the constituents in the sentence that fill a semantic role of the predicate have to be recognized. Typical semantic arguments include core arguments such as Agent, Patient, and Instrument, as well as adjunct arguments like Locative, Temporal, and Manner.

One prevalent approach is based on BIO tagging schemes, which (Zhou and Xu, 2015; He et al., 2017) have used with neural SRL models. Utilizing features generated by neural networks, they assign a BIO tag to each word: "B" to words at the beginning of an argument span, "I" to those inside a span, and "O" to words outside an argument span. Although this approach has achieved high accuracy, it reconstructs argument spans from the predicted BIO tags rather than directly predicting the spans. In another approach, labeled span modeling (Koomen et al., 2005), the models first identify candidate argument spans and then classify each span into one of the semantic role labels. Several effective methods have been proposed for instance, such as structural constraint inference by using integer linear programming (Punyakanok et al., 2008) or dynamic programming (Täckström et al., 2015; Zhou and Xu, 2015). One advantage of this approach is that it allows us to design and utilize span-level features, which leads to the capture of rich contextual information and interactions between different parts of the text. However, identifying the appropriate spans from many candidates remains challenging, and has thus lagged behind the state-of-the-art performance of BIO-based neural models. Another approach is based on span-based scoring for semantic arguments (Ouchi et al., 2018), which has demonstrated a high performance comparable to state-of-the-art models on the CoNLL-2005 dataset. This approach also employs span-level features for span identification; however, it differs in that it predict spans for semantic role labels. Consequently, while it allows for narrowing down candidate spans more effectively than other span-based modeling approaches, it necessitates learning the appropriate spans from a very broad range of candidates.

In light of this background, we propose a model that, while utilizing span-level features, can effectively learn both spans for labels and labels for spans. Specifically, we prepare feature spaces for both spans and labels and link them appropriately, enabling the learning of both feature spaces through contrastive learning. During decoding, we compute the scores based on the similarity of each feature space, allowing for the appropriate selection of spans without relying solely on the prediction probability of one side.

In our experiments, we focus on the span-based task of Japanese SRL as NPCMJ-PT (Takeuchi et al., 2020) contains span information of arguments with PropBank-style semantic roles. The sense repository is publicly available on the web as the Predicate-Argument Structure Thesaurus[1],which defines the frames of predicates involving verbs, deverbal nouns, and adjectives with example sentences in Japanese. The experimental results show that our model outperforms the aforementioned conventional span-based models, achieving a highest F1 score of 81.2. Our contributions are the proposal of a new SRL approach employing contrastive learning, demonstrating that it outperforms conventional models, and utilizing a relatively unexplored PropBank-style Japanese dataset (NPCMJ-PT).

2 Related Work

The span-based SRL task is usually considered a sequence labeling problem (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2017) and often employs BIO tagging with CRF. Another approach is labeled span modeling (Koomen et al., 2005). Notable models in this area include (Täckström et al., 2015; Zhou and Xu, 2015), and models that also utilize span features (Ouchi et al., 2018) have demonstrated very high performance. Recently, there are approaches that predict the start and end positions and labels of spans using separate classifiers based on word representations (Kurita et al., 2022).

There is also active research on SRL that utilizes syntactic information. Traditionally, syntactic structure was considered essential for SRL models (Gildea and Palmer, 2002; Punyakanok et al., 2008). However, until recently, models that utilize deep neural network architectures have surpassed syntax-aware architectures without explicitly incorporating syntactic structure. Nevertheless, several studies (Zhou et al., 2020; Strubell et al., 2018; He et al., 2017; Marcheggiani and Titov, 2017) argue that deep neural network models can benefit from integrating syntactic information rather than ignoring it. Additionally, it has been demonstrated that providing both syntactic structures and dependency tree structures (Fei et al., 2021) contributes to performance improvement. Given this background, (Mohammadshahi and Henderson, 2023) proposed an effective way to incorporate auxiliary syntactic information into deep learning architectures for SRL. Recently, considering tree structures within arguments has been shown to be effective (Zhang et al., 2022), and utilizing various forms of knowledge, such as syntactic structures and part-of-speech tags (Tian et al., 2022), has achieved state-of-the-art results. While various works have utilized structural knowledge, one of the significant reasons for their improvement is the use of high-performance parsers or the provision of gold-standard syntactic structures. This approach may not necessarily be applicable to Japanese SRL. Therefore, we propose a new model that leverages only span features to improve performance without relying on such structural information.

[1]Predicate-Argument Structure Thesaurus: `https://pth.cl.cs.okayama-u.ac.jp/testp/pth/Vths`

3 Models

The key idea of our model is that it effectively learns the span and label feature spaces by minimizing the distance between features of similar span-label pairs and maximizing the distance between features of dissimilar span-label pairs. In CLIP (Radford et al., 2021), the feature space is learned by using contrastive learning with pairs of images and texts, making significant contributions in that domain. Inspired by this, we prepare semantic role labels and labels representing other specific spans, enabling contrastive learning by linking them with appropriate spans.

While there are many span-based models, in this study, we employ the models from (Tulloch and Takeuchi, 2024). Thus, we refer to the model that adapts a typical labeled span modeling as the L4S model and the model based on the idea of (Ouchi et al., 2018) as the S4L model. We explain the implementation differences between these and the proposed model.

The flow from the input to the decoding of each model is depicted in the left diagram of Figure 1, with the blue box representing the span encoder and the red box representing the label encoder. The span encoder generates span representations by feeding the input into a language model and utilizing the obtained hidden states. The label encoder generates label representations by feeding label embeddings into an MLP layer. The L4S and S4L models feed the span representations obtained from the span encoder through the MLP layer and then calculate scores for each span and decode. The L4S and S4L models learn tasks that "predict labels

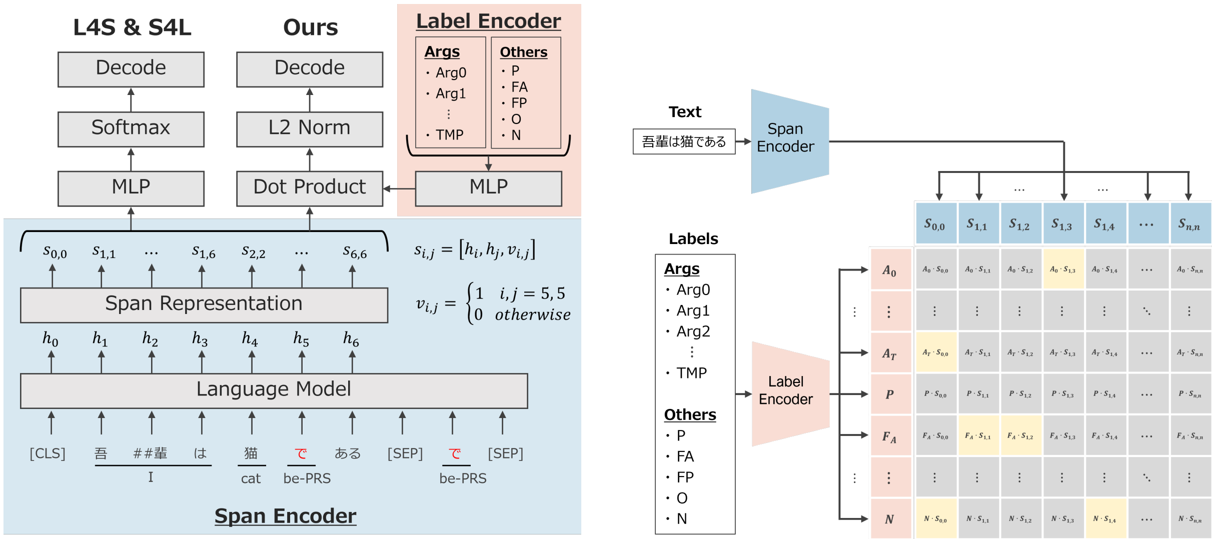

Figure 1: Overviw of the models. The left figure illustrates the flow from input to decoding for each model. The blue frame represents the span encoder, which generates span representations from the input. The red frame represents the label encoder, which generates label representations. The right figure depicts the process of inner product computation in the proposed model.

for spans" and "predict spans for labels," respectively, and the scores represent "the probability of a label for a span" and "the probability of a span for a label." In contrast, our model calculates scores and decodes by taking the dot product of the embeddings from the encoders, which have been passed through an L2 normalization layer. This means that the scores are represented by the cosine similarity between each span and label. The impact of these differences on the models is discussed in detail in Section 4.2. The following section describes each module that constitutes the model and the training process.

3.1 Task Explanation

Consider the following sentence with the set of correct argument labeled spans.

1 2 3 4 5 6
吾 輩 は 猫 で ある
[*A0*] [*A1*]

where the numbers are the position of each token. In this sentence, for the predicate "で," which means "be," "吾輩は," which means "I," is the A0 argument, and "猫," which means "cat," is the A1 argument.

The L4S model is tasked with predicting the label of a given span. Specifically, it predicts that the span "吾輩は" is labeled as A0 and the span "猫" is labeled as A1. Conversely, the S4L model is tasked with predicting the span of given semantic role. Here, it predicts that A0 argument is the span from 1 to 3 and A1 argument is the span from 5 to 5. Our model is designed to predict both the label and the span.

3.2 Word Representation

We exploit BERT (Devlin et al., 2019) as inputs for our model, which has demonstrated its effectiveness for a range of NLP tasks. Unlike English, Japanese does not use spaces to separate words: instead, sentences are written as continuous strings of characters. Thus, in the pre-trained BERT module[2] utilised in this study, the texts are first tokenized by MeCab[3] with the Unidic 2.1.2 dictionary[4] and then split into subwords by the WordPiece algorithm.

Given a sentence $X = [x_1, x_2, \ldots, x_k]$ where x_i is a character, the sentence is divided into subwords $W = [w_1, w_2, \ldots, w_n]$ by the tokenizer. For example, in Figure 1, the sentence 吾輩は猫である" is tokenized into "吾, ##輩, は, 猫, で, ある." After tokenization, we feed a token sequence $T = [\text{[CLS]}, t_1, \ldots, t_n, \text{[SEP]}, t_p, \text{[SEP]}]$ consisting of a CLS token, SEP tokens, and the target predicate t_p, into the pre-trained BERT to obtain hidden states $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n]$ which are used as word representations for span representations.

[2] cl-tohoku/bert-large-japanese-v2, Apache 2.0.
[3] MeCab, http://taku910.github.io/mecab/
[4] Unidic 2.1.2, https://clrd.ninjal.ac.jp/unidic/back_number.html

3.3 Span Representation

To represent a text span, we utilize the approach in (Li et al., 2021), which uses the concatenation of the word representations of the start and end points of the span. Additionally, we define a one-hot vector $\boldsymbol{V}$ to indicate the target predicate position, as

$$\boldsymbol{V} = [v_{0,0}, v_{1,1}, v_{1,2}, \ldots, v_{n,n}] \ , \quad (1)$$

where $v_{i,j}$ is 1 if the span (i, j) is a predicate, and 0 otherwise. Thus, given the word representations $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n] \in \mathbb{R}^{n \times 768}$ and a span (i, j) that starts at position i and ends at j, the span representation will be

$$\boldsymbol{s}_{i,j} = [\boldsymbol{h}_i, \boldsymbol{h}_j, v_{i,j}] \ , \quad (2)$$

For the representation of a null span (Section 3.5), we use $\boldsymbol{h}_0$, which is a representation of a CLS token. For all other span representations, we use $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n$. Hereafter, span representations $\boldsymbol{S}$ and a set of spans S' are defined as

$$\boldsymbol{S} = [\boldsymbol{s}_{0,0}, \boldsymbol{s}_{1,1}, \boldsymbol{s}_{1,2}, \ldots, \boldsymbol{s}_{n,n}] \ , \quad (3)$$

$$S' = \{(0,0), (1,1), (1,2), \ldots, (n,n)\} \ , \quad (4)$$

where n denotes the number of tokens.

3.4 Label Representation

In this section, we provide an explanation of the labels used in our experiments and define the label representation utilized in the proposed model.

Each span corresponds to one of the labels shown below, with no overlaps.

1) Semantic Roles (32 types): Spans that correspond to an argument.
2) P : Spans that correspond to a target predicate.
3) $\mathrm{F_A}$: Spans within spans of arguments.
4) $\mathrm{F_P}$: Spans within the span of a target predicate.
5) O : Spans that are not any of the above and do not overlap with them.
6) N : Spans that overlap with the other spans.

The N label implies that the span is not sufficient to be considered as an argument.

We define a set of all labels as L and a set of semantic role labels as R, as follows:

$$L = \{A_0, A_1, \ldots, N\} \quad (5)$$

$$R = \{A_0, A_1, \ldots, A_{TMP}\} \quad (6)$$

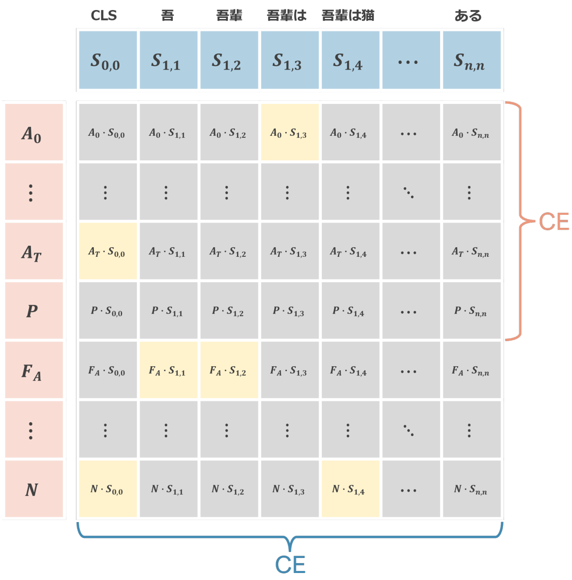

Figure 2: Contrastive learning for our model. This is the process of inner product computation between the label representation (red frame) and the span representation (blue frame). "CE" refers to cross-entropy, and the areas indicated by "CE" are normalized along each axis. After this normalization, the loss value is calculated using cross-entropy loss.

Note that these labels are not necessarily given as target labels for the model to predict; rather they may serve merely as symbols to represent spans. Specifically, in the S4L model, span indices are utilized as targets for the purpose of predicting spans.

Our model is designed to use both span feature and label feature spaces. Accordingly, we prepare embeddings for each label and feed them into an MLP to obtain the label representations, as

$$\boldsymbol{M} = [\boldsymbol{m}_{A_0}, \boldsymbol{m}_{A_1}, \ldots, \boldsymbol{m}_N] \ , \quad (7)$$

where $\boldsymbol{M} \in \mathbb{R}^{|L| \times 768}$.

3.5 Training

In this section, we describe the training methods for each model. The learning process involves updating the weights and bias values in order to minimize the loss function described in the following section. The training details are in Appendix A.2.

3.5.1 Our Model

The scoring function of our model is defined as the cosine similarity of a label l ($l \in L$) and a span (i, j), as

$$\text{Score}(i, j, l) = \frac{\boldsymbol{m}_l \cdot \boldsymbol{s}_{i,j}}{\|\boldsymbol{m}_l\|_2 \|\boldsymbol{s}_{i,j}\|_2} \ , \quad (8)$$

where $\|\cdot\|_2$ denotes the $l2$ norm.

To facilitate smooth learning by scaling the score values, we introduce a temperature parameter α and define the logits as follows:

$$y_{i,j,l} = \text{Score}(i, j, l) * \alpha \tag{9}$$

The proposed model utilizes contrastive learning, which necessitates training based on distributions normalized across each axis. Our model is designed to predict a null span $\boldsymbol{s}_{0,0}$ in the absence of semantic roles. Consequently, the span corresponding to each semantic role label will be uniquely determined across all spans. Thus, for all possible spans S' concerning the label indicated by the red "CE" in Figure 2, normalization is performed, and the loss is calculated using the cross-entropy loss, as

$$\mathrm{P}(i, j|r) = \frac{\exp(y_{i,j,r})}{\sum_{(i',j')\in S'} \exp(y_{i',j',r})} , \tag{10}$$

$$\mathcal{L}_{label} = -\sum_{r\in R'} \sum_{(i,j)\in S'} t_{i,j} \log \mathrm{P}(i, j|r) , \tag{11}$$

where $\boldsymbol{t}$ denotes the one-hot encoded true label vector over label r and R' denotes $R \cup \{P\}$.

Similarly, by assigning the N label as the correct label for the null span, the label corresponding to each span will be uniquely determined across all labels L. Thus, for the span indicated by the blue "CE" in Figure 2, normalization is performed over the labels L, and the loss is calculated using the cross-entropy loss, as

$$\mathrm{P}(l|i, j) = \frac{\exp(y_{i,j,l})}{\sum_{l'\in L} \exp(y_{i,j,l'})} , \tag{12}$$

$$\mathcal{L}_{span} = -\sum_{(i,j)\in S'} \sum_{l\in L} t_l \log \mathrm{P}(l|i, j) , \tag{13}$$

where $\boldsymbol{t}$ denotes the one-hot encoded vector over span (i, j).

We use the final loss to train the model, which is the average of $\mathcal{L}_{label}$ and $\mathcal{L}_{span}$, as follows:

$$\mathcal{L} = \frac{1}{2}(\mathcal{L}_{label} + \mathcal{L}_{span}) \tag{14}$$

By using the average of losses, the model can prevent gradients from becoming excessively large, thereby stabilizing the training process.

3.5.2 L4S Model

The scoring function of the L4S model is defined as the probability of label l for span (i, j). It is formulated using the softmax function and the multilayer perceptron (MLP), as

$$\begin{aligned} \text{Score}(i, j, l) &= \mathrm{P}(l|i, j) \\ &= \text{softmax}(\text{MLP}(\boldsymbol{s}_{i,j}))_l , \end{aligned} \tag{15}$$

where $\boldsymbol{s}_{i,j}$ denotes the span representation between i and j.

We calculate the loss using cross-entropy loss, as

$$\ell(i, j) = -\sum_{l\in L} t_l \log \mathrm{P}(l|i, j) ,$$

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=i}^{n} \ell(i, j) , \tag{16}$$

where $\boldsymbol{t}$ denotes a one-hot encoded vector over span (i, j) and $\ell(i, j)$ is the loss at span (i, j).

3.5.3 S4L Model

The scoring function of the S4L model is defined as the probability of span (i, j) for span label r. It is formulated using the softmax function and the MLP, as

$$\begin{aligned} \text{Score}(i, j, r) &= \mathrm{P}(i, j|r) \\ &= \frac{\exp(\text{MLP}_r(\boldsymbol{s}_{i,j}))}{\sum_{\boldsymbol{s}_{i',j'}\in S} \exp(\text{MLP}_r(\boldsymbol{s}_{i',j'}))} , \end{aligned} \tag{17}$$

where $\text{MLP}_r(s_{i,j})$ denotes the output value for label r ($r \in R$) after passing the span representation $\boldsymbol{s}_{i,j}$ through the MLP.

This model learns to predict spans for semantic role labels. Thus, if there is no span corresponding to semantic roles, following the approach of (Ouchi et al., 2018), the model predict null span, which is the span of the predicate. Similar to the proposed model, this comes from the fact that there is no semantic role label assigned to the span.

We calculate the loss using cross-entropy loss, as

$$\ell(r) = -\sum_{i=1}^{n} \sum_{j=i}^{n} t_{i,j} \log \mathrm{P}(i, j|r) ,$$

$$\mathcal{L} = \sum_{r\in R} \ell(r) , \tag{18}$$

where $\boldsymbol{t}$ denotes a one-hot encoded vector over label r and $\ell(r)$ is the loss at label r. Note that, by definition, span $(0, 0)$ is not included.

3.6 Decoding

Decoding refers to the task in which the model selects the most appropriate combinations of label and span in a sentence. The selection is conducted based on the scores calculated by a scoring function.

3.6.1 Scoring Function

To calculate the span scores, we use the scoring function defined in Section 3.5. Thus, the proposed model uses Equation 8 to compute the score values, while the L4S and the S4L models use Equations 15 and 17, respectively.

3.6.2 Inference

A simple argmax inference over the scores (Equations 8, 15, and 17) selects one label for each span or one span for each label. While this inference is computationally efficient, it faces the following two problematic issues.

1. The argmax inference sometimes selects spans that overlap with each other.
2. The argmax inference cannot select multiple spans for one label.

To deal with these challenges, we employ the approach of (Ouchi et al., 2018), which uses a greedy search to keep the consistency among spans and can return multiple spans. Specifically, we greedily select higher-scoring labeled spans subject to some constraints, which vary by model.

In (Ouchi et al., 2018), it is noted that core label, which are obligatory arguments for the predicate (such as Arg0), are constrained to a single span, and thus the spans of the labels must only be selected once during decoding. However, in Japanese SRL, although the number of such cases is very limited, there is a possibility that multiple spans correspond to a single core semantic role. Therefore, while ideally no constraints should be placed on core labels, in our experiments, we conduct validation with this constraint in place. For testing, we decode under both constrained and unconstrained conditions.

The following are constraints common to all models.

i) Any spans that overlap with the predicate span cannot be selected.

ii) Any spans that overlap with the selected spans cannot be selected.

iii) At most one span can be selected for each core label. (optional)

iv) Spans whose scores are lower than a certain threshold cannot be selected. (optional)

These constraints ensure the consistency of spans and the reliability. The value of the threshold for constraint 3 is set to maximize the F1 score on the development data. Note that constraint 3 is used in training and 4 is used only in testing.

L4S model: Spans are selected based on the following constraints.

i) The label has to be one of the semantic role labels or an O label.

ii) The score has to be higher than that of the N label in the same span.

The first constraint is established not only to extract the target label but also to eliminate spans that are not arguments. Even if the O label is selected, it is not considered in the evaluation. The second constraint is rooted in the fact that the N label indicates that the span is not an argument; hence, the selected spans must be higher than N labels in the scores.

S4L model: Spans are selected based on the following constraint.

i) The score should be higher than that of the null span in the same label.

This constraint indicates that the scores lower than the null span are insufficient to be output as arguments.

Our model: Spans are selected based on the following constraints.

i) The span has to satisfy the same constraints as the L4S and S4L models.

ii) The score must not be negative.

The second constraint comes from the fact that if the score is negative, it indicates that the span is not similar to the label.

4 Experiments

We use L4S, S4L, and proposed models in our experiments. The L4S model is based on the approach of labeled span modeling, which predicts spans for each label, while the S4L model is a pseudo-model inspired by (Ouchi et al., 2018). All models are trained and tested on Japanese semantic role labeling data (NPCMJ-PT). The following sections describe the details of the data, evaluations metrics, results, and discussion.

4.1 Experimental Setup

NPCMJ-PT is a tagged corpus that assigns PropBank-style semantic roles to Japanese sentences based on the conceptual frame of the Predicate-Argument Structure Thesaurus. The details are in Appendix A.1. The training and test data extracted from the NPCMJ-PT is in a format in which each predicate is assigned a related semantic role. Since one sentence contains several predicates, annotation data of a predicate and its semantic role labels are separately recorded for each annotated predicate, even for the same sentence.

NPCMJ-PT consists of 52,528 entries, and the data is divided into training, development, and test sets in an 8:1:1 ratio, with the respective numbers of entries being 42,022, 5,253, and 5,253. With regard to the training data for the S4L model and our model, due to the limitation that the models can only be trained in cases where one semantic role label corresponds to one span ,the training examples containing instances where one semantic role label corresponds to multiple spans are duplicated and separated to ensure that one label corresponds to one span. As a result, the number of training entries for the S4L is 43,310, while the development and test data remain the same as previously described.

To shorten the learning time, we reduce the number of paddings by sorting the data in ascending order by the length of a sentence. We also limit the maximum span width to 30 tokens, while the length of the sentence is still the same. In the experiments, training is terminated when the F1 score in decoding on the development dataset does not improve consecutively for five times and are based on a single run of the training process. In our evaluation, we count the cases where the span and semantic role label match as correct. The evaluation metrics used in this study are precision, recall, and F1.

4.2 Experimental Results and Discussion

As shown in Table 1, our model outperforms all other models in every category, exhibiting the highest F1 scores. This superior performance can be attributed to two main factors: “two types of learning” and “the nature of the the score.”

First, we explain the two types of learning, which refers here to the two types of learning in contrastive learning: “learning to predict spans for labels” and “learning to predict labels for spans.” This enables our model to evaluate spans by considering both the accuracy of labels in spans (L4S model) and the accuracy of spans in labels (S4L model). Particularly, in decoding (Section 3.6), the constraints of both models are used to narrow down the spans.

Model	Precision	Recall	F1
L4S	79.6	77.4	78.5
L4S*	79.4	77.7	78.5
L4S†	82.2	76.2	79.1
S4L	82.5	77.9	80.1
S4L*	75.8	79.3	77.5
S4L†	83.3	77.6	80.3
Ours	83.0	79.4	**81.2**
Ours*	80.5	**80.5**	80.5
Ours†	**83.1**	79.4	**81.2**

Table 1: Experimental results on NPCMJ-PT dataset. “*” indicates versions without a constraint on the number of span selections for core labels. “†” indicates versions without a constraint on the lower bound of the score value.

Next, we explain the nature of the score. The score here is represented by the cosine similarity between span representation and label representation, as shown in Equation 8. This allows for the independent calculation of span scores for each label, enabling fair comparison of multiple spans within the same label. When comparing the basic model and the model with “*”, removing the constraints results in a higher number of incorrect span predictions, thereby reducing the accuracy. This is because, in most of the data, one span is assigned to one semantic role. However, due to the two factors mentioned above, the performance degradation between S4L and S4L* is 2.6, whereas the degradation between Ours and Ours* is only 0.7.

Another advantage of the nature of the score is its ability to appropriately set thresholds for a similarity constraints during decoding. Specifically, in Ours†, although it does not significantly contribute to performance improvement, it raises precision without compromising recall. The score values of the L4S and S4L models indicate probabilities for specific labels or spans but do not show relative values between labels or spans. Thus, even with a low score, there exist cases where the correct prediction is made, forcing the L4S† and S4L† models to set thresholds that sacrifice recall to increase the F1 score.

4.3 Error Analysis of SRL

To analyze the types of errors made by the model in the semantic role labeling task, we reference the analytical method of (He et al., 2017). This method involves manually correcting the model's output step by step for each type of error, recalculating the F1 score after each correction, and measuring the degree of improvement. Since corrections are made incrementally, the graph will show an upward trend, with steeper slopes indicating more frequent errors. Below, we outline the correction methods for each type of error:

1) Fix Labels : Correct the span label if its boundary matches gold.

2) Move Arg : Move a unique core argument to its correct position.

3) Merge Spans : Combine two predicted spans into a gold span if they are separated by at most one word.

4) Split Spans : Split a predicted span into two gold spans that are separated by at most one word.

5) Fix Boundary : Correct the boundary of a span if its label matches an overlapping gold span.

6) Drop Arg : Drop a predicted argument that does not overlap with any gold span.

7) Add Arg : Add a gold argument that does not overlap with any predicted span.

Based on our analysis(Figure 3), three significant areas of improvement are identified: "Fix Labels," "Fix Boundary," and "Add Arg." All models show most significant performance improvements with "Fix Labels," but it is evident that the improvement in the L4S model is more pronounced than that in the S4L model, with our model showing an intermediate level of improvement between the two. Additionally, the improvement of "Fix Boundary" is highest in the S4L model, while the L4S and our model show similar levels of improvement. Similarly, for "Add Arg," the improvement is highest in the S4L model, followed by our model, and then the L4S model.

These results indicate that the L4S model excels in span boundary identification because of its learning labels for spans but struggles with labeling when considering the entire span. This is reflected in the poor precision of the L4S model as discussed in Sec 4.2. On the other hand, the S4L model, which learns spans for labels, demonstrates proficiency in identifying labels and avoiding unnecessary spans, although it is relatively less effective in span boundary identification compared to the other models. Our model, however, shows intermediate or the lowest improvement values across the three areas, suggesting that it successfully integrates the strengths of both models.

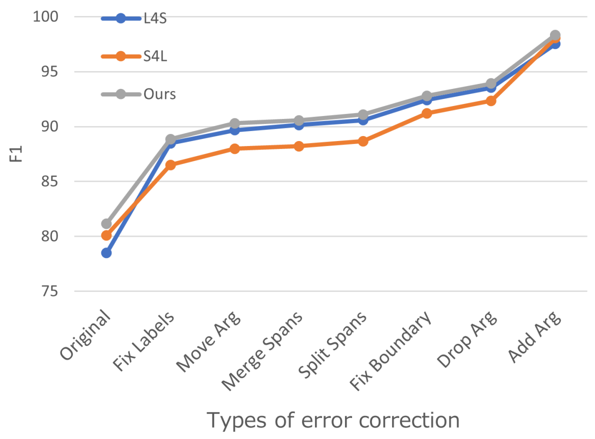

Figure 3: Error analysis of each model. The blue, red, and gray lines correspond to the L4S, S4L, and our models, respectively. The figure depicts how each model progressively improves its F1 score through various types of error corrections.

5 Conclusions

In this work, we proposed a novel model for the SRL task utilizing contrastive learning. Our approach involves learning to align the feature spaces of spans and labels, enabling accurate modeling of their relationship without relying on the probability distribution of one space. Experimental results show that our model outperforms traditional span-based models, achieving a maximum F1 score of 81.2 on NPCMJ-PT dataset.

Limitations

First, it is important to note that our experiments have not been conducted on English datasets, and hence, we cannot guarantee success in other languages. However, for languages like English, where each core argument typically corresponds to a single span, this modeling approach would appear to be appropriate and is likely to facilitate effective learning. Conversely, for languages such as Japanese, where a single argument can be distributed across multiple spans, there remains room for improvement. One potential enhancement could be the utilization of learning techniques capable of multi-label classification, such as binary cross-entropy (BCE), rather than converting the task to a single-class classification through data augmentation. The advantages of this approach include the ability to leverage the correct semantic role structure of the entire sentence and the potential to learn across all spans. However, this would also increase the complexity of the task, raising the possibility of ineffective learning, which necessitates thorough investigation.

Ethical Considerations

For the dataset we use, we have verified that the data does not contain any personal information. According to the data providers, annotation work was requested at 1,200 yen per hour, which is appropriate pay. Annotators were informed in advance about how the data would be used.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Fei, Shengqiong Wu, Yafeng Ren, Fei Li, and Donghong Ji. 2021. Better combine them together! integrating syntactic constituency and dependency representations for semantic role labeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 549–559, Online. Association for Computational Linguistics.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 239–246, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Peter Koomen, Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2005. Generalized inference with multiple semantic role labeling systems. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 181–184, Ann Arbor, Michigan. Association for Computational Linguistics.

Shuhei Kurita, Hiroki Ouchi, Kentaro Inui, and Satoshi Sekine. 2022. Iterative span selection: Self-emergence of resolving orders in semantic role labeling. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5383–5397, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. 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)*, pages 4814–4828, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

Alireza Mohammadshahi and James Henderson. 2023. Syntax-aware graph-to-graph transformer for semantic role labelling. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 174–186, Toronto, Canada. Association for Computational Linguistics.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41.

Koichi Takeuchi, Alastair Butler, Iku Nagasaki, Takuya Okamura, and Prashant Pardeshi. 2020. Constructing web-accessible semantic role labels and frames for Japanese as additions to the NPCMJ parsed corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3153–3161, Marseille, France. European Language Resources Association.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2017. Deep semantic role labeling with self-attention. *arXiv preprint arXiv:1712.01586*, 2017.

Yuanhe Tian, Han Qin, Fei Xia, and Yan Song. 2022. Syntax-driven approach for semantic role labeling. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7129–7139, Marseille, France. European Language Resources Association.

Callum Kodai Tulloch and Koichi Takeuchi. 2024. Semantic role labeling for japanese using span-based models. In *Proceedings of the 2023 7th International Conference on Natural Language Processing and Information Retrieval*, NLPIR '23, page 161–167, New York, NY, USA. Association for Computing Machinery.

Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu, and Min Zhang. 2022. Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4212–4227, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4438–4449, Online. Association for Computational Linguistics.

Category	Count
Sentences	33,510
Predicates	52,528
Conceptual Frame Types	1,012
Arguments	90,140
Semantic Role Types	31
Thematic Role Types	127

Table 2: Dataset statistics.

Semantic Role	Count
Arg1	40,197
Arg0	17,014
Arg2	16,259
ArgM-ADV	4,241
ArgM-TMP	2,347
ArgM-LOC	1,664
ArgM-MNR	1,262
Arg3	1,057
ArgM-PRX	926
ArgM-NEG	792

Table 3: Top 10 Semantic Role Labels by count.

Thematic Role Label	Count
Patient (対象)	33,251
Agent (動作主)	11,702
Experiencer (経験者)	5,247
Adverbial (副詞相当)	4,239
Complement (補語相当（は）)	4,191
Goal (着点)	3,158
Location (場所)	2,588
Time (時間)	2,348
Patient (Action) (対象（動作）)	1,894
Patient (Person) (対象（人）)	1,874

Table 4: Top 10 Thematic Role Labels by count.

A Appendix

A.1 NPCMJ-PT Dataset

NPCMJ-PT is a tagged corpus we use for the dataset in our experiments. First, we will describe NPCMJ (NINJAL Parsed Corpus of Modern Japanese), which is a treebank. NPCMJ provides syntactic and semantic parsing information for written and spoken Japanese texts and is publicly available on the web[5]. NPCMJ-PT is derived from NPCMJ by automatically extracting predicates and their arguments, followed by manual annotation of semantic roles and predicate conceptual frames based on the Predicate-Argument Structure Thesaurus (Takeuchi et al., 2020). The annotators are native Japanese speakers who graduated from a university with a humanities or liberal arts program in Okayama Prefecture, Japan. This is also publicly available on the web[6]. Figure 4 shows a part of the NPCMJ-PT data. The data is converted to a format similar to CoNLL2012 (Pradhan et al., 2012). Each column, tab-separated, represents different information, and each row represents information for one character. Columns 1 through 6 correspond to sentence ID, character index, character, part of speech, syntactic structure, and predicate FrameID, respectively. From the 7th column onwards, semantic roles corresponding to the FrameID in the 6th column are noted. The correspondence between left and right brackets clarifies the range of each piece of information, allowing the embedding of syntactic structures in tree form through nested structures.

Table 2 shows the number of sentences, predicates, conceptual frames, and semantic roles in NPCMJ-PT dataset used in the experiments. Since we predict the semantic roles of their arguments for the target predicates in the sentences, the number of predicates in the table represents the number of instances used in the experiments. There are 1096 types of conceptual frames defined in the Predicate-Argument Structure Thesaurus, and about 92% of them (1012 types) appear in NPCMJ-PT dataset. In the dataset, semantic roles are annotated in two independent formats: PropBank-style roles (such as Arg0 and Arg1) and thematic roles (such as agent and patient). We use PropBank format semantic roles only in the experiments. In the PropBank format, Arg0 through Arg5 are core roles, while labels beginning with ArgM are adjunct roles. According to the annotation guidelines for English PropBank, core arguments are defined within a single span. However, in NPCMJ-PT, it is possible for a single core argument to be distributed across multiple spans.

Table 3 enumerates the ten most frequent PropBank-style semantic role labels. Arg1 is the most frequently occurring role, generally denoting the patient or theme of the predicate. The role of ArgM-ADV, often attributed to adverbial elements, is the most common among adjunct roles. Additionally, Table 4 presents the ten most frequently

[5]NPCMJ: https://npcmj.ninjal.ac.jp/index.html

[6]Predicate-Argument Structure Thesaurus: https://pth.cl.cs.okayama-u.ac.jp/testp/pth/Vths, MIT License.

```
74	0	戸	(N*)	(IP-MAT(PP-SBJ(NP*)	*	*	(Arg1_対象*
74	1	は	(P-OPTR*)	*)	*	*	*)
74	2	直	(ADV*(PP(ADVP*	*	*	(ArgM_MNR_様態*
74	3	ぐ	*)	*)	*	*	*
74	4	に	(P-ROLE*)	*)	*	*	*)
74	5	開	(VB*	*	(FID:271*	*	*
74	6	き	*)	*	*)	*	*
74	7	ま	(AX*	*	*	*	*
74	8	し	*)	*	*	*	*
74	9	た	(AXD*)	*	*	*	*
74	10	。	(PU*)	*)	*	*	*
```

Figure 4: Data form of NPCMJ-PT.

appearing thematic role labels. “Patient” ranks as the most frequent, followed closely by “Agent.” These roles are assigned to elements that represent the patient of the action or the entity (whether a person or an object) executing the action.

A.2 Training Details

In the experiments, we utilize AdamW (Loshchilov and Hutter, 2017) as the optimization method to minimize error during training. Regarding the learning rates, the final four layers of the BERT encoder module are set to 5e-5, while the label encoder and the MLP layers for classification are set to 1e-4. The MLP we utilize is a two-layer neural network. The model is trained on our machine with A6000 GPU cards.