SSDAU: Structured Semantic Data Augmentation for Joint Entity and Relation Extraction

Anonymous ACL submission

Abstract

Joint entity and relation extraction (JERE) are highly susceptible to weak-generalization due to low-quality text on various natural language processing (NLP) tasks. Data augmentation is a common approach to enhance text quality. However, traditional data augmentation methods tend to compromise intrinsic textual logic, making it challenging to design effective augmentation techniques. In this paper, we propose a novel paradigm called Structured Semantic Data Augmentation (SSDAU) that that preserves the structured semantics of augmented text. SSDAU applies context-awareness to encode semantic entities in the text, then designs a Tarjan algorithm to complete the neighborhood semantic analysis of related entity, and finally completes the data augmentation by entity reconstruction of neighboring text with a structured semantic model. We compare different baselines under different JERE scenarios and evaluate the performance and efficiency of SSDAU. The results demonstrate that SSDAU can generate high-quality data, which greatly improves the performance of the JERE model and outperforms the state of the art.

1 Introduction

Joint entity and relation extraction (JERE) have been widely used for representation learning on text due to their superior performance in various applications such as information retrieval (Lin et al., 2020), question answering (Abdelaziz et al., 2021) and text summarization systems (Zhong et al., 2020), etc. JERE have been proven effective by achieving impressive performance for diverse datasets such as search engine, media networks, linguistics, and knowledge graphs. However, the performance of JERE models are influenced by the amount and quality of dataset used in the training process. Researchers usually address this issue through data augmentation. Data augmentation can be used to increase the diversity of data by introducing small perturbations. This technique is widely applied in various fields that require improving the generalization ability of models. For instance, MixUp generates augmented data by mixing labeled data with low-entropy, high-quality unlabeled data (Cheng et al., 2020). Back-translation (Xie et al., 2020) achieves data augmentation by producing repetitive sentences. LLMs generate coherent text content directionally by learning patterns in the representations of text (Zhou et al., 2022).

However, these approaches still pose some challenges. First, data augmentation generally injects slight noise into the text, thus increasing the direction of variation in the data, but this also reduces the correctness of the generated text. Especially in the JERE tasks, this is a severe problem. Since entity relations comprise entities, the introduction of external noise may weaken the entity relevance of the augmented text (Kambhatla et al., 2022), consequently reducing the accuracy of JERE models in the extraction process. Moreover, traditional data augmentation methods may disturb the semantic logic of the text that affect the semantic structure of the text leading to overlapping relationships. And LLMs are susceptible to fixed paradigms leading to the existence of semantic cascades in the generated sentences. Such issues can lead to the emergence of problems such as security risks in the target model (Liu et al., 2020).

To address the above dilemma, we propose a novel data augmentation paradigm known as Structured Semantic Data Augmentation (SSDAU), which achieves data augmentation by preserving the semantic structure of the text. Firstly, we devise a text feature-based encoder that encodes semantic entities in the text through context awareness, thereby reducing noise interference. Secondly, we design a decoder based on the Tarjan algorithm. We match the encoded text, identify similar text segments with adjacent semantics, and then reconstruct the neighboring semantic entity text to complete data augmentation. Thirdly, we devise a structured semantic model to select substitute texts with high similarity and the same semantics for reorganizing the original text. Finally, inspired by dense clustering techniques for topic modeling based on ERT embeddings and c-TF-IDF (Grootendorst, 2022), we utilize the BertTopic model to record important vocabulary retained in topic descriptions and filter out irrelevant topics from the augmented text.

We validate the effectiveness of SSDAU using four datasets that are widely used in the JERE task. We compared six conventional baselines and one LLMs (Zhou et al., 2022) baseline in different JERE models. The experimental results demonstrate that text augmented by SSDAU outperforms the state-of-the-art methods in improving the accuracy of the JERE task. In addition, further experiments demonstrate that SSDAU has great potential in enhancing low-resource JERE tasks.

Our contributions are summarized as follows:

- We propose a novel structured semantic data augmentation paradigm. To the best of our knowledge, this is the first work that investigates data enhancement done through text structured semantics.

- We design a text feature encoding and strong associative semantic decoding strategy that can improve the diversity and quality of text while ensuring its semantic structure.

- We validate the applicability of SSDAU and demonstrate theoretically and experimentally that the augmented text generated by SSDAU has great advantages in terms of validity and performance.

2 Related Work

Information Extraction JERE is a fundamental NLP task that aims to map entity and relation, generate a text-to-triplet model based on their correlation, and assign the triple to a new annotation(Fu et al., 2019). Previous JERE models mostly employ joint modeling (Ren et al., 2017) or sequential annotation (zhe, 2017) to extract entities and relations together. They focus on structured learning by manually constructing features, building information tables or knowledge to enhance the relevance of entity extraction and relation recognition(Miwa and Bansal, 2016). However, manually constructed features make it hard to achieve positive results in different applications. To address this challenge, Zhao et al. (Zhao et al., 2021) propose decomposing the JERE task and completing contextual learning by modifying the classification process. They divided the JERE models into three categories: multi-module multi-step (Zheng et al., 2021; Wei et al., 2020), multi-module one-step (Sui et al., 2020; Wang et al., 2020) and one-module one-step (Shang et al., 2022). The accuracy of these models is limited by the training data, and our structured semantic data augmentation method can help generate a large amount of high quality data, which has a great advantage in the basic and downstream applications of JERE models.

Semantic Match Semantic matching is a subtask of text matching, which is mainly applied to retrieve semantically similar texts from libraries in search scenarios (Wu et al., 2022). Some typical approaches include cosine similarity, term frequency-inverse document frequency (TF-IDF) calculation, and deep structured semantic model (DSSM) (Gao et al., 2021). Recent studies have shown that pretraining semantic classification models can effectively compress massive text and improve the generalization ability of semantic matching models (Brown et al., 2020). For example, the emergence of *Similarities* (Zhang Bingyu, 2022) provides solid foundation for developing practical applications for text semantic matching tasks. In particular, the semantic matching function of *Similarities* has been widely recognized for its superior effect in text relation extraction. Based on the existing text similarity matching techniques, we designed a Tarjan-based strong linkage semantic logic text matching algorithm to improve the existing JERE work through text semantic matching.

Data Augmentation Data augmentation is a cost-effective and efficient method that can improve the performance and accuracy of machine learning models, especially in a data-constrained environment (Cashman et al., 2020). Common data augmentation techniques used in NLP include proximal word replacement (Wei and Zou, 2019), word vector replacement (Wang and Yang, 2015), masked language model replacement (Jiao et al., 2020), back translation (Zhang et al., 2020), adding noise (Min et al., 2020; Yan et al., 2019; Hou et al., 2018), etc. Among them, Jonas et al. (Mueller and Thyagarajan, 2016) propose a lexical substitution method for augmented data that preserves

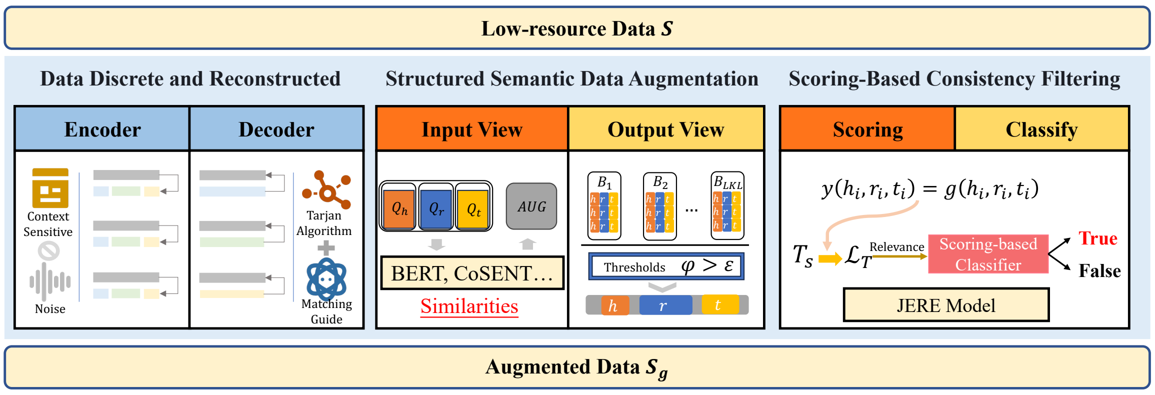


Figure 1: Overview of SSDAU. *Data Discrete and Reconstructed* split the text by semantically segmenting and injecting tags into the segmented text. The color blocks in the *Encoder* represent the text in different semantic regions. We have separated sets by *Decoder* to support *Structured Semantic Data Augmentation* where the *Input View* conditions on the similarity match and *Output View* conditions on data augmentation. Finally, we filter the low-resource data using *Scoring-Based Consistency Filtering* and get the final augmented data $\pounds$, combined with T, to train a more robust JERE model.

original semantics by word proxemics. In addition, text generation methods based on designing prompts using pre-trained language models have garnered significant attention. However, these approaches are limited by the size of synonym lists, vocabulary coverage, and adaptability constraints. Unlike existing methods employing simple perturbation (Liu et al., 2020) or extra augmentor model (Hou et al., 2021; Hu et al., 2019), we propose the sampling-based augmentation, generating data with the same semantic structure by maintaining the semantic logic of the samples.

3 Method

In this section, we provide an overview of our proposed data augmentation method. Figure 1 depicts the framework of SSDAU. We first define the tasks and then describe the three essential components of SSDUA, including i) data discrete and reconstructed, ii) structured semantic data augmentation, and iii) sorcing-based consistency filtering.

3.1 Preliminaries

In this work, triples serve as the foundation for text augmentation, achieved by discretizing the text within its context and utilizing contextual features as semantic labels. This methodology encompasses the formulation of rules for text discretization and subsequent reconstruction, anchoring the semantic structure via semantic labels, thereby establishing a repository for semantic text. Subsequently, semantic rules are integrated with semantic similarity modeling, culminating in the development of a Tarjan algorithm. This algorithm is tasked with matching text semantics and generating data while ensuring the integrity of the text's semantic structure. Finally, the dataset is refined through augmented text filtration employing scoring and filtering modeling techniques. Implementation of SSDAU facilitates the creation of augmented data endowed with comprehensive semantic structures, effectively addressing the quality and quantity prerequisites for datasets in diverse domains, particularly for large-scale language modeling endeavors.

3.2 Data Discrete and Reconstructed

Given set of sentences $S = \{s_1, s_2, ..., s_N\}$ containing L token and K predefined relations $R = \{r_1, r_2, ..., r_K\}$, we extract entities and relations to construct triples $T = \{(h_i, r_i, t_i)\}_{i=1}^{M}$ in S, where h_i, t_i are the head and tail entities, respectively, N represents the number of sentences, M represents the number of triples. In this process, we maintain a three-dimensional matrix M^{L*K*L} to store the existing knowledge. Since the triplet is used as the basic unit of data augmentation, we partition the text according to the triplet labels ρ to obtain three series of text collections.

Encoder We use the triplet as the basic unit of data augmentation to eliminate the noise from textual perturbations. We design a text feature-based encoder E. The input of the encoder is the sentence text S, and for each sentence s_i, we find the specified text block $(q_{h_i}, q_{r_i}, q_{t_i})$ based on the triplet tags $(\rho_{h_i}, \rho_{r_i}, \rho_{t_i})$, and record the context token $(l_{h_i}, l_{r_i}, l_{t_i})$ and the cut position $(p_{h_i}, p_{r_i}, p_{t_i})$. The

Algorithm 1 The execution process of data augmentation through SSDAU.

```
Input: Sentences: S; Token: L, Predefined relations: R; Similarity threshold: $\varepsilon$.
Output: Augmented dataset: $S_g$.
foreach i=1 to N do
    Feed $s_i$, $L$ and $R$ to encoder $E$.
    Split the $R$ to get the triplet labels.
    Obtain the split text block, context tokens, and segmentation point.
    Perform entity type classification to obtain text collections $Q_h$, $Q_t$, $Q_r$.
Feed $Q_h$, $Q_t$, $Q_r$ to decoder $D$.
Classifying relation types and dividing triplet labels to obtain text library $B = B_1, B_2, ..., B_{LKL}$.
foreach i=1 to LKL do
    Get the number of text blocks $m$ in $B_i$.
    foreach j=1 to m do
        Feed text block and context token to text matcher $SIM$.
        Record the similarity value $\Theta$
        if $\Theta \geq \varepsilon$ then
            Add $\Theta$ to the priority queue $P_i$.
foreach i=1 to LKL do
    Calculate the topic score value for each text block. (Equation 1)
    Calculate the topic vector of each text block's corresponding text. (Equation 2)
    if The target replacement text block is not a thematic text block. then
        Meets data augmentation criteria. (Equation 3)
        Perform text replacement to obtain text $S'$
        Add $S'$ to $S_g$.
```

encoder processes all the input text and gets three output text collections according to the tag types: head entity collection Q_h, tail entity collection Q_t and relation entity collection Q_r.

Decoder We design a j Tarjan-based semantic structure similarity matching algorithm based on the strong association property of neighbor semantics. The algorithm will analyze the correlation of K relation types and M ternary labels in the set of sentences S, and match the highly correlated semantic labels by constructing a strong connectivity graph, which will be used as a basis for designing a formal similarity based text matching decoder D. The input of decoder D is (Q_h, Q_t, Q_s), and it divides the text collections according to the relation types and label types to get LKL groups text library $B = \{B_1, B_2, ..., B_{LKL}\}$ with the same relation type and the same label.

3.3 Structured Semantic Data Augmentation

We designed a text matcher based on the semantic similarity evaluation tool *Similarities* to perform similarity matching. Algorithm 1 showcases the execution steps of the structured semantic data augmentation module. The text blocks b in the text corpus $B_i = b_1, b_2, ..., b_j$ record the text q, context tokens l, label type ρ, and segmentation position p. We perform a complete match of all b in different text corpora B_i, including semantic, syntactic, and lexical matching of the text, as well as similarity evaluation of context tokens. In this process, we normalize the matching results to a value between 0 and 1, and add them to a priority queue sorted by decreasing similarity. Finally, for each text corpus B_i, we obtain a similarity priority queue P_i.

After completing the similarity matching, we filter out the text in the priority queue $P_i = P_1, P_2, ..., P_{LKL}$ with similarity less than ε. For the remaining, we replace the text information of the corresponding text block based on the recorded segmentation position l in the information of each text block, thus obtaining the augmented text.

3.4 Sorcing-Based Consistency Filtering

We use the BertTopic model to record important words retained in topic descriptions and filter the augmented text of irrelevant topics, thus ensuring the topic coherence of the augmented text. To this end, we design a score-based BertTopic text filter.

First, we extract entities and relations from the text. Then, we encode the tokens by BERT (Kenton and Toutanova, 2019), obtaining the corresponding entity tokens $l_1, l_2, ..., l_L$. Afterwards, we combine entities and relations in the form of (l_h, r, l_t) and perform triplet extraction using JERE (Shang et al., 2022). Finally, a function is applied to calculate the correlation between the head and tail entity. The scoring function is defined as:

$$h \star t = \phi(W[l_h; l_t]^T + b) \tag{1}$$

Where h and t represent the head and tail, respectively. $\star : \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ denotes circular correlation. $W \in \mathbb{R}^{d_e \times 2d}$ and b are trainable weights and biases, respectively, where d_e denotes the dimension of the entity. $[;]$ is the concatenation operation and $\phi(\cdot)$ represents the ReLU activation function. We then incorporate the highly evaluated entity pairs with the relations and use the relational representation function $R \in \mathbb{R}^{d_e \times 4K}$. The vector function is defined as follows:

$$\upsilon_{(l_h,r_k,l_t)}{}_{k=1}^{K} = R^T \phi(drop(W[l_h; l_t]^T + b)) \tag{2}$$

Where υ represents the score vector and $drop(\cdot)$ refers to the dropout strategy (Srivastava et al., 2014). Next, we add the scoring vector υ to the softmax function to predict the corresponding labels. The formulated triples is as follows:

$$\zeta_{triple} = -\frac{1}{LKL} \times \sum_{i=1}^{L} \sum_{k=1}^{K} \sum_{j=1}^{L} logP(y_{(l_i,r_k,l_j)} = g_{(l_i,r_k,l_j)} | S) \tag{3}$$

Where $g_{(l_i,r_k,l_j)}$ represents the gold tag obtained from annotations. We match all triplets with the golden label triplets to obtain the topic score for each triplet. Finally, we select the high-scoring triplets as the topic tag for the text. We filter out the augmented text where the topic tags are replaced, and obtain augmented data that is both topic-related and has a complete structured semantics.

4 Experiment

4.1 Experimental Setup

Baseline We consider seven of the most commonly used data augmentation methods as our baseline for comparison. Including Word Substitution (WS) (Wei and Zou, 2019), Back Translation (BT) (Xie et al., 2020), Noise Introduction (NI) (Fanghua Ye, 2022), Same-tag Semantic Noise (SSN) (Yan et al., 2019), Generative Models (GM) (Hou et al., 2021), Mixup (Hu et al., 2019), and FlipDA (Zhou et al., 2022). For details on the selection and implementation of the baselines, please refer to Appendix A.1.

Dataset Since the JERE task fits better with short texts, we conduct comprehensive experiments on two authoritative datasets, NYT and WebNLG. Among other things, both types of datasets contain two versions: fully annotated type (NYT, WebNLG) and partially annotated type (NYT, WebNLG). Detailed statistics for these four datasets are provided in Appendix A.2.

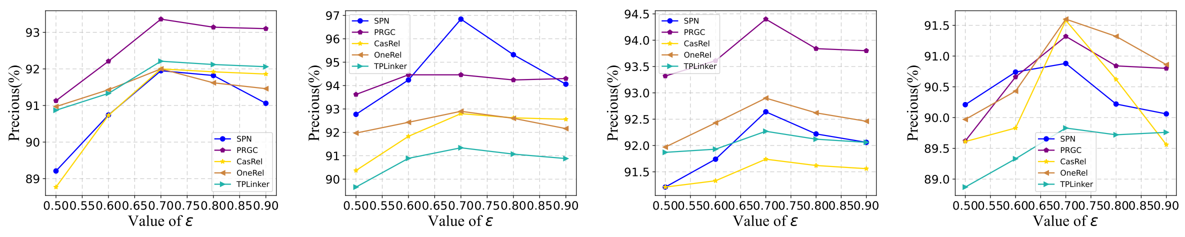
Evaluation and Selection of Thresholds Table 1 describes the number of augmented samples generated by SSDAU for different sets of semantic domains under various similarity thresholds. We counted the effective augmented texts for entities and relations of the four datasets under different variable settings. The results indicate that the number of augmented samples decreases as the threshold value increases. Figure 2 shows the precision of the four augmented datasets under different JERE models with various similarity thresholds. For different datasets, we selected the threshold with the best results as the threshold parameter in the baseline comparison experiments.

Dataset	ε	Head	Relation	Tail	Sum.
	$0.5 \sim 0.6$	15062	243	11300	26605
	$0.6 \sim 0.7$	9439	38	4631	14108
NYT^*	$0.7 \sim 0.8$	1825	19	1365	3209
	$0.8 \sim 0.9$	2927	0	1137	4064
	$0.9 \sim 1.0$	960	0	1546	2506
	$0.5 \sim 0.6$	7082	2742	8116	17940
	$0.6 \sim 0.7$	3933	1946	5342	11221
$WebNLG^*$	$0.7 \sim 0.8$	2049	2162	1557	5768
	$0.8 \sim 0.9$	814	2005	1021	3840
	$0.9 \sim 1.0$	5463	890	2929	9282
	$0.5 \sim 0.6$	13507	234	10076	23817
	$0.6 \sim 0.7$	7721	36	4063	11820
NYT	$0.7 \sim 0.8$	4922	13	1588	6523
	$0.8 \sim 0.9$	2198	0	1140	3338
	$0.9 \sim 1.0$	3700	0	1051	4751
	$0.5 \sim 0.6$	4023	3186	6028	13237
	$0.6 \sim 0.7$	2673	2009	4445	9127
$WebNLG$	$0.7 \sim 0.8$	968	1345	1123	3436
	$0.8 \sim 0.9$	309	919	923	2151
	$0.9 \sim 1.0$	3019	444	6935	10398

Table 1: The number of augmented samples produced by SSDAU at various thresholds and semantic domains varies for each dataset, with the same five similarity thresholds applied to all.

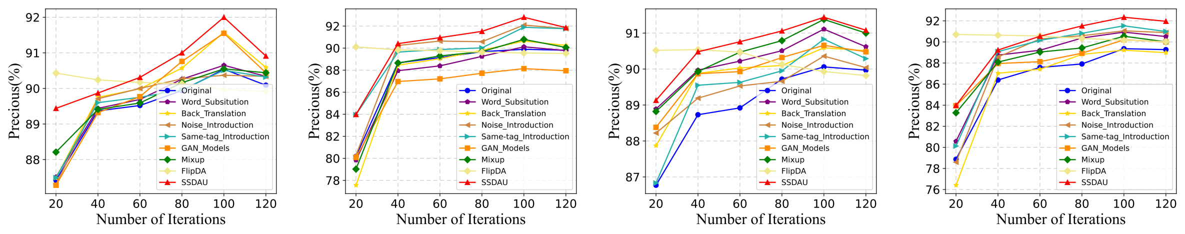
4.2 Main Result

Results of the baseline comparison Table 2 shows the results of SSDAU with seven baseline methods. From the experimental results, it is evident that the augmented text generated by SSDAU is able to train a JERE model with better robustness compared to the baseline methods, which implies that SSDAU has a better performance capability



(a) NYT* (b) WebNLG* (c) NYT (d) WebNLG

Figure 2: Extraction precision of the JERE models with different similarity thresholds. (a), (b), (c), and (d) describe the precision of different JERE models under different datasets, respectively.



(a) NYT* (b) WebNLG* (c) NYT (d) WebNLG

Figure 3: The effects of various baselines on different datasets are examined. Specifically, (a), (b), (c), and (d) illustrate the precision of the data augmentation baseline at different iterations, with the CasRel model serving as the prediction model.

in the JERE task. This indicates that the strategy of data augmentation by holding the structured semantics of the text has better results in ensuring the accuracy of the JERE task.

In addition, Table 2 shows the validity (Prec), performance (F1), and adaptability (IoU) results of SSDAU and the seven baseline methods in different JERE tasks. The findings consistently show that SSDAU outperforms the other baselines in effectively augmenting the data for various JERE tasks. In addition, it is observed that the data augmentation methods for LLMs is able to show better results in low iteration scenarios compared to the other methods, but the quality of the generated data will be lower than the other methods as the number of iterations is augmented. While in JERE task, due to the specificity of the scenarios, data resources are extremely scarce, so it is necessary to iteratively augment the data continuously to ensure the robustness of the trained model. Therefore, SSDAU is more suitable for the current application scenario.

Results in different JERE tasks To validate the generality of the data generated by SSDAU, we conduct comparative experiments across different JERE tasks. We choose BT as the comparative baseline, which is a representative augmentation method that best preserves the semantic structure of augmented text. Table 3 presents the effectiveness of SSDAU and the BT method across various JERE models. The results show that the dataset augmented by SSDAU improves in different types of JERE models. For instance, in the SPN model, the precision of the $WebNLG_g^*$ dataset increased by 3.03%, while in the TPLinker model, the precision of the NYT_g dataset improved by 0.94%. These outcomes demonstrate that the structured semantic data generated by our approach effectively enhances the robustness of JERE models in performing tasks.

4.3 Ablation Study of SSDAU

We conduct an ablation study on the NYT^* and $WebNLG^*$ benchmarks to evaluate three components: Data Discrete and Reconstructed, Structured Semantic Data Augmentation, and Scoring-Based Consistency Filtering. In the process, we maintain the other component settings consistent.

Data Discrete and Reconstruction First, we remove the pre-processing component, Data Discrete and Reconstruction, and instead directly split the data based on the triad message without semantic tags (No Label Split). Additionally, we employ conventional text split methods, the no split and complete full split schemes (Gao et al., 2020). As Table 4 shows, we evaluate the effectiveness of the pre-processing components before and after

Category	Partial Match						Exact Match					
	NYT*			WebNLG*			NYT			WebNLG		
	Prec.	F1	IoU	Prec.	F1	IoU	Prec.	F1	IoU	Prec.	F1	IoU
Original	90.17	91.45	84.24	90.62	90.25	82.23	92.83	92.17	85.47	90.66	89.08	80.31
WS (Wei and Zou, 2019)	88.82	88.98	80.16	91.47	91.51	84.35	89.91	89.61	81.17	89.66	88.88	79.98
BT (Prabhumoye et al., 2018)	88.97	89.52	81.02	91.77	91.97	85.14	89.10	89.54	81.07	89.46	89.90	81.70
NI (Fanghua Ye, 2022)	89.37	89.91	81.67	92.41	92.16	85.46	88.38	89.70	81.32	88.41	87.64	78.00
SSN (Yan et al., 2019)	89.03	89.55	81.08	91.89	92.44	85.94	88.25	89.77	81.44	84.77	85.93	75.34
GM (Hou et al., 2021)	88.30	89.38	80.79	91.84	92.41	85.89	88.60	89.35	80.75	90.82	89.15	80.42
Mixup (Hu et al., 2019)	90.56	90.06	81.92	91.29	92.22	85.56	91.36	90.16	82.08	90.35	88.50	79.37
FlipDA (Zhou et al., 2022)	90.43	89.92	81.43	90.11	90.03	82.77	90.52	88.32	80.20	90.71	89.85	81.47
SSDAU	**92.00**	**92.05**	**85.27**	**92.80**	**92.95**	**86.83**	**91.74**	**92.90**	**86.74**	**91.58**	**89.94**	**81.77**

Table 2: Precision $(\%)$, F1 score $(\%)$ and Intersection over Union $(\%)$ of our proposed SSDAU and baselines in CasRel model. All results are for multiple pattern models and the same training set size.

Model	NYT*	WebNLG*	NYT	WebNLG
SPN(*Suiet al.*, 2020)	91.44/**91.95**	93.81/**96.84**	92.67/92.64	90.21/**90.88**
PRGC(*Zhenget al.*, 2021)	93.33/**93.36**	94.00/**94.46**	93.54/**94.40**	89.92/**91.32**
CasRel(*Weiet al.*, 2020)	88.97/**91.47**	91.77/**92.13**	89.10/**91.74**	89.46/**91.58**
OneRel(*Shanget al.*, 2022)	90.17/**92.00**	90.62/**92.80**	92.83/**92.90**	90.66/**91.60**
TPLinker(*Wanget al.*, 2020)	90.23/**92.21**	90.89/**91.34**	91.33/**92.27**	89.12/**89.93**

Table 3: The precision of different models under different datasets. PRE: A/B denotes BT/SSDAU, indicating the prediction precision of the model obtained by the original training set and the augmented one by SSDAU. Among the seven baselines, BT has the most superior performance.

DataSet	NYT*	WebNLG*	Ave.
CasRel Baseline	90.17	90.62	90.39
SSDAU	92.00	92.80	92.40
Ablation for Pre-processing			
No Split	89.32	90.17	89.75
No Label Split	90.33	90.42	90.38
Full Split	88.64	89.76	89.20
Ablation for Augmentation			
(h,t)	64.21	73.83	69.02
(r)	77.42	84.31	80.87
(h,r,t)	90.41	91.13	90.77
(h,r,h)	85.66	88.53	87.10
(t,r,t)	82.12	84.44	83.28
Ablation for Filtering			
No Filtering	89.92	90.84	90.38

Table 4: Ablation study for SSDAU. No Split denotes not splitting the text. No Label Split denotes splitting by semantics without semantic tag. Full Split denotes complete splitting of the words in the text.

removal by precision. We observe that the outperforms with Discrete Data Reconstruction get an improvement of approximately 2.02%-3.20%. Furthermore, we find that the inclusion of semantic tagging prompts has a positive impact on discrete text data augmentation in low-resource JERE tasks.

Structured Semantic Data Augmentation We apply an exact matching method to verify the effectiveness of the augmentation components. In this process, the tags of the discrete text are replaced with the augmented data's tags, and we classify the augmented data based on the triple's type and use the classified data to train a model to compare the validity of the augmented data after removing the augmented component.

According to the findings in Table 4, the augmented data consists of five types of ternary labels. Among these five types of labels, only the augmented text belonging to the third triad (h, r, t) had a limited positive effect (0.38%) on the JERE task. On the contrary, the remaining four types negatively affected the accuracy of the JERE task. When the augmentation component was removed, the threshold limit was lifted and low quality data was included in the augmentation process. This led to a significant increase in negative data, which in turn reduced the accuracy of the model. It can be seen that the augmentation component helps to maintain the semantic structure and facilitates the mapping process between augmented text and ternary labels. This component determines the accuracy of the coming text extraction process. The experimental results show that the accuracy of the JERE model decreases significantly when the enhancement component is removed, which emphasizes the key role of the enhancement component

Source	Text = South Africa, and the rest of Africa. Triple = [[Africa, /location/location/contains, South Africa]] Structured Semantic = location contain location
Syntax Matching	Text1 = South Africa is a part of Africa. ν = 0.516 Text2 = North Africa, and the rest of Africa. ν = 0.923 Triple = [[Africa, /location/location/contains, North Africa]] Structured Semantic = location contain location

Table 5: Semantic consistency verification of augmented text. ν is the syntactic coherence.

in semantically structured data augmentation.

Scoring-Based Consistency Filtering We verify the impact of the consistency filtering component in SSDAU. Table 4 shows the precision of the JERE models with and without filtered data. The results show that the filtered data positively impacts the model's precision, which decreases when low-quality augmented data are not removed. This implies that within the augmented text produced by SSDAU, there still exist small amounts of data with accurate semantic structures but lower quality. Therefore, to uphold model accuracy, it is essential to filter the augmented text as well.

4.4 Analysis

Semantic coherence analysis. To ensure semantic consistency of the augmented text, we take two steps. First, we consider similarities between text annotations of the same type and entity to label the text by semantic annotations. And we use the Biber Tagger (A. Bergman, 2022) to match ternary texts with the same tags. The high level of syntactic consistency between $Text1$ and $Text2$ is shown in the appendix table 5. Secondly, we filter out texts with low relevance (below 0.8) and include the remaining data in the training set to ensure semantic consistency of the augmented texts. After that, we distinguish between entities and relations in the triples, perform separate similarity matching on entity texts, and replace the triples containing replacement texts. This approach effectively solves the problem of mutual exclusion of multiple relations due to text relevance and ensures semantic coherence between the augmented text and the original text. In addition, in the Appendix A.5, we perform a linguistic validation of the model results.

Training Cost and Convergence. Appendix A.6 provides details about the original and augmented texts containing varying numbers of triples. By classifying the augmented data according to triplets and incorporating them into the training set, we assess different JERE models using the same test set. The results demonstrate the efficacy of SSDAU for texts with different numbers of triplets. Our method proves valuable for texts with varying triplet counts, indicating that as the number of triplets in the training set decreases, the availability of augmented data increases, leading to improved precision of the model.

5 Limitation

Though the proposed SSDAU outperforms all baseline methods, it still has some limitations. First, we initially employed score filtering to screen out lower-quality augmented data, a process that relies heavily on the selection of filtering criteria with varying reference metrics across JERE application domains, so the method still needs to take into account different domain characteristics when migrating across domains. Second, in scenarios where both quality and quantity are ensured, SSDAU may be less efficient in achieving the same effect since LLM's data augmentation methods typically require a fewer number of iterations. In our future work, it will be instructive to validate our approach on JERE tasks in different domains and further optimize the efficiency of SSDAU at the algorithmic and model level.

6 Conclusion

We propose SSDAU, a data augmentation paradigm that provides instances of augmentation for the low-resource JERE task by labeling semantic segmentation of entity texts and evaluating the similarity of neighboring semantic regions. Compared with existing methods, SSDAU solves the problems of noise introduction, relationship overlap, and semantic cascading in low-resource data augmentation scenarios. We experimentally demonstrate that preserving the structured semantics of text may be a more favorable choice for NLP data augmentation.

References

2017. Zheng, suncong and wang, feng and bao, hongyun and hao, yuexing and zhou, peng and xu, bo. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 1227–1236, Vancouver, Canada. Association for Computational Linguistics.

Mona Diab A. Bergman. 2022. Towards responsible natural language annotation for the varieties of arabic. In *Findings of the Association for Computational Linguistics: ACL 2022*, page 364–371, Dublin, Ireland. Association for Computational Linguistics.

Ibrahim Abdelaziz, Srinivas Ravishankar, Pavan Kapanipathi, Salim Roukos, and Alexander Gray. 2021. A semantic parsing and reasoning-based approach to knowledge base question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 15985–15987.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Dylan Cashman, Shenyu Xu, Subhajit Das, Florian Heimerl, Cong Liu, Shah Rukh Humayoun, Michael Gleicher, Alex Endert, and Remco Chang. 2020. Cava: A visual analytics system for exploratory columnar data augmentation using knowledge graphs. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1731–1741.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. Advaug: Robust adversarial augmentation for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5961–5970.

Emine Yilmaz Fanghua Ye, Yue Feng. 2022. Assist: Towards label noise-robust dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022*, page 2719–2731, Dublin, Ireland. Association for Computational Linguistics.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1409–1418, Florence, Italy. Association for Computational Linguistics.

Pan Gao, Qi Wan, and Linlin Shen. 2020. Split and merge: Component based segmentation network for text detection. In *International Conference on Pattern Recognition and Artificial Intelligence*, pages 14–27. Springer.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Issam Laradji Gaurav Sahu, Pau Rodriguez. 2022. Data augmentation for intent classification with off-the-shelf large language models. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 47–57, Dublin, Ireland. Association for Computational Linguistics.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Yutai Hou, Sanyuan Chen, Wanxiang Che, Cheng Chen, and Ting Liu. 2021. C2c-genda: Cluster-to-cluster generation for data augmentation of slot filling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13027–13035.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245.

Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom Mitchell, and Eric P Xing. 2019. Learning data manipulation for augmentation and weighting. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 15764–15775.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, page 4163–4174, Online. Association for Computational Lingustics.

Nishant Kambhatla, Logan Born, and Anoop Sarkar. 2022. Cipherdaug: Ciphertext based data augmentation for neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 201–218, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 7999–8009, Online. Association for Computational Linguistics.

Sisi Liu, Kyungmi Lee, and Ickjai Lee. 2020. Document-level multi-topic sentiment classification of email data with bilstm and data augmentation. *Knowledge-Based Systems*, 197:105918.

Junghyun Min, R Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of* *the Association for Computational Linguistics*, page 2339–2352, Online. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Seoul, South Korea.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2786–2792.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 866–876, Melbourne, Australia. Association for Computational Linguistics.

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1015–1024.

Yu-Ming Shang, Heyan Huang, and Xianling Mao. 2022. Onerel: Joint entity and relation extraction with one module in one step. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11285–11293.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, Xiangrong Zeng, and Shengping Liu. 2020. Joint entity and relation extraction with set prediction networks. *arXiv preprint arXiv:2011.01675*.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2557–2563.

Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. Tplinker: Single-stage joint extraction of entities and relations through token pair linking. In *Proceedings of the 28th International Conference on Computational Linguistics)*, page 1572–1582, Barcelona, Spain (Online). Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, page 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 1476–1488, Online. Association for Computational Linguistics.

Xinyi Wu, Zhenyao Wu, Yuhang Lu, Lili Ju, and Song Wang. 2022. Style mixing and patchwise prototypical matching for one-shot unsupervised domain adaptive semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2740–2749.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2020. Unsupervised data augmentation for consistency training. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 6256–6268.

Ge Yan, Yu Li, Shu Zhang, and Zhenyu Chen. 2019. Data augmentation for deep learning of judgment documents. In *International Conference on Intelligent Science and Big Data Engineering*, pages 232–242. Springer.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Yi Zhang, Tao Ge, and Xu Sun. 2020. Parallel data augmentation for formality style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 3221–3228, Online. Association for Computational Linguistics.

Nikolay Arefyev Zhang Bingyu. 2022. The document vectors using cosine similarity revisited. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, page 129–133, Dublin, Ireland. Association for Computational Linguistics.

Tianyang Zhao, Zhao Yan, Yunbo Cao, and Zhoujun Li. 2021. A unified multi-task learning framework for joint extraction of entities and relations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14524–14531.

Hengyi Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Xu Ming, and Yefeng Zheng. 2021. Prgc: Potential relation and global correspondence based joint relational triple extraction. 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)*, page 6225–6235, Online. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 6197–6208, Online. Association for Computational Linguistics.

Jing Zhou, Yanan Zheng, Jie Tang, Li Jian, and Zhilin Yang. 2022. Flipda: Effective and robust data augmentation for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8646–8665.

A Appendix

A.1 Details of Seven Data Augmentation Methods

Word Substitution Approaches that rely on near-synonyms center around word list replacement techniques (Mueller and Thyagarajan, 2016), which include word vector-based replacement (Wang and Yang, 2015) and mask-based models (Jiao et al., 2020). By replacing the original words with their synonyms, a new text can be generated while maintaining a similar semantic structure.

Back Translation The process of back-translation involves utilizing an English-French translation model (Xie et al., 2020), wherein the original text is translated and then translated back into the original language. This approach generates a new text that retains the semantic structure of the original. Rather than simply replacing individual words, back-translation achieves data augmentation by effectively repeating entire sentences.

Noise Introduction The process of augmenting text involves adding, removing, or modifying words and sentences by introducing various forms of noise, such as changes to the text's form, order, and semantics (Fanghua Ye, 2022). By adding subtle noise that does not relate to the original text, the diversity of the text is expanded, resulting in a larger number of distinct texts.

Same-tag Semantic Noise Introduction SSN achieves data augmentation by introducing noise associated with text tags at the word level (Yan et al., 2019), as well as at the sentence level (Gaurav Sahu, 2022), while preserving the original text's semantics.

Generative Models The GM method involves constructing a generative model based on the original texts to perform targeted data augmentation (Hou et al., 2021). To train the generative model, the original data is passed through a pre-existing generative adversarial network, and the resulting augmented data is added to the training set for JERE models.

Mixup Mixup achieves cross-label data augmentation through hybrid interpolation at both the word and sentence levels (Zhang et al., 2017; Cheng et al., 2020). This method involves fine-grained word vector interpolation and mixing of data or tags to generate new, smoother data and augment the existing dataset.

FlipDA FlipDA first generates text using splicing, then randomly masks off some input tokens and predicts them, and finally sifts through the new samples with the help of a classifier to pick out the ones with the highest confidence to get augmented data (Zhou et al., 2022).

A.2 Details of Low-Source Datasets

We evaluate our method and all the baseline models on two widely used datasets, NYT and WebNLG. The former is generated initially by a remote supervision method, consisting of 1.18 million sentences with 24 predefined relation types. The latter is built for the Natural Language Generation (NLG) task, using triples from DBPedia and including six categories: astronauts, architecture, monuments, universities, sports teams, and writings. NYT and WebNLG come in two versions: one version annotates only the last word of entities, while the other annotates the entire span of entities. We refer to the first version of the datasets as NYT* and WebNLG*, and the second version as NYT and WebNLG. Table 6 provides detailed information about these four datasets.

A.3 Details of Different JERE Models

SPN SPN is a transformer network with non-autoregressive parallel decoding capability. Unlike traditional methods that output triples in sequence, SPN generates the entire set of triples all at once. In this process, SPN disregards the content of the triples and instead focuses solely on relation and entity types. The method of unique prediction by two-part matching provides accurate training signals for SPN.

PRGC PRGC solves the issues of redundant relation prediction, poor generalization, and low efficiency of span-based extraction in the JERE tasks.

Table 6: Statistics of datasets. N is the number of triples in a sentence.

Category	Dataset				Triples in Train Set					
	Train	Valid	Test	Relations	N=1	N=2	N=3	N=4	N≥5	Triples
NYT	56195	5000	5000	24	36835	12065	3672	2623	1001	88366
*NYT**	56195	4999	5000	24	36868	12058	3663	2618	988	88253
WebNLG	5019	500	703	216	1865	1237	1033	641	243	11313
*WebNLG**	5019	500	703	171	1716	1264	1043	648	348	11776

Its core idea is to use a prediction component to estimate the potential relation between entities and extract relevant constraints between them to construct a subset of relation, thereby addressing the issue of overlapping statements during text extraction. Besides, PRGC organizes the subjects and objects into a low-complexity triplet using an internal global communication component.

CasRel CasRel solves the issue of overlapping triples in the JERE tasks by primarily modeling relations as subject-to-object mapping functions (Wei et al., 2020). It revisits the task of relational triple extraction and enhances the current approach by addressing the problem of overlapping triples with identical entities.

OneRel OneRel solves the issue of string-level errors and relation redundancy in the joint extraction process of the JERE tasks (Shang et al., 2022). It breaks down the JERE process into a classification problem and incorporates tokens into the texts to establish a directed mapping between the decoding and encoding processes. A classifier is used to score the triples and ensure consistent performance of the extracted triples across different scenarios.

TPLinker TPLinker focuses on addressing the issue of exposure bias caused by overlapping relations of shared entities in JERE tasks (Wang et al., 2020). It utilizes a single-stage joint extraction framework that guarantees the extracted triples to be free from the exposure bias. TPLinker introduces a new labeling scheme that aligns the entities under each relation type to identify overlapping relation that share multiple entities.

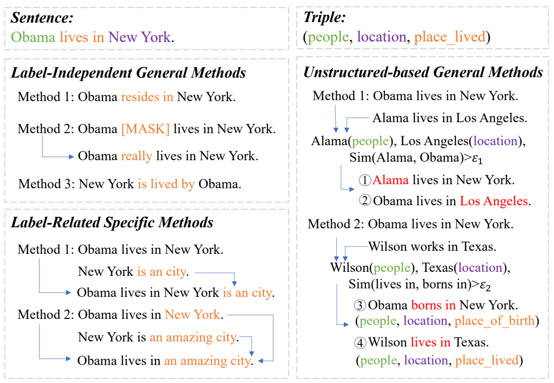
The performance of the four augmented datasets under different JERE models with various similarity thresholds is depicted in Figure 2. The results indicate that the five JERE tasks perform optimally when the threshold value is set to 0.7.

A.4 Implementation Details

In our experiments, we complete all processes on a single server equipped with an Intel Xeon Gold 6248 2.50GHz CPU, two Tesla V100 SXM2 32GB GPUs, and Ubuntu 18.04.6 operating system. We reuse the pre-trained base-cased English model released by Huggingface for BERT.

We store the unstructured texts that need to be extracted along with their corresponding categories. To deal with a large sample size, we divide the dataset into multiple samples of size 1000, and randomly select three samples for augmentation. This process results in a sample set of 1,000,000 texts, with a sample capacity of 1000, to reduce the time cost. Additionally, we match unstructured texts with the same category and store text pairs with a similarity score greater than 0.60. For text augmentation, we filter out entity and relation texts with similarity scores greater than 0.50 and 0.70, respectively. Finally, we obtain the head entity, tail entity, and relation entity structured augmented texts and combine them to generate a structured augmented text.

In our experiments, we utilize three standard evaluation metrics, namely Precision (Prec.), F1-score (F1), and Intersection over Union (IoU). During the evaluation process, we employ an exact matching approach for the dataset, where the predicted triples are considered correct only if the entire span of the two entities and relation are matched.



Sentence:
Obama lives in New York.
Triple:
(people, location, place_lived)
Label-Independent General Methods
Method 1: Obama resides in New York.
Method 2: Obama [MASK] lives in New York.
Obama really lives in New York.
Method 3: New York is lived by Obama.
Label-Related Specific Methods
Method 1: Obama lives in New York.
New York is an city.
Obama lives in New York is an city.
Method 2: Obama lives in New York.
New York is an amazing city.
Obama lives in an amazing city.
Unstructured-based General Methods
Method 1: Obama lives in New York.
Alama lives in Los Angeles.
Alama(people), Los Angeles(location), Sim(Alama, Obama)>ε_1
①Alama lives in New York.
②Obama lives in Los Angeles.
Method 2: Obama lives in New York.
Wilson works in Texas.
Wilson(people), Texas(location), Sim(lives in, borns in)>ε_2
③Obama borns in New York. (people, location, place_of_birth)
④Wilson lives in Texas. (people, location, place_lived)

Figure 4: The process for our augmentation method. Arrows from texts to triplet indicate text extraction; arrows from triple to texts indicate data augmentation. The threshold of similarity between head and tail entities is denoted by ε_1, and the threshold of similarity of relation is denoted by ε_2.

A.5 Validity Verification

Verification of Lexical Fluency As shown in Figure 4, we are comparing our method with the traditional data augmentation paradigm. During our augmentation process, we aim to mitigate any lexical damage to the text by preserving its structure. To maintain the lexical fluency and relevance of the augmented texts, we use similarity constraints while keeping the structure of the text intact, as long as they have the same entity annotation type. For instance, although the morphological similarity between "son" and "sun" is 0.6824, their semantic similarity is only 0.3723. Therefore, we only use texts with the same entity text annotation and similarity above 0.7 for our data augmentation to ensure the text's lexical fluency.

Table 7: Recall (%) of SSDAU versus seven baselines.

Category	NYT*	WebNLG*	NYT	WebNLG
$Original$	92.76	91.45	91.51	87.55
WS	91.17	89.88	89.31	88.11
BT	90.08	91.55	89.98	90.34
NI	90.46	92.17	91.06	86.88
SSN	90.08	91.91	91.34	87.12
GM	90.49	93.00	90.11	87.54
$Mixup$	89.57	93.17	89.99	86.72
$FlipDA$	91.83	91.20	90.31	87.93
$SSDAU$	92.10	93.10	94.09	88.36

Verification of Usability Table 7 shows the recall of various data augmentation methods. It is observed that while training the JERE models, conventional methods cause a slight decrease in the model's recall. However, the data augmented by SSDAU can still maintain a high recall, indicating that our data augmentation method is more effective than the baselines in terms of usability.

Besides, the experimental results show that conventional data augmentation methods are not significantly effective in improving the accuracy of joint entity and relation models and may even have adverse effects. On the other hand, our proposed method, SSDAU, leads to a positive recall improvement. Additionally, we compare the IoU values of models trained on data augmented by SSDAU and conventional methods. It can be observed that SSDAU produces the highest IoU value, indicating that the augmented data is more closely correlated.

Verification of Effectiveness Table 8 illustrates the effectiveness of SSDAU in different types of JERE tasks. It is evident that SSDAU yields positive results in various JERE tasks. This is mainly due to the fact that SSDAU transforms the JERE tasks into a triple classification problem by applying structured semantic labeling of features, which eliminates the induced association of similar texts.

For the NYT dataset, the entire entity information is annotated to match the annotated entity types. During this process, the text is divided into unstructured discrete texts based on semantic tags, and data augmentation is performed by replacing texts with the same annotation type and similar semantics. The conventional methods tend to directly ignore texts with high similarity when the semantics are the same, and the structure is consistent, leading to lower text extraction accuracy. In contrast, SSDAU overcomes this issue by employing consistency filtering to ensure the effectiveness of the augmented text.

A.6 Training Cost and Convergence

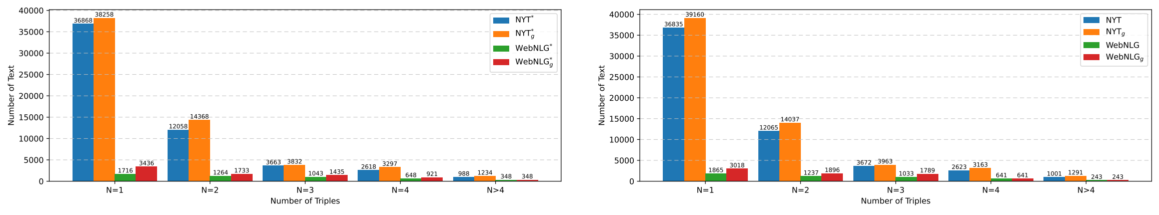
Comprasion of partial and exact datasets Figure 5(a) displays the partially annotated datasets before and after augmentation, including the number of triads in the texts. Additionally, exactly matched datasets are also included in the analysis. To test the effectiveness of our data augmentation method across all datasets, we apply it to the NYT and WebNLG datasets, which match the annotated entity types, resulting in the augmented datasets NYT_g and $WebNLG_g$. To evaluate the effectiveness of SSDAU under different JERE tasks, we validate it using five models, namely SPN, PRGC, CasRel, OneRel, and TPLinker. Figure 5(b) presents the information before and after data augmentation.

During the similarity matching process, as described in Section 3.3, our approach distinguishes between the entities and relation in the triples. It conducts separate similarity matching for the head entities, tail entities, and entity relationships. This approach effectively avoids the issue of multiple groups of relations being mutually exclusive due to text-relatedness. During this process, the entity and relation texts retain only their text type annotations and are reconstructed into semantically consistent structured text using these annotations. Therefore, to complete the text matching, we need to replace all the triples corresponding to the texts of the entity in the entire set of multiple groups.

We use text type annotations as semantics in the matching process for entities and relations. Each entity or relation text only represents its annotated

Table 8: Effect of SSDAU in different JERE models. PRE: A/B denotes Original/SSDAU, indicating the prediction precision of the model obtained by the original training set and the augmented by SSDAU.

Module Type	**Datasets**			
	NYT*	WebNLG*	NYT	WebNLG
Multi-module Multi-Step	91.48/92.42	92.89/93.33	91.32/93.07	89.69/91.45
Multi-module One-Step	90.81/91.98	92.22/94.82	92.75/92.77	90.44/91.24
One-module One-Step	90.17/92.00	90.62/92.80	92.83/92.90	90.66/91.60



(a) The Partially Data

(b) The Exactly Data

Figure 5: The comparison between the number of triads included in SSDAU after augmentation and the initial one for different types of datasets.

type in the text, which helps avoid semantic incompatibility issues caused by the transformation of entities from different types in different triples, while ensuring the completeness of the semantic text.

Topology of Augmented Data Figure 6 displays the topology of data with the location type of the head entity, comprising of five triple relations. For two relation groups, *"/location/country/capital"* and *"/location/country/administrative_divisions"*, their spatial locations are adjacent, indicating that they have stronger associations during data augmentation. We perform data augmentation on two data samples of the same type of triplet relation, and the resulting augmented sample lies between these two samples in terms of spatial structure. These results demonstrate that our data augmentation method 1) ensures consistency of tuple relationships, 2) aligns with real-world general knowledge, and 3) has generalization capabilities in complementary applications of knowledge graphs.

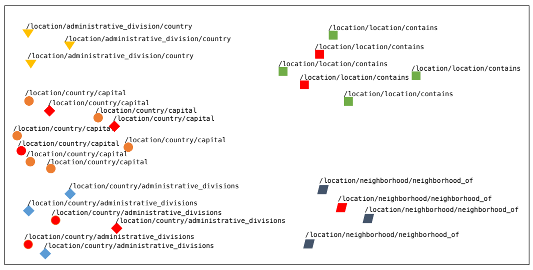


Figure 6: (Best viewed in color and zoom in.) Visual representation of augmented data on spatial structure. The augmented data are labeled in red.

A.7 Case Study

Table 9 shows the seven baseline methods for the sample after the augmentation of the original data.

WS For the original text, the keyword *"reviewed the tire"* is replaced with *"took a closer review to the tire"* by word substitution.

BT Back translation is familiar to convert the text to French, then convert the text to the original language to get *"Goodyear officials rushed back to their headquarters in Akron, Ohio, and carefully examined the tire data from the June 12 race."*.

NI Noise insertion gets augmented text by adding the words *"curried"* and *"race competition"*.

SSN The same-tag noise insertion gets augmented text by adding the word *"anxiously"* with a similar tag to the text *"scurried"*.

GM Generative models are obtained by training four datasets as training sets. And the augmented text *"Officials from Goodyear hurried back to the racing headquarters in Akron, Ohio, to double-check the tire data for the June 12 race."* with high similarity to the original text.

Mixup Mixup obtains the augmented text *"Goodyear officials scurried back to their racing headquarters in Akron, Ohio, city, and carefully reviewed the tissue data from the June 12 race."* by mixed interpolation at the sentence level. As we can see, compared to the data obtained by conventional data augmentation methods, SSDAU is able

Table 9: Augmented cases by conventional data augmentation methods.

Source	Text: Goodyear officials scurried back to their racing headquarters in Akron, Ohio, and carefully reviewed the tire data from the June 12 race. Triples: Abu Ohio(location)|Akron(location)|contains
WS	Text: Goodyear officials scurried back to their racing headquarters in Akron, Ohio, and carefully took a closer review to the tire data from the June 12 race. Triples: Abu Ohio(location)|Akron(location)|contains
BT	Text: Goodyear officials rushed back to their headquarters in Akron, Ohio, and carefully examined the tire data from the June 12 race. Triples: Abu Ohio(location)|Akron(location)|contains
NI	Text: Goodyear officials scurried rushed back to their racing headquarters in Akron, Ohio, and carefully reviewed the tire data from the June 12 race competition. Triples: Abu Ohio(location)|Akron(location)|contains
SSN	Text: Goodyear officials scurried anxiously back to their racing headquarters in Akron, Ohio, and carefully reviewed the tire data from the June 12. Triples: Abu Ohio(location)|Akron(location)|contains
GM	Text: Officials from Goodyear hurried back to the racing headquarters in Akron, Ohio, to double-check the tire data for the June 12 race. Triples: Abu Ohio(location)|Akron(location)|contains
Mixup	Text: Goodyear officials scurried back to their racing headquarters in Akron, Ohio, city, and carefully reviewed the tissue data from the June 12 race. Triples: Abu Ohio(location)|Akron(location)|contains
FlipDA	Text: Officials from Goodyear hurriedly returned to their racing headquarters in Akron, Ohio, where they meticulously examined the tire data from the race held on June 12th. Triples: Abu Ohio(location)|Akron(location)|contains
SSDAU	Text: Goodyear officials rushed back to their racing headquarters in Akron, Ohio, where they meticulously reviewed the tire data from the race on June 12th. Triples: Abu Ohio(location)|Akron(location)|contains

to augment diverse text data with different triadic semantic structures while maintaining the semantic structure of the text.

FlipDA FlipDA generates a text with similar semantics:*"Officials from Goodyear hurriedly returned to their racing headquarters in Akron, Ohio, where they meticulously examined the tire data from the race held on June 12th."*

SSDAU SSDAU will search for semantically labeled text with high similarity based on neighboring semantically strong connectivity intervals, and get new text after filtering: *"Goodyear officials rushed back to their racing headquarters in Akron, Ohio, where they meticulously reviewed the tire data from the race on June 12th"*