CycleOIE: An Unsupervised Low-Resource Training Framework For Open Information Extraction

Anonymous ACL submission

Abstract

In Open Information Extraction (OpenIE), the acquisition of manually annotated sentence-extraction pairs is expensive, while automatically labeled datasets may struggle to accurately reflect real-world requirements for OpenIE systems. Existing neural models often demonstrate impressive performance on large-scale training sets but falter when tested on smaller-scale datasets due to the discrepancy in attributes between the training and test sets. In real-world scenarios, it is crucial for OpenIE systems to align closely with test sets, even when faced with limited annotated data for training.

This paper introduces CycleOIE, a novel training framework applied to a pair of inverse text-to-text models. Through CycleOIE, we train a pair of T5 models on our curated dataset, LSOIE-g, achieving performance levels that surpass baselines trained on significantly larger fully supervised training sets. The ablation study offers a detailed comparison between fully supervised training and CycleOIE, highlighting the effectiveness of CycleOIE on LSOIE-g as the primary factor in enhancing T5's OpenIE performance.

1 Introduction

Open Information Extraction (OpenIE) is a type of Information Extraction task that extracts structured information, such as triples (subject; relation; object), from a given sentence (Yates et al., 2007; Angeli et al., 2015). This task is crucial for various downstream NLP applications, including summarization, knowledge graph construction, and knowledge base question answering.

Neural models for OpenIE demand substantial training data, but manually labeling datasets meeting requirements of real-world applications is costly. Publicly available large-scale OpenIE datasets like OIE2016(Stanovsky and Dagan, 2016), IMoJIE(Kolluru et al., 2020b) and LSOIE(Solawetz and Larson, 2021) are often weakly labeled. Annotations of these datasets are either high confidence extractions filtered out of predictions of existing SOTA OpenIE systems or automatically converted from datasets of other NLP tasks. Though these weakly labeled datasets are employed for training, they are seldom used for evaluation for limited quality. Some OpenIE benchmarks like WiRe57(Lechelle et al., 2019), CaRB(Bhardwaj et al., 2019) and BenchIE(Gashteovski et al., 2022) are presented. The data scale of these benchmarks are so small that they can only be used for evaluation. They have no split for training set for their small scales which are usually no more than 1k sentences. Meanwhile, these small-scale test sets are either labeled by experts or crowdsourcing under a explicitly stated annotation guidelines. Containing richer and inferred relations(Pei et al., 2023), these small-scale test sets play better roles in simulating the scenarios of real world applications and evaluating OpenIE systems.

Researchers design novel neural architectures to fit large-scale weakly labeled training sets, aiming to enhance performance on small-scale test sets labeled by experts or crowdsourcing. However, disparities in attributes between training and test sets, such as the source and number of inferred relations persist(Pei et al., 2023). Consequently, model architectures, regardless of novelty, only impact how well the model fits the training set. Fine-tuning is then required for neural models to align with test data, but obtaining annotated data for fine-tuning is challenging in both academic and real application scenarios.

This paper introduces CycleOIE, an unsupervised training framework tailored for low-resource scenarios. We first leverage gpt-3.5 to generate extractions, constructing a training set for CycleOIE. Specifically, we instruct gpt-3.5 with the prompt words consisting the sentence from LSOIE

wiki[1] validation set and annotation guidelines of BenchIE which adapts to both BenchIE and CaRB dataset. The combination of LSOIE wiki validation sentences and extractions generated by gpt-3.5 is named as LSOIE-g. Subsequently, we train our OpenIE model with CycleOIE. Though LSOIE-g is an parallel dataset, during the cycle training process, its sentences and extractions are shuffled and loaded separately, which means only one side of the sentence-extraction pair is utilized in training one model. This characteristic defines CycleOIE as unsupervised, aligning with the practical consideration that acquiring one side of the data is often more accessible in real-world scenarios. In each epoch of CycleOIE, there are two cycles, which are Sentence-Extraction-Sentence (SES) cycle and Extraction-Sentence-Extraction (ESE) cycle. Each cycle needs two models which are Sentence-to-Extraction(S2E) model and Extraction-to-Sentence(E2S) model. The data is input to model-1[2] in eval mode. Then the prediction of the model-1 is input to model-2 in train mode. So, only model-2's parameter is updated in a cycle by computing the loss between the prediction of model-2 and the input data to model-1 which is from one side of LSOIE-g. In SES cycle, S2E model is model-1 and E2S model is model-2 while in ESE cycle, E2S model is model-1 and S2E model is model-2. In CycleOIE, we run SES cycle and ESE cycle by turns so that E2S model and S2E model can be trained as model-2 by turns. In evaluation, only S2E model is needed to predict extractions.

The central idea behind CycleOIE is that a good extraction should contain all facts appeared in the sentence. Therefore, an extraction can be predicted by an S2E model, and conversely, a sentence can be predicted by an E2S model. No parallel annotation data (i.e., sentence-extraction pair) is needed for training, assuming a well-performing pretrained model capable of predicting the other side of the data. In a cycle, we freeze the parameters of model-1 and train model-2. We use model-1 to predict the other side of data so the input of model-2 is obtained. Then compute the loss of model-2 with the data pair composed up of the output of model-2 and the input of model-1 (i.e., the original input data). We assume model-1 is a well performing pretrained model, capable to predict data of the other side while it is not. So, in the next cycle, we reverse the position of two models and let it get trained as model-2. In CycleOIE, we run SES cycle and ESE cycle by turns so that E2S model and S2E model can be trained as model-2 by turns. After each epoch, the parameters of model-2 is updated and various intermediate products will be output in the next cycle. So CycleOIE can be viewed as a type of data augmentation along with the training process. This approach eliminates the need for a highly performing pretrained S2E or E2S model at the outset and avoids the requirement for large-scale, high-quality parallel data.

sentence	Earlier today , Thailand 's Prime Minister Yingluck Shinawatra formally dissolved the country 's parliament and called for new elections .
extraction	subject <is> the country 's parliament <and> relation <is> dissolved <and> object <is> Earlier today Thailand 's Prime Minister Yingluck Shinawatra <then> subject <is> Thailand 's Prime Minister Yingluck Shinawatra <and> relation <is> called <and> object <is> Earlier today new elections

Table 1: <*is*>, <*and*>, <*then*> is added to format dataset to comply with a text-to-text model.

Our S2E model and E2S model are both fine-tuned from flan-t5-base[3](Chung et al., 2022), a text-to-text transformer that utilizes textual input and output. To adapt T5 for the S2E and E2S tasks, we design a template for extractions with three additional tokens: <*is*>, <*and*>, <*then*> respectively, as illustrated in Table 1.

We conduct experiments on various OpenIE benchmarks, revealing performance that surpasses baselines trained on much larger-scale training set in full supervision. Our contributions are summarized as follows:

(i) We introduce CycleOIE, a training framework that employs cycle training for the OpenIE task, implemented on a text-to-text pretrained language model, T5.

(ii) We design a format for the extraction sequence to align with both the input and output format of a text-to-text model.

(iii) We construct LSOIE-g, a training set for CycleOIE containing approximately 2k sentences, significantly fewer than typical OpenIE training

[1]https://huggingface.co/datasets/wardenga/lsoie/viewer/wiki

[2]In this ariticle, we use **model-1** to refer to the first model and **model-2** to refer to the second model of current cycle.

[3]https://huggingface.co/google/flan-t5-base

sets. LSOIE-g can be used to mirror the highly limited annotation data in real world scenarios.

(iv) We conduct extensive experiments demonstrating that unsupervised CycleOIE is able to outperform some fully supervised OpenIE systems, even with a very limited amount of training data.

2 Related Work

2.1 Open Information Extraction

Open Information Extraction is a fundamental NLP task aiming at extracting structured information from a given sentence. Early OpenIE methods (Yates et al., 2007; Angeli et al., 2015; Del Corro and Gemulla, 2013; Gashteovski et al., 2017) rely on linguistic expertise for sentence parsing, while recent neural methods harness deep neural networks to represent the semantics with hidden states.

Neural methods in OpenIE can be broadly categorized into sequence labeling and sequence generation methods. Sequence Labeling methods (Ro et al., 2020; Kolluru et al., 2020a) usually use an encoder (e.g., BERT) to encode the given sentence into its embedding. A sequence labeling head connected to the encoder labels each token in the sequence. On the other hand, sequence generation methods (Cui et al., 2018; Kolluru et al., 2020b) design prompts or generation templates to transform information extraction task into text generation task. Generally, sequence labeling methods offer faster predictions since they can be performed in parallel while their extractions are limited to tokens from the given sentence, potentially leading to incoherence or grammatical mistakes. In contrast, sequence generation methods are slower as the decoder generates from left to right sequentially, while their generated outputs tend to be more coherent and adaptive.

2.2 Cycle training

Cycle training, also known as cycle-consistency training, is a training framework that use unpaired data to train a pair of inverse models (i.e., output and input of model-1 could become the input and output of model-2 in reverse.). It is initially introduced in the machine translation task as the term iterative back-translation(Hoang et al., 2018) to solve the challenge of lack of sentence pairs composed of source language sentences and target language sentences.

Cycle training is widely used in text generation tasks. Hoang et al. (2018); Wei et al. (2020); Dou et al. (2020) manage to use this method to overcome the scarcity of paired sentence datasets in machine translation area. Iovine et al. (2022b) introduce CycleKQR, leveraging cycle training to enhance Question Answering (QA) performance by rewriting queries into appropriate forms while retaining semantics. Wang et al. (2023) evaluate the effectiveness of cycle training in ensuring consistency between structured data and text, achieving performance comparable to fully supervised approaches for data-to-text generation tasks. Recently, cycle training has been applied to train information extraction models. Iovine et al. (2022a) apply cycle training on T5 to address the lack of in-domain annotation data, achieving competitive performance with fully supervised models on Named Entity Recognition (NER) tasks. With NER and many other information extraction tasks, OpenIE share a similar challenges of insufficient in-domain annotation data. this highlights the potential of cycle training to handle OpenIE tasks. The key lies in designing templates that induce extraction generations consistent with their source sentences.

3 Method

3.1 CycleOIE

We present CycleOIE to address the challenge of lack of paralleled training set and train our model for OpenIE. CycleOIE is implemented by two text-to-text models (with T5 serving as our backbone model), referred to as S2E(Sentence to Entity) model and E2S(Entity to Sentence) model as shown in Figure 1. S2E model (at the top-right corner of Figure 1) is expected to predict the extraction given a sentence as input. Each extraction is made up of triples consisting a **head entity**, a **relation entity** and a **tail entity**. (**subject**, **relation** and **object** syntactically.) E2S model is expected to predict the corresponding sentence given an extraction. CycleOIE is trained through two cycles. In each cycle, the first model is freezed and only the second model's weights will be updated.

Specifically, in ESE cycle (the left part of Figure 1, the e is input to the E2S model in eval mode to predict $\hat{s}$. Then $\hat{s}$ is input to the S2E model (in train mode) to predict $\hat{e}$. Loss between $\hat{e}$ and e is computed to update the parameters of the S2E model through back propagation. After training the S2E model for 1 epoch, the training process comes to SES cycle. In SES cycle (the right part of Figure 1, the s is input to the S2E model (in

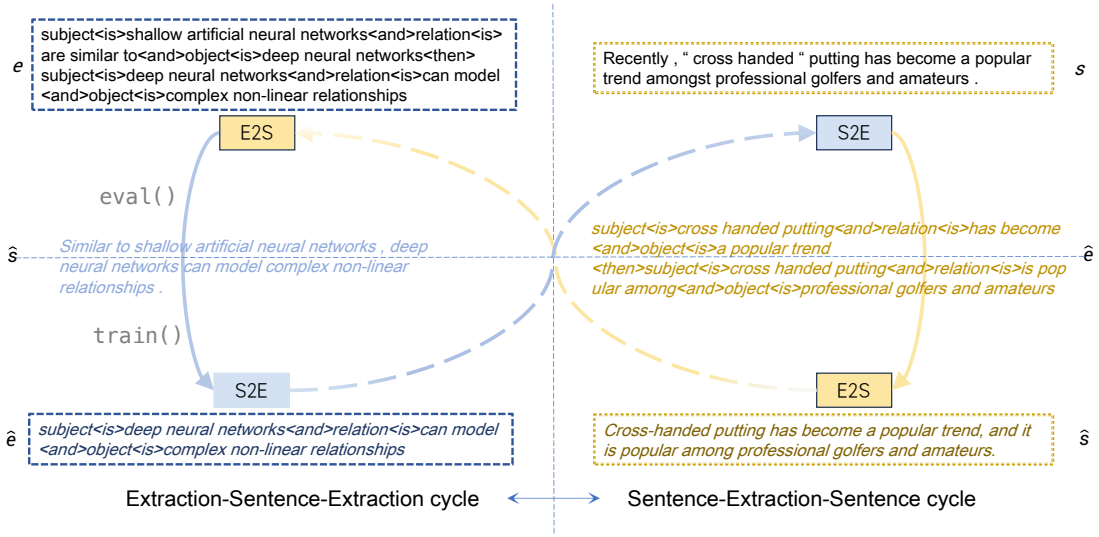

Figure 1: CycleOIE training framework

eval mode) to predict $\hat{e}$. Then $\hat{e}$ is input to the E2S model (in train mode) to predict $\hat{s}$. Loss between $\hat{s}$ and s is computed to update the parameter of the E2S model through back propagation. Once both models have completed one epoch of training, CycleOIE is one epoch updated. Table 2 demonstrates CycleOIE conducted on batched training data.

Algorithm: CycleOIE

Input: Dataset of sentences D_S extractions D_E
Output: S2E model M_{S2E} and E2S model M_{E2S}
  while M_{S2E} and M_{E2S} have not converged do
    for every batch S in D_S:
      Transform S into E' using M_{S2E}
      Train M_{E2S} with (E', S)
    end **for**
    for every batch E in D_E:
      Transform E into S' using M_{E2S}
      Train M_{S2E} with (S', E)
    end **for**
  end **while**

Table 2: Algorithm: CycleOIE

3.2 OpenIE as sequence generation

For taking better advantages of pretrained text-to-text model, we design a generation template that guides the language model to generate textual extractions. The model's tokenizer is extended with three additional tokens: *<is>*, *<and>*, *<then>*. As illustrated in Table 1 , <is> is utilized, in combination with its previous token, to indicate which part of a triple should be generated next; <and> is used to separate head entity, relation entity and tail entity; <then> is used to delimit triples in a extraction.

For training the S2E model, the extraction sequence, denoted as $e = \{e_1, e_2, e_3, ..., e_n\}$, is input to the freezed E2S model, letting E2S model to predict a sentence sequence $\hat{s} = \{\hat{s}_1, \hat{s}_2, \hat{s}_3, ..., \hat{s}_n\}$. The generated sentence sequence $\hat{s}$ is then fed into the S2E model, aiming to predict $\hat{e} = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ..., \hat{e}_n\}$ that aligns with the original extraction sequence e. The loss is computed between $\hat{e}$ and e.

$$\mathcal{L}_\theta(E, \hat{E}) = -\frac{1}{|E|} \sum_{e \in E} \frac{\sum_{i<|e|} p(e_i) log\, p(\hat{e}_i)}{|e|} \quad (1)$$

Here, E represents a batch of e and $|E|$ is the batch size, θ denotes the parameter of S2E model, $p(.)$ signifies probability of token, e_i is the i-th token in e and $\hat{e}_i$ is the i-th token in $\hat{e}$, and $|e|$ represents the length of the sequence of extractions.

For training E2S model, the sentence sequence $s = \{s_1, s_2, s_3, ..., s_n\}$ is input to the freezed S2E model, letting S2E model to predict the extraction sequence $\hat{e} = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ..., \hat{e}_n\}$. The generated extraction sequence $\hat{e}$ is then fed into the E2S model, expecting its prediction $\hat{s} = \{\hat{s}_1, \hat{s}_2, \hat{s}_3, ..., \hat{s}_n\}$ to fit s. The loss is then calculated between $\hat{s}$ and the gold s.

$$\mathcal{L}_\phi(S, \hat{S}) = -\frac{1}{|S|} \sum_{s \in S} \frac{\sum_{i<|s|} p(s_i) log\, p(\hat{s}_i)}{|s|} \quad (2)$$

Here S represents a batch of s and $|S|$ is the batch size, ϕ denotes the parameters of the E2S model, s_i is the i-th token in s, $\hat{s}_i$ is the i-th token in $\hat{s}$, and $|s|$ represents the length of text sequence of the given sentence.

3.3 Build CycleOIE Dataset

As pointed out by Pei et al. (2023), attribute discrepancies generally exists between weakly labeled large-scale training sets and manually labeled small-scale test sets. These discrepancies manifest in factors such as the source of sentences and the proportion of inferred relations and N-ary relations they retain. Table 3 illustrates these differences; LSOIE lacks inferred relations presented in WiRe57, CaRB, while IMoJIE lacks N-ary relations found in WiRe57 and CaRB. Both IMoJIE and LSOIE sentences originate from sources different from CaRB. Achieving high performance on these human-labeled test sets through training on weakly labeled training sets poses a challenge. To address this, we utilize gpt-3.5 to generate extractions adhere to the annotation guidelines of BenchIE[4] given LSOIE wiki validation set sentences as input. Thus we obtain extractions for those sentences. We name the dataset **LSOIE-g** which comprises LSOIE wiki validation set sentences and gpt-3.5 generated extractions. Although LSOIE-g pairs these extractions with their input sentences, during the cycle training, we use sentences in the SES cycle and extractions in the ESE cycle separately. Through experiments, we observe that our model cycle trained on LSOIE-g outperforms some neural baselines fine-tuned in fully supervised settings on much larger training sets.

4 Datasets

As mentioned in the introduction, neural OpenIE systems are typically trained and tested on different datasets. Datasets used in our experiments are listed in Table 3.

LSOIE(Solawetz and Larson, 2021) is a large-scale OpenIE dataset converted from QA-SRL 2.0 with similar conversion method to OIE2016(Stanovsky and Dagan, 2016). **IMoJIE**(Kolluru et al., 2020b) aims to construct a high quality OpenIE dataset with Wikipedia sentences and high confidence extractions predicted by former OpenIE systems including OpenIE4, ClausIE, and RNNOIE. **Wire57**(Lechelle et al., 2019) releases a tiny dataset comprising 57 sentences from 5 documents and extractions annotated by 2 experts. **CaRB**(Bhardwaj et al., 2019) is an OpenIE benchmark with 641 sentences in its test set. Obtained through crowdsourcing, CaRB's annotations are generally considered as an noise reduction from OIE2016(Stanovsky and Dagan, 2016). **BenchIE**(Gashteovski et al., 2022) releases a dataset of 300 sentences. Two expert annotators try to exhaust every possible extraction of facts in a sentence. Annotations of the same fact are classified into one cluster. Unlike WiRE57 and CaRB utilizing token-level scoring functions, BenchIE judges an predicted triple to be true only when it exactly matches a gold triple in the cluster of the fact. we find this scoring function to be too strict, resulting in significant differences in results on BenchIE compared to other benchmarks. To address this, We introduce a new benchmark, CaRB-B, which combines the scoring function of CaRB and the dataset of BenchIE, replacing BenchIE's scoring function and retaining its dataset.

CaRB, due to its quantity advantage among those human-labeled datasets, is selected as our main target for evaluation. Considering the similarity between LSOIE and CaRB[5], we use sentences of LSOIE wiki validation set to compose our training set for CycleOIE to simulate a real scenario where the number of extraction annotations is very limited.

5 Experiment

5.1 Baseline Systems

We compare our methods with 4 previous OpenIE systems: ClausIE(Del Corro and Gemulla, 2013), MINIE(Gashteovski et al., 2017), **Multi^{2}OIE**(Ro et al., 2020), and OpenIE6(Kolluru et al., 2020a). The former two are classical rule-based methods while the latter two are neural-based methods.

ClausIE(Del Corro and Gemulla, 2013) takes the advantage of English grammatical knowledge to extract information from a sentence without any training. By its core step, Dependency Parsing, the syntactic relation between words in a sentence is analysed. Then matching the dependency to the clauses, combining the properties of predicates and the structure of clauses, extractions from clauses are generated. Some worries are presented that the extraction of ClausIE may consist multiple propositions merging into a over specific value(subject,

[4]BenchIE samples 300 sentences and re-annotated by experts under their guideline. Through our research on CaRB and BenchIE, we think the annotation guidelines of BenchIE has many common requirements with CaRB and can instruct gpt-3.5 to generate extraction annotations for CaRB sentences.

[5]OIE2016 is automatically converted from QA-SRL(He et al., 2015) while LSOIE is automatically converted from QA-SRL 2.0(FitzGerald et al., 2018). CaRB is a re-annotation of OIE2016

	Dataset	Source	Sentences	Extractions	Inferred Relations	N-ary relations
training sets	IMoJIE	wikipedia	71209	215K	3K	0
	LSOIE(wiki)	QA-SRL, wikipedia	12832	101K	0	32K
	LSOIE-g	LSOIE, gpt-3.5	2147	6820	383	0
test sets	WiRe57	Wikipedia, Newswire	57	343	173	79
	CaRB	OIE2016	641	2715	736	683
	BenchIE	CaRB	300	783	0	0

Table 3: Statistics of datasets used in our experiments

relation, or object) of a tuple representing an extraction result. These extraction results of less conciseness probably don't meet the demand of its downstream tasks like knowledge graph constructions. Minie(Gashteovski et al., 2017) is an OpenIE system built on the top of ClausIE, aiming to provide compact extractions, i.e., minimizing each value in the tuple representing extraction result.

Multi^{2}OIE(Ro et al., 2020) first leverages a pretrained language model to encode the context of the sentence. Multi^{2}OIE regards OpenIE as a kind of sequence labeling task. Predicates are first labeled by a predicate classifier head concatenated to the BERT model. Then arguments are predicted by an argument classifier given the output of BERT model including the hidden state vector and predicate average vector. Among neural methods, While methods of sequence labeling outputs predictions with faster speed, methods of generation produce extractions of better quality. OpenIE6(Kolluru et al., 2020a), using Iterative Grid Labeling architecture to label a 2-D grid which represents an extraction result with a row, surpassing generative methods(Cui et al., 2018; Kolluru et al., 2020b) achieves SOTA at that time.

5.2 Main results

We compare our CycleOIE with baseline systems. As shown in Table 4, our unsupervised method, CycleOIE, achieves the highest precision and F1 on both CaRB and CaRB-B benchmark, comparing with tradition rule-based methods and two fully supervised neural methods. When evaluating on WiRe57 CycleOIE's F1 only defeats Multi^{2}OIE. Neural methods' performance can be influenced by various factors beyond model architecture, such as the scoring function and attribute discrepancies between the training set and the test set. In Table 5, we conduct additional experiments controlling variables to evaluate the performances of neural methods when they are trained on LSOIE whose sentences share a same source with our LSOIE-g. We compare CycleOIE's performance with neural methods trained on LSOIE wiki training set. The results indicate CycleOIE outperforms these two OpenIE6 and Multi^{2}OIE on WiRe57[7], CaRB[8] and WiRe57-C[9] benchmark even though CycleOIE is training on a dataset with a significantly smaller scale than the LSOIE training set. Notably, when evaluating on CaRB, the performance gap between CycleOIE and other baseline methods becomes more pronounced than the gap illustrated in Table4.

5.3 Ablation study

We conduct an ablation study and results are presented in Table 6. Here are the details of our experimental settings.

Setting **(1) sft on LSOIE** stands for training flan-t5-base on LSOIE wiki training set in fully supervised fine-tuning and **(2) sft on LSOIE+IMoJIE** stands for a similar setting except for training on the union of LSOIE wiki and IMoJIE training set.

In **(3) sft on LSOIE → sft on LSOIE-g** setting, we train the model by 2 steps. At step 1, we do the same training as **(1)**. At step 2, we train the weights output by step 1 on LSOIE-g data with full supervision. In the step1 of **(4) sft on LSOIE+IMoJIE → sft on LSOIE-g**, we do the same training as **(2)**, and the other operations keep the same with the previous one.

In **(5) sft on LSOIE → CycleOIE** setting, we train flan-t5-base in fully supervised fine-tuning on LSOIE wiki training set at step 1, then cycle trained the model output by step 1 on LSOIE-g data. While in setting **(6) sft on LSOIE+IMoJIE → CycleOIE** the training set at step 1 is augmented.

Setting **(10) CycleOIE** stands for we cycle train flan-t5-base on LSOIE-g without any supervised fine-tuning to bootstrap. Setting **(7), (8), (9)** stands for 1/4, 1/2, 3/4 of sentences and extractions in

[6]https://github.com/zhanjunlang/Span_OIE

[7]WiRe57 scoring function + Wire57 test set

[8]CaRB scoring function + CaRB test set

[9]WiRe57 scoring function + CaRB test set

	WiRe57			CaRB			CaRB-B		
	P	R	F1	P	R	F1	P	R	F1
ClausIE	0.401	0.298	0.342	0.411	**0.496**	0.450	0.580	0.534	0.556
MINIE	0.400	0.323	0.358	0.429	0.382	0.404	0.466	0.436	0.441
OpenIE6	**0.465**	**0.326**	**0.383**	0.589	0.476	0.527	0.478	**0.671**	0.559
Multi^{2}OIE	0.457	0.182	0.261	0.609	0.458	0.523	0.598	0.613	0.605
CycleOIE	0.388	0.205	0.268	**0.691**	0.427	**0.528**	**0.603**	0.613	**0.608**

Table 4: Overall evaluation. CaRB-B stands for combining the scoring function of CaRB and the dataset of BenchIE, while WiRe57 and CaRB use the scoring function and dataset of their own. In this table, the author-published baselines are evaluated, which means the training set is defined by their author and probably not unified. OpenIE6 was trained on the OpenIE4 training dataset used to train IMoJIE. Mult2OIE is trained on the training dataset of SpanOIE[6](Zhan and Zhao, 2020).

scoring function			CaRB			WiRe57		
model	training set	test set	P	R	F1	P	R	F1
OpenIE6	LSOIE	WiRe57	0.311	0.247	0.275	0.311	0.194	0.239
Multi^{2}OIE	LSOIE	WiRe57	**0.440**	0.202	0.276	**0.440**	0.128	0.198
CycleOIE	LSOIE-g	WiRe57	0.385	**0.271**	**0.318**	0.388	**0.205**	**0.268**
OpenIE6	LSOIE	CaRB	0.403	0.389	0.396	-	-	-
Multi^{2}OIE	LSOIE	CaRB	0.611	0.369	0.461	-	-	-
CycleOIE	LSOIE-g	CaRB	**0.691**	**0.427**	**0.528**	-	-	-

Table 5: Above the dashline, we compare our CycleOIE, which is trained on LSOIE-g to those models trained on LSOIE wiki training set, on the test data of WiRe57 with CaRB's scoring function and WiRe57's scoring function respectively. Below the dash line, we compare our CycleOIE, which is trained on LSOIE-g, to those models trained on LSOIE wiki training set, on the test data of CaRB, with scoring function of CaRB.

LSOIE-g is sampled as training set respectively.

In settings above, when models are trained on LSOIE-g in supervised fine-tuning, the LSOIE-g is paralleled, which means the sentences and the corresponding extractions are paired up. When models are trained on LSOIE-g in CycleOIE, LSOIE-g is unpaired to simulate the real world scenarios, which means sentences and extractions are shuffled and loaded separately.

Table 6 indicates that the **(10) CycleOIE** setting, without any supervised fine-tuning bootstrap, achieves similar performance with *(6) sft on LSOIE+IMoJIE → CycleOIE*, demonstrating the promise of CycleOIE in real-world OpenIE sceneriios, where annotation data is often insufficient.

Comparing setting **(1)** and **(2)**, we observe that augmenting the training set with IMoJIE significantly improves recall, albeit at the cost of precision. Comparing **(3)(4)** or **(5)(6)** to **(1)(2)**, it is observed that once the model is trained on LSOIE-g (either in sft or CycleOIE), precision and f1 show significant improvements. This suggests that our built LSOIE-g is effective as a training dataset for CaRB and CaRB-B. Our designed prompt words, which are derived from BenchIE annotation guidelines, successfully instruct gpt-3.5 to be a qualified annotator.

Comparing **(5)** to **(3)**, the former exhibits higher performances in precision, recall and f1. Comparing **(6)** to **(4)**, the former demonstrates advantages in recall and F1, but the gap, taking F1 as the main metric for comparison, is smaller than the gap between **(5)** and **(3)**. This aligns with our understanding that cycle training can be considered a form of continuous data augmentation along with the training process, significantly improving the performance when the model has not yet acquired enough domain knowledge from existing training data. Conversely, When the existing training data contains sufficient domain knowledge for this task, cycle training doesn't have a prominent advantage over sft.

We observe that **(5)**, **(6)** and **(10)** have similar performances, indicating the model cycle trained

	CaRB			CaRB-B		
	P	R	F1	P	R	F1
(1) sft on LSOIE	0.573	0.339	0.426	0.488	0.459	0.473
(2) sft on LSOIE+IMoJIE	0.402	**0.453**	0.426	0.332	0.598	0.427
(3) sft on LSOIE → sft on LSOIE-g	0.654	0.418	0.51	0.592	0.576	0.584
(4) sft on LSOIE+IMoJIE → sft on LSOIE-g	**0.704**	0.415	0.523	**0.636**	0.576	0.604
(5) sft on LSOIE → CycleOIE	0.694	0.424	0.526	0.600	0.589	0.594
(6) sft on LSOIE+IMoJIE → CycleOIE	0.694	0.428	**0.530**	0.618	0.602	**0.610**
(7) CycleOIE(1/4 LSOIE-g)	0.557	0.386	0.456	0.521	0.56	0.54
(8) CycleOIE(1/2 LSOIE-g)	0.616	0.412	0.494	0.557	0.577	0.567
(9) CycleOIE(3/4 LSOIE-g)	0.651	0.419	0.510	0.594	0.598	0.596
(10) CycleOIE	0.691	0.427	0.528	0.603	**0.613**	0.608

Table 6: Ablation study

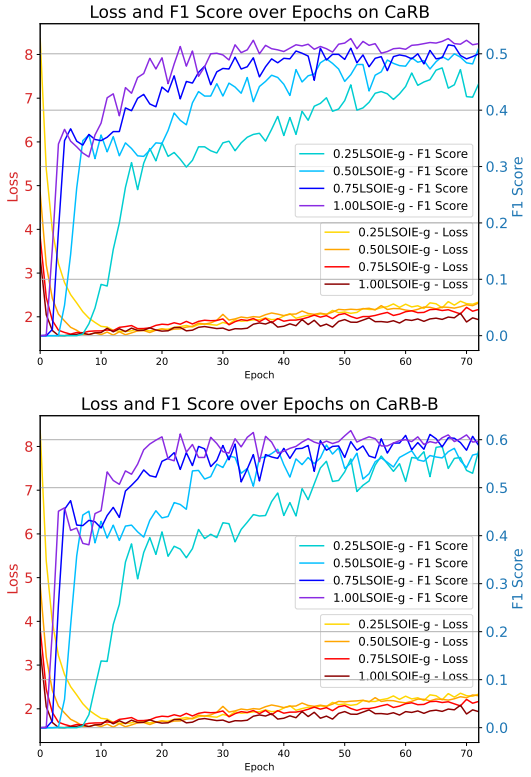

Figure 2: Loss and F1 score of each epoch.

on LSOIE-g can acquire sufficient knowledge to solve the OpenIE task and hit the test data of CaRB. Supervised fine-tuning on weakly labeled large-scale training sets helps to bootstrap but provide almost no additional domain knowledge compared to CycleOIE on LSOIE-g.

The bottom four lines demonstrates the performance of CycleOIE given different scale of LSOIE-g training data. We observe that the model performs better on every metric with an increasing scale of training data, indicating our built LSOIE-g is suitable as the training set of CycleOIE, especially when target benchmark is CaRB and CaRB-B. More detailed influences caused by the training data size is illustrated in Figure 2 where we record the loss θ and F1 score of each epoch of CycleOIE. We observe that training on 1.00LSOIE-g, the loss descends in the fastest rate and then remains at the lowest position and the F1 score converges in the first place at after about 20 epochs. The fluctuation of the F1 score also gets smaller when CycleOIE is conducted on a larger training set.

6 Conclusion

In this paper, we present CycleOIE, a novel framework training a pair of inverse text-to-text model to address OpenIE. We design a template to ensure extractions align with the input and output format of T5. Leveraging the capabilities of gpt-3.5, we construct LSOIE-g as the training set for CycleOIE. Experimental results demonstrate that our approach outperforms baselines trained under full supervision on much larger-scale OpenIE training sets.

Limitations

We implemented CycleOIE based on T5 without introducing novel model structures or task pipelines for further performance enhancement. Due to current constraints related to available GPUs and time limitations, we refrained from applying CycleOIE to fine-tune large language models such as LLaMA(Touvron et al., 2023) and ChatGLM[10](Du et al., 2022), which are more powerful generative models. A speculative avenue for future exploration is the potential of large language models to

[10]https://github.com/THUDM/ChatGLM2-6B

seamlessly transition between sentence-generating and extraction-generating modes by altering instructions after instruction tuning. This implies the possibility of training a single model instead of a pair. We aspire to delve into this aspect of large language models in future research.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. 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 344–354, Beijing, China. Association for Computational Linguistics.

Sangnie Bhardwaj, Samarth Aggarwal, and Mausam Mausam. 2019. CaRB: A crowdsourced benchmark for open IE. 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)*, pages 6262–6267, Hong Kong, China. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Lei Cui, Furu Wei, and Ming Zhou. 2018. Neural open information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 407–413, Melbourne, Australia. Association for Computational Linguistics.

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366.

Zi-Yi Dou, Antonios Anastasopoulos, and Graham Neubig. 2020. Dynamic data selection and weighting for iterative back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5894–5904, Online. Association for Computational Linguistics.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, Dublin, Ireland. Association for Computational Linguistics.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia. Association for Computational Linguistics.

Kiril Gashteovski, Rainer Gemulla, and Luciano del Corro. 2017. MinIE: Minimizing facts in open information extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2630–2640, Copenhagen, Denmark. Association for Computational Linguistics.

Kiril Gashteovski, Mingying Yu, Bhushan Kotnis, Carolin Lawrence, Mathias Niepert, and Goran Glavaš. 2022. BenchIE: A framework for multi-faceted fact-based open information extraction evaluation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4472–4490, Dublin, Ireland. Association for Computational Linguistics.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Andrea Iovine, Anjie Fang, Besnik Fetahu, Oleg Rokhlenko, and Shervin Malmasi. 2022a. Cyclener: An unsupervised training approach for named entity recognition. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 2916–2924, New York, NY, USA. Association for Computing Machinery.

Andrea Iovine, Anjie Fang, Besnik Fetahu, Jie Zhao, Oleg Rokhlenko, and Shervin Malmasi. 2022b. CycleKQR: Unsupervised bidirectional keyword-question rewriting. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11875–11886, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. 2020a. OpenIE6: Iterative Grid Labeling and Coordination Analysis for Open Information Extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3748–3761, Online. Association for Computational Linguistics.

Keshav Kolluru, Samarth Aggarwal, Vipul Rathore, Mausam, and Soumen Chakrabarti. 2020b. IMoJIE: Iterative memory-based joint open information extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5871–5886, Online. Association for Computational Linguistics.

William Lechelle, Fabrizio Gotti, and Phillippe Langlais. 2019. WiRe57 : A fine-grained benchmark for open information extraction. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 6–15, Florence, Italy. Association for Computational Linguistics.

Kevin Pei, Ishan Jindal, Kevin Chen-Chuan Chang, ChengXiang Zhai, and Yunyao Li. 2023. When to use what: An in-depth comparative empirical analysis of OpenIE systems for downstream applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 929–949, Toronto, Canada. Association for Computational Linguistics.

Youngbin Ro, Yukyung Lee, and Pilsung Kang. 2020. Multiˆ2OIE: Multilingual open information extraction based on multi-head attention with BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1107–1117, Online. Association for Computational Linguistics.

Jacob Solawetz and Stefan Larson. 2021. LSOIE: A large-scale dataset for supervised open information extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2595–2600, Online. Association for Computational Linguistics.

Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2300–2305, Austin, Texas. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Zhuoer Wang, Marcus Collins, Nikhita Vedula, Simone Filice, Shervin Malmasi, and Oleg Rokhlenko. 2023. Faithful low-resource data-to-text generation through cycle training. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2847–2867, Toronto, Canada. Association for Computational Linguistics.

Hao-Ran Wei, Zhirui Zhang, Boxing Chen, and Weihua Luo. 2020. Iterative domain-repaired back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5884–5893, Online. Association for Computational Linguistics.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael Cafarella, Oren Etzioni, and Stephen Soderland. 2007. TextRunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26, Rochester, New York, USA. Association for Computational Linguistics.

Junlang Zhan and Hai Zhao. 2020. Span model for open information extraction on accurate corpus. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9523–9530.