COMBO: A Complete Benchmark for Open KG Canonicalization

Chengyue Jiang[✿☾†], **Yong Jiang**[☆*], **Weiqi Wu**[✿], **Yuting Zheng**[✿],
Pengjun Xie[☆], **Kewei Tu**[✿☾*]
[✿]School of Information Science and Technology, ShanghaiTech University
[☾]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[☆]DAMO Academy, Alibaba Group, China
{jiangchy,tukw,wuwq}@shanghaitech.edu.cn;
{yongjiang.jy,chengchen.xpj}@alibaba-inc.com

Abstract

Open knowledge graph (KG) consists of *(subject, relation, object)* triples extracted from millions of raw text. The *subject* and *object* noun phrases and the *relation* in open KG have severe redundancy and ambiguity and need to be canonicalized. Existing datasets for open KG canonicalization only provide gold entity-level canonicalization for noun phrases. In this paper, we present **COMBO**, a **Com**plete **B**enchmark for **O**pen KG canonicalization. Compared with existing datasets, we additionally provide gold canonicalization for relation phrases, gold ontology-level canonicalization for noun phrases, as well as source sentences from which triples are extracted. We also propose metrics for evaluating each type of canonicalization. On the **COMBO** dataset, we empirically compare previously proposed canonicalization methods as well as a few simple baseline methods based on pretrained language models. We find that properly encoding the phrases in a triple using pretrained language models results in better relation canonicalization and ontology-level canonicalization of the noun phrase. We release our dataset, baselines, and evaluation scripts at https://github.com/jeffchy/COMBO/tree/main.

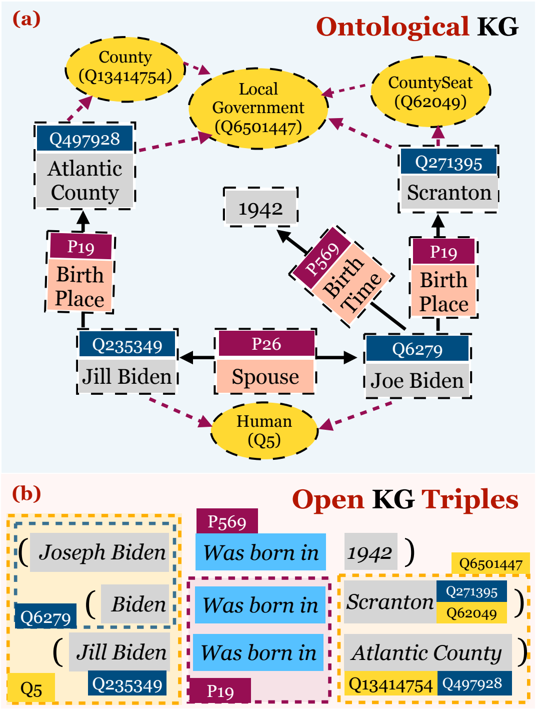


Figure 1: Example of ontological KG (a) and Open KG triples (b). The differently colored bounding boxes and the tags on the open KG triples illustrate three types of gold canonicalization. Yellow (e.g., *Q5* Human) shows the gold ontology-level NP cluster, the salvia blue (e.g., *Q6279*) indicates the gold entity-level NP cluster, and the purple (e.g., P19) indicates gold RP cluster.

1 Introduction

Large ontological knowledge graphs (KG) such as Wikidata (Vrandečić and Krötzsch, 2014), DBpedia (Bizer et al., 2009), Freebase (Bollacker et al., 2008) use a complex ontology to formalize and organize all the entities and relations. Figure 1(a) shows an example ontological knowledge graph (Wikidata): "*Joe Biden (Q6279)*" is categorized as "*Human (Q5)*" in Wikidata and linked to "*Scranton (Q271395)*" with relation "*birth place (P19)*", where prefix *Q* and *P* denote unique identities for entity and relation respectively in Wikidata[1]. As ontological KGs are well organized and canonicalized, one can efficiently query information and extract knowledge from them to assist NLP models in various tasks (Rao et al., 2013; Luo et al., 2015; Cui et al., 2019; Murty et al., 2018; Wang et al., 2021; Liu et al., 2023; Gao et al., 2022; Liu et al., 2022). However, building and maintaining an accurate ontological KG requires large human effort (Färber et al., 2015).

In contrast, open knowledge graphs such as Re-

[†] This work was done during Chengyue Jiang's internship at DAMO Academy, Alibaba Group.

[*] Yong Jiang and Kewei Tu are corresponding authors.

[1] Wikidata links https://www.wikidata.org/wiki/Q5, https://www.wikidata.org/wiki/Property:P19

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 340–357
May 2-6, 2023 ©2023 Association for Computational Linguistics

Verb (Fader et al., 2011) and OLLIE (Mausam et al., 2012) are built using *(subject, relation, object)* triples automatically extracted from millions of raw text by OpenIE systems (Angeli et al., 2015; Fader et al., 2011; Mausam et al., 2012). They are frequently used to assist in building ontological KGs (Martinez-Rodriguez et al., 2018; Dessì et al., 2021) and slot filling (Broscheit et al., 2017). As OpenIE systems do not rely on pre-defined ontologies or human supervision, the extracted triples contain noun phrases (NPs) and relation phrases (RPs) that are not canonicalized. Take the open KG triples shown in Figure 1(b) as an example. The NP *"Joseph Biden"* and *"Biden"* both refer to the US president Joe Biden, but the open KG regards them as two different nodes because of their different surface forms. On the other hand, *"was born in"* in the first and second triple means *"birth place of"* and *"birth time of"* respectively, but the open KG cannot disambiguate them. These examples reveal the redundancy and ambiguity of uncanonicalized open KG (Vashishth et al., 2018), which makes querying open KG inaccurate and inefficient. To this end, open KG canonicalization aims to improve the quality of open KGs to the level of ontological KGs. It is therefore different from tasks such as entity linking (Rao et al., 2013) and KB aligning (Elsahar et al., 2018) that align entity mentions or sentences to an existing ontological KG.

Existing open KG canonicalization datasets such as ReVerb-base, ReVerb-ambiguous (Galárraga et al., 2014), ReVerb45K (Vashishth et al., 2018) and CanonicNELL (Dash et al., 2021) mainly focus on entity-level canonicalization of NPs, providing the gold **E**ntity-level **NP** **C**anonicalization (**NPC-E**). The blue tags and dashed boxes in Figure 1(b) show examples of **NPC-E**, e.g., *"Biden"* and *"Joseph Biden"* should be canonicalized as the same entity *Q6279*. However, these datasets do not provide the gold **RP** **C**anonicalization (**RPC**), and do not consider the **O**ntology-level **C**anonicalization of **NP** (**NPC-O**). **RPC** is to canonicalize RPs that mean same relation together, for example, the second and the third *"was born in"* in Figure 1(b) should be canonicalized into the same cluster of *birth place (P19)*, different from the first one which means *"birth time (P569)"*. Similarly, **NPC-O** is to canonicalize NPs that have same type together, for example, the *"Scranton"* should be canonicalized into class *"CountySeat"* and into class *"Local Government"* together with *"Atlantic County"*, it can be viewed as canonicalizing special ontological relations such as "*instance of*", "*subclass of*" represented by dotted arrows in Fig. 1(a). We formally define these tasks in Sec. 3.

RPC and **NPC-E** are important as parts of a canonicalization benchmark (1) Relations and ontology are necessary for an expressive KG (Klyne and Carroll, 2004) (2) Most KG queries involve relations and ontology (e.g., the query *"actress that was born in California"*, involve the relational constraint "*X, birth place, California*" and the ontological constraint "*X, instance of, Actress*").

In this paper, we present **COMBO**, a complete benchmark for open KG canonicalization consisting of three subtasks: besides **NPC-E** which has been adequately studied in previous work, we additionally provide gold **RPC** and **NPC-O** along with their evaluation metrics. Gold **NPC-O** is obtained by querying the Wikidata using SPARQL, and **RPC** is obtained by performing Stanford OpenIE on sentences from Wiki20 (a distantly labeled relation extraction dataset), and a per-instance human revision process to ensure the quality of extracted RPs. We introduce the data construction process detailedly in Sec. 4.

Our new benchmark makes it possible for the first time to quantitatively evaluate the full range of open KG canonicalization. We conduct comprehensive experiments to compare existing canonicalization methods as well as a few simple baseline methods proposed by us. Somewhat surprisingly, none of the existing methods utilizes pretrained contextualized word embedding, probably because previous work only focuses on NPC-E and NPs are often not very ambiguous, making contextualization not so helpful. For example, the "*Joe Biden*" and "*Joseph Biden*". However, contexts are more helpful in RPC and NPC-O. For RPC, relations are more ambiguous and diverse in surface forms (e.g., "*was born in*" in Figure 1) and contexts are needed for disambiguation. For NPC-O, the RP and the other NP in the triple will help understand the type of an NP. Therefore, our proposed baseline methods are based on pretrained language models (PLM) (Devlin et al., 2019; Liu et al., 2019b; Sun et al., 2019) which produce contextualized embedding and have been shown to contain a certain amount of factual knowledge (Petroni et al., 2019; Lauscher et al., 2020). We found that, after properly encoding triples and contexts, our baseline methods outperform well on all three subtasks compared with previous state-of-the-art methods, especially on RPC and NPC-O. We also propose a triple-based pretraining method and find that it further boosts the performance on all subtasks. Therefore, our work provides strong baselines for future research on open KG canonicalization.

In summary, our contributions are threefold. First, we propose a complete definition of the open KG canonicalization problem along with the metrics. Second, we construct the complete benchmark for open KG canonicalization consisting of entity-level and ontology-level NP canonicalization and RP canonicalization. Third, we propose a stronger baseline based on autoencoding PLMs and conduct a comprehensive empirical comparison of canonicalization methods on our benchmark.

2 Open KG Canonicalization Datasets

We introduce existing open KG canonicalization datasets and **COMBO**. The statistics of datasets are shown in Table 1.

ReVerb-Base (Galárraga et al., 2014) Constructed using the ReVerb open KB. As half of the NPs in ReVerb triples are linked to an entity in the ontological database FreeBase (Bollacker et al., 2008), the authors sample 150 FreeBase entities that have at least two surface forms, collect all triples containing these 150 entities, and use the entity labels as the gold NP clusters.

ReVerb-Ambiguous (Galárraga et al., 2014) ReVerb-Ambiguous is constructed similarly as ReVerb-Base, it has 37K triples, but with only 445 gold NP clusters (entities). One problem with the ReVerb-Base and ReVerb-Ambiguous datasets is they contain too few NP clusters and too many NP aliases, which is inconsistent with real open KGs.

ReVerb45K(Vashishth et al., 2018) ReVerb45K increases the entity number to 7.5K and has 45K triples in total. ReVerb45K, Reverb-Base, and ReVerb-Ambiguous extract a source sentence for each triple from ClueWeb09 Callan et al. (2009). However, some of the source sentences are simply the concatenation of triples.

CanonicNELL (Dash et al., 2021) Constructed using the open KB NELL (Mitchell and Fredkin, 2014) and the entity linking information for NPs (Pujara et al., 2013). They remove triples containing NPs without aliases. CanonicNELL does not provide source sentences.

COMBO (Ours) As shown in the Table 1, the main differences of our dataset between others are that we additional provide gold RP canonicalization and ontology-level NP canonicalization. Constructed based on the large Ontological KG Wikidata[2], the OpenIE system, a relation extraction dataset Wiki20m and human revisions, as detailed in next section. Our dataset contains 18K triples with their source sentences and we provide gold NPC-E, RPC and NPC-O annotations. We compare **COMBO** with existing datasets in Table 1. Although our dataset is middle-sized, it has the longest average triple length and the largest number of unique NPs, indicating the diversity of the surface forms of NPs and RPs. Providing source sentences of extracted OpenIE triples is natural but important since additional contextual information can be helpful in understanding and disambiguating NPs and RPs. We ensure all triples contain rich context, and the average length of source sentences is 21. We show some data samples in Appendix A, and analyze our data in Sec. 4.

3 Task Definition and Evaluation Metrics

Task Definition The goal of open KG canonicalization is to assign NPs and RPs in triples into clusters, such that NPs that refer to the same entity (NPC-E) or have the same type (NPC-O) are clustered together, and similarly, RPs that refer to the same relation are clustered together. Note that the task is unsupervised, meaning that the canonicalizer does not have access to gold annotations. We have N samples containing triples and their corresponding source sentences: $\mathcal{T} = \{c_i, t_i = (s_i, r_i, o_i) | \, i = 1 \dots N\}$, where c_i is the i-th sentence, t_i is the i-th triple containing subject NP s_i, RP r_i, and object NP o_i. $\mathcal{S} = \{(s_i, i) | \, i = 1 \dots N\}$ is the indexed subject NP set. The indexed RP set $\mathcal{R}$ and object NP set $\mathcal{O}$ are defined similarly as $\mathcal{R} = \{(r_i, i) | \, i = 1 \dots N\}$ and $\mathcal{O} = \{(o_i, i) | \, i = 1 \dots N\}$. We have $|\mathcal{S}| = |\mathcal{O}| = |\mathcal{R}| = N$. The gold NPC-E, RPC, and NPC-O annotations are defined as sets of clusters. As subject NPs and object NPs are asymmetric (Juffs and Harrington, 1995; McGinnis, 2002), we follow Vashishth et al. (2018) and evaluate the clusters of subject NPs and object NPs separately. The gold NPC-E and NPC-O for subject NPs are defined as *NPC-E (Subj)* $= \{C_1 \dots C_{K_s^E}\}$, *NPC-O (Subj)* $= \{C_1 \dots C_{K_s^O}\}$, where C_i denotes the i-th cluster of NP. The NPC-E and NPC-O

[2] The official suggested replacement of Freebase after it retired shorturl.at/kmnBR

	# NP	# NPC-E	# RP	# RPC	# NPC-O	# Triples	Avg triple len	Context (%)
ReVerb-Base	290	150	3K	✗	✗	9K	5.26	78%
ReVerb-Ambiguous	717	446	11K	✗	✗	37K	5.27	78%
ReVerb45K	15.5K	7.5K	22K	✗	✗	45K	6.17	91%
CanonicNELL	8.7K	1.4K	139	✗	✗	20K	6.38	✗
COMBO (Ours)	16.5K	13.8K	3.2K	79	2946	18K	8.12	100%

Table 1: Statistics and comparison of open KG canonicalization datasets including ours. ✗ means not available in the dataset (i.e., zero). "Avg triple len" is the average number of words in the triple. The last column shows the ratios of triples containing additional context in their source sentences.

	NPC-E	*RPC*	*NPC-O*
Gold	non-overlapping	non-overlapping	overlapping
Predicted	non-overlapping	non-overlapping	non-overlapping / overlapping
Metric	Ma, Mi, Pair	Mi, Pair	Ma, Mi, Pair / $J_{g\to p}$, $J_{p\to g}$

Table 2: Evaluation of the three subtasks. Ma, Mi, Pair are abbreviations of macro, micro and pairwise metrics.

of object NPs are defined similarly, the gold RPC is defined as *RPC* as $\{C_1 \ldots C_{K_r}\}$. *NPC-E (Subj)* is a *non-overlapping* cluster assignment and satisfies two conditions: (1) $\bigcup_{i=1}^{K_s^E} C^i = \mathcal{S}$; (2) $C_i \cap C_j = \emptyset, i \neq j$. *NPC-E (Obj)* and *RPC* satisfy similar conditions. *NPC-O (Subj)* and *NPC-O (Obj)* are *overlapping* cluster assignments, i.e., we allow an NP to belong to multiple clusters, so they only need to satisfy the first condition. The task is to predict the cluster assignments of NPs and RPs given their source triples and sentences. Following previous works, we assume the cluster number is unknown beforehand and split our data into the dev (20%) and test (80%) sets.

Task Evaluation Most clustering algorithms such as K-means (Lloyd, 1982) and Hierarchical Agglomerative Clustering (HAC) (Maimon and Rokach, 2005) produce non-overlapping cluster assignments, and several algorithms (e.g., HAC) can also produce hierarchical and overlapping cluster assignment. For the *NPC-E* subtask, we adopt the classic macro, micro and pairwise metrics to compare the gold and predicted *NPC-E* cluster assignments (please refer to App. C for details). For *RPC*, the macro metrics that calculate the fractions of pure clusters are too strict because gold RP clusters are large and hence are unlikely to be pure. Therefore we only use the micro and pairwise metrics to evaluate *RPC*.

For *NPC-O*, the gold cluster assignments are overlapping. If the predicted clusters are non-overlapping, we can apply the macro and pairwise metrics and a modified micro metric (Appendix D). If the predicted clusters are overlapping, say $\mathcal{P} = \{C_1^p \ldots C_M^p\}$, we propose evaluation metrics $J_{g\to p}$ and $J_{p\to g}$ based on the Jaccard index (Jaccard, 1908; Tanimoto, 1958). $J_{g\to p}$ (Eq.1) calculates the average Jaccard index of a gold cluster and its best matched predicted cluster. $J_{p\to g}$ is similarly defined but with the roles of *NPC-O* and $\mathcal{P}$ switched. Table 2 summarizes the evaluation metrics of each subtask.

$$\text{Jaccard}(g,p) = \frac{|g \cap p|}{|g \cup p|}$$
$$J_{g\to p} = \frac{1}{|\mathit{NPC\text{-}O}|} \sum_{g \in \mathit{NPC\text{-}O}} \max_{p \in \mathcal{P}} \left(\text{Jaccard}(g,p)\right) \quad (1)$$

4 Construction of Our Dataset

We illustrate the construction process of **COMBO** in Figure 2. We rely on the Wiki20 dataset (Han et al., 2020) to obtain the source sentence and the gold NPC-E. Wiki20 is a large multi-domain relation extraction dataset constructed by aligning the Wikipedia corpus with Wikidata using distant supervision. As shown in the bottom of Figure 2, each sample of Wiki20 contains a sentence with the object and subject NP spans labeled and linked to entities in Wikidata and the relation between them is also labeled. To ensure data quality, we use the recently revised version of Wiki20 (Gao et al., 2021), which aligns the Wiki20 relation labels with the supervisedly constructed Wiki80 dataset (Han et al., 2019) and provides 56K human-annotated data samples. The object and subject NP spans and its entity linking information (e.g., *Q6275*) are from Wikipedia and have high precision, so we directly use it for task NPC-E.

Extracting Relational Phrases Wiki20 only provides the relation label of two NPs for each instance. We further extract RP for Wiki20 instances to obtain full open KG triples. We first discard samples with the relation label "NA" and then run the Stanford OpenIE system on Wiki20 sentences to extract triples. We choose Stanford OpenIE[3] because compared with older OpenIE systems such as ReVerb and NELL that are used in constructing previous datasets, Stanford OpenIE can leverage the linguistic structure of a sentence and generalizes better to out-of-domain and longer utterances (Angeli et al., 2015). We empirically find that Stanford OpenIE yields a better recall and can extract more triples per sentence compared to ReVerb. We use the default model configuration of Stanford OpenIE. After obtaining the OpenIE triples of each non-NA Wiki20 instance, we select the triples whose subject NP and object NP are consistent with the NP spans provided by Wiki20. This triple selection step ensures the NPs in the extracted triples have gold NPC-E annotations, and remove wrong relation spans caused by wrongly extracted head and tail entities. We filter out 88% of the original triples through this step. Although this step reduces noises caused by OpenIE, the extracted relation spans could still be wrong in two ways:

1. **Invalid RP between correct NPs**. For example, for sentence *"... the Althing, the ruling legislative body of Iceland ... "*, OpenIE wrongly extracts *(the Althing, body of, Iceland)*, while the true triple should be *(the Althing, ruling legislative body of, Iceland)*.

2. **Correct NPs and valid RP but RP does not imply the relation given by Wiki20**. For the given relation *mother of* and sentence *"... bart and lisa got sent out of the house by marge simpson ... "*, the extracted triple *(lisa, got sent out of the house by, marge simpson)* is valid but cannot imply the *mother of* relation.

Therefore, we manually check all the extracted triples for these two types of errors, correcting invalid relational phrase spans and removing triples whose RP cannot imply the given relation. We also standardize the form of RP (e.g., OpenIE sometimes includes "a" and "the" and sometimes does not). The detailed guidelines for the check and revision process are shown in the Appendix B. The error analysis is shown in Table 3.

After all these steps, we obtain an open KG consisting of 18K triples. Similar to NPC-E, we use the relation labels given by the Wiki20 annotations (e.g., *P19*) as the gold RPC. As shown in Figure 3, the constructed open KG contains 79 relations in various domains, such as relations between geopolitical entities (*mouth of the watercourse* (7.3%), *mountain range* (3.8%), etc), relations between people (*spouse of* (1.7%), *child of* (1.6%), etc), and various relations between people and other objects (*citizenship* (2.4%), *work location* (3.7%), etc). The extracted RPs are diverse in surface forms. The number of distinct RPs is 3.2K. We show RP examples in Table 4. There exist some RPs that represent multiple relations and one representative example is *"in"*.

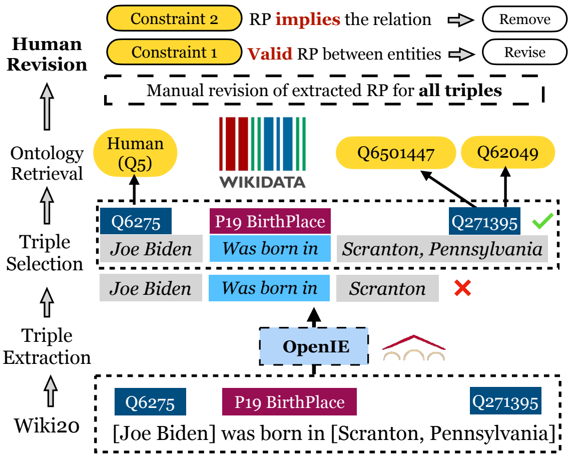


Figure 2: Steps of dataset construction.

Error Type	Rate
Invalid RP	23.5%
RP doesn't imply relation	5.5%

Table 3: OpenIE error analysis.

Extracting Ontology To obtain ontology-level NP clusters for the NPC-O subtask, we query Wiki-

spouse of
was married twice to , was married to, *lover, consort of, second husband, widow of* *'s wife, 's second wife, arranged a wedding with*
mountain range
peak, large nunatak, summits of, the only crossing of the *most prominent feature of, small glacier, summits in* *valley in, only crossing of, northernmost subrange of,* ***in***
location
is headquartered in, moved to, is carved on *took place at, ironworks in, was again held at,* ***in***

Table 4: RP examples.

[3] https://stanfordnlp.github.io/CoreNLP/

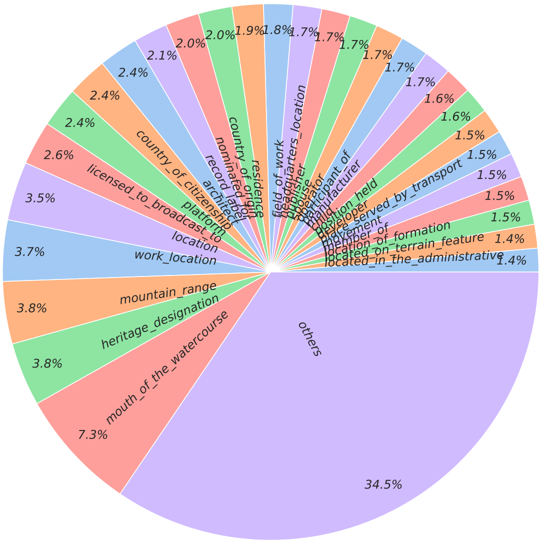


Figure 3: Pie charts of 79 RP clusters.

data for the classes of each entity. For example, to obtain the classes of *"Joe Biden (Q6275)"*, we run the SPARQL (RDF query language) query " `Q6275 P31 ?`", where P31 represents the "instance of" relation in Wikidata. This query obtains all the classes of an NP. If an NP does not have a class, its NPC-O annotation is the same as its NPC-E annotation. If an NP has more than one class, we include all of them in the NPC-O annotation (e.g., *city* and *big city* for "New York"). We query Wikidata using a third-party client Wikidata Integrator[4].As the ontology information in Wikidata is crowdsourced and contains errors, we apply pattern-based corrections to the extracted ontological NP clusters, for example, if an NP belongs to the cluster *million cities*, it should also belong to the cluster *city*. The resulting 2.9K ontological NP clusters form a 6-level overlapping hierarchy which allows a node to have more than one parent. We illustrate part of the hierarchy in Figure 5 and show the statistics of the top 12 ontological NP clusters in Figure 4.

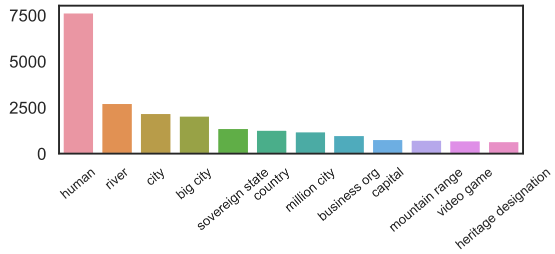


Figure 4: Size of top 12 ontological NP clusters.

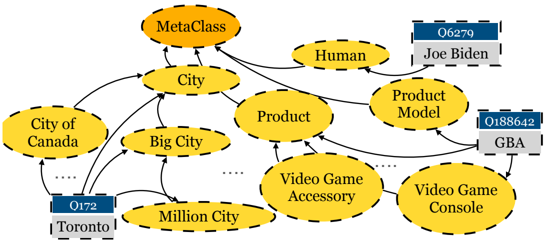


Figure 5: Part of the class hierarchy in our dataset.

5 Comprehensive Evaluation of Methods

Our benchmark makes it possible to conduct a comprehensive empirical comparison of different methods on the full range of open KG canonicalization. Below we first give an overview of existing methods and propose a few new baseline methods. Then we present our experimental settings and results.

5.1 Previous Methods

Non-neural Methods Galárraga et al. (2014) utilizes token features such as TF-IDF scores and Jaccard token similarity to canonicalize NPs. They merge similar NPs based on a threshold (tuned on the validation set) to form clusters. For RPs, they use AMIE (Galárraga et al., 2013), an unsupervised algorithm based on statistical rule mining, to obtain relation clusters. Vashishth et al. (2018) use additional side information obtained from various sources (such as PPDB (Ganitkevitch et al., 2013), WordNet (Miller, 1992)) to produce clusters.

SE-HAC Trivial baseline that performs HAC clustering over phrase embeddings produced by averaging static word embeddings such as GloVe (Pennington et al., 2014) or random embeddings.

CESI Vashishth et al. (2018) encode phrases using the same method as in SE-HAC; then apply the HolE graph embedding algorithm (Nickel et al., 2016) on triples and incorporate side information to finetune the embedding, and finally run HAC clustering on the learned embeddings.

CUVA Dash et al. (2021) adopt VAEGMM (Jiang et al., 2017) to jointly learn and cluster the embeddings. They initialize VAEGMM by performing HAC clustering on GloVe NP and RP embeddings, and then simultaneously optimize the knowledge embedding loss, side information loss, and VAE loss for the final clustering.

Previous methods encode NPs and RPs using either token frequency features or static word embedding. Although CESI and CUVA learn graph embedding on open KG triples, they assign the

[4] https://github.com/SuLab/WikidataIntegrator

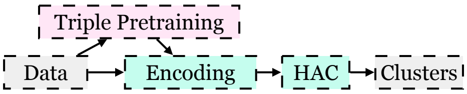


Figure 6: Pipeline of the proposed PLM-based method.

same representation to phrases with the same surface form and therefore cannot deal with ambiguity. No method utilizes original sentences to provide additional contexts. HAC is a popular choice of the clustering algorithm because it does not require knowing the number of clusters, but instead requires a distance threshold indicating when to stop merging. Unlike the number of clusters, the threshold can be tuned on a validation set and directly applied to the test set.

5.2 PLM-Based Baseline Methods

We propose a set of new baseline methods based on PLMs that produce contextualized embedding. We use a pipeline similar to CESI as shown in Figure 6. We encode NPs and RPs using different PLMs, PLM layers, and span representation methods and apply HAC clustering over their representations. We use the cosine similarity as the distance function and apply the complete linkage variant of HAC clustering because we prefer compact clusters and the single linkage variant suffers from the chaining phenomenon. Before encoding, an optional triple-level continuous pretraining step can be applied for better canonicalization. Token similarity and other side information are not used in our PLM-based method, but we generate them for our data using the code provided by Vashishth et al. (2018) to facilitate running of other methods.

5.2.1 Encoding

PLMs We use autoencoding PLMs[5] including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), ERNIE2.0 (Sun et al., 2019) which integrates entity information, and SpanBERT (Joshi et al., 2020) which is pretrained with span masks.

Input Given a triple $t_i = (s_i, r_i, o_i)$ and its corresponding source sentence c_i, we formulate the input of PLM in the following four ways to obtain the contextualized embedding of words in the NPs and RP. Note that the fourth method *sep* independently encodes each phrase in the triple.

$$\begin{aligned} \textit{sentence:}\quad & [CLS]\ \ldots s_i \ldots r_i \ldots o_i \ldots\ [SEP] \\ \textit{triple:}\quad & [CLS]\ s_i\ r_i\ o_i\ [SEP] \\ \textit{triple-sep:}\quad & [CLS]\ s_i\ [SEP]\ r_i\ [SEP]\ o_i\ [SEP] \\ \textit{sep:}\quad & [CLS]\ s_i/r_i/o_i\ [SEP] \end{aligned} \tag{2}$$

Phrase Representation After obtaining the contextualized embedding of words in an NP or RP span, denoted as $h_i \ldots h_j$, we follow Toshniwal et al. (2020) and use three methods to produce a single span embedding. Following Timkey and van Schijndel (2021), we also standardize the embeddings to remove rogue dimensions (Appendix E). Previous work (Vulić et al., 2020; Liu et al., 2019a) shows that different layers of a PLM contain different information, so we investigate contextualized embedding from different layers .

$$\begin{aligned} \textit{mean:}\quad & average\_pooling(h_i \ldots h_j) \\ \textit{max:}\quad & max\_pooling(h_i \ldots h_j) \\ \textit{diff-sum:}\quad & [h_i - h_j; h_i + h_j] \end{aligned} \tag{3}$$

5.2.2 Triple-level Pretraining

Inspired by the HolE algorithm used in previous works (Vashishth et al., 2018; Dash et al., 2021), we may perform an optional triple-level continuous pretraining step before encoding to mimic the link prediction objectives in knowledge graph embedding learning. For each sentence in our dataset, we randomly mask a phrase in the triple and then train the PLM to predict the whole masked span. We perform pretraining for 10 epochs using the AdamW optimizer (Loshchilov and Hutter, 2019) with a linear scheduler and a start learning rate of 5e-5. We then use the continuously pretrained version of PLM as the phrase encoder. We also use the causal subword-level MLM strategy in BERT (Devlin et al., 2019) for comparison.

5.3 Experimental Setup

For each subtask, we use grid search to tune the HAC distance threshold on the dev set to obtain non-overlapping clusters for all the methods. We select the best threshold based on the average of the metrics shown in Table 2. We obtain overlapping clusters for NPC-O from the full HAC hierarchy. As HAC is deterministic, we run experiments once for methods without randomness and four times for methods involving random initialization (CUVA, Random+HAC). As Token Sim+SI and VAEGMM based methods cannot provide overlapping cluster assignments, we do not evaluate them by metrics based on the Jaccard index. For our PLM-based

[5] Huggingface models https://huggingface.co/

	NPC-E		***RPC***	***NPC-O***			
	Subj	***Obj***	***Relation***	***Subj***	***Subj-Jaccard***	***Obj***	***Obj-Jaccard***
Token Sim+SI (Galárraga et al., 2014)	82.90	79.35	33.94	33.14	-	40.59	-
Random+HAC	**85.32**	85.11	35.98	37.31	38.79	44.90	44.40
GloVe+HAC	78.31	86.57	35.57	38.00	39.04	47.35	45.45
GloVe+HolE+HAC (Vashishth et al., 2018)	80.13	87.33	17.93	39.63	39.87	48.25	47.18
GloVe+SI+HAC (Vashishth et al., 2018)	80.42	**87.34**	17.91	39.87	39.83	**48.26**	47.11
CESI (Vashishth et al., 2018)	80.11	86.82	18.37	**39.93**	**39.89**	48.25	**47.19**
VAEGMM+SI (Dash et al., 2021)	80.86	82.91	34.10	37.41	-	46.90	-
VAEGMM+HolE (Dash et al., 2021)	80.15	82.87	36.12	37.22	-	46.88	-
CUVA (Dash et al., 2021)	80.68	82.95	**36.13**	37.09	-	46.89	-
Bert-base	**86.93**	86.91	54.47	**42.97**	**44.16**	50.71	46.99
Roberta-base	82.85	85.00	41.08	39.21	41.24	46.07	44.14
SpanBert-base	84.38	86.32	44.04	41.61	43.16	47.29	45.35
ERNIE2.0-base	86.68	**88.11**	**54.66**	42.60	44.05	**52.27**	**47.05**
Bert-base-triple	86.01	**88.92**	**58.45**	**43.71**	**45.19**	**51.78**	**47.49**
Roberta-base-triple	85.37	87.22	50.81	42.29	44.33	50.31	46.81
SpanBert-base-triple	85.73	85.89	46.18	42.53	44.23	47.60	45.56
ERNIE2.0-base-triple	**87.21**	86.93	57.28	43.21	44.33	50.66	47.27
Bert-large	**87.09**	**89.05**	**50.31**	42.34	**44.16**	50.71	**47.25**
Roberta-large	83.50	85.81	40.51	39.88	42.35	48.35	45.58
SpanBert-large	86.32	86.67	45.84	40.96	42.90	47.92	45.68
ERNIE2.0-large	86.21	88.86	49.80	**42.71**	44.01	**51.98**	47.22

Table 5: Averaged metrics (of Table 2) on all the subtasks. For example, ***NPC-O, Subj*** is the average of Ma,Mi and Pair metrics on the ontology-level canonicalization of subject NPs, and ***NPC-O, Obj-Jaccard*** is the average of $J_{p\rightarrow g}$ and $J_{g\rightarrow p}$ for the overlapping clustering assignment of object NPs. Full results including the results of *large-triple* models are shown in Appendix F

methods, we select the best input form and span representation strategy based on the dev set performance. We also compare different **encoding strategies** in Appendix G, and **layer-wise performances** in Appendix H.

5.4 Overall Results

We report averaged metrics for each subtask in Table 5 because of limited space. The full results are shown in Appendix F. The results show that our PLM-based baseline methods outperform previous methods in most cases, especially on RPC and NPC-O, indicating the importance of contextual information. Trivial baselines such as Token Sim+SI, Random+HAC and GloVe+HAC already perform well (around 80%) on NPC-E, because NPs referring to the same entity usually have similar surface forms and do not have to rely on contexts for correct prediction. However, they perform badly on RPC and NPC-O, because surface forms alone are no longer adequate for these two subtasks because of higher ambiguity. CESI has bad RPC performance but is very competitive on NPC-E (Obj), and better than SpanBERT and RoBERTa without triple-level pretraining, but is still worse than the other PLM-based methods. CUVA performs generally badly, probably because it is sensitive to VAEGMM initialization and relies heavily on side information. As our dataset has the longest average triple length and consists of texts from various domains, it could be more challenging for methods that do not use contextualized embedding.

For PLM-based methods, BERT leads to the best overall performance on NPC-E (Obj), RPC and NPC-O (Subj); ERNIE2.0 performs best on NPC-E (Subj) and NPC-O (Obj) and is comparable to Bert on NPC-E (Obj) and RPC; RoBERTa and SpanBERT fall behind, but are still better than most other non-PLM methods on NPC-O and RPC. Large PLMs are better than base PLMs on NPC-E, comparable on NPC-O, but worse on RPC. We also found the triple-level pretraining effective, having a positive influence in most cases, especially on RPC (e.g., +9.73 for RoBERTa). In contrast, using the causal subword-level pretraining for Bert improves the object NPC but harms the subject NPC and RPC (-1.07 points). A detailed comparison between triple-level and subword-level pretraining is shown in Appendix I.

6 Conclusion

We present **COMBO**, a complete benchmark for open KG canonicalization. **COMBO** consists of three subtasks, entity-level and ontology-level NP canonicalization, and RP canonicalization. We construct the data and propose the evaluation metrics for the RPC and NPC-O that are not been adequately studied before. We also propose a stronger canonicalization method based on autoencoding PLMs and conduct a comprehensive comparison of different canonicalization methods on our dataset.

For future study, NPC-O and RPC still have a lot of room for improvement and the efficiency of canonicalization methods is also worth studying. We also note that **COMBO** can be additionally used as a probing benchmark for PLMs and as a phrase-level relation classification dataset.

7 Acknowledgement

This work was supported by the National Natural Science Foundation of China (61976139) and by Alibaba Group through Alibaba Innovative Research Program.

Limitations

One limitation of our work is that, the size of our dataset (18K) is relatively small compared to previous datasets (Table 1). Another limitation is that, similar to previous work, we perform clustering for three subtasks and evaluate the canonicalization results independently, but canonicalization of the head NP, tail NP and RP is in fact closely correlated. For example, the NPC-O clusters of the head NP and tail NP reveal the domain and range of the relation given by RPC. We leave jointly canonicalization and evaluation as future work. Our proposed baseline is straightforward. We encourage future studies to investigate better canonicalization methods based on pretrained language models.

Ethics Statement

Our dataset is constructed based on Wiki20 and Wikidata. The two sources are both publicly available. Wiki20 is under the MIT Licence and the Wikidata is under the Creative Commons CC0 License. Both of them allow modification and distribution. Regarding human revision during dataset construction, the annotators were properly paid. The annotating procedure lasted 12 days and the daily workload was relatively light: around 2.5 hours per day. During human inspection, we did not identify any unethical instances in our dataset. Regarding baseline models, we use PLMs as our text encoder and our task is inherently unsupervised. As PLMs are learned on large corpora, our method can potentially create biased clustering results. How to de-bias PLM embedding is worth further investigation.

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.

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. Dbpedia - a crystallization point for the web of data. *Journal of Web Semantics*, 7(3):154–165. The Web of Data.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Samuel Broscheit, Kiril Gashteovski, and Martin Achenbach. 2017. Openie for slot filling at tac kbp 2017-system description. In *TAC*.

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. Clueweb09 data set.

Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. 2019. Kbqa: learning question answering over qa corpora and knowledge bases. *arXiv preprint arXiv:1903.02419*.

Sarthak Dash, Gaetano Rossiello, Nandana Mihindukulasooriya, Sugato Bagchi, and Alfio Gliozzo. 2021. Open knowledge graphs canonicalization using variational autoencoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10379–10394.

Danilo Dessì, Francesco Osborne, Diego Reforgiato Recupero, Davide Buscaldi, and Enrico Motta. 2021. Generating knowledge graphs by employing natural language processing and machine learning techniques within the scholarly domain. *Future Generation Computer Systems*, 116:253–264.

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

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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 55–65, Hong Kong, China. Association for Computational Linguistics.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Michael Färber, Basil Ell, and Carsten Menne. 2015. A comparative survey of dbpedia , freebase , opencyc , wikidata , and yago.

Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. 2014. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 1679–1688. ACM.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 413–422. International World Wide Web Conferences Steering Committee / ACM.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318, Online. Association for Computational Linguistics.

Yunjun Gao, Xiaoze Liu, Junyang Wu, Tianyi Li, Pengfei Wang, and Lu Chen. 2022. Clusterea: Scalable entity alignment with stochastic training and normalized mini-batch similarities. In *KDD*, pages 421–431.

Adam Grycner, Gerhard Weikum, Jay Pujara, James Foulds, and Lise Getoor. 2015. RELLY: Inferring hypernym relationships between relational phrases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 971–981, Lisbon, Portugal. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758, Suzhou, China. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. 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): System Demonstrations*, pages 169–174, Hong Kong, China. Association for Computational Linguistics.

Paul Jaccard. 1908. Nouvelles recherches sur la distribution florale. *Bull. Soc. Vaud. Sci. Nat.*, 44:223–270.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1965–1972. ijcai.org.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Alan Juffs and Michael Harrington. 1995. Parsing effects in second language sentence processing: Subject and object asymmetries in wh-extraction. *Studies in second language acquisition*, 17(4):483–516.

Graham Klyne and Jeremy J. Carroll. 2004. Resource description framework (rdf): Concepts and abstract syntax. W3C Recommendation.

Anne Lauscher, Olga Majewska, Leonardo F. R. Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavaš. 2020. Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 43–49, Online. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Shuliang Liu, Xuming Hu, Chenwei Zhang, Shuang Li, Lijie Wen, and Philip S. Yu. 2022. Hiure: Hierarchical exemplar contrastive learning for unsupervised relation extraction. *ArXiv*, abs/2205.02225.

Xiaoze Liu, Junyang Wu, Tianyi Li, Lu Chen, and Yunjun Gao. 2023. Unsupervised entity alignment for temporal knowledge graphs. In *WWW*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Stuart Lloyd. 1982. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888, Lisbon, Portugal. Association for Computational Linguistics.

Oded Maimon and Lior Rokach. 2005. Data mining and knowledge discovery handbook.

Jose L Martinez-Rodriguez, Ivan López-Arévalo, and Ana B Rios-Alvarado. 2018. Openie-based approach for knowledge graph construction from text. *Expert Systems with Applications*, 113:339–355.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, Korea. Association for Computational Linguistics.

Martha McGinnis. 2002. Object asymmetries in a phase theory of syntax.

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Tom Mitchell and Edward Fredkin. 2014. Never ending language learning. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 1–1.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1955–1961. AAAI Press.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. 2013. Knowledge graph identification. In *International Semantic Web Conference*, pages 542–557. Springer.

Delip Rao, Paul McNamee, and Mark Dredze. 2013. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, multilingual information extraction and summarization*, pages 93–115. Springer.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Taffee T Tanimoto. 1958. Elementary mathematical theory of classification and prediction.

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. *arXiv preprint arXiv:2109.04404*.

Shubham Toshniwal, Haoyue Shi, Bowen Shi, Lingyu Gao, Karen Livescu, and Kevin Gimpel. 2020. A cross-task analysis of text span representations. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 166–176, Online. Association for Computational Linguistics.

Shikhar Vashishth, Prince Jain, and Partha P. Talukdar. 2018. CESI: canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1317–1327. ACM.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

A Dataset Examples

We show examples of our dataset in Figure 7.

B Guidelines for Revising Relational Phrases

B.1 RP Annotating Procedure

1. We first split all triples by relations and form 79 json files for two major paid annotators, each annotator is responsible for around 40 relations.

2. Annotators should check one relation file at a time for annotating consistency, and start the next one after the former one is finished.

3. For each relation, annotators are given: (a) The original sentences of the relation with markers indicating the head NP, tail NP and RP extracted by OpenIE. (b) The name (e.g., *composer*), and the Wikidata ID (e.g., *P86*) of the gold relation.

4. Annotators should first understand the relation by querying the Wikidata, take the relation *"composer (P86)"* as an example, annotators should first query Wikidata through the link `https://www.wikidata.org/wiki/Property:P86` to obtain the definition of the relation and skim through example relational phrases of RPs. The Figure 8 shows the Wikidata page containing the definition and examples of *"composer"*.

5. After fully comprehend the relation, annotators can start to check and revise triples in each file, the details and examples of RP revision and justification of if RP implies the relation are shown in the next two subsection (Appendix B.2, Appendix B.3).

6. After two annotators finished their part, we randomly sample 100 samples from each part and ask the annotator responsible for the other part to check. The annotators reached a consensus for approving 97% of these samples.

B.2 Guideline for checking the validity of RPs and revision

Definition of relational phrases Relational phrases are textual representations of relations between named entities (Grycner et al., 2015), we follow ReVerb (Fader et al., 2011) and require the relational phrases be continuous span in the sentence. We summarize most cases of relational phrases occurring in our dataset, and the guideline for annotating each case. The annotated RPs are shown in pink.

- **Case 1: Verb** example: *[A] married [B]*, include different tenses of verbs.

- **Case 2: Verb+preposition** example: *[A] located at [B]*.

- **Case 3: Passive voice** example: *[A] is designed by [B]*, *[A] is headquartered in [B]* the linking verb is sometimes omitted: *[A], designed by [B]*.

- **Case 4: When the tail entity is the appositive of the head entity** example: *[A] 's son is [B]*, *[A] 's son , [B]*, *[A] 's masterpiece , [B]*, the content between the appositives is usually informative and regarded as RPs.

- **Case 5: Compound predicate** A common case is that the tail NP is a part of the compound predicate of the head NP, e.g., *[A] is the civil branch of [B]*, *[A] is an agency of [B]*, *[A] is a novel by [B]*. When encountering these cases, we do not include the linking verb and article because they are not informative (e.g., *"is the"*).

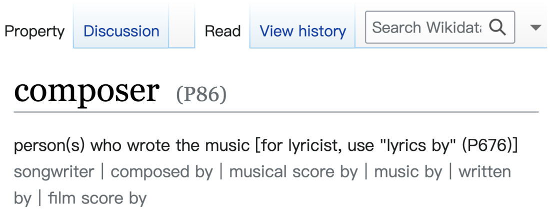
- **Case 6: Cases omit verb** Some cases omit verb, we only annotate the preposition. For example, *[yokosuka arts theatre], part of the bay square complex by [kenzou tange]*, this sentence omits the verb *"built"*.

```
{'h': {'id': 'Q533336', 'instance': ['Q18812508'], 'name': ['node', '1'], 'pos': [16, 18]}, 'r':
{'label': 'manufacturer', 'name': ['built', 'by'], 'pos': [19, 21]}, 't': {'id': 'Q66', 'instance':
['Q891723', 'Q4830453', 'Q936518', 'Q6881511', 'Q2538889', 'Q2995256'], 'name': ['boeing'], 'pos':
[21, 22]}, 'text': ['this', 'was', 'followed', 'in', 'december', 'by', 'the', 'first', 'u.s.',
'module', ',', '`', 'unity', '`', 'also', 'called', 'node', '1', ',', 'built', 'by', 'boeing', 'in',
'facilities', 'at', 'msfc', '.']}
```

```
{'h': {'id': 'Q18391244', 'instance': ['Q13406463', 'Q105543609'], 'name': ['motets'], 'pos': [18,
19]}, 'r': {'label': 'composer', 'name': ['are', 'composed', 'by'], 'pos': [19, 21]}, 't': {'id':
'Q81752', 'instance': ['Q5'], 'name': ['anton', 'bruckner'], 'pos': [21, 23]}, 'text': ['two',
'asperges', 'me', ',', 'wab', '3', 'the', 'two', 'thou', 'wilt', 'sprinkle', 'me', ',', 'wab', '3',
',', 'are', 'sacred', 'motets', 'composed', 'by', 'anton', 'bruckner', '.']}
```

```
{'h': {'id': 'Q3764815', 'instance': ['Q47461344'], 'name': ['pedda', 'bala', 'siksha'], 'pos': [0,
3]}, 'r': {'label': 'language of work or name', 'name': ['is', 'encyclopedia', 'in'], 'pos': [3,
7]}, 't': {'id': 'Q8097', 'instance': ['Q34770', 'Q1288568'], 'name': ['telugu'], 'pos': [8, 9]},
'text': ['pedda', 'bala', 'siksha', 'is', 'an', 'encyclopedia', 'in', 'the', 'telugu', ',',
'suitable', 'for', 'children', 'and', 'adults', '.']}
```

Figure 7: Examples of our dataset, "h" means head or subject NP, "r" means relation, "t" means tail or object NP. "instance" stands for the gold ontology-level clusters.



composer (P86)

person(s) who wrote the music [for lyricist, use "lyrics by" (P676)]

songwriter | composed by | musical score by | music by | written by | film score by

Figure 8: Wikidata query example.

We also provide several revision examples of wrong OpenIE triples, part of them are shown in the Table 6 below.

B.3 Guideline for justifying if RP implies the given relation

As we stated in the fourth step of the overall annotating process in the Appendix B.1, we require annotators to fully understand the meaning of the given relation. For each triple, annotators should ask themselves if the relational phrase could express the relation of the head and tail NP in the given context sentence. Note that we don't require the relation could be solely implied by the RP, for example, given the triple and its context: *"[mount elbert]* ***in*** *the [sawatch range] is the highest summit of the rocky mountains"*, it is impossible to infer the relation by the RP *"in"*, but RP is a reasonable text representation of the relation *mountain range* in this context. We found that the extracted RPs can imply the relation in most cases, we show some concrete bad cases to the annotators to help them identify the bad RPs, part of the examples are shown in the Table 7.

C Classic Metrics

Gold cluster assignment: $\mathcal{G} = \{C_1^g \dots C_K^g\}$, predicted cluster assignment: $\mathcal{P} = \{C_1^p \dots C_M^p\}$, where C_i^g and C_j^p are gold and predicted cluster respectively.

Macro Metrics

$$\begin{aligned} P_{macro}(\mathcal{G},\mathcal{P}) &= \frac{|\{p \in \mathcal{P} | \exists g \in \mathcal{G}, p \subseteq g\}|}{|\mathcal{P}|} \\ R_{macro}(\mathcal{G},\mathcal{P}) &= P_{micro}(\mathcal{P},\mathcal{G}) \end{aligned} \tag{4}$$

Micro Metrics

$$\begin{aligned} P_{micro}(\mathcal{G},\mathcal{P}) &= \frac{\sum_{g \in \mathcal{G}} \max_{p \in \mathcal{P}} |g \cap p|}{N} \\ R_{micro}(\mathcal{G},\mathcal{P}) &= P_{micro}(\mathcal{P},\mathcal{G}) \end{aligned} \tag{5}$$

Where N is the total number of different phrases that appear in $\mathcal{G}$ (or $\mathcal{P}$).

Pairwise Metrics

$$\begin{aligned} &P_{pair}(\mathcal{G},\mathcal{P}) = \\ &\frac{\sum_{p \in \mathcal{P}} |\{(u,u') \in p, \exists g \in \mathcal{G}, \forall (u,u') \in p\}|}{\sum_{p \in \mathcal{P}} C_2^{|p|}} \\ &R_{pair}(\mathcal{G},\mathcal{P}) = \\ &\frac{\sum_{p \in \mathcal{P}} |\{(u,u') \in p, \exists g \in \mathcal{G}, \forall (u,u') \in p\}|}{\sum_{g \in \mathcal{G}} C_2^{|g|}} \end{aligned} \tag{6}$$

For more details about classic metrics, please refer to the Sec. 7.2 of the CESI paper (Vashishth et al., 2018).

Bad OpenIE RP	*... [the generalitat de catalunya], the governing* ***body of*** *[catalonia], approved ...*
Revised	*... [the generalitat de catalunya], the* ***governing body of*** *[catalonia], approved ...*
Bad OpenIE RP	*... the "[althing]" , the ruling legislative* ***body of*** *[iceland]*
Revised	*... the "[althing]" , the* ***ruling legislative body of*** *[iceland]*
Bad OpenIE RP	*... [t-d center] dominion* ***centre****, designed by [ludwig mies van der rohe] ...*
Revised	*... [t-d center] dominion centre,* ***designed by*** *[ludwig mies van der rohe] ...*
Bad OpenIE RP	*[ace attorney investigations 2] ... and features* ***character designs by*** *tatsuro iwamoto and music by [noriyuki iwadare]*
Revised	*[ace attorney investigations 2] ... and features character designs by tatsuro iwamoto and* ***music by*** *[noriyuki iwadare]*
Bad OpenIE RP	*[A] is a* ***farcical musical comedy with*** *music by [walter alfred slaughter]*
Revised	*[A] is a farcical musical comedy with* ***music by*** *[walter alfred slaughter]*
Bad OpenIE RP	*[beta cygni a]* ***is a bright*** *star from the constellation [cygnus]*
Revised	*[beta cygni a] is a* ***bright star from*** *the constellation [cygnus]*
Bad OpenIE RP	*... [daya district], taichung,* ***taiwan in*** *the [chinese taipei]*
Revised	*... [daya district], taichung, taiwan* ***in*** *the [chinese taipei]*

Table 6: Examples of revising RPs in OpenIE triples.

P1001 *applies to jurisdiction*	*...the process to amend the constitution cannot be initiated in times of war or when the [belgian federal parliament]* ***is unable to freely meet in*** *[belgium]*
P84 *architect*	*[u. b. city], the* ***headquarters of*** *the [united breweries group], is a high-end commercial zone.*
P40 *child*	*he was a great-grandson of [berge sigval natanael bergeson],* ***grand-naphew of*** *[ole bergeson]*
P25 *mother*	*bart and [lisa],* ***sent out of*** *the house by [marge simpson] ...*

Table 7: Examples of bad RPs that cannot imply the given relations.

D Extension of Micro Metrics

Gold overlapping clusters: $\textit{NPC-O} = \{C_1^O \ldots C_{K_3}^O\}$, predicted clusters $\mathcal{P} = \{C_1^p \ldots C_M^p\}$.

$$P_{micro}(\textit{NPC-O}, \mathcal{P}) = \frac{\sum_{g \in \textit{NPC-O}} \max_{p \in \mathcal{P}} |g \cap p|}{\sum_{g \in \textit{NPC-O}} |g|}$$
$$R_{micro}(\textit{NPC-O}, \mathcal{P}) = P_{micro}(\mathcal{P}, \textit{NPC-O}) \tag{7}$$

We modify the denominator compared to the micro metric in (Vashishth et al., 2018).

E Standardization

Following Timkey and van Schijndel (2021), we perform standardization for phrase embeddings to remove the rogue dimensions. Denote $\boldsymbol{E}_{\mathcal{R}} \in \mathbb{R}^{N \times D}$ as the embedding matrix of all RP phrases, where N is the number of triples, and D is the dimension of contextual embedding. The standardized RP embedding matrix $\boldsymbol{E}'_{\mathcal{R}}$ is:

$$\begin{aligned} \boldsymbol{\mu} &= \frac{1}{N}\sum_{i}^{N} \boldsymbol{E}_{\mathcal{R}}[i] \\ \boldsymbol{\sigma} &= \sqrt{\frac{1}{N}\sum_{i}^{N}(\boldsymbol{E}_{\mathcal{R}}[i] - \boldsymbol{\mu})^2} \\ \boldsymbol{E}'_{\mathcal{R}}[i] &= \frac{\boldsymbol{E}_{\mathcal{R}}[i] - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \end{aligned} \tag{8}$$

We empirically find that the standardized phrase embedding is better than the original one in most cases.

F Full Result

We show the full results in Table 8 and Table 9.

	NPC-E (subject)				*NPC-E (object)*				*RPC*		
	Ma	*Mi*	*Pair*	*AVG*	*Ma*	*Mi*	*Pair*	*AVG*	*Mi*	*Pair*	*AVG*
Token Sim+SI	86.85	88.41	73.43	82.90	78.43	85.34	74.29	79.35	50.26	17.62	33.94
Random+HAC	87.97	89.90	78.10	85.32	81.24	88.11	85.97	85.11	48.47	23.49	35.98
GloVe+HAC	85.63	87.22	62.07	78.31	79.90	87.06	92.74	86.57	47.92	23.22	35.57
GloVe+HolE+HAC	88.74	89.59	62.06	80.13	80.85	88.20	92.95	87.33	27.05	8.80	17.93
GloVe+SI+HAC	88.72	89.82	62.72	80.42	80.85	88.21	92.95	87.34	27.02	8.79	17.91
CESI	88.72	89.58	62.02	80.11	82.93	88.37	89.17	86.82	27.31	9.43	18.37
VAEGMM+SI	85.63	87.51	69.44	80.86	78.55	85.93	84.26	82.91	47.86	20.34	34.10
VAEGMM+HolE	85.50	87.26	67.69	80.15	78.54	85.91	84.17	82.87	47.92	24.31	36.12
CUVA	85.58	87.45	69.00	80.68	78.56	85.95	84.33	82.95	47.94	24.32	36.13
Bert-base	90.84	92.11	77.84	86.93	86.84	90.53	83.36	86.91	55.85	53.09	54.47
Roberta-base	87.74	89.59	71.22	82.85	83.12	88.22	83.66	85.00	44.81	37.35	41.08
SpanBert-base	91.14	91.86	70.14	84.38	85.82	89.92	83.22	86.32	43.49	39.36	44.04
ERNIE2.0-base	91.19	92.50	76.35	86.68	83.15	88.89	92.29	88.11	53.94	55.38	54.66
Bert-base-triple	90.05	91.57	76.41	86.01	83.57	89.67	93.51	88.92	57.53	59.36	58.45
Roberta-base-triple	90.40	91.05	74.66	85.37	81.07	87.96	92.62	87.22	52.75	48.86	50.81
SpanBert-base-triple	91.95	91.87	73.36	85.73	85.26	89.19	83.23	85.89	48.09	44.26	46.18
ERNIE2.0-base-triple	91.14	92.34	78.16	87.21	85.71	90.12	84.96	86.93	56.58	57.98	57.28
Bert-large	90.47	91.93	78.87	87.09	83.81	89.78	93.56	89.05	53.81	46.81	50.31
Roberta-large	88.34	90.14	72.03	83.50	84.48	89.01	83.93	85.81	43.01	38.02	40.51
SpanBert-large	85.82	89.92	83.22	86.32	87.14	90.81	82.07	86.67	45.5	46.18.	45.84
ERNIE2.0-large	91.06	92.26	75.30	86.21	83.09	89.56	93.92	88.86	49.48	48.32	48.90
Bert-large-triple	90.02	92.35	93.64	92.00	87.47	90.99	84.49	87.65	56.27	60.04	58.16
Roberta-large-triple	87.93	89.35	72.93	83.40	84.24	88.57	83.56	85.46	47.37	41.41	44.39
SpanBert-large-triple	90.57	94.26	94.74	93.19	85.87	89.80	82.34	86.00	49.24	45.68	47.46
ERNIE2.0-large-triple	93.32	94.41	85.90	91.21	86.19	90.65	91.21	89.35	54.37	53.32	53.85

Table 8: Full results of NPC-E and RPC.

G Encoding Strategy Comparison

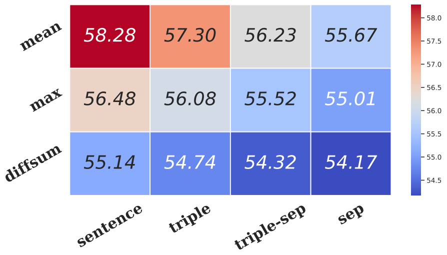


Figure 9: Overall average performance of different encoding strategies.

We compare the performance of encoding strategies averaged on all subtask metrics and PLM models in Figure 9, and the task-specific and model-specific comparison of encoding strategies are shown in Figure 10. *sentence* is the best input form in general, probably because it is easier for a PLM to encode a valid sentence and the source sentence contains more context. *sep* is the worst on the RPC because it separately encodes the RPs and NPs. However, it is comparable to *triple-sep* and *triple* on NPC-E because NPC-E requires less context. *mean* is the best strategy for phrase representation, which is consistent with the results obtained by (Toshniwal et al., 2020)[6], and *diffsum* is a bad choice for phrase canonicalization.

H Layerwise PLM Performance

We show the layerwise performance for all PLMs (base) on all subtasks in Figure 11, and find different layers perform differently on three subtasks. We empirically find that lower layers [1,2,3] perform well for NPC-E, upper layers [10,11,12] perform best for RPC and NPC-O (subj), while middle layers [3,4,5,6,7] perform relatively better on NPC-O (obj). As context-specificity increases in upper layers (Ethayarajh, 2019), these results make sense as NPC-E requires less context while RPC and NPC-O need more context.

I Triple Pretraining

We show the average performance difference after triple-level or causal subword-level pretraining in Figure 12 for different PLMs and subtasks.

[6] Toshniwal et al. (2020) shows that mean pooling is best for named entity labeling, which is a semantic task for NPs.

	NPC-O (subject)							***NPC-O (object)***						
	Ma	***Mi***	***Pair***	***Avg***	$J_{g\to p}$	$J_{p\to g}$	***Avg***	***Ma***	***Mi***	***Pair***	***Avg***	$J_{g\to p}$	$J_{p\to g}$	***Avg***
Token Sim+SI	63.28	35.86	0.29	33.14	-	-	-	66.24	51.61	3.91	40.59	-	-	-
Random+HAC	69.04	42.20	0.69	37.31	62.54	15.05	38.79	74.72	54.25	5.72	44.90	66.78	22.02	44.40
GloVe+HAC	70.37	42.80	0.84	38.00	63.00	15.07	39.04	74.08	60.25	7.73	47.35	68.73	22.17	45.45
GloVe+HolE+HAC	71.41	46.06	1.43	39.63	63.95	15.78	39.87	76.68	60.38	7.68	48.25	71.06	23.30	47.18
GloVe+SI+HAC	71.68	46.32	1.62	39.87	63.88	15.78	39.83	76.63	60.43	7.72	48.26	70.96	23.26	47.11
CESI	71.66	46.51	1.62	39.93	63.98	15.90	39.89	76.57	60.45	7.74	48.25	71.14	23.24	47.19
VAEGMM+SI	69.32	42.08	0.83	37.41	-	-	-	72.52	60.10	8.07	46.90	-	-	-
VAEGMM+HolE	69.09	41.81	0.75	37.22	-	-	-	72.48	60.09	8.07	46.88	-	-	-
CUVA	68.76	41.76	0.76	37.09	-	-	-	72.48	60.12	8.08	46.89	-	-	-
Bert-base	78.77	47.20	2.93	42.97	73.27	15.04	44.16	68.10	63.71	20.32	50.71	76.41	17.56	46.99
Roberta-base	74.18	42.67	0.78	39.21	68.06	14.42	41.24	78.22	54.17	5.81	46.07	71.24	17.04	44.14
SpanBert-base	79.00	44.58	1.25	41.61	71.54	14.78	43.16	80.58	55.29	6.00	47.29	68.52	22.18	45.35
ERNIE2.0-base	78.64	46.85	2.31	42.60	72.93	15.17	44.05	68.15	66.89	21.76	52.27	71.81	22.29	47.05
Bert-base-triple	80.71	47.58	2.84	43.71	75.02	15.36	45.19	67.87	65.96	21.51	51.78	77.56	17.42	47.49
Roberta-base-triple	78.49	46.94	1.44	42.29	73.36	15.31	44.33	82.31	60.14	8.47	50.31	76.09	17.53	46.81
SpanBert-base-triple	80.49	45.47	1.63	42.53	73.47	14.98	44.23	80.17	56.15	6.49	47.60	68.84	22.29	45.56
ERNIE2.0-base-triple	79.10	47.61	2.91	43.21	73.51	15.15	44.33	69.07	66.02	16.90	50.66	72.22	22.33	47.27
Bert-large	77.93	46.33	2.76	42.34	73.09	15.22	44.16	71.92	63.47	16.75	50.71	76.92	17.58	47.25
Roberta-large	75.16	43.63	0.86	39.88	69.96	14.73	42.35	78.54	59.72	6.80	48.35	69.07	22.09	45.58
SpanBert-large	77.40	43.99	1.48	40.96	71.08	14.72	42.90	81.05	56.38	6.33	47.92	69.00	22.36	45.68
ERNIE2.0-large	76.86	48.16	3.11	42.71	72.90	15.11	44.01	71.49	64.90	19.56	51.98	72.06	22.39	47.22
Bert-large-triple	80.19	47.92	2.11	43.41	75.46	15.58	45.52	66.59	64.34	21.86	50.93	77.63	17.96	47.79
Roberta-large-triple	77.56	46.91	1.99	42.15	71.82	14.99	43.40	76.83	59.18	7.64	47.88	74.44	17.42	45.93
SpanBert-large-triple	82.09	44.88	0.62	42.53	70.47	14.71	42.59	81.97	56.08	5.81	47.95	73.79	16.85	45.32
ERNIE2.0-large-triple	79.81	48.04	3.04	43.63	75.28	15.52	45.40	71.23	64.80	20.93	52.32	76.67	17.71	47.19

Table 9: Full results of NPC-O.

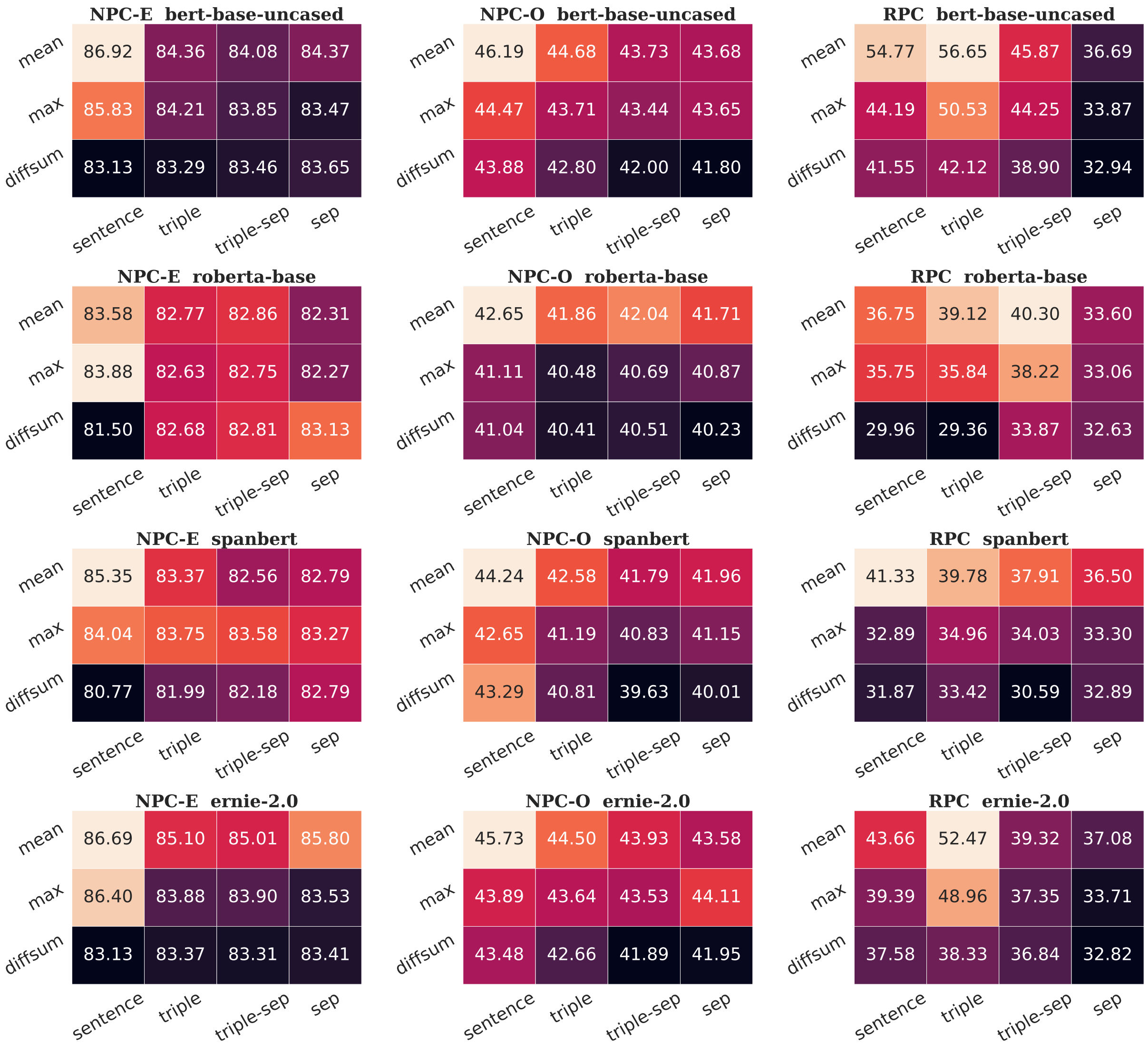


Figure 10: Comparison on different input forms and span representations for tasks and PLMs.

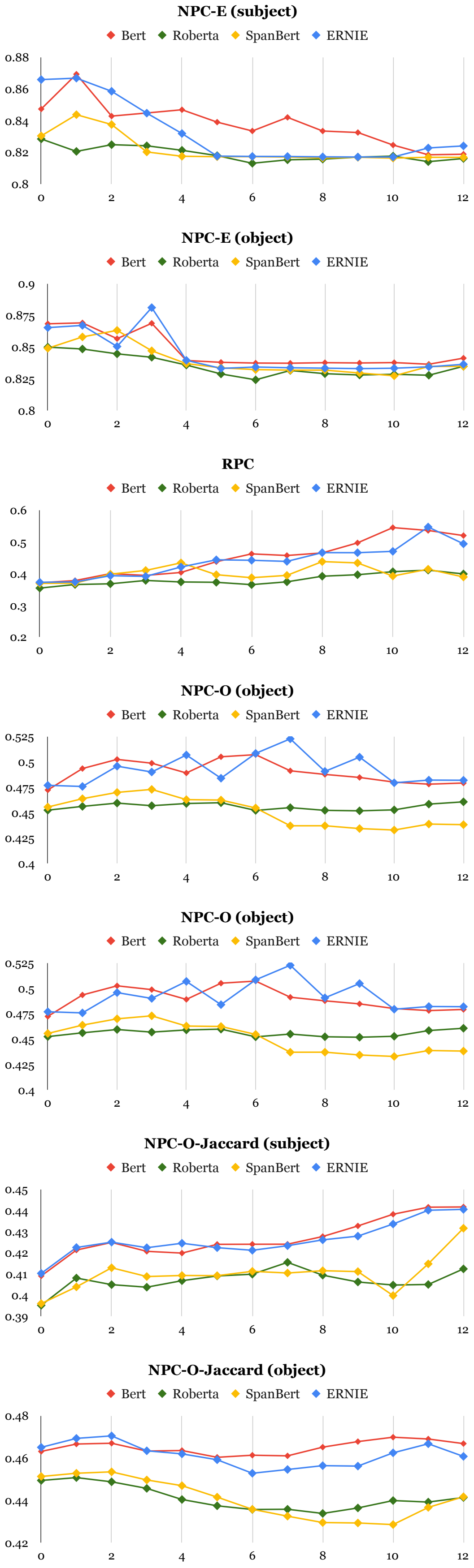
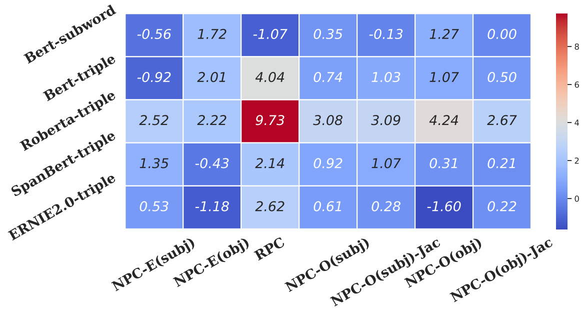


Figure 11: Layerwise performances



	NPC-E(subj)	NPC-E(obj)	RPC	NPC-O(subj)	NPC-O(subj)-Jac	NPC-O(obj)	NPC-O(obj)-Jac
Bert-subword	-0.56	1.72	-1.07	0.35	-0.13	1.27	0.00
Bert-triple	-0.92	2.01	4.04	0.74	1.03	1.07	0.50
Roberta-triple	2.52	2.22	9.73	3.08	3.09	4.24	2.67
SpanBert-triple	1.35	-0.43	2.14	0.92	1.07	0.31	0.21
ERNIE2.0-triple	0.53	-1.18	2.62	0.61	0.28	-1.60	0.22

Figure 12: Average performance difference after triple-level or causal subword-level pretraining for different PLMs and subtasks.