End-to-end Case-Based Reasoning for Commonsense Knowledge Base Completion

Zonglin Yang[1], Xinya Du[2], Erik Cambria[1] and Claire Cardie[3]

[1]Nanyang Technological Univerisity
[2]University of Texas at Dallas
[3]Cornell University

Abstract
Pretrained language models have been shown to store knowledge in their parameters and have achieved reasonable performance in commonsense knowledge base completion (CKBC) tasks. However, CKBC is knowledge-intensive and it is reported that pretrained language models' performance in knowledge-intensive tasks are limited because of their incapability of accessing and manipulating knowledge. As a result, we hypothesize that providing retrieved passages that contain relevant knowledge as additional input to the CKBC task will improve performance. In particular, we draw insights from Case-Based Reasoning (CBR) – which aims to solve a new problem by reasoning with retrieved relevant cases, and investigate the direct application of it to CKBC. On two benchmark datasets, we demonstrate through automatic and human evaluations that our **E**nd-to-end **C**ase-**B**ased **R**easoning **F**ramework (ECBRF) generates more valid knowledge than the state-of-the-art COMET model for CKBC in both the fully supervised and few-shot settings. From the perspective of CBR, our framework addresses a fundamental question on whether CBR methodology can be utilized to improve deep learning models.

Keywords
Pretrained language models, in-context learning, commonsense knowledge base completion

1. Introduction

Commonsense knowledge helps humans navigate everyday situations seamlessly [1] and is required for many intelligent scenarios [2]. To automatically enlarge the scale of commonsense knowledge base for the benefit of reducing labeling labor and expense, Knowledge Graph Completion (KGC) has become a hot research topic [3]. The general KGC task is to expand existing knowledge graphs by using well-trained classifiers—they are trained with existing annotated samples and predict whether or not there is a relationship between two *existing* entities in a knowledge graph [4].

Although KGC methods can automatically find unlabeled relationships, they are always classification or ranking tasks and are limited to *existing* entities in a knowledge graph and can't

ICCBR TMG'23: Workshop on Text Mining and Generation at ICCBR2023, July 17 – 20, 2023, Aberdeen, Scotland

zonglin.yang@ntu.edu.sg (Z. Yang); xinya.du@utdallas.edu (X. Du); cambria@ntu.edu.sg (E. Cambria); cardie@cs.cornell.edu (C. Cardie)
http://zongliny.github.io/ (Z. Yang); https://xinyadu.github.io/ (X. Du); https://sentic.net/erikcambria/ (E. Cambria); https://www.cs.cornell.edu/home/cardie/ (C. Cardie)

CEUR Workshop Proceedings (CEUR-WS.org)

Subject	Relation	Object
hardware shop	at location	mall
world map	has property	draw with grid-lines
PersonX receives its reward	wants to	keep the prize
PersonX wins the big Jackpot	wants to	get its money

Table 1
Commonsense knowledge base tuples. Examples are from ConceptNet and Atomic.

extend to new entities [3]. To extend to new entities, COMET [5] proposes to use *text generation* for exploring and discovering new entities, which is called commonsense knowledge base completion (CKBC) task, utilizing the knowledge within the pretrained language models (PLM), which has been with process in recent years [6, 7]. Specifically, COMET uses subject and relation as direct input to PLM and aims to *generate* objects, most of which are novel and unseen entities.

However, CKBC is knowledge-intensive, requiring wide-ranging and detailed knowledge; and it is reported that the ability of pretrained language models to access and precisely manipulate knowledge is limited [8]. One potential solution to this is to provide "non-parametric knowledge" through additional input. Lewis et al. [8] and Guu et al. [9], for example, have shown that by retrieving passages that contain knowledge relevant to the current task, performance can be improved. For CKBC, unfortunately, it might be especially difficult to find useful passages that contain relevant commonsense knowledge from the web due to a *reporting bias* [10] in which people rarely express the obvious (i.e., commonsense knowledge).

An example of reporting bias from Table 1 is that people rarely say "when a person wins a big Jackpot, he/she will want to get its money" because it's too obvious and meaningless to say. Therefore, instead of retrieving passages from the web, we propose that benefits can still be gained by retrieving relevant knowledge from a "case base" of existing commonsense knowledge tuples[1] and using the retrieved knowledge as non-parametric knowledge (i.e., beyond that represented in the model parameters) to augment the current CKBC input example. In addition, to prevent ECBRF from overfitting to some commonly retrieved cases, we propose $random\ mask$ as a training strategy that randomly masks the retrieved cases during training, which functions similar to dropout [11] and further improves the performance of the framework. We also analyze several variations to better understand the process.

Although past attempts suggest that similar retrieval-based methods cannot improve the performance of CKBC [12], on two benchmark datasets, we demonstrate through automatic and human evaluations that our **E**nd-to-end **C**ase-**B**ased **R**easoning **F**ramework (ECBRF) [2] generates more valid and informative knowledge than (1) the state-of-the-art COMET model [5] for CKBC which employs no case retrieval, and (2) a baseline model that employs random case retrieval – on both fully supervised and few-shot settings. We also provide an analysis on why different conclusions are reached.

In addition, our framework draws insight from Case-Based Reasoning (CBR), and also has contributions to the CBR research. CBR is a subject in classical AI that solves a new problem

[1] Initialized with tuples from training set or external data.
[2] Code available at https://github.com/ZonglinY/ECBRF_Case_Based_Reasoning_with_PLM.git

by reusing the solutions of retrieved seen similar problems stored in the case base [13]. CBR's methodology has four steps — case retrieval, reuse, revise and retain. Past years of accomplishment in deep learning (DL) have led to enthusiasm in the CBR community to apply DL in the service of CBR. However, based on the observation that many challenges remain in DL where CBR has advantages (e.g. few-shot learning), some CBR researchers [14] advocate using CBR to complement DL. However, past works on using CBR to complement DL only limit to shallow Neural Networks (NN) [15, 16, 17, 18]. The latest work even suggests that in many tasks NN itself outperforms CBR-complemented NN [18], which raises fundamental questions on whether CBR methodology is useful for DL.

Our work addresses this doubt by being the first to show a concrete implementation of the integration of the full methodology of CBR to PLM (as one typical model in DL) (we show in §6 that we simulate the third step in the methodology of CBR instead of actually implement it, since it requires huge human efforts) and show that the integration method can benefit from multiple steps in the methodology of CBR, and can lead to better performance over PLM itself in both fully supervised settings and few-shot settings on CKBC. Notably our proposed framework has a larger advantage in few-shot settings, where CBR methods typically have advantage. We also find that the generation of our framework is largely related to the retrieved case especially when they are similar, which exhibits strong case-based reasoning patterns. In addition, a detailed analysis of our framework from a CBR perspective is provided in §6.

Our contributions can be summarized as follows: (1) Drawing insights from CBR, we introduce a new end-to-end framework for CKBC task. We also propose training strategies that can better utilize the retrieved knowledge. (2) We conduct extensive experiments on the CKBC task in various settings (e.g. fully supervised and few-shot), and the results consistently demonstrate that our proposed framework achieves improvements over the state-of-the-art baseline methods. (3) From the perspective of the CBR community, whether CBR methodology can be used to improve DL models remains a fundamental research question. We address this doubt by being the first to show a concrete implementation of the integration of the full methodology of CBR to PLM, and showing that such integration can achieve better performance than single PLM. A thorough analysis of the integration from CBR perspective is also provided.

2. Related Work

Case-Based Reasoning CBR is a subject in classical AI which consists of 4 sub-processes in its methodology: *retrieve*, *reuse*, *revise* and *retain* [13]. Leake and Crandall [14] advocate using CBR to complement the challenges in deep learning (e.g., few-shot learning). Specifically, our framework is inspired by Watson [19]'s proposal that compared to CBR being described as an artificial intelligence technology , it is better to describe CBR as a methodology for problem solving, that may use any appropriate technology. Here we treat CBR as a methodology and deep learning as technology that uses CBR as the general high-level process and deep learning as components of the process.

Reasoning in NLP CBR could be seen as a type of analogical reasoning [20], and analogical reasoning belongs to inductive reasoning [21]. Inductive reasoning [22] is different from deductive reasoning [23] (both belong to logical reasoning) that the premise in inductive reasoning can not provide conclusive support to its conclusion.

Commonsense Knowledge Base Completion Here we mainly describe works that use text generation models for this task. Li et al. [24] propose models to evaluate the full knowledge tuple rather than generate new knowledge. Saito et al. [25] make an extension by proposing a joint model for the completion and generation of commonsense tuples. However, their work focuses on augmenting knowledge base completion model, rather than to increase coverage in commonsense knowledge base construction. Yao et al. [26] and Malaviya et al. [27] focus on link prediction and ranking of knowledge, which is a different task with our generative CKBC task. Sap et al. [28] use LSTM [29] to generate commonsense knowledge and Bosselut et al. [5] further leverage PLMs to generate commonsense knowledge. Gabriel et al. [30] present the task of discourse-aware commonsense inference and proposes a memory-based model to generate commonsense knowledge that is more coherent with context. Wang et al. [12] give an analysis on knowledge capacity, transferability, and induction of pretrained language models to perform generalizable commonsense inference. Da et al. [31] analyze the few-shot learning ability of pretrained language models for CKBC task. Unlike these works, we propose a model that can improve the performance of generative CKBC tasks in both fully supervised settings and few-shot settings.

Language Model Prompting First developed by the GPT series [32], retrieved data are used as augmented input to improve few-shot performance of remarkable large models. However, past research suggest that such in-context learning cannot improve the CKBC task [12], and we are the first to show how in-context learning is useful for CKBC. In addition, such large models are hard to obtain and Brown et al. [32] do not explore the finetuning performance, neither do they explore the full CBR methodology's effect on PLM. Gao et al. [33] use prompting and also incorporate demonstrations into context to improve few-shot performance. Their work, however, only focuses on classification tasks and regression tasks, which is different from the CKBC. Similar to our work, Das et al. [34] use retrieved cases as prompt to improve the performance of PLM. However, they only focus on question answering task and do not integrate the full methodology of CBR, missing important steps such as retain.

3. Task Definition

In the generative CKBC task, a knowledge data instance is represented as a tuple of subject, relation, and object: (sub, rel, obj). All sub and obj are in natural language phrases (Figure 1). rel can be used as either a special token or the corresponding natural language phrases [5]. Here we use rel as natural language phrases. The task is that given a pair of sub and rel, the goal is to *generate* the corresponding obj.

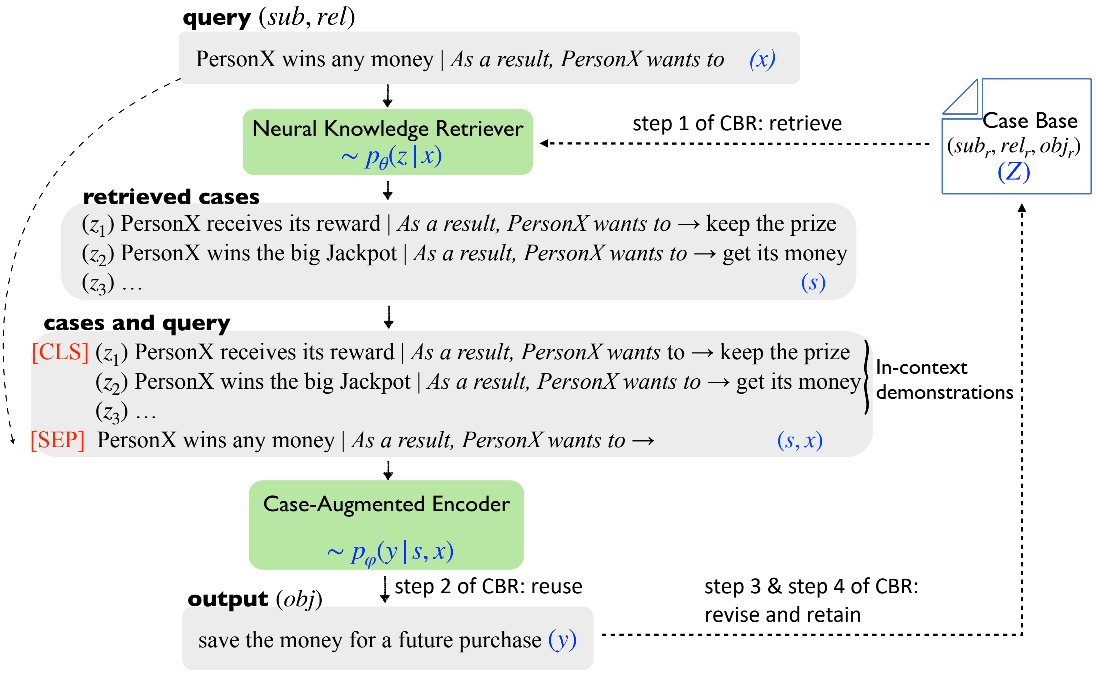

query (sub, rel)
PersonX wins any money | As a result, PersonX wants to (x)
Neural Knowledge Retriever
~ pθ(z | x)
step 1 of CBR: retrieve
Case Base
(subr, relr, objr)
(Z)
retrieved cases
(z1) PersonX receives its reward | As a result, PersonX wants to → keep the prize
(z2) PersonX wins the big Jackpot | As a result, PersonX wants to → get its money
(z3) …
(s)
cases and query
[CLS] (z1) PersonX receives its reward | As a result, PersonX wants to → keep the prize
(z2) PersonX wins the big Jackpot | As a result, PersonX wants to → get its money
(z3) …
[SEP] PersonX wins any money | As a result, PersonX wants to →
(s, x)
In-context demonstrations
Case-Augmented Encoder
~ pφ(y | s, x)
step 2 of CBR: reuse
output (obj)
save the money for a future purchase (y)
step 3 & step 4 of CBR: revise and retain

Figure 1: Our end-to-end case-based reasoning framework (ECBRF) for commonsense knowledge base completion. It involves all four steps of the CBR methodology (*retrieve*, *reuse*, *revise* and *retain*).

4. Methodology

We start by formalizing our framework as a retrieve-then-predict generative process. Then in §4.2, we describe our ECBRF's modules for the generative process in detail. Finally, we present a hybrid training strategy for better regularization.

Figure 1 describes our method. In the figure, "query" stands for a sub and rel pair which is used as input to ECBRF to generate obj. "Case Base" is initialized with knowledge triples from the training set. "Cases" means the retrieved knowledge triples from the "Case Base". "In-context demonstrations" stand for the retrieved cases that are used for input augmentation (concatenate with the query). The subject, relation, and object of the retrieved cases are sub-scripted with "r" (e.g., sub_r).

4.1. ECBRF's Generative Process

ECBRF takes x as input and learns a distribution $p(y|x)$ over possible outputs y. Here x consists of sub and rel, and y consists of obj. More specifically, ECBRF decomposes $p(y|x)$ into two steps: *retrieve* and *predict*. Given an input x, we first retrieve similar cases z_1, z_2, ... (each case z_i consists of sub_r, rel_r and obj_r) from case base Z, while $(x, y) \notin z_i$. We model this as a sample from the distribution $p(z|x)$.

Then we use z_i in a number of m to compose a supporting set s (for each query, one supporting set is used for input augmentation). Specifically,

$$\hat{p}(s|x) = \sum_{z_i \in \text{top-m}(p(.|x))} p(z_i|x) \quad ((x,y) \notin z) \tag{1}$$

$$p(s_i|x) = \frac{\exp \hat{p}(s_i|x)}{\sum_j \exp \hat{p}(s_j|x)} \tag{2}$$

Then we condition on both the supporting set s and the query x to generate the output y, modeled as $p(y|s,x)$. To obtain the overall likelihood of generating y, we treat s as a latent variable and marginalize s via a top-k approximation, yielding:

$$p(y|x) = \sum_{s \in \text{top-k}(p_\theta(.|x))} p_\theta(s|x) p_\varphi(y|s,x) \tag{3}$$

4.2. Model Architecture

We now detail two key components – the *neural knowledge retriever*, which models $p_\theta(z|x)$; and *case-augmented encoder*, which models $p_\varphi(y|s,x)$.

Neural Knowledge Retriever The retriever uses max inner product search (MIPS) to retrieve z. Specifically, the retriever is defined using a dense inner product model:

$$p_\theta(z|x) = \frac{\exp f(x,z)}{\sum_{z'} \exp f(x,z')} \tag{4}$$

$$f(x,z) = Embed_{query}(x)^T Embed_{case}(z) \tag{5}$$

where $Embed_{query}$ is an embedding function that maps sub and rel in the query input to a $d-$dimensional vector, and $Embed_{case}$ is an embedding function that maps sub_r, rel_r and obj_r in the knowledge tuples in memory store to a $d-$dimensional vector. The relevance score $f(x,z)$ between x and z is defined as the inner product of the vector embeddings. The retrieval distribution is the softmax over relevance scores between top-k retrieved cases and current query input.

We implement the embedding functions $Embed_{query}$ and $Embed_{case}$ using two DPR-based models [35]. The input format for query x is the concatenation of subject and relation: [CLS] sub [SEP] rel [SEP]; And the input format for case z is the concatenation of the subject, relation, and object: [CLS] sub_r [SEP] rel_r obj_r [SEP].

Case-Augmented Encoder Given an input x and a supporting set s, the case-augmented encoder defines:

$$p_\varphi(y|s,x) = \prod_i^N p_\varphi(y_i|x,s,y_{1:i-1}) \tag{6}$$

We use BART [36] and GPT-2 [7] as the base model for case-augmented encoder.

We also add prompts which we find is helpful. The input format (with prompts and in-context demonstrations) for case-augmented encoder is:

ConceptNet	5-shot	20-shot	40-shot	160-shot	320-shot	Full (100%)
COMET (GPT2)*	374.32 / 0.33	339.19 / 0.58	282.76 / 0.38	58.59 / 1.44	41.23 / 2.29	**13.04** / **3.37**
ECBRF (GPT2)	**284.67** / **0.51**	**220.07** / **0.59**	**102.46** / **1.63**	**52.10** / **1.61**	**38.63** / **2.59**	13.60 / 2.83
COMET (BART)*	14.26 / 1.15	11.31 / 1.64	9.48 / **3.70**	6.60 / 6.70	5.44 / 9.57	2.90 / 20.19
ECBRF (BART)	**12.85** / 1.20	**9.53** / 2.11	**8.70** / 3.17	6.19 / 6.20	5.22 / 9.31	2.93 / 18.55
w/ *random mask*	13.68 / **1.40**	9.82 / 2.02	8.98 / 2.96	**6.14** / **6.88**	**5.05** / **10.71**	**2.86** / 19.97
w/o *reverse demonstration*	13.02 / 1.20	9.60 / 1.84	8.74 / 2.95	6.21 / 6.13	5.06 / 10.12	2.92 / 19.80
w/ rand retrieval	13.23 / 1.00	10.15 / **2.61**	9.24 / 3.45	6.42 / 6.21	5.40 / 8.93	2.91 / **20.29**

ATOMIC	5-shot	20-shot	40-shot	160-shot	320-shot	Full (100%)
COMET (GPT2)*	753.93 / 2.11	512.11 / **3.44**	409.30 / **2.32**	209.78 / 2.73	165.28 / 2.68	67.95 / 4.00
ECBRF (GPT2)	**653.90** / **2.30**	**416.16** / 2.89	**319.12** / 2.26	**182.43** / **2.92**	**163.56** / **2.86**	**67.35** / **4.05**
COMET (BART)*	19.72 / **5.76**	16.83 / **5.38**	13.58 / 9.20	**14.30** / 11.56	**14.45** / 12.67	6.98 / **19.34**
ECBRF (BART)	18.17 / 5.16	14.73 / 3.85	13.05 / **10.13**	15.19 / 10.13	14.61 / 11.66	**6.95** / 19.06
w/ *random mask*	18.50 / 5.29	14.93 / 4.04	**13.01** / 7.47	14.52 / **12.42**	14.53 / 12.64	6.96 / 19.22
w/o *reverse demonstration*	**18.13** / 5.58	14.70 / 3.89	12.99 / 5.14	15.15 / 10.18	15.65 / 10.95	**6.95** / 19.24
w/ rand retrieval	18.16 / 4.33	**14.66** / 3.64	12.96 / 4.70	14.60 / 11.76	14.80 / **12.91**	6.96 / 19.02

Table 2
Perplexity (↓) / BLEU (↑) scores on ConceptNet (upper) and ATOMIC (down). The best scores for each setting are boldfaced. *: baseline models (our own implementation).

Here are some similar cases to infer from: z_0 z_1 *...* z_{m-1} *From the similar cases we can infer that: [SEP]* sub rel

Zhao et al. [37] show that pre-trained language model has "Recency Bias", which is the tendency to repeat answers that appear in the last in-context demonstration in classification tasks. We analyse this strategy for the generative CKBC task (we call it "*reverse demonstration*") that the most similar case from the retriever is placed as the last demonstration, the second most similar case in the second last demonstration, and so on.

4.3. Training Method

Since the purpose of in-context demonstration is only to provide ancillary information, the model should be able to predict *obj* w/ or w/o it. Therefore here we design a specific training strategy for ECBRF – during training, we *randomly mask* out in-context demonstrations and only keep the (sub, rel) query for some training examples with probability p_{mask}. It is designed to function similarly to dropout to prevent overly relying on retrieved cases.

5. Experiments & Analysis

In this section, we introduce the experiment datasets and evaluation details, as well as experiment setups and the experiment results, measured with automatic and human evaluations.

5.1. Datasets and statistics

We evaluate ECBRF using two automatic commonsense knowledge base completion benchmarks — ConceptNet [38] and ATOMIC [28]. In total, ConceptNet contains 101,800 tuples and ATOMIC

	20-shot (BART)	160-shot (BART)	Full (BART)
ConceptNet			
COMET	0.37 / 1.76 / 1.74	0.42 / 2.86 / 2.69	0.47 / 3.87 / 3.49
ECBRF	**0.63 / 2.47 / 2.38**	**0.58 / 3.26 / 3.13**	**0.53 / 3.95 / 3.58**
ATOMIC			
COMET	0.43 / 2.21 / 2.27	0.44 / 3.05 / 3.05	0.47 / 3.59 / 3.36
ECBRF	**0.57 / 2.44 / 2.58**	**0.56 / 3.22 / 3.17**	**0.53 / 3.64 / 3.43**

Table 3
Human evaluation results using *preference score*, *validness*, and *informativeness.*

sub: PersonX spends ___ working; rel: As a result, others feel
Ground truth: ['happy', 'happy to have x in their life']
COMET's generation: happy (BLEU: 31.62) ECBRF's generation: satisfied with personx's work (BLEU: 0.00)

Table 4
An example to show that BLEU is not a perfect metric for CKBC.

	20-shot (BART)	160-shot (BART)	Full (BART)
ConceptNet			
COMET	98.00 / 13.46 / **58.99**	92.22 / 14.83 / 61.83	57.83 / **5.57** / 71.29
ECBRF	96.64 / **21.24** / 51.10	93.27 / **16.09** / **65.83**	**59.62** / 4.84 / **73.08**
ATOMIC			
COMET	100.0 / 58.18 / 15.83	100.0 / 30.92 / 29.11	100.0 / **9.21** / **17.93**
ECBRF	100.0 / **82.16** / **22.79**	100.0 / **34.96** / **29.70**	100.0 / 6.49 / 15.37

Table 5
Novelty evaluation results using %N/T-sro, %N/T-o, and %N/U-o.

contains 877,077 tuples. We use the same data split as COMET [5] did. In ATOMIC, around 17% of the labeled knowledge tuples use "None" as the object. As a result, models can easily get high performance by always generating "None". To better evaluate, we don't use knowledge tuples with the object being "None" for both training and evaluation. Apart from using the entire train set for training, we also conduct experiments in the few-shot settings — where the model is only trained with 5 to 320 knowledge tuples[3].

5.2. Evaluation Details

For automatic evaluation metrics, following Bosselut et al. [5] we use BLEU-2, perplexity, and *novelty* metrics (including %N/T-sro, %N/T-o, and %N/U-o). Specifically for *novelty* metrics, we report the proportion of all generated tuples that are novel tuple (%N/T-sro) (here novel means unseen in train set), have a novel obj (%N/T-o), and the proportion of the set of unique obj in all generated objects (%N/U-o).

[3] Note that for the few-shot settings, our ECBRF's case base is also initialized with 5 to 320 tuples

In addition to automatic evaluation, we also perform human evaluation, including *validness*, *informativeness*, and *preference score*. For *validness* and *informativeness*, following Gabriel et al. [30], the score is based on a 5-point Likert scale (with 5 points the highest score). For *validness*, following Gabriel et al. [30], we judge the validness of the generated new knowledge by the likelihood of inferences based on a 5-point Likert scale (with 5 points the highest score). Specifically, obviously true (5), generally true (4), plausible (3), neutral or unclear or basically a repetition (sub-sentence) of the query (2), and doesn’t make sense (1). For *informativeness*, the rating standard is also based on a 5-point Likert scale. Specifically, rich in relevant details (5), has relevant details (4), it seems some details are provided (3), basically a repetition (sub-sentence) of the query (2), unfinished generation (1). For *preference score*, We ask the human raters to *compare* the generations between ECBRF and COMET. Specifically, a valid generation with more information provided will be assigned 1.0 point, and a generation that is not valid or with less information will be assigned 0.0 instead. However, if the two generations perform comparably, both generations will be assigned 0.5 points.

Following Bosselut et al. [5], for each experiment and for each model, we sample 100 generations for human evaluation. Each generation is rated by three graduate students. During the evaluation the order of the two generations to be compared are randomized for each selection, therefore human raters have no clue on which choice is associated with which model.

5.3. Experimental Setup

Baselines We use COMET [5] as our baseline. COMET is originally implemented with GPT [39], a pretrained language model as the base model and uses subject and relation as direct input and uses the generation result as object. Here we compare two versions of COMET, one is GPT-2 [7] based and another is BART [36] based. Both GPT-2 and BART are more powerful pretrained language models than GPT.

5.4. Main Results

Automatic evaluation results on BLEU-2 and perplexity are shown in Table 2. We present results with human evaluations in Table 3. Automatic evaluation of *novelty* is shown in Table 5. Table 2 shows that *random mask* is constantly helpful for ECBRF when the train set is equal or larger than 160. Therefore we adopt *random mask* for ECBRF when the train set is equal to or larger than 160 in Table 3 and Table 5.

Table 2 shows that regardless of the selection of base models, ECBRF consistently outperforms the COMET baseline in almost all perplexity measures and most BLEU measures. We argue that BLEU is not a perfect metric for CKBC [40], since each sub and rel pair can lead to more than one feasible obj, and BLEU can only refer to a limited set of ground truth obj. Even if a model generates a reasonable obj, it may yield a low BLEU, because the generated obj is not in the ground truth set.

Table 4 shows one typical example from ATOMIC that shows although ECBRF’s generations are reasonable, they only receive low BLEU scores. Table 3 shows that ECBRF consistently outperforms COMET in *preference score*, *validness*, and *informativeness* in human evaluation, especially in few-shot setting. In practice, we observe that in few-shot setting, BART without

sub: PersonX wants to play with PersonY rel: Before, this person needed
One retrieved case by ECBRF: PersonX plays tennis with PersonY's friend, Before, this person needed, get a tennis racket
COMET's generation: to have a game ECBRF's generation: to find a tennis court

Table 6
An example to show how the retrieved cases influence the generated obj.

retrieval tends to repeat the query during generation, while retrieved cases seem to be able to provide knowledge and guidance to generate more proper obj.

Table 5 shows that the generated obj of ECBRF are generally *novel* especially in few-shot settings. We empirically attribute the lower novelty of ECBRF in full train set to that ECBRF sometimes tends to copy proper retrieved obj as generation. The reason %N/T-sro score is always 100.0 in ATOMIC is that the (sub, rel) pairs in ATOMIC's train set and test set do not overlap.

We attribute the different conclusions reached on whether in-context demonstrations (ICD) with finetuning can be beneficial to CKBC [12] to that (1) ICD is more useful in few-shot settings, so that investigation on full train set setting might not discover this advantage; (2) human evaluation is the most precise metric for the task while BLEU is not so only evaluating with automatic metrics could not be precise; (3) ECBRF uses *random mask*, which is empirically found to be helpful in performance when using a large train set.

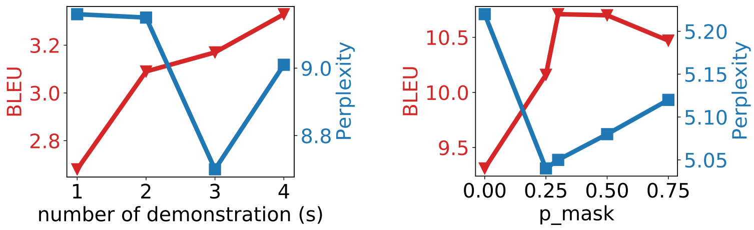
5.5. Ablation Study of ECBRF

In Table 2, we show some ablation studies of ECBRF. "w/ rand mask" stands for the ECBRF model using random mask ($p_{mask} = 30\%$); "w/o reverse demonstrations" stands for the ECBRF model without using reverse demonstrations; and "w/ rand retrieval" represents an ECBRF model that uses randomly searched cases instead of MIPS search.

Both tables for automatic evaluation show that using *random mask* can generally lead to better performance for ECBRF in both perplexity and BLEU when the training set is larger, while lead to worse performance when the number of training set is less than 160. Our interpretation is that, when the train set is very small, the model can benefit from over-relying to the retrieved cases; while when the train set is large, PLM can still benefit from the retrieved cases but the over-relying could be harmful.

The tables also show that "reverse demonstration" only leads to comparable performance, which might indicate that the order of retrieved cases does not make a difference in a generative task (as CKBC).

From the tables, we also observe that ECBRF with MIPS retrieval consistently leads to better performance than ECBRF with random retrieval in terms of perplexity in ConceptNet experiments, while performs comparably with ECBRF in ATOMIC experiments. Notice that the required generation in ConceptNet is usually shorter and more similar compared to ATOMIC.



(a) effect of # cases (b) effect of p_{mask}

Figure 2: ECBRF's performance (perplexity (↓) / BLEU (↑)) with regard to different numbers of demonstrations and different p_{mask}. Experiments on the left figure use ConceptNet 40-shot train set, and the right figure use ConceptNet 320-shot train set (since *random mask* is only helpful on large train set).

Therefore our interpretation is that, only when the retrieved cases are enough similar to the input query and its designed golden generation, can the retrieved cases significantly benefit the generation process (towards golden generation).

5.6. Qualitative Analysis on How Retrieved Cases Influence obj Generation

Table 6 shows one example of the generation of ECBRF and COMET. It shows that ECBRF's generation is related to the retrieved case, exhibiting the case-based reasoning ability of *reusing* the *retrieved* old experience to solve new problems.

6. Further Analysis from Perspective of CBR

CBR methodology contains 4 sub-processes, which are *retrieve*, *reuse*, *revise* and *retain*. More specifically, when given a new problem, the method first *retrieves* the most similar cases, then *reuses* the information in that case to solve the new problem by proposing a new solution, then *revises* the proposed solution according to the feedback of adopting it in real application scenarios (*revise* step usually involves human's effort), and finally select high quality revised solutions together with their problems as new cases to *retain* to case base.

We provide an analysis of how the high-level methodology of CBR (*retrieve*, *reuse*, *revise* and *retain*) shapes the design and how the selection details of CBR-related components improve the performance of our end-to-end DL framework.

Step 1: Retrieve *Retrieve* is an important step since the effectiveness of a CBR system largely relies on its ability to retrieve useful previous cases [41]. Here we use *neural knowledge retriever* (DPR) for retrieving the most similar cases. Table 2 shows the results of ECBRF using MIPS retrieval and random retrieval. As illustrated in §5.5, from the experimental results we hypothesize that ECBRF tends to make generations that are similar to the retrieved cases. This hypothesis is consistent with insights from CBR that the $retrieve$ step is essential for guiding the $reuse$ step. However, the difference lies in that CBR insights rely on $retrieve$ step

	Perplexity	BLEU
ECBRF	8.70	3.17
w/ only obj_r	8.90	3.26
w/o prompt	9.01	3.44
w/ larger case base	**7.90**	**3.69**

Table 7
Ablation Study: effect of sub_r, prompt, and the retain step (perplexity (↓) / BLEU (↑)). Results of this table use ConceptNet 40-shot train set.

more (with irrelevant retrieval it would be particularly hard for $reuse$), while PLM seems to be able to benefit from even random retrieval.

Step 2: Reuse Here we use *case-augmented encoder* to automatically *reuse* the retrieved cases.

Figure 2a shows the effects of number of retrieved cases. We observe that when *case-augmented encoder* uses 3 cases, it reaches the best perplexity, and nearly the best BLEU performance.

Figure 2b shows the effects of p_{mask}. Only when p_{mask} is 1.0, in-context demonstrations are not used at test time, which makes the model the same as COMET. As we gradually increase p_{mask}, perplexity keeps improving and BLEU-2 reaches the global maximum when p_{mask} is 0.3. It is also interesting to see empirically how the *case-augmented encoder* gradually learns to *reuse* the retrieved cases to increase the performance of the deep learning model as we gradually decrease p_{mask}.

In CBR, the *reuse* of the retrieved case's solution contains two steps: (a) find the difference between the past and the current queries and (b) adapt the retrieved solution to the current query [13]. So it is important to know the difference between the past queries and the current input query for better adaptation. Table 7 shows the comparison between the result of only using obj_r (the retrieved cases' object phrases) as in-context demonstrations and the result of ECBRF (uses both sub_r, rel_r and obj_r), and we observe that ECBRF performs better in perplexity. but a little bit worse in BLEU. This result indicates that deep learning based *case-augmented encoder* is possible to automatically learn and reason from the difference between the past queries and the current input query for *reuse*. We leave further investigations on whether PLMs can learn to compare the difference between the past queries and the current input query as an open research question.

We use prompts to indicate the role of retrieved cases and current query in input. Table 7 shows that *case-augmented encoder* with the prompt performs better in perplexity while a bit worse on BLEU, indicating that usage of a prompt is possible to help the model better *reuse* the retrieved cases.

Step 3 & 4: Revise and Retain Since *revise* typically involves human efforts, here we simulate *revise* and *retain* and see their effect on our framework. The result of *revise* and *retain* is a larger case base with more high quality data, and the parameters of the model for *reuse* are not necessarily updated according to the new data. Here we simulate the effect of *revise* and *retain* by first training ECBRF in a low-resource experiment (with a small case base), then at test time we expand the case base to the full train set. Table 7 shows that, at test time, ECBRF with access to a larger case base substantially outperforms ECBRF (with access to only a small case base), although the parameters have not been updated with the new data. This result demonstrates that our framework can benefit from CBR's methodology as *revise* and *retain.*

7. Conclusion

Drawing insights from CBR, we propose an end-to-end framework for the CKBC task. We demonstrate through automatic and human evaluations that our framework generates more valid knowledge than the state-of-the-art COMET model in both the fully supervised and few-shot settings. From the perspective of CBR, our framework addresses a fundamental question on whether CBR methodology can be utilized to improve deep learning models.

8. Future Works and Challenges

In general, we hope this work could provide some insights to bridge the two research areas, classic AI (Case-Based Reasoning) and deep learning based NLP methods together, and therefore to advance the research of both fields from each other's research developments.

From the aspect of NLP methods, for example, new prompting methods could be further developed based on insights from CBR research; The concept of *revise* and *retain* from CBR could be paid more attention to investigate their interaction with in-context demonstrations (prompting).

From the aspect of CBR, this work provides a tentative answer to the two long-remaining challenges — (1) whether CBR can be used to complement DL [14], given that the latest work even suggests that in many tasks NN itself outperforms CBR-complemented NN [18]; (2) the adaptation (*reuse* step) of previous cases to the current case is a very challenging problem, so that in many fields the CBR methodology is used only as a retriever [42]. How to further answer these two questions could be a challenging research topic.

9. Limitations

From the perspective of CBR, we have shown through experiments that our framework can perform *retrieve* and *reuse* steps, and can benefit from *revise* and *retain* steps. But the *revise* step in CBR typically involves human efforts, and this paper does not focus on addressing this challenge. As a result, our framework might still need manual efforts to benefit from *revise* and *retain.*

However, human efforts could be more efficiently utilized for *revise* than writing new data from scratch. Since comparing with requesting the workers to write the knowledge from scratch, *revising* the existing generations of ECBRF could be much faster.

References

[1] I. Apperly, Mindreaders: the cognitive basis of" theory of mind", Psychology Press, 2010.

[2] E. Davis, G. Marcus, Commonsense reasoning and commonsense knowledge in artificial intelligence, Communications of the ACM (2015).

[3] S. Ji, S. Pan, E. Cambria, P. Marttinen, P. S. Yu, A survey on knowledge graphs: Representation, acquisition, and applications, IEEE Trans. Neural Networks Learn. Syst. 33 (2022) 494–514. URL: https://doi.org/10.1109/TNNLS.2021.3070843. doi:10.1109/TNNLS.2021.3070843.

[4] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, IEEE Transactions on Knowledge and Data Engineering 29 (2017) 2724–2743.

[5] A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, Y. Choi, COMET: Commonsense transformers for automatic knowledge graph construction, in: ACL 2019, Association for Computational Linguistics, 2019.

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, 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), 2019, pp. 4171–4186.

[7] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, OpenAI blog (2019).

[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.

[9] K. Guu, K. Lee, Z. Tung, P. Pasupat, M.-W. Chang, Realm: Retrieval-augmented language model pre-training, ICML (2020).

[10] J. Gordon, B. Van Durme, Reporting bias and knowledge acquisition, in: Proceedings of the 2013 workshop on Automated knowledge base construction, 2013, pp. 25–30.

[11] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958. URL: https://dl.acm.org/doi/10.5555/2627435.2670313. doi:10.5555/2627435.2670313.

[12] P. Wang, F. Ilievski, M. Chen, X. Ren, Do language models perform generalizable commonsense inference?, in: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Association for Computational Linguistics, Online, 2021, pp. 3681–3688. URL: https://aclanthology.org/2021.findings-acl.322. doi:10.18653/v1/2021.findings-acl.322.

[13] A. Aamodt, E. Plaza, Case-based reasoning:foundational issues,methodological variations,and system approaches, AI communications (1994).

[14] D. Leake, D. Crandall, On bringing case-based reasoning methodology to deep learning, in: International Conference on Case-Based Reasoning, Springer, 2020.

[15] C. Liao, A. Liu, Y. Chao, A machine learning approach to case adaptation, in: First IEEE International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2018, Laguna Hills, CA, USA, September 26-28, 2018, IEEE Computer Society, 2018, pp. 106–109. URL: https://doi.org/10.1109/AIKE.2018.00023. doi:10.1109/AIKE.2018.00023.

[16] D. Leake, X. Ye, D. J. Crandall, Supporting case-based reasoning with neural networks: An illustration for case adaptation, in: A. Martin, K. Hinkelmann, H. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2021 Spring Symposium on

Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021), Stanford University, Palo Alto, California, USA, March 22-24, 2021, volume 2846 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021. URL: http://ceur-ws.org/Vol-2846/paper1.pdf.

[17] X. Ye, D. Leake, V. Jalali, D. J. Crandall, Learning adaptations for case-based classification: A neural network approach, in: A. A. Sánchez-Ruiz, M. W. Floyd (Eds.), Case-Based Reasoning Research and Development - 29th International Conference, ICCBR 2021, Salamanca, Spain, September 13-16, 2021, Proceedings, volume 12877 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 279–293. URL: https://doi.org/10.1007/978-3-030-86957-1_19. doi:10.1007/978-3-030-86957-1_19.

[18] X. Ye, D. Leake, D. J. Crandall, Case adaptation with neural networks: Capabilities and limitations, in: M. T. Keane, N. Wiratunga (Eds.), Case-Based Reasoning Research and Development - 30th International Conference, ICCBR 2022, Nancy, France, September 12-15, 2022, Proceedings, volume 13405 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 143–158. URL: https://doi.org/10.1007/978-3-031-14923-8_10. doi:10.1007/978-3-031-14923-8_10.

[19] I. Watson, Case-based reasoning is a methodology not a technology, in: Research and Development in Expert Systems XV, Springer, 1999.

[20] J. L. Kolodner, Educational implications of analogy: A view from case-based reasoning., American psychologist 52 (1997) 57.

[21] M. H. Salmon, Introduction to logic and critical thinking (1989).

[22] Z. Yang, L. Dong, X. Du, H. Cheng, E. Cambria, X. Liu, J. Gao, F. Wei, Language models as inductive reasoners, CoRR abs/2212.10923 (2022). URL: https://doi.org/10.48550/arXiv.2212.10923. doi:10.48550/arXiv.2212.10923. arXiv:2212.10923.

[23] P. Clark, O. Tafjord, K. Richardson, Transformers as soft reasoners over language, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org, 2020, pp. 3882–3890. URL: https://doi.org/10.24963/ijcai.2020/537. doi:10.24963/ijcai.2020/537.

[24] X. Li, A. Taheri, L. Tu, K. Gimpel, Commonsense knowledge base completion, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 1445–1455. URL: https://aclanthology.org/P16-1137. doi:10.18653/v1/P16-1137.

[25] I. Saito, K. Nishida, H. Asano, J. Tomita, Commonsense knowledge base completion and generation, in: Proceedings of the 22nd Conference on Computational Natural Language Learning, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 141–150. URL: https://aclanthology.org/K18-1014. doi:10.18653/v1/K18-1014.

[26] L. Yao, C. Mao, Y. Luo, KG-BERT: BERT for knowledge graph completion, CoRR abs/1909.03193 (2019). URL: http://arxiv.org/abs/1909.03193. arXiv:1909.03193.

[27] C. Malaviya, C. Bhagavatula, A. Bosselut, Y. Choi, Commonsense knowledge base completion with structural and semantic context, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 2925–2933. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5684.

[28] M. Sap, R. Le Bras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N. A. Smith,

Y. Choi, Atomic: An atlas of machine commonsense for if-then reasoning, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 3027–3035.

[29] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation (1997).

[30] S. Gabriel, C. Bhagavatula, V. Shwartz, R. Le Bras, M. Forbes, Y. Choi, Paragraph-level commonsense transformers with recurrent memory, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 12857–12865.

[31] J. Da, R. L. Bras, X. Lu, Y. Choi, A. Bosselut, Understanding few-shot commonsense knowledge models, AKBC (2021).

[32] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[33] T. Gao, A. Fisch, D. Chen, Making pre-trained language models better few-shot learners, in: ACL 2021, Association for Computational Linguistics, Online, 2021, pp. 3816–3830. URL: https://aclanthology.org/2021.acl-long.295. doi:10.18653/v1/2021.acl-long.295.

[34] R. Das, M. Zaheer, D. Thai, A. Godbole, E. Perez, J. Y. Lee, L. Tan, L. Polymenakos, A. McCallum, Case-based reasoning for natural language queries over knowledge bases, in: EMNLP 2021, 2021, pp. 9594–9611.

[35] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, Dense passage retrieval for open-domain question answering, in: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 6769–6781. URL: https://aclanthology.org/2020.emnlp-main.550. doi:10.18653/v1/2020.emnlp-main.550.

[36] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, in: ACL 2020, Association for Computational Linguistics, Online, 2020, pp. 7871–7880. doi:10.18653/v1/2020.acl-main.703.

[37] Z. Zhao, E. Wallace, S. Feng, D. Klein, S. Singh, Calibrate before use: Improving few-shot performance of language models, in: International Conference on Machine Learning, PMLR, 2021, pp. 12697–12706.

[38] R. Speer, J. Chin, C. Havasi, Conceptnet 5.5: An open multilingual graph of general knowledge, in: Thirty-first AAAI conference on artificial intelligence, 2017.

[39] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding by generative pre-training (2018).

[40] A. B. Sai, A. K. Mohankumar, M. M. Khapra, A survey of evaluation metrics used for NLG systems, ACM Comput. Surv. 55 (2023) 26:1–26:39. URL: https://doi.org/10.1145/3485766. doi:10.1145/3485766.

[41] A. R. Montazemi, K. M. Gupta, A framework for retrieval in cbr systems, Annals of operations research (1997).

[42] N. Choudhury, S. A. Begum, A survey on case-based reasoning in medicine, International Journal of Advanced Computer Science and Applications 7 (2016).