Language models are open knowledge graphs

Chenguang Wang, Xiao Liu, Dawn Song (2020)

Under Review

https://openreview.net/forum?id=aRTRjVPkm-

ICRL 2021 Reject

Review1: Interesting unsupervised IE approach but not 100% convinced

Review2: An interesting question but the methodology needs improvement.

Review3: Interesting approach but unconvincing evaluation and a lack of details

Review4: Conceptually interesting paper but with some limitations and missing related work that may need to be addressed

Github X

0. Abstract

1. KG의 Factual knowledge (사실에 관한 지식)은 reasoning이나 understanding에 사용, 하지만 유명한 KG(Wikidata, NELL)은 supervised 혹은 semi-supervised manner로 만들어 졌다. (사람 손 탐)
2. Deep LM은 automatically acquired knowledge from large-scale corpora via pre-training, LM이 저장한 knowledge들은 downstream NLP task(qa, writing, summary etc)를 향상 시켰다.

3. 이 논문은 사람의 감시 없이, 어떻게 pre-trained LM에서 KG를 만들어 내는지에 대한 논문이다.

   : KG are constructed with a single forward pass of LM

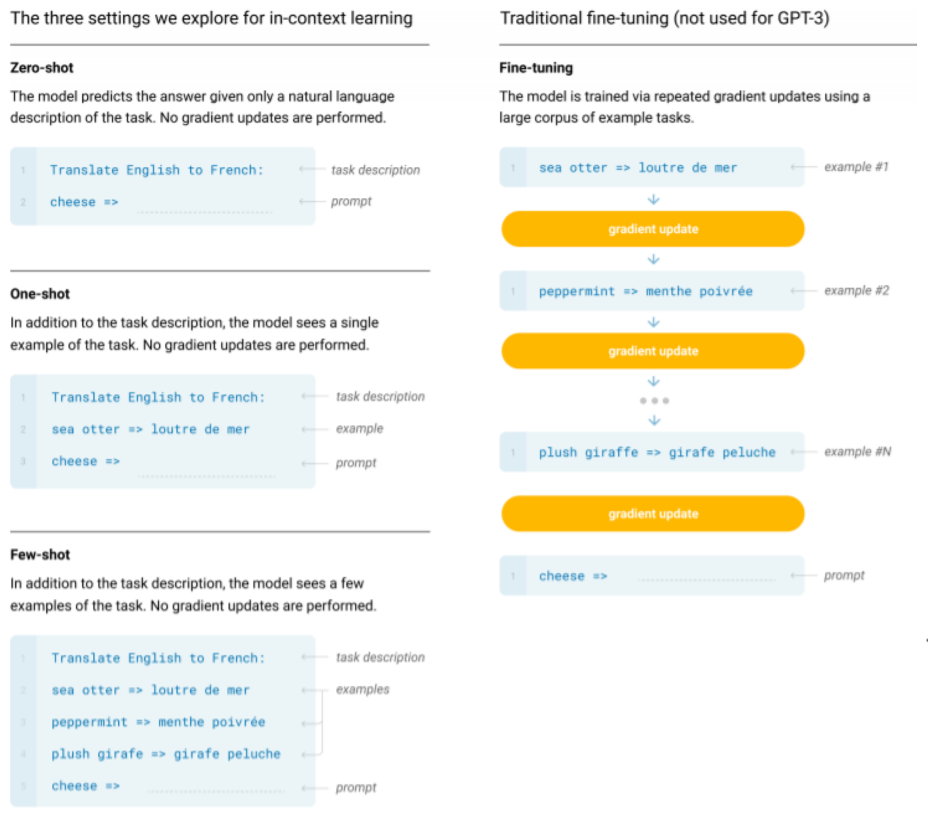
   : without fine-tunning over the corpora

   : compared to two KG (Wikidata, TAC KBP)

   : our KG also provide open factual knowledge (new)

0. LM contain knowledges (GPT-3)

Language Models are Few-Shot Learners


The three settings we explore for in-context learning

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French:      ← task description
2 cheese =>                         ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1 Translate English to French:      ← task description
2 sea otter => loutre de mer        ← example
3 cheese =>                         ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

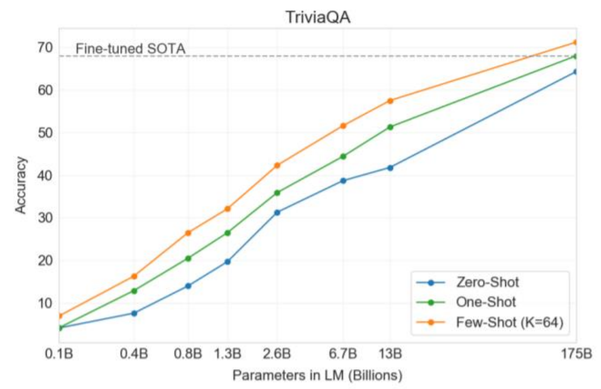
```
1 Translate English to French:      ← task description
2 sea otter => loutre de mer        ← examples
3 peppermint => menthe poivrée
4 plush girafe => girafe peluche
5 cheese =>                         ← prompt
```

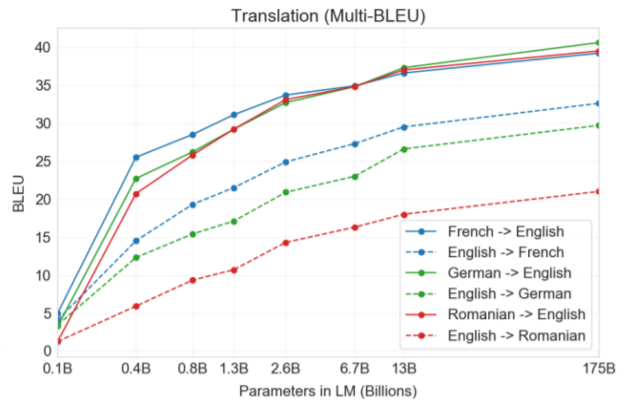
Traditional fine-tuning (not used for GPT-3)

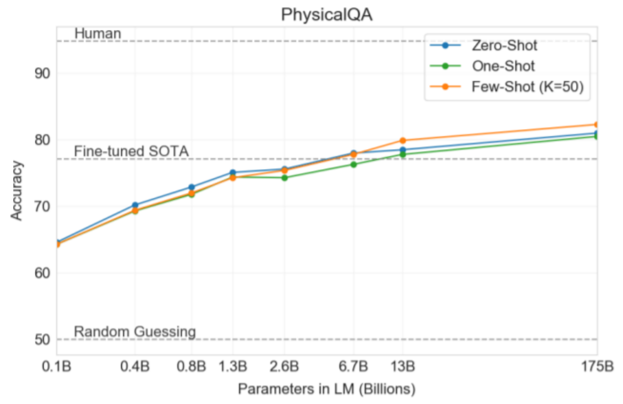
Fine-tuning

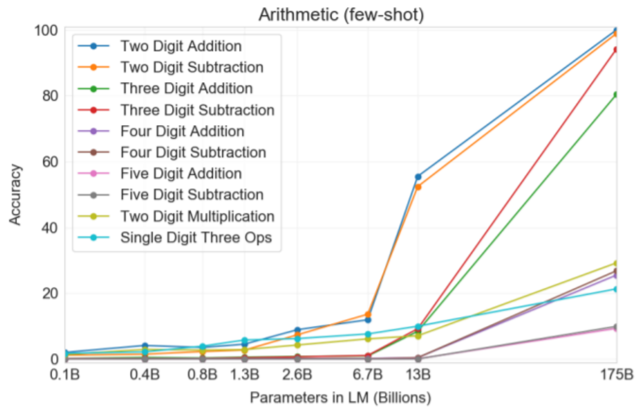
The model is trained via repeated gradient updates using a large corpus of example tasks.

```
1 sea otter => loutre de mer        ← example #1
        gradient update
1 peppermint => menthe poivrée      ← example #2
        gradient update
        ...
1 plush giraffe => girafe peluche   ← example #N
        gradient update
1 cheese =>                         ← prompt
```









1. Introduction

MAMA (Match and Map)

- Match: 후보 지식 생성 – LM내 textual corpus와 matching해서 (beam search in attention weight matrices)
- Map: 후보 지식이 이미 KG schema에 있는 형태면 추가, 아니면 open schema 생성

1. Introduction

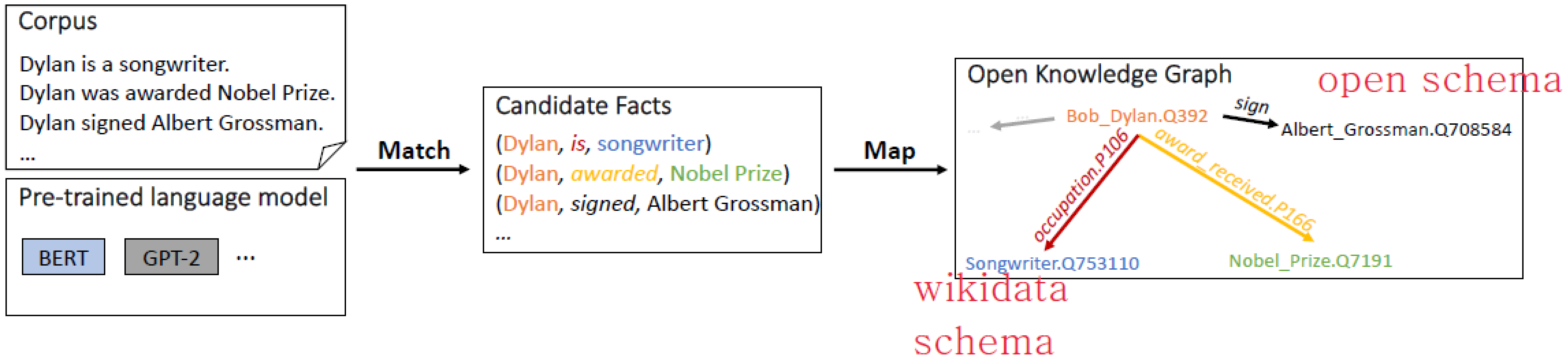
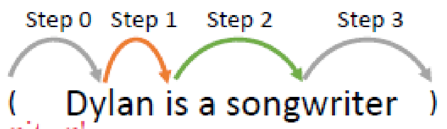


Figure 1: Overview of the proposed approach MAMA. MAMA constructs an open knowledge graph (KG) with a single forward pass of the pre-trained language model (LM) (without fine-tuning) over the corpus. Given the input: a textual corpus containing passages and sentences, e.g., English Wikipedia, and a pre-trained LM, e.g., BERT, GPT-2/3, MAMA (1) generates a set of candidate facts via *matching* the knowledge in the pre-trained LM with facts in the textual corpus, e.g., a candidate fact *(Dylan, is, songwriter)* from the sentence "Dylan is a songwriter.", and (2) produces an open KG by *mapping* the matched candidate facts to both an existing KG schema, e.g., *(Bob_Dylan.Q392, occupation.P106, Songwriter.Q753110)* in Wikidata schema, and an open schema, e.g., *(Bob_Dylan.Q392, sign, Albert_Grossman.Q708584)*.

2-1. Match



Given 'Dylan', 'songwriter'

start = add head

yield = not reach tail

stop = tail

Step	Action	Intermediate candidates	Matching degrees
0	START	(Dylan,	0
1	YIELD	(Dylan, is (0.3)	0.3
2	YIELD	(0.4) (Dylan, is songwriter	0.7
3	STOP	(Dylan, is, songwriter)	0.7

(a) Matching example.

Query:

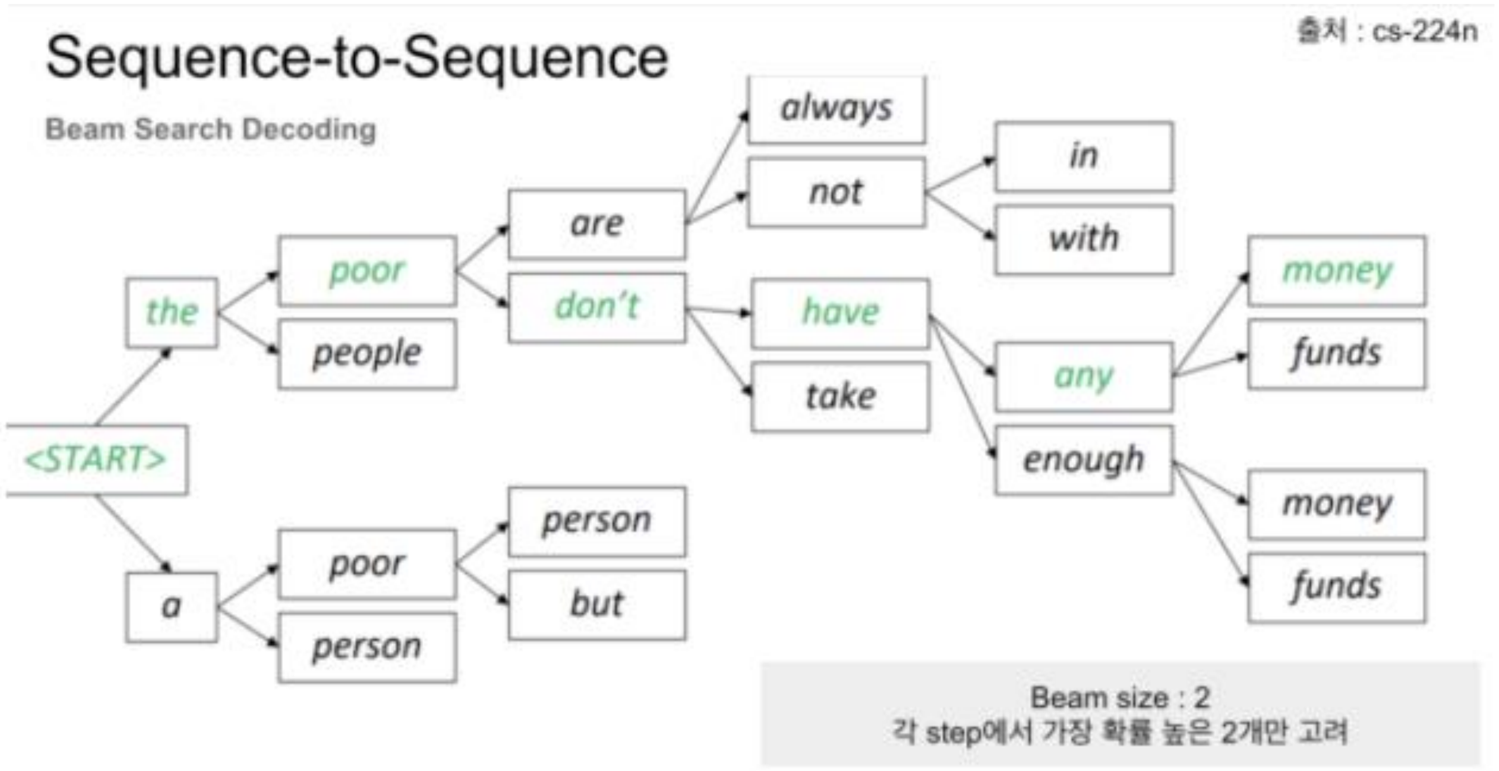
Key:	Dylan	is	a	songwriter
Dylan	x	x	x	x
is	0.3	x	x	x
a	0.1	0.2	x	x
songwriter	0.1	0.4	0.2	x

(b) Attention matrix for matching degree calculation.

Figure 2: Illustration of Match stage. The upper part of (a) represents the general matching steps of generating the best matched candidate fact *(Dylan, is, songwriter)* from the sentence "Dylan is a songwriter." The lower portion shows the corresponding step-by-step process. Given a head-tail pair *(Dylan, songwriter)*, at each step, the search chooses one of the actions, i.e., START, YIELD, STOP to produce an intermediate candidate fact. The search *starts* by adding the head "Dylan" as an initial candidate (step 0). The matching degree of the candidate is initialized as 0. Next, a new candidate is *yielded* if the candidate has not reached the tail "songwriter", by appending the next largest attended token (with the largest score from the attention matrix (b) of the sentence) to the end of the current candidate, and the corresponding matching degrees are increased by the associated attention scores (step 1 and step 2). Otherwise, the search *stops*, and the candidate fact with the best matching degree is returned for the head-tail pair (step 3). The attention matrix (b) is from the forward pass of the LM without fine-tuning over the sentence. "x" marks the tokens to prevent searching backward.

2.1.1 BEAM Search

Beam Search란 **최고우선탐색(Best-First Search)** 기법을 기본으로 하되 기억해야 하는 노드 수를 제한해 효율성을 높인 방식



2.1.1 BEAM Search

- **START**

: head 를 initial candidate 로 추가 (ex. ‘Dylan’ , into the beam)

: *START* (h)

- **YIELD**

: candidate not reached the tail

: the next largest attended token (largest score from attention)

: *YIELD* (c, s, As) -> c=current candidate, s=sentence, As = attention matrices

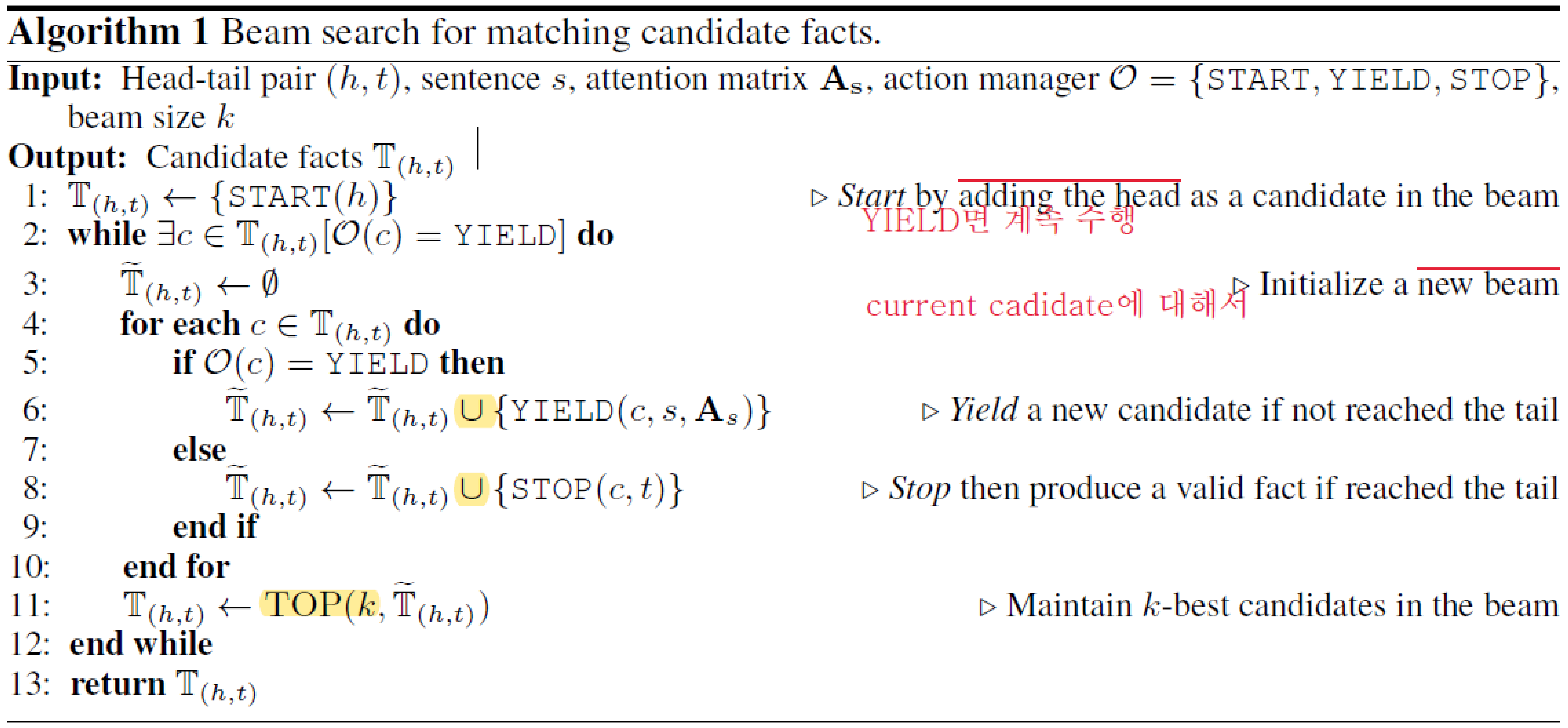
- **STOP**

: candidate reached the tail

: *STOP* (c, t) -> t= tail

2.1.1 BEAM Search

Running in both direction left->right, right->left
: for reverser order sentence



Algorithm 1 Beam search for matching candidate facts.

```
Input: Head-tail pair $(h, t)$, sentence $s$, attention matrix $\mathbf{A}_s$, action manager $\mathcal{O} = \{\mathtt{START}, \mathtt{YIELD}, \mathtt{STOP}\}$,
    beam size $k$
Output: Candidate facts $\mathbb{T}_{(h,t)}$
 1: $\mathbb{T}_{(h,t)} \leftarrow \{\mathtt{START}(h)\}$          ▷ Start by adding the head as a candidate in the beam
 2: while $\exists c \in \mathbb{T}_{(h,t)}[\mathcal{O}(c) = \mathtt{YIELD}]$ do    YIELD면 계속 수행
 3:     $\widetilde{\mathbb{T}}_{(h,t)} \leftarrow \emptyset$          current cadidate에 대해서 ▷ Initialize a new beam
 4:     for each $c \in \mathbb{T}_{(h,t)}$ do
 5:         if $\mathcal{O}(c) = \mathtt{YIELD}$ then
 6:             $\widetilde{\mathbb{T}}_{(h,t)} \leftarrow \widetilde{\mathbb{T}}_{(h,t)} \cup \{\mathtt{YIELD}(c, s, \mathbf{A}_s)\}$   ▷ Yield a new candidate if not reached the tail
 7:         else
 8:             $\widetilde{\mathbb{T}}_{(h,t)} \leftarrow \widetilde{\mathbb{T}}_{(h,t)} \cup \{\mathtt{STOP}(c, t)\}$   ▷ Stop then produce a valid fact if reached the tail
 9:         end if
10:     end for
11:     $\mathbb{T}_{(h,t)} \leftarrow \mathrm{TOP}(k, \widetilde{\mathbb{T}}_{(h,t)})$          ▷ Maintain $k$-best candidates in the beam
12: end while
13: return $\mathbb{T}_{(h,t)}$
```

2.1.2 Filter

Beam search 효과적이게 하기 위한 몇가지 제약사항

- **Threshold**

: matching degree of (h,r,t)에 threshold

: ex. (Rolling stone, wrote, pop song) <=   "Rolling stone wrote: "No other pop song has so thoroughly challenged artistic conventions""

: => 정확하지 않은 정보->below threshold

: ex. (Dylan, is, songwriter)같이 정확한 정보는 threshold위다.

2.1.2 Filter

- **Distinct Frequency**

: 자주 반복되는 사실은 threshold보다 위, **frequency r**로 이상만 선택

: 이를 통해, distinct head-tail pair를 얻을 수 있음

- **Relation**

: contiguous sequence in the sentence (인접성)

: 인접성에 기반하여 추출, 의미 없는 것은 삭제

Ex. (Rolling stone, wrote, pop song) => (Rolling stone, wrote challenged, conventions)

"Rolling stone wrote: "No other pop song has so thoroughly challenged artistic conventions""

2.2.1 Mapped Facts in KG Schema

?

: candidate fact (h,r,t)를 KG schema 내에 있는 (h_k, r_k, t_k) 과 map 하는 것

1) Entity linking

- Word embedding 를 이용한 entity linker to map h,t to $h_k,\ t_k$

2) Relation mapping

- Angeli et al. (2015)'s relation mapping method
- More often linked head-tail pairs co-occur between the candidate facts and KG facts. (자주 발생되는것 부터 r map)
- Normalized: lemmatization, removing inflection, auxiliary verbs, adjectives, adverbs.
- *Manually check* top 15 relation phrases are true mapping for each KG relation (하루 걸림)

2.2.2 Unmapped Facts in Open schema

?

Partially unmapped facts

- At least one of h, r, t are mapped to the KG schema
- h,t -> h_k, t_k / r -> r_k
- Ex. (Dylan, singed, Albert Grossman) -> Dylan, albert are linked to Wikidata schema (2.2.1 Entity linking에 의해)
- “singed”는 없는데, 이건 open schema로 처리한다.

Completely unmapped facts

- All h,r,t 모두 KG schema에 없을때, entity linker나 relation linker 소용 없음
- E.g (Jacob, was, a registered Mennonite) -> 없으니까 open schema에 남겨둠

Suggest mixes the fixed KG schema with the flexible open schema
-> extend the fixed KG with an additional open schema
-> benefit the downstream KG based application (QA, Reasoning)

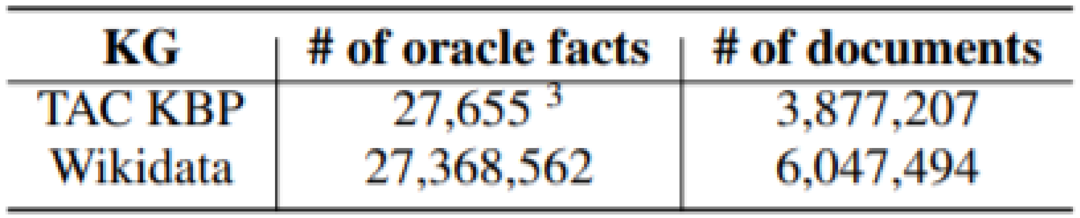
3. EXPERIMENTS

3.1 Mapped Fact

: comparing the mapped facts to oracle KG annotated by human

: **TAC KBP** (Knowledge Base Population) – tail/object entity사이에 predefined relation(slot)을 예측하는 task

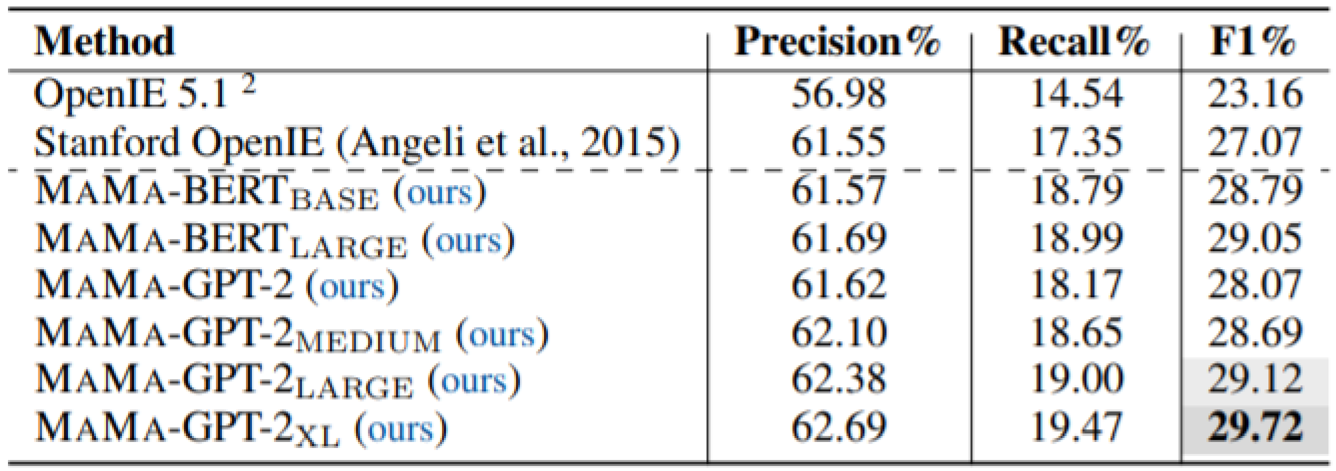
: **Wikidata** – English Wikipedia



KG	# of oracle facts	# of documents
TAC KBP	27,655 [3]	3,877,207
Wikidata	27,368,562	6,047,494

3.1.2 TAC KBP

MAMA에서 추출한 candidate fact랑 two open information system (Stanford OpenIE, OpenIE) 과 비교



Method	Precision%	Recall%	F1%
OpenIE 5.1 [2]	56.98	14.54	23.16
Stanford OpenIE (Angeli et al., 2015)	61.55	17.35	27.07
MAMA-BERT$_{BASE}$ (ours)	61.57	18.79	28.79
MAMA-BERT$_{LARGE}$ (ours)	61.69	18.99	29.05
MAMA-GPT-2 (ours)	61.62	18.17	28.07
MAMA-GPT-2$_{MEDIUM}$ (ours)	62.10	18.65	28.69
MAMA-GPT-2$_{LARGE}$ (ours)	62.38	19.00	29.12
MAMA-GPT-2$_{XL}$ (ours)	62.69	19.47	**29.72**

Table 2: Compare the quality of mapped facts on TAC KBP.

기존꺼보다 모두 성능 좋음

LM은 클수록 성능 좋음
비슷한 크기면 BERT가 좋음
BERTbase = GPT-2
BERTlarge = GPT-2 Medium

- Precision 60%이상 -> 만든 것 중에 실제로 kg에 있는 것, KG construct를 잘했다.
- Extra linguistic feature (POS tag, dependency parser)이런것 없이 LM을 통해서 triplet을 잘 뽑았다
- Recall 18%-19% -> incorrect entities caused by spaCy noun chunk 때문이다.
- 비슷한크기면 BERT -> BERT 의 Cloze-style loss function (masked LM)이 knowledge담는데 더 효과적? (Recall)
- Precision은 GPT-2가 높음. Autoregressive LM loss가 Cloze-style loss보다 noise가 없으니.

3.1.2 TAC KBP

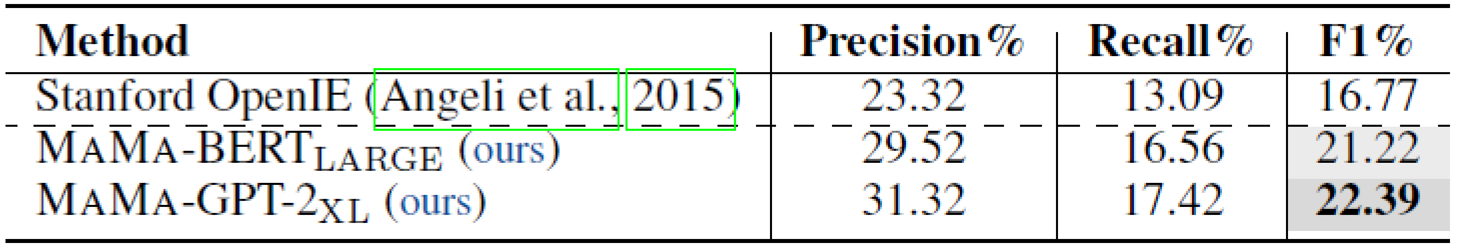
Stanford OpenIE

- POS tag & dependency parser, generate self-contained clauses (extract the triplet from sentence)
- After collecting triplet, same MAP procedure with MAMA

OpenIE 5.1

- Noun relation, numerical sentences, conjunctive sentences depending on the linguistic patterns.
- After collecting triplet, same MAP procedure with MAMA

3.1.3 Wikidata



Method	Precision%	Recall%	F1%
Stanford OpenIE (Angeli et al., 2015)	23.32	13.09	16.77
MAMA-BERT$_{\text{LARGE}}$ (ours)	29.52	16.56	21.22
MAMA-GPT-2$_{\text{XL}}$ (ours)	31.32	17.42	**22.39**

기존꺼보다 모두 성능 좋음

Table 3: Compare the quality of mapped facts on Wikidata.

- **MAMA is scalable to larger corpora**

: GPT-2xl > BERT large, finetuning 없이 inference만 하니 scalable하다.

- **Larger corpora embed more compled KGs**

: Wikidata결과차이가 TAC KBP차이보다 크다. (F1score 5.6 > 2.7)

: MAMA could improve the recall by running on those larger corpora to collect more facts (????)

3.2 Unmapped Facts

?

Manually judge from 100 sampled documents in two datasets

The quality of unmapped facts is verified

- 35.3% of the unmapped facts are true and 83.2% partially unmapped facts on Wikidata.

Accurate entity detection is desired

- 45.5% of the untrue unmapped facts on Wikidata
- Cause) Incorrect entity detected by spaCy noun chunk때문
- Cause) Incorrect or missing entity linking (9.1% error)
- Cause) missing relation mapping (4.5% error)
- Cause) incorrect relation phrases (OpenIE와 동일한 error)

E.g. (Dylan, made, his breakthrough) ->uninformative

MAP stage rely heavily on the accuracy of entity detection from the spaCy noun chunk

Future work

1. Crowd sourcing으로 채점 더 객관적으로
2. Attention에 기반해서 더 정확한 entity 추출 (spaCy 말고)
3. Entity linker더 좋은것 활용, robust relation mapping 모델 개발
4. Relation generation 좀 더 정교하게, GNN같은걸로 attention weight matrices랑 structural information같이 쓰게

Under Review

- The **head and the tail entities** are always **single words**, whereas many entities such as names have two or more words.
- The comparison is made only against **Stanford OpenIE,** which was proposed **5 years ago.**
- The paper highly depends on the grounding techniques from many years go, namely Spitkovsky & Chang (2012) and Stanford OpenIE (2015).
- If all possible head-tail pairs are enumerated in a sentence, my expectation is that there will be a lot of **garbage triplets**. It is amazing yet hard to believe that such simple thresholding techniques make it work and get rid of all of the bad triplets. More analysis will be helpful.
- the insight that knowledge base is embedded in the world knowledge, which is **captured by the pretrained embeddings**, **is not new**. [1] *Wang, Zhen, et al. "Knowledge graph and text jointly embedding." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.*
- The main **evaluation in the paper is done on the grounded facts** after the mapping stage. While the details are hazy, based on Section 2.2.1, it seems the mapping stage relies on a pre-existing KB aligned with a corpus to do the entity linking and relation linking steps.
- important details are missing about **the interaction between the KG used for the entity and relation linkers** and the KG used for evaluation. Specifically, what is the size of the KG available for entity / relation linkers, and were there any overlapping facts between this and the facts used for evaluation?
- Several recent papers have looked at probing **LMs for knowledge facts** (e.g. "Language Models as Knowledge Bases" Petroni et al, EMNLP 2019, and follow up papers). These are very relevant, as another approach for constructing KGs **from pretrained LMs is to use natural language templates**. But there is no discussion of these works in the paper
- it seems **unclear whether these improvements are due to the factual knowledge encoded** in the pre-trained LM, as claimed.