Should We Rely on Entity Mentions for Relation Extraction? Debiasing Relation Extraction with Counterfactual Analysis (2022)

Yiwei Wang
Muhao Chen
Wenxuan Zhou
Yujun Cai
Yuxuan Liang
Dayiheng Liu
Baosong Yang
Juncheng Liu
Bryan Hooi

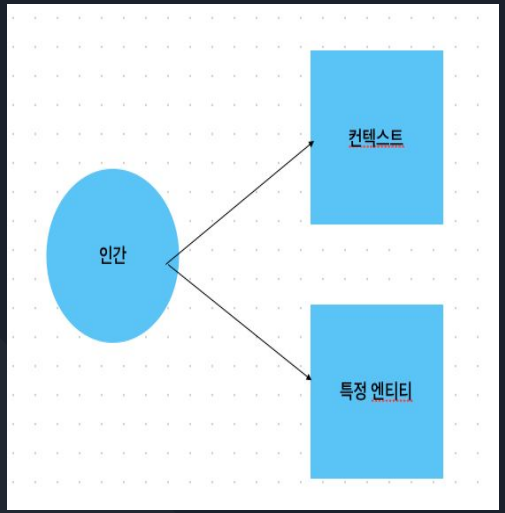
인용 10회

발표자 최윤진

📌 목차

특정 엔티티에 편향되지 않는
컨텍스트에 편향되지 않는

일반적인 성능을 보이는 RE 모델을 찾고 싶다!



📌 몇 가지 탐색

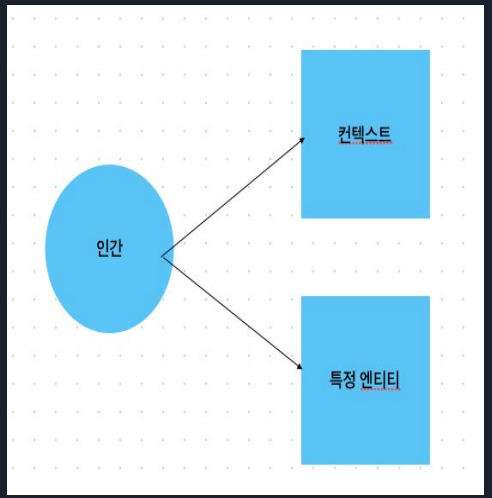
- **An Improved Baseline for Sentence-level Relation Extraction**
  - 마커 추가 -> 엔티티 식별
- **Relation Classification with Entity Type Restriction, ACL2021**
  - 라벨 영역 제한 ->
- **PTR: Prompt Tuning with Rules for Text Classification**
  - 프롬프트 튜닝 ->
- **Improving Sentence-Level Relation Extraction Through Curriculum Learning**
  - 난이도 별 단계 학습 ->

1. Introduction & Related Work

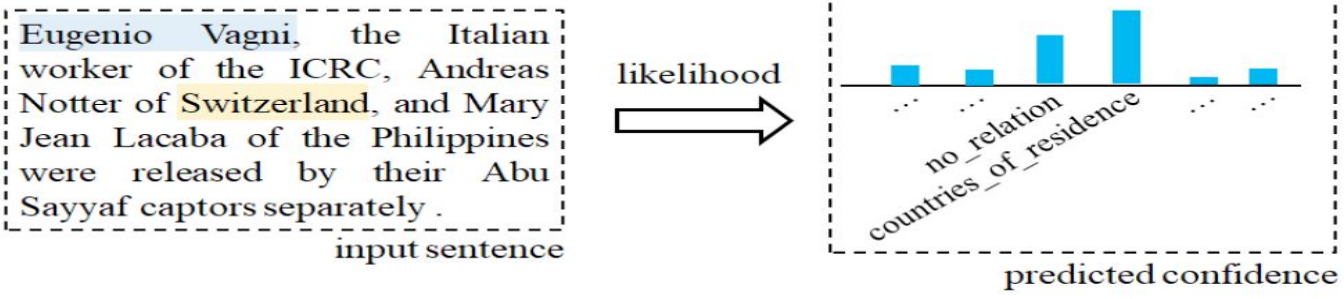
- 엔티티 멘션에 의존하게 되면 엔티티 편향이 발생한다.
  - 대표적 사례 : duke university 만 떴다 하면 “다닌다"
  - 일하는 공간일 수도 있음

- 마스킹 하게 되면 semantic information (의미 정보) 를 잃게 됨
  - I graduate duke university
  - I graduate [MASK]

- 문장 수준의 RE
  - entity 정보, 편향성
  - 엔티티 마스킹 -> GCN, SpanBERT -> 성능 저하
- 자연어 처리를 위한 디베이싱은 근본적 문제

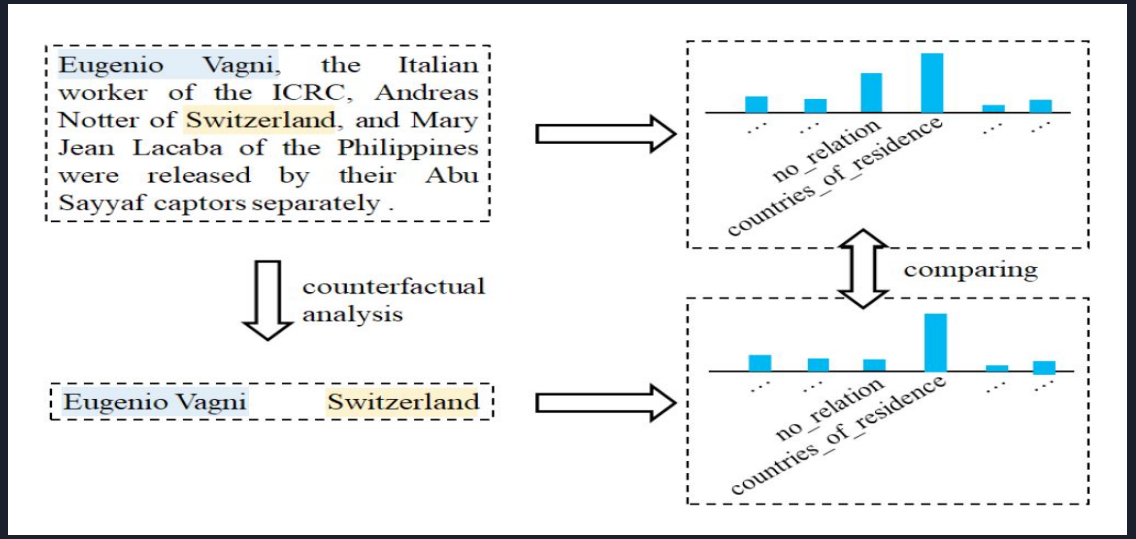


1. Introduction & Related Work



- 인간은 어떻게 판단하는가?
  - 텍스트 문맥의 인과 관계를 고려한다.
- 기존의 RE 모델은 어떻게 판단하는가?
  - 거대한 확률 테이블에서 개체 언급과 텍스트 문맥을 찾는 것에 불과하다.
- 📌 이제, CoRE 방법을 통해 인간을 따라해보자.

1. Introduction & Related Work



- CoRE
  - 텍스트 문맥을 보지 않았다면 여전히 동일한 관계를 추출할 수 있을까요?
  - 1) 위 그림에서 기존 문장을 토대로 판단을 내렸을 때
  - 2) 엔티티 자체로 (텍스트 문맥을 보지 않고) 판단을 내렸을 때
  - 1) 2) 의 결과 분석을 종합하는 것이 반사실적 분석

2. Methodology (방법론)

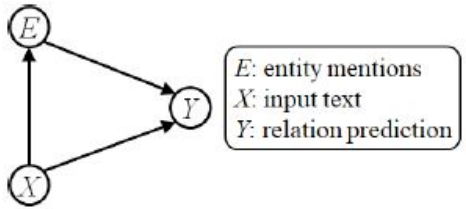
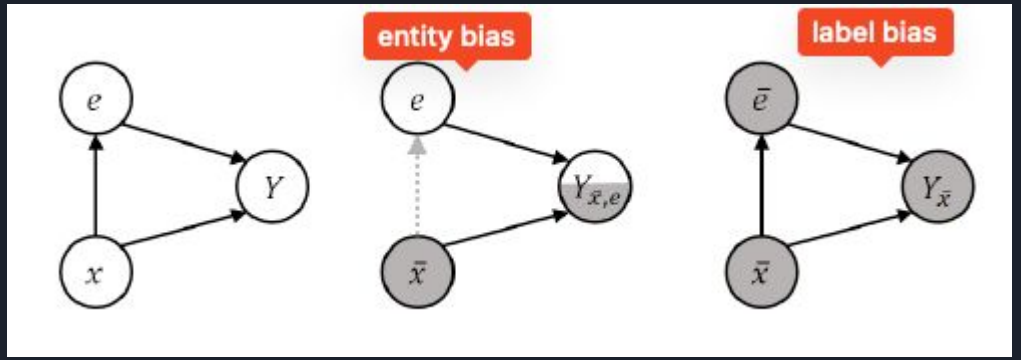


그림 2: RE 모델의 인과 그래프.

- 인과 관계 그래프를 통해 엔티티 편향성을 추출
- RE 모델이 의사 결정을 내릴 때 엔티티 멘션에 얼마나 많이 의존하는 지 분석

- 3.1 Causality of Relation Extraction
  - X -> E
    - NER, 사람 주석 을 통해 주어와 목적어의 스팬(노드 E) 를 얻는다.
    - X : “Mary give birth to Jerry”
    - E : [‘Marry’, ‘Jerry’]
  - (X, E) -> Y
    - C-GCN (컴폴류션 그래프)
    - IRE (엔티티 마스킹, 스페셜 토큰)

2. Methodology



- 3.2 Bias Distillation
  - do(-) : 인과 관계 그래프에서 들어오는 모든 노드 link를 지운다.
    - 개입을 통하여 인과 관계의 노드를 조작한다.
  - [맨 왼쪽 그림] 기존 RE 모델 : (x,e)-> Y
  - [맨 오른쪽 그림] textual context 와 entity mentions 이 모두 제거하는 방식 - 결국 아무것에도 접근할 수 없기 때문에 때문에 모델에 존재하는 레이블 편향성을 반영

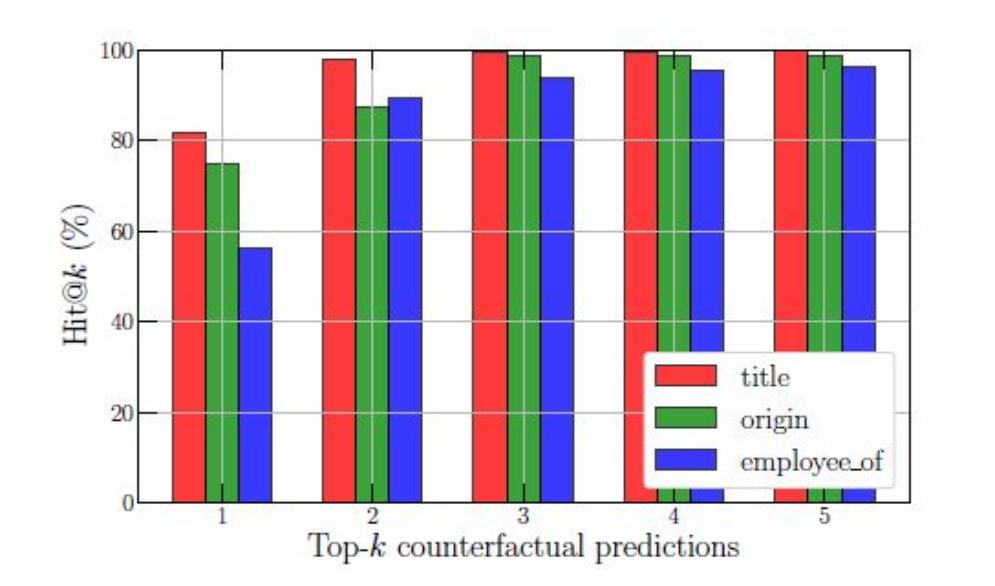
$$Y_{\bar{x}} = Y(do(X = \bar{x})).$$

  - [가운데 그림] e 는 그대로 납둔 상태에서, x 토큰은 마스킹 처리
    - 컨텍스트는 제거하고 엔티티 멘션만으로 Y 를 추론할 수가 있다.

$$Y_{\bar{x},e}$$

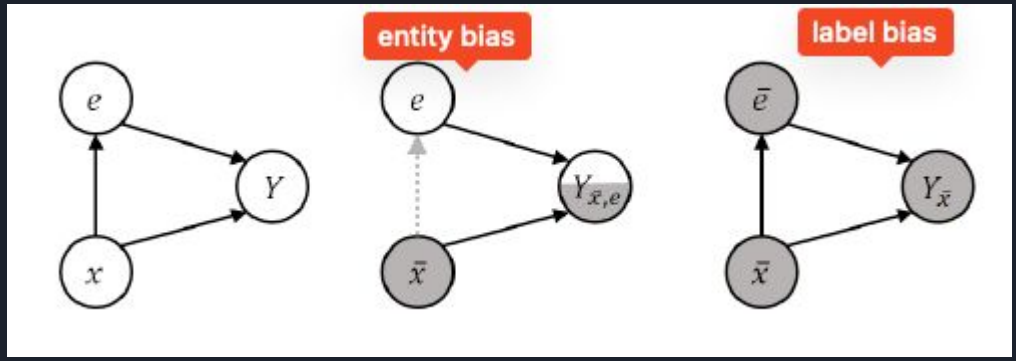
- Counterfacutal ? -> 자연스럽게 보이는 사실의 내부를 두 번 생각하기 때문에 반 사실적 !

2. Methodology



- Hit@k
  - 엔티티 멘션만 주어졌을 때 올바른 예측을 할 수 있는 비율
  - 더 많은 인스턴스에서 텍스트 컨텍스트가 제공 되었는지 여부와 관계없이 개체 멘션만 주어졌을 때 동일한 관계를 추론한다는 의미
  - **이 수치가 높다는 것은 엔티티 멘션에 의존성이 높고 편향성이 높다는 것임.**
    - 컨텍스트 없이도 동일한 관계를 추출할 수 있으므로.

2. Methodology



새로운 라벨은 다음 식을 통해 도출된다.

$$Y_{\text{final}} = Y_x - \lambda_1 Y_{\bar{x},e} - \lambda_2 Y_{\bar{x}},$$

$$\lambda_1^\star, \lambda_2^\star = \arg\max_{\lambda_1,\lambda_2} \psi(\lambda_1, \lambda_2) \quad \lambda_1, \lambda_2 \in [a, b],$$

3. Experiments

- 4.1 실험적 설정
  - RE bechmark
    - TACRED
    - SemEval
    - TACRED-Revisit
    - RE-TACKED
  - 비교 방법
    - LUKE, IRE
  - debiasing
    - Focal : Focal loss - loss 를 reweights
    - Resample - 희귀 카테고리를 upsampling
    - Entity Mask : 엔티티 가리기
    - CFIE : 인과 관계를 강화 시키는 방법

Method	TACRED	Re-TACRED
LUKE (Yamada et al., 2020)	51.9	65.3
w/ Resample (Burnaev et al., 2015)	53.2	66.7
w/ Focal (Lin et al., 2017)	52.4	65.9
w/ CFIE (Nan et al., 2021)	52.1	65.6
w/ Entity Mask (Zhang et al., 2017)	54.5	67.1
w/ CoRE (ours)	**69.3**	**83.1**
$\text{IRE}_{\text{RoBERTa}}$ (Zhou and Chen, 2021)	56.4	68.1
w/ Resample (Burnaev et al., 2015)	58.1	70.3
w/ Focal (Lin et al., 2017)	56.8	68.7
w/ CFIE (Nan et al., 2021)	57.1	68.4
w/ Entity Mask (Zhang et al., 2017)	57.3	68.9
w/ CoRE (ours)	**73.6**	**85.4**

3. Experiments

Input sentence	Original	Debiased	Counterfactual
More than 1,100 miles (1,770 kilometers) away, Alan Gross passes his days in a Cuban military hospital, watching baseball on a small television or jamming with his jailers on a stringed instrument they gave him.	origin ✗	countries_of_residence ✓	origin
He said that according to his investigation, Bibi drew the ire of fellow farmhands after a dispute in June 2009, when they refused to drink water she collected and she refused their demands that she convert to Islam.	religion ✗	no_relation ✓	religion
ShopperTrak also estimates foot traffic in the U.S. was 11.2 percent below what it would have been Sunday if the blizzard had not occurred and 13.9 percent below what it could have been Monday.	country_of_headquarters ✗	no_relation ✓	country_of_headquarters

- 설명가능성에 대한 확보.
- bibi 와 Islam

4. Conclusion

- CoRE
- 인과 그래프를 통해 엔티티 편향성을 추출 -> 편향성을 완화
  - 엔티티 정보를 활용하는 데 초점을 맞추는 것은 엔티티 편향을 만들어 냄
    - RE 모델이 텍스트에 존재하지 않는 관계를 추출하도록 오도할 수 있음.
  - 엔티티 멘션을 마스킹하는 방법은 엔티티의 의미론적 정보를 잃기 때문에 성능이 저하

CoRE : Counterfactual Analysis based Relation Extraction 는 엔티티 정보를 잃지 않으면서 텍스트 문맥의 주요 효과에 집중할 수 있도록 디바이어싱(debiasing) 한다.

- 방법론 : 인과 그래프로부터 엔티티 언급의 편향성을 완화 시킨다.
- 학습(train) 과정을 건드리지 않고 추론 중에 RE 시스템의 편향을 제거할 수 있다.

5. Code

```
for batch_start_idx in trange(0, len(test_examples), batch_size):
    batch_examples = test_examples[batch_start_idx:batch_start_idx + batch_size]
    texts = [example["text"] for example in batch_examples]
    entity_spans = [example["entity_spans"] for example in batch_examples]
    gold_labels = [example["label"] for example in batch_examples]

    inputs = tokenizer(texts, entity_spans=entity_spans, return_tensors="pt", padding=True)
    inputs = inputs.to("cuda")
    with torch.no_grad():
        outputs = model(**inputs)

    predicted_indices = outputs.logits.argmax(-1)
    predicted_labels = [model.config.id2label[index.item()] for index in predicted_indices]
    pred_ls.append(outputs.logits.cpu().numpy())
    label_ls = label_ls + [label_ for label_ in gold_labels]
```

5. Code

```
for batch_start_idx in trange(0, len(test_examples), batch_size):
    batch_examples = test_examples[batch_start_idx:batch_start_idx + batch_size]
    texts = [example["text"] for example in batch_examples]
    entity_spans = [example["entity_spans"] for example in batch_examples]
    gold_labels = [example["label"] for example in batch_examples]

    texts = [
            texts[i_][entity_spans[i_][0][0] : entity_spans[i_][0][1]]
            +
            ' '
            +
            texts[i_][entity_spans[i_][1][0] : entity_spans[i_][1][1]]
            for i_ in range(len(texts))
            ]
    entity_spans = [
                    [(0, entity_spans[i_][0][1] - entity_spans[i_][0][0]),
                    (entity_spans[i_][0][1] - entity_spans[i_][0][0] + 1,
                     entity_spans[i_][0][1] - entity_spans[i_][0][0] + 1 + entity_spans[i_][1][1] - entity_spans[i_][1][0]
                    )]
                    for i_ in range(len(texts))
                   ]

    inputs = tokenizer(texts, entity_spans=entity_spans, return_tensors="pt", padding=True)
    inputs = inputs.to("cuda")
    with torch.no_grad():
        outputs = model(**inputs)

    predicted_indices = outputs.logits.argmax(-1)
    predicted_labels = [model.config.id2label[index.item()] for index in predicted_indices]
    pred_ls.append(outputs.logits.cpu().numpy())
    label_ls = label_ls + [label_ for label_ in gold_labels]
```

5. Code

```
# normalize the predicted logits as probabilities by softmax
luke_prob = sp.softmax(luke_prob, axis = 1)
luke_prob_mask = sp.softmax(luke_prob_mask, axis = 1)

# transform the luke prediction indices to the original label indices.
luke_label_to_id = {value_ : key_ for key_, value_ in luke_id_to_label.items()}
org_to_luke = [luke_label_to_id[ID_TO_LABEL[i_]] for i_ in range(len(luke_label_to_id.values()))]
luke_prob = luke_prob[:, org_to_luke]
luke_prob_mask_1 = luke_prob_mask[:, org_to_luke]
luke_preds = luke_prob.argmax(1)
luke_preds_tde = luke_prob_mask_1.argmax(1)

# filter the challenge set that the relation labels implied by the entity bias does not exist in the sentence.
challenge_set = [i_ for i_ in range(len(luke_prob)) if luke_preds_tde[i_] != keys[i_]]

keys = keys[challenge_set]
luke_preds = luke_preds[challenge_set]
luke_prob = luke_prob[challenge_set]
luke_prob_mask_1 = luke_prob_mask_1[challenge_set]
luke_prob_mask_2 = luke_prob_mask_2[challenge_set]
label_constraint = label_constraint[challenge_set]

print('f1 score before bias mitigation: ', getF1(keys, luke_preds))

lamb_1 = -1.6
lamb_2 = 0.1
new_preds = (luke_prob + lamb_1 * luke_prob_mask_1 + lamb_2 * luke_prob_mask_2 + label_constraint).argmax(1)
print('f1 score after bias mitigation: ', getF1(keys, new_preds))
```

6. Q&A

감사합니다.

6. Appendix

- CoRE 코드가 구현된 깃허브 레포지토리
  - https://github.com/vanoracai/CoRE/blob/main/core.py
-