NAACL 2022
논문 리뷰

구선민

Index

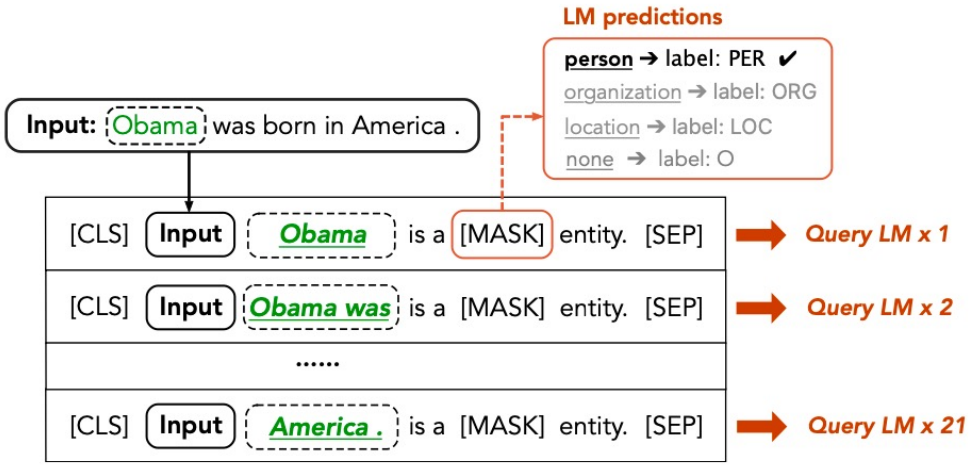
Template-free Prompt Tuning for Few-shot NER

Motivation

- Motivation of Entity-oriented LM (EntLM)
  - 최근에는 classification task를 cloze questions으로 reformulating하는 prompt-based learning이 few-shot classification task에서 좋은 성능을 냈음
    - Typically, for each input [X], a template is used to convert [X] into an unfilled text (e.g., "[X] It was __.") → 모델이 언어 모델링 능력에 따라 빈칸을 채움
  - few-shot classification에서 prompt-based learning 좋은 성능 낸 주요인
    - masked LM objective를 재사용하여 pre-training and fine-tuning에 사용되는 다양한 training object 간의 격차를 줄이는데 도움
    - 정교한 template and label word design은 LM이 task-specific answer distribution에 더 fit 하게 만들어줘서 few-shot 성능에 도움

Motivation

- Motivation of Entity-oriented LM (EntLM)
  - 그러나 template-based prompt methods는 sentence-level tasks를 위해서 디자인되어 있기 때문에 NER과 같은 token-level labeling tasks에 적용하기 어려움
    - NER에서 span-level querying가 발생할 때 search space가 커질 수록 적절한 템플릿 찾기 어려움
    - 각 토큰의 레이블을 얻으려면 가능한 모든 possible spans을 나열해야 하기때문에 시간이 많이 걸릴 수 있음



Main Idea

- Entity-oriented LM (EntLM) fine-tuning
  - prompt-tuning의 장점을 유지하면서 few-shot NER에 대해 template-free prompt tuning method인 Entity-oriented LM (EntLM) fine-tuning 제안

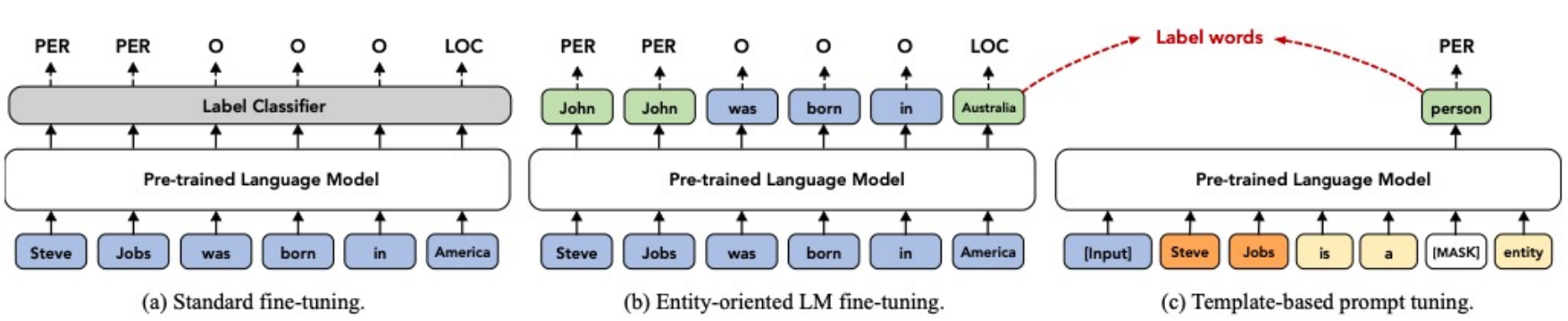
  - entity position에서 class-related pivot word (or label word)를 예측하기위해 PLM의 word prediction paradigm를 유지하면서 template construction process 없앰
    - output head를 수정하지 않고 PLMs는 entity position에서 original word 대신 class-related pivot word (or label word)를 예측하도록 fine-tuning, none-entity positions에서는 기존 방법대로 original word를 예측

    - label word set을 task label set과 매핑한 다음, input the input sentence X와 corresponding label sequence Y가 주어지면 entity position의 토큰을 label word로 replace하고 none-entity positions는 original words로 남겨둬서 target sentence를 만듦

    - 그 다음에 original input X가 주어지면 LM은 target sentence의 확률을 maximize하도록 훈련

Main Idea

- Comparison of different fine-tuning methods for NER



(a) Standard fine-tuning process for NER은 LM head를 token-level classification head로 바꾸고, 새로운 파라미터와 PLM을 optimizing
(b) EntLM
(c) Template-based prompt tuning은 classification tasks를 LM task로 reformulate하고 LM을 fine-tuning 하여 label word를 예측

Main Idea

- Label Word Engineering
  - PLM이 쉽게 adapt할 수 있는 적절한 label word를 자동으로 검색할 수 있는 방법 탐구

  - Low-resource Label word selection
    - Entity lexicon를 통해 annotation을 얻기 위해 external KBs인 위키데이터를 source of lexicon annotation를 사용하는 기존 연구 방법 사용

  - Label word searching
    - Searching with data distribution (Data search): 주어진 클래스에서 가장 빈도가 높은 word 선택
    - Searching with LM output distribution (LM search): 각 sample을 LM에 넣어서 각 position의 word를 예측하는 확률 분포값 얻어서 클래스에서 positions labeled의 top k predictions 얻음
    - Searching with both data & LM output distribution (Data&LM seach): data distribution와 LM output distribution을 동시에 고려해서 label word 선택

  - Removing conflict label words
    - label word 선택 후 manually set threshold를 넘는 클래스와 confict하는 label word를 제거

Experiments

- Settings
  - 3개의 서로 다른 도메인 데이터셋 사용
    - CoNLL03: newswire domain
    - OntoNotes 5.0: general domain
    - MIT-Movie: review domain
  - Different few-shot settings
    - K=5,10,20,50

Datasets	Methods	K=5	K=10	K=20	K=50
CoNLL03	BERT-tagger (IO)	41.87 (12.12)	59.91 (10.65)	68.66 (5.13)	73.20 (3.09)
	NNShot	42.31 (8.92)	59.24 (11.71)	66.89 (6.09)	72.63 (3.42)
	StructShot	45.82 (10.30)	62.37 (10.96)	69.51 (6.46)	74.73 (3.06)
	Template NER	43.04 (6.15)	57.86 (5.68)	66.38 (6.09)	72.71 (2.13)
	EntLM (Ours)	49.59 (8.30)	64.79 (3.86)	69.52 (4.48)	73.66 (2.06)
	EntLM + Struct (Ours)	**51.32** (7.67)	**66.86** (3.01)	**71.23** (3.91)	**74.80** (1.87)
OntoNotes 5.0	BERT-tagger (IO)	34.77 (7.16)	54.47 (8.31)	60.21 (3.89)	68.37 (1.72)
	NNShot	34.52 (7.85)	55.57 (9.20)	59.59 (4.20)	68.27 (1.54)
	StructShot	36.46 (8.54)	57.15 (5.84)	62.22 (5.10)	68.31 (5.72)
	Template NER	40.52 (8.62)	49.89 (3.66)	59.53 (2.25)	65.15 (2.95)
	EntLM (Ours)	45.21 (9.17)	57.64 (4.18)	65.64 (4.24)	71.77 (1.31)
	EntLM + Struct (Ours)	**46.60** (10.35)	**59.35** (3.24)	**67.91** (4.55)	**73.52** (0.97)
MIT-Movie	BERT-tagger (IO)	39.57 (6.38)	50.60 (7.29)	59.34 (3.66)	71.33 (3.04)
	NNShot	38.97 (5.54)	50.47 (6.09)	58.94 (3.47)	71.17 (2.85)
	StructShot	41.60 (8.97)	53.19 (5.52)	61.42 (2.98)	72.07 (6.41)
	Template NER	45.97 (3.86)	49.30 (3.35)	59.09 (0.35)	65.13 (0.17)
	EntLM (Ours)	46.62 (9.46)	57.31 (3.72)	62.36 (4.14)	71.93 (1.68)
	EntLM + Struct (Ours)	**49.15** (8.91)	**59.21** (3.96)	**63.85** (3.7)	**72.99** (1.80)

Experiments

- Results

Datasets	Methods	K=5	K=10	K=20	K=50
CoNLL03	BERT-tagger (IO)	41.87 (12.12)	59.91 (10.65)	68.66 (5.13)	73.20 (3.09)
	NNShot	42.31 (8.92)	59.24 (11.71)	66.89 (6.09)	72.63 (3.42)
	StructShot	45.82 (10.30)	62.37 (10.96)	69.51 (6.46)	74.73 (3.06)
	Template NER	43.04 (6.15)	57.86 (5.68)	66.38 (6.09)	72.71 (2.13)
	EntLM (Ours)	49.59 (8.30)	64.79 (3.86)	69.52 (4.48)	73.66 (2.06)
	EntLM + Struct (Ours)	**51.32** (7.67)	**66.86** (3.01)	**71.23** (3.91)	**74.80** (1.87)
OntoNotes 5.0	BERT-tagger (IO)	34.77 (7.16)	54.47 (8.31)	60.21 (3.89)	68.37 (1.72)
	NNShot	34.52 (7.85)	55.57 (9.20)	59.59 (4.20)	68.27 (1.54)
	StructShot	36.46 (8.54)	57.15 (5.84)	62.22 (5.10)	68.31 (5.72)
	Template NER	40.52 (8.62)	49.89 (3.66)	59.53 (2.25)	65.15 (2.95)
	EntLM (Ours)	45.21 (9.17)	57.64 (4.18)	65.64 (4.24)	71.77 (1.31)
	EntLM + Struct (Ours)	**46.60** (10.35)	**59.35** (3.24)	**67.91** (4.55)	**73.52** (0.97)
MIT-Movie	BERT-tagger (IO)	39.57 (6.38)	50.60 (7.29)	59.34 (3.66)	71.33 (3.04)
	NNShot	38.97 (5.54)	50.47 (6.09)	58.94 (3.47)	71.17 (2.85)
	StructShot	41.60 (8.97)	53.19 (5.52)	61.42 (2.98)	72.07 (6.41)
	Template NER	45.97 (3.86)	49.30 (3.35)	59.09 (0.35)	65.13 (0.17)
	EntLM (Ours)	46.62 (9.46)	57.31 (3.72)	62.36 (4.14)	71.93 (1.68)
	EntLM + Struct (Ours)	**49.15** (8.91)	**59.21** (3.96)	**63.85** (3.7)	**72.99** (1.80)

- template-based process를 없애는 동시에 pre-training과 fine-tuning의 서로 다른 object 간의 차이를 줄여 few-shot 성능을 높이는데 도움이 될 수 있음
- fine-tuning 중에 새로운 파라미터가 추가 되지 않기 때문에 prompt-based learning의 장점을 유지
- 모든 엔티티 레이블을 얻기 위해 one-pass decoding만 요구되기 때문에 template-based methods 보다 훨씬 시간 효율적

Joint Extraction of Entities, Relations, and Events via Modeling Inter-Instance and Inter-Label Dependencies

Motivation

- IE pipeline involves four major tasks
  - (1) Event trigger detection(ETD) (2) Entity mention recognition(EMR) (3) Event argument extraction(EAE) (4) Relation extraction(RE)

  - 최근에는 error propagation을 막고 leverage dependency between prediction instances of the four IE tasks를 위해 ETD, EMR, EAE, and RE을 joint Information Extraction (JointIE)해서 해결
    - ex. if a *Person* entity mention is a *Victim* argument for a *Die* event → entity mention이 같은 sentence에서 *Attack* event에 대한 *Target* argument일 가능성이 높음

Motivation

- IE pipeline involves four major tasks
  - 이전 JointIE에서 태스크 인스턴스 간의 dependency를 캡처하기 위해 entity/event mention candidates에 대해 모든 가능한 text spans을 같이 가져와서 dependency graph에 대한 representation 향상시킴
  - 이는 non-entity/event mentions에 대한 text span이 많고,모델링 할 때 노이즈가 들어갈 수 있기 때문에 모델의 효율성을 해칠 수 있는 large dependency graph 만들어질 수 있음
  - 이러한 한계점을 극복하기 위해 데이터로부터 태스크 인스턴스 간의 dependency graph를 유도해서 representation learning 강화하는 것을 제안

Main Idea

- Problem Statement
  - ETD의 object는 input 문장이 주어지면 사전 정의된 event type set(e.g., “Attack” and “Transport”)을 기반으로 event triggers에 대한 text spans과 event types을 예측하는 것
  - EMR은 문장 안의 entity mention에 대한 text spans and entity types (e.g., “Person”, “Organization”)을 결정하는 역할
  - EAE는 entity mention이 이벤트 트리거에 대한 argument role (e.g, “Victim”)을 예측
    - entity mention이 이벤트 트리거에 대한 argument role 아니면 “Not-an-argument” 으로 표시
  - RE는 주어진 엔티티 멘션 pair에 대한 classification of relation
    - 두 엔티티 간에 relation이 없으면 “No-relation”으로 표시

Main Idea

- **Identifying event and entity mentions**
  - Given an input sentence w = [w1,...,wN] with N words, two corresponding sequence tagging problems for event and entity mentions를 해결함으로써 event triggers and entity mentions을 identify
  - BIO tagging schema를 사용하여 w의 각 word에 2개의 레이블을 할당하여 text spans of event triggers and entity mentions를 표시
    - {*"B-TRIGGER", "I-TRIGGER", "O"*} labels for event triggers
    - {*"B-ENTITY", "I-ENTITY", "O"*} labels for entity mentions
  - BERT를 사용하여 시퀀스 안의 word들에 대한 contextualized embeddings(X)을 얻고 벡터 시퀀스 X를 서로 다른 2개의 Conditional Random Fields(CRF) layers에 넣어서 two distributions for the tag sequences of w for event triggers and event mentions을 계산

Main Idea

- Identifying event arguments and relations
  - Given the detected event triggers and entity mentions, 각 event-entity mentions pair와 entity-entity mentions대한 representation vector를 얻음

$$z_{ij}^{a} = FFN_{a}^{down}\left(concat\left(z_i^t, z_i^e\right)\right) and\ z_{ij}^{r} = FFN_{r}^{down}\left(concat(z_i^e, z_i^e)\right)$$

  - FFN들은 동일한 차원을 갖고있음, pair representation vectors가 2개의 FFN으로 각각 들어가서 event arguments and relations의 positive example이 될 확률 계산

Main Idea

- Inducing Instance Dependency
  - Given the detected event, entity, argument, and relation instances in set of span, JointIE의 인스턴스의 information type을 결정해야 함
  - 두 인스턴스 간의 dependency를 결정할 수 있는 2가지 information 소스(semantic, syntactic information) 고려하여 결정
- Computing Joint Distribution of Labels
  - 기존 연구에서는 instance labels에 대한 독립성을 가정하기 때문에 IE 태스크에 대한 beneficial label dependency를 완벽히 capture 불가능
  - 따라서 Conditional Random Fields를 사용하여 joint distribution를 직접 estimate하여 dependency 향상시켜 예측 성능 향상

Experiments

- Settings

PLMs	Model	ACE05-R		ACE05-E			ACE05-E+				ERE-EN				ERE-ES			
		Ent	Rel	Ent	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg
T5	Text2event	-	-	-	71.9	53.8	-	-	71.8	54.4	-	-	59.4	48.3	-	-	-	-
BART	DEGREE	-	-	-	72.2	56.0	-	-	71.7	58.0	-	-	56.6	51.1	-	-	-	-
BERT	OneIE	88.6	63.4	90.2	74.7	56.8	89.6	58.6	72.8	54.8	87.0	53.2	57.0	46.5	81.3	48.1	56.8	40.3
	AMRIE*	88.7	67.2	90.8	75.3	58.2	90.4	62.9	72.8	56.3	86.9	55.5	58.3	44.2	-	-	-	-
	FourIE	88.9	68.9	**91.3**	75.4	58.0	**91.1**	63.6	73.3	57.5	**87.4**	56.1	57.9	48.6	**82.2**	57.9	57.1	42.3
	GraphIE	**88.9**	**69.5**	90.6	**75.7**	**58.8**	91.0	**65.4**	**74.8**	**59.9**	87.2	**57.8**	**61.4**	**52.2**	81.4	**58.9**	**61.3**	**45.7**
RoBERTa	OneIE*	89.0	65.2	90.2	74.7	55.6	90.8	60.4	72.5	56.3	86.3	52.8	57.1	47.1	83.7	57.5	58.3	42.5
	AMRIE	89.2*	66.8*	**92.1**	75.0	58.6	91.0*	62.8*	72.7*	57.7*	87.9	55.2	61.4	45.0	-	-	-	-
	FourIE*	89.1	67.5	91.6	74.9	58.7	91.1	63.1	72.8	58.3	**88.0**	56.2	61.5	49.1	83.9	61.0	62.3	44.2
	GraphIE	**89.3**	**68.5**	91.4	**75.1**	**59.4**	**91.6**	**66.0**	**73.3**	**60.2**	87.7	**57.0**	**62.0**	**54.7**	**84.3**	**62.3**	**65.7**	**46.9**

  - Generative baselines(Text2event, DEGREE)
    - 태스크를 text generation로 formulating하여 ETD와 EAE 수행
    - 문장이 input으로 들어가면 event triggers and event arguments에 대한 text spans과 labels이 들어있는 text output

  - Classification baselines(OneIE, AMRIE, FourIE)
    - Shared encoder를 통해 ETD, EMR, EAE 및 RE의 인스턴스를 나타내고 태스크 별 레이블 분포를 기반으로 인스턴스에 대한 분류를 수행

Experiments

- Results

PLMs	Model	ACE05-R		ACE05-E			ACE05-E+				ERE-EN				ERE-ES			
		Ent	Rel	Ent	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg
T5	Text2event	-	-	-	71.9	53.8	-	-	71.8	54.4	-	-	59.4	48.3	-	-	-	-
BART	DEGREE	-	-	-	72.2	56.0	-	-	71.7	58.0	-	-	56.6	51.1	-	-	-	-
BERT	OneIE	88.6	63.4	90.2	74.7	56.8	89.6	58.6	72.8	54.8	87.0	53.2	57.0	46.5	81.3	48.1	56.8	40.3
	AMRIE*	88.7	67.2	90.8	75.3	58.2	90.4	62.9	72.8	56.3	86.9	55.5	58.3	44.2	-	-	-	-
	FourIE	88.9	68.9	**91.3**	75.4	58.0	**91.1**	63.6	73.3	57.5	**87.4**	56.1	57.9	48.6	**82.2**	57.9	57.1	42.3
	GraphIE	**88.9**	**69.5**	90.6	**75.7**	**58.8**	91.0	**65.4**	**74.8**	**59.9**	87.2	**57.8**	**61.4**	**52.2**	81.4	**58.9**	**61.3**	**45.7**
RoBERTa	OneIE*	89.0	65.2	90.2	74.7	55.6	90.8	60.4	72.5	56.3	86.3	52.8	57.1	47.1	83.7	57.5	58.3	42.5
	AMRIE	89.2*	66.8*	**92.1**	75.0	58.6	91.0*	62.8*	72.7*	57.7*	87.9	55.2	61.4	45.0	-	-	-	-
	FourIE*	89.1	67.5	91.6	74.9	58.7	91.1	63.1	72.8	58.3	**88.0**	56.2	61.5	49.1	83.9	61.0	62.3	44.2
	GraphIE	**89.3**	**68.5**	91.4	**75.1**	**59.4**	**91.6**	**66.0**	**73.3**	**60.2**	87.7	**57.0**	**62.0**	**54.7**	**84.3**	**62.3**	**65.7**	**46.9**

- Generative baselines은 대부분의 세팅에서 분류 모델보다 성능이 떨어짐
- GraphIE가 대부분 베이스라인보다 성능 뛰어나기 때문에 dependency graph, joint label distribution estimation가 효과적임을 보여줌

NewsEdits: A News Article Revision Dataset and a Document-Level Reasoning Challenge

Motivation

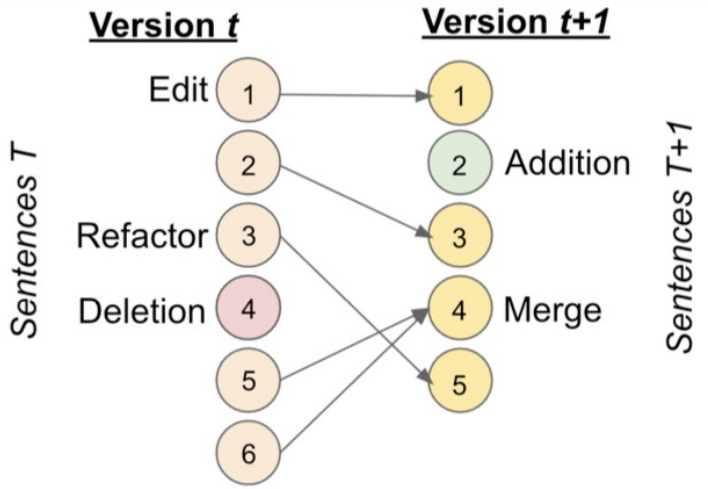
- Novel domain for revision histories
  - Wikipedia나 Wikihow에서의 문서 개정 이력은 주로 grammatical error correction 이나 argumentation design 연구를 위해 사용
  - 그러나 content updates와 narrative evolution에 대한 보다 깊은 연구가 되어 있지 않음
  - 기존의 Existing edits corpora는 연구 domain 특성 상 위와 같은 질문을 다루지 않고 syntax or style edits에 focus
  - 본 논문에서 event update를 다루는 novel domain for revision histories 데이터셋인 NewsEdits 제안
  - NewsEdits를 활용한 새로운 3가지 태스크 설계

Main Idea

- Definition of article-level edit actions
  - 뉴스가 narrative, factual and stylistic development 연구를 위한 중요하고 실용적인 매체라고 주장
    - 뉴스 기사 개정 이력은 narrative와 factual evolution에 대한 단서를 제공
    - 뉴스 주기에서 기사가 업데이트되는 방식에 일관된 패턴이 있다고 가정
  - Article-level edit actions 제안
    - Addition / Deletion /Edit / Refactor

Main Idea

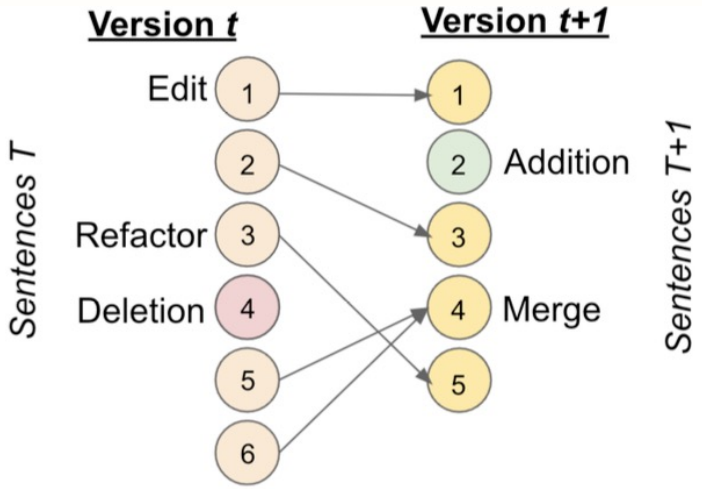
- Edit-Action Operations
  - 전체 뉴스 기사가 버전 간에 어떻게 업데이트 되는지에 관심 있기 때문에 word edits (sentence-level actions)이 아닌 sentence edits (document-level actions)에 focus



  - 두 가지 버전의 뉴스 기사 간의 Addition, Deletion, Edit and Refactor 같은 sentence-level 태스크 identify

Main Idea

- Edit-Action Extraction
  - Edit-actions을 추출하기 위해서 2가지 버전의 뉴스 기사 사이에서 문장을 연결하는 bipartite graph를 구성할 수 있어야 함
  - 버전 간의 문장에 엣지가 존재하면 해당 문장은 Edit (or Unchanged)
  - 엣지가 존재하지 않은 경우는 Addition(새로운 버전에만 문장이 존재) 또는 Deletion(이전 버전에만 문장 존재)



  - 문장 사이의 단방향 유사도를 측정할 수 있도록 word-similarity를 측정하여 두 문장 간의 many-to-many word-matches 수를 세서 sentence-similarity 측정

Main Idea

- Edit-Action Extraction Quality
  - Sentence-similarity algorithm은 un-supervised이지만 하이퍼파라미터(문장이 일치한다고 간주되는 similarity threshold) 세팅하고 다른 알고리즘을 평가하기 위해 ground-truth data 수집해야 함
  - 이를 위해 280 documents에서 수동으로 문장 일치 여부를 식별
  - 2명의 expert annotators에게 문장이 동일하거나, 문장이 동일한 정보를 포함하지만 어체가 다르거나, 의미와 narrative가 실질적으로 중복되는지를 식별하도록 요구

Main Idea

- Task Description
  - Task 1: Will this document update?
    - 기사가 업데이트될지 여부 예측, 기사가 최신 버전이면 1, 아니면 0
  - Task 2: How much will it update?
    - 기사가 주어지면 그 다음 버전에서 얼만큼의 Additions, Deletions, Edits, Refactors이 발생할 지 예측
    - 모델이 각 edit-action category가 기사를 변경하는 방법을 학습하도록 요구
  - Task 3: How will it update?
    - 각 문장에서 Additions, Deletions, Edits, Refactors이 발생할 지 예측
    - 모델이 각 문장의 informational components에 대해 구체적인 reasoning과 기사의 구조와 형식에 대한 뉘앙스를 이해해야 하기 때문에 가장 어려운 태스크

Experiments

- Results

	F1		F1
Most Popular	56.6	Baseline	60.8
Random	50.6	+Partially Frozen	66.0
Human	**80.1**	+Contextual	61.7
		+Version	**77.6**

Task 1 Benchmarks

	Num. Additions		Num. Deletions		Num. Edits		Num. Refactors	
	Mac F1	Mic F1	Mac F1	Mic F1	Mac F1	Mic F1	Mac F1	Mic F1
Most Popular	19.8	25.0	25.6	47.8	21.9	32.0	39.2	64.5
Random	32.5	33.9	30.2	36.4	31.7	35.1	25.8	35.1
Baseline ($n = 30,000$)	22.1	27.9	25.6	46.5	21.4	30.6	35.2	64.5
($n = 150,000$)	29.7	36.3	25.7	48.1	22.4	32.8	39.2	64.6
+Partially Frozen	**52.2**	54.0	44.8	59.0	49.3	53.1	44.3	**65.6**
+Contextual	50.7	52.2	41.0	57.4	**50.8**	**54.8**	**45.0**	64.3
+Version	52.0	**54.5**	**45.3**	**59.8**	49.9	53.7	43.8	63.1
+Multitask	46.7	50.2	28.2	48.4	42.1	49.5	40.3	55.1
Human	**66.4**	**69.3**	**64.6**	**67.5**	**65.9**	**75.6**	**71.3**	**70.7**

Task 2 Benchmarks

	Additions		Sentence Operations		Refactors	
	Above (F1)	Below (F1)	Mac. F1	Mic. F1	Mac. F1	Mic. F1
Most Popular	0.0	0.00	18.1	20.2	34.7	53.3
Random	**11.8**	**14.4**	28.0	38.3	24.7	34.7
Baseline	8.3	0.1	**36.5**	**61.9**	35.2	54.2
+Partially Frozen	3.5	0.0	35.4	60.9	35.4	54.6
+Version	0.1	0.0	30.3	59.0	**41.6**	**57.2**
+Multitask.	0.0	0.0	27.5	57.8	39.5	54.8
Human	**38.6**	**46.7**	**63.8**	**63.5**	**45.6**	**91.5**

Task 3 Benchmarks

- 거의 모든 다운스트림 태스크에서 Random 및 Most Popular보다 성능 높음
- Task 3이 Human에게도 가장 어려운 태스크임을 알 수 있음

Thank you