집현전 2기 중급 7조

What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties

Conneau et al. (2018)

박석종 | 소현지 | 이성인
발표자 : 이성인

2 1 . 0 7 . 2 5

Contents

I. Introduction

- Sentence embedding이 각종 NLP tasks에서 인상깊은 결과를 달성
  - Machine translation, entailment detection 등

- 이로 인하여 universal embeddings에 대한 관심이 증대됨
  - 임베딩 모델을 한 번 훈련시킨 후 다양한 분야에서 활용하는 방식

  ➢ Sentence embedding이 문장의 중요한 언어학적 속성 (linguistic properties)을 포착하는 것으로 보임

I. Introduction

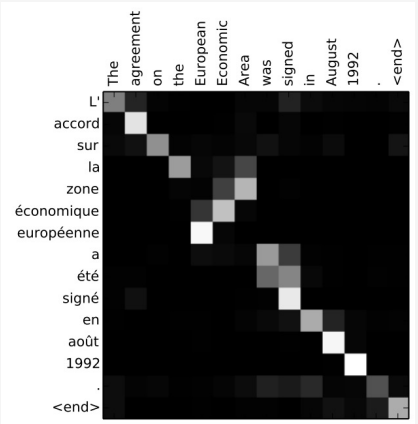
- Downstream task는 복잡한 형태의 추론을 필요로 하므로, 모델이 문장 내의 어떤 정보에 의존하는지를 특정하기 어려움
  - 모델은 일반적으로 임의의 결과 (what) 를 도출할 수 있으나, 근거 (how) 는 도출하지 못함
    - 예시: “A movie that doesn’t aim too high, but it doesn’t need to”라는 문장에 대해서…
      - ‘주관적 관점’을 표현하는 문장임을 구분 가능함 (what)
      - 판단 근거는 알 수 없음 (how)
  - 또한 복잡한 task는 hidden biases를 유도할 수 있음
    - 예시: entailment task에서 명시적으로 부정적인 단어를 찾는 단순한 heuristic을 도입 시 정확도가 상당히 향상됨 (Lai and Hockenmaier (2014))

I. Introduction

- 문장 임베딩의 속성 분석 기법: model introspection
  - 문장 인코더가 문장의 어떤 속성을 유지하는지 알아보고자 함
  - 해당 기법들은 인코더의 아키텍처에 의존적이기 때문에 여러 인코더 간의 비교를 위해 사용하기에는 적합하지 않다는 한계가 존재함

  - 보충설명 (Levy et al (2014), Spring (2020), Christoph et al (2020))
    - 신경망 기반의 word embedding 모델은 기존 sparse vector space representations와 달리 해석 불가능함
    - 모델에 의하여 어떤 종류의 정보가 포착되는지를 분석하기 위한 모든 기법을 총칭하는 말이 model introspection
      - 기존 방법론: visualization, input/hidden representation deletion techniques 등의 방식을 활용하였음

      ➢ 이들은 인코더의 구조에 의존적이므로, general-purpose cross-model evaluation에 활용되기 어려움



I. Introduction

- 문장 임베딩의 **범용적인** 속성 분석 기법: probing tasks
  - Shi et al. (2016) 및 Adi et al. (2017) 에 의하여 제안됨
    - 문장의 단순한 언어학적 속성에 초점을 맞추는 classification problem임
      - 예시: 주 동사의 tense(시제)에 따라 문장 분류하기
    - 특정한 과업 (예: machine translation)에 대하여 사전 훈련된 특정한 인코더 (예: LSTM)가 존재할 때, 이 인코더가 생성하는 sentence embedding을 추가적인 튜닝 없이 시제 분류기에 사용함
    - 분류기가 잘 작동한다면, 사전 훈련된 인코더가 임베딩을 생성할 때 그 안에 읽을 수 있는 시제 정보를 저장한다는 것을 의미함

I. Introduction

- Probing tasks의 3가지 특징
  - 간단한 질문을 기반으로 interpretability 문제를 최소화함
  - Task가 단순하기 때문에 기존 downstream tasks 대비 biases를 제어하기 더 쉬움
  - Encoder architecture agnostic함
    - 문장의 벡터 표현을 생성할 수만 있다면 어떤 인코더에든 적용 가능함

I. Introduction

- 본 논문의 연구
  - Probing tasks에 대한 연구를 크게 확장함
    - 10가지의 probing tasks를 제안함
      - 각 task는 probe 하려는 언어학적 속성의 유형에 따라 구분됨
    - Probing task methodology를 체계화함
      - 가능한 뉘앙스 요인을 통제함
      - 단일 문장 표현만을 입력으로 받도록 모든 task를 프레이밍함
      - 보편성을 최대화하고 결과 해석이 쉽도록 함
    - 제안된 probing task를 활용하여 다양한 SOTA encoder 및 training methods를 탐색하고 Probing and downstream task의 성능을 비교함
    - Probing dataset 및 tools를 공개함
      - 이것들이 문장 임베딩의 linguistic properties를 연구하는 표준 방법이 될 수 있기를 희망함

II. Probing tasks

- Probing benchmarks 구성 시 고려한 4가지 기준
  - ① Generality와 interpretability를 위하여, 모든 classification problem은 single sentence embedding만을 입력으로 요구해야 함
  - ② 관련 속성이 문장 벡터에 비선형적으로 인코딩된 경우에 대비하여, parameter-rich multi-layer 분류기를 학습시킬 대규모 훈련 세트를 구성할 수 있어야 함
  - ③ Lexical cues(어휘 단서) 또는 문장 길이와 같은 뉘앙스 변수들이 통제되어야 함
  - ④ 흥미로운 언어학적 속성을 분석하는 tasks가 필요함

II. Probing tasks

- 앞의 제약을 만족하면서, 광범위한 현상을 탐지할 수 있는 일련의 tasks를 제시하기 위해 노력함
  - 문장의 피상적 속성 (e.g. 문장이 어떤 단어를 포함하는지)부터 문장의 계층 구조, 의미론적 수용 가능성의 미묘한 양상까지
- 이러한 task의 집합이 다양한 언어학적 영역을 상당히 잘 반영하고 있다고 생각함
- 그러나 이것이 완전한 것은 아니며, 향후 연구를 통하여 확장되어야 함

II. Probing tasks

- 모든 tasks에 사용된 문장은 Toronto Book Corpus (Zhu et al., 2015) 로부터 추출된 것임
  - 단어 수가 5~28개인 문장만을 추출함
  - Stanford Parser의 사전 학습된 PCFG를 사용하여 파싱함
  - 각 task에 대하여 train set = 100k 문장, validation set, test set = 10k 문장임
  - 모든 세트는 balanced함 (각 타겟 클래스 별 인스턴스 수는 동일함)

II. Probing tasks

- Tasks의 종류 (총 10개)
  - Surface information
    - SentLen
    - WC
  - Syntactic information
    - Bshift
    - TreeDepth
    - TopConst
  - Semantic information
    - Tense
    - SubjNum
    - ObjNum
    - SOMO
    - CoordInv

II. Probing tasks

- Surface information
  - 문장 임베딩이 문장의 피상적 속성을 얼마나 잘 보존하는지 평가함
  - 단순히 입력 문장의 토큰을 활용하여 분석이 가능하며, 언어학적 지식이 요구되지 않음

  - Task1. SentLen (Length of Sentences): 문장의 길이(문장 내 단어의 수) 예측
    - 문장을 6개의 동일 너비 길이 그룹(bins) 으로 그룹화함
    - 6-way classification task로 취급함

II. Probing tasks

- Surface information
  - Task2. WC (Word Content): 문장 임베딩 벡터로부터 문장의 원래 단어에 대한 정보를 복구할 수 있는지 테스트
    - 코퍼스 내 단어 중 1,000개의 중간 빈도 단어를 선정함
      - 빈도순으로 정렬 시 2,000~3,000위에 해당되는 단어
    - 위 1,000개의 단어 중 1개의 단어만을 포함하는 문장을 샘플링함
    - 1k-way classification task로 취급함
      - 1,000개의 단어 중 해당 문장에 포함된 단어를 예측함
    - 위 과정을 통하여 기존 연구와 달리 보조적인 단어 임베딩을 수행하지 않고, 문장 임베딩만으로 문장 내 단어에 대한 조사를 수행할 수 있음

II. Probing tasks

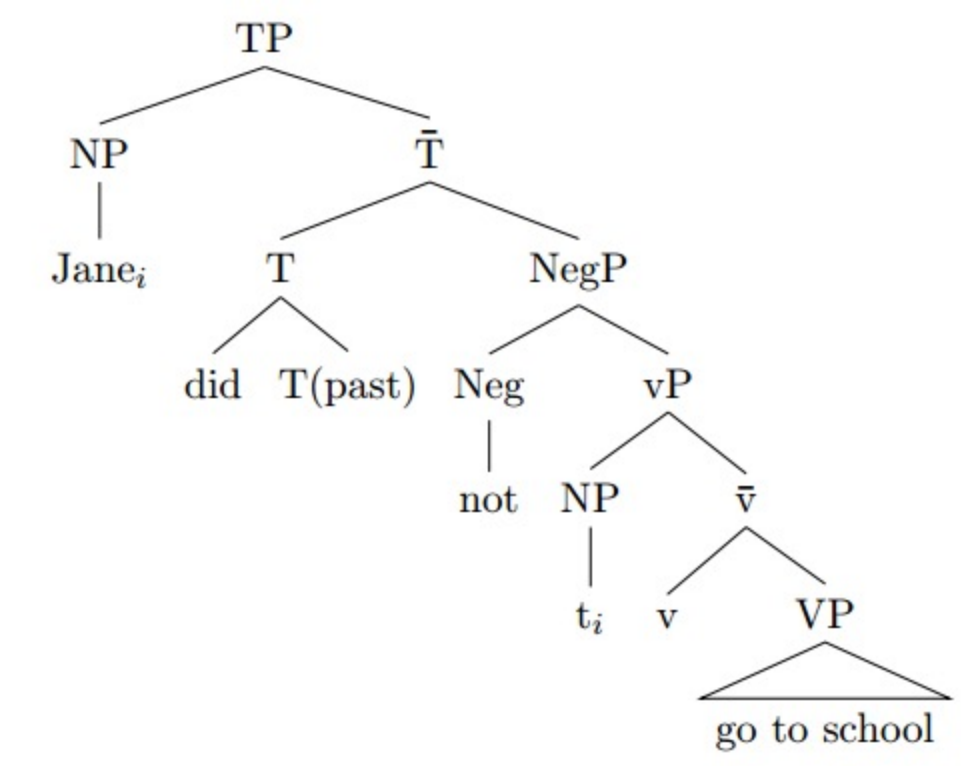
- Syntactic information
  - 문장 임베딩이 문장의 문법적 속성에 민감한지 평가함

  - Task 3. BShift (Bigram Shift): 인코더가 어순에 민감한지 평가
    - 온전한 문장 및 두 개의 무작위 인접 단어의 순서를 교체한 문장을 구별해야 함 (binary classification task로 취급함)
    - 예. “What *you are* doing out there?”

  - Task 4. TreeDepth: 인코더가 문장의 계층적 구조를 추론하는지 평가
    - 루트 노드 ~ 임의의 리프 노드까지의 깊이에 따라 문장을 분류
    - 트리의 깊이는 5~12 값으로 한정함
    - 문장을 트리 깊이에 따라 분류하는 8-way classification task로 취급함
    - 트리의 깊이는 문장의 길이와 연관성이 있음
      - 이러한 연관성을 제거하기 위하여 샘플링을 진행함

II. Probing tasks

- (Syntactic) Tree
  - 문장의 문법 구조를 hierarchical하게 표현한 것
  - 구체적인 표현 방법은 이론에 따라 다르지만 일반적으로 binary tree를 기반으로 표현됨
- Constituent
  - 계층 구조 내에서 하나의 단위로 기능하는 단어 / 단어의 그룹
  - e.g. **Jane** (and John) did not go to school.
    → 'Jane'은 constituent (NP)임
  - e.g. Jane **did not** (and will not) go to school.
    → 'did not'은 constituent (VP)임
  - e.g. Jane did not **do so** (go to school).
    → 'go to school'은 constituent (VP) 임
  - cf. *Jane **did so** (did not go to school).
    → 'did not go to school'은 constituent가 아님



→ **NP(Jane) VP(did not) VP(go to school)**

II. Probing tasks

- Syntactic information
  - Task 5. TopConst (Top Constituent) : 문장을 최상위 constituent의 순서로 분류할 수 있는지 평가
    - 20-way classification task로 취급함
    - 19개의 클래스는 가장 빈번한 순서에 대응, 마지막 클래스는 나머지 전부 다

- Syntactic information probing tasks의 필요성
  - 언어학적으로 훈련받지 않은 인간이 일반적으로 트리 깊이, 최상위 constituent 등을 인식하지는 못함
  - 그러나 문장을 올바르게 파싱하기 위해서는 이와 유사한 정보들이 암시적으로 계산되어야 함
  - 또한 뇌가 문장 처리 중에 트리 깊이와 유사한 정보를 추적한다는 암시적인 증거가 있음

II. Probing tasks

- Semantic Information
  - 앞의 tasks와 동일하게 문법 구조에 의존적이지만, 추가로 문장이 의미하는 바에 대한 이해가 요구되는 과업

  - Task 6. Tense: 주절 동사의 시제에 따른 문장 분류 (현재/과거)
  - Task 7. SubjNum (Subject Number): 주절 내 주어의 수 분류
  - Task 8. ObjNum (Object Number): 주절 내 직접 목적어의 수 분류

  - 분류기가 특정 단어에 의존하지 않도록 train/val/test 세트 간에 중복되는 형태가 없도록 샘플링함

II. Probing tasks

- Semantic Information
  - Task 9. SOMO (Semantic Odd Man Out)
    - 임의의 명사 또는 동사(o) 를 다른 명사 또는 동사(r) 로 교체하여 문장을 수정한 후, 문장이 수정되었는지 여부를 분류 (binary classification task)
    - o 를 포함하는 bigram과 r 을 포함하는 bigram의 빈도가 유사해야 함
    - 원래 문장과 수정된 문장이 모두 데이터셋에 포함된 경우는 없음
    - Train/val/test 세트 간에 동일한 대체가 일어나지 않음
    - 예. “No one could see this Hayes and I wanted to know if it was real or a ***spoonful*** (orig. ***play***).”

II. Probing tasks

- Semantic Information
  - Task 10. CoordInv (Coordination Inversion)
    - 두 개의 등위절로 구성된 문장 데이터셋 중 절반 문장에서 절의 순서를 교체
    - 문장이 수정되었는지 여부를 분류 (binary classification task)
    - 절의 길이는 balanced 함
    - 같은 문장의 원래 버전과 바뀐 버전이 모두 데이터셋에 존재하는 경우는 없음
    - 예. “They might be only memories, but I can still feel each one” →
      “I can still feel each one, but they might be only memories.”

II. Probing tasks

- Human Evaluation
  - 모든 task에 대하여 인간이 판단하기에 합리적인 상한선

  - Surface tasks
    - 인간이 충분한 시간을 들인다면 정답을 찾을 수 있으므로, 상한선이 100%로 지정됨

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Baseline representations										
Majority vote	20.0	0.5	17.9	5.0	50.0	50.0	50.0	50.0	50.0	50.0
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
Length	**100**	0.2	18.1	9.3	50.6	56.5	50.3	50.1	50.2	50.0
NB-uni-tfidf	22.7	**97.8**	24.1	41.9	49.5	77.7	68.9	64.0	38.0	50.5
NB-bi-tfidf	23.0	95.0	24.6	53.0	**63.8**	75.9	69.1	65.4	39.9	**55.7**
BoV-fastText	66.6	91.6	**37.1**	**68.1**	50.8	**89.1**	**82.1**	**79.8**	**54.2**	54.8

II. Probing tasks

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Baseline representations										
Majority vote	20.0	0.5	17.9	5.0	50.0	50.0	50.0	50.0	50.0	50.0
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
Length	**100**	0.2	18.1	9.3	50.6	56.5	50.3	50.1	50.2	50.0
NB-uni-tfidf	22.7	**97.8**	24.1	41.9	49.5	77.7	68.9	64.0	38.0	50.5
NB-bi-tfidf	23.0	95.0	24.6	53.0	**63.8**	75.9	69.1	65.4	39.9	**55.7**
BoV-fastText	66.6	91.6	**37.1**	**68.1**	50.8	**89.1**	**82.1**	**79.8**	**54.2**	54.8

- Human Evaluation
  - Syntactic / Semantic tasks
    - TopConst, Tense, SubjNum, ObjNum
      - 자동화된 PoS 태깅이나 문장 파싱 등이 얼마나 정확한가에 달림
      - ➢ 언어학적으로 훈련된 저자가 각 task에서 무작위로 샘플링 된 테스트 문장 200개의 annotation을 확인함
    - BShift, SOMO, CoordInv
      - 수정된 문장이 수용 가능한 문장일 수 있음
      - 예. ”He pulled out the large round ***onion*** (orig. ***cork***) and saw the amber balm inside.”
      - ➢ Amazon Mechanical Turk 실험 진행: 피험자에게 무작위로 샘플링 된 테스트 문장 1000개의 수용 가능성을 판단케 하여 다수결로 정확도 측정

III. Sentence embedding models

- Sentence encoder architectures (3종)
  - 문장을 고정된 크기의 벡터로 encode하는 구조들을 사용

  - BiLSTM-last/max
    - Bidirectional LSTM 모델
    - T개의 단어로 이뤄진 문장이 입력되었을 때, 이 모델은 T 개의 벡터를 생성
    - Hidden vector $(h_1, \ldots, h_T)$ 가 있을 때 다음과 같은 두 종류의 벡터를 비교함
      - 가장 마지막 벡터 h_T 를 사용 (**last**)
      - 각 차원의 최대값을 사용 (**max**)
  - Gated ConvNet
    - $h_l(X) = (X * W + b) \otimes \sigma(X * V + c)$
    - Convolution의 결과와 sigmoid 계산한 convolution결과를 element-wise 곱
    - LSTM의 gate와 유사한 역할

III. Sentence embedding models

- Training tasks (5종 + 무작위 가중치 1종)
  - 여러 Seq2Seq training task를 통해 model을 학습
    - Seq2Seq 시스템: encoder와 decoder로 구성
    - Encoder를 통해 문장을 고정된 크기의 벡터로 변환
    - Decoder는 조건부 언어 모델로 기능하여 타겟 문장 생성

1. **Neural Machine Translation**
   - 문장 번역 Task
   - 3종의 언어쌍을 사용하여 학습 진행
   - English-French : 유사한 언어
   - English-German : 상대적으로 syntactic 한 차이가 큼
   - English-Finnish : 거리가 먼 쌍

III. Sentence embedding models

- Training tasks
  2. **AutoEncoder**
  3. **Seq2Tree**
     - 문법적 tree구조를 생성하는 Seq2Seq모델
  4. **SkipThought**
     - 주어진 문장으로 다음 문장을 예측하는 task (skip-gram의 문장 확장 형태)
  5. **Natural Language Inference(NLI)**
     - 두 문장 사이의 관계를 추론하는 task
     - 주어진 두 문장을 각각 encode하고 이 두 문장 사이의 관계를 추론하도록 학습
  - **Untrained**
     - 비교를 위해 추가된 학습되지 않은 무작위 가중치

III. Sentence embedding models

- Training details
  - BiLSTM
    - 2 Layers
    - 512 hidden units
  - Gated ConvNet
    - 8 convolution layers
    - 512 hidden units
  - Embedding
    - Pretrained fast-Text (size : 300)
    - Finetuning 없이 사용함

IV. Probing task experiments

- Baselines
  - Length
    - 문장 길이만 이용한 선형 분류기
  - NB-uni-tfidf :
    - tfidf 점수를 사용한 Naive Bayes 분류기
  - NB-bi-tfidf
    - bigram tfidf 점수 사용
  - BoV-fastText
    - fastText 임베딩의 평균

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Baseline representations										
Majority vote	20.0	0.5	17.9	5.0	50.0	50.0	50.0	50.0	50.0	50.0
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
Length	**100**	0.2	18.1	9.3	50.6	56.5	50.3	50.1	50.2	50.0
NB-uni-tfidf	22.7	**97.8**	24.1	41.9	49.5	77.7	68.9	64.0	38.0	50.5
NB-bi-tfidf	23.0	95.0	24.6	53.0	**63.8**	75.9	69.1	65.4	39.9	**55.7**
BoV-fastText	66.6	91.6	**37.1**	**68.1**	50.8	**89.1**	**82.1**	**79.8**	**54.2**	54.8
BiLSTM-last encoder										
Untrained	36.7	43.8	28.5	76.3	49.8	84.9	84.7	74.7	51.1	64.3
AutoEncoder	**99.3**	23.3	35.6	78.2	62.0	84.3	84.7	82.1	49.9	65.1
NMT En-Fr	83.5	**55.6**	42.4	81.6	62.3	88.1	89.7	89.5	52.0	71.2
NMT En-De	83.8	53.1	42.1	81.8	60.6	88.6	89.3	87.3	51.5	**71.3**
NMT En-Fi	82.4	52.6	40.8	81.3	58.8	88.4	86.8	85.3	52.1	71.0
Seq2Tree	94.0	14.0	**59.6**	**89.4**	**78.6**	**89.9**	**94.4**	**94.7**	49.6	67.8
SkipThought	68.1	35.9	33.5	75.4	60.1	89.1	80.5	77.1	**55.6**	67.7
NLI	75.9	47.3	32.7	70.5	54.5	79.7	79.3	71.3	53.3	66.5
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8
GatedConvNet encoder										
Untrained	90.3	17.1	30.3	47.5	62.0	78.2	72.2	70.9	61.4	59.6
AutoEncoder	**99.4**	16.8	46.3	75.2	71.9	87.7	88.5	86.5	**73.5**	72.4
NMT En-Fr	84.8	41.3	44.6	77.6	67.9	87.9	88.8	86.6	66.1	72.0
NMT En-De	89.6	49.0	50.5	81.7	72.3	90.4	91.4	89.7	72.8	**75.1**
NMT En-Fi	89.3	**51.5**	49.6	81.8	70.9	90.4	90.9	89.4	72.4	**75.1**
Seq2Tree	96.5	8.7	**62.0**	**88.9**	**83.6**	**91.5**	**94.5**	**94.3**	**73.5**	73.8
SkipThought	79.1	48.4	45.7	79.2	73.4	90.7	86.6	81.7	72.4	72.3
NLI	73.8	29.2	43.2	63.9	70.7	81.3	77.5	74.4	73.3	71.0

Table 2: **Probing task accuracies.** Classification performed by a MLP with sigmoid nonlinearity, taking pre-learned sentence embeddings as input (see Appendix for details and logistic regression results).

IV. Probing task experiments

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Baseline representations										
Majority vote	20.0	0.5	17.9	5.0	50.0	50.0	50.0	50.0	50.0	50.0
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
Length	**100**	0.2	18.1	9.3	50.6	56.5	50.3	50.1	50.2	50.0
NB-uni-tfidf	22.7	**97.8**	24.1	41.9	49.5	77.7	68.9	64.0	38.0	50.5
NB-bi-tfidf	23.0	95.0	24.6	53.0	**63.8**	75.9	69.1	65.4	39.9	**55.7**
BoV-fastText	66.6	91.6	**37.1**	**68.1**	50.8	**89.1**	**82.1**	**79.8**	**54.2**	54.8

- Baselines
  - NB-uni-tfidf
    - 문장의 단어 분포만으로 해결할 수 있는 task가 무엇인지 보여줌
    - Tense, SubjNum, ObjNum등에서 상대적으로 높은 성능
    - Bigram과 큰 차이가 없음 (BShift는 task 특성상 당연히 bigram에서 높음)
  - BoV-fastText
    - 전체적으로 좋은 성능을 보여줌
    - Aggregated word embedding이 문장에 대해 놀랍도록 많은 정보를 추출해 낸다는 기존 연구 결과를 입증함
    - 단어의 순서를 고려하지 않는 특성상 Bsift, SOMO, CoordInv에 대해서는 무작위에 가까움 (50)

IV. Probing task experiments

- Encoding architectures
  - 적절한 encoding 구조가 BoV보다 좋음
  - 다른 encoder가 같은 training task를 통해 학습되어 유사한 성능을 보이더라도, 언어학적으로 다른 embedding을 만들어낼 수 있음
  - Gated ConvNet의 전체적인 성능이 BiLSTM의 가장 좋은 성능과 비슷함
  - BiLSTM-max가 BiLSTM-last보다 좋은 성능

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Baseline representations										
Majority vote	20.0	0.5	17.9	5.0	50.0	50.0	50.0	50.0	50.0	50.0
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
Length	**100**	0.2	18.1	9.3	50.6	56.5	50.3	50.1	50.2	50.0
NB-uni-tfidf	22.7	**97.8**	24.1	41.9	49.5	77.7	68.9	64.0	38.0	50.5
NB-bi-tfidf	23.0	95.0	24.6	53.0	**63.8**	75.9	69.1	65.4	39.9	**55.7**
BoV-fastText	66.6	91.6	**37.1**	**68.1**	50.8	**89.1**	**82.1**	**79.8**	**54.2**	54.8
BiLSTM-last encoder										
Untrained	36.7	43.8	28.5	76.3	49.8	84.9	84.7	74.7	51.1	64.3
AutoEncoder	**99.3**	23.3	35.6	78.2	62.0	84.3	84.7	82.1	49.9	65.1
NMT En-Fr	83.5	**55.6**	42.4	81.6	62.3	88.1	89.7	89.5	52.0	71.2
NMT En-De	83.8	53.1	42.1	81.8	60.6	88.6	89.3	87.3	51.5	**71.3**
NMT En-Fi	82.4	52.6	40.8	81.3	58.8	88.4	86.8	85.3	52.1	71.0
Seq2Tree	94.0	14.0	**59.6**	**89.4**	**78.6**	**89.9**	**94.4**	**94.7**	49.6	67.8
SkipThought	68.1	35.9	33.5	75.4	60.1	89.1	80.5	77.1	**55.6**	67.7
NLI	75.9	47.3	32.7	70.5	54.5	79.7	79.3	71.3	53.3	66.5
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8
GatedConvNet encoder										
Untrained	90.3	17.1	30.3	47.5	62.0	78.2	72.2	70.9	61.4	59.6
AutoEncoder	**99.4**	16.8	46.3	75.2	71.9	87.7	88.5	86.5	**73.5**	72.4
NMT En-Fr	84.8	41.3	44.6	77.6	67.9	87.9	88.8	86.6	66.1	72.0
NMT En-De	89.6	49.0	50.5	81.7	72.3	90.4	91.4	89.7	72.8	**75.1**
NMT En-Fi	89.3	**51.5**	49.6	81.8	70.9	90.4	90.9	89.4	72.4	**75.1**
Seq2Tree	96.5	8.7	**62.0**	**88.9**	**83.6**	**91.5**	**94.5**	**94.3**	**73.5**	73.8
SkipThought	79.1	48.4	45.7	79.2	73.4	90.7	86.6	81.7	72.4	72.3
NLI	73.8	29.2	43.2	63.9	70.7	81.3	77.5	74.4	73.3	71.0

Table 2: **Probing task accuracies.** Classification performed by a MLP with sigmoid nonlinearity, taking pre-learned sentence embeddings as input (see Appendix for details and logistic regression results).

IV. Probing task experiments

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8

- Training tasks
  - BiLSTM-max를 기준으로, 각 학습 task의 차이
  - NMT(Neural Machine Translation) vs. NLI(Natural Language Inference)
    - Downstream task에서는 NLI가 성능이 더 좋은데, probing task는 반대
    - NLI가 NMT에 비해 확실히 더 좋은 성능을 발휘하는 task는 WC(Word Content)
    - Downstream task에 필요한 문법적 특성은 더 '얕은' 특성일지도…
  - 비지도학습 (SkipThoughts, AutoEncoder)
    - 지도학습에 비해 성능이 떨어졌지만 여전히 효과적임
    - AutoEncoder의 경우 SentLen에는 좋은 성능 / WC에서는 낮은 성능

IV. Probing task experiments

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8

- Training tasks
  - Seq2Tree
    - 일부 probing task를 만드는 데 사용된 parser와 동일한 parser를 사용
    - 데이터셋의 parser 오류를 모방하여 학습, 몇몇 task에서 human bound를 넘어섬
    - Tagging/parser 정보에 간접적으로만 관계가 있는 task (SOMO, CoordInv) 에서는 NMT와 유사한 성능

IV. Probing task experiments

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8

- Training tasks
  - Untrained BiLSTM-max
    - 학습 없이도 좋은 결과 (downstream task도)
    - Fasttext embedding과 위치 정보를 잘 활용함
    - 내부 구조적인 bias도 존재
      - SOMO에서 인간은 acceptable쪽으로, 모델은 modified쪽으로 bias
      - 예. "I didn't come to ***reunite*** (orig. ***undermine***) you"

IV. Probing task experiments

Task	SentLen	WC	TreeDepth	TopConst	BShift	Tense	SubjNum	ObjNum	SOMO	CoordInv
Hum. Eval.	100	100	84.0	84.0	98.0	85.0	88.0	86.5	81.2	85.0
BiLSTM-max encoder										
Untrained	73.3	**88.8**	46.2	71.8	70.6	89.2	85.8	81.9	73.3	68.3
AutoEncoder	**99.1**	17.5	45.5	74.9	71.9	86.4	87.0	83.5	73.4	71.7
NMT En-Fr	80.1	58.3	51.7	81.9	73.7	89.5	90.3	89.1	73.2	75.4
NMT En-De	79.9	56.0	52.3	82.2	72.1	90.5	90.9	89.5	73.4	**76.2**
NMT En-Fi	78.5	58.3	50.9	82.5	71.7	90.0	90.3	88.0	73.2	75.4
Seq2Tree	93.3	10.3	**63.8**	**89.6**	**82.1**	**90.9**	**95.1**	**95.1**	73.2	71.9
SkipThought	66.0	35.7	44.6	72.5	73.8	90.3	85.0	80.6	**73.6**	71.0
NLI	71.7	87.3	41.6	70.5	65.1	86.7	80.7	80.3	62.1	66.8

- Probing task comparision
  - NMT 훈련된 BiLSTM-max와 같은 좋은 encoder는 일반적으로 모든 probing task에 대해 좋은 성능을 보임
    - Surface tasks에 대해 특별히 높은 성능을 보이지 않음
      - 인코더가 '더 깊은' 언어학적 속성을 추출한다는 간접적인 증거일 수 있음
    - TreeDepth나 BShift, SOMO, CoordInv와 같은 task는 떨어지는 성능
      - 뒤의 3종은 문장에 대한 통사적/의미적 이상 탐지를 하는 task라서 acceptability에 대해 직접적으로 학습되지 않은 모델은 구분이 어려움
      - 최고의 모델들은 직접적 학습 없이 어느정도 합리적인 이상 판단이 가능
    - 어려운 TreeDepth, 쉬운 TopConst
      - TreeDepth는 좀 더 미묘한 통사적 정보가 필요하기 때문에?

IV. Probing task experiments

- Probing task comparision
  - Figure 1 : Epoch에 따른 probing task의 정확도 변화
    - NMT의 probing 성능은 언어에 독립적임
      - BLEU의 차이가 있음에도 probing score는 유사함
    - WC성능은 epoch에 따라서 계속 증가, 나머지는 downstream task 성능의 개선이 있는 와중에도 flat해짐
    - 특히 SentLen은 epoch증가에 따라 감소함
      - 모델이 문장의 깊은 언어학적 속성을 배우면 표면적인 부분에 대해선 잊어버리는 경향이 있음
    - SOMO task는 전체적으로 flat
      - 학습을 통해서는 개선이 없고 모델 구조적인 부분만 영향

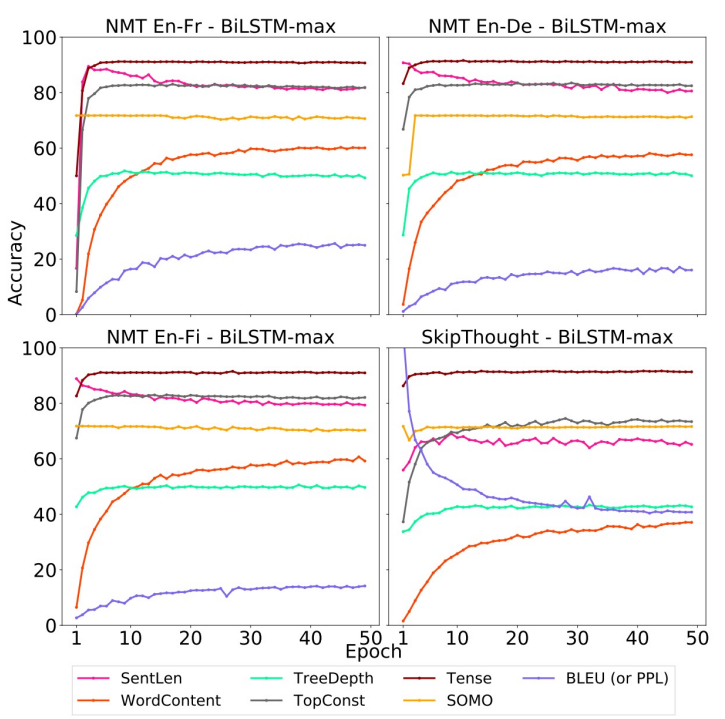


Figure 1: **Probing task scores after each training epoch, for NMT and SkipThought.** We also report training score evolution: BLEU for NMT; perplexity (PPL) for SkipThought.

IV. Probing task experiments

- Probing vs. downstream task
  - SentEval(Conneau and Kiela, 2018)
    - 문장 임베딩(universal sentence representations)를 평가하는 toolkit.
    - 논문 시점에선 아래와 같은 downstream task를 이용하여 평가
    - 현재는 이 논문에서 나온 probing task도 추가됨

    - MR: Sentiment prediction (movie reviews)
    - CR: Sentiment prediction (customer product reviews)
    - SUBJ: Subjectivity prediction of sentences from movie reviews and plot summaries
    - MPQA: Phrase level opinion polarity classification from newswire
    - SST: Stanford Sentiment Treebank with binary labels
    - TREC: Fine grained question-type classification from TREC
    - MRPC: Microsoft Research Paraphrase Corpus from parallel news sources
    - SICK-E: SICK(Sentences Involving Compositional Knowledge) dataset for entailment
    - SICK-R: SICK dataset for semantic textual similarity
    - STS: Semantic textual similarity

IV. Probing task experiments

- Probing vs. downstream task
  - Figure 2 : probing – downstream 상관관계
    - WC는 모든 downstream task와 양의 상관관계
      - 최소한 현재 살펴보는 모델에서는 데이터에 대해 단어 이상의 특별한 추상적 지식이 필요하지 않음
    - SentLen과 downstream task와 음의 상관관계
      - 모델이 문장에 대한 깊은 지식을 배울수록 문장의 길이 같은 표면적 지식을 잃어버림
    - SOMO, CoordInv
      - 복잡한 의미론적 지식이 필요한 task
      - WC 다음으로 많은 downstream task와 양의 상관관계

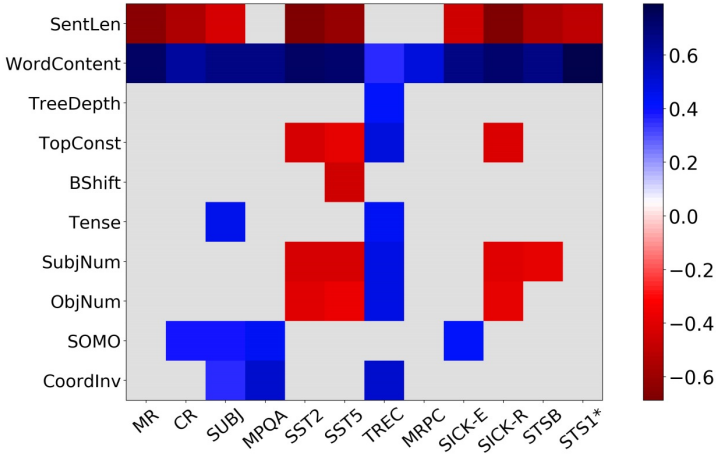


Figure 2: **Spearman correlation matrix between probing and downstream tasks.** Correlations based on all sentence embeddings we investigated (more than 40). Cells in gray denote task pairs that are not significantly correlated (after correcting for multiple comparisons).

IV. Probing task experiments

- Probing vs. downstream task
  - Figure 2 : probing – downstream 상관관계
    - SOMO
      - SICK-E와 높은 상관관계 SICK-R과 낮은 상관관계
        - SICK-E : NLI / SICK-R : semantic relatedness
      - 논리적 함의 분석이 단순 유사도 판단보다 더 깊은 의미 파악을 요구
    - TopConst와 number task들
      - 일부 task(유사도, 감성분석)와 음의 상관관계
      - 이는 이 데이터셋에 bias가 있을 수 있다는 사실을 의미
    - TREC이 많은 probing task와 양의 상관관계

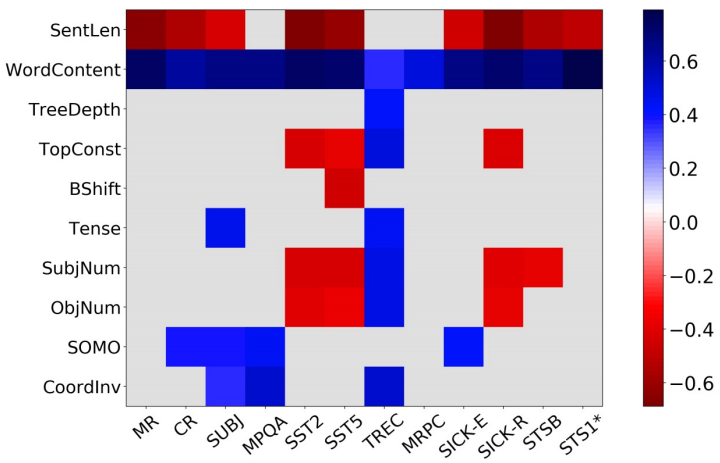


Figure 2: **Spearman correlation matrix between probing and downstream tasks.** Correlations based on all sentence embeddings we investigated (more than 40). Cells in gray denote task pairs that are not significantly correlated (after correcting for multiple comparisons).

V. Conclusion

- 문장 임베딩 방법의 언어학적 지식을 probe할 수 있는 일련의 task 소개
- 최신 문장 인코더의 언어학적 평가를 실시하여 인코더가 baseline (uni/bigram, Bag-of-Vectors) 이상으로 다양한 언어학적 속성을 포착하고 있다는 것을 밝혀냄
- Probing task와 복잡한 downstream task 간의 연관성 패턴을 밝혀내고 다양한 임베딩 방법의 언어학적 속성에 대한 흥미로운 발견을 제시함
  - 예. Bag-of-Vectors가 문장 레벨의 속성을 포착하는 데 놀랍도록 좋음
- 동일한 목적 하에 학습한 다른 인코더 구조가 비슷한 성능을 내더라도 임베딩은 다를 수 있다 → 문장 임베딩에 있어 아키텍처의 중요성 지적

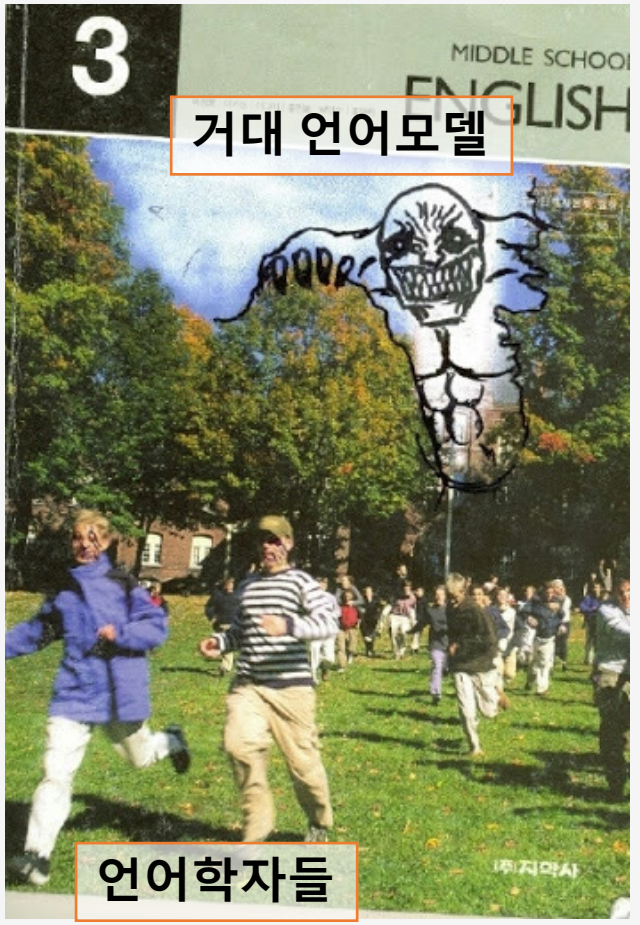
V. Conclusion

- 특히 BiLSTM-max 임베딩이 학습 전에도 이미 흥미로운 언어학적 지식을 포착하고 있으며, 학습 후에는 anomaly에 노출된 적이 없어도 의미론적 수용 가능성을 평가할 수 있다는 점을 발견함
- 우리의 public하게 공개된 task set이 새로운 인코더의 언어학적 속성을 평가할 수 있는 벤치마킹 툴이 되고, 이것이 인코더가 무엇을 학습하는가에 대한 더 나은 이해로 나아가기를 희망함
- In future works…
  - Probing task를 다른 언어로 확장
  - Multi-task training이 probing task에 어떻게 영향을 미치는지 조사
  - 더 linguistically-aware한 보편 인코더를 찾기 위해 probing task 활용

VI. 의의 및 한계

- 의의
  - 빅 데이터 + 빅 모델의 강세로 NLP 분야에서 언어학 및 언어학자의 입지가 점점 좁아지고 있는 상황
  - 언어학적 지식을 가지고 대형 언어 모델을 개선할 여지가 있을 수도 있지 않을까 하는 몸부림(으로 보임)

  - Explainable AI를 향한 움직임
    - 거대한 '블랙박스' 언어모델이 언어에 대해서 어떠한 정보를 저장하고 있는지 알아내려는 노력



VI. 의의 및 한계

- 한계
  - Probing task의 성능과 downstream task의 성능 간에 의미 있는 상관 관계를 밝혀내지 못함
    - 실제로 WC(Word Content)를 제외한 다른 probing task는 epoch가 증가하고 downstream task의 성능이 상승하는 동안에도 성능 변화가 없었음
    - 저자 본인들도 언급했지만, ‘downstream task가 데이터에 대해 단어 이상의 특별히 추상적인 지식을 필요로 하지 않는 것 같음’

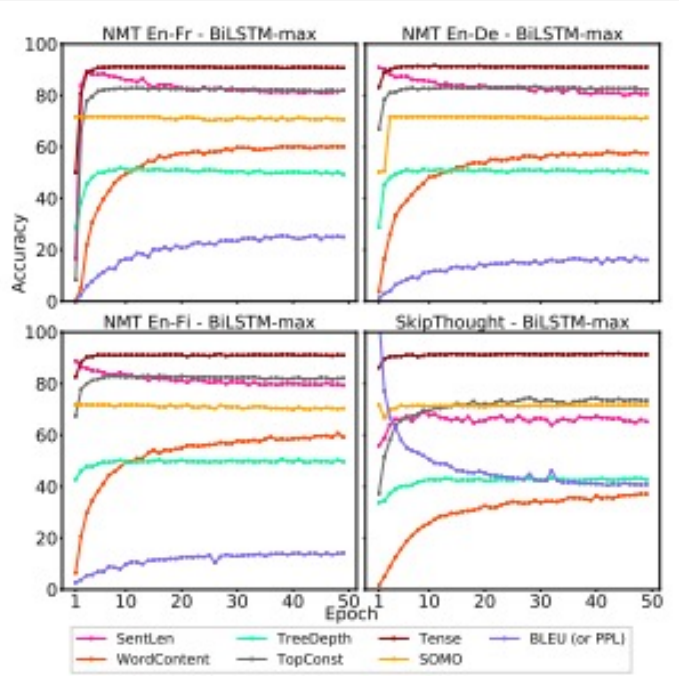


Figure 1: **Probing task scores after each training epoch, for NMT and SkipThought.** We also report training score evolution: BLEU for NMT; perplexity (PPL) for SkipThought.

감사합니다!