Augmented-SBERT:

Data Augmentation Method for Improving Bi-Encoders for Pairwise Sentence Scoring Tasks

Keyword
Data Augmentation, Domain Adaptation, Sentence Pair Modeling, SBERT, BI-ENCODER

집현전
민지웅

CONTENTS

01 Contribution

- 두 문장을 입력으로 받아 하나의 값을 반환하는 문제(STS, NLI task 등)에서 BERT는 Cross-Encoder 방식을 채택

- Cross-Encoder 방식은 연산양이 너무 많아 검색 등의 실시간성 작업에 적용이 어려움

- 이러한 문제를 극복하기 위해 Bi-Encoder를 사용한 Sentence-BERT(SBERT) 도입

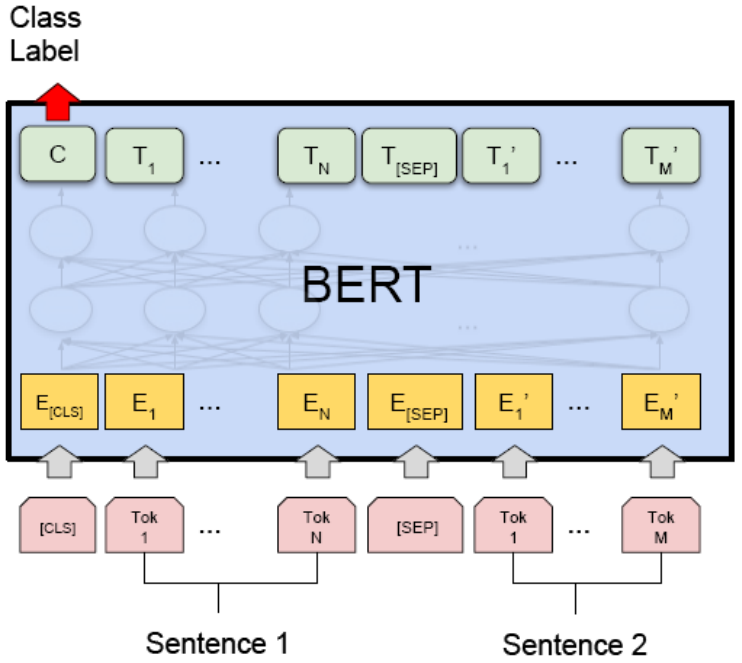
- SBERT는 연산 속도는 빠르지만 Cross-Encoder 5~10% 정도 성능이 떨어짐

- **논문에서는 Data-Augmentation 활용하여 AugSBERT 제시**
  Data-Augmentation을 적용한 데이터를 이용해 Silver Dataset을 만들고 이를 Gold Dataset과 함께 이용하면 in-Domain에서는 6%p, domain adaptation에서는 37%p까지 성능 향상

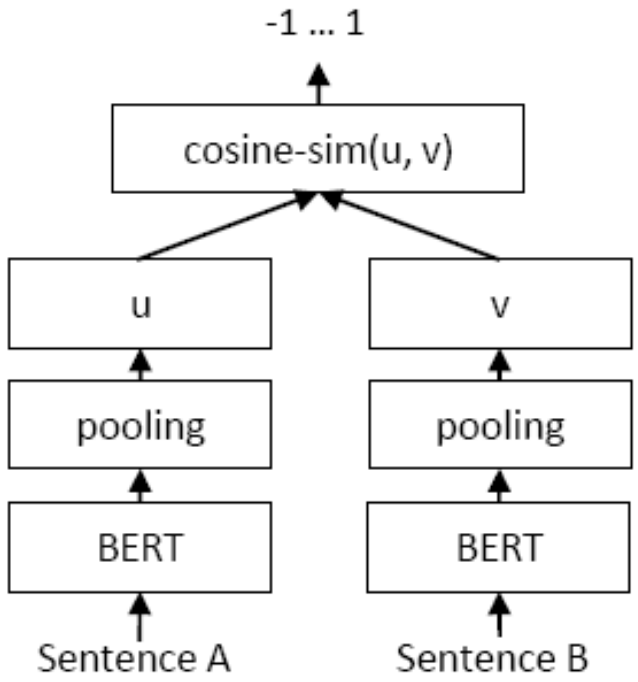
02 Introduction

- Pairwise sentence scoring은 information, retrieval, question answering, duplicate question detection, clustering 등에 활용
- BERT는 [SEP] 토큰을 활용한 Cross-encoder 방식을 채택
  Cross-encoder 방식의 장점은 두 문장이 Self-attention을 통해 상호 간 attend 될 수 있으므로 두 문장 사이의 관계를 면밀하게 파악할 수 있음
- 하지만 연산양이 많다는 한계
  ex) 검색 엔진에 10,000개 문서를 대상으로 한 서치 쿼리를 보낸다고 할 때 서치 쿼리와 유사한 Top-k 문서를 찾기 위해 우리는 "[CLS] 서치 쿼리 [SEP] n-th 문서 [SEP]"의 임베딩을 10,000 번의 Cross-encoder 연산을 통해 구해야함
- 이러한 문제를 극복하기 위해 나온 것이 UKP Lab에서 내놓았던 Sentence BERT (SBERT)

02 Introduction



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

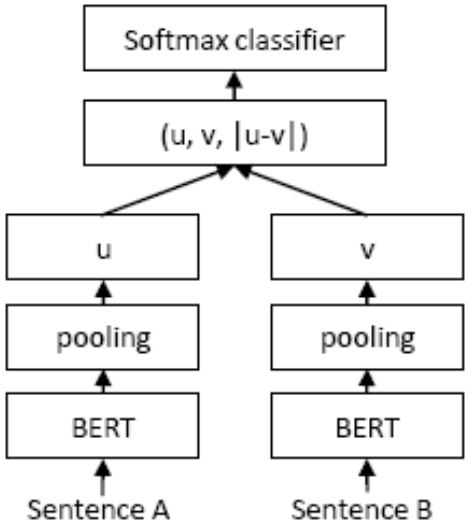


BERT **SBERT**

02 Introduction

SBERT

- SBERT는 Cross-Encoder가 아닌 Bi-Encoder 구조를 채택
- Siamese Network를 이용하여 BERT를 학습
- 각 문장이 개별의 BERT Layer를 타고 나온 결과를 이용하여 문장간 score 계산
- Ex) 앞의 사례와 동일하게 10000개의 문서에 대한 서치를 진행할 때 SBERT 방식에서는 각 문장의 임베딩 연산이 독립적으로 수행되고 이렇게 해서 얻어진 임베딩을 기준으로 유사도 계산. 임베딩은 미리 저장해둘 수 있음



SBERT 파인튜닝 을 위한 목적함수

02 Introduction

Bi-Encoder vs. Cross-Encoder

- Bi-Encoder가 Cross-Encoder에 비해 월등히 빠른 속도를 가지고 있지만 Cross-Encoder에 비해 5~10% 정도의 성능 손실이 발생

- 그래서 이러한 문제를 극복하기 위해 Data-Augmentation 방법을 도입
- BERT Cross-Encoder를 활용해 SBERT Bi-Encoder 성능을 끌어올림

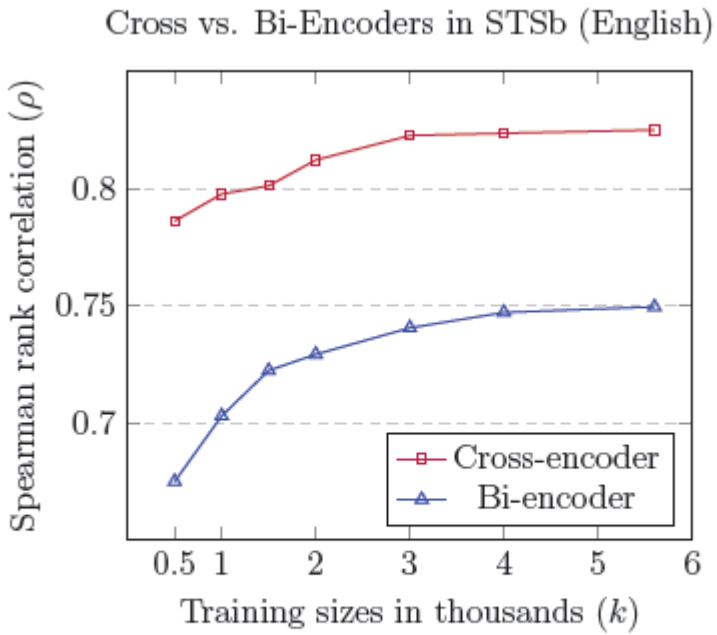


Figure 1: Spearman rank correlation (ρ) test scores for different STS Benchmark (English) training sizes.

03. Methods

3.1 Augmented-SBERT

- Sampling Strategy에 따라 학습에 사용할 sample을 뽑은 후 Cross-Encoder를 활용해 label 진행. 이를 silver dataset이라고 부름
- 실버 데이터셋과 원래의 gold dataset을 이용해 Bi-Encoder를 학습. 이렇게 나온 모델이 **Augmented-SBERT**

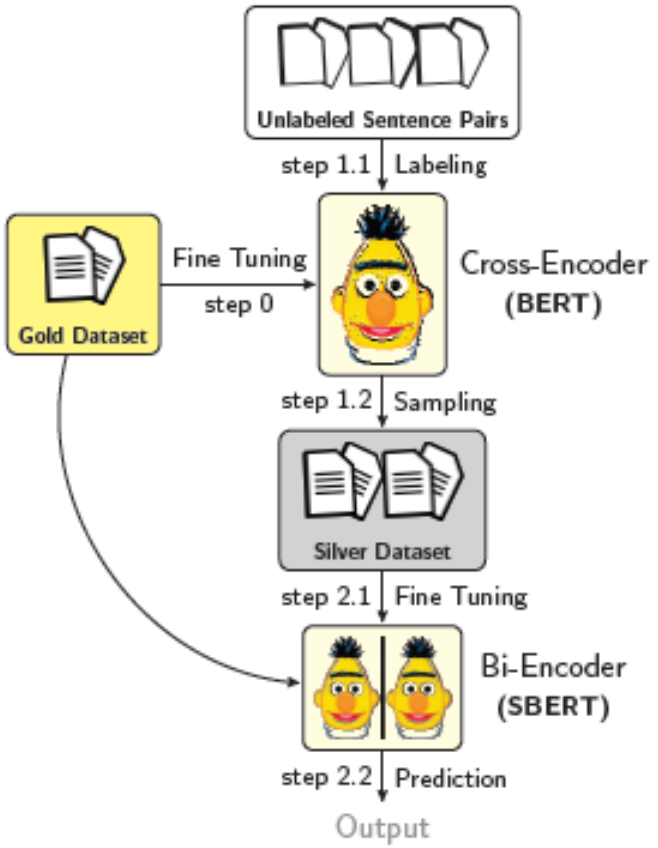


Figure 2: Augmented SBERT In-domain approach

03. Methods

Pair Sampling Strategies

- Un-labeled 페어 데이터를 성능이 좋은 Cross-encoder 형태의 BERT로 Weakly-labeled 데이터로 변환 후, 이를 Gold 데이터셋에 증강해 훈련에 활용하는 구조
- Un-labeled 데이터셋은 임의의 문장 두 쌍으로 구성된 아무 데이터셋으로 설정할 수도 있지만 이는 엄청나게 많은 데이터가 만들어지는 문제점이 발생
- 논문에서는 전체 Un-labeled 데이터를 라벨링 하는 것이 아닌, 여러 샘플링 기법을 활용

Sampling Strategies

1) Random Sampling: 랜덤하게 두 문장을 선택. 이 경우 dissimilar pair가 압도적으로 많아짐

2) Kernel Density Estimation (KDE)

Classification: 긍정으로 분류된 데이터는 남겨두고 부정으로 분류된 데이터를 임의로 제거

Regression: kernel density estimation을 이용해 실버 데이터셋과 골드 데이터셋의 확률 밀도 함수의 KL Divergence 최소화

03. Methods

Pair Sampling Strategies

3) BM25

Un-labeled 데이터셋에서 문장 하나를 뽑아 다른 문장들을 대상으로 BM 25 알고리즘을 적용해 Top-k 개 문장을 추출하고, 추출된 문장들을 페어로 설정해 Cross-encoder 연산을 통해 Weakly-labeled 데이터로 활용

4) Semantic Search Sampling

- BM 25를 활용한 샘플링은 Lexical overlapped 문장들을 대상으로만 라벨링이 수행되기 때문에 Lexcially-overlapped 되지는 않았지만 의미론적으로 유사한 문장들 간 라벨링은 전혀 수행되지 않는다는 단점
- 문제를 해결하기 위해 기학습된 SBERT를 활용해 시맨틱 유사도가 높은 Top-k 개 문장을 추출해, 추출된 문장들을 페어로 설정해 Weakly-labeled 데이터로 활용

03. Methods

Seed Optimization

- BERT와 같은 트랜스포머 기반의 모델은 랜덤 시드에 크게 의존
- 시드가 달라지면 다른 minima에 수렴
- 특히 학습 데이터의 수가 적을 때 강하게 나타나는 경향
- 이를 극복하기 위해서 Seed optimization 도입
- 5개의 랜덤시드에 대하여 학습을 진행하고 성능이 가장 좋은 모델을 선택
- 20% 학습이 되면 early-stop을 진행해서 속도를 올림. Best model보다 성능이 좋은 경우만 진행

03. Methods

3.2 Domain Adaptation

- 동일한 도메인 뿐만 아니라 다른 도메인에서도 높은 성능이 나오길 기대
- 하지만 SBERT는 학습에 사용되지 않았던 용어들에 대해서는 제대로 매핑을 하기 어려웠지만 Augmented-SBERT에서는 좋은 성능을 내주기를 기대
- 이를 위해서 source domain에 Cross-Encoder를 fine-tuning하고 target domain에 대해서 라벨링. 라벨링된 데이터를 이용하여 SBERT를 학습해서 성능을 확인

04. Datasets

Single-Domain Datasets

Dataset	Spanish-STS	BWS (cross-topic)	BWS (in-topic)	Quora-QP	MRPC
# training-samples	1,400	2125	2471	10,000	4,340
# development-samples	220	425	478	3,000	731
# testing-samples	250	850	451	3,000	730
# total-samples	**1,870**	**3,400**	**3,400**	**16,000**	**5,801**

Table 1: Summary of all datasets being used for diverse in-domain sentence pair tasks in this paper.

1) SemEval Spanish STS

- Semantic Textual Similarity task는 두 문장의 유사도를 0~5점으로 나타내는 task
- training and development dataset

  뉴스 기사와 위키피디아에서 수집된 SemEval STS 2014, SemEval STS 2015
- Testset

  SNLI에서 나온 이미지 캡션 문장 쌍 SemEval STS 2017
- 0~1로 표준화

2) BWS Argument Similarity Dataset

- 8개의 주제에 대한 찬반 주장이 있는 데이터셋을 이용해 0~1 사이의 주장 유사도
- Cross-topic sampling: T1~T5를 train, T6를 Develop, T7~8을 testset으로 사용
- In-topic sampling: T1~T8을 각각의 주제의 데이터 수에 비례하여 train, dev, test로 분할

04. Datasets

Single-Domain Datasets

3) Quora Question Pairs

- 두 질문 동일 질문 여부
- 404290 pairs -> 10000 pairs로 down sampling

4) Microsoft Research Paraphrase Corpus (MRPC)

- 온라인 뉴스에서 추출한 두 문장 동일 여부

Dataset	Sentence 1	Sentence 2	Score
BWS	Cloning treats children as objects.	It encourages parents to regard their children as property.	0.89
Quora-QP	How does one cook broccoli?	What are the best ways to cook broccoli?	1
Spanish-STS	Dos hombres en trajes rojos practicando artes marciales.	Dos hombre en uniformes de artes marciales entrenando.	0.80
MRPC	The DVD-CCA then appealed to the state Supreme Court.	DVD CCA appealed that decision to the U.S. Supreme Court.	1

Table 2: Dataset examples for our in-domain tasks. We report the normalized similarity score $[0, 1]$ for regression tasks and the binary label $\{0, 1\}$ for classification tasks.

04. Datasets

Multi-Domain Datasets

- Multiple domains 데이터셋을 이용한 **duplicate question detection**(동일 질문 탐지) tasks 데이터셋 활용

AskUbuntu, SuperUser from Stack Exchange(technical community support forums)

Sprint FAQ from Sprint technical forum website

Quora dataset from Quora website

Dataset k	Train / Dev / Test (Total Pairs)	Train (Ratio)	Dev / Test (Ratio)
AskUbuntu	919706 / 101k / 101k	1 : 100	1 : 100
Quora	254142 / 10k / 10k	3.71 : 100	1 : 1
Sprint	919100 / 101k / 101k	1 : 100	1 : 100
SuperUser	919706 / 101k / 101k	1 : 100	1 : 100

Table 3: Summary of multi-domain datasets originally proposed by Shah et al. (2018) and used for our domain adaptation experiments. Ratio denotes the duplicate pairs (positives) vs. not duplicate pairs (negatives).

05. Experimental Setup

- Huggingface's Transformers, sentence-transformers framework
- English datasets: bert-base-uncased
- Spanish dataset: bert-base-multilingual-cased

Cross-encoders (fine-tune)

- LR=1e-5, hidden-layer sizes=200 or 400, batch-size = 16.

Bi-encoders(fine-tune)

- batch-size = 16, a fixed learning rate = 2e-6,

BM25: top k = 3 or 5

Evaluation: 10 different random Seeds

in-domain regression tasks(STS and BWS): Spearman's rank

in-domain classification: F1 score of the positive label

Baselines

Universal Sentence Encoder (USE)

NLPAug: replaces words in sentences with synonyms

BERT model	bert-base (uncased/multi.-cased)
hidden layer sizes	{100, 200, 400, 800, 1600, 3200}
Learning rates	{1e-4, 1e-5, 1e-6}
Batch sizes	{8, 16}

Table 7: Experimental setup for hyperparameter tuning of cross-encoder (BERT).

BERT model	bert-base (uncased/ multi.-cased)
Learning rates	{2e-5, 1e-6, 1e-7}
Learning rate scheduler	constant
Batch sizes	{8, 16}

Table 8: Experimental setup for hyperparameter tuning of bi-encoder (SBERT).

06. Results

Task		Regression ($\rho \times 100$)			Classification (F_1)	
Model / Dataset	(Seed Opt.)	**Spanish-STS**	**BWS (cross-topic)**	**BWS (in-topic)**	**Quora-QP**	**MRPC**
Baseline	-	30.27	5.53	6.98	66.67	80.80
USE (Yang et al., 2019)	-	86.86	53.43	57.23	74.16	81.51
BERT	✗	77.50 ± 1.49	65.06 ± 1.06	65.91 ± 1.20	80.40 ± 1.05	88.95 ± 0.67
SBERT	✗	68.36 ± 5.28	58.04 ± 1.46	61.20 ± 1.66	73.44 ± 0.65	84.44 ± 0.68
BERT (*Upper-bound*)	✓	77.74 ± 1.24	65.78 ± 0.78	66.54 ± 0.94	81.23 ± 0.93	89.00 ± 0.56
SBERT (*Lower-bound*)	✓	72.07 ± 2.05	60.54 ± 0.99	63.77 ± 2.29	74.66 ± 0.31	84.39 ± 0.51
SBERT-NLPAug	✓	74.11 ± 2.58	58.15 ± 1.66	61.15 ± 0.86	73.08 ± 0.42	84.47 ± 0.79
AugSBERT-R.S.	✓	62.05 ± 2.53	59.95 ± 0.70	64.54 ± 1.90	73.42 ± 0.74	82.28 ± 0.38
AugSBERT-KDE	✓	74.67 ± 1.01	**61.49 ± 0.71**	**69.76 ± 0.50**	**79.31 ± 0.46**	84.33 ± 0.27
AugSBERT-BM25	✓	**75.08 ± 1.94**	61.48 ± 0.73	68.63 ± 0.79	79.01 ± 0.45	**85.46 ± 0.52**
AugSBERT-S.S.	✓	74.99 ± 2.30	61.05 ± 1.02	68.06 ± 0.93	77.20 ± 0.41	82.42 ± 0.32

Table 4: Summary of all the datasets being used for the in-domain tasks in this paper. STS and BWS are regression tasks, where we report Spearman's rank correlation $\rho \times 100$. Quora-QP and MRPC are classification tasks, where we report F_1 score of the positive class. Scores with the best AugSBERT strategy are highlighted. Corresponding development set performances can be found in Appendix G, Table 12.

- 일반적인 bi-encoder는 4.5~9.1p 낮은 성능 (seed-optim하면 차이 조금 감소)
- 가장 작은 데이터셋(sts)에서 random seed 영향 크게 나타남, 데이터셋 사이즈가 커질수록 영향력 감소
- AugSBERT 모든 task에서 1~6p 성능 향상
- Cross-topic에서는 AugSBERT 성능 향상 없지만, In-topic에서는 Cross-Encoder보다 좋은 성능을 보이기도 함

06. Results

Task		Regression ($\rho \times 100$)			Classification (F_1)	
Model / Dataset	(Seed Opt.)	**Spanish-STS**	**BWS (cross-topic)**	**BWS (in-topic)**	**Quora-QP**	**MRPC**
Baseline	-	30.27	5.53	6.98	66.67	80.80
USE (Yang et al., 2019)	-	86.86	53.43	57.23	74.16	81.51
BERT	✗	77.50 ± 1.49	65.06 ± 1.06	65.91 ± 1.20	80.40 ± 1.05	88.95 ± 0.67
SBERT	✗	68.36 ± 5.28	58.04 ± 1.46	61.20 ± 1.66	73.44 ± 0.65	84.44 ± 0.68
BERT (*Upper-bound*)	✓	77.74 ± 1.24	65.78 ± 0.78	66.54 ± 0.94	81.23 ± 0.93	89.00 ± 0.56
SBERT (*Lower-bound*)	✓	72.07 ± 2.05	60.54 ± 0.99	63.77 ± 2.29	74.66 ± 0.31	84.39 ± 0.51
SBERT-NLPAug	✓	74.11 ± 2.58	58.15 ± 1.66	61.15 ± 0.86	73.08 ± 0.42	84.47 ± 0.79
AugSBERT-R.S.	✓	62.05 ± 2.53	59.95 ± 0.70	64.54 ± 1.90	73.42 ± 0.74	82.28 ± 0.38
AugSBERT-KDE	✓	74.67 ± 1.01	**61.49 ± 0.71**	**69.76 ± 0.50**	**79.31 ± 0.46**	84.33 ± 0.27
AugSBERT-BM25	✓	**75.08 ± 1.94**	61.48 ± 0.73	68.63 ± 0.79	79.01 ± 0.45	**85.46 ± 0.52**
AugSBERT-S.S.	✓	74.99 ± 2.30	61.05 ± 1.02	68.06 ± 0.93	77.20 ± 0.41	82.42 ± 0.32

Table 4: Summary of all the datasets being used for the in-domain tasks in this paper. STS and BWS are regression tasks, where we report Spearman's rank correlation $\rho \times 100$. Quora-QP and MRPC are classification tasks, where we report F_1 score of the positive class. Scores with the best AugSBERT strategy are highlighted. Corresponding development set performances can be found in Appendix G, Table 12.

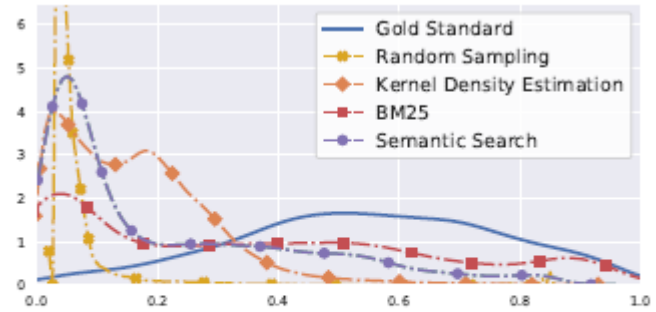


Figure 4: Comparison of the density distributions of gold standard with silver standard for various sampling techniques on Spanish-STS (in-domain) dataset.

- R.S.는 오히려 성능 감소
- BM25, KDE가 가장 좋은 성능 보여줌
- BM25가 골드 데이터셋과 가장 유사한 분포를 보여줌

06. Results

Source (Train)	Target (Evaluate)	In-Domain SBERT (*Upper-bound*)	Cross-Domain AugSBERT	Cross-Domain SBERT (*Lower-bound*)	Cross-Domain Bi-LSTM (Direct)	Cross-Domain Bi-LSTM (Adversarial)
AskUbuntu	Quora	0.504	**0.496**	**0.496**	0.059	0.066
	Sprint	0.869	0.852	0.747	**0.93**	0.923
	SuperUser	0.802	0.779	0.738	**0.806**	0.798
Quora	AskUbuntu	0.715	**0.602**	0.501	0.351	0.328
	Sprint	0.869	**0.875**	0.505	**0.875**	0.867
	SuperUser	0.802	**0.645**	0.504	0.523	0.485
SuperUser	AskUbuntu	0.715	**0.709**	0.637	0.629	0.627
	Quora	0.504	**0.495**	0.495	0.058	0.067
	Sprint	0.869	0.876	0.785	0.936	**0.937**
Sprint	AskUbuntu	0.715	**0.663**	0.613	0.519	0.543
	Quora	0.504	0.495	**0.496**	0.048	0.063
	SuperUser	0.802	**0.769**	0.660	0.658	0.636

Table 5: AUC(0.05) scores for domain adaptation experiments. All except SBERT (in-domain) are evaluated in cross-domain setup with the best transfer strategy highlighted. We adapt (Shah et al., 2018) Bi-LSTM models. Corresponding development set performances can be found in Appendix G, Table 13.

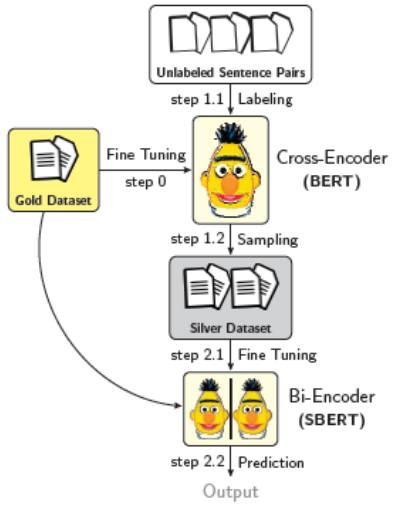


Figure 2: Augmented SBERT In-domain approach

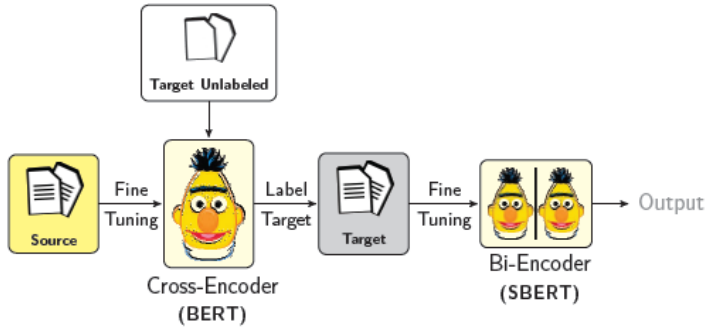


Figure 3: Domain adaptation with AugSBERT.

- 거의 모든 조합에서 AugSBERT가 SBERT보다 좋은 성능
- Source domain이 조금 더 일반적이거나 target domain이 조금 더 specific한 경우 AugSBERT의 장점이 두드러짐
- Sprint dataset (target)에서는 무려 37p 성능 향상

07. Conclusion

- 논문에서 bi-encoder의 성능 향상을 위해서 Data Augmentation 활용
- 적절한 Data Augmentation을 위해서는 sampling 방법이 매우 중요
- BM25 sampling이 성능과 연산양을 고려했을 때 가장 좋은 샘플링 기법
- Domain-adaptation에 활용할 때도 상당한 수준의 성능 향상을 보임

⇒ BERT와 같은 pre-trained 언어모델들은 여전히 inferencing time의 문제로 상용화에 어려움을 겪고 있음
경량화 혹은 Transformer의 구조를 바꾸는 해당 논문과 같은 시도들을 통해 서비스에 발전된 언어 모델의 탑재를 가능하게 한다는 시사점
⇒ 앞으로도 이러한 방향으로 서비스에 적용될 수 있는 시도들이 계속되지 않을까?

Q&A