Sentence-BERT

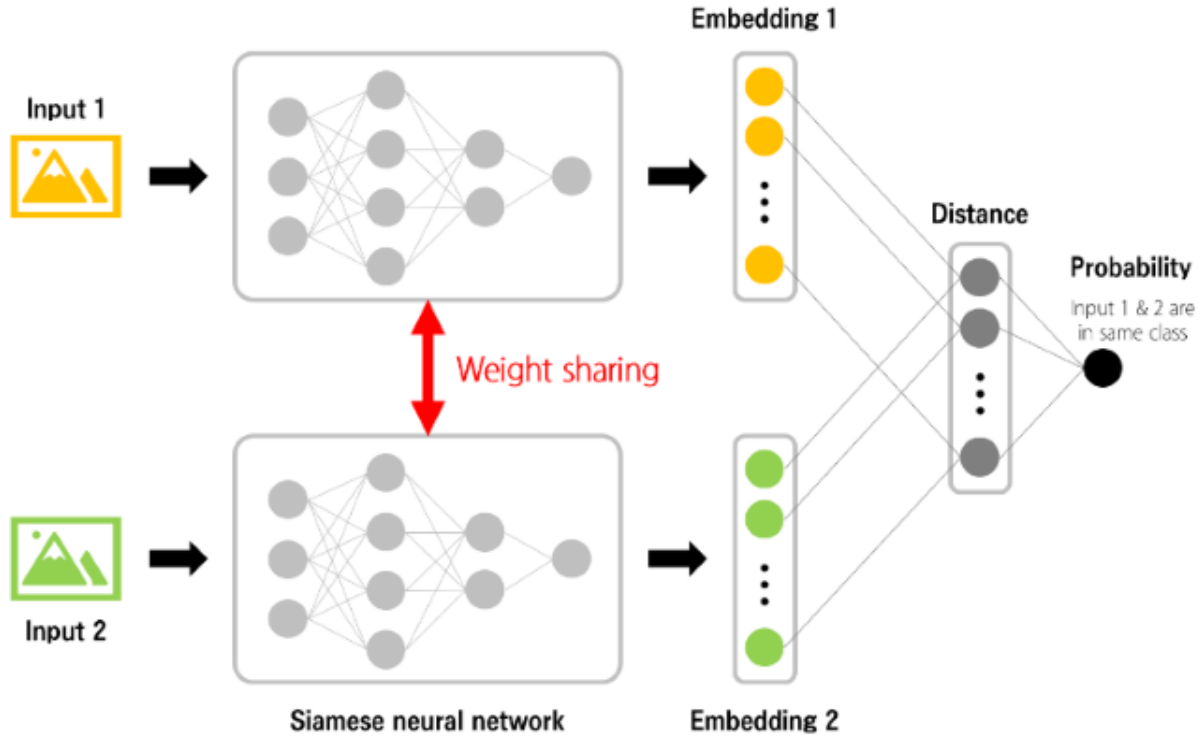
Sentence-BERT:Sentence Embeddings using Siamese BERT-Networks

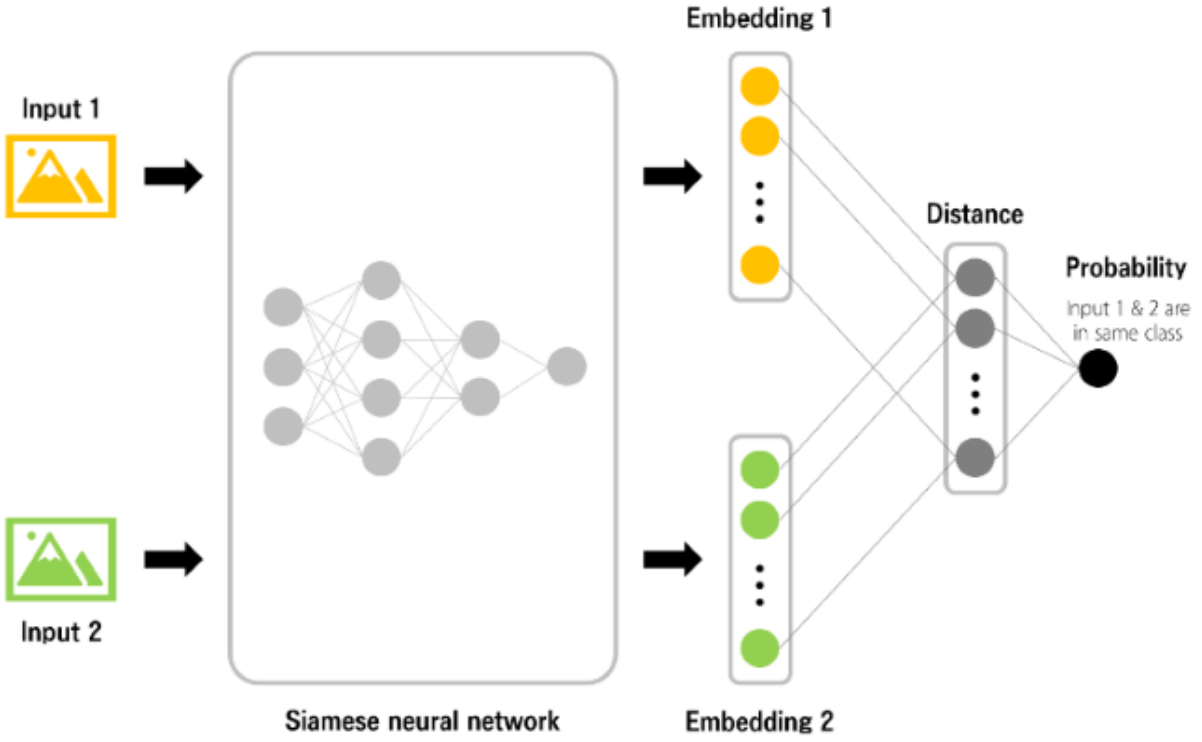
1. Introduction

- 본 논문은 2019년에 등장하였으며, sentence embedding을 위해 기존의 BERT 모델에 Siamese Network를 사용하여 빠른 연산과 좋은 성능을 이루어낸 논문
- BERT 모델의 경우 문장 분류, 의미론적 텍스트 유사성 문제에서 좋은 성능을 보이지만, pair regression과 같은 문제에서 입력 문장에 대해 매번 연산을 수행하기 때문에 많은 계산량을 요구
- 임베딩 값을 얻기 위해 Output 레이어에 평균값을 사용하거나, [CLS] 토큰을 사용하지만, GloVe 모델 보다 낮은 성능을 보여줌
- 이를 해결하기 위해 기존 BERT 모델에 Siamese Network 구조를 활용한 Sentence-BERT 제안

2. Background

2.1 Siamese Network Architecture





- Siamese Network는 weight를 공유하는 두 네트워크로 이루어짐
- 두 개의 데이터(Input 1, Input 2)가 입력 데이터
- 각 입력 데이터가 네트워크를 통과하여 임베딩 값 Embedding 1, Embedding 2 값이 생성

- L1 norm, L2 norm 등의 방법을 사용하여, 두 임베딩 값 사이의 거리를 계산합니다.
- 두 입력이 같은 클래스에 속한다면 거리를 가깝게, 다른 클래스에 속한다면 거리를 멀게 Siamese Neural Network를 학습

3. Model

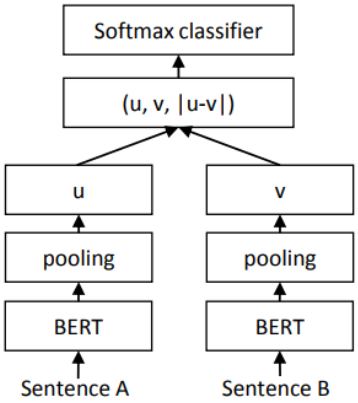


Figure 1: SBERT architecture with classification objective function, e.g., for fine-tuning on SNLI dataset. The two BERT networks have tied weights (siamese network structure).

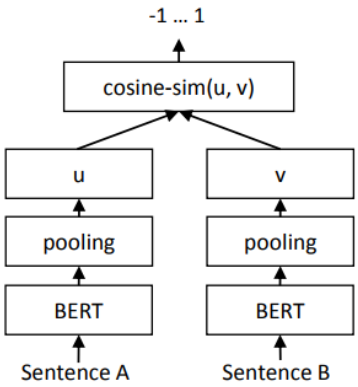


Figure 2: SBERT architecture at inference, for example, to compute similarity scores. This architecture is also used with the regression objective function.

- 공통적으로 의 구조는 각 문장A와 문장B가 BERT 모델을 통과하고 pooling을 통해 각 문장에 대한 임베딩 벡터 **u** 와 **v** 를 생성함
- SBERT의 네트워크 구조는 학습 데이터 셋에 따라 Classification Objective Funtion, Regression Objective Funtion, Triplet Objective Funtion을 사용 할 수 있음

3.1 Classification Objective Funtion

> Figure 1
> (1) 각 문장이 BERT 모델을 통과한 후, MEAN 방법을 사용하여 pooling을 진행
> (2) pooling을 통해 생성된 임베딩 벡터값 u 와 v 의 element wise difference한 |u-v| 계산
> (3) 이후 각 임베딩 벡터값 u 와 v 그리고 |u-v| 를 trainable weight 값인 Wt 와 곱함
> (4) 이 값을 softmax를 통해 확률값으로 변환해 주고, cross-entropy loss로 weight 업데이트

$$o = \text{softmax}(W_t(u, v, |u - v|))$$

3.2 Regression Objective Funtion

> Figure 2
> (1) Classification Objective Funtion과 동일하게 pooling을 통해 임베딩 벡터값 u 와 v 생성
> (2) 두 임베딩 벡터값 u 와 v 간의 cosine-similarity 계산
> (3) 두 문장의 유사도와 MSE loss를 통해 weight 업데이트

3.3 Triplet Objective Funtion

> (1) 기준이 되는 문장 anchor sentence a 와 일치하는 문장을 positive sentence p, 일치하지 않는 문장을 negative sentence n 라고 정의
> (2) 앞에서 정의한 a 와 p 의 벡터 공간상에 거리가 더 가깝게, a 와 n 은 거리가 멀어지게 학습하는 손실 함수
> (3) sx 값은 각 입력 문장들의 임베딩을 의미하고, epsilon 의 경우 sp 가 sn 보다 sa 에 가깝게 해주는 margin 역할
> (4) 본 논문에서 거리 계산은 Euclidean distance을 사용하였고, epsilon 값은 1로 설정

$$max(||s_a - s_p|| - ||s_a - s_n|| + \epsilon, 0)$$

3.4 Training Details

```
# Dataset
SNLI : 570,000 sentence pairs
MultiNLI : 430,000 sentence pairs

# Parameter
batch size = 16
optimizer = Adam
learning rate = 2e-5
pooling strategy = MEAN
```

4. Evaluation - Semantic Textual Similarity

NLI (Natural Language Inference) Dataset

Example	English Translation	Label
P: 저는, 그냥 알아내려고 거기 있었어요. H: 이해하려고 노력하고 있었어요.	I was just there just trying to figure it out. I was trying to understand.	Entailment
P: 저는, 그냥 알아내려고 거기 있었어요. H: 나는 처음부터 그것을 잘 이해했다.	I was just there just trying to figure it out. I understood it well from the beginning.	Contradiction
P: 저는, 그냥 알아내려고 거기 있었어요. H: 나는 돈이 어디로 갔는지 이해하려고 했어요.	I was just there just trying to figure it out. I was trying to understand where the money went.	Neutral

STS (Semantic Textual Similarity) Dataset

Example	English Translation	Label
한 남자가 음식을 먹고 있다. 한 남자가 뭔가를 먹고 있다.	A man is eating food. A man is eating something.	4.2
한 비행기가 착륙하고 있다. 애니메이션화된 비행기 하나가 착륙하고 있다.	A plane is landing. A animated airplane is landing.	2.8
한 여성이 고기를 요리하고 있다. 한 남자가 말하고 있다.	A woman is cooking meat. A man is speaking.	0.0

- NLI Dataset의 경우 두 입력 문장에 대해서 참(Entailment), 거짓(Contradiction), 중립(Neutral)의 관계를 Classification 하는데 사용
- STS Dataset의 경우 두 입력 문장의 유사도(Similarity)를 0~5점으로 나타내어 Regression에서 사용
- 본 논문의 저자들은 STS Dataset을 통해 SBERT의 성능을 평가했고, 예측된 유사도의 평가지표는 spearman correlation을 사용했으며 목적 데이터를 학습하지 않은 경우(Unsupervised STS)와 학습한 경우(Supervised STS)으로 나누어 실험을 진행

4.1 Unsupervised STS

Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	**76.69**	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	**78.46**	**74.90**	80.99	76.25	**79.23**	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	**74.53**	77.00	73.18	**81.85**	**76.82**	79.10	74.29	**76.68**

- Unsupervised STS의 경우 STS 2012-2016, STS benchmark, SICK-Relatedness Dataset을 사용
- SBERT, SRoBERTa가 SICK-R을 제외한 모든 task에서 SOTA를 달성
- BERT의 [CLS] 토큰을 이용한 경우는 가장 좋지 않은 결과를 보이며, BERT는 GloVe embedding보다 더 떨어지는 성능을 보임

4.2 Supervised STS

Model	Spearman
Not trained for STS	
Avg. GloVe embeddings	58.02
Avg. BERT embeddings	46.35
InferSent - GloVe	68.03
Universal Sentence Encoder	74.92
SBERT-NLI-base	77.03
SBERT-NLI-large	79.23
Trained on STS benchmark dataset	
BERT-STSb-base	84.30 ± 0.76
SBERT-STSb-base	84.67 ± 0.19
SRoBERTa-STSb-base	**84.92** ± 0.34
BERT-STSb-large	**85.64** ± 0.81
SBERT-STSb-large	84.45 ± 0.43
SRoBERTa-STSb-large	85.02 ± 0.76
Trained on NLI data + STS benchmark data	
BERT-NLI-STSb-base	**88.33** ± 0.19
SBERT-NLI-STSb-base	85.35 ± 0.17
SRoBERTa-NLI-STSb-base	84.79 ± 0.38
BERT-NLI-STSb-large	**88.77** ± 0.46
SBERT-NLI-STSb-large	86.10 ± 0.13
SRoBERTa-NLI-STSb-large	86.15 ± 0.35

- STS에 대해 학습하지 않은 경우, STSb만을 사용한 경우와 NLI도 사용한 경우 총 3가지에 대한 결과를 같이 제시
- STS에 대해 학습하지 않은 경우에는 SBERT가 가장 좋은 결과를 보임
- SBERT와 SRoBERTa의 차이는 매우 미세한 수준인 것을 알 수 있음

5. Evaluation - SentEval

Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	Avg.
Avg. GloVe embeddings	77.25	78.30	91.17	87.85	80.18	83.0	72.87	81.52
Avg. fast-text embeddings	77.96	79.23	91.68	87.81	82.15	83.6	74.49	82.42
Avg. BERT embeddings	78.66	86.25	94.37	88.66	84.40	92.8	69.45	84.94
BERT CLS-vector	78.68	84.85	94.21	88.23	84.13	91.4	71.13	84.66
InferSent - GloVe	81.57	86.54	92.50	**90.38**	84.18	88.2	75.77	85.59
Universal Sentence Encoder	80.09	85.19	93.98	86.70	86.38	**93.2**	70.14	85.10
SBERT-NLI-base	83.64	89.43	94.39	89.86	88.96	89.6	**76.00**	87.41
SBERT-NLI-large	**84.88**	**90.07**	**94.52**	90.33	**90.66**	87.4	75.94	**87.69**

- SentEval은 다양한 task에서 모델의 sentence embedding 품질을 평가하는 도구
- 서로 다른 7개의 task(MR, CR, SUBJ 등)에 대하여 Logistic regression classifier를 실행한 결과, SBERT가 대부분의 task에서 우수한 성능이 나타나는 것을 알 수 있음

6. Ablation Study

	NLI	STSb
Pooling Strategy		
MEAN	**80.78**	**87.44**
MAX	79.07	69.92
CLS	79.80	86.62
Concatenation		
(u, v)	66.04	-
$(\|u - v\|)$	69.78	-
$(u * v)$	70.54	-
$(\|u - v\|, u * v)$	78.37	-
$(u, v, u * v)$	77.44	-
$(u, v, \|u - v\|)$	**80.78**	-
$(u, v, \|u - v\|, u * v)$	80.44	-

- pooling의 방법으로 NLI와 STSb 데이터에서 **MEAN** 을 사용했을 때 가장 성능이 좋은 것을 알 수 있음
- Concatenation에 대해서는 임베딩 벡터 **u**, **v** 그리고 **|u-v|** 의 성능이 가장 좋음

7. Computational Efficiency

Model	CPU	GPU
Avg. GloVe embeddings	6469	-
InferSent	137	1876
Universal Sentence Encoder	67	1318
SBERT-base	44	1378
SBERT-base - smart batching	83	2042

- 계산의 효율성에 대한 실험을 진행하였으며, 수치가 높을수록 효율적인 계산을 의미
- Glove와 Infersent가 CPU 연산에서 좋은 성능을 보이지만, GPU 환경에서는 SBERT가 더 좋은 연산 성능을 보임

8. Conclusion

- 본 논문의 저자들은 BERT를 사용한 sentence embedding 방법이 시간 및 효율성 측면에서 부적절함을 지적하며, Simaese Network 구조를 사용한 SBERT를 제안
- SBERT의 경우 다양한 데이터 셋에서 좋은 sentence embedding 방법임을 실험을 통해 보였으며, 효율성 측면에서도 기존의 BERT보다 우수함을 보여줌