Ko-bert2BERT: 지식 전이를 통한 인코더-디코더 언어모델의 효율적인 사전 학습

전북대학교 인지컴퓨팅 연구실

석사과정 이건희

서론

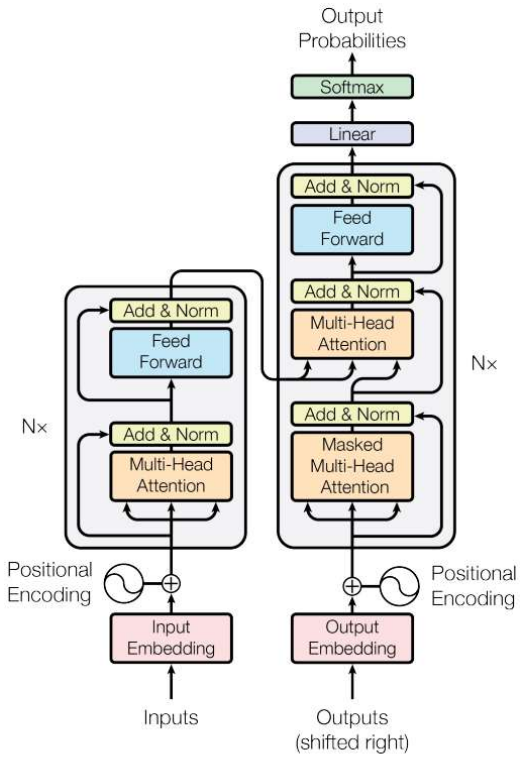


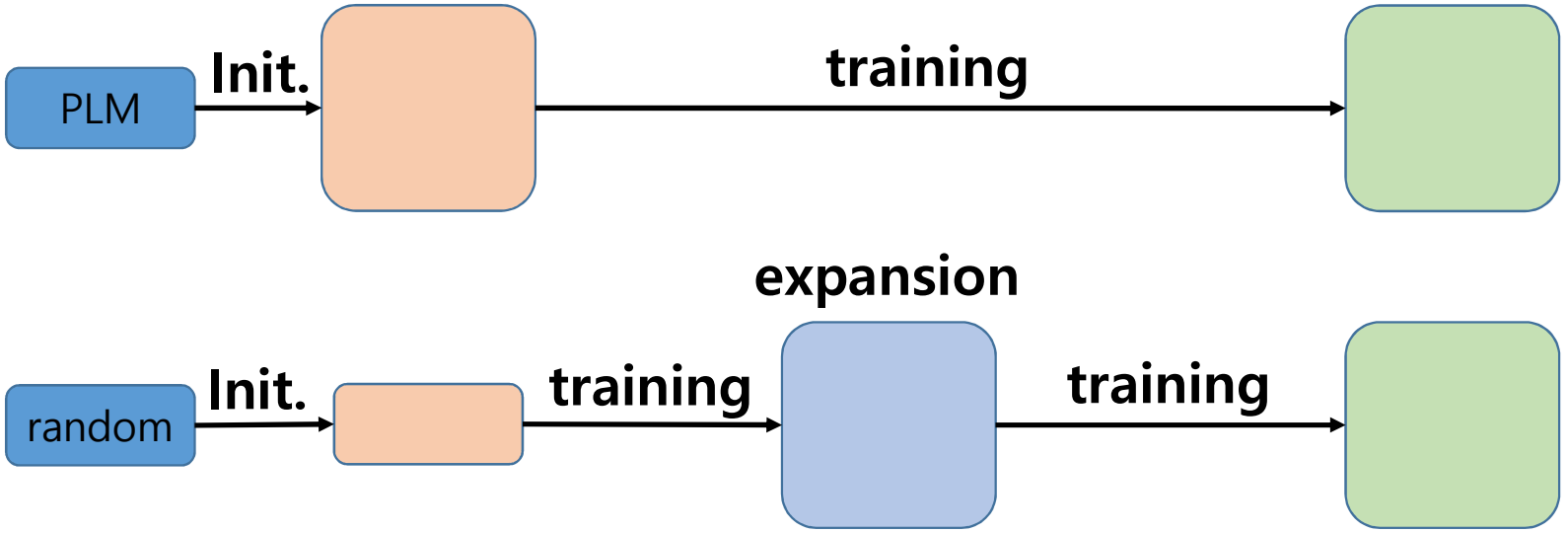
Figure 1: The Transformer - model architecture.

- Transformer 모델의 등장으로 NLP는 크게 성장
- 대부분의 새로운 모델들은 Transformer의 변형

- 다양한 변형, 다양한 규모
- 같은 구조의 다른 규모

- 매번 새로운 크기의 모델을 만들 때 처음부터 학습하는 것은 비효율적

기존 연구



- 작은 사전학습 모델을 재사용하여 목표 모델을 초기화
- 작은 모델을 학습 중에 목표 모델로 확장

기존 연구

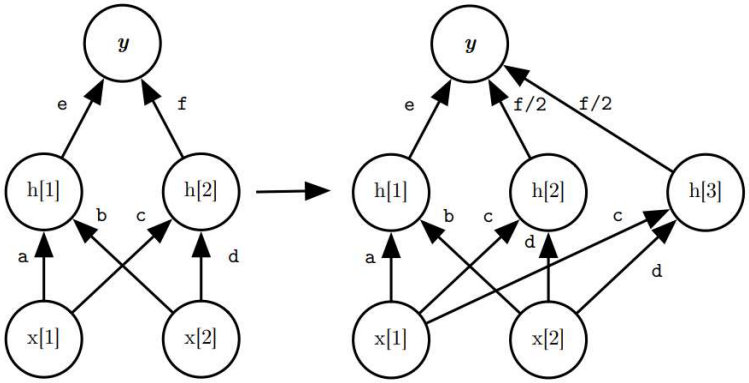


Figure 2: The `Net2WiderNet` transformation.

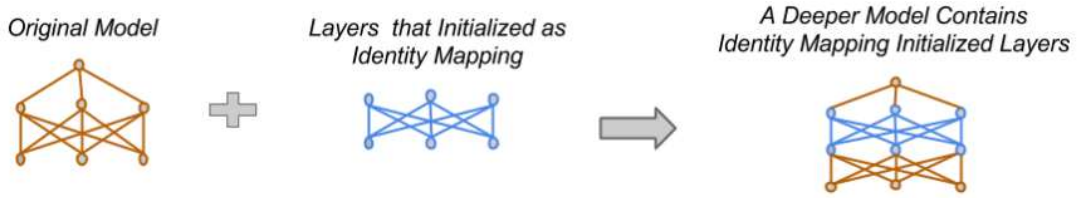


Figure 3: The Net2DeeperNet Transformation

- Net2Net(ICLR 2016)
  - 확장 전후 입력과 출력이 같아지도록 Function Preserving Initialization(FPI) 제안
  - 깊이 확장 - 입력과 출력이 같아지도록 초기화된 레이어를 추가
  - 너비 확장 - 입력 차원은 복사된 만큼 나눔, 출력 차원은 단순 복사

기존 연구

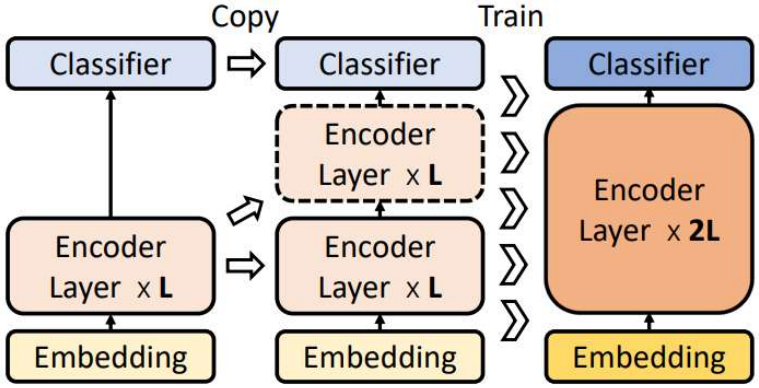


Figure 3. The diagram of the *stacking* algorithm.

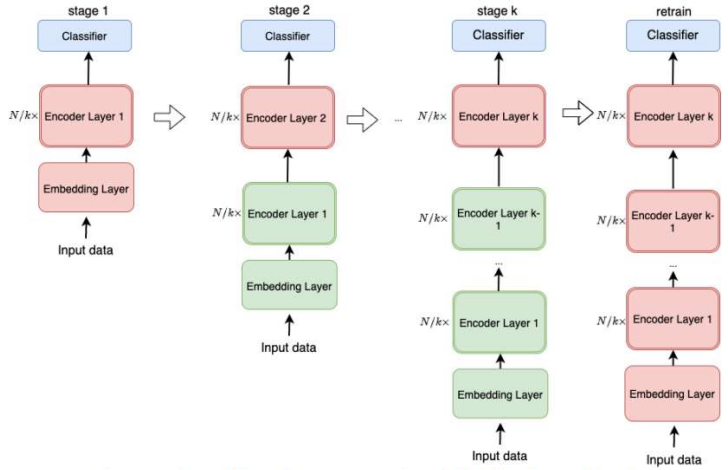


Figure 1: The framework of MSLT method.

- Progressively Stacking(2019), Progressively Stacking 2.0(2020)
  - L개 레이어를 복사하여 위에 N번 쌓는 방식으로 모델의 깊이를 정수배 확장
  - 작은 모델부터 시작하여 연산량 감소, 추가 레이어만 학습하여 연산량 감소

기존 연구

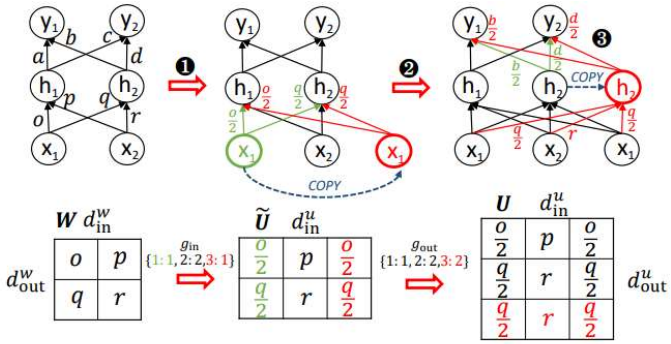


Figure 4: Overview of the function preserving initialization (FPI).

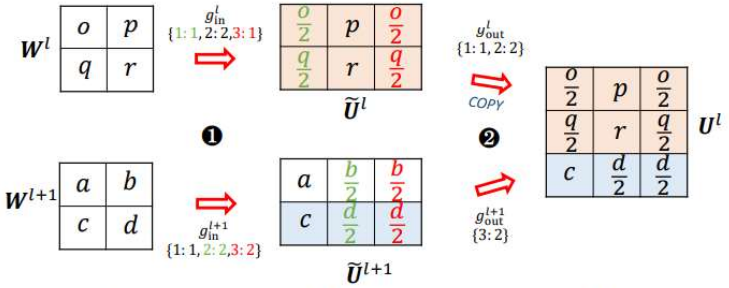


Figure 5: Overview of the advanced knowledge initialization (AKI).

- bert2BERT()
  - 깊이와 너비에 대한 확장을 동시에 수행, 깊이는 Progressively Stacking의 방식
  - 너비는 기본적으로 FPI개념을 적용하여 Net2Net의 방식을 따름
  - Advanced Knowledge Initialization(AKI) 방식 제안
    - 복사로 인한 대칭성을 해소하고 상위 레이어의 지식을 섞어주는 역할
  - 인코더와 디코더에서 각각 실험

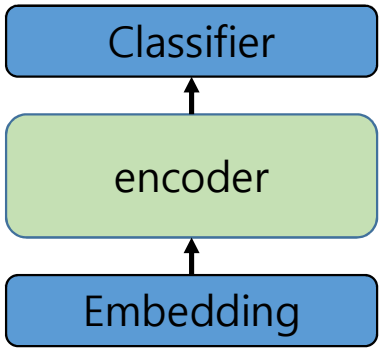
실험 – 사전 학습

- 인코더-디코더 모델인 T5 사용
- 3가지 규모 – Base, Small, Tiny
- 한국어 위키데이터 700MB를 학습:검증=199:1 분할
- 1000스텝 마다 validation loss측정
- 1epoch 당 11,136 step씩 총 5epoch 학습

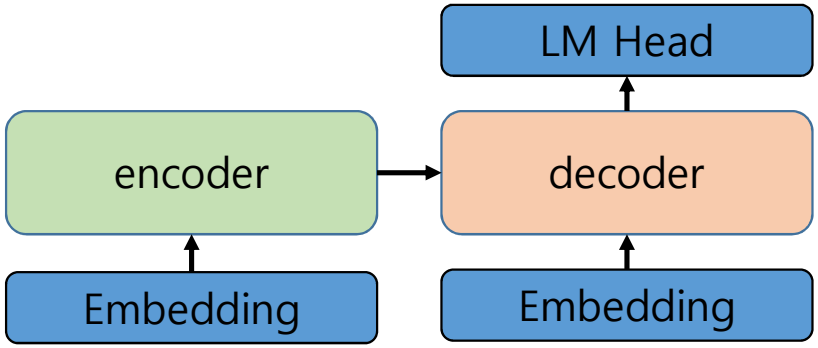
표 1. 모델 크기

Model	num_layers	d_model	num_heads	d_ff
Base	12+12	768	12	3072
Small	12+12	512	8	1024
Tiny	6+6	512	8	1024

실험 – 미세 조정



- 인코더
  - KLUE-NLI,
  - KorQuAD1.0(정답 스판 추출(Ext.))



- 인코더-디코더
  - AIHub 신문기사 요약 태스크
  - KorQuAD1.0(정답 생성(Gen.))

실험 – 모델 명명

- 모델의 크기
  - **B, S, T**: Base, Small, Tiny
- 확장에 사용한 방식
  - **_d_**: 깊이 확장
  - **_f_**: 너비 확장 FPI
  - **_a_**: 너비 확장 AKI
  - **_init-**: 사전학습 모델로 초기화

- 예시) **T_init-d_S_a_B**
  - PLM **T**로 모델 **S**를 초기화**(init-)**하는 방법으로 깊이 확장**(d)**를 사용
  - 이후 학습 도중 **S**에서 너비 확장 AKI**(a)**를 사용하여 **B**로 확장

학습 곡선

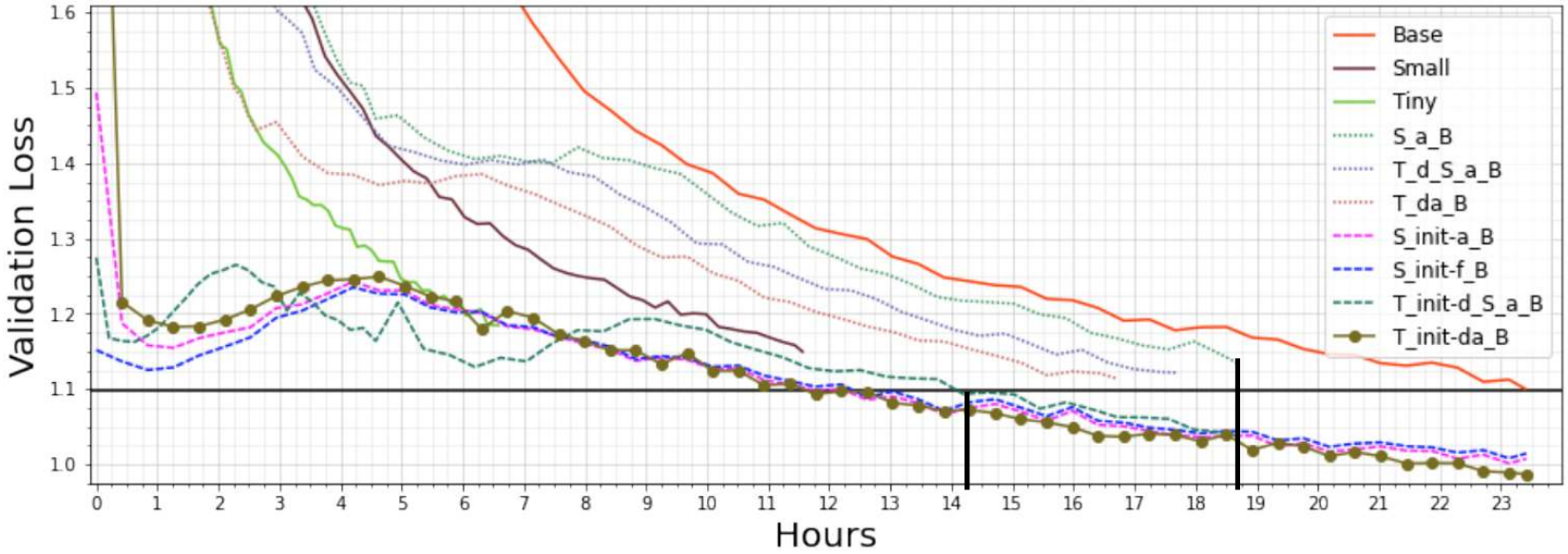


그림 1. 학습 곡선

- Base 23.4h
- S_a_B 18.7h 약 80%
- T_init-d_S_a_B 14.25h 약 60.8%

학습 곡선

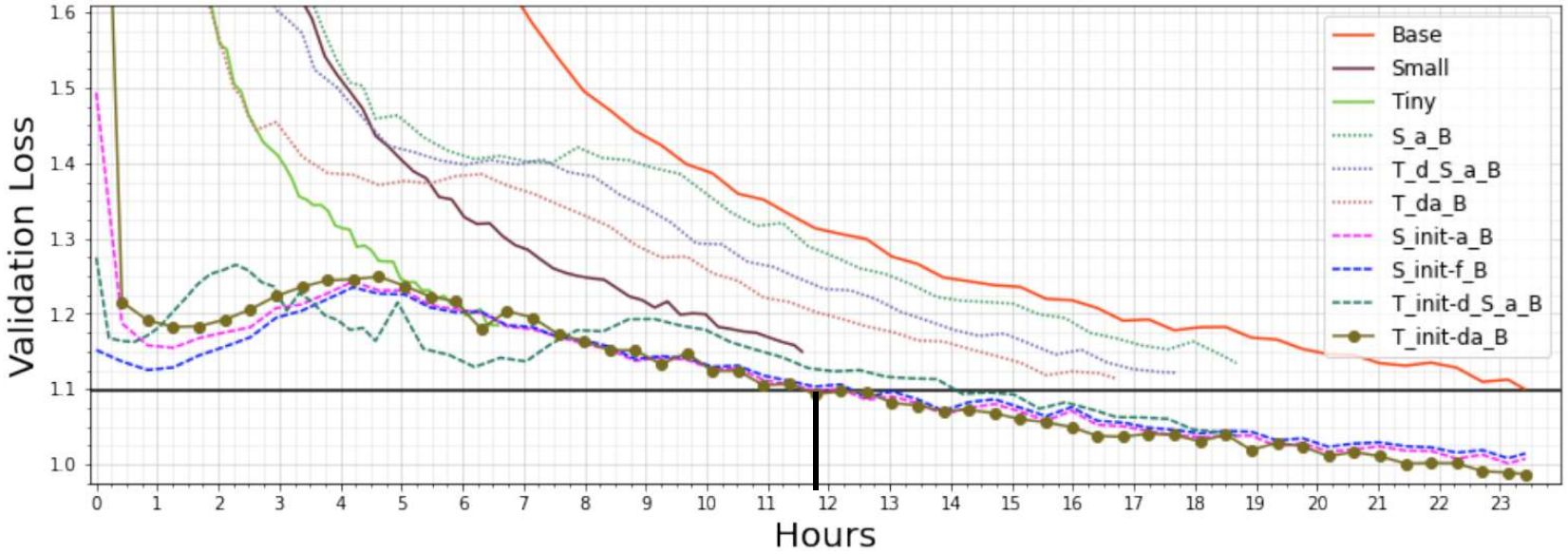


그림 1. 학습 곡선

- S_init-f_B, S_init-a_B 11.75h 약 50.2%
- T_init-d_S_a_B 가장 낮은 loss

미세 조정

표 2. 각 사전 학습 모델 별 다운스트림 태스크 성능

n	Model	Train epochs or steps	Train time (h)	Ratio (/Base)	Valid loss	KLUE NLI Acc.	KorQuAD1.0 Ext. EM	KorQuAD1.0 Ext. F1	KorQuAD1.0 Gen. EM	KorQuAD1.0 Gen. F1	Summarization Rouge-L F1	Avrg. score
1	T	5	6.6	28.21	1.1845	66.63	63.44	76.88	67.94	80.83	40.72	66.27
2	S	5	11.6	49.57	1.1487	68.60	72.67	84.15	71.27	83.91	40.94	69.40
3	S_a_B	2-3	18.7	79.91	1.1346	68.17	74.61	85.49	71.51	84.22	40.98	69.71
4	T_d_S_a_B	1-1-3	17.2	73.50	1.1221	69.33	76.48	87.01	72.77	85.41	40.78	70.63
5	T_da_B	2-3	15.7	67.09	1.1146	70.73	76.65	87.09	72.79	85.06	40.74	70.91
6	B	5	23.4	100	1.0994	68.60	74.61	85.41	71.20	84.23	40.72	69.74
7	S_init-a_B	28000	11.75	50.21	1.0986	69.57	73.81	84.77	73.81	85.50	40.86	70.17
8	T_init-d_S_a_B	2-3	23.4	100	1.0383	71.93	75.37	86.00	75.63	86.97	41.04	71.49
9	S_init-f_B	5	23.4	100	1.0146	72.13	74.77	85.87	75.65	87.15	**41.21**	71.59
10	S_init-a_B	5	23.4	100	1.0076	71.07	77.00	87.11	75.94	87.60	41.03	71.70
11	T_init-da_B	5	23.4	100	0.9870	**72.20**	**78.07**	**88.39**	**76.17**	**87.88**	40.97	**72.36**

- 6(Base) vs 7 – 유사 loss, 학습 시간 50.21%, 더 좋은 결과(+0.43)
- 6(Base) vs 11(best) – 같은 학습 시간, 더 좋은 결과(+2.62)

결론 및 향후 연구

- 결론
  - 기존 방법론들이 인코더-디코더에서도 유효함을 확인
  - 사전학습 모델의 재사용을 통해 빠른 학습이 가능하고 따라서 같은 시간을 사용하면 더 좋은 모델을 얻을 수 있음

- 향후 연구
  - 깊이 확장 시 실수배(xN)가 아닌 정수배(8L->12L) 확장을 효율적으로 하는 방법

감사합니다