Extremely Weakly Supervised Text Classification(MEGClass) with Semi-Supervised Learning(MixText)

2025. 02. 24

김윤서

Data **M**ing **Q**uality **A**nalytics

Data Mining Quality Analytics

고려대학교 KOREA UNIVERSITY

자기소개

김윤서

성균관대학교 경영학과/데이터사이언스학과(21)
관심분야 : Text mining, Explainable AI, Semi-Supervised learning

연구배경

Extremely weakly supervised text classification (XWS-TC)

학습방법	Supervised learning	Weakly Supervised learning	**Extremely Weakly supervised learning**
레이블 사용량	많음	일부 존재	극소량
특징	모든 데이터에 레이블 존재	일부 데이터에만 라벨, 또는 라벨 정확성이 떨어짐	1% 미만의 데이터에만 라벨, 대다수는 비지도
데이터 확장성	낮음 특정 도메인에 고정	-	높음 다양한 도메인에 적용 가능

- [클래스 이름]과 [레이블이 없는 문서]의 입력만으로 문서 분류
- 비라벨 데이터에 pseudo-label를 부여 후, 지도 학습

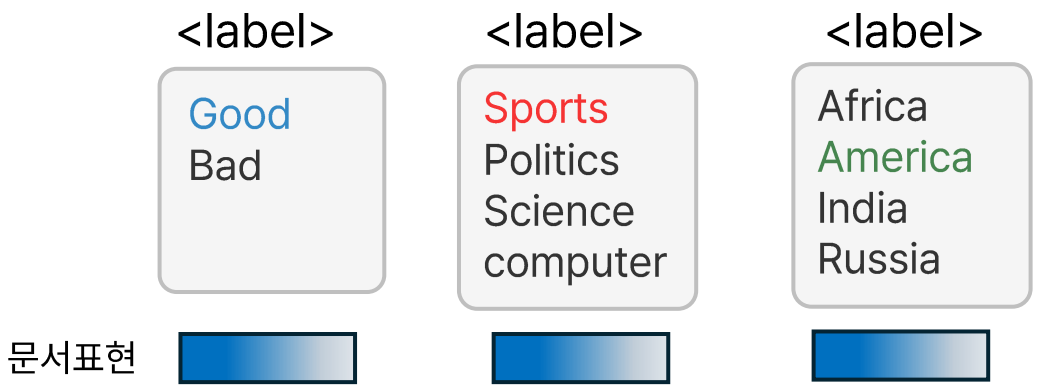
→ Pseudo-labeling 기법이 성능을 좌우함.

연구배경

Extremely weakly supervised text classification (XWS-TC)

Supervised learning

(학습) I really love America baseball!
<label> : good

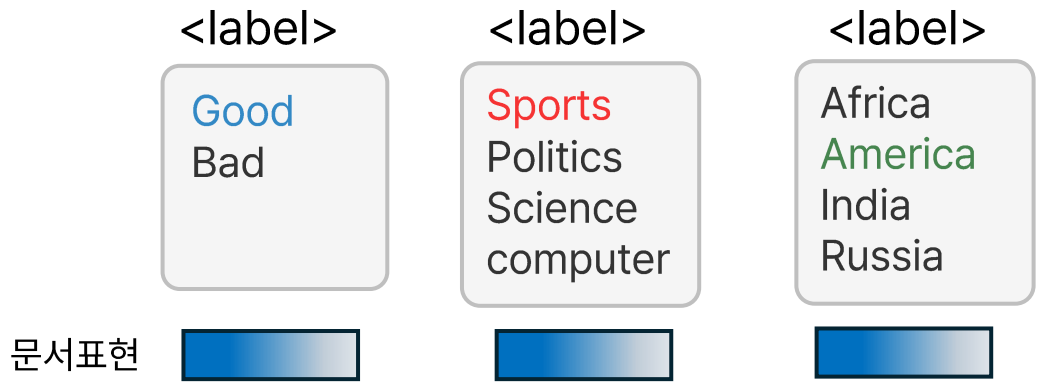


연구배경

Extremely weakly supervised text classification (XWS-TC)

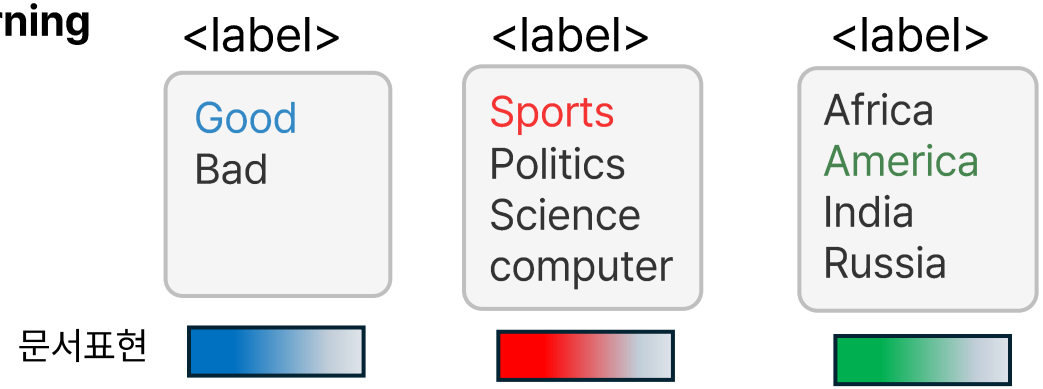
Supervised learning

(학습) I really love America baseball!
<label> : good



Extremely weakly supervised learning

I really love America baseball!

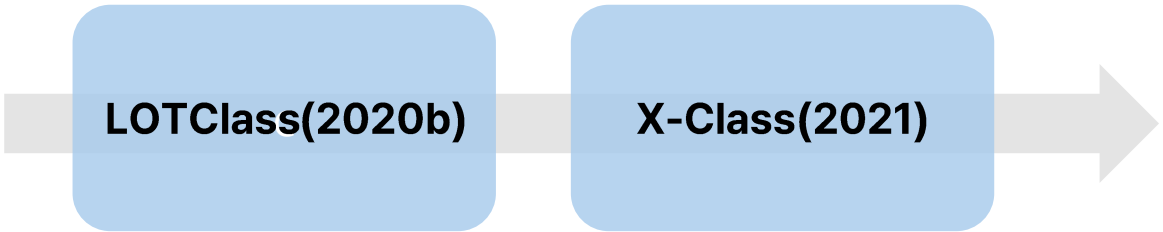


→ 사용자 정의 클래스 이름에 적응할 수 있는 문서 표현 생성 가능

연구배경

Extremely weakly supervised text classification (XWS-TC)

연구 동향



Representation learning 관점

키워드 그래프 기반

프롬프트 기반

연구배경

Extremely weakly supervised text classification (XWS-TC)

연구 동향

LOTClass(2020b) → X-Class(2021)

Representation learning 관점

유사한 문서와 클래스는 거의 동일하게 표현되거나
같은 분포에 있어야 한다!

→ 알맞은 의사라벨 부여 가능
→ 주어진 문서 집합과 클래스 집합을 얼마나
잘 표현(벡터화) 할 것인가?

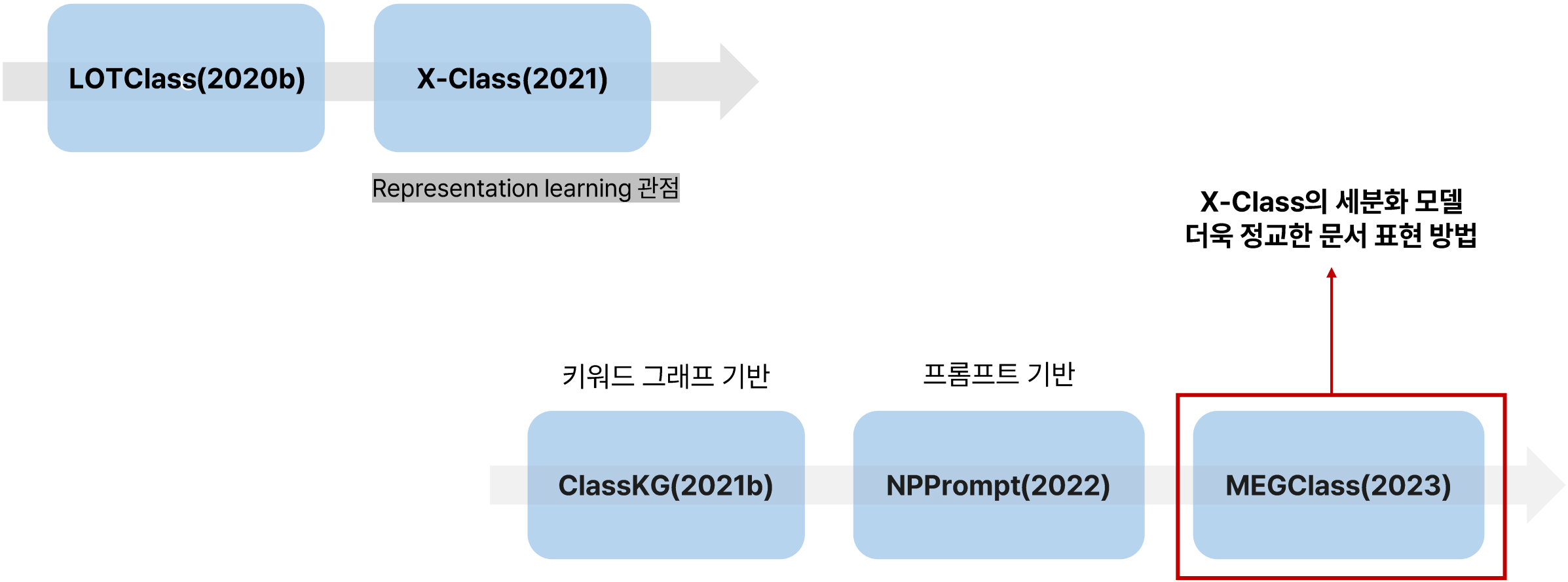
키워드 그래프 기반 | 프롬프트 기반

ClassKG(2021b) → NPPrompt(2022) → MEGClass(2023)

연구배경

Extremely weakly supervised text classification (XWS-TC)

연구 동향



MEGClass

XWS-TC

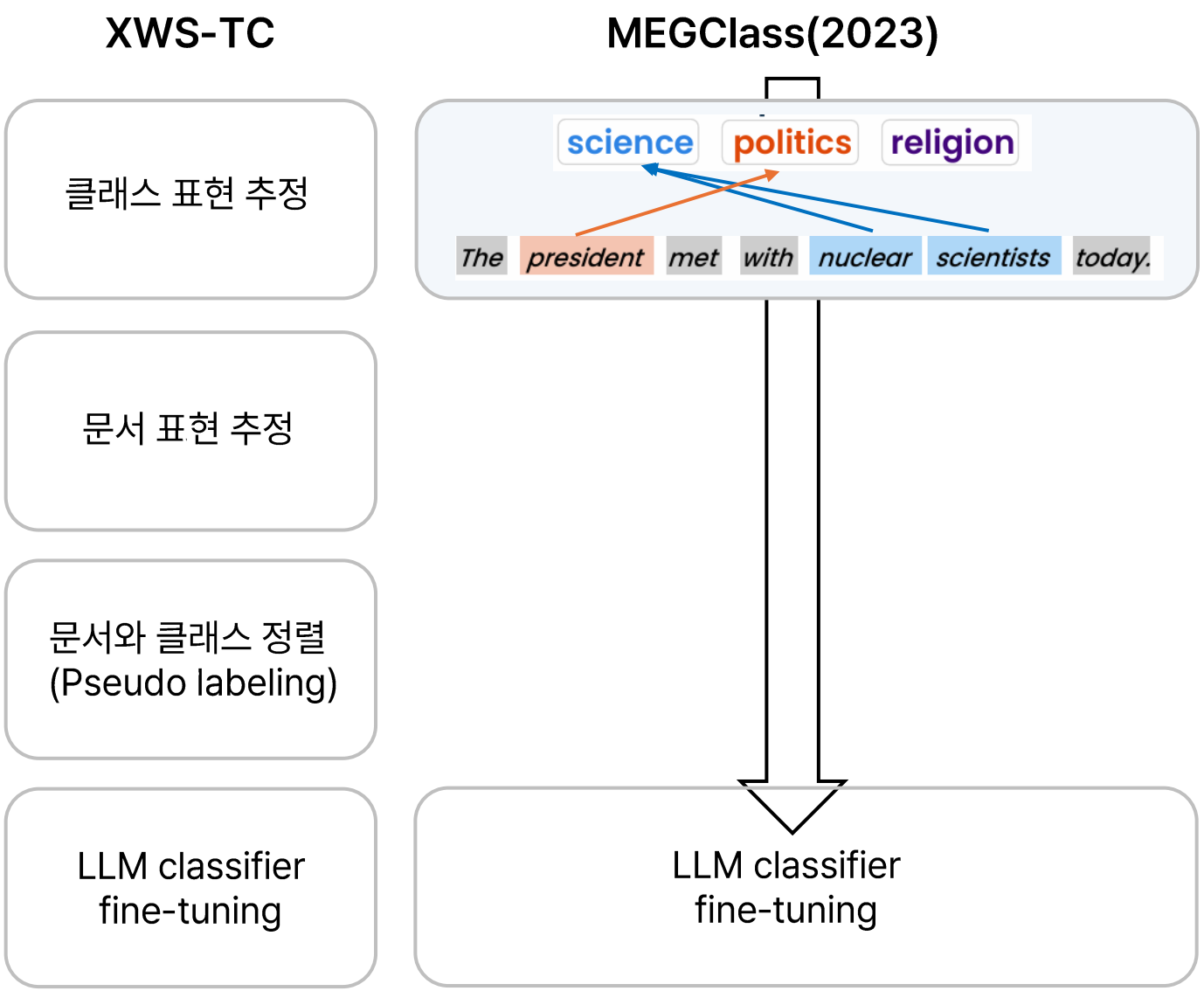
클래스 표현 추정

문서 표현 추정

문서와 클래스 정렬
(Pseudo labeling)

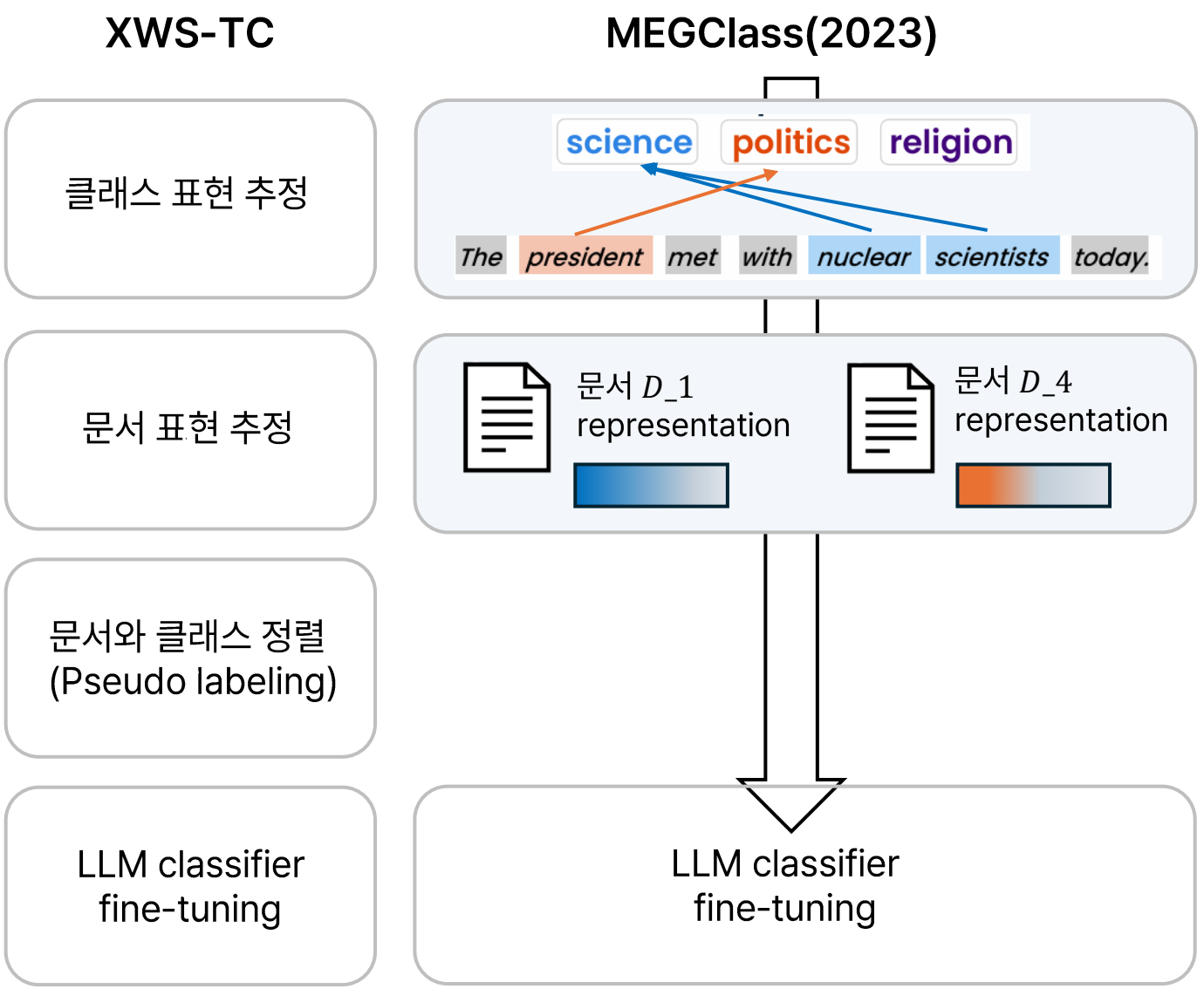
LLM classifier
fine-tuning

MEGClass



① 각 클래스 별 표현 임베딩 값
{science : 0.83694, politics : 0.2314, religion : 0.123}

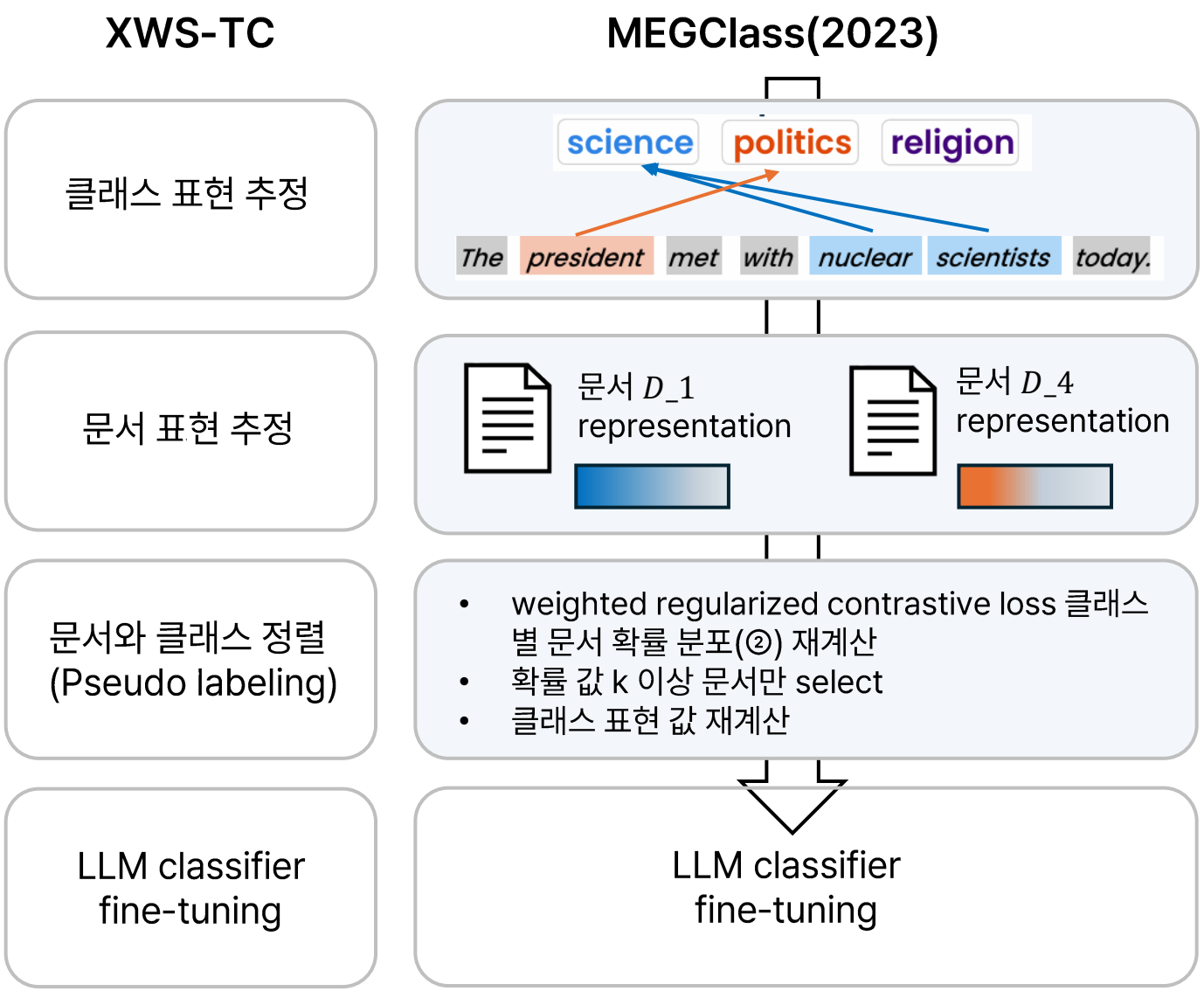
MEGClass



① 각 클래스 별 표현 임베딩 값
{science : 0.83694, politics : 0.2314, religion : 0.123}

② 클래스별 문서 확률 분포
P(문서1 ∈ science) = 0.123...
③ 문서 표현 임베딩 값
{ 0 : 0.1, 1 : 0.32, 2 : 0.5...}

MEGClass

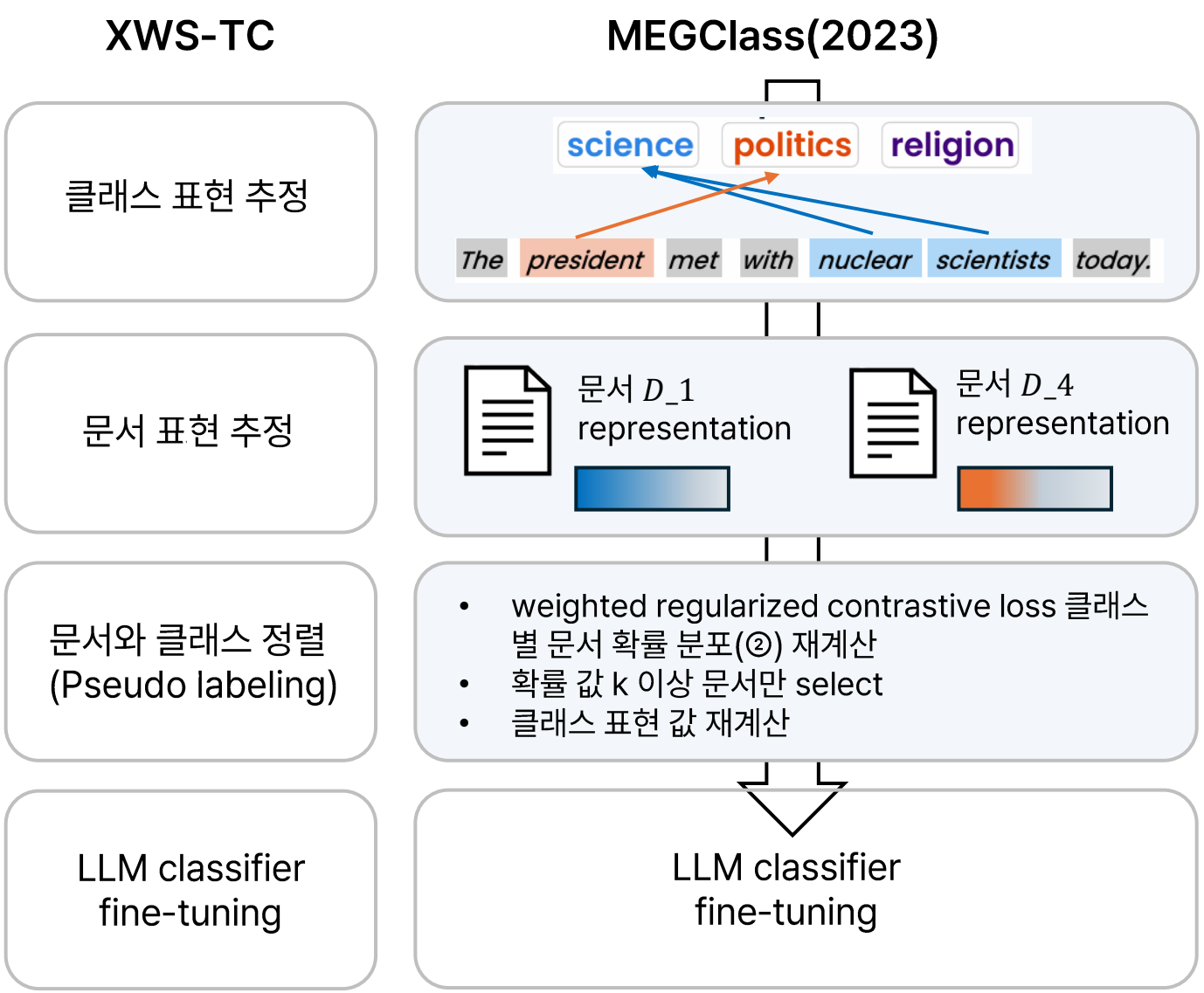


① 각 클래스 별 표현 임베딩 값
{science : 0.83694, politics : 0.2314, religion : 0.123}

② 클래스별 문서 확률 분포
P(문서1 ∈ science) = 0.123...
③ 문서 표현 임베딩 값
{ 0 : 0.1, 1 : 0.32, 2 : 0.5...}

③ Soft label data

MEGClass



+ MixText(2020)

Consistency learning과 텍스트 증강을 이용한 Semi-Supervised learning 방법론

XWS-TC task에서 Semi를 어떻게 이용?

Framework

MEGClass + MixText

MEGClass는 확률 값의 상위 k% 문서만을 사용

문제 인식

본래 데이터에서 50%(논문 값)만이 학습 데이터로 만들어지는 것.
→ 나머지 데이터의 손실

상위 k%의 문서도 결국 노이즈가 있는 라벨
→ 노이즈가 있는 라벨로 지도 학습 시,
모델이 잘못된 데이터를 학습할 가능성이 높음

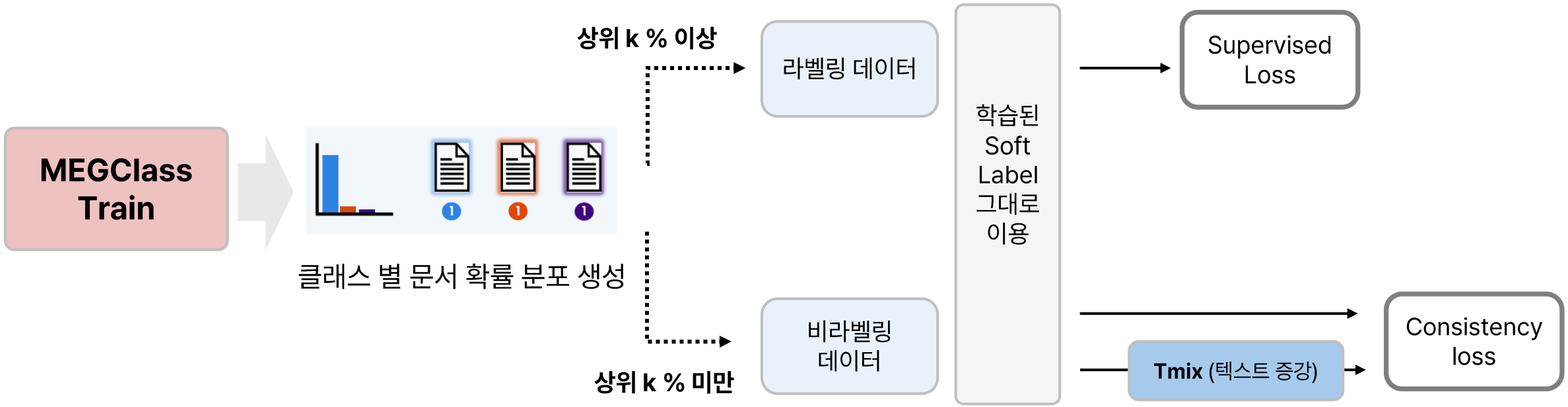
Proposed Method (SSL MixText)

SSL에서는 라벨/비라벨 데이터를 동시에 사용
[k%이상의 데이터를 라벨 데이터]
[K%미만의 데이터를 비라벨 데이터 취급]
→ 데이터 보존 가능

K값을 줄여서 라벨의 신뢰도를 높이고,
SSL loss를 통해 전체 데이터 학습

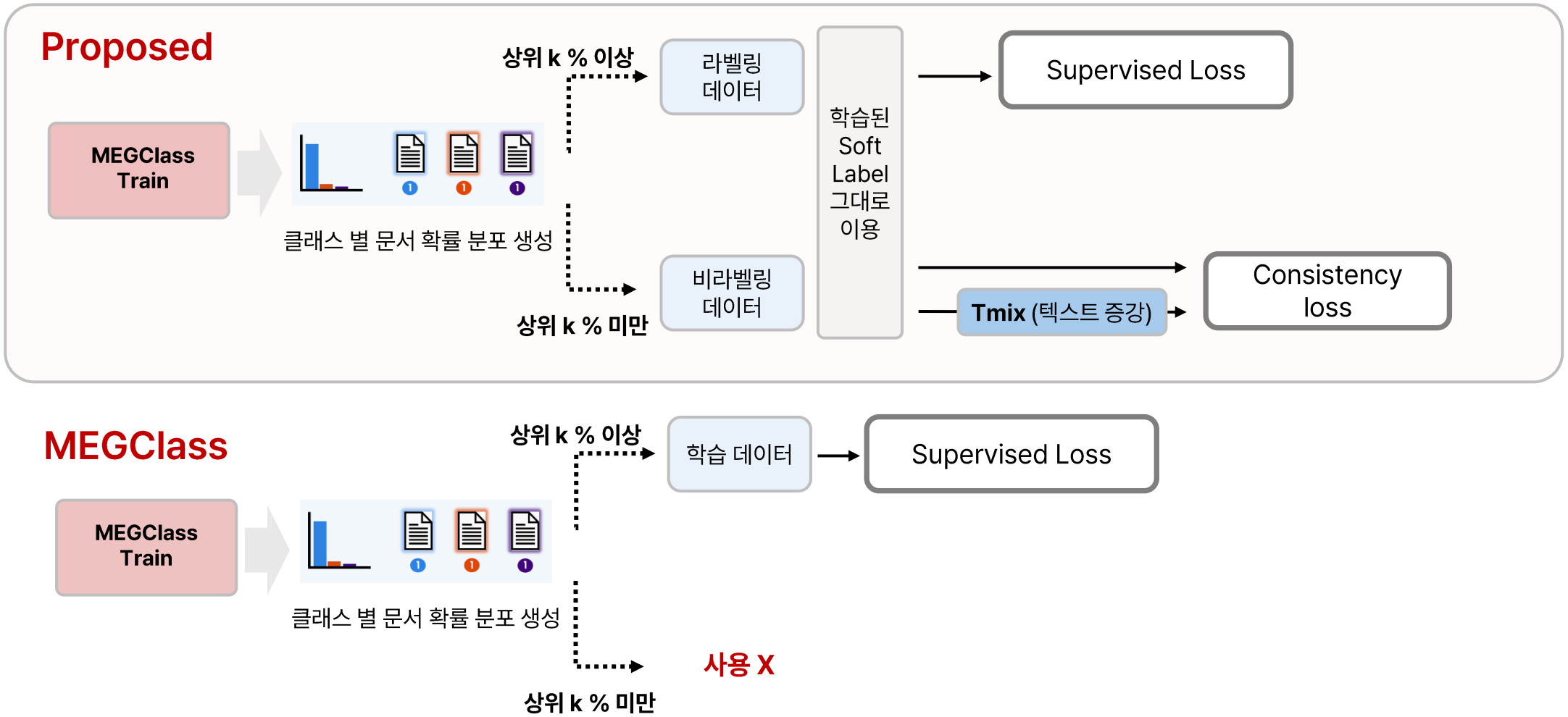
Framework

MEGClass + MixText



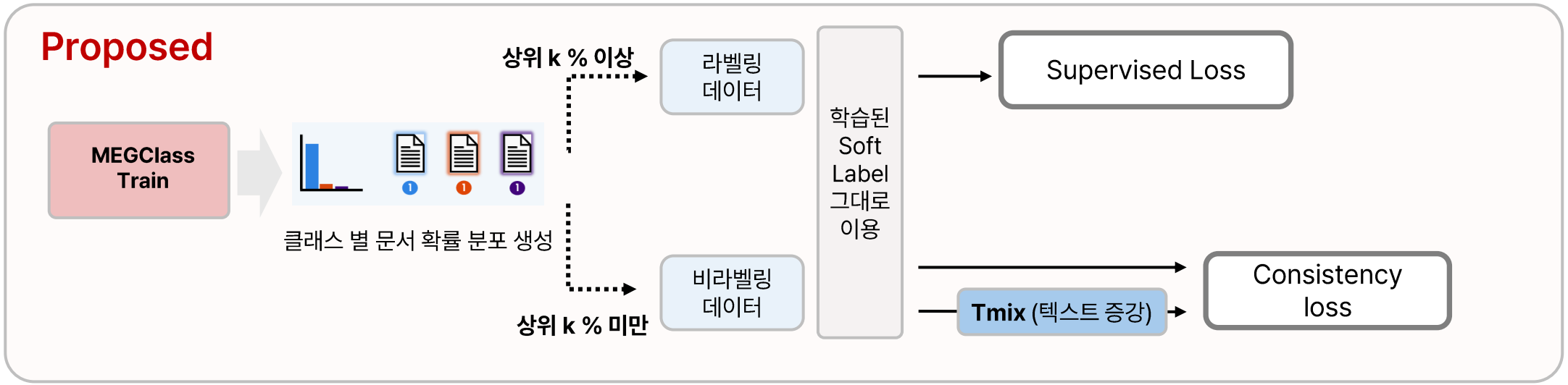
Framework

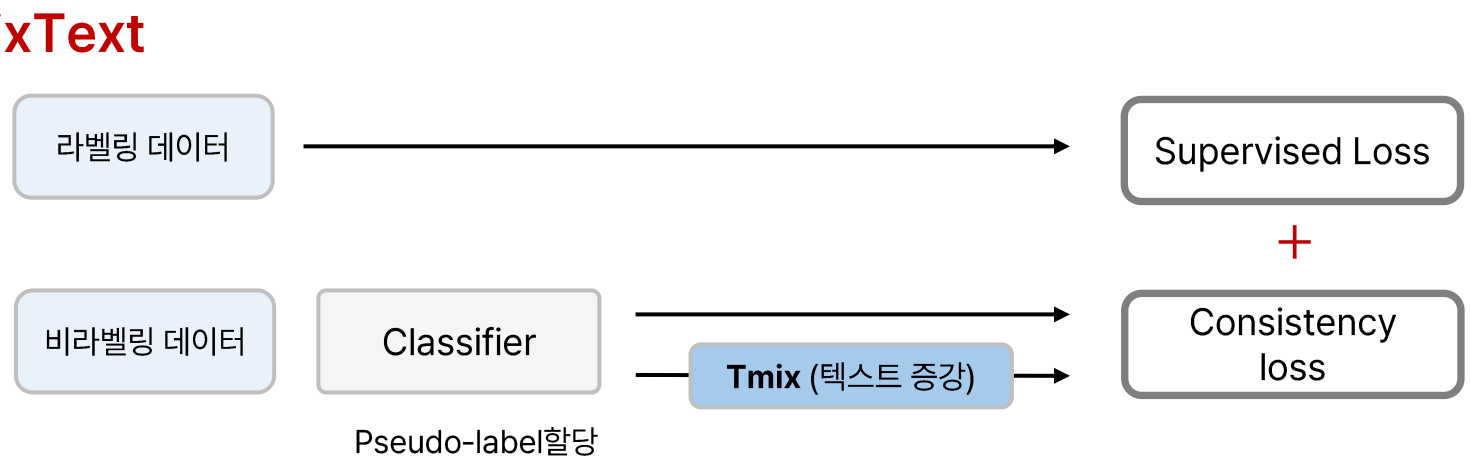
기존과 비교 - MEGClass



Framework

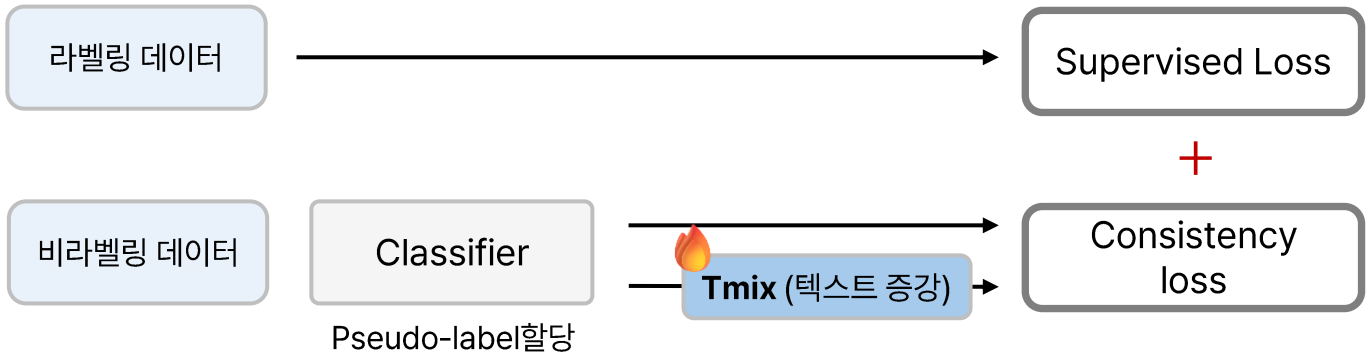
기존과 비교 - MEGClass





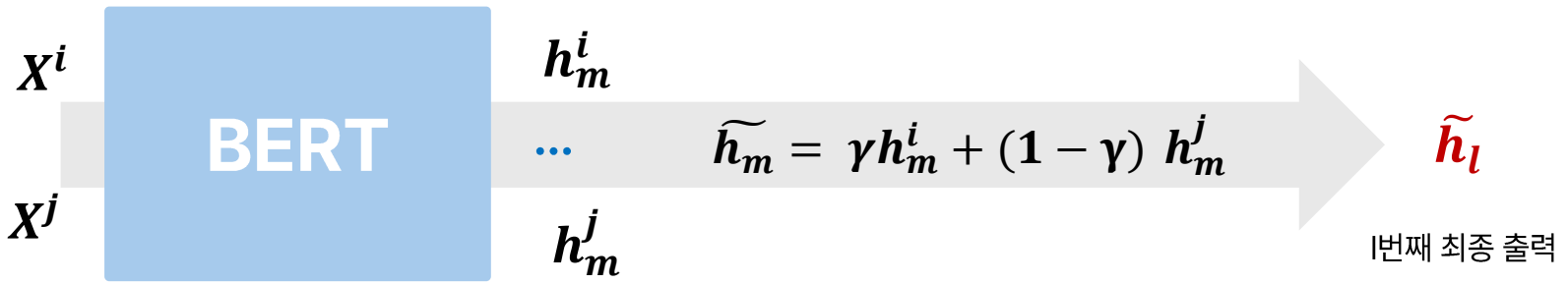
Framework

MixText



- 본래 image semi-supervised learning에서 사용된 증강 기법인 **Mixup**을 텍스트에 적용한 방법론
- 텍스트는 이미지와 다르게 이산적인 토큰이기 때문에 MixUp 기반 선형 보간이 쉽지 않음

→ 텍스트의 hidden vectors차원에서 보간을 수행하자!



Experiment

Datasets

- MEGClass 에서 실험한 7개의 benchmark dataset에서 Yelp, 20News, NYT-fine 사용
- 5:1 로 train/test 분리, 9:1 로 train/val 분리

Dataset	Domain	Classes	Train	Val	Test	total
NYT-fine	News	**26**	8,634	960	1,933	11,527
20News	News	**5**	13,401	1,490	2,980	17,871
Yelp	Sentiment	**2**	13,500	1,500	3,000	18,000

Experiment

Metrics

- MEGClass는 학습 파라미터 논문과 동일하게 설정(iteration만 GPU로 인해 4에서 1로 수정)
- Test Accuracy, f1_macro, f1_micro 측정
- MEGClass에서 데이터셋을 선택하는 k 값(threshold)를 0.5(논문제안)과 0.2 2가지로 진행하여 비교

Why?

K값을 작게 하면 더 신뢰있는 문서만을 선택하기 때문에 성능을 올릴 수 있을 것이라고 생각.

Experiment

Results

- Test score기준 가장 높은 성능
- MEGClass 논문에서는 RoBERTa-base이지만, MixText는 bert기반 구조였기 때문에 BERT-base MEGClass와 성능 비교
- MixText를 추가하여 학습 시켰을 때, accuracy 향상

Accuracy	NYT-fine	20News	yelp
MEGClass – 0.5	0.7369	0.6117	0.8543
+ MitText – 0.5	0.8774	0.7167	0.869
MEGClass – 0.2	0.8510	0.5617	0.8533
+ MitText – 0.2	0.8799	0.7322	0.8886

Experiment

Results

- F1_macro/f1_micro 성능 향상
- BERT-base의 MixText 성능이 RoBERTa-base의 MEGClass 성능과 유사
- K값에 따른 스코어는 데이터 도메인 별로 상이

F1_macro/f1_micro	NYT-fine	20News	yelp
MEGClass – 0.5	0.5469/0.7369	0.4729/0.6117	0.8543/0.8543
+ MitText – 0.5	0.7233/0.8774	0.5353/0.7221	0.8689/0.869
MEGClass – 0.2	0.5455/0.8510	0.4185/0.5617	0.8533/0.8533
+ MitText – 0.2	0.6128/0.8799	0.5449/0.7332	0.8884/0.8886
MEGClass (RoBERTa-base)	0.7248/0.8813	0.8038/0.8124	0.8554/0.8554
논문 성능	0.7106/0.8924	0.8063/0.8172	0.8741/0.8741

Future works

Discussion

- 여러 번의 성능 측정 및 더 큰 데이터셋으로 실험
- RoBERTa-base의 MixText로 코드 수정한 뒤 실험
- Ablation study를 통해 MixText 학습 과정의 유의미함을 확인
- Explainable적인 요소를 모델에 추가할 수 있을지 구상

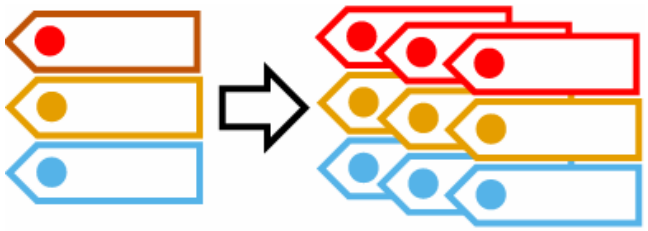
고맙습니다

Appendix

대표적인 XWS-TC 기법 : X-Class(2021)

Seed matching methods

① 클래스 표현 확장 및 추정
② 문서 표현 생성
③ 문서의 클래스를 정렬 – 의사 라벨 부여
④ 분류기 fine-tuning

① 

Expand seed words

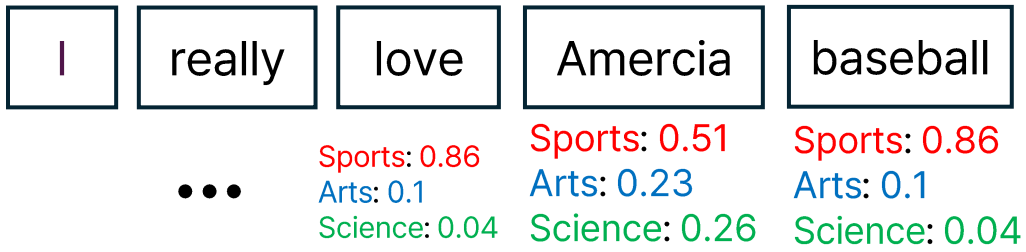
{Sports} -> {Sports : play, teams, soccer, rule....}

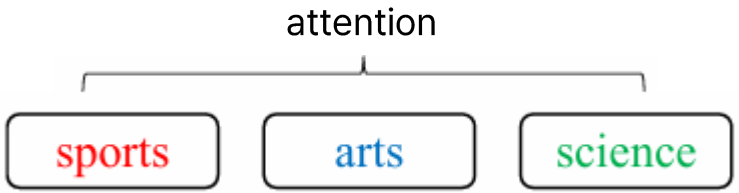
0.323 0.323 0.1 0.4 0.142 0.37 (LLM을 통한 정적 임베딩 값)

Average

0.267

② 각 단어에 클래스별 가중치 계산 -> weighted average 하여 문서 표현 생성





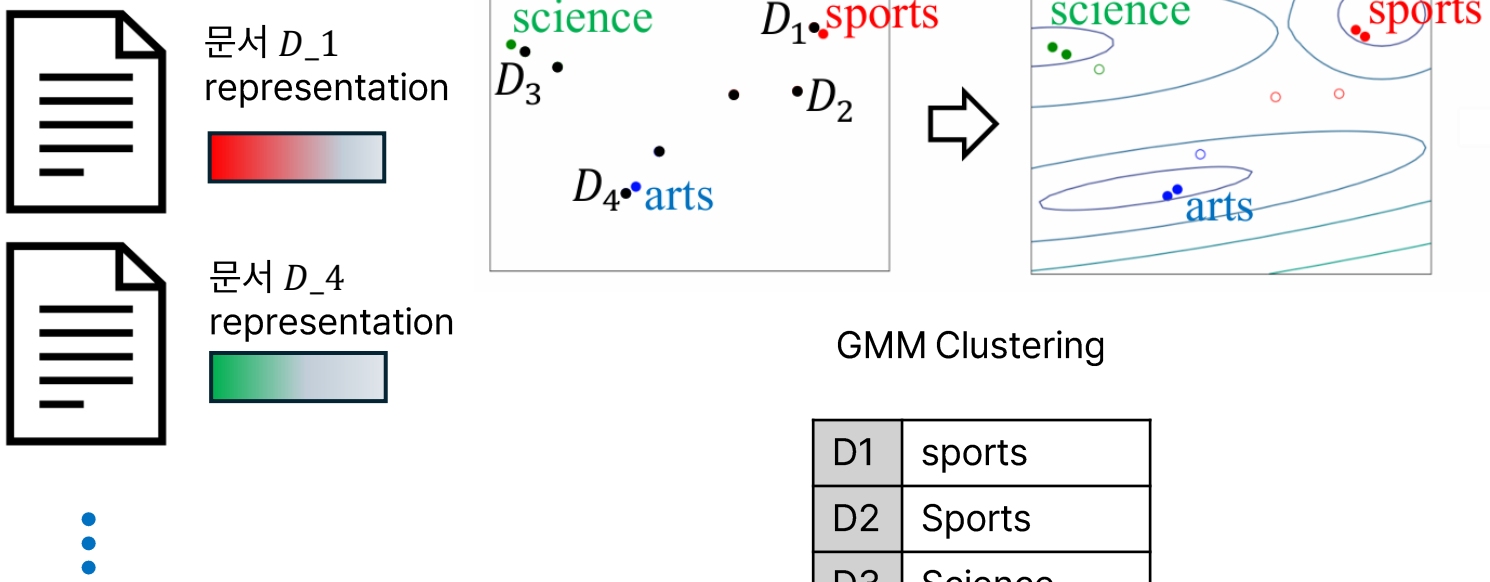
Appendix

대표적인 XWS-TC 기법 : X-Class(2021)

Seed matching methods

① 클래스 표현 확장 및 추정
② 문서 표현 생성
③ 문서의 클래스를 정렬 – 의사 라벨 부여
④ 분류기 fine-tuning

③



GMM Clustering

D1	sports
D2	Sports
D3	Science

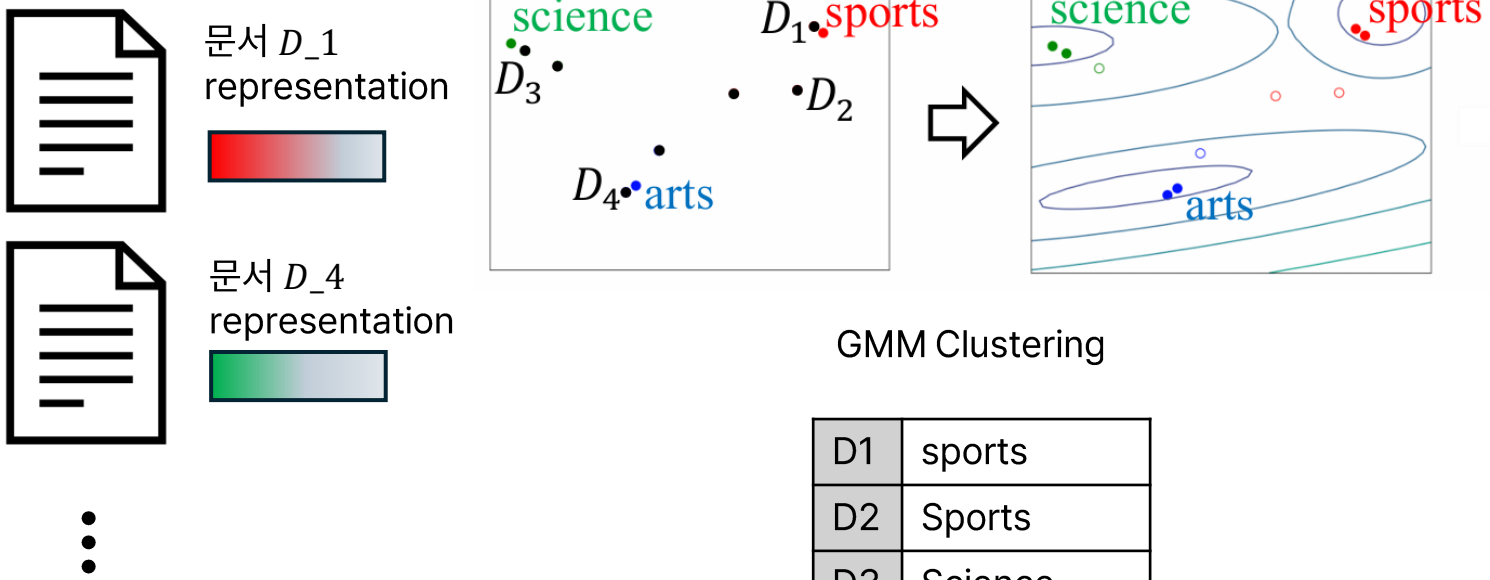
Appendix

대표적인 XWS-TC 기법 : X-Class(2021)

Seed matching methods

① 클래스 표현 확장 및 추정
② 문서 표현 생성
③ 문서의 클래스를 정렬 – 의사 라벨 부여
④ 분류기 fine-tuning

③



D1	sports
D2	Sports
D3	Science

MEGClass(2023)

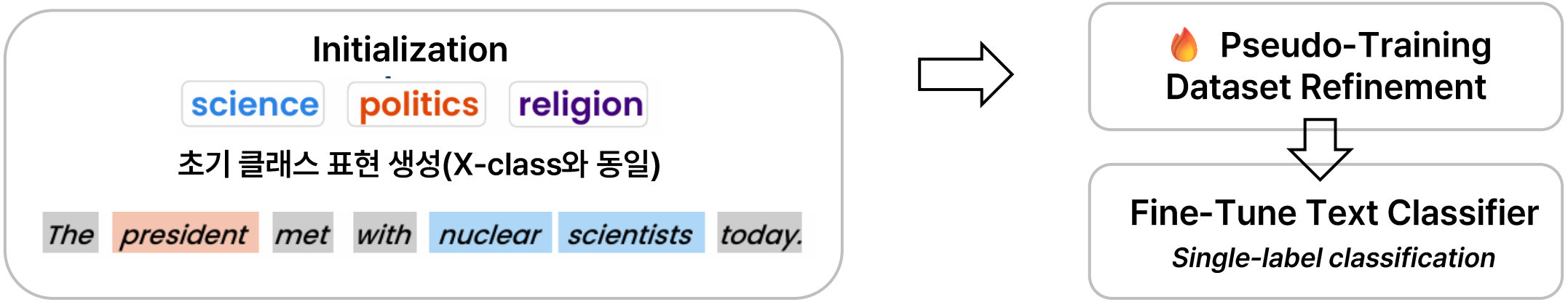
오직 단어의 표현으로 클래스와 문서를 표현할 수 없다.

Mutually-Enhancing Text Granularities
(입자, 낟알 -> 더 세분화)

문서와 클래스 표현 과정을 더욱더 정교화 하겠다!

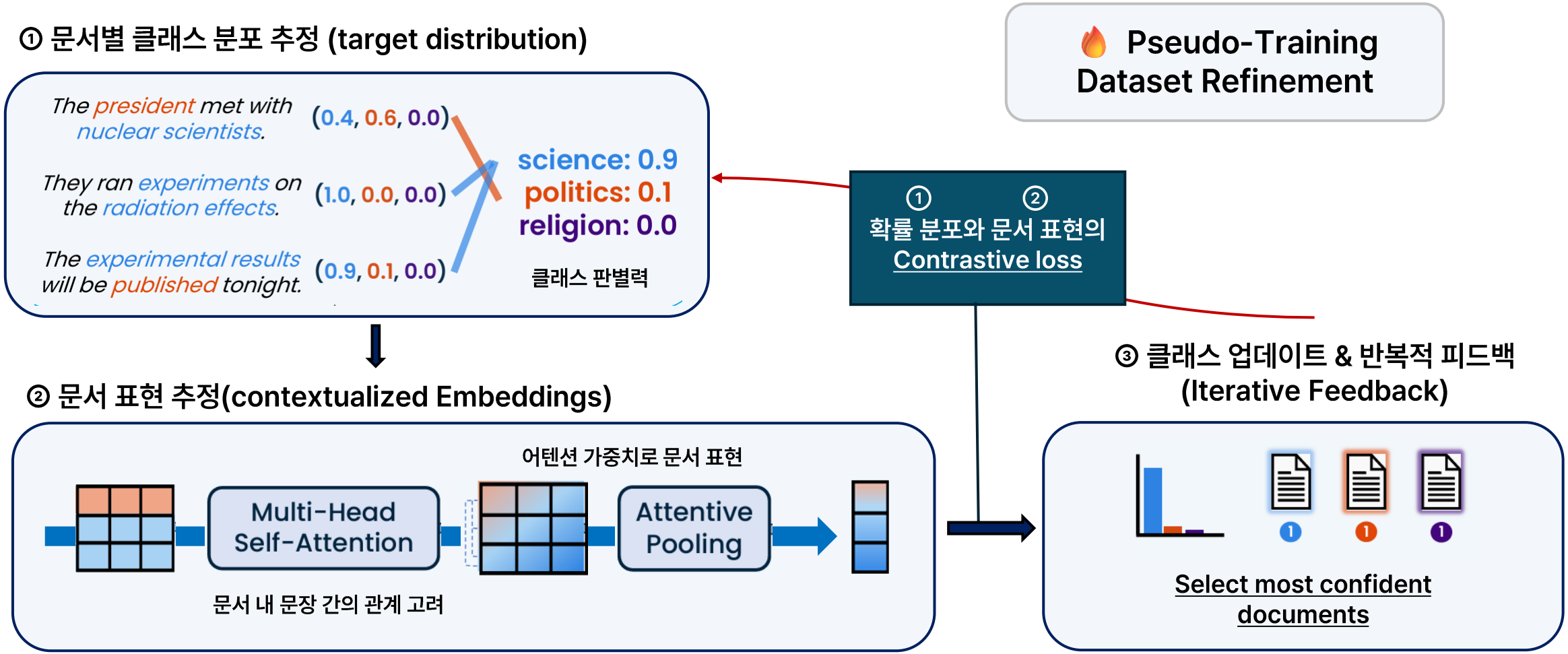
Appendix

MEGClass(2023)



Appendix

MEGClass(2023)



Appendix - MEGClass

1. 문서별 클래스 분포 추정

- 가장 단순한 문서 분포 추정 방법 : 문장의 임베딩 값을 평균내어 문서를 표현하고, 클래스 표현과 코사인 유사도 계산
  → 모든 문장이 동일한 중요도를 가진다는 가정
- 문장이 주는 정보성을 고려하자 : 클래스 판별력(class – discriminative)

	야구	농구	축구	달리기	판별력
ex) The player got the ball → 여러 스포츠에 해당(낮은 가중치)	0.46	0.32	0.16	0.04	→ 0.42
ex) He hit a home run. → 야구에만 해당함(높은 가중치)	0.89	0.03	0.07	0.01	→ 0.88

- ❖ 가장 유사한 클래스(top class): q_j^0
- ❖ 두 번째로 유사한 클래스(second class): q_j^1
- ❖ 클래스 판별력(Class Gap): q_j^0 - q_j^1

$$s_{j,weight} = \frac{q_j^0 - q_j^1}{\sum_{l=1}^{|d_i|}(q_l^0 - q_l^1)}$$

문서 개수 i, 문장 개수 j

Appendix - MEGClass

1. 문서별 클래스 분포 추정

- 클래스 판별력을 문장 가중치로 하여, 클래스별 문서 분포 추정

	science	politics	religion	판별력
The president met with nuclear scientists.	0.4	0.6	0.0	0.6
They ran experiments on the radiation effects.	1.0	0.0	0.0	1.0
The experimental results will be published tonight.	0.9	0.1	0.0	0.8

$$P(d_i \in C_k) = \sum_{s_j \in d_i} s_{j,weight} \cdot \cos(s_j, c_k)$$

P(문서1 ∈ science) = 0.4*0.6 + 1.0*1.0 + 0.9*0.8 = 1.96
P(문서1 ∈ politics) = 0.6*0.6 + 0.0*1.0 + 0.1*0.8 = 0.44
P(문서1 ∈ science) = 0.0*0.6 + 0.0*1.0 + 0.0*0.8 = 0

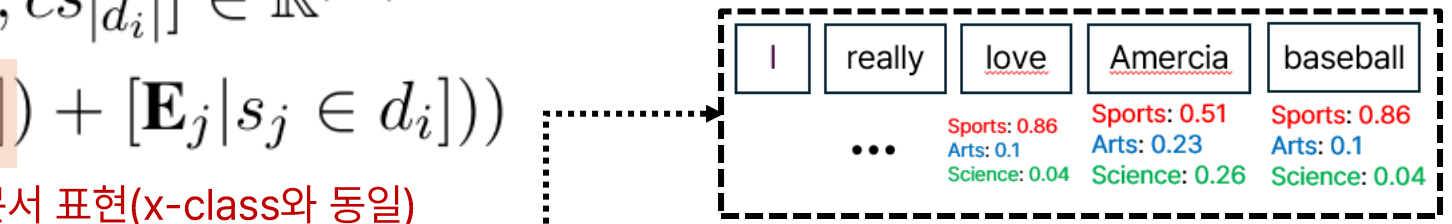
Appendix - MEGClass

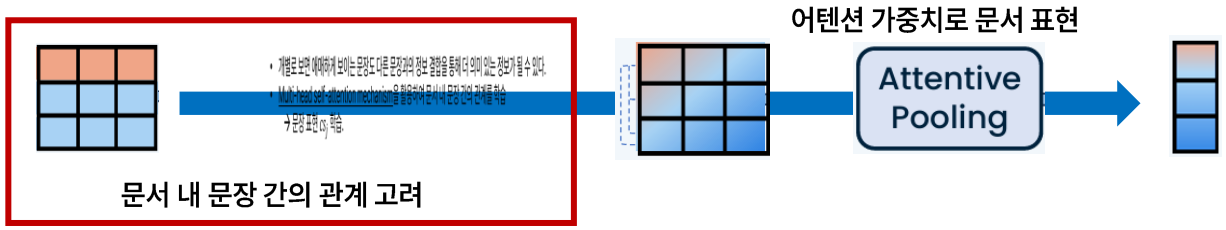
2. 문서 표현 추정(Contextualized Embeddings)

- 개별로 보면 애매하게 보이는 문장도 다른 문장과의 정보 결합을 통해 더 의미 있는 정보가 될 수 있다.
- Multi-head self-attention mechanism을 활용하여 문서 내 문장 간의 관계를 학습
  → 문장 표현 cs_j 학습.

$$CS(d_i) = [cs_1, cs_2, \ldots, cs_{|d_i|}] \in \mathbb{R}^{|d_i| \times h_{cs}}$$
$$= l_{ln}(l_{mhs}([\mathbf{E}_j | s_j \in d_i]) + [\mathbf{E}_j | s_j \in d_i]))$$

초기 클래스 표현에 대한 문서 표현(x-class와 동일)
X-class에서는 단어 수준의 정보만을 활용하였지만,
MEGClass는 mutli-head self-attention layer와 layer normalization을 통해 문서 수준의 정보를 동시에 활용





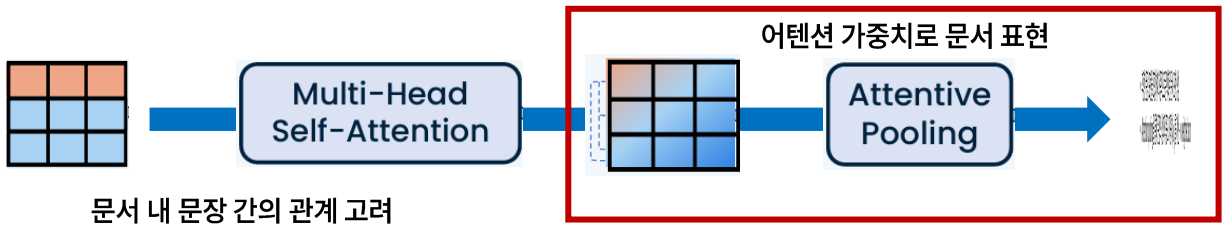
Appendix - MEGClass

2. 문서 표현 추정(Contextualized Embeddings)

- 추정한 문장 표현을 조합하여 최종적으로 문맥화된 문서 표현 생성
- Attention pooling을 통해 문장의 상대적 중요 가중치(α_j)를 계산 -> weighted sum

$$CD(d_i) = cd_i = \sum_{j=1}^{|d_i|} \alpha_j cs_j \in \mathbb{R}^{h_{cd}}$$

$$= \sum_{j=1}^{|d_i|} \frac{e^{(l_\alpha(cs_j))}}{\sum_{k=1}^{|d_i|} e^{(l_\alpha(cs_k))}} cs_j$$



Appendix - MEGClass

3. 클래스와 문서 정렬

- 다음 weighted regularized contrastive loss를 최소화하여 클래스 별 문서 분포를 업데이트

$$\mathcal{L}_{WCon}(d_i, C, P(d_i)) = -\sum_{k=1}^{|C|} P(d_i \in C_k) \times \log \frac{e^{(\cos(\mathbf{cd}_i, \mathbf{c}_k))/\tau}}{\sum_{c_n \in C} e^{(\cos(\mathbf{cd}_i, \mathbf{c}_n)/\tau)}}$$

i번째 문서와 k번째 클래스와의 코사인 유사도

i번째 문서와 모든 클래스와의 코사인 유사도 합

가중치

K번째 클래스에 대한 i번째 문서의 확률 분포값

If 문서와 클래스의 유사도가 높음
→ 가중치가 커짐
→ loss값을 최소화하기 위해서는 가중치가 큰 것의 확률분포를 크게 하는 것이 유리
→ 해당 클래스의 문서 확률분포 값이 커짐.

Appendix - MEGClass

3. 클래스와 문서 정렬

- 추정한 확률분포를 기반으로 지도학습 데이터 구축
- 클래스별로 상위 k%에 해당하는 문서만 사용

P(문서 ∈ science)

총 문서가 100개, k=50%라면 클래스 별 50개의 문서 선택됨.

Appendix - MEGClass

3. 반복적 피드백 (iterative Feedback)

- 최종적으로 추정된 문서 표현과 클래스 표현 간의 코사인 유사도 값 구함.

→ 신뢰도(threshold)보다 큰 경우 해당 문서 표현을 클래스 집합에 추가

초기 클래스 집합

{**Sports**} -> {**Sports** : play, teams, soccer, rule....}

0.323 0.323 0.1 0.4 0.142 0.37

0.267

1 iter 후

{**Sports** : play, teams, soccer, rule, D_1, D_5..}

0.232 0.45

Update : 0.28814

Appendix

parameters

Parameter	값
Model	Roberta-base
Max_seq_length	128
Batch_size	8
Epoch	3
lr	5e-5

megclass train

Parameter	값
Batch_size	64
Epochs	4
Max_sent	150
Lr	1e-3
K	0.075
Doc_thres	0.5

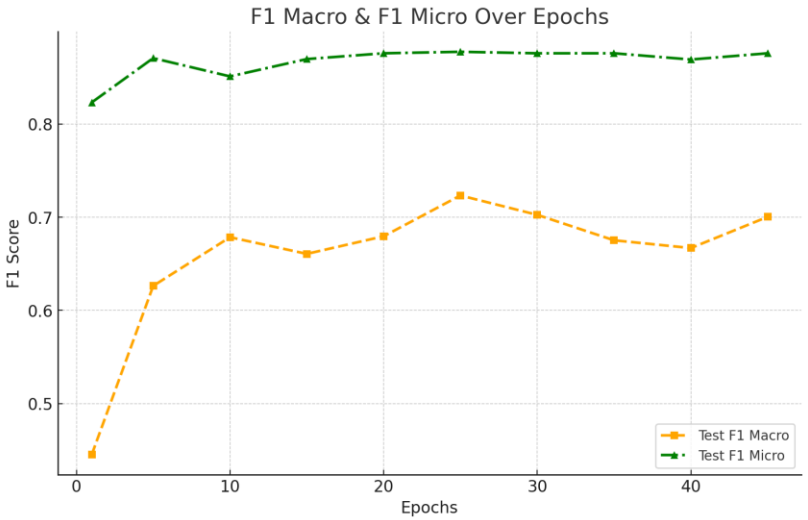
megclass fine-tune

Parameter	값
Batch_size	2
epochs	50
Mix-layers-set	[0,1,2,3]
lr	0.001

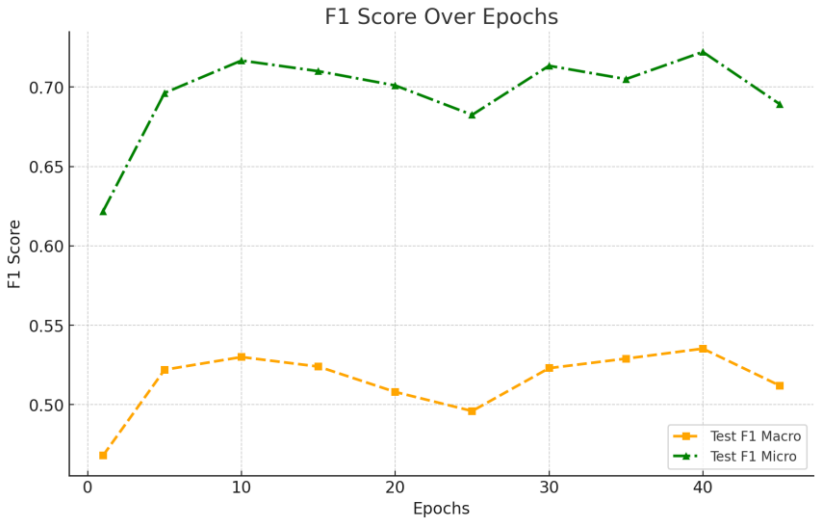
Mixtext train

Appendix

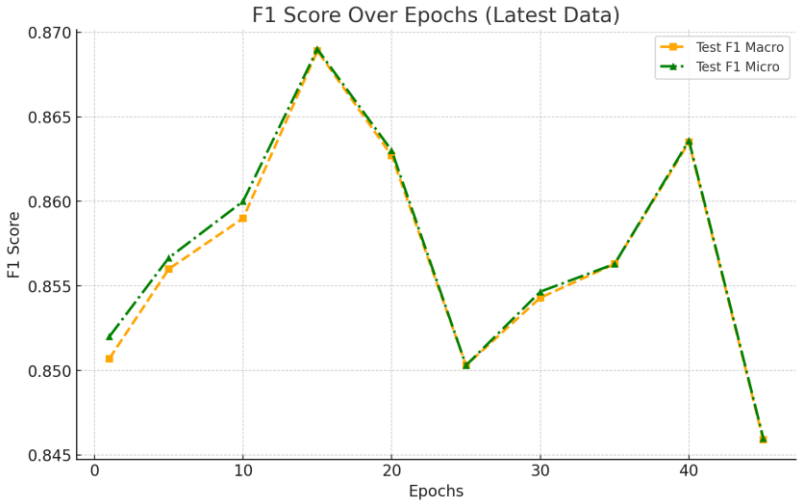
results graph



MEGClass + MixText 0.5
NYT-fine



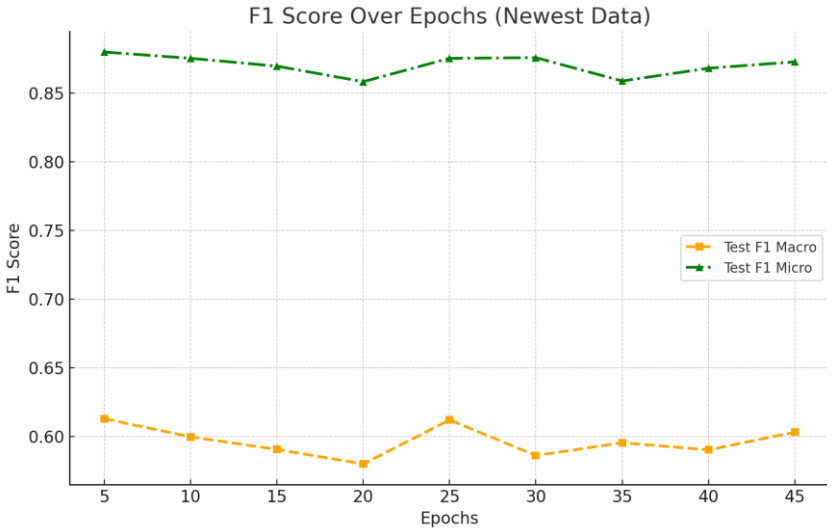
MEGClass + MixText 0.5
20News



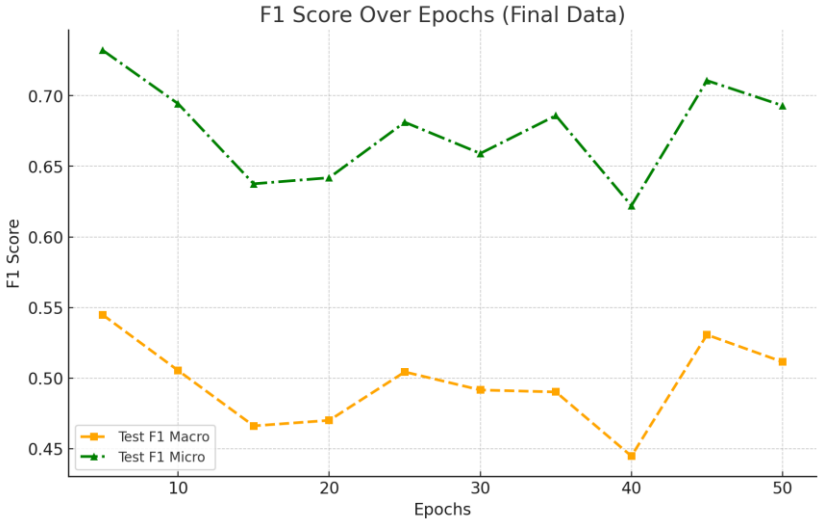
MEGClass + MixText 0.5
Yelp

Appendix

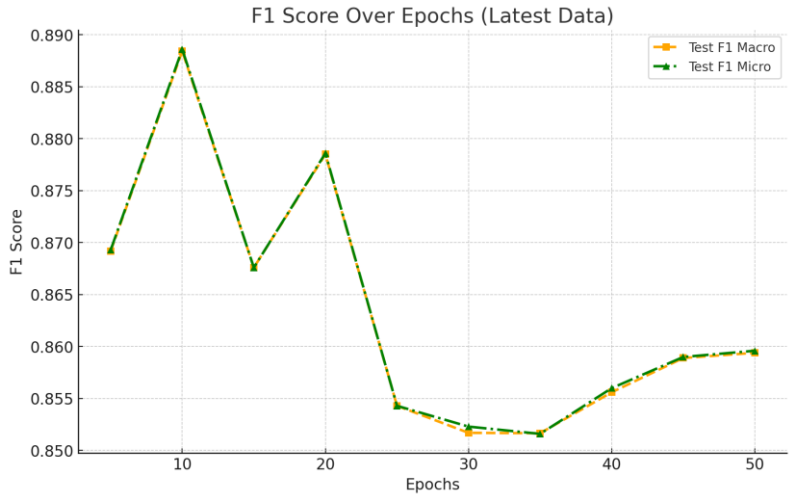
results graph



MEGClass + MixText 0.5
NYT-fine



MEGClass + MixText 0.5
20News



MEGClass + MixText 0.5
Yelp