Domain Adaptation for Robust Question Answering

AIX7023-01

Jiyong Moon[1]

[1]Department of Artificial Intelligence, Dongguk University, Seoul, Korea

CONTENTS

01

Introduction

Robust Question Answering

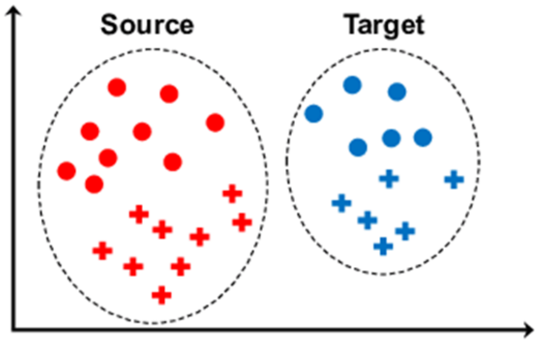
- Question Answering (QA) aims to extract the correct answer span given the context and question
- Pre-trained transformers showed good performance in QA, but this requires a large amount of labeled data
- The model performs well in domains in which it is trained with large amounts of labeled data (**in-domain**)
- But the model performs poorly in domains that share some similarities but are different (**out-of-domain**)
- The model generalizes poorly (**poor robustness**)

Method	Backbone	In-Domain		Out-of-Domain	
		F1	EM	F1	EM
IND-only	TinyBERT	72.15	56.45	49.68	35.08

Causes of Poor Robustness

(1) Domain Shift

- Domain shift refers to the difference in distribution between the model's training data (**source**) and test data (**target**)
- Weak at generalizing learned knowledge

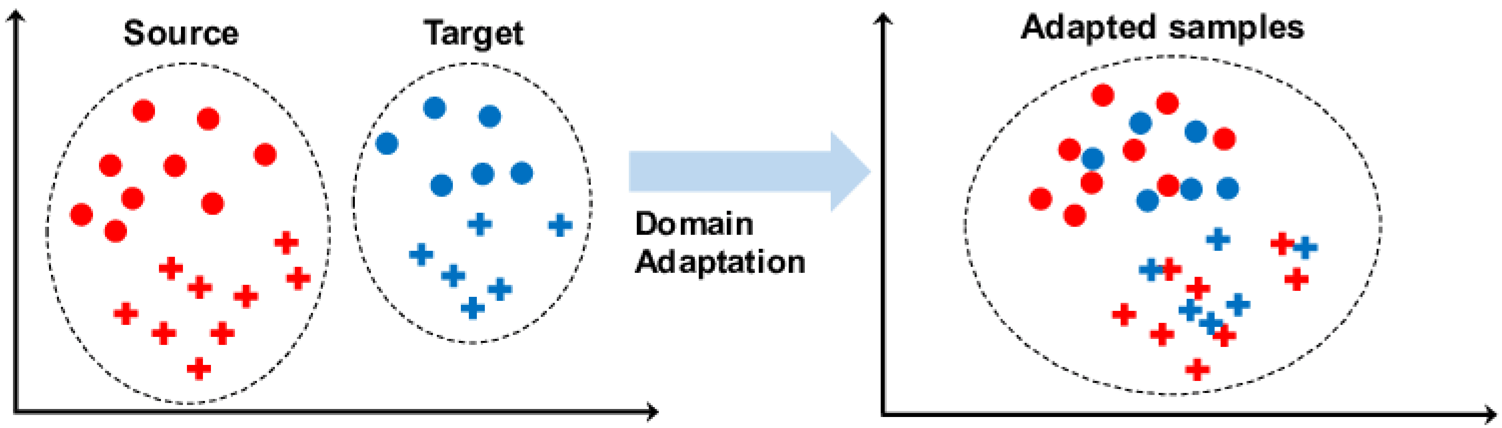


(2) Insufficient Labeled Data

- There are very few labeled samples in the out-of-domain training set (**150,000 ⟩⟩⟩ 381**)
- It is difficult to improve only with supervised tuning for out-of-domain

Domain Adaptation

- In this project, we improve the robustness of QA model by using **domain adaptation** method
- Learn **domain-invariant feature representations**
- Learning a model that can be applied to both source and target domains (**improved generalization**)

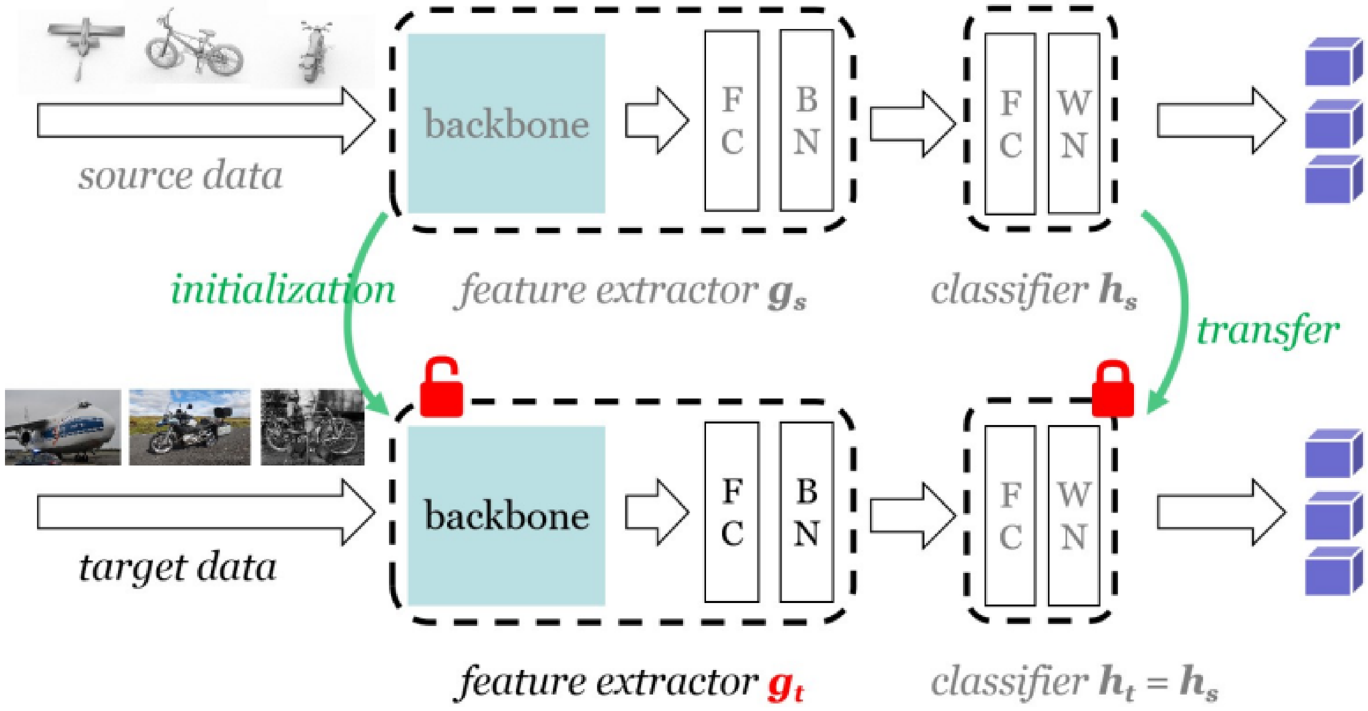


02

Proposed Method

SHOT

- *"Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation", ICML 2020.*

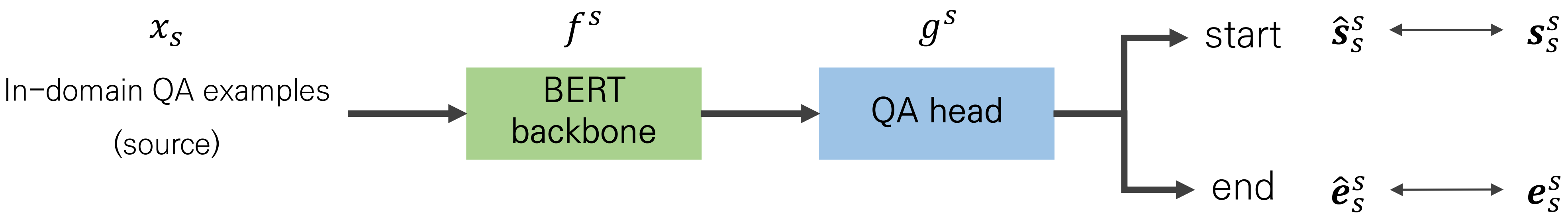


Difference in Scenario

- There are differences between SHOT's scenario and ours
- Therefore, there are some modifications to the implementation

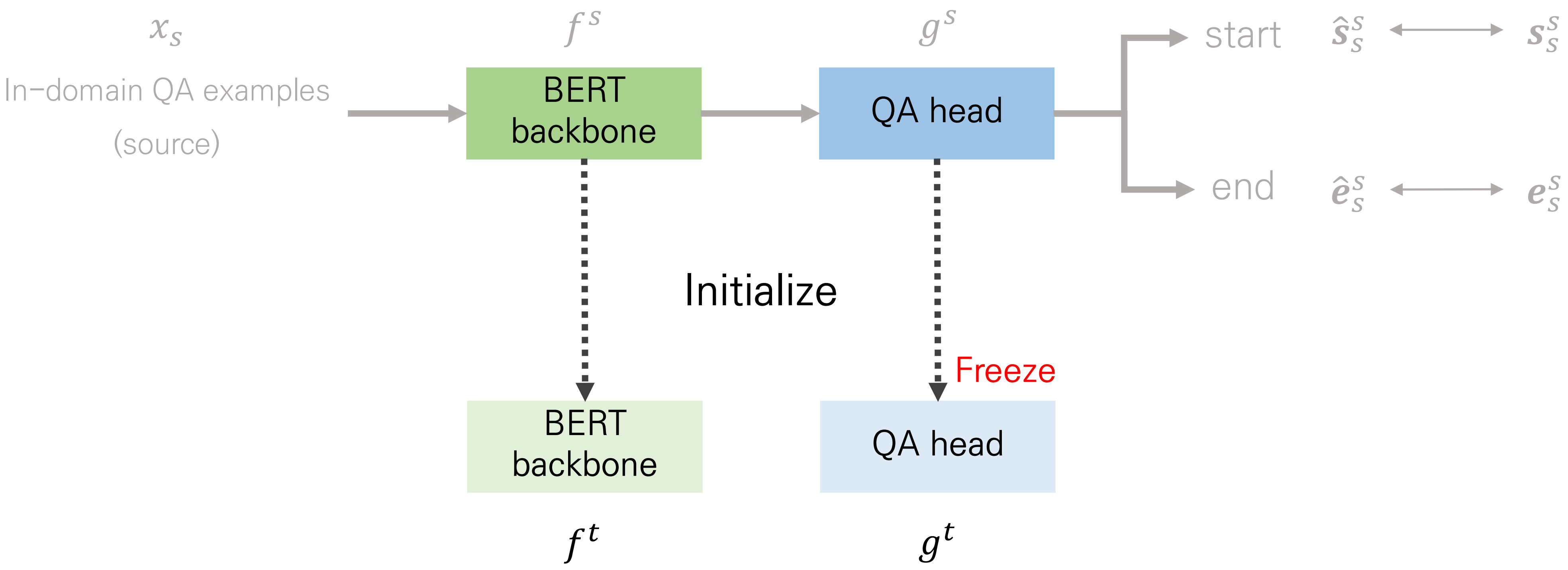
SHOT	Ours
- Unsupervised (No label for target domain) - For image classification (CV)	- Supervised (But very few labels for target data) - For question answering (NLP)

Proposed Method



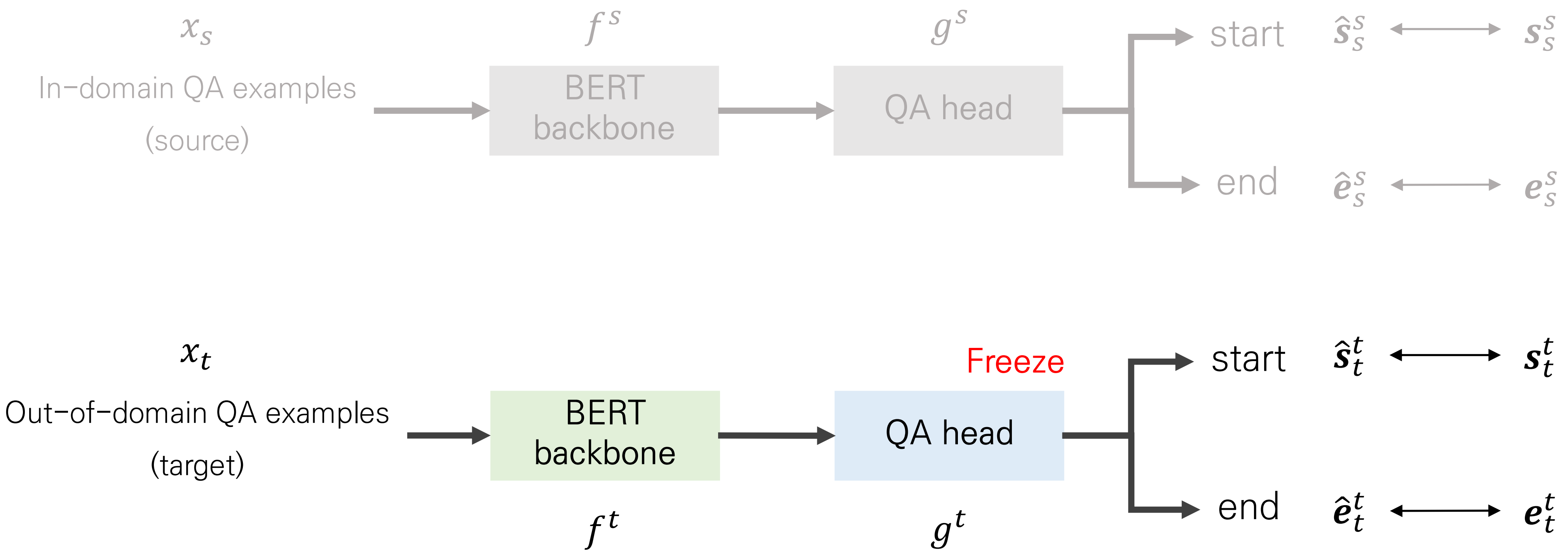
$$\mathcal{L} = CE(\hat{\boldsymbol{s}}_s^s, \boldsymbol{s}_s^s) + CE(\hat{\boldsymbol{e}}_s^s, \boldsymbol{e}_s^s), \qquad where\ CE(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_k^K \boldsymbol{y}_k \log \hat{\boldsymbol{y}}_k$$

Proposed Method



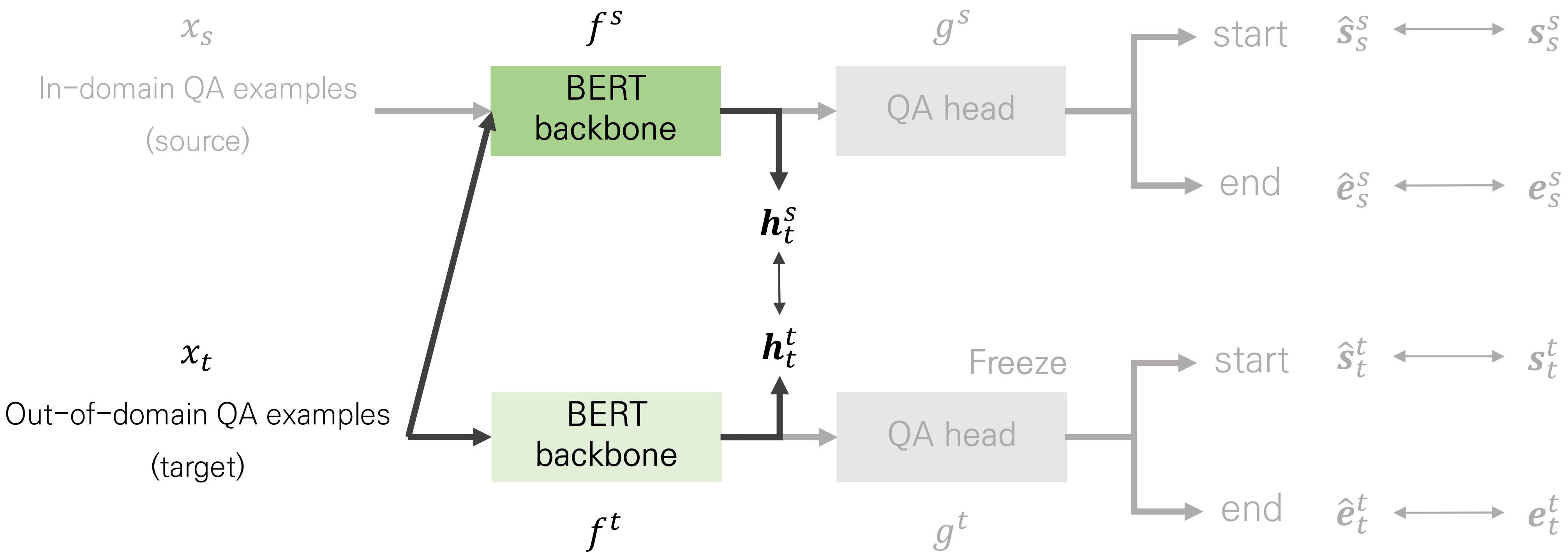
The target model is optimized by **three loss functions**

Proposed Method



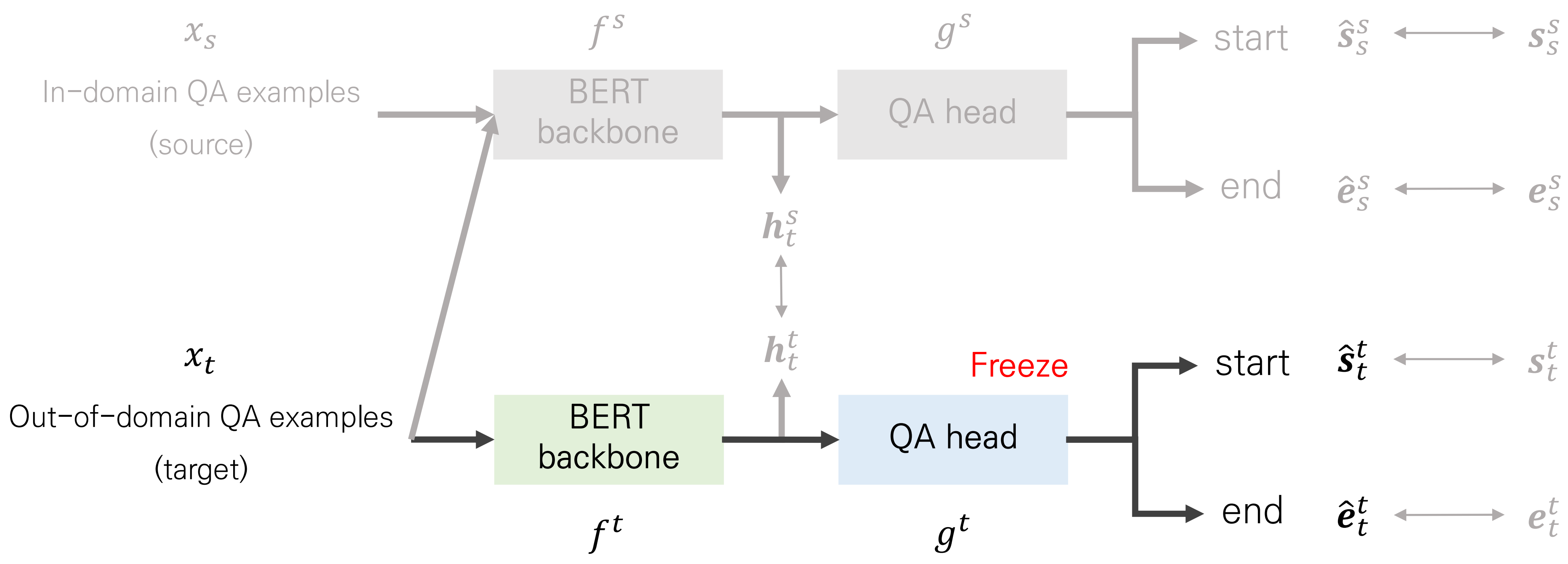
$$\mathcal{L}_{cls} = CE(\hat{\boldsymbol{s}}_t^t, \boldsymbol{s}_t^t) + CE(\hat{\boldsymbol{e}}_t^t, \boldsymbol{e}_t^t), \qquad where\ CE(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_k^K \boldsymbol{y}_k \log \hat{\boldsymbol{y}}_k$$

Proposed Method



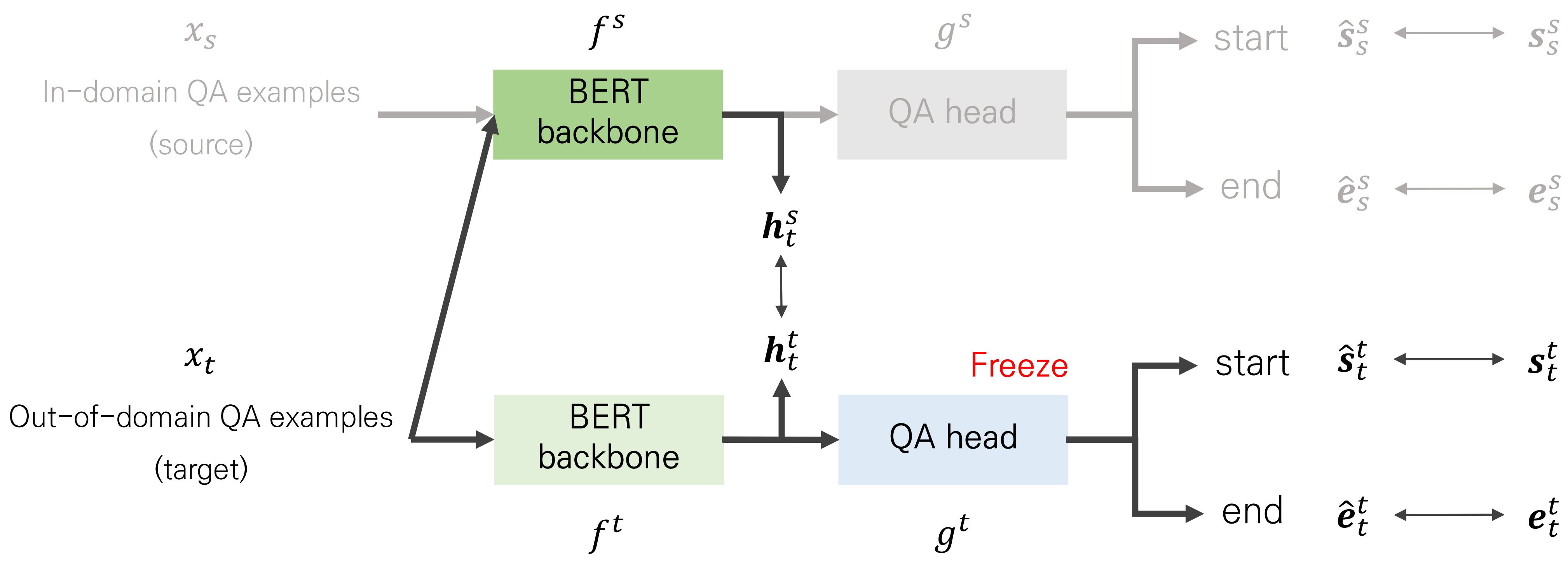
$$\mathcal{L}_{sim} = S(\boldsymbol{h}_t^t, \boldsymbol{h}_t^s), \qquad where\ S(\boldsymbol{u}, \boldsymbol{v}) = 2 - 2\left(\frac{\boldsymbol{u} \cdot \boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|}\right)$$

Proposed Method



$$\mathcal{L}_{ent} = E(\hat{\boldsymbol{s}}_t^t) + E(\hat{\boldsymbol{e}}_t^t), \quad where\ E(\boldsymbol{y}) = -\sum_k^K \boldsymbol{y}_k \log \boldsymbol{y}_k$$

Proposed Method



$$\mathcal{L} = \mathcal{L}_{cls} + \alpha\mathcal{L}_{sim} + \beta\mathcal{L}_{ent}$$

03

Experimental Result

Experimental Evaluation

- Performance comparison with other methods

Table 1: Experimental Evaluation

Method	Backbone	In-Domain		Out-of-Domain	
		F1	EM	F1	EM
IND-only	TinyBERT	72.15	56.45	49.68	35.08
OOD-only	TinyBERT	53.42 (-18.73)	37.40 (-19.05)	43.24 (-6.44)	30.10 (-4.98)
Fine-tuning	TinyBERT	70.40 (-1.75)	54.40 (-2.05)	50.66 (+0.98)	35.34 (+0.35)
Ours	TinyBERT	65.87 (-6.28)	49.16 (-7.29)	52.20 (+2.52)	36.65 (+1.57)

Experimental Evaluation

- Our method records the highest performance on out-of-domain dataset

Table 1: Experimental Evaluation

Method	Backbone	In-Domain		Out-of-Domain	
		F1	EM	F1	EM
IND-only	TinyBERT	72.15	56.45	49.68	35.08
OOD-only	TinyBERT	53.42 (-18.73)	37.40 (-19.05)	43.24 (-6.44)	30.10 (-4.98)
Fine-tuning	TinyBERT	70.40 (-1.75)	54.40 (-2.05)	50.66 (+0.98)	35.34 (+0.35)
Ours	TinyBERT	65.87 (-6.28)	49.16 (-7.29)	52.20 (+2.52)	36.65 (+1.57)

Experimental Evaluation

- However, there is a problem that the performance on in-domain dataset is significantly traded-off

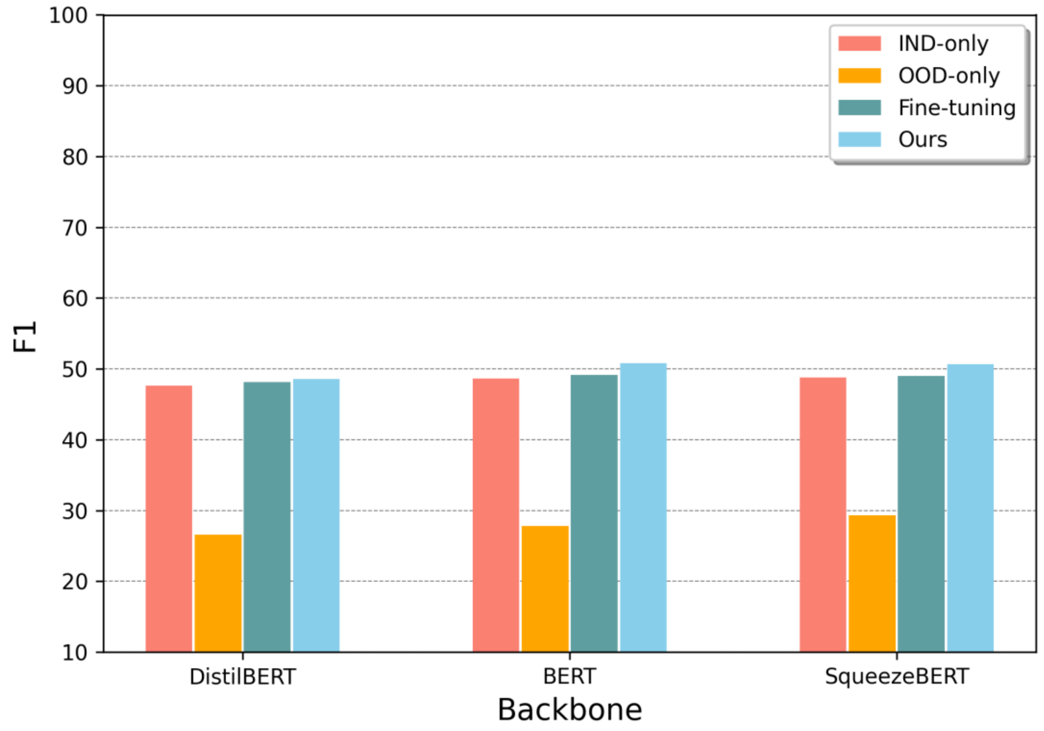
Table 1: Experimental Evaluation

Method	Backbone	In-Domain		Out-of-Domain	
		F1	EM	F1	EM
IND-only	TinyBERT	72.15	56.45	49.68	35.08
OOD-only	TinyBERT	53.42 (-18.73)	37.40 (-19.05)	43.24 (-6.44)	30.10 (-4.98)
Fine-tuning	TinyBERT	70.40 (-1.75)	54.40 (-2.05)	50.66 (+0.98)	35.34 (+0.35)
Ours	TinyBERT	65.87 (-6.28)	49.16 (-7.29)	52.20 (+2.52)	36.65 (+1.57)

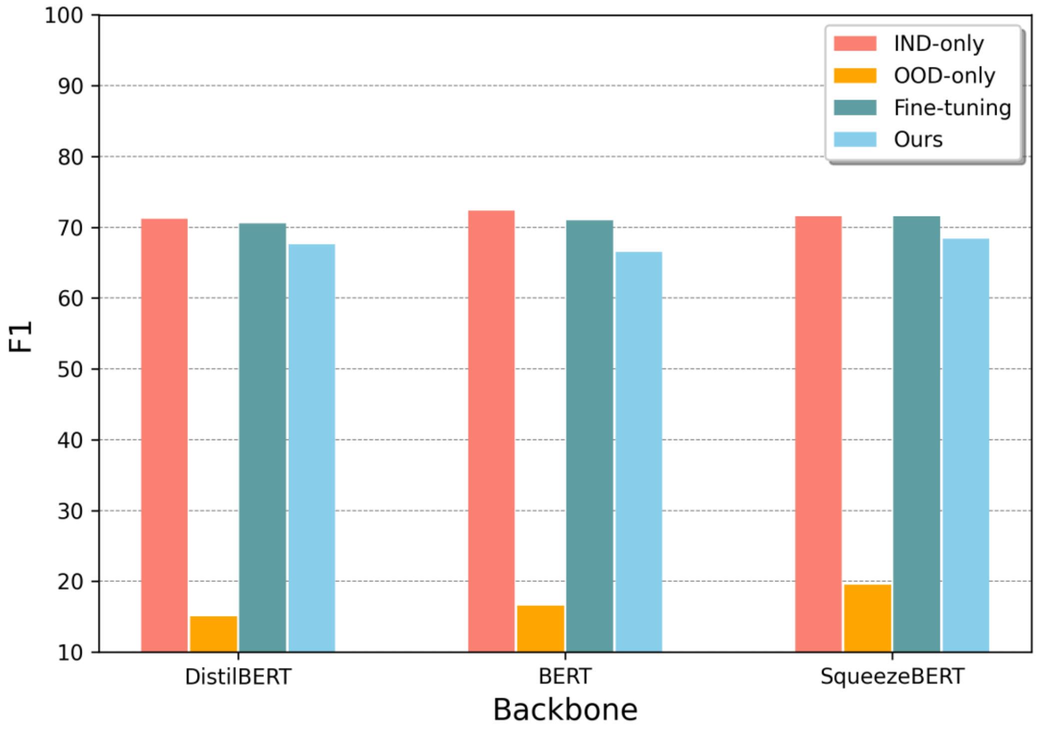
Other Baselines

- Evaluate performance when using DistilBERT, BERT, and SqueezeBERT as backbones

Out-of-domain



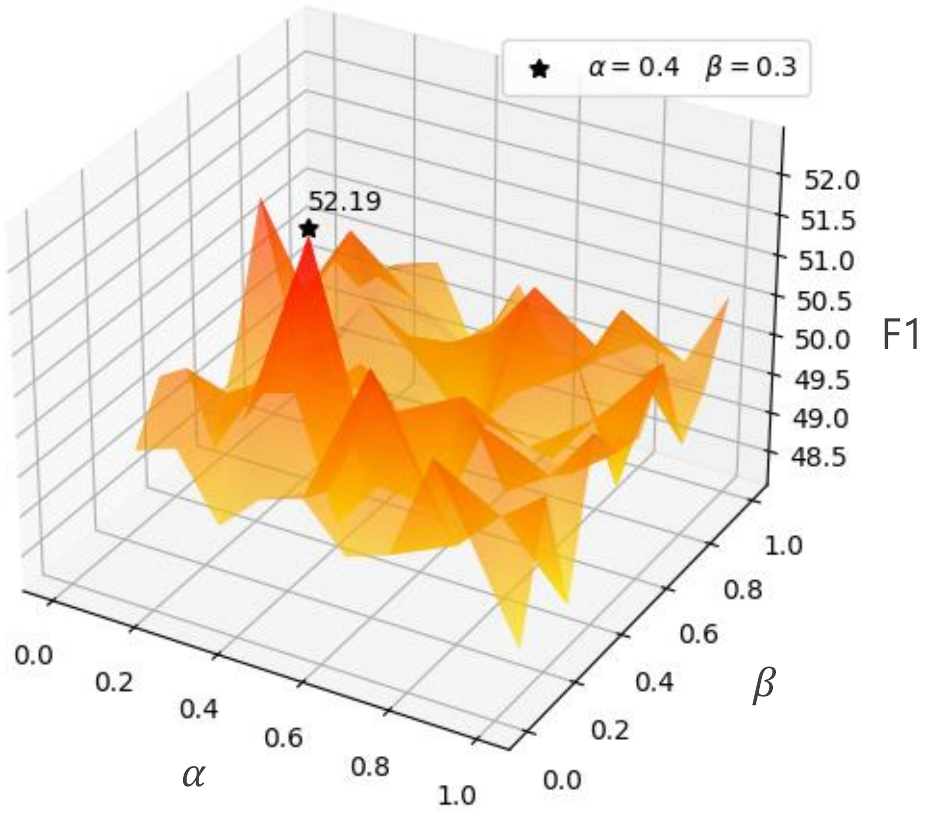
In-domain



Ablation Study: α, β

- Grid search to find the optimal loss weights α, β
- Best when $\alpha = 0.4$ and $\beta = 0.3$

$$\mathcal{L} = \mathcal{L}_{cls} + \alpha\mathcal{L}_{sim} + \beta\mathcal{L}_{ent}$$



Ablation Study: Similarity Measure

- Ablation on other similarity measures

Table 2: Ablation on Similarity Measure

Similarity Measure	Out-of-Domain	
	F1	EM
Cross Entropy	48.41	34.29
KL-divergence	49.67	34.55
Negative Cosine Similarity	49.09	34.55
BYOL (ours)	**52.20**	**36.65**

Cross Entropy: $-\sum_{k}^{K} \varphi(\boldsymbol{h}_t^s)_k \log \varphi(\boldsymbol{h}_t^t)_k$

KL-divergence: $-\sum_{k}^{K} \varphi(\boldsymbol{h}_t^s)_k \log \frac{\varphi(\boldsymbol{h}_t^s)_k}{\varphi(\boldsymbol{h}_t^t)_k}$

Negative Cosine Similarity: $-\left(\frac{\boldsymbol{h}_t^t \cdot \boldsymbol{h}_t^s}{\|\boldsymbol{h}_t^t\|\|\boldsymbol{h}_t^s\|}\right)$

BYOL: $2 - 2\left(\frac{\boldsymbol{h}_t^t \cdot \boldsymbol{h}_t^s}{\|\boldsymbol{h}_t^t\|\|\boldsymbol{h}_t^s\|}\right)$

Ablation Study: Loss Composition

- Analyze the effect of each loss term

Table 3: Ablation on Loss Composition

Loss Composition	In-Domain		Out-of-Domain	
	F1	EM	F1	EM
$\mathcal{L}_{cls}$	65.15	48.33	49.61	33.77
$\mathcal{L}_{cls} + \mathcal{L}_{ent}$	64.09 (−1.06)	47.45 (−0.88)	50.67 (+1.06)	35.60 (+1.83)
$\mathcal{L}_{cls} + \mathcal{L}_{sim}$	67.40 (+2.25)	50.38 (+2.05)	48.49 (−1.12)	32.25 (−1.52)
$\mathcal{L}_{cls} + \mathcal{L}_{sim} + \mathcal{L}_{ent}$ (ours)	65.87 (+0.72)	49.16 (+0.83)	52.20 (+2.59)	36.65 (+2.88)

Ablation Study: Loss Composition

- $\mathcal{L}_{ent}$ strengthens representation learning for the out-of-domain dataset
- But it weakens the representation for in-domain datasets

Table 3: Ablation on Loss Composition

Loss Composition	In-Domain		Out-of-Domain	
	F1	EM	F1	EM
$\mathcal{L}_{cls}$	65.15	48.33	49.61	33.77
$\mathcal{L}_{cls} + \mathcal{L}_{ent}$	64.09 (−1.06)	47.45 (−0.88)	50.67 (+1.06)	35.60 (+1.83)
$\mathcal{L}_{cls} + \mathcal{L}_{sim}$	67.40 (+2.25)	50.38 (+2.05)	48.49 (−1.12)	32.25 (−1.52)
$\mathcal{L}_{cls} + \mathcal{L}_{sim} + \mathcal{L}_{ent}$ (ours)	65.87 (+0.72)	49.16 (+0.83)	52.20 (+2.59)	36.65 (+2.88)

Ablation Study: Loss Composition

- $\mathcal{L}_{sim}$ forces the model to retain learning knowledge from in-domain dataset
- But it inhibits learning on out-of-domain datasets

Table 3: Ablation on Loss Composition

Loss Composition	In-Domain		Out-of-Domain	
	F1	EM	F1	EM
$\mathcal{L}_{cls}$	65.15	48.33	49.61	33.77
$\mathcal{L}_{cls} + \mathcal{L}_{ent}$	64.09 (−1.06)	47.45 (−0.88)	50.67 (+1.06)	35.60 (+1.83)
$\mathcal{L}_{cls} + \mathcal{L}_{sim}$	67.40 (+2.25)	50.38 (+2.05)	48.49 (−1.12)	32.25 (−1.52)
$\mathcal{L}_{cls} + \mathcal{L}_{sim} + \mathcal{L}_{ent}$ (ours)	65.87 (+0.72)	49.16 (+0.83)	52.20 (+2.59)	36.65 (+2.88)

Ablation Study: Loss Composition

- We can use $\mathcal{L}_{ent}$ and $\mathcal{L}_{sim}$ at the same time to get all the advantages
- Learning on out-of-domain datasets is balanced with learning on in-domain datasets

Table 3: Ablation on Loss Composition

Loss Composition	In-Domain		Out-of-Domain	
	F1	EM	F1	EM
$\mathcal{L}_{cls}$	65.15	48.33	49.61	33.77
$\mathcal{L}_{cls} + \mathcal{L}_{ent}$	64.09 (−1.06)	47.45 (−0.88)	50.67 (+1.06)	35.60 (+1.83)
$\mathcal{L}_{cls} + \mathcal{L}_{sim}$	67.40 (+2.25)	50.38 (+2.05)	48.49 (−1.12)	32.25 (−1.52)
$\mathcal{L}_{cls} + \mathcal{L}_{sim} + \mathcal{L}_{ent}$ (ours)	65.87 (+0.72)	49.16 (+0.83)	52.20 (+2.59)	36.65 (+2.88)

04

Conclusion

Conclusion

- Inspired by SHOT, **we propose a domain adaptation method for robust question answering**
- Improving QA performance in out-of-domain dataset
- But the performance on in-domain dataset is traded-off