KOREA UNIVERSITY

Detecting Critical Errors Considering Cross-Cultural Factors in English-Korean Translation

Sugyeong Eo, Jungwoo Lim, Chanjun Park, Dahyun Jung,
Seonmin Koo, Hyeonseok Moon, Jaehyung Seo, Heuiseok Lim

LREC-COLING 2024

Introduction

Recent studies have exhibited remarkable achievements in machine translation (MT), **overcoming language barriers for a broad spectrum of users**

Introduction

Recent studies have exhibited remarkable achievements in machine translation (MT), **overcoming language barriers for a broad spectrum of users**

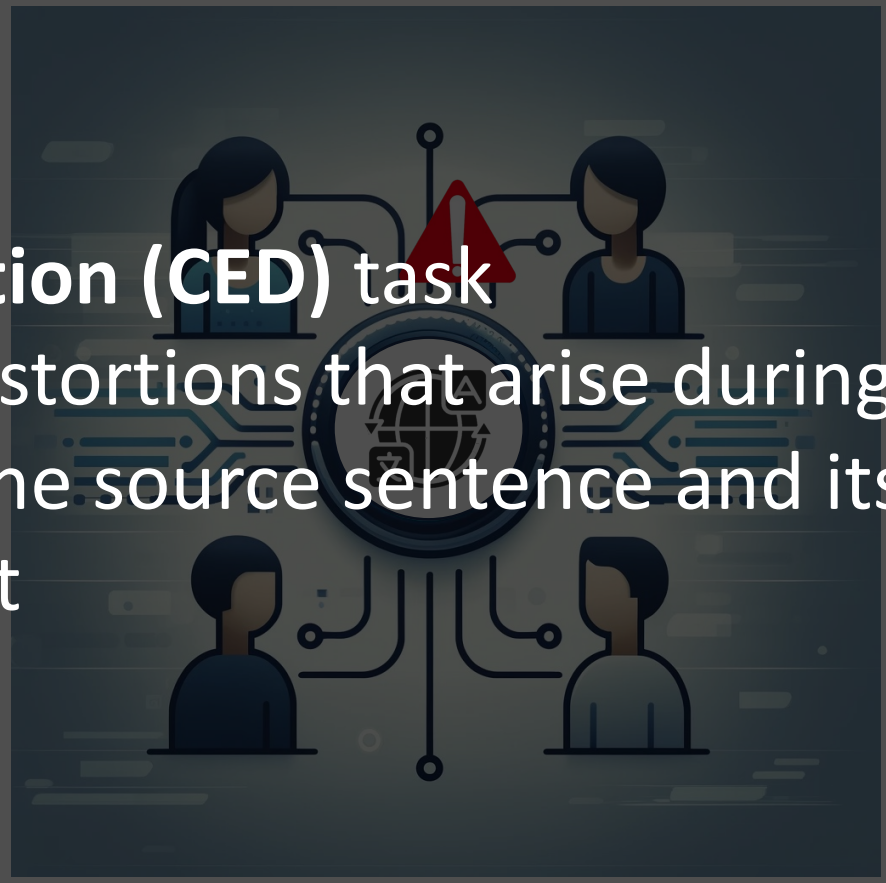
→ Yet, MT output is under a risk of catastrophic meaning deviations

Introduction

Recent studies have exhibited remarkable achievements in machine translation (MT), overcoming language barriers for a broad spectrum of users

→ Yet, MT output is under a risk of catastrophic meaning deviations

Critical Error Detection (CED) task
: aims at identifying the meaning distortions that arise during the translation process by referring to the source sentence and its MT output



Introduction

- CED task details:
  - Task design: binary classification
  - Critical error cases: toxicity, safety, named entity, sentiment, number

Those risky cases rarely emerge

But,

A single fatal error may incur devastating consequences in daily life
(serious ethical, social, financial, legal, etc. issues)

→ The significance of the task is emphasized

Introduction

- However, cultural elements remain overlooked

- Honorific expressions in Korean
  - Honorific forms: (pro)nouns, verbs, suffixes, affixes, etc.
  - Honorific hierarchy: degree of formality/politeness

	Speech-level	Formality
Higher Levels	Hasoseo-che(하소서체)	Very high
	Hasipsio-che(하십시오체)	High
Middle Levels	Haeyo-che(해요체)	Low
	Hao-che(하오체)	High
	Hage-che(하게체)	Neutral
Lower Levels	Haera-che(해라체)	High
	Hae-che(해체)	Low

Introduction

- We introduce KNOTICED dataset

  (1) A critical error detection dataset for English–Korean MT

  (2) A new “politeness” error type
  : addressing honorific issues

  (3) Fine-grained error type labels
  : enabling two types of tasks

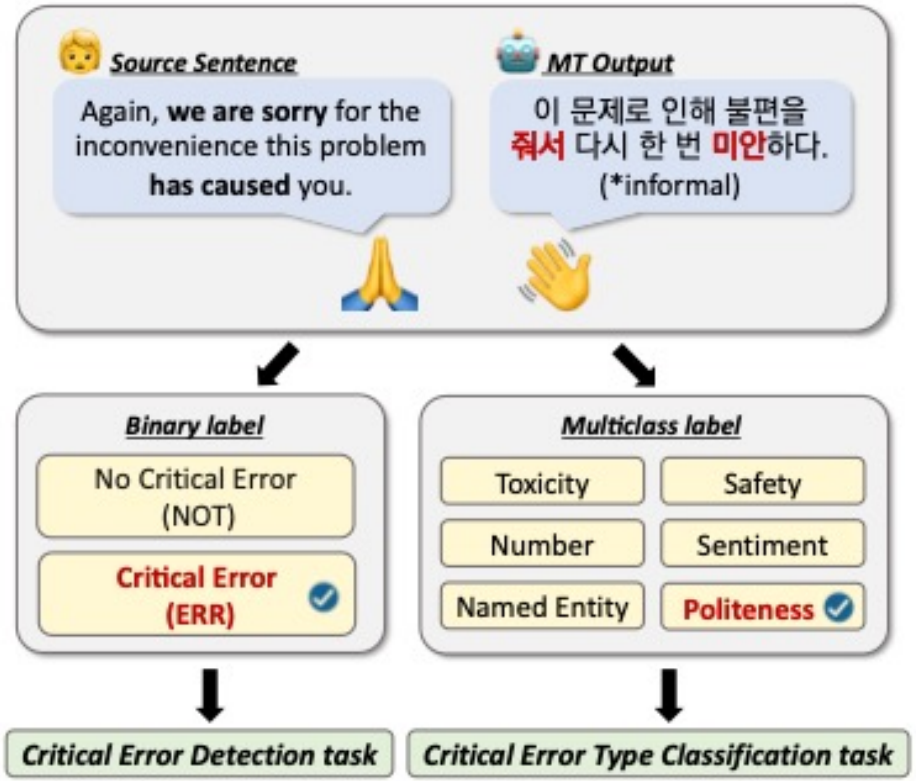
  [1] Critical error detection – binary classification
  [2] Critical error type classification (CETC) – multiclass classification

Dataset Construction

- KNOTICED dataset sample

- Dataset comprises:

  - Source sentence
  - MT output
  - Critical error label(binary class)
  - Critical error label(multiclass)



Dataset Construction

- KNOTICED dataset generation process

(1) Source data selection and refinement
→ AIHUB Daily conversation and colloquial translation corpus

(2) Critical error injection
→ Schema: Toxicity, Safety and health, Named entity, Number, Sentiment, Politeness

(3) Quality evaluation
→ Filtering out instances where all annotators(no overlap with data generator) disagree with the generated example

Dataset Construction

- Schema for KNOTICED

- **Toxicity(TOX):** meaning distortion poses a risk related to religion, gender, race (e.g. n-word, profanity)
- **Safety and health(SAF):** meaning distortion could potentially lead to safety or health problems (e.g. instructions for medicines directly related to human life)
- **Named Entity(NAM):** error impede the restoration of named entities (e.g. names, organization, places)

Dataset Construction

- Schema for KNOTICED

- **Number(NUM):** distortion in quantities or units (e.g. potential issues in economic, financial, legal scenarios)
- **Sentiment(SEN):** polarity of the sentiment is changed
- **Politeness(POL):** improperly judge the degree of honorifics according to the context (translation is perceived as impolite due to informal expressions)

Experiments

Two tasks

- **Critical error detection(CED):**
  - Binary classification
  - Source sentence | MT Output → detecting critical errors “NOT/ERR”

- **Critical error type classification(CETC):**
  - Multiclass classification
  - Source sentence | MT Output → classifying critical error types “TOX/SAF/NAM/NUM/SEN/POL/NOT”

Experiments – Proposed Methods

Augmentation via Perturber

:A simple yet effective data augmentation method

- Perturber: a generative model that produces MT output containing critical errors from an input source sentence

  - Perturber model training:
    (1) We extract instances with the “ERR” label from KNOTICED
    (2) We then train Transformer model to generate MT output containing critical errors by feeding the source sentence
  - Inefficient data size:
    - We leverage En-Ko translation model

Experiments – Proposed Methods

Augmentation via Perturber

- Label assigned to the perturbed data
  - CED task
    (1) **+Pert (All Err) :** All samples to the "ERR" label
    (2) **+Pert (ChatGPT) :** ChatGPT annotation
  - CETC task
    (1) +Pert (ED) : Equal assign (7 labels)
    (2) +Pert (TD) : Distribution-aware labeling (especially training data distribution)
    (3) +Pert (ChatGPT) : ChatGPT annotation

Experiments – Baseline

- PLMs
  - mBERT
  - mBART
  - mBART50
  - XLM-R-base
  - XLM-R-large

- LLMs
  - ChatGPT-Plain : asking whether critical error exists
  - ChatGPT-Demo : Plain+(task samples per critical error type)
  - ChatGPT-Description : Plain+(task description with detailed critical error schema)

Experiments - Results

(1) CED task

Method	Model	Eval				Test			
		MCC	F1-Bad	F1-Good	F1-Multi	MCC	F1-Bad	F1-Good	F1-Multi
Baseline	mBERT (Devlin et al., 2019)	0.1520	0.9042	0.2478	0.2240	0.1227	0.9160	0.2000	0.1832
	mBART25 (Liu et al., 2020)	0.5605	0.9597	0.5542	0.5319	0.4793	0.9636	0.4742	0.4570
	mBART50 (Tang et al., 2021)	0.5433	0.9587	0.5250	0.5033	0.4459	0.9672	0.4364	0.4220
	XLM-R-base (Conneau et al., 2020)	0.2917	0.9377	0.3294	0.3089	0.2484	0.9566	0.2679	0.2562
	XLM-R-large (Conneau et al., 2020)	0.7239	0.9726	0.7126	0.6931	0.5497	0.9719	0.5470	0.5316
	ChatGPT-Demo	0.2088	0.9398	0.2000	0.1880	0.2428	0.9583	0.2476	0.2373
	ChatGPT-Plain	0.2596	0.9368	0.2927	0.2742	0.3544	0.9654	0.2826	0.2728
	ChatGPT-Description	0.0962	0.9050	0.1905	0.1724	0.3633	0.9653	0.3125	0.3017
Ours	XLM-R-large (+Pert (All Err))	**0.7352**	**0.9736**	**0.7391**	**0.7196**	0.5709	0.9735	0.5455	0.5310
	XLM-R-large (+Pert (ChatGPT))	0.6598	0.9670	0.6667	0.6447	**0.6019**	**0.9752**	**0.5607**	**0.5468**

- Our result outperforms PLM, ChatGPT results
- Our approach using perturber showed positive impact
- We achieve additional performance gain by incorporating ChatGPT's linguistic knowledge

Experiments – Results

(2) CETC task

Method	Model	Eval ACC	Test ACC	F1-score per label						
				TOX	SAF	NAM	NUM	SEN	POL	NOT
Baseline	mBERT (Devlin et al., 2019)	0.872	0.915	0.00	0.00	0.00	0.29	0.18	0.00	0.96
	mBART25 (Liu et al., 2020)	0.908	0.934	0.33	0.25	0.17	0.35	0.32	0.56	0.97
	mBART50 (Tang et al., 2021)	0.882	0.914	0.40	0.12	0.00	0.22	0.11	0.12	0.96
	XLM-R-base (Conneau et al., 2020)	0.886	0.926	0.31	0.00	0.24	0.24	0.38	0.50	0.96
	XLM-R-large (Conneau et al., 2020)	0.918	0.945	**0.67**	0.15	**0.31**	0.40	0.58	0.61	0.97
	ChatGPT-Demo	0.886	0.910	**0.05**	0.17	0.00	0.15	0.00	0.00	0.96
	ChatGPT-Plain	0.902	0.924	0.00	**0.31**	0.00	0.15	0.12	0.00	0.96
	ChatGPT-Description	0.878	0.928	0.00	**0.31**	0.00	**0.40**	**0.13**	**0.12**	**0.97**
Ours	XLM-R-large (+Pert (ED))	0.912	0.949	0.36	**0.43**	0.24	0.67	0.57	0.58	**0.98**
	XLM-R-large (+Pert (TD))	0.916	0.949	0.62	0.17	**0.31**	0.70	0.54	0.60	**0.98**
	XLM-R-large (+Pert (ChatGPT))	**0.930**	**0.953**	0.55	0.35	0.26	**0.74**	**0.72**	**0.67**	**0.98**

- Consistent with the results from the CED task, our method outperforms all other baselines

Experiments – Results

(2) CETC task

Method	Model	Eval ACC	Test ACC	F1-score per label						
				TOX	SAF	NAM	NUM	SEN	POL	NOT
Baseline	mBERT (Devlin et al., 2019)	0.872	0.915	0.00	0.00	0.00	0.29	0.18	0.00	0.96
	mBART25 (Liu et al., 2020)	0.908	0.934	0.33	0.25	0.17	0.35	0.32	0.56	0.97
	mBART50 (Tang et al., 2021)	0.882	0.914	0.40	0.12	0.00	0.22	0.11	0.12	0.96
	XLM-R-base (Conneau et al., 2020)	0.886	0.926	0.31	0.00	0.24	0.24	0.38	0.50	0.96
	XLM-R-large (Conneau et al., 2020)	0.918	0.945	**0.67**	0.15	**0.31**	0.40	0.58	0.61	0.97
	ChatGPT-Demo	0.886	0.910	**0.05**	0.17	0.00	0.15	0.00	0.00	0.96
	ChatGPT-Plain	0.902	0.924	0.00	**0.31**	0.00	0.15	0.12	0.00	0.96
	ChatGPT-Description	0.878	0.928	0.00	**0.31**	0.00	**0.40**	**0.13**	**0.12**	**0.97**
Ours	XLM-R-large (+Pert (ED))	0.912	0.949	0.36	**0.43**	0.24	0.67	0.57	0.58	**0.98**
	XLM-R-large (+Pert (TD))	0.916	0.949	0.62	0.17	**0.31**	0.70	0.54	0.60	**0.98**
	XLM-R-large (+Pert (ChatGPT))	**0.930**	**0.953**	0.55	0.35	0.26	**0.74**	**0.72**	**0.67**	**0.98**

- ChatGPT results: correctly predicted labels are limited to 'SAF', 'NUM', 'NOT' error types
  → ChatGPT truly struggles immensely in distinguishing the 'POL' type

Experiments - Results

(2) CETC task

Method	Model	Eval ACC	Test ACC	F1-score per label						
				TOX	SAF	NAM	NUM	SEN	POL	NOT
Baseline	mBERT (Devlin et al., 2019)	0.872	0.915	0.00	0.00	0.00	0.29	0.18	0.00	0.96
	mBART25 (Liu et al., 2020)	0.908	0.934	0.33	0.25	0.17	0.35	0.32	0.56	0.97
	mBART50 (Tang et al., 2021)	0.882	0.914	0.40	0.12	0.00	0.22	0.11	0.12	0.96
	XLM-R-base (Conneau et al., 2020)	0.886	0.926	0.31	0.00	0.24	0.24	0.38	0.50	0.96
	XLM-R-large (Conneau et al., 2020)	0.918	0.945	**0.67**	0.15	**0.31**	0.40	0.58	0.61	0.97
	ChatGPT-Demo	0.886	0.910	**0.05**	0.17	0.00	0.15	0.00	0.00	0.96
	ChatGPT-Plain	0.902	0.924	0.00	**0.31**	0.00	0.15	0.12	0.00	0.96
	ChatGPT-Description	0.878	0.928	0.00	**0.31**	0.00	**0.40**	**0.13**	**0.12**	**0.97**
Ours	XLM-R-large (+Pert (ED))	0.912	0.949	0.36	**0.43**	0.24	0.67	0.57	0.58	**0.98**
	XLM-R-large (+Pert (TD))	0.916	0.949	0.62	0.17	**0.31**	0.70	0.54	0.60	**0.98**
	XLM-R-large (+Pert (ChatGPT))	**0.930**	**0.953**	0.55	0.35	0.26	**0.74**	**0.72**	**0.67**	**0.98**

- Our results: Our model effectively captures each error case
- Yet, 'SAF', 'NAM' results show low performance → precise scope for those error types are vague for the model; the task is inherently challenging

Experiments – Analysis

- Comparison for the experiments using Binary and multiclass

- We convert the predictions from the CETC task into a binary class.
  - Regardless of classified error types, we change all predictions that are not predicted as 'NOT' to 'ERR'
  - Analysis 1: We then compare MCC performance with the prediction from the CED
  - Analysis 2 : We divide the test set by critical error type and measure the accuracy for each type

  - XLM-R-large, XLM-R-large(+Pert(ChatGPT))

Experiments – Analysis

Method	Used Label	MCC	F1-Bad	F1-Good	F1-Multi
XLM-R-large	Binary Class	0.5497	0.9719	0.5470	0.5316
	Multi Class	**0.5835**	**0.9740**	**0.5664**	**0.5517**
XLM-R-large (+Pert (ChatGPT))	Binary Class	0.6019	0.9752	0.5607	0.5468
	Multi Class	**0.6540**	**0.9777**	**0.6379**	**0.6237**

[Analysis 1]

- Multiclass results show outperforming performance than binary results

Experiments – Analysis

	XLM-R-large			XLM-R-large (+Pert (ChatGPT))		
	Binary Class	Multi Class	Δ	Binary Class	Multi Class	Δ
TOX	42.86	71.43	+28.57	42.86	57.14	+14.29
SAF	50.00	40.00	-10.00	50.00	50.00	0.00
NAM	10.00	20.00	+10.00	20.00	15.00	-5.00
NUM	58.33	25.00	-33.33	33.33	58.33	+25.00
SEN	57.14	57.14	0.00	57.14	71.43	+14.29
POL	53.85	61.54	+7.69	53.85	61.54	+7.69
NOT	99.03	99.46	+0.43	98.38	99.68	+1.30

[Analysis 2]

- Consistent with [Analysis 1] results
- Four ('TOX', 'NAM', 'POL', and 'NOT') out of seven types achieve a performance improvement

Experiments – Analysis

Method	Used Label	MCC	F1-Bad	F1-Good	F1-Multi
XLM-R-large	Binary Class	0.5497	0.9719	0.5470	0.5316
	Multi Class	**0.5835**	**0.9740**	**0.5664**	**0.5517**
XLM-R-large (+Pert (ChatGPT))	Binary Class	0.6019	0.9752	0.5607	0.5468
	Multi Class	**0.6540**	**0.9777**	**0.6379**	**0.6237**

	XLM-R-large			XLM-R-large (+Pert (ChatGPT))		
	Binary Class	Multi Class	Δ	Binary Class	Multi Class	Δ
TOX	42.86	71.43	+28.57	42.86	57.14	+14.29
SAF	50.00	40.00	-10.00	50.00	50.00	0.00
NAM	10.00	20.00	+10.00	20.00	15.00	-5.00
NUM	58.33	25.00	-33.33	33.33	58.33	+25.00
SEN	57.14	57.14	0.00	57.14	71.43	+14.29
POL	53.85	61.54	+7.69	53.85	61.54	+7.69
NOT	99.03	99.46	+0.43	98.38	99.68	+1.30

- We analyze that **informing the model by specifying the label**, rather than assigning a single 'ERR', **can reduce label ambiguity and increase explainability.**

→ Demonstrating its necessity of providing fine-grained critical error types in the KNOTICED

Conclusion

- A new dataset KNOTICED
  - We introduce a culture-aware politeness label
  - We facilitate two tasks: CED and CETC
- A simple and effective data augmentation approach via perturber
- Our approach demonstrates the effectiveness of our approach

- Given that each language has its own cultural elements, creating datasets for other language pairs would be a promising avenue for future work.

Thank you!

Q&A