AutoAugment Is What You Need: Enhancing Rule-based Augmentation Methods in Low-resource Regimes

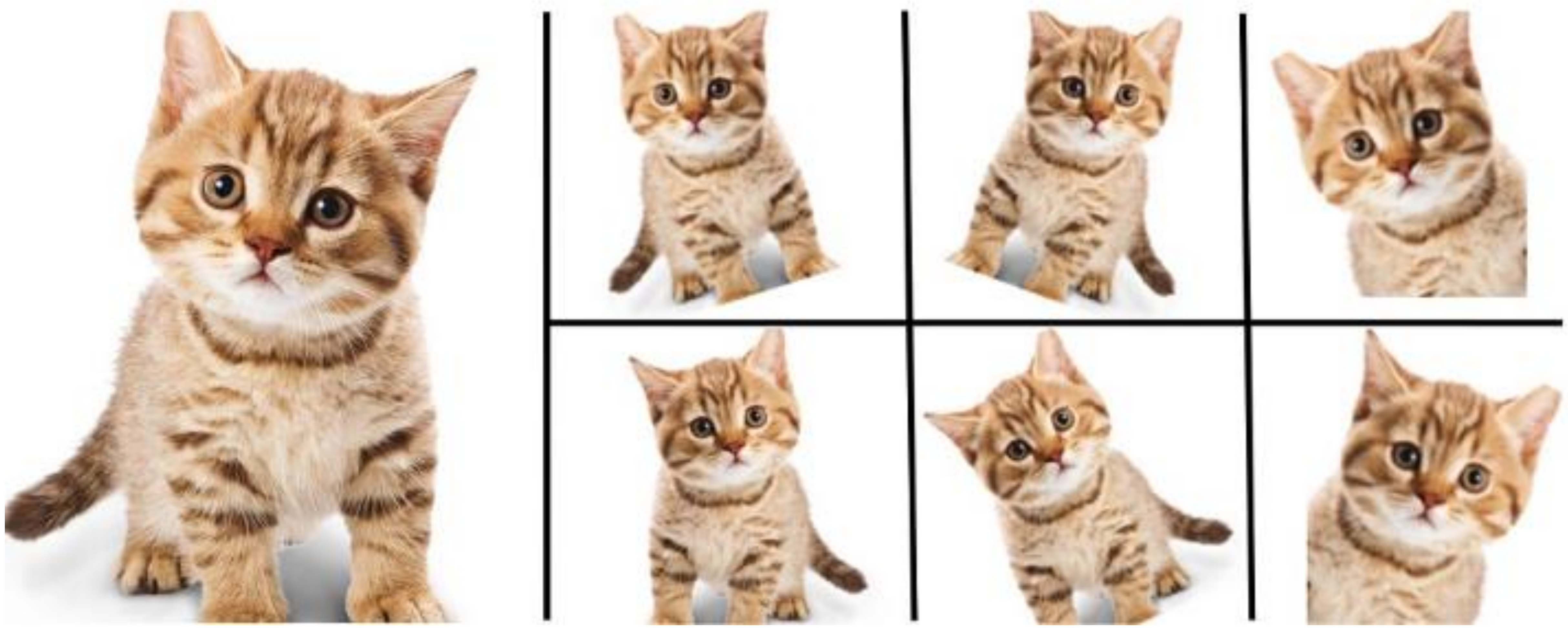
Juhwan Choi, Kyohoon Jin, Junho Lee, Sangmin Song and Youngbin Kim

Chung-Ang University

Data Augmentation

In the field of deep learning, **data augmentation** is widely used for regularization

- **Data augmentation** aims to transform given data to enlarge the training dataset
- For instance, we can augment an image by rotating or flipping it
- Data augmentation enhances the performance and generalizability of the model



Rule-based Text Data Augmentation

Text data augmentation is achieved through various strategies:

- Rule-based methods: introduce modifications by pre-defined rules (e.g., EDA, AEDA…)
  - EDA[1]: relies on random word-level changes
  - Our work focuses on enhancing rule-based augmentation methods
- Model-based methods: utilize other deep learning models to augment given text (e.g., Back-Translation, GPT3Mix…)
- Mixup-based methods: adopt mixup techniques into the text domain

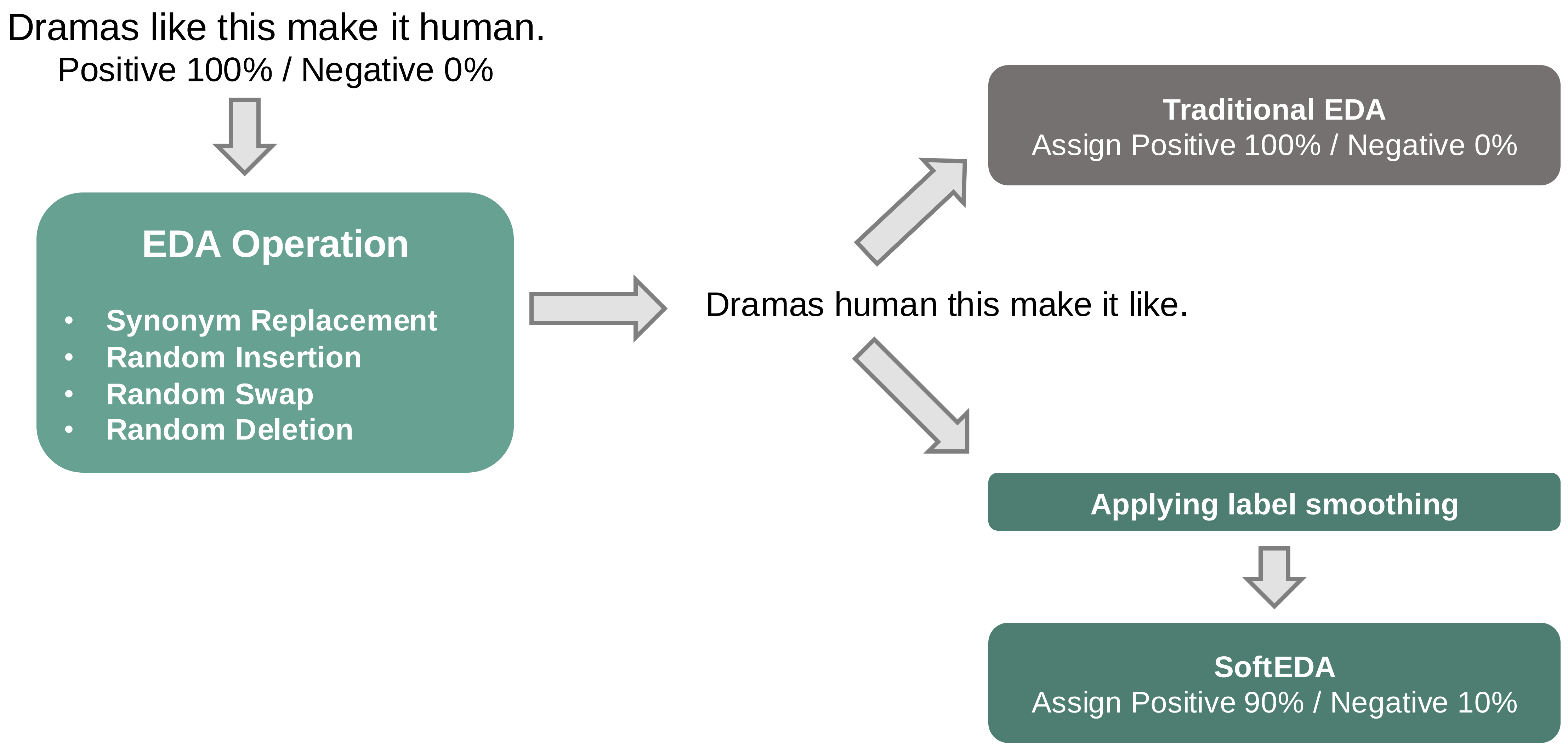
Operation	**Sentence**
None	A sad, superior human comedy played out on the back roads of life.
SR	A ***lamentable***, superior human comedy played out on the ***backward*** road of life.
RI	A sad, superior human comedy played out on ***funniness*** the back roads of life.
RS	A sad, superior human comedy played out on ***roads*** back ***the*** of life.
RD	A sad, superior human out on the roads of life.

1. Wei and Zou, [EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks](), EMNLP 2019

Preliminary: SoftEDA

Despite of its simplicity, EDA methods often lose the original semantics

- Recently, SoftEDA[1] has been proposed to mitigate this problem
- **SoftEDA introduces label smoothing[2] to augmented data**
- The usage of label smoothing compensates for the semantic damage and uncertainty of augmented data



1. Choi et al., *SoftEDA: Rethinking Rule-Based Data Augmentation with Soft Labels*, ICLR 2023 Tiny Papers.
2. Szegedy et al., *Rethinking the Inception Architecture for Computer Vision*, CVPR 2016

Our Motivation

However, finding an optimal label smoothing value for every dataset is difficult

- In previous SoftEDA work, they manually assigned a factor for label smoothing and **conducted a grid search**
- This diminishes the real-world applicability of SoftEDA
- To this end, we adapted AutoAugment[1] to find optimal values for SoftEDA
- Additionally, we aim to enhance the performance of cutting-edge PLMs with the proposed method, not just BERT

1. Cubuk et al., *AutoAugment: Learning Augmentation Strategies from Data*, CVPR 2019.

Proposed Method

Following previous work[1], **we designed an augmentation policy** including:

- p_{aug} and N_{aug}: Probabilities of augmentation and the amount of augmentation
- $p_{SR}, p_{RI}, p_{RS}, p_{RD}$: Probabilities of each sub-operation
- $\alpha_{SR}, \alpha_{RI}, \alpha_{RS}, \alpha_{RD}$: Strengths of each sub-operation
- $\epsilon_{ori}, \epsilon_{aug}$: Factor of label smoothing for original and augmented data, respectively

We optimize this policy based on SMBO[2] as a hyperparameter search

1. Ren et al., *Text AutoAugment: Learning Compositional Augmentation Policy for Text Classification*, EMNLP 2021
2. Bergstra et al., *Algorithms for Hyper-Parameter Optimization*, NeurIPS 2011

Experimental Design

We conducted our experiment with eight different datasets in a **low-resource scenario**

- We used only 100 and 500 randomly selected data for each dataset
- We employed BERT and DeBERTaV3 models
- We assessed our method with EDA, AEDA, and softEDA
- We repeated each experiment five times with different random seeds

Experimental Result

Our method showcased best performance improvement across baselines

- Baselines had performance degradation in several cases
- Furthermore, our method shows a low standard deviation, suggesting its stability

	SST2	SST5	CoLA	SUBJ	TREC	MR	CR	PC
BERT w/o Aug	80.46$_{1.84}$	35.13$_{0.74}$	71.49$_{1.40}$	92.85$_{0.44}$	78.42$_{1.30}$	72.11$_{1.39}$	79.88$_{0.82}$	88.12$_{0.58}$
	86.08$_{1.03}$	43.64$_{0.50}$	75.50$_{0.58}$	95.07$_{\mathbf{0.22}}$	93.27$_{0.42}$	81.29$_{0.52}$	87.53$_{0.60}$	91.15$_{0.21}$
w/ EDA	80.76$_{1.39}$	36.63$_{1.33}$	70.70$_{0.98}$	93.39$_{\mathbf{0.25}}$	81.56$_{1.71}$	73.18$_{1.36}$	79.54$_{1.15}$	89.64$_{0.80}$
	86.71$_{0.63}$	45.08$_{1.16}$	73.18$_{0.52}$	94.69$_{0.33}$	93.99$_{1.05}$	80.41$_{0.29}$	87.71$_{0.57}$	90.81$_{0.40}$
w/ AEDA	80.96$_{1.63}$	36.54$_{0.97}$	72.24$_{1.85}$	93.29$_{0.23}$	81.27$_{2.19}$	74.37$_{2.84}$	80.67$_{1.64}$	88.75$_{0.90}$
	86.66$_{0.63}$	44.53$_{1.02}$	74.44$_{0.41}$	94.60$_{0.48}$	93.87$_{0.75}$	81.57$_{\mathbf{0.15}}$	87.66$_{0.55}$	91.03$_{0.31}$
w/ softEDA	80.80$_{3.22}$	37.13$_{1.60}$	72.41$_{0.95}$	93.24$_{0.40}$	82.92$_{1.70}$	74.40$_{1.27}$	78.95$_{2.65}$	88.82$_{1.63}$
	87.84$_{0.65}$	45.04$_{1.28}$	74.16$_{0.99}$	94.85$_{0.39}$	94.68$_{0.51}$	81.16$_{0.88}$	87.94$_{0.85}$	91.12$_{0.63}$
w/ Ours	**85.48**$_{0.57}$	**39.88**$_{0.41}$	**74.63**$_{\mathbf{0.33}}$	**94.10**$_{0.35}$	**85.88**$_{\mathbf{1.06}}$	**79.32**$_{\mathbf{0.37}}$	**86.49**$_{\mathbf{0.22}}$	**91.54**$_{\mathbf{0.11}}$
	88.53$_{\mathbf{0.27}}$	**46.16**$_{0.63}$	**76.66**$_{0.81}$	**95.54**$_{0.33}$	**95.17**$_{0.54}$	**83.10**$_{0.34}$	**89.98**$_{\mathbf{0.25}}$	**92.16**$_{0.19}$
w/ Ours w/o LS	84.71$_{\mathbf{0.44}}$	39.22$_{\mathbf{0.38}}$	73.80$_{0.79}$	93.71$_{0.35}$	84.85$_{1.40}$	77.86$_{0.53}$	85.70$_{0.88}$	91.13$_{0.19}$
	88.13$_{0.48}$	45.45$_{\mathbf{0.39}}$	76.30$_{\mathbf{0.34}}$	95.15$_{\mathbf{0.22}}$	94.70$_{\mathbf{0.46}}$	82.19$_{0.60}$	89.66$_{0.35}$	91.98$_{\mathbf{0.18}}$
DeBERTaV3 w/o Aug	88.36$_{0.36}$	35.95$_{1.69}$	72.62$_{4.24}$	92.23$_{0.24}$	80.19$_{3.23}$	82.84$_{0.39}$	85.61$_{1.20}$	91.22$_{0.43}$
	92.59$_{0.73}$	48.77$_{1.52}$	82.21$_{0.82}$	94.66$_{0.22}$	94.06$_{0.43}$	86.22$_{0.37}$	91.40$_{0.36}$	91.85$_{0.26}$
w/ EDA	86.61$_{0.70}$	37.64$_{1.23}$	74.83$_{1.10}$	92.85$_{0.48}$	83.65$_{1.84}$	83.18$_{\mathbf{0.32}}$	84.86$_{0.73}$	90.51$_{0.47}$
	93.25$_{0.55}$	49.04$_{0.78}$	79.24$_{0.66}$	94.81$_{0.53}$	94.33$_{0.99}$	86.71$_{0.65}$	91.24$_{0.39}$	92.3$_{0.15}$
w/ AEDA	88.44$_{0.80}$	36.87$_{2.88}$	79.29$_{0.65}$	92.81$_{0.47}$	84.17$_{0.79}$	82.87$_{0.75}$	85.76$_{1.37}$	90.61$_{0.49}$
	92.54$_{0.78}$	49.16$_{0.83}$	82.78$_{\mathbf{0.40}}$	94.92$_{0.58}$	94.45$_{0.80}$	85.77$_{1.63}$	91.09$_{0.49}$	92.29$_{\mathbf{0.11}}$
w/ softEDA	88.94$_{1.03}$	38.37$_{1.65}$	79.40$_{1.51}$	92.90$_{1.08}$	84.58$_{1.29}$	83.50$_{0.65}$	86.33$_{1.65}$	91.28$_{0.82}$
	93.12$_{1.05}$	50.34$_{1.44}$	78.97$_{1.16}$	94.77$_{0.21}$	94.71$_{0.69}$	87.02$_{0.50}$	91.81$_{0.76}$	92.16$_{0.20}$
w/ Ours	**91.38**$_{0.32}$	**42.92**$_{0.52}$	**82.56**$_{0.51}$	**94.47**$_{0.26}$	**87.70**$_{0.90}$	**85.31**$_{0.79}$	**89.95**$_{\mathbf{0.51}}$	**92.32**$_{\mathbf{0.19}}$
	93.94$_{\mathbf{0.30}}$	**52.77**$_{\mathbf{0.62}}$	**84.32**$_{0.49}$	**95.29**$_{0.31}$	**94.92**$_{0.62}$	**87.96**$_{\mathbf{0.17}}$	**92.46**$_{\mathbf{0.18}}$	**92.72**$_{0.40}$
w/ Ours w/o LS	90.47$_{\mathbf{0.26}}$	42.44$_{\mathbf{0.49}}$	82.10$_{\mathbf{0.43}}$	94.22$_{\mathbf{0.15}}$	86.57$_{\mathbf{0.61}}$	85.07$_{0.58}$	89.47$_{0.67}$	92.22$_{0.21}$
	93.40$_{0.58}$	52.54$_{0.66}$	83.67$_{0.86}$	95.15$_{\mathbf{0.12}}$	94.92$_{\mathbf{0.18}}$	87.41$_{0.37}$	92.28$_{0.27}$	92.49$_{0.33}$

Conclusion

We proposed:

- A method to automatically optimize SoftEDA and improve the performance of the model in a low-resource scenario

We found that:

- The proposed method is **effective and stable** for boosting performance
- Rule-based augmentation methods are applicable for cutting-edge PLMs

We plan to:

- Expand the proposed method to other tasks such as NLI

Thank You!