

# ScholarWorks@GSU

## Trustworthy Artificial Intelligence in the NLP Domain: Designing Efficient Adversarial Attacks and Generating Robust Text Representations

Authors	Rafiei Asl, Javad
Citation	Rafiei Asl, Javad (2024). Trustworthy Artificial Intelligence in the NLP Domain: Designing Efficient Adversarial Attacks and Generating Robust Text Representations. Dissertation, Georgia State University. <a href="https://doi.org/10.57709/37360743">https://doi.org/10.57709/37360743</a>
DOI	<a href="https://doi.org/10.57709/37360743">https://doi.org/10.57709/37360743</a>
Download date	2026-05-12 00:02:37
Link to Item	<a href="https://hdl.handle.net/20.500.14694/4000">https://hdl.handle.net/20.500.14694/4000</a>

Trustworthy Artificial Intelligence in the NLP Domain: Designing Efficient Adversarial  
Attacks and Generating Robust Text Representations

by

Javad Rafiei Asl

Under the Direction of Zhipeng Cai, Ph.D. and Daniel Takabi, Ph.D.

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2024

## ABSTRACT

State-of-the-art machine learning techniques, particularly pre-trained language models, have demonstrated exceptional performance across a range of computer science domains. These models, often fine-tuned for specific tasks, have shown exceptional proficiency in natural language understanding, text generation, and various language-related tasks like sentiment analysis, machine translation, and question-answering. Their success can be attributed to extensive training on vast amounts of text data, enabling them to capture intricate linguistic patterns and contextual information. Furthermore, the accessibility of these pre-trained models has democratized research and development, fostering innovation and advancements in fields such as information retrieval, chatbots, and automated content generation.

However, it is crucial to recognize the vulnerability of these state-of-the-art language models to adversarial attacks and tasks. Recent studies have revealed their susceptibility to subtle input manipulations aimed at misleading them. Adversarial attacks on language models involve injecting carefully crafted perturbations into input data, leading the model to produce incorrect or undesirable outputs. This vulnerability raises concerns about the reliability and security of these models, particularly in security-critical domains such as automated content moderation and fraud detection. Addressing and mitigating these vulnerabilities are paramount to ensuring the responsible and secure deployment of pre-trained language models in real-world applications.

To tackle these security challenges, this dissertation focuses on developing efficient adversarial attacks and robust pre-trained language models against such attacks, highlighting the importance of addressing this critical issue in natural language processing. In our first work, we introduce an efficient adversarial attack model for generating context-aware adversarial examples to enhance the adversarial training of pre-trained language models. Our subsequent work centers on the design of a robust sentence embedding framework that improves generalization and robustness across various text representation tasks, even when facing different adversarial attacks and tasks.

As a more comprehensive endeavor, we ultimately propose and implement straightforward yet effective sentence embeddings utilizing a token-level perturbation generator with a novel adversarial token-detection objective. This approach aims to generate high-quality and more resilient sentence embeddings, thereby enhancing the overall robustness of language models in both adversarial and non-adversarial settings.

**INDEX WORDS:** Natural Language Processing, Robust Machine Learning, Adversarial Attack, Trustworthy Artificial Intelligence, Robust Sentence Embeddings, and Imperceptible Adversarial Examples

Copyright by  
Javad Rafiei Asl  
2024

Trustworthy Artificial Intelligence in the NLP Domain: Designing Efficient Adversarial Attacks  
and Generating Robust Text Representations

by

Javad Rafiei Asl

Committee Chair:

Zhipeng Cai

Committee:

Daniel Takabi

Anu Bourgeois

Yi Ding

Electronic Version Approved:

Office of Graduate Services

College of Arts and Sciences

Georgia State University

August 2024

## DEDICATION

I dedicate this dissertation to my wonderful parents. Without them, I would never have made it this far. They have always supported me, motivated me, and given me the strength to see this through to the end.

Thank you, my dear parents. I love you!

## ACKNOWLEDGMENTS

I am deeply grateful for the opportunity to pursue my Ph.D. While much of my time at GSU was affected by the challenges of Covid, lockdowns, and remote work, I hold dear the memories created and the connections formed. I extend my sincerest appreciation to my advisors, Dr. Cai, and Dr. Daniel Takabi, whose guidance steered me through the demanding phases of my research. Their counsel has been invaluable throughout this journey. Thank you for the numerous opportunities you have presented to me.

I also extend my gratitude to the other members of my committee, Dr. Anu Bourgeois, and Dr. Yi Ding, whose insights and encouragement have been instrumental.

To my current and former lab mates, I express my thanks for the insightful discussions, camaraderie, and moments of respite you provided. Your presence has enriched this experience immeasurably.

I am indebted to the faculty and staff of the Computer Science Department and the college for their support and assistance throughout my Ph.D. journey.

Lastly, I want to acknowledge the unwavering support of my friends and family, who have been my pillars of strength. To my parents, Ahmad and Rouhnaz, my sisters Sahar and Sana, and my friends Leila and Prajwal, your encouragement has sustained me. I am grateful for the support and camaraderie of friends from around the globe who have stood by me.

## TABLE OF CONTENTS

ACKNOWLEDGMENTS . . . . .	v
LIST OF TABLES . . . . .	x
LIST OF FIGURES . . . . .	xiii
LIST OF ABBREVIATIONS . . . . .	xv
<b>1 INTRODUCTION AND MOTIVATION . . . . .</b>	<b>1</b>
<b>1.1 Background and Motivation . . . . .</b>	<b>1</b>
<b>1.2 Adversarial Attack and Defense . . . . .</b>	<b>6</b>
<i>1.2.1 Foundational Concepts in Adversarial Attack and Defense . . . . .</i>	<i>6</i>
<i>1.2.2 Vulnerabilities of Pre-Trained Language Models . . . . .</i>	<i>10</i>
<i>1.2.3 Taxonomy of Adversarial Attacks in Natural Language Processing . . . . .</i>	<i>10</i>
<i>1.2.4 Adversarial Defense Strategies in Natural Language Processing . . . . .</i>	<i>15</i>
<b>1.3 Research Problem and Our Contribution . . . . .</b>	<b>19</b>
<b>1.4 Organization . . . . .</b>	<b>20</b>
<b>2 BACKGROUND . . . . .</b>	<b>22</b>
<b>2.1 Text Classification Tasks . . . . .</b>	<b>22</b>
<b>2.2 Adversarial Attack Methods in Textual Data . . . . .</b>	<b>24</b>
<b>2.3 Generating Adversarial Perturbations . . . . .</b>	<b>26</b>
<b>2.4 Adversarial Defense Methods in Natural Language Processing . . . . .</b>	<b>28</b>
<b>2.5 Advancements in Text Representation Learning for Natural Language Processing</b>	<b>30</b>
<b>2.6 Enhancing Text Representation through Contrastive Adversarial Learning . .</b>	<b>32</b>
<b>2.7 Discussions and Open Issues . . . . .</b>	<b>33</b>
<i>2.7.1 Perceptibility . . . . .</i>	<i>34</i>

2.7.2	<i>Transferability</i>	34
2.7.3	<i>Exploration of New Architectures</i>	35
2.7.4	<i>Iterative versus One-shot Approaches</i>	35
<b>3</b>	<b>A Semantic, Syntactic, And Context-Aware Natural Language Adversarial Example Generator</b>	<b>37</b>
3.1	<b>Introduction</b>	37
3.1.1	<i>Contribution</i>	40
3.1.2	<i>Adversarial Examples</i>	41
3.2	<b>Proposed SSCAE Computational Model</b>	41
3.2.1	<i>Background</i>	43
3.2.2	<i>Step 1: Selection of an Input Sample and Identification of Word Importance</i>	45
3.2.3	<i>Step 2: Identifying Influential Words and Context-Aware Substitutions</i>	45
3.2.4	<i>Step 3: Enhanced Refinement of Linguistic Requirements through Dynamic Thresholds</i>	47
3.2.5	<i>Step 4: Enhancing Adversarial Example Generation with Local Greedy Search</i>	49
3.2.6	<i>Step 5: Iterative Input Replacement for Enhanced Adversarial Example Generation</i>	51
3.3	<b>Computational Experiments</b>	54
3.3.1	<i>Expanding Experimentation with Datasets and Target Models</i>	55
3.3.2	<i>Comparison of Adversarial Attack Models Against BERT</i>	56
3.3.3	<i>Exploring Adversarial Attack Across Diverse Target Models</i>	61
3.4	<b>Ablation Studies</b>	63
3.4.1	<i>Investigating the Impact of <math>K</math> in Context-Aware Substitutions</i>	64
3.4.2	<i>Significance of Semantic Refinement</i>	65
3.4.3	<i>Exploration of Dynamic Thresholds</i>	67
3.4.4	<i>Exploring Window Sizes</i>	68
3.4.5	<i>Optimizing Local Greedy Search</i>	69
3.4.6	<i>Illustrative Examples of Adversarial Texts</i>	71
3.4.7	<i>Adversarial Training</i>	73

3.4.8	<i>Human Evaluation</i>	73
3.5	Summary and Discussion	76
4	<b>RobustEmbed: Robust Sentence Embeddings Using Self-Supervised Contrastive Pre-Training</b>	<b>79</b>
4.1	Introduction	79
4.1.1	<i>Contribution</i>	81
4.2	The Proposed Adversarial Self-supervised Contrastive Learning	83
4.2.1	<i>Perturbation Generation</i>	84
4.2.2	<i>Robust Contrastive Learning</i>	86
4.3	Evaluation and Experimental Results	87
4.3.1	<i>Semantic Textual Similarity (STS) Tasks</i>	88
4.3.2	<i>Transfer Tasks</i>	90
4.3.3	<i>Adversarial Attacks</i>	91
4.3.4	<i>Adversarial Semantic Textual Similarity (AdvSTS)</i>	95
4.4	Ablation Studies	97
4.4.1	<i>Optimizing Step Sizes in Perturbation Generation</i>	97
4.4.2	<i>Optimizing Step Numbers in Perturbation Generation</i>	99
4.4.3	<i>Norm Constraint</i>	99
4.4.4	<i>Modulation Factor</i>	100
4.4.5	<i>Comparison of Adversarial Training Techniques</i>	101
4.4.6	<i>Contrastive Learning Loss</i>	101
4.5	Evaluation of Sentence Embedding Distribution	102
4.6	Summary and Discussion	103
5	<b>RobustSentEmbed: Robust Sentence Embeddings Using Adversarial Self-Supervised Contrastive Learning</b>	<b>105</b>
5.1	Introduction	105
5.1.1	<i>Contribution</i>	107
5.2	The Proposed Approach	107

5.2.1	<i>RobustSentEmbed Algorithm</i>	109
5.2.2	<i>Adversarial Perturbation Generator</i>	109
5.2.3	<i>Robust Contrastive Learning</i>	113
5.3	<b>Evaluation and Experimental Results</b>	116
5.3.1	<i>Evaluation Against Adversarial Attacks</i>	116
5.3.2	<i>Assessment of Robustness</i>	119
5.3.3	<i>Evaluation of Semantic Textual Similarity (STS)</i>	121
5.3.4	<i>Transfer Learning Evaluation</i>	122
5.4	<b>Evaluation of Sentence Embedding Distribution</b>	125
5.5	<b>Ablation Studies</b>	127
5.5.1	<i>Optimizing Step Sizes for Perturbation Generation</i>	127
5.5.2	<i>Impact of Step Numbers in Perturbation Generation</i>	128
5.5.3	<i>Magnitude Regulation for Imperceptibility</i>	129
5.5.4	<i>Enhancing Contrastive Learning for Improved Performance</i>	130
5.5.5	<i>Adjusting Perturbation Importance with Modulation Factor</i>	132
5.6	<b>Summary and Discussion</b>	132
6	<b>Conclusion and Future Work</b>	134
6.1	<b>Conclusion</b>	134
6.2	<b>Future Work</b>	135
6.2.1	<i>Future work 1: Advancing SSACAE with Advanced Language Models</i>	136
6.2.2	<i>Future work 2: Enhancing Robustness in Generative Pre-Trained Models</i>	136
	<b>Appendices</b>	138
A	<b>Training Details (RobustEmbed)</b>	139
B	<b>Training Details (RobustSentEmbed)</b>	140
C	<b>Adversarial Attack Methods</b>	141
	<b>REFERENCES</b>	142

## LIST OF TABLES

Table 1.1	Main capabilities of different pre-trained language models. . . . .	5
Table 3.1	Qualitative examples of adversarial samples generated by different adversarial attack methods (TextFooler, BAE, SSCAE). We only attack premises in the MNLI task. . . . .	42
Table 3.2	Average results of the proposed SSCAE, TextFooler, BERT-Attack, and BAE adversarial attack models on 1000 randomly selected testing instances from each of seven datasets using the BERT target model. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage; Query #: Query Number; H: Hypothesis; P: Premise.) . . . . .	57
Table 3.3	Average results of the proposed SSCAE, TextFooler, and BERT-Attack adversarial attack models on 1000 randomly selected testing instances using WordLSTM (with YELP dataset), ALBERT-Base (with YELP dataset), ESIM (with MNLI mismatched), and BERT-Large (with MNLI mismatched) as the target models. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage) . . . . .	61
Table 3.4	Average results of the proposed SSCAE and TEXTBUGGER adversarial attack models on 500 randomly selected testing instances from the IMDB and YELP datasets using different NLP-based cloud APIs. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage)	63
Table 3.5	Results of an ablation study on the proposed SSCAE model using YELP (text classification task) and SNLI (text entailment task) experiments with and without semantic refinement (semantic similarity/consistency) in step 3 of the SSCAE model (Figure 3.2) (E#: Experiment Number; Perturb %: Perturbation Percentage; w: With; wo: Without) . . . . .	65
Table 3.6	A comparison study between the utilization of the USE and Sentence-BERT semantic embedding models over experiments corresponding to YELP, SNLI, IMDB, and MNLI-Mismatched datasets in step 3 of the SSCAE model (Figure 3.2). The standard deviation is included within the brackets. (E#: Experiment Number; USE: Universal Sentence Encoder) . . . . .	66

Table 3.7	A comparative study of the Average_threshold, Median_threshold, TopN_threshold, and Top_maxes_distance heuristics to compute the dynamic threshold in step 3 of the SSCAE model (Figure 3.1). The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage) . . . . .	67
Table 3.8	The efficacy of the minor hyperparameter $\lambda$ within the Top_maxes_distance technique. (E#: Experiment Number; Perturb %: Perturbation Percentage) . . . . .	68
Table 3.9	Examples of original and adversarial sentences from experiments corresponding to YELP and MNLI datasets (E#: Experiment Number; Pair #: Pair Number; P: Positive; N: Negative; E: Entailment; NE: Neutral) . . . . .	72
Table 3.10	Average results of adversarial attacks on the normal and adversarial fine-tuning of the BERT-Large model using two different attack methods: TextFooler and the proposed SSCAE. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage) . . . . .	74
Table 3.11	Human Evaluation Task (E#: Experiment Number) . . . . .	75
Table 4.1	Semantic Similarity performance on STS tasks (Spearman’s correlation, “all” setting) for sentence embedding models. We emphasize the top-performing numbers among models that share the same pre-trained encoder. ♡: results from Reimers & Gurevych (2019); ♣: results from Gao et al. (2021); All remaining results have been reproduced and reevaluated by our team. The ★ symbol shows our framework. . . . .	89
Table 4.2	Results of transfer tasks for different sentence embedding models. ♣: results from Reimers & Gurevych (2019); ♡: results from Zhang et al. (2020b); We emphasize the top-performing numbers among models that share the same pre-trained encoder. All remaining results have been reproduced and reevaluated by our team. The ★ symbol shows our framework. . . . .	91
Table 4.3	Attack success rates of various adversarial attacks applied to three sentence embeddings (SimCSE-BERT, USCAL-BERT, and RobustEmbed-BERT) across five text classification and two natural language inference tasks. . . . .	93
Table 4.4	Attack success rates of five adversarial attack techniques applied to three sentence embeddings (SimCSE, USCAL, and RobustEmbed) across two Adversarial STS (AdvSTS) tasks (i.e. AdvSTS-B and AdvSICK-R). . . . .	96
Table 4.5	The influence of the norm constraint on perturbation generation on the average performance of various Semantic Textual Similarity (STS) tasks. . . . .	99
Table 4.6	The impact of the modulation factor on the average performance of different Semantic Textual Similarity (STS) tasks in generating the final perturbation. . . . .	100

Table 4.7	Performance Comparison of Adversarial Training Methods . . . . .	101
Table 4.8	Effect of Regularization Parameter ( $\gamma$ ) on our Framework Performance . . .	102
Table 5.1	Attack success rates (lower is better) of various adversarial attacks applied to three sentence embeddings (SimCSE, USCAL, and RobustSentEmbed) across five text classification and two natural language inference tasks. RobustSentEmbed reduces the attack success rate to less than half across all attacks. . . . .	117
Table 5.2	Attack success rates (lower is better) of five adversarial attack techniques applied to three sentence embeddings (SimCSE, USCAL, and RobustSentEmbed) across two Adversarial Semantic Textual Similarity (AdvSTS) tasks (i.e. AdvSTS-B and AdvSICK-R). RobustSentEmbed reduces the attack success rate to less than half across all attacks. . . . .	120
Table 5.3	Semantic Similarity performance on STS tasks (Spearman’s correlation, “all” setting) for sentence embedding models. We emphasize the top-performing numbers among models that share the same pre-trained encoder. ♡: results from Reimers & Gurevych (2019); ♣: results from Gao et al. (2021); All remaining results have been reproduced and reevaluated by our team. RobustSentEmbed produces the most effective sentence representations that are more general in addition to robust representation (section 5.3.2 and 5.3.1). . . . .	122
Table 5.4	Results of transfer tasks for different sentence embedding models. ♣: results from Reimers & Gurevych (2019); ♡: results from Zhang et al. (2020b); We emphasize the top-performing numbers among models that share the same pre-trained encoder. All remaining results have been reproduced and reevaluated by our team. RobustSentEmbed outperforms all other methods, regardless of the pre-trained language model ( $BERT_{base}$ , $RoBERTa_{base}$ , or $RoBERTa_{large}$ ). . . . .	123
Table 5.5	The impact of the norm constraint on perturbation generation on the average performance of various STS tasks. . . . .	130
Table 5.6	The impact of the modulation factor on the average performance of different Semantic Textual Similarity (STS) tasks in generating the final perturbation. . . . .	132

## LIST OF FIGURES

Figure 1.1	BERT’s Masked Language Modeling task . . . . .	3
Figure 1.2	BERT’s main architecture including Masked Language Modeling and Next Sentence Prediction tasks . . . . .	4
Figure 1.3	Categories of adversarial attacks on textual deep-learning models. . . . .	11
Figure 1.4	General principle of concatenation adversaries. . . . .	13
Figure 1.5	Edit adversarial attack on sentiment analysis DNN. After editing words (red), the prediction changes from 100% of Negative to 89% of Positive . . . . .	14
Figure 1.6	General principle of paraphrase-based adversaries. . . . .	15
Figure 3.1	General architecture of the proposed SSCAE model . . . . .	43
Figure 3.2	An ablation study of $K$ context-aware substitutions set generated by the BERT-MLM model in step 3 of the SSCAE model . . . . .	64
Figure 3.3	An ablation study of the influence of different window sizes on the retrieval of distinct sets of the top $K = 50$ context-aware substitutions. . . . .	69
Figure 3.4	The impact of the number of important words and candidate substitutions on the average attack success percentage of the IMDB dataset. . . . .	70
Figure 4.1	The general architecture of the RobustEmbed framework. In contrastive learning step, a blue arrow indicate gathering positive pairs together, and a red arrow refers to keeping distance among negative pairs . . . . .	82
Figure 4.2	Average number of queries and the resulting accuracy reduction for a set of 1000 attacks on two fine-tuned sentence embeddings. Green points represent attacks on the RobustEmbed framework, while red points represent attacks on the USCAL approach. . . . .	94
Figure 4.3	The impact of step sizes in perturbation generation on the average performance of STS tasks. . . . .	97
Figure 4.4	The effect of the step number (denoted as $N = K$ or $T$ ) in the $T$ -step FGSM and $K$ -step PGD methods on the averaged correlation of the different Semantic Textual Similarity (STS) tasks. . . . .	98

Figure 4.5	$\ell_{\text{align}} - \ell_{\text{uniform}}$ plot of models based on $\text{BERT}_{\text{base}}$ . . . . .	104
Figure 5.1	The general architecture of the RobustSentEmbed framework. . . . .	108
Figure 5.2	Average number of queries and the resulting accuracy reduction for two fine-tuned embeddings. . . . .	118
Figure 5.3	$\ell_{\text{align}} - \ell_{\text{uniform}}$ plot of models based on $\text{BERT}_{\text{base}}$ . Lower uniformity and alignment is better. . . . .	126
Figure 5.4	The impact of step sizes in perturbation generation on the average performance of STS tasks. . . . .	128
Figure 5.5	The impact of the step number (represented by $N = K$ or $T$ ) in the T-step FGSM and K-step PGD methods on the averaged correlation of the STS tasks. . . .	129
Figure 5.6	The impact of weighting coefficients in the total loss function on the average performance of STS tasks. . . . .	131

**LIST OF ABBREVIATIONS**

<b>AE</b>	<i>Adversarial Example</i>
<b>AI</b>	<i>Artificial intelligence</i>
<b>BERT</b>	<i>Bidirectional Encoder Representations from Transformers</i>
<b>BIM</b>	<i>Basic Iterative Method</i>
<b>DNN</b>	<i>Deep Neural Network</i>
<b>FGSM</b>	<i>Fast Gradient Sign Method</i>
<b>GAN</b>	<i>Generative Adversarial Network</i>
<b>GPT</b>	<i>Pre-trained Transformer</i>
<b>IMDb</b>	<i>Internet Movie Database</i>
<b>ML</b>	<i>Machine Learning</i>
<b>MLM</b>	<i>Masked Language Modeling</i>
<b>MR</b>	<i>Movie Reviews</i>
<b>NLI</b>	<i>Natural Language Inference</i>
<b>NLP</b>	<i>Natural Language Processing</i>
<b>NSP</b>	<i>Next Sentence Prediction</i>
<b>PLM</b>	<i>Pre-trained Language Model</i>
<b>PGD</b>	<i>Projected Gradient Descent</i>
<b>POS</b>	<i>Part-Of-Speech</i>
<b>USE</b>	<i>Universal Sentence Encoder</i>

## CHAPTER 1

### INTRODUCTION AND MOTIVATION

Portions of this dissertation rely on either previously published material or work that is presently undergoing review:

- Chapter 3 is based on *A Semantic, Syntactic, And Context-Aware Natural Language Adversarial Example Generator*

**Javad Rafiei Asl**, Mohammad H Rafiei, Manar Alohalay and Daniel Takabi in IEEE Transactions on Dependable and Secure Computing, Pages 1-17, 2024 Asl et al. (2024a)

- Chapter 4 is based on *RobustEmbed: Robust Sentence Embeddings Using Self-Supervised Contrastive Pre-Training*

**Javad Rafiei Asl**, Eduardo Blanco and Daniel Takabi in Empirical Methods in Natural Language Processing: EMNLP Findings, Pages 4587-4603, 2023 Asl et al. (2023a)

- Chapter 5 is based on *RobustSentEmbed: Robust Sentence Embeddings Using Adversarial Self-Supervised Contrastive Learning*

**Javad Rafiei Asl**, Prajwal Panzade, Eduardo Blanco, Daniel Takabi and Zhipeng Cai in North American Chapter of the Association for Computational Linguistics, under review, Asl et al. (2024b)

#### 1.1 Background and Motivation

The advent of state-of-the-art machine learning techniques, particularly emphasizing Pre-trained Language Models (PLMs), has marked a significant turning point across various computer science

domains. These models, meticulously pretrained on extensive text corpora and meticulously fine-tuned for specific tasks, have showcased unparalleled proficiency in natural language understanding and generation. Their remarkable performance spans critical areas such as Natural Language Processing (NLP), sentiment analysis, machine translation, and question-answering, often surpassing the capabilities of human experts. This exceptional success can be attributed to their adeptness in capturing intricate linguistic nuances and contextual information, enabling them to excel across a broad spectrum of language-related tasks.

Moreover, the widespread availability of PLMs has democratized Artificial intelligence (AI) research and application development, democratizing access for both academia and industry professionals. This accessibility has led to an explosion of innovation in NLP applications, facilitating the emergence of transformative technologies such as chatbots, automated content generation, and voice assistants. The continuous refinement and expansion of these models continue to redefine the boundaries of what is achievable across various computer science disciplines, ushering in an era characterized by unprecedented linguistic sophistication and automated language processing capabilities.

Within the realm of NLP, pre-trained language models, epitomized by Transformers, stand as the bedrock of modern language processing technologies. These models have reshaped the landscape by mastering intricate language representations drawn from vast text corpora, meticulously honing their capabilities for specific NLP tasks. Transformers exhibit remarkable versatility, excelling across a wide array of applications, including Natural Language Inference (NLI), paraphrasing, sentiment analysis, and question-answering. Their strength lies in their ability to discern nuanced

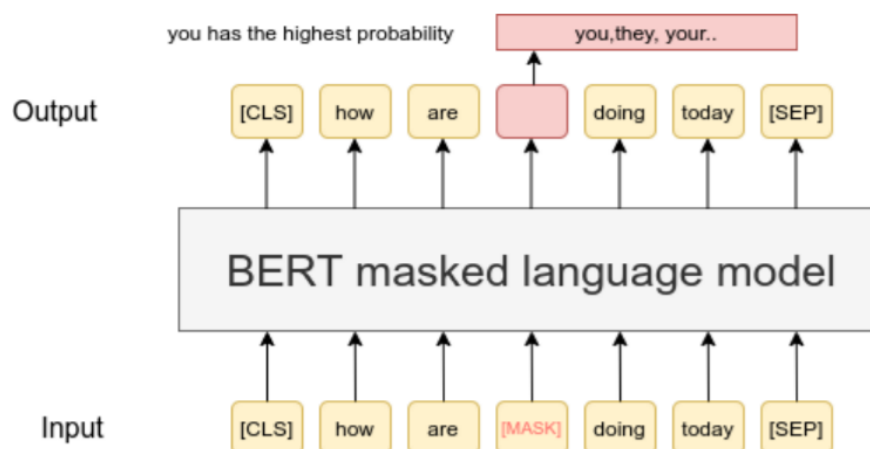


Figure 1.1 BERT's Masked Language Modeling task

linguistic nuances and adapt to diverse tasks by capturing contextual and hierarchical relationships in language. This adaptability enables them to navigate complex linguistic structures with ease, providing developers with a powerful toolkit to achieve state-of-the-art performance in addressing a myriad of NLP challenges.

BERT, standing for Bidirectional Encoder Representations from Transformers, is among the most influential pre-trained language models. BERT undertakes two essential pre-training tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). In MLM, a subset of words in the input text is randomly hidden, and the model is tasked with predicting these concealed words based on the surrounding context, which encourages it to develop comprehensive bidirectional representations. Conversely, in NSP, the model learns to differentiate whether two sentences in a pair follow one another, thus augmenting its grasp of context and sentence relationships. These pre-training procedures, visualized in Figures 1.1 and 1.2, equip BERT with a profound comprehension of language semantics and syntax, rendering it highly proficient in various subsequent NLP tasks.

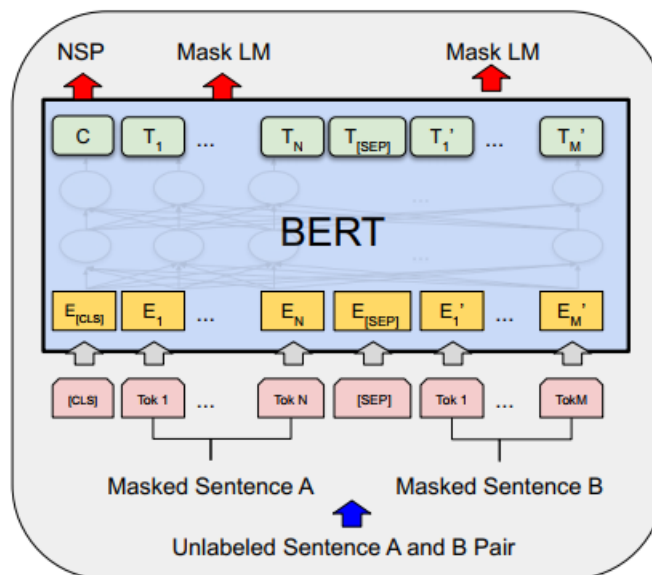


Figure 1.2 BERT's main architecture including Masked Language Modeling and Next Sentence Prediction tasks

For a comprehensive examination of diverse PLMs and their respective capabilities, please refer to Table 1.1.

Nevertheless, the impressive advancements in pre-trained language models are accompanied by significant challenges, notably their susceptibility to adversarial attacks and adversarial NLP tasks. These attacks involve intentionally introducing disturbances into input data, often undetectable to human observers yet sufficient to deceive the model into generating inaccurate or undesirable results. This vulnerability to subtle manipulations raises serious concerns regarding the reliability and security of these models, especially in critical applications like automated content moderation, fraud detection, and automated decision-making systems, where the consequences are high-stakes.

Adversarial NLP tasks go a step further by crafting inputs specifically tailored to exploit weaknesses in the model, thereby highlighting its limitations in comprehending and reasoning about language. These vulnerabilities extend beyond mere security implications, casting doubt

<b>Model</b>	<b>Provider</b>	<b>Year</b>	<b>Description</b>
BERT	Google	2018	Excels in understanding the contextual nuances of language, making it particularly effective for tasks like sentiment analysis and natural language inference.
GPT-2	OpenAI	2019	Known for its exceptional text generation capabilities, producing coherent and contextually relevant language, which is valuable in applications like content generation and creative writing.
RoBERTa	Facebook	2019	Excels in fine-tuning on various NLP tasks with its robust pre-training on a massive text corpus, making it highly effective for natural language understanding, sentiment analysis, and question answering.
ALBERT	Google Research	2019	Achieves strong performance on various NLP tasks, making it well-suited for applications where computational resources are limited, such as mobile devices.
ELECTRA	Stanford University	2020	Known for its efficiency and effectiveness in training, making it particularly suitable for large-scale pre-training and downstream NLP tasks.
FAIRSEQ	ASAPP	2021	Deploy custom models for a variety of natural language processing (NLP) tasks, including translation, summarization, and language modeling.
QDQBert	NVIDIA	2021	Reduce model size and improve inference performance, making it ideal for deploying BERT on mobile and resource-constrained devices
Megatron	Nvidia and Microsoft	2022	Capable of generating realistic and coherent text, translating languages, writing different kinds of creative content, and answering your questions in an informative way
Bard	Google	2023	Capable of generating different creative text formats, translating languages, writing different kinds of creative content, and answering your questions in an informative way

Table 1.1 Main capabilities of different pre-trained language models.

on the trustworthiness and generalization capabilities of pre-trained language models in dynamic, real-world environments. As we delve deeper into the complexities of AI systems, addressing these vulnerabilities becomes imperative to ensure their robustness and effectiveness across various applications.

## **1.2 Adversarial Attack and Defense**

### *1.2.1 Foundational Concepts in Adversarial Attack and Defense*

In the realm of Machine Learning (ML), a pivotal concept is that of Adversarial Examples (AEs). Adversarial examples are carefully crafted perturbations applied to input data, designed to deceive machine learning models, particularly pre-trained language models. These attacks exploit vulnerabilities in the models' decision boundaries, revealing inherent weaknesses in their learning algorithms. Mathematically, an adversarial example can be represented as  $\mathbf{X}' = \mathbf{X} + \epsilon \cdot \mathbf{D}$ , where  $\mathbf{X}$  is the original input data,  $\epsilon$  controls the magnitude of perturbation, and  $\mathbf{D}$  represents the direction in which the perturbation is applied. The main objective of an ideal adversarial attack is to find the optimal perturbation  $\epsilon$  and  $\mathbf{D}$  to maximize model misclassification or produce undesirable outputs. Understanding these foundational concepts is crucial for developing robust machine learning models that are resilient to adversarial attacks.

In the realm of adversarial attacks in machine learning, several approaches have been developed to craft adversarial examples that deceive machine learning models. These approaches can be broadly categorized into three main types Liang et al. (2022): gradient-based attacks, score-based attacks, and optimization-based attacks.

**1. Gradient-Based Attacks:** Gradient-based attacks Goodfellow et al. (2015a) leverage the gradients of the model’s loss function concerning the input data to iteratively adjust the perturbation. An example is the Fast Gradient Sign Method (FGSM), which alters the input data along the sign of the gradient of the loss function, aiming to maximize the loss and induce misclassification. Mathematically, FGSM can be expressed as:

$$\mathbf{X}_{\text{adv}} = \mathbf{X} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{X}} J(\theta, \mathbf{X}, y_{\text{true}}))$$

Here,  $\mathbf{X}_{\text{adv}}$  is the adversarial example,  $\mathbf{X}$  denotes the original input data,  $\epsilon$  controls the perturbation magnitude,  $J(\theta, \mathbf{X}, y_{\text{true}})$  represents the loss function with parameters  $\theta$ ,  $y_{\text{true}}$  is the true label, and  $\nabla_{\mathbf{X}} J(\theta, \mathbf{X}, y_{\text{true}})$  is the gradient of the loss function concerning the input data.

**2. Score-Based Attacks:** Score-based attacks concentrate on optimizing the model’s output scores directly to generate adversarial examples. For instance, the Basic Iterative Method (BIM) Kurakin et al. (2017) iteratively updates the input data to maximize the probability of the target class. Mathematically, BIM can be formulated as:

$$\mathbf{X}_{\text{adv}}^{(t+1)} = \text{clip}_{\mathbf{X}, \epsilon} \left( \mathbf{X}_{\text{adv}}^{(t)} + \alpha \cdot \text{sign}(\nabla_{\mathbf{X}} J(\theta, \mathbf{X}_{\text{adv}}^{(t)}, y_{\text{target}})) \right)$$

Here,  $\mathbf{X}_{\text{adv}}^{(t)}$  represents the adversarial example at iteration  $t$ ,  $\alpha$  denotes the step size,  $y_{\text{target}}$  is the target class label, and  $\text{clip}_{\mathbf{X}, \epsilon}(\cdot)$  ensures that the perturbed data remains within a specified  $\epsilon$  neighborhood of the original data.

**3. Optimization-Based Attacks:** Optimization-based attacks frame the creation of adversarial examples as an optimization problem, aiming to find the perturbation that maximizes the loss within certain constraints. The Carlini-Wagner (CW) attack Carlini & Wagner (2017) is a popular

optimization-based approach that minimizes a loss function to encourage misclassification while controlling the perturbation magnitude. Mathematically, CW attack can be represented as:

$$\underset{\delta}{\text{minimize}} \quad \|\delta\|_p + c \cdot f(\mathbf{X} + \delta)$$

Here,  $\delta$  denotes the perturbation to be optimized,  $\|\cdot\|_p$  represents a norm constraint,  $c$  balances the perturbation magnitude and the loss function, and  $f(\cdot)$  is a loss function that promotes misclassification.

These strategies exemplify the diverse methods employed in adversarial attacks in machine learning, each with its strengths and limitations. Understanding these methods is essential for developing robust machine learning models. ML models, including state-of-the-art neural networks, while achieving remarkable performance in various downstream tasks, are particularly vulnerable to these adversarial attacks. To mitigate this vulnerability, **adversarial training** is employed during the model's training phase to improve the robustness of machine learning models against adversarial attacks. Adversarial training involves augmenting the training dataset with adversarial examples, thus enhancing the model's robustness against such attacks. To defend against adversarial attacks, various adversarial training approaches have been proposed Silva & Najafirad (2020). Two main approaches to adversarial training are:

**1. Standard Adversarial Training:** Standard adversarial training Madry et al. (2018a) involves augmenting the training data with adversarial examples generated using gradient-based methods. The objective is to make the model more robust by exposing it to adversarial perturbations during

training. Mathematically, the objective function for SAT can be formulated as:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(f_{\theta}(\mathbf{x}_i), y_i) + \lambda \cdot \frac{1}{N} \sum_{i=1}^N \mathcal{L}(f_{\theta}(\mathbf{x}_i + \delta_i), y_i)$$

Here,  $\theta$  represents the model parameters,  $\mathbf{x}_i$  and  $y_i$  denote the input and target label of the  $i$ th example,  $f_{\theta}(\cdot)$  is the model's output,  $\delta_i$  is the adversarial perturbation,  $\mathcal{L}(\cdot)$  is the loss function,  $N$  is the number of training examples, and  $\lambda$  is a hyperparameter controlling the trade-off between standard and adversarial losses.

**2. Robust Adversarial Training (RAT):** Robust adversarial training Bai et al. (2021) aims to enhance model robustness by generating adversarial examples that are specifically targeted at the model's vulnerabilities. This approach typically involves solving a min-max optimization problem, where the inner maximization seeks adversarial perturbations, and the outer minimization updates the model parameters to minimize the worst-case loss. Mathematically, the objective function for RAT can be expressed as:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \max_{\|\delta_i\| \leq \epsilon} \mathcal{L}(f_{\theta}(\mathbf{x}_i + \delta_i), y_i)$$

Here,  $\epsilon$  represents the maximum allowable perturbation, and the inner maximization is performed over the  $\epsilon$ -ball around each input sample  $\mathbf{x}_i$ .

These approaches demonstrate two different strategies for training machine learning models to be robust against adversarial attacks. Each method comes with its own trade-offs in terms of computational complexity, effectiveness, and interpretability.

### *1.2.2 Vulnerabilities of Pre-Trained Language Models*

Despite the promising performance exhibited by pre-trained language models across various downstream classification tasks Sun et al. (2019); Gao et al. (2021), showcasing proficiency in generalization, these models are constrained by their susceptibility to adversarial attacks Nie et al. (2020); Wang et al. (2021). This vulnerability stems from their heavy reliance on surface-level statistical patterns, leading to a lack of robustness in adversarial settings. Additionally, they often struggle to grasp semantic and contextual intricacies. Recent studies Garg & Ramakrishnan (2020); Wu et al. (2023); Hauser et al. (2023) have underscored the inadequate robustness of representations derived from PLMs like BERT, which can be easily misled by minor alterations (such as replacing a few words) to the input sentence. However, while these vulnerabilities pose significant challenges, ongoing research Asl et al. (2023b) aims to address these limitations and enhance the robustness of PLMs in diverse contexts.

### *1.2.3 Taxonomy of Adversarial Attacks in Natural Language Processing*

In this section, we review the categorization of adversarial attack approaches in NLP from various perspectives. Figure 1.3 provides a detailed illustration of these viewpoints, and five distinct criteria are employed to categorize the attack methods:

- **Model Access** pertains to the extent of knowledge about the targeted model at the time of the attack.
- **Semantic Application** relates to the methods used in different NLP applications.
- **Target Type** signifies the objective of the attack, whether it aims to induce incorrect predic-

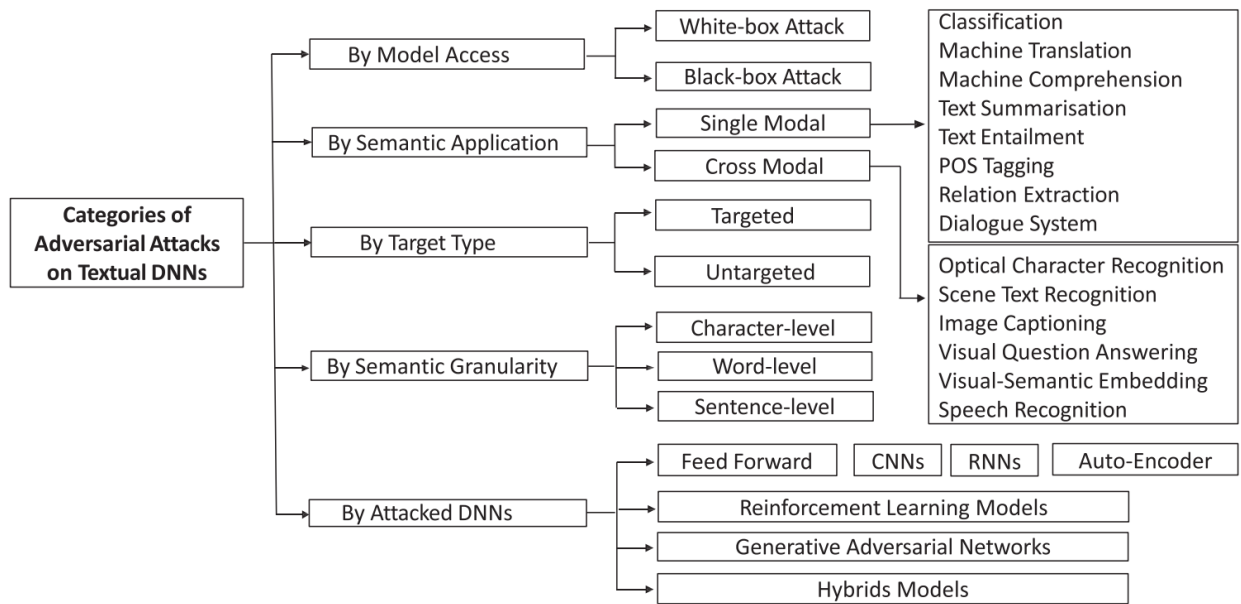


Figure 1.3 Categories of adversarial attacks on textual deep-learning models.

tions or target specific outcomes.

- **Semantic Granularity** considers the level of granularity at which the model is attacked.
- **Attacked DNNs** represent the victim deep neural networks involved in the attacks.

In this proposal, our main focus is on categorizing attacks based on model access, which includes two main groups: white-box and black-box attacks. In a white-box attack, the attacker requires access to the model's complete information, including architecture, parameters, loss functions, activation functions, and input and output data. White-box attacks typically approximate the worst-case attack for a particular model and input, incorporating a set of perturbations. We group white-box attacks on textual Deep Neural Networks (DNNs) into five categories:

1. **FGSM-based Attack:** The FGSM-based attack, originally developed for image classification tasks but adapted for use in NLP, perturbs input data (typically text) to maximize the loss or error

of a neural network model. It computes the gradient of the model's loss function with respect to the input data and perturbs the input data in the direction that increases the loss most rapidly. The amount of perturbation is controlled by a small value called epsilon ( $\epsilon$ ).

2. JSMA-based Attack: The JSMA-based attack, initially designed for image classification tasks but adapted for NLP, perturbs text data to maximize the change in the model's predicted output while keeping perturbations minimal. It computes the Jacobian matrix, representing the sensitivity of model predictions to each input feature (e.g., words in text), and strategically perturbs the most salient input features to craft adversarial examples.

3. C&W-based Attack: The C&W-based attack, developed by Nicholas Carlini and David Wagner, is a state-of-the-art adversarial attack method adapted for NLP from its original use in image classification. C&W formulates an optimization problem to find the smallest perturbation to input text while causing model misclassification. The optimization minimizes both the size of the perturbation and a loss function quantifying model misclassification.

4. Direction-based Attack (HotFlip): HotFlip performs atomic flip operations to generate adversarial examples. Instead of relying on the gradient of loss, HotFlip uses directional derivatives. It represents character-level operations (e.g., swap, insert, delete) as vectors in the input space and estimates loss changes by directional derivatives with respect to these vectors.

5. Attention-based Attack: In an attention-based white-box attack, the attacker manipulates the attention mechanisms within the model. The primary goal is to perturb attention weights to cause the model to make incorrect predictions while maintaining overall text structure and semantics. By targeting attention, the attacker influences which parts of the input the model focuses on during

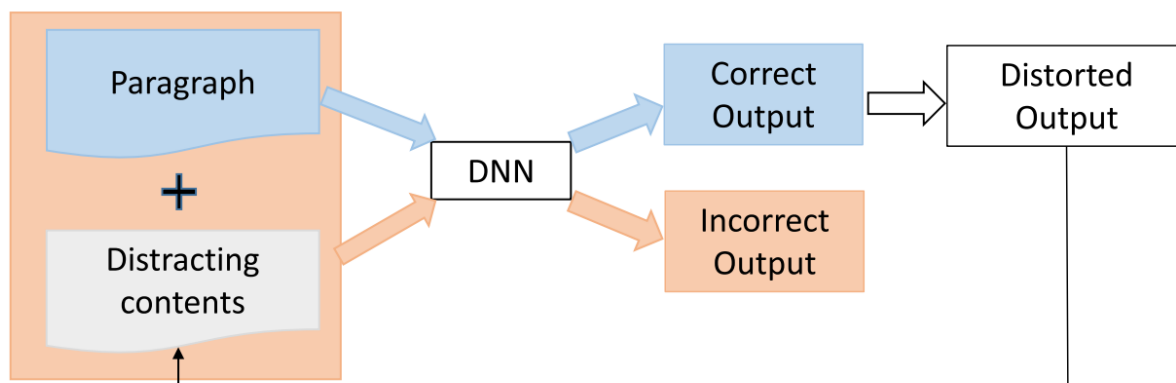


Figure 1.4 General principle of concatenation adversaries.

processing.

The black-box attack does not require access to the details of the neural networks but can access the input and output. These types of attacks often rely on heuristics to generate adversarial examples, which is more practical since, in many real-world applications, the details of the DNN are a black box to the attacker. We categorize black-box attacks on textual DNNs into five categories:

1. Concatenation Adversaries: Concatenation adversaries are adversarial strategies designed to manipulate the concatenation process of multiple text segments, exploiting the way models or systems process such concatenated inputs. As shown in Figure 1.4, these adversaries aim to create adversarial examples by strategically arranging and altering the content of these text segments in a way that leads to unintended outcomes, misclassification, or undesired interpretations when the concatenated text is processed by the target NLP models.

- 2.) Edit Adversaries: Edit adversaries employ adversarial strategies aimed at manipulating text inputs by making subtle and often imperceptible modifications to the content. These adversaries operate under the assumption that only limited knowledge of the target model is available, such

**Task:** Sentiment Analysis. **Classifier:** Amazon AWS. **Original label:** 100% Negative. **Adversarial label:** 89% Positive.

**Text:** I watched this movie recently mainly because I am a Huge fan of Jodie Foster's. I saw this movie was made right between her 2 Oscar award winning performances, so my expectations were fairly high. Unfortunately **UnfOrtunately**, I thought the movie was terrible **terrib1e** and I'm still left wondering how she was ever persuaded to make this movie. The script is really weak **wea k**.

Figure 1.5 Edit adversarial attack on sentiment analysis DNN. After editing words (red), the prediction changes from 100% of Negative to 89% of Positive

as the model's input-output behavior. Edit adversaries employ various text-editing techniques, including synonym substitution, character-level modifications, or grammatical alterations, with the goal of creating adversarial examples.

3.) Paraphrase-based Adversaries: Paraphrase-based adversaries produce a paraphrase of the given sentence with the desired syntax by inputting the sentence and a targeted syntactic form into an encoder-decoder architecture. Specifically, the method first encodes the original sentence, and then inputs the paraphrases generated by back-translation and the targeted syntactic tree into the decoder, whose output is the targeted paraphrase of the original sentence.

4.) GAN-based Adversaries: GAN-based adversaries leverage the principles of Generative Adversarial Networks (GANs) to craft adversarial examples. These adversaries comprise two neural networks: a generator and a discriminator. The generator's objective is to create text sequences that are adversarial, while the discriminator's role is to differentiate between genuine and adversarial text. Through a competitive training process, the generator becomes adept at generating text that resembles real language but subtly misleads the target NLP model, such as a classifier or language model.

5.) Substitution-based Adversaries: Substitution-based adversaries consist of two key compo-

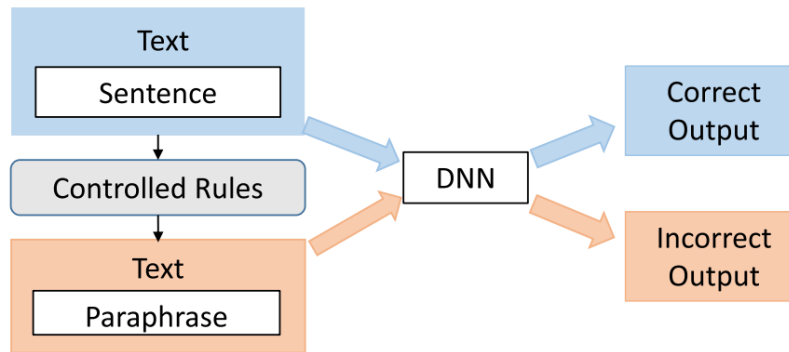


Figure 1.6 General principle of paraphrase-based adversaries.

nents: a generative RNN and a substitute RNN. The generative RNN's role is to produce adversarial API sequences from existing malware API sequences. It accomplishes this by generating a small segment of the API sequence and inserting it into the input sequence. On the other hand, the substitute RNN, which incorporates a bidirectional RNN with an attention mechanism, aims to emulate the behavior of the targeted RNN model.

#### ***1.2.4 Adversarial Defense Strategies in Natural Language Processing***

One essential purpose of generating adversarial examples for neural networks is to utilize them to enhance the model's robustness (Goodfellow et al. 2015b). In textual deep neural networks, two common approaches are employed to achieve this goal: adversarial training and knowledge distillation. Adversarial training involves incorporating adversarial examples into the model training process, while knowledge distillation manipulates the neural network model and trains a new model. In this section, we introduce some representative studies belonging to these two directions.

#### *1.2.4.1 Adversarial Training*

Szegedy et al. (Szegedy et al. 2014) introduced adversarial training, a strategy involving the training of a neural network to correctly classify both normal and adversarial examples. Goodfellow et al. (Goodfellow et al. 2015b) employed explicit training with adversarial examples. In this section, we describe works that utilize data augmentation, model regularization, and robust optimization for defense against textual adversarial attacks.

**1. Data Augmentation:** Data augmentation extends the original training set with the generated adversarial examples, allowing the model to encounter a broader range of data during the training process. Data augmentation is often employed as a defense strategy against black-box attacks, involving additional training epochs on the targeted DNN using adversarial examples. In the context of NLP, data augmentation typically involves tasks such as paraphrasing, synonym substitution, or introducing minor textual alterations to the input data while preserving its original meaning. By exposing the model to these perturbed examples during training, the model learns to be less sensitive to small variations in the input and becomes more resilient to adversarial attacks. Data augmentation not only helps improve the model's generalization but also enhances its ability to handle real-world noisy and diverse textual inputs, making it a valuable component of adversarial training strategies for NLP deep learning models.

**2. Model Regularization:** Model regularization is a crucial aspect of adversarial training in NLP deep learning models, designed to mitigate overfitting and enhance their robustness. In adversarial training, regularization techniques typically involve adding a regularization term to the loss function, encouraging the model to exhibit specific characteristics. Common forms of regularization include

L1 and L2 regularization. Here’s an equation illustrating L2 regularization, where  $J(\theta)$  represents the original loss,  $\theta$  denotes the model parameters, and  $\lambda$  controls the regularization strength:

$$J_{\text{regularized}}(\theta) = J(\theta) + \frac{\lambda}{2} \sum_{i=1}^n \theta_i^2 \quad (1)$$

In this equation, the regularization term  $\frac{\lambda}{2} \sum_{i=1}^n \theta_i^2$  encourages smaller parameter values, preventing the model from fitting the adversarial examples too closely. Regularization is a fundamental component of adversarial training, ensuring that the model maintains its generalization capabilities and robustness in the face of adversarial inputs.

**3. Robust Optimization:** Robust optimization is a fundamental concept in adversarial training for NLP deep learning models, as exemplified by Madry et al.’s approach (Madry et al. 2018b). It involves formulating the learning process as a min-max optimization problem, where the goal is to simultaneously minimize the model’s loss function while maximizing its vulnerability to adversarial attacks. This min-max formulation consists of an inner non-concave maximization problem, representing the adversarial attack, and an outer non-convex minimization problem, representing the model’s defense. Danskin’s theorem plays a pivotal role in this approach, as it establishes that gradients at inner maximizers correspond to descent directions for the min-max problem. This insight allows for the application of back-propagation to optimize the model’s parameters iteratively. Through this robust optimization framework, the approach proposed by Madry et al. successfully demonstrated the enhancement of DNNs’ robustness against adversarial attacks in the context of image classification. The min-max optimization problem can be expressed

as follows:

$$\min_{\theta} \max_{\delta} \mathbb{E}_{(x,y) \sim \text{data}} \left[ \max_{\|\delta\|_{\infty} \leq \epsilon} \mathcal{L}(\theta, x + \delta, y) \right] \quad (2)$$

In this equation,  $\theta$  represents the model parameters,  $\delta$  represents the perturbation added to the input data,  $\epsilon$  is the magnitude constraint on the perturbation,  $(x, y)$  denotes the data examples and their labels, and  $\mathcal{L}(\theta, x + \delta, y)$  is the loss function that combines the model's prediction on the adversarially perturbed input  $x + \delta$  and the true label  $y$ . This formulation captures the essence of robust optimization in adversarial training, seeking to minimize the worst-case loss by finding both the optimal model parameters  $\theta$  and the adversarial perturbation  $\delta$  that maximizes the loss.

#### 1.2.4.2 Knowledge Distillation

Knowledge distillation is a technique used in adversarial training of NLP deep learning models, as proposed by Papernot et al. (Papernot et al. 2016). The idea behind knowledge distillation is to enhance the robustness of a second neural network (DNN) by training it using the knowledge from a more complex original DNN. This process involves utilizing the softmax output, which represents probabilities (e.g., class probabilities in classification DNNs) from the original DNN, to train the second DNN. The second DNN has an identical structure to the original one. In this knowledge distillation process, a temperature parameter denoted as  $T$  is introduced. This parameter controls the level of knowledge distillation. When  $T$  is set to 1, Equation 3 represents the normal softmax function. However, as  $T$  increases, the output probabilities  $q_i$  become closer to a uniform distribution, while lower values of  $T$  lead to more extreme values in the output. The equation for

the modified softmax function is as follows:

$$q_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \quad (3)$$

Where:  $q_i$  is the output probability for class  $i$ ,  $z_i$  is the input to the softmax layer, and  $T$  is the temperature parameter. It's worth noting that a higher temperature ( $T$ ) softens the probability distribution, making it more uniform, while a lower temperature sharpens it, emphasizing the highest probabilities.

### 1.3 Research Problem and Our Contribution

In this dissertation, we address several of the aforementioned challenges by proposing solutions. Our investigation centers on descriptive pre-trained language models, including prominent ones such as BERT and RoBERTa, with the aim of understanding and mitigating their vulnerabilities through the introduction of an effective adversarial attack method. Additionally, we augment the robustness of sentence embeddings derived from pre-trained language models during the pre-training phase, rendering them suitable for deployment in both adversarial and non-adversarial contexts within real-world applications. The contributions made throughout this dissertation are outlined as follows:

- First, we propose a practical and efficient adversarial attack model called SSCAE for generating Semantic, Syntactic, and Context-aware Adversarial Examples (AE). Previous research has failed to produce natural and high-quality AEs. Therefore, there is a need for a comprehensive model that adheres to all linguistic principles for generating well-crafted AEs. To address this issue, SSCAE identifies important words and employs a masked language model to generate an initial set of substitutions. Subsequently, two well-known language

models are utilized to evaluate the initial set in terms of semantic and syntactic characteristics. We introduce (1) a dynamic threshold to capture more efficient perturbations and (2) a local greedy search to generate high-quality AEs.

- We introduce RobustEmbed, a novel self-supervised framework for sentence embeddings that generates robust representations capable of withstanding various adversarial attacks. Existing sentence embeddings are susceptible to such attacks, highlighting a vulnerability in their security. RobustEmbed fills this gap by generating high-risk adversarial perturbations in the embedding space and leveraging the perturbation within a novel contrastive objective approach.
- We develop a comprehensive text representation model aimed at establishing robustness at both token and sentence levels. Through the generation of high-risk adversarial perturbations designed to enhance invariance within the embedding space and their utilization in a novel contrastive objective and effective difference prediction objective, the model adeptly learns high-quality and robust sentence embeddings. Our experimental results demonstrate that our approach is more robust against various adversarial attacks and adversarial tasks.

## **1.4 Organization**

This dissertation is organized as follows. In Chapter 2, we discuss the existing research conducted in the field of adversarial attacks and their respective defense techniques in the NLP domain. In Chapter 3, we propose an efficient adversarial attack model for generating Semantic and Context-aware AEs. In Chapter 4, we introduce robust sentence embeddings that achieve outstanding performance

in various NLP tasks and adversarial scenarios. In Chapter 5, we present a comprehensive text representation at both the sentence and token levels, which generates more robust sentence embeddings in both adversarial and non-adversarial scenarios. Finally, we discuss future research directions and provide conclusions in Chapter 6.

## CHAPTER 2

### BACKGROUND

In this chapter, we present diverse text classification tasks employed to evaluate the efficacy of different adversarial attacks and defense mechanisms. Subsequently, we offer a succinct overview of state-of-the-art adversarial attack and defense methods. Furthermore, we explore techniques for generating adversarial perturbations in the embedding space. We then delve into Text Representation Learning methods, investigating the application of contrastive adversarial learning to produce effective and robust text representations. Finally, we conclude with a section dedicated to discussions and the exploration of open issues in the domain of adversarial attack techniques.

#### 2.1 Text Classification Tasks

This section provides comprehensive details on the text classification tasks utilized to evaluate the generalization and robustness capabilities of various adversarial attacks and defense mechanisms. The MR (Movie Reviews) dataset Pang & Lee (2005a) comprises sentence-level instances annotated with sentiment polarity. It consists of 8,530 training samples and 1,066 testing instances with highly polar sentiments. The CR dataset Hu & Liu (2004) is constructed from customer reviews obtained in three phases: product extraction with customer comments, identification of opinion sentences, and sentiment labeling as positive or negative. Meanwhile, the SUBJ dataset Pang & Lee (2004) includes 5,000 subjective and 5,000 objective sentences extracted from movie reviews, categorized based on subjectivity status and polarity. The MPQA dataset Wiebe et al. (2005) consists of annotated documents sourced from various news outlets, classifying opinion states such as beliefs, emotions, sentiments, and speculations.

Furthermore, the SST2 dataset Socher et al. (2013a) is a sentence-level corpus with 8,544 training and 2,210 testing samples collected from movie reviews and classified into negative or positive sentiments. The MRPC dataset Dolan & Brockett (2005) comprises 5,801 pairs of sentences extracted from news articles, annotated by human labelers to indicate semantic equivalence relationships. The YELP Polarity Review (YELP) dataset Zhang et al. (2015) contains document-level instances, with 560,000 training and 38,000 testing samples labeled as negative (1- and 2-star) or positive (4- and 5-star) reviews. The Internet Movie Database (IMDb) Review dataset Maas et al. (2011) includes 25,000 training and 25,000 testing samples with highly polar sentiments, categorized as negative (4 out of 10) or positive (7 out of 10) based on review scores.

Additionally, the Rotten Tomatoes Movie Reviews (MR) dataset Pang & Lee (2005b) is composed of 8,530 training and 1,066 testing highly polar instances, with negative and positive classes assigned based on consensus among various critics. SNLI Bowman et al. (2015) and MNLI Williams et al. (2018) are three-class datasets consisting of 550,152 and 392,702 training samples, respectively, and 10,000 and 19,643 testing samples. SNLI and MNLI utilize human-written sentence pairs in English, generated using different image captions from the Flickr30K dataset Young et al. (2014) and ten sources of text. Each set of three pairs in SNLI (MNLI) is categorized into entailment, contradiction, and neutral categories based on the relationship between premise and hypothesis sentences. While SNLI employs premise sentences from a relatively uniform image caption dataset, MNLI covers a broader spectrum of text styles. The MNLI testing samples are further categorized into "Matched" and "Mismatched" subsets, where MNLI-Matched pairs share similar contexts and resemble the training pairs compared to MNLI-Mismatched pairs.

## 2.2 Adversarial Attack Methods in Textual Data

This section provides a comprehensive overview of various adversarial attack methods targeting textual data, categorized into two main groups: white-box and black-box attacks.

In the black-box setting, a range of approaches has been proposed, spanning from character-level to sentence-level techniques, with word-level methods demonstrating superior performance Jia & Liang (2017); Li et al. (2019); Zhang et al. (2020a). Jia & Liang (2017) introduced concatenative adversaries, which append distracting sentences at the end of paragraphs to attack systems like the Stanford Question Answering reading comprehension model. Jin et al. (2020) developed TextFooler, which identifies crucial words, assembles a pool of potential synonyms, and substitutes each significant word with the most semantically analogous and grammatically appropriate synonym. Li et al. (2020b) proposed BERT-Attack, a two-step method involving the search for vulnerable tokens and the use of BERT MLM to generate semantic-preserving substitutes. Garg & Ramakrishnan (2020) presented the BAE model, utilizing four adversarial attack strategies based on word replacement or insertion operations. Maheshwary et al. (2021) proposed a decision-based attack strategy that discovers word replacements maximizing semantic similarity. Liu et al. (2023) introduced a new score-based attack model that tackles the challenge of identifying critical words in textual attack models. This model creates semantic adversarial instances to trick text classification models. It integrates a self-attention mechanism and confidence probabilities to assess the importance of words. Lee et al. (2022) presented a black-box attack strategy that effectively interrogates the target model through Bayesian optimization. Their method employs an automatic relevance determination (ARD) categorical kernel to dynamically recognize crucial positions in the

input. To improve scalability for longer input sequences, they introduced techniques such as block decomposition and history subsampling within Bayesian optimization. Wu et al. (2023) introduced a task aimed at improving the ranking of a particular document through the introduction of adversarial perturbations in its text, termed the word substitution ranking attack. To achieve this objective, they introduced a novel technique named Pseudo Relevance-based ADversarial ranking Attack (PRADA). PRADA entails training a surrogate model utilizing Pseudo Relevance Feedback to derive gradients, which are instrumental in identifying suitable adversarial perturbations. Fursov et al. (2022) presented a sentence-level black-box attack technique, fine-tuning a pre-trained language model with a differentiable loss function. Liu et al. (2022) introduced a mimicry-based adversarial attack tailored for black-box neural passage ranking models. By employing a ranking mimicry model, they strategically alter the ranking outcomes and transfer this manipulation attack to the target ranking model. They introduced an innovative gradient-based attack approach to create adversarial triggers using just a few tokens, resulting in deliberate disorder in the rankings. Lu et al. (2023) presented two effective black-box adversarial attackers aimed at compromising three target systems: MLP, AutoEncoder, and DeepLog. The first attacker, based on attention mechanisms, utilizes attention weights to identify susceptible logkeys and produce adversarial examples. The second attacker, based on gradient calculations, determines gradients based on potential vulnerable logkeys to identify the optimal adversarial sample. Both attackers successfully compromise the target systems, posing a threat to their security.

In contrast, the white-box setting provides access to the target model architecture, parameters, and training dataset. Ebrahimi et al. (2018) introduced HotFlip, a pioneering method that leverages

an atomic flip operation to enact character-level modifications. This approach revolutionized the field by enabling precise and targeted alterations at the most granular level of textual data. By employing this atomic flip operation, HotFlip allows for the seamless manipulation of individual characters within text, facilitating the creation of adversarial examples with high precision and efficacy. Li et al. (2019) proposed the TEXTBUGGER framework, designed to operate in both white-box and black-box scenarios. In the white-box context, TEXTBUGGER identifies crucial words by analyzing the Jacobian matrix of the target model and then selects the optimal perturbation from among five types of generated perturbations. In the black-box context, the framework first identifies significant sentences and subsequently determines important words using a scoring mechanism. Finally, an optimal perturbation selection algorithm is employed to modify the selected words. Song et al. (2021) introduced Natural Universal Trigger Search (NUTS), which utilizes a regularized autoencoder to produce adversarial attack triggers. A gradient-based search is then employed to identify triggers that demonstrate effective attack performance. Guo et al. (2021) proposed Gradient-based Distributional Attack (GBDA), featuring two primary components: firstly, adversarial examples (AEs) are instantiated using the Gumbelsoftmax distribution, and secondly, perceptibility and fluency characteristics are enforced utilizing BERTScore and a causal language model perplexity, respectively.

### **2.3 Generating Adversarial Perturbations**

The process of generating adversarial perturbations involves introducing maliciously crafted alterations to benign data, aiming to deceive Machine Learning (ML) models, including those based

on deep learning architectures Goodfellow et al. (2015b). These perturbations are strategically designed to be imperceptible to human observers while inducing the model to make erroneous predictions Metzen et al. (2017). Adversarial training, which integrates adversarial perturbations during the model training phase, has demonstrated efficacy in enhancing the model’s resilience against adversarial attacks Madry et al. (2018c); Shafahi et al. (2020); Xu et al. (2020); Wang et al. (2019b).

Despite advancements in perturbation generation techniques for machine vision tasks Chakraborty et al. (2021), progress in the field of Natural Language Processing (NLP) has been comparatively slower due to the discrete nature of textual data Jin et al. (2020). In recent years, rather than directly applying adversarial perturbations to raw text, researchers have shifted focus towards perturbing the embedding space Wang et al. (2019a); Dong et al. (2021). However, these methods encounter challenges related to generalization, as they may not be universally applicable across different ML models and NLP tasks.

A more generalized approach to generating high-risk adversarial perturbations involves introducing small perturbations  $\delta$  within a specified norm ball in the embedding space, to maximize the adversarial loss:

$$\arg \max_{\|\delta\| \leq \epsilon} L(f_{\theta}(X + \delta), y), \quad (4)$$

Here,  $f_{\theta}(\cdot)$  represents an ML model parameterized with  $X$  as the sub-word embeddings, and  $y$  denotes the true label. To tackle this optimization problem, several gradient-based algorithms have been proposed. These algorithms aim to iteratively adjust the perturbations  $\delta$  within the specified

constraint, thereby maximizing the adversarial loss and generating effective adversarial examples. Some notable gradient-based algorithms include the Fast Gradient Sign Method (FGSM) Goodfellow et al. (2015b), the Basic Iterative Method (BIM) Kurakin et al. (2018), and the Projected Gradient Descent (PGD) Madry et al. (2018c). These techniques iteratively perturb the input embeddings based on the computed gradients of the loss function concerning the perturbations, effectively generating adversarial examples with increased success rates.

## 2.4 Adversarial Defense Methods in Natural Language Processing

In the realm of Natural Language Processing (NLP), the discrete nature of text poses unique challenges for developing robust adversarial defense methods. Unlike in image-based tasks where perturbations can be applied directly to pixel values, textual perturbations must be carefully crafted to preserve syntactic and semantic coherence. Additionally, the high computational cost associated with vanilla adversarial training, which involves training models directly in the raw text space, further complicates the development of effective defense strategies. However, researchers have devised several innovative approaches to mitigate the vulnerability of NLP models to adversarial attacks. This research explores three prominent approaches: Projected Gradient Descent (PGD), Free Adversarial Training with Likelihood Boosting (FreeLB), and Smoothing with Adaptive Regularization Techniques (SMART).

**Projected Gradient Descent (PGD):** Inspired by its success in image classification, PGD iteratively modifies the input text along the direction of the gradient that increases the model's loss on the specific target label. Mathematically, let  $x$  be the original input and  $l$  be the target label. At

each iteration  $t$ , PGD updates the input as:

$$\mathbf{x}^{(t+1)} = \text{clip}(\mathbf{x}^{(t)} + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}} L(f(\mathbf{x}^{(t)}), l)), \mathbf{x}_{\min}, \mathbf{x}_{\max}) \quad (2.1)$$

where  $\alpha$  is the learning rate,  $L$  is the loss function,  $f$  is the NLP model,  $\nabla_{\mathbf{x}}$  is the gradient with respect to the input, and clip restricts the modified input to lie within a feasible range  $(\mathbf{x}_{\min}, \mathbf{x}_{\max})$ .

This process creates adversarial examples that are minimally different from the original text but maximize the model's confusion.

**Free Adversarial Training with Likelihood Boosting (FreeLB):** Unlike PGD's targeted approach, FreeLB leverages unlabeled data for defense. It first generates adversarial examples using a multi-step PGD technique. Then, during training, it utilizes both labeled and unlabeled data, but assigns an additional "boosting" weight to the loss calculated on adversarial examples:

$$L_{\text{FreeLB}} = L_{\text{labeled}} + w \cdot L_{\text{adversarial}} \quad (2.2)$$

where  $L_{\text{FreeLB}}$  is the overall loss,  $L_{\text{labeled}}$  is the loss on labeled data,  $L_{\text{adversarial}}$  is the loss on adversarial examples, and  $w$  is the boosting weight. This encourages the model to learn features that are robust to adversarial perturbations.

**Smoothing with Adaptive Regularization Techniques (SMART):** Recognizing that adversarial examples often exploit the model's overconfidence in low-probability predictions, SMART introduces two regularization techniques:

**1. Label Smoothing:** This reduces the sharpness of the one-hot encoded label distribution by distributing the probability mass more uniformly across all classes. Mathematically, let  $y_i$  be the original one-hot encoded label for class  $i$  and  $K$  be the total number of classes. The smoothed label

$\hat{y}_i$  is calculated as:

$$\hat{y}_i = (1 - \epsilon) \cdot y_i + \frac{\epsilon}{K} \quad (2.3)$$

where  $\epsilon$  is the smoothing factor (typically a small value like 0.1).

**2. Confidence Penalty:** This discourages the model from assigning high confidence scores to incorrect predictions. The confidence penalty term can be added to the loss function, for example:

$$L_{\text{SMART}} = L_{\text{original}} + \lambda \cdot H(y, \hat{y}) \quad (2.4)$$

where  $L_{\text{SMART}}$  is the overall loss with SMART,  $L_{\text{original}}$  is the original loss,  $\lambda$  is a hyperparameter,  $H(y, \hat{y})$  is the cross-entropy between the true label  $y$  and the predicted label  $\hat{y}$ . These techniques, incorporated during training, make the model more robust to small perturbations in the input.

Choosing the optimal defense method depends on various factors, including the NLP task, available data, and computational resources. While PGD remains a powerful baseline, FreeLB offers data-efficient training, and SMART introduces interpretability through its focus on model confidence. The field of adversarial defense in NLP is constantly evolving, and combining these methods or exploring novel approaches holds promise for building more robust and secure language models.

## 2.5 Advancements in Text Representation Learning for Natural Language Processing

In the domain of natural language processing (NLP), the quest for effective text representations has been a longstanding pursuit. Early endeavors in this area revolved around leveraging the distributional hypothesis, which posits that words appearing in similar contexts tend to have similar

meanings. This principle underpinned seminal works such as those by Mikolov and colleagues, who introduced methods like Word2Vec Mikolov et al. (2013b,a). These approaches aimed to learn word embeddings by predicting words within their contextual surroundings, thereby capturing semantic relationships among words.

As research progressed, attention turned towards developing more comprehensive representations capable of capturing the meaning of entire sentences or documents. A prominent avenue of exploration involved the creation of universal sentence embeddings, which aimed to encode the semantic content of sentences in a fixed-length vector space. Both supervised and unsupervised techniques were explored for this purpose. Notable contributions include Doc2Vec, proposed by Le & Mikolov (2014), which extended the Word2Vec framework to learn document embeddings, and SkipThought, introduced by Zhu et al. (2015), which aimed to generate sentence embeddings by predicting the surrounding sentences in a sequence.

More recently, the development of the Universal Sentence Encoder (USE) by Cer et al. (2018) marked a significant milestone in the field. USE utilized a deep neural network architecture to generate high-quality sentence embeddings capable of capturing both semantic and syntactic information. This model was trained on a diverse range of text data, making it adept at encoding the meaning of sentences across various domains and languages.

Another noteworthy advancement came with the introduction of Sentence-BERT (SBERT) by Reimers & Gurevych (2019). SBERT adapted the popular BERT (Bidirectional Encoder Representations from Transformers) architecture to the task of sentence embedding. By fine-tuning BERT on a siamese and triplet network structure, SBERT was able to learn sentence embeddings that

effectively captured semantic similarity between sentences. This approach demonstrated superior performance on various semantic textual similarity tasks, showcasing the efficacy of leveraging pre-trained transformer models for sentence representation learning.

## 2.6 Enhancing Text Representation through Contrastive Adversarial Learning

The primary goal of contrastive learning is to acquire effective low-dimensional representations by enhancing the similarity between semantically related samples while maximizing the dissimilarity between unrelated ones Hadsell et al. (2006). Self-supervised contrastive learning has emerged as a promising technique for data representation across various domains, including machine vision Chen et al. (2020), natural language processing (NLP) Gao et al. (2021); Neelakantan et al. (2022), and speech recognition Lodagala et al. (2023). In this study, we leverage the concept of contrastive learning proposed by Chen et al. (2020) to generate high-quality representations.

Let  $\{(x_i, x_i^+)\}_{i=1}^N$  denote a set of  $N$  positive pairs, where  $x_i$  and  $x_i^+$  represent semantically correlated samples, and  $(z_i, z_i^+)$  are their corresponding embedding vectors. We define  $z_i$ 's positive set as  $\{x_i^{pos}\} = z_i^+$ , and the negative set  $\{x_i^{neg}\}$  as the set of other positive pairs. The contrastive training objective can be formulated as follows:

$$\mathcal{L}_{con,\theta}(x_i, \{x_i^{pos}\}, \{x_i^{neg}\}) = -\log\left(\frac{\sum_{\{x_i^{pos}\}} \exp(\text{sim}(z_i, \{x_i^{pos}\})/\tau)}{\sum_{\{x_i^{pos}, x_i^{neg}\}} \exp(\text{sim}(z_i, \{x_i^{pos}, x_i^{neg}\})/\tau)}\right), \quad (2.5)$$

where  $\tau$  represents a temperature hyperparameter, and  $\text{sim}(u, v)$  denotes the cosine similarity between two representation vectors.

Recently, self-supervised contrastive learning methods have gained traction for learning robust and effective text representations. For instance, SimCSE proposed by Gao et al. (2021) introduced

minimal augmentation techniques involving the application of distinct dropout masks to predict input sentences. The ConSERT model Yan et al. (2021) utilized various data augmentation strategies, including adversarial attacks, token shuffling, cut-off, and dropout, to provide diverse perspectives for contrastive objectives. Miao et al. (2021) integrated adversarial training into contrastive learning to enhance robustness by incorporating regularization into the learning objective. Rima et al. (2022) proposed a novel method that combines adversarial training and contrastive learning, introducing linear perturbations to input embeddings to minimize the distance between original and perturbed representations. Additionally, Pan et al. (2022) presented a technique to enhance the fine-tuning of Transformer-based encoders, leveraging adversarial examples generated through word embedding perturbations and employing contrastive learning to achieve noise-invariant representations.

These advancements highlight the effectiveness and versatility of self-supervised contrastive learning methods in improving the quality and robustness of text representations across various NLP tasks.

## **2.7 Discussions and Open Issues**

The development of adversarial examples in textual data has a relatively shorter history compared to the generation of adversarial examples in images for Deep Neural Networks (DNNs). This discrepancy arises from the increased complexity of introducing perturbations in discrete data while ensuring the preservation of valid syntax, grammar, and semantics. In this section, we delve into these challenges and propose avenues for future exploration.

### ***2.7.1 Perceptibility***

Perturbations in image pixels are typically imperceptible to human observers yet have the capacity to deceive deep neural networks. Conversely, perturbations in text, such as character flips or word substitutions, are often conspicuous. Invalid words and syntactic errors are readily detectable by humans and grammar check software, rendering the perturbation ineffective against real Natural Language Processing (NLP) systems. Despite these limitations, many research endeavors focus on generating such adversarial examples. This approach may be acceptable if the aim is to bolster the resilience of targeted DNN models. However, from a semantic preservation standpoint, altering a word in a sentence can significantly alter its meaning, easily discernible by humans.

### ***2.7.2 Transferability***

Transferability is a common property observed in adversarial examples, reflecting the generalization capability of attack methods. This phenomenon implies that adversarial examples crafted for one DNN model on a particular dataset can effectively deceive another DNN model (cross-model generalization) or dataset (cross-data generalization). Transferability is more commonly exploited in black-box attacks, where the specific characteristics of the targeted DNN models have minimal influence on the attack strategy. Moreover, it has been observed that untargeted adversarial examples exhibit greater transferability compared to targeted ones. Within the realm of DNNs, transferability can be categorized into three levels: (i) similar architectures with different datasets; (ii) different architectures with similar applications; and (iii) different architectures with diverse datasets.

### ***2.7.3 Exploration of New Architectures***

While many common DNN architectures for textual data have been scrutinized for adversarial vulnerabilities, numerous other architectures remain largely unexplored in this context. For instance, generative neural models like Generative Adversarial Networks (GANs) and Variational Auto-Encoders (VAEs), frequently employed in NLP tasks for text generation, have received limited attention regarding adversarial attacks. The training of deep generative models demands advanced skills, potentially contributing to their relative absence from adversarial attack studies. Future investigations may delve into the generation of adversarial examples tailored for these generative DNN architectures.

### ***2.7.4 Iterative versus One-shot Approaches***

Iterative attacks entail a systematic search and update process for perturbations based on the gradient of the targeted DNN model’s output. Consequently, these attacks often yield high-quality and effective perturbations, which may be subtle and difficult to defend against. However, iterative methods typically require significant computational resources and time to discover optimal perturbations, posing challenges for real-time attacks. To address this limitation, one-shot attacks have been proposed, aiming to streamline the perturbation generation process.

In comparison to recent adversarial attack methodologies Maheshwary et al. (2021); Liu et al. (2023); Li et al. (2020b), our proposal introduces novel enhancements. Firstly, we present a new thresholding technique, termed dynamic thresholding, to refine efficient and context-aware perturbations. Secondly, we introduce a word substitution technique called local greedy search, which

generates high-quality adversarial examples capable of deceiving state-of-the-art target models and exhibiting superior performance across various evaluation metrics compared to existing adversarial attacks. Our proposed model incorporates a comprehensive array of linguistic refinements to enhance the imperceptibility of the resulting adversarial examples from a human perspective. Moreover, in contrast to several adversarial learning approaches in the text representation domain Yan et al. (2021); Qiu et al. (2021); Rima et al. (2022); Pan et al. (2022), our framework stands out by generating more efficient and high-risk iterative perturbations in the embedding space. This framework introduces two main innovations: (a) the utilization of a practical token-level perturbation generator based on an adversarial contrastive learning objective, and (b) the introduction of a novel adversarial token-detection objective to generate more robust sentence embeddings.

## CHAPTER 3

### A Semantic, Syntactic, And Context-Aware Natural Language Adversarial Example Generator

#### 3.1 Introduction

Despite the dominance of machine learning across diverse computer science domains, the susceptibility to **Adversarial Examples (AEs)**, characterized as subtly crafted perturbations, remains a focal point of investigation Goodfellow et al. (2015b); Kurakin et al. (2018); Zhang et al. (2020a). These imperceptible alterations aim to coerce a finely tuned machine learning model, referred to as the target model, into making erroneous decisions aligning with the attacker's objectives. Recent advancements advocate for integrating AEs into the training process, known as adversarial training, to enhance the resilience and consistency of models against adversarial manipulations Shafahi et al. (2020); Xu et al. (2020); Wang et al. (2019b). Such endeavors pave the way for the development of computational frameworks capable of generating meticulously designed, humanly indistinguishable adversarial examples tailored for adversarial training.

While strides have been made in adversarial attack and defense methodologies in machine vision Goodfellow et al. (2015b); Papernot et al. (2017); Chakraborty et al. (2021), progress in natural language processing (NLP) has been comparatively slower. The intricate nature of textual data poses significant challenges in designing effective adversarial attack and defense strategies in NLP Jin et al. (2020). Crafting perturbations in NLP necessitates adherence to three fundamental principles: (1) congruence with human decision-making processes Jin et al. (2020); Li et al. (2020b), (2) preservation of semantic fidelity to the original text Jin et al. (2020); Song et al. (2021), and (3)

adherence to the syntactic and grammatical conventions of the source language Jin et al. (2020); Song et al. (2021).

TextFooler Jin et al. (2020), a widely recognized adversarial attack model in Natural Language Processing (NLP), initially employs word embeddings to explore synonymous replacements for each crucial word. Subsequently, it conducts grammatical and semantic similarity assessments to refine the set of synonyms, aiming to identify suitable substitutions as perturbations. Nonetheless, studies have revealed that TextFooler tends to generate intricate, contextually incongruent, and easily detectable perturbations Garg & Ramakrishnan (2020); Li et al. (2020b). To mitigate these issues, recent advancements have introduced two alternative adversarial attack models: Bidirectional Encoder Representations from Transformers (BERT) Attack (BERT-Attack) Li et al. (2020b) and BERT-based Adversarial Examples (BAE) Garg & Ramakrishnan (2020). These models focus on generating context-aware perturbations by masking and replacing crucial words with substitutions provided by the BERT Masked Language Model (BERT MLM) Devlin et al. (2019a). However, despite their efforts to create contextually sensitive substitutions, BERT-Attack and BAE occasionally sacrifice the semantic and syntactic coherence of the original text Li et al. (2020b), thereby compromising their robustness and efficacy. Hence, there is a compelling need for a holistic model that concurrently satisfies all aforementioned principles for crafting well-calibrated perturbations.

This study presents SSCAE (Semantic, Syntactic, and Context-aware Adversarial Examples), a practical and efficient adversarial attack model tailored for generating natural language adversarial examples that adhere to semantic consistency, syntactic coherence, and contextual relevance.

SSCAE operates as a black-box attack method, producing humanly imperceptible and context-aware adversarial examples while upholding semantic and syntactic integrity. Initially, SSCAE utilizes the BERT MLM to generate an initial set of substitution candidates. Subsequently, it leverages two well-established language models, namely the Universal Sentence Encoder (USE) Cer et al. (2018) and the Generative Pre-trained Transformer 2 (GPT-2) Radford et al. (2019), to assess the semantic and syntactic characteristics of the substitutions, respectively. Additionally, we introduce a novel dynamic thresholding technique that adapts the semantic and syntactic thresholds per crucial word, departing from the conventional static thresholding approach Jin et al. (2020); Li et al. (2019); Garg & Ramakrishnan (2020). This dynamic thresholding method accounts for variations in semantic similarity across different substitutions, thereby enhancing the efficacy of perturbation generation. Furthermore, SSCAE incorporates a word permutation-substitution technique named local greedy search to facilitate simultaneous substitution of multiple crucial words, thereby generating high-quality adversarial examples. Local greedy search efficiently identifies candidate substitutions that minimize the target model’s confidence score, leading to the production of high-quality adversarial examples through diverse combinations of substitutions.

We conducted fifteen computational experiments across commonly used text classification and Natural Language Inference (NLI) datasets to evaluate SSCAE’s performance against three state-of-the-art methods: TextFooler, BERT-Attack, and BAE using the BERT target model (seven experiments). Additionally, we assessed SSCAE’s effectiveness on other target models and diverse NLP-based cloud applications (eight experiments). SSCAE consistently outperformed the other methods across all experiments, demonstrating superior semantic consistency, lower query count,

and comparable perturbation rates.

### ***3.1.1 Contribution***

Our primary contributions can be summarized as follows:

- We present SSCAE, an effective adversarial attack approach designed to generate context-aware adversarial examples that are imperceptible to human observers. SSCAE incorporates various contextual linguistic factors, such as semantic, syntactic, and grammatical features, to refine candidate substitutions proposed by BERT-MLM, resulting in high-quality perturbations. Notably, SSCAE introduces two innovative techniques: dynamic thresholding and word permutation-substitution, which outperform existing adversarial attack models by a considerable margin.
- We introduce dynamic thresholding to adaptively meet semantic and syntactic requirements for each important word, departing from traditional constant threshold approaches. By computing individual thresholds for every critical word, SSCAE leverages the non-linear nature of semantic and syntactic correctness distributions, significantly improving after-attack accuracy, average perturbation percentage, and semantic consistency across text classification and natural language inference tasks.
- We propose a word permutation-substitution technique named local greedy search to concurrently substitute multiple important words, yielding high-quality adversarial examples. Experimental results demonstrate the effectiveness of local greedy search over conventional

sequential substitution methods, as evidenced by improvements in after-attack accuracy, average query number, and semantic consistency across different NLP tasks.

- We evaluate SSCAE across various deep learning models on four common text classification and three natural language inference tasks. Our method surpasses state-of-the-art techniques by maintaining higher after-attack accuracy and semantic consistency while requiring fewer queries and achieving comparable perturbation rates.

### ***3.1.2 Adversarial Examples***

Table 3.1 presents qualitative instances of adversarial samples crafted by three distinct adversarial attack techniques: TextFooler, BAE, and SSCAE methods. Unlike the former two adversarial methods, our approach produces adversarial examples of superior quality and naturalness with minimal substitutions, all while maintaining semantic coherence and adhering to the syntactical and grammatical norms of the source language.

## **3.2 Proposed SSCAE Computational Model**

Figure 3.1 illustrates the workflow of the proposed SSCAE model, which comprises five distinct steps outlined in this section. We first provide a brief overview of two prominent NLP models utilized to address linguistic constraints before delving into the details of the proposed model.

Dataset	Adversarial Attack	Text Instance	Label
YELP	Original	Great location! Close to shops and theatre. Nice staff.	Positive
	TextFooler	<b>Poor</b> location! <b>Distance</b> from shops and theatre. <b>Unpleasant</b> staff.	Negative
	BAE	<b>Excellent</b> location! <b>Incredibly near</b> to shops and theatre. Nice staff.	Negative
	SSCAE (ours)	<b>Terrible</b> location! Close to shops and theatre. Nice staff.	Negative
MNLI	Original	Hypothesis: Poirot was disappointed with me → Premise: Still, it would be interesting to know. 109 Poirot looked at me very earnestly, and again shook his head.	Neutral
	TextFooler	Still, it would be <b>enthusiastic</b> to know. 109 Poirot looked at me very <b>cautious</b> , and again <b>vibrate</b> his head.	Entailment
	BAE	Still, it would be <b>captivating</b> to know. 109 Poirot <b>gazed</b> at me <b>extremely</b> earnestly, and again shook his head	Entailment
	SSCAE (ours)	Still, it would be interesting to know. 109 Poirot looked at me very <b>carefully</b> , and again shook his head.	Entailment
IMDB	Original	You’d better choose Paul Verhoeven’s even if you have watched it.	Negative
	TextFooler	You’d <b>brilliantly</b> choose Paul Verhoeven’s even if you have <b>seen</b> it.	Positive
	BAE	You’d <b>certainly</b> choose Paul Verhoeven’s even if you <b>haven’t</b> watched it.	Positive
	SSCAE (ours)	You’d <b>rather</b> choose Paul Verhoeven’s even if you have watched it.	Positive

Table 3.1 Qualitative examples of adversarial samples generated by different adversarial attack methods (TextFooler, BAE, SSCAE). We only attack premises in the MNLI task.

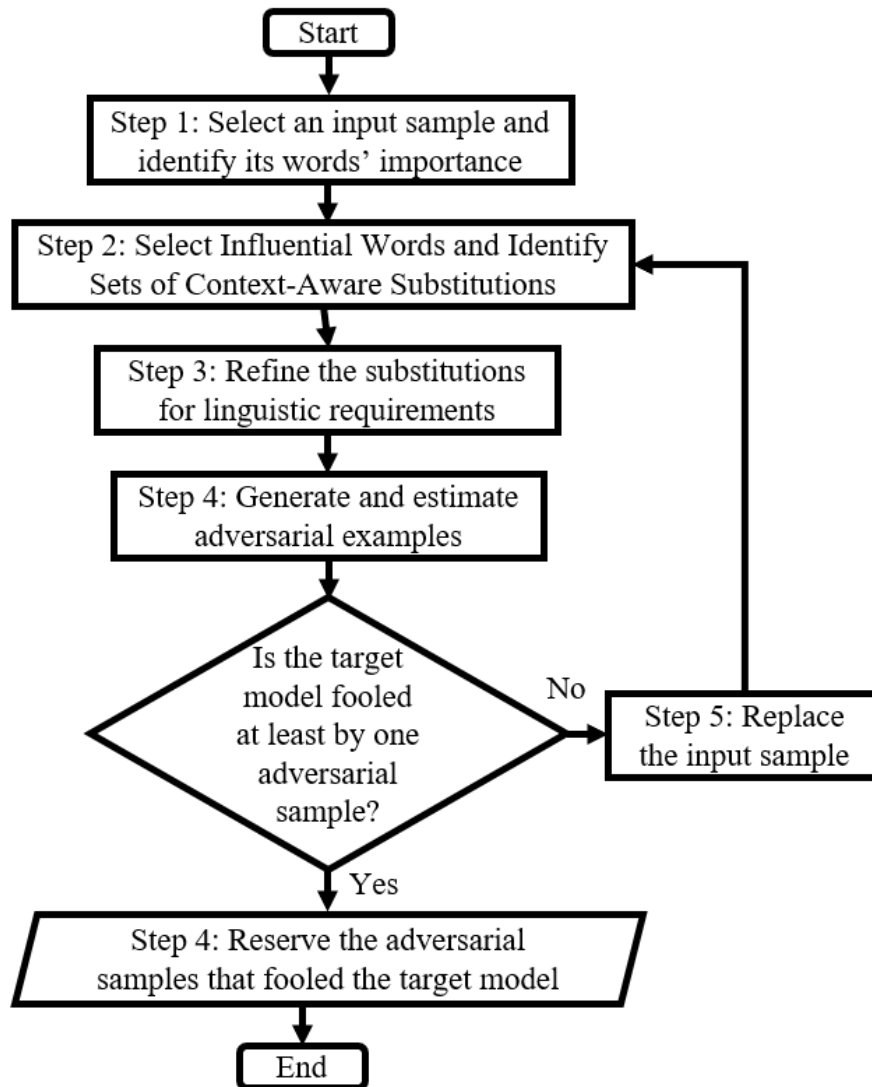


Figure 3.1 General architecture of the proposed SSCAE model

### 3.2.1 Background

#### 3.2.1.1 Universal Sentence Encoder (USE) for Semantic Similarity Measurement

Cer et al. (2018) introduced the Universal Sentence Encoder (USE) for English, a model capable of encoding textual content of varying lengths into embedding vectors of a fixed size (typically 512). These vectors statistically capture the semantic characteristics of the input text, making them

suitable for measuring semantic similarity between texts of different lengths (Jin et al., 2020).

The proposed SSCAE model leverages USE to generate embedding vectors enriched with semantic information from the input sentence ( $V_{input}$ ) and the encoded Answer Explanation ( $V_{AE}$ ). The cosine similarity ( $S_{similarity}$ ) between these vectors is then computed to assess the semantic closeness of each generated AE to its corresponding input sentence:

$$S_{similarity} = \frac{V_{input} \cdot V_{AE}}{\|V_{input}\| \|V_{AE}\|} \quad (3.1)$$

### 3.2.1.2 Generative Pre-trained Transformer 2 (GPT-2) for Syntactic Fluency Assessment

Radford et al. (2019) introduced GPT-2, a transformer-based language model trained to predict the next word in a sequence based on the preceding context. This powerful pre-trained model has been employed in various studies (Chen et al., 2022; Pang et al., 2020) to evaluate the syntactic fluency of text by leveraging its understanding of language structure and rules.

Within the SSCAE model, GPT-2 plays a key role in assessing the syntactic correctness of proposed answer explanation substitutions. It calculates the probability of the original important word,  $P_I$ , and each of its suggested replacements,  $P_S$ . The syntactic correctness score for each substitution is then determined by:

$$S_{syntactic} = P_S - P_I \quad (3.2)$$

### ***3.2.2 Step 1: Selection of an Input Sample and Identification of Word Importance***

A text input sample is chosen randomly from an available dataset and fed into the SSCAE model. This input sample could be a sentence, a paragraph, or even an entire document. To identify the important words within the input sample, we employ a greedy search technique Gao et al. (2018). This method involves iteratively masking individual words in the input sample, replacing them with a predefined symbol. Subsequently, the target model evaluates the modified input sample and generates a confidence score for the true label. The confidence score, ranging from 0 to 1, indicates the probability of the input belonging to a particular label, computed by feeding the input sample (with or without masking) to a trained model (referred to as the target model). In the machine learning literature, this confidence score is also known as logits Jordan & Mitchell (2015); LeCun et al. (2015); Mahesh (2020). The difference between the confidence score before and after masking, denoted as  $\delta$ , is then calculated for each word in the input sample. This  $\delta$  value is determined for all words in the sample. Finally, the words are sorted based on their  $\delta$  values; a higher  $\delta$  value indicates greater influence (importance) of the corresponding word in the input sample on the target model's prediction.

### ***3.2.3 Step 2: Identifying Influential Words and Context-Aware Substitutions***

The SSCAE model employs a two-step process to identify influential words and their corresponding context-aware substitutions for adversarial attacks.

#### **1. Influential Word Selection:**

The model first identifies the  $M$  most influential words within the input text. These words are

determined by their associated  $\delta$  values, with larger values indicating greater impact on the target model’s output. This selection process prioritizes words that can potentially lead to significant changes in the target model’s prediction, maximizing the effectiveness of the attack.

## **2. Context-Aware Substitution Generation:**

For each selected influential word, the model employs a context window to capture its surrounding textual context. This window, with a size of  $2N_W$  centered on the target word, incorporates  $N_W$  neighboring words both before and after. An empirical evaluation determined that a window size of 4 ( $N_W = 2$ ) provides a sufficient balance between capturing relevant context and computational efficiency.

Instead of directly applying the BERT Masked Language Model (MLM) on the influential word, the SSCAE model adopts an alternative approach. It applies the BERT MLM on each neighboring word within the context window. This strategic shift allows the model to retrieve context-aware substitutions that are specifically meaningful for the influential word considering its surrounding context.

The BERT MLM applied to each neighboring word generates a distinct set of the top  $K$  context-aware substitutions, resulting in a total of four sets (due to the four neighboring words). These individual sets capture substitutions from different contextual perspectives surrounding the influential word. Subsequently, the model combines these four sets using a union operation, creating a comprehensive initial set of context-aware substitutions for the chosen influential word.

It is noteworthy that the initial set might exceed the size of  $K$  but remains close to it. This is because some substitutions might be identical across several of the top  $K$  sets generated from

different neighboring words. By incorporating both identical and non-identical substitutions, the model enhances its ability to potentially bypass the target model's defenses, as identical substitutions offer opportunities to exploit known vulnerabilities, while non-identical ones explore alternative attack vectors.

### ***3.2.4 Step 3: Enhanced Refinement of Linguistic Requirements through Dynamic Thresholds***

In response to the intricate distribution patterns of semantic similarity and syntactic correctness observed in context-aware substitutions, we propose the integration of an adaptive and personalized threshold for every significant word, which we term the "dynamic threshold." This innovative approach is tailored to alleviate the constraints imposed by rigid, fixed thresholds. The dynamic threshold dynamically computes an appropriate threshold based on linguistic features, specifically semantic similarity and syntactic correctness, for each significant word's candidate substitutions. The overarching goal is to generate perturbations that surpass existing standards in quality and fluency Devlin et al. (2019a); Liu et al. (2019); Lan et al. (2020).

Despite the established dominance of BERT-based models across a myriad of NLP tasks, a substantial portion of top  $K$ -generated substitutions fail to yield valid adversarial examples (AEs) that meet the benchmarks for language fluency, semantic consistency, and imperceptibility. It is imperative to meticulously examine substitutions to ensure the credibility of the generated perturbations and adherence to linguistic criteria. Each substitution must meticulously preserve the input sample's meaning and grammatical requisites while adhering to the syntactic structures of the source language. The SSCAE model employs sophisticated refinement strategies aimed at identifying substitutions with a heightened probability of preserving semantic, syntactic, and

grammatical linguistic characteristics.

Semantic embedding models such as USE and transformer-based language models like GPT-2 are instrumental in this process, assigning semantic similarity and syntactic correctness scores respectively to each substitution. These scores provide insights into the extent to which a substitution maintains the input sample’s meaning and adheres to the syntactic norms of the source language. Recent studies have predominantly relied on a constant threshold, either through iterative comparison with a previous adversarial example Jin et al. (2020) or by filtering substitutions based on semantic consistency with the original text Li et al. (2019, 2020b). However, the exclusive dependence on a constant threshold proves to be inadequate for effective refinement, particularly when juxtaposed with previously generated adversarial examples. While a fixed threshold may effectively refine a subset of high-quality perturbations for a significant word, it often results in a compromise in semantic similarity with the original text for other crucial words. Consequently, although such an adversarial attack method may yield examples capable of deceiving the target model, the semantic coherence of the final adversarial examples is severely undermined, rendering them susceptible to detection from a human perspective.

This work delves into an exploration of four distinct potential heuristics for computing dynamic thresholds to refine substitutions: (1) Average\_threshold, (2) Median\_threshold, (3) TopN\_threshold, and (4) Top\_maxes\_distance. The Average\_threshold (Median\_threshold) method computes the average (median) of substitution scores. Meanwhile, TopN\_threshold selects the score of the  $N^{th}$  substitution after descending sorting, where  $N$  serves as a minor hyper-parameter. On the other

hand, `Top_maxes_distance` computes the threshold as delineated in Equation 3.3:

$$DT = S_N - \lambda \cdot \Delta S_N, \quad (3.3)$$

where  $S_N$  represents the score corresponding to the  $N^{th}$  substitution after descending sorting,  $\lambda$  denotes a minor hyperparameter, and  $\Delta S_N$  signifies the difference between the highest score among substitutions and  $S_N$ . Collectively, these dynamic thresholds engender substitutions with elevated semantic consistency, thereby substantially augmenting the quality of generated AEs. Through an exhaustive array of trial and error experiments conducted across diverse datasets, it was discerned that a lower bound of 0.7 proved optimal for the dynamic threshold, thereby ensuring the validity of the substitutions.

Furthermore, the Part-Of-Speech (POS) tag of substitutions must align with that of the selected word to uphold grammatical linguistic characteristics. Deviations from this criterion result in the filtering out of substitutions. However, exceptions are made for singular/plural mismatches and verb root similarities, thereby ensuring grammatical correctness during substitution implementation. It is pertinent to note that in this method, the identification of correct substitution POS tags occurs upon implementation of the substitution in the input sample.

### ***3.2.5 Step 4: Enhancing Adversarial Example Generation with Local Greedy Search***

In pursuit of an advanced and refined methodology, we propose the simultaneous substitution of multiple significant words as a more effective alternative to the conventional sequential substitution approach. This innovative technique, termed "local greedy search," represents a pragmatic and intuitive concept that not only yields worst-case perturbations but also significantly expedites the

process of deceiving the target model. Despite its simplicity, this approach ensures the generation of high-quality and imperceptible adversarial examples Kurakin et al. (2018); Goodfellow et al. (2015b).

The local greedy search method serves as a cornerstone in the proposed SSCAE model for generating adversarial samples. Here’s how it works: For each of the  $M$  important words (i.e., the  $M$  top words with the largest  $\delta$ s) specified in step 2, SSCAE selects the top  $N$  refined substitutions that lead to the most significant reduction in confidence scores. Subsequently, to explore potential adversarial samples, the local greedy search is employed to simultaneously substitute all  $M$  important words. The values of  $M$  and  $N$  are carefully chosen to ensure that the number of combinations of different substitutions, i.e.,  $N^M$ , remains within the available computation resources. For instance, in our study, we investigated magnitudes ranging from 1 to 4 for both  $M$  and  $N$ , ensuring that the total number of combinations does not exceed 256 (when  $N = 4$  and  $M = 4$ ). Leveraging the adaptive refinements from the previous step, the adversarial samples generated through this process have a significantly higher likelihood of preserving semantic, syntactic, and grammatical consistency with the input samples.

Subsequently, the adversarial samples are estimated by the target model and are descendingly sorted based on the confidence gap magnitude, as defined by Equation 3.4:

$$G_y^{\text{adv}} = T_y(S^{\text{orig}}) - T_y(S^{\text{adv}}), \quad (3.4)$$

where  $T(\cdot)$  denotes the target model, with  $T_y(\cdot)$  assigning a confidence score corresponding to the truth label  $y$  for a given input sentence. Here,  $S^{\text{orig}}$  and  $S^{\text{adv}}$  represent the input sample and

adversarial sample, respectively. The term  $G_y^{adv}$  signifies the gap between the confidence scores of the input sample and the adversarial sample for the truth label.

Through the comprehensive exploration of all  $N^M$  possible combinations of substitutions, wherein  $M$  important words are replaced with one of the combinations each time, the SSCAE model evaluates a vast array of potential adversarial samples. This exhaustive search enables the model to identify the most effective adversarial example early in the permutation process, thereby minimizing the number of queries required to achieve the adversarial attack objective. Thus, the local greedy search strategy proves instrumental in streamlining the adversarial example generation process and enhancing the efficiency of adversarial attacks.

### ***3.2.6 Step 5: Iterative Input Replacement for Enhanced Adversarial Example Generation***

In scenarios where none of the generated adversarial samples manage to deceive the target model, a systematic approach is employed to enhance the efficacy of the generated adversarial examples. Specifically, the adversarial sample exhibiting the lowest probability of belonging to the truth label, as determined by the highest confidence gap in Equation 3.4, is designated as the new input text sample. Subsequently, steps 2 through 5 are iteratively repeated, thereby ensuring continual refinement and optimization of the adversarial example generation process. It is important to note that in this iterative process, the new input sample is poised to generate adversarial samples with a heightened probability of deceiving the target model compared to the current input sample Goodfellow et al. (2015b); Kurakin et al. (2018).

The pseudocode outlining the operations of the proposed SSCAE model is encapsulated in Algorithm 1. This algorithm encompasses various essential functions aimed at facilitating the

**Algorithm 1: SSCAE Pseudocode**

**Input** : input sentence  $S = [w_1, \dots, w_n]$ , truth label  $Y$ , target model  $T$ , Fixed-size window  $W$ , thresholds' lower bounds ( $ST_{semantic}$ ,  $ST_{syntactic}$ )

**Output** : Adversarial Sentence  $S^{adv}$

**Function** *Compute\_Importance* ( $S^{adv}$ ,  $T$ ,  $Y$ ):

```

foreach word  $w_i \in S^{adv}$  do
  | Calculate  $p_i(Y)$  of  $S_{i-masked}^{adv}$  through  $T$ 
end
return Sorted words  $WL = [w_{min_1(P(Y))}, w_{min_2(P(Y))}, \dots]$ 

```

**End Function**

**Procedure** *SSCAE\_Attack* ( $S$ ,  $M$ ,  $N$ ,  $T$ ,  $Y$ ,  $W$ ,  $ST_{semantic}$ ,  $ST_{syntactic}$ ):

```

Initialization  $S^{adv} \leftarrow S$ 
 $WL = \text{Compute\_Importance}(S^{adv}, T, Y)$ 
foreach word  $w_i \in WL$  do
  | Apply MLM_Bert on  $w_{(i\pm j)}$  using  $W$ , gather semantic candidates ( $SC$ )
  |  $S_{semantic} = \text{USE}(S^{adv}, SC, w_i)$ 
  |  $S_{syntactic} = \text{GPT\_2}(S^{adv}, SC, w_i)$ 
  | Calculate dynamic thresholds ( $DT_{semantic}$ ,  $DT_{syntactic}$ ) based on
  | ( $S_{semantic}$ ,  $S_{syntactic}$ )
  | foreach semantic candidate  $sc_k \in SC$  do
  | | if  $S_{semantic}[sc_k] > (DT_{semantic}, ST_{semantic})$  AND
  | |  $S_{syntactic}[sc_k] > (DT_{syntactic}, ST_{syntactic})$  then
  | | | if  $\text{POS\_checker}(w_i, sc_k) == \text{True}$  then
  | | | |  $PC.add(sc_k)$ 
  | | | end
  | | end
  | end
  |  $GAP = []$ 
  | foreach purified candidate  $pc_k \in PC$  do
  | |  $S_t^{adv} = [\dots, w_{i-1}, pc_k, w_{i+1}, \dots]$ 
  | | if  $\text{argmax}(T(S_t^{adv})) \neq Y$  then
  | | | return  $S_t^{adv}$ 
  | | | end
  | | else if  $T_y(S_t^{adv}) < T_y(S^{adv})$  then
  | | |  $GAP.add(T_y(S^{adv}) - T_y(S_t^{adv}))$ 
  | | | end
  | | end
  | end
end
return  $S^{adv}$ 

```

**End Procedure**

generation of robust adversarial examples.

In essence, the main function, `SSCAE_Attack()`, acts as the entry point, receiving parameters such as the input text, truth label, target model, fixed-size window, hyper-parameters for local greedy search, and lower bounds for semantic and syntactic requirements. Subsequently, it orchestrates the iterative process of generating the ultimate adversarial example.

The `Compute_Importance()` function plays a pivotal role in computing the importance score of words within the input text, subsequently sorting them based on their respective importance levels. This step is crucial for identifying and prioritizing significant words for substitution.

The functions `USE()` and `GPT_2()` undertake the task of evaluating candidate substitutions by assigning semantic and syntactic scores, respectively. These scores are instrumental in gauging the suitability of substitutions in preserving the semantic and syntactic integrity of the input text.

Moreover, the `POS_checker()` function is responsible for validating the Part-Of-Speech (POS) tags of substitutions, ensuring alignment with the corresponding important word upon application to the input sample. This step is crucial for maintaining grammatical consistency throughout the substitution process.

Lastly, the `PRODUCT()` function facilitates the generation of all possible combinations of substitutions for  $M$  important words, thus enabling comprehensive exploration of substitution options within the input text.

Through the orchestrated execution of these functions within the SSCAE model, a systematic and iterative approach is employed to iteratively refine and optimize the generation of adversarial examples, ultimately enhancing their efficacy in deceiving target models.

### 3.3 Computational Experiments

#### **Dataset Selection and Experimental Framework:**

This section describes the text classification and Natural Language Inference (NLI) datasets employed for evaluation, along with the target models fine-tuned on various subsets. To mitigate statistical bias, each experiment is repeated five times with distinct randomly generated test samples. Reported results represent the average across all five repetitions. We utilize a comprehensive set of metrics to compare the adversarial attack performance of the proposed SSCAE model against recent baselines on these diverse target models. Additionally, we assess the efficacy of SSCAE across various Machine-Learning-as-a-Service (MLaaS) platforms encompassing diverse natural language understanding tasks like text classification.

#### **Implementation Details and Computational Resources:**

The SSCAE model implementation leverages several popular libraries: PyTorch Paszke et al. (2019) for deep learning operations, spaCy Honnibal & Montani (2017) for natural language processing tasks, and Gensim Rehurek & Sojka (2011) for topic modeling and document similarity computations. For loading target models and datasets, we utilize the transformers library Wolf et al. (2020). Additionally, TextAttack Morris et al. (2020) is employed to implement other adversarial attack methods for comparative analysis. Experiments were conducted on four Tesla V100 GPUs with 32GB of memory each. On average, the SSCAE model requires 2-4 hours to process 1000 randomly selected test instances on this infrastructure.

### *3.3.1 Expanding Experimentation with Datasets and Target Models*

To evaluate the robustness and efficacy of the SSCAE model comprehensively, a diverse array of datasets and target models were leveraged across eleven distinct Natural Language Processing (NLP) task experiments. The chosen datasets encompass four binary text classification datasets tailored for text classification tasks and three Natural Language Inference (NLI) datasets aimed at text entailment tasks.

The text classification datasets utilized in the experiments include the YELP Polarity Review (YELP) Zhang et al. (2015), Internet Movie Database (IMDb) Review Maas et al. (2011), Rotten Tomatoes Movie Reviews (MR) Pang & Lee (2005b), and Stanford Sentiment Treebank Version 2 (SST2) Socher et al. (2013b). On the other hand, the NLI datasets consist of Stanford NLI (SNLI) Bowman et al. (2015), Multi-NLI (MNLI) Williams et al. (2018), specifically MNLI-Matched, and MNLI-Mismatched. Comprehensive details regarding these datasets can be found in section 2.1.

The initial seven experiments focus on benchmarking the SSCAE model against three leading adversarial attack models, with the BERT model Devlin et al. (2019a) serving as the target model. Each experiment involves fine-tuning the BERT model on a different dataset to assess the SSCAE model's performance across varied text classification tasks. Notably, BERT stands as a prominent transformer-based language model that has demonstrated remarkable performance across diverse NLP tasks, often achieving human-level accuracies Murugesan (????); Rajpurkar et al. (2016).

Subsequent experiments delve into assessing the SSCAE model's effectiveness against four neural network target models other than BERT, namely WordLSTM Hochreiter & Schmidhuber (1997), ALBERT-Base Lan et al. (2020), ESIM Chen et al. (2017), and BERT-Large Devlin et al.

(2019a). Each of these models brings unique architectural nuances and computational characteristics to the table, contributing to a comprehensive evaluation of the SSCAE model’s robustness across various NLP frameworks.

In the final set of experiments, the focus shifts towards evaluating the SSCAE model’s performance on three prominent commercial MLaaS platforms: Google Cloud NLP, IBM Watson Natural Language Understanding (IBM Watson), and Microsoft Azure Text Analytics (Microsoft Azure). These experiments aim to gauge the SSCAE model’s efficiency in real-world deployment scenarios where the attacker lacks intricate knowledge of the target model’s architecture and training data. A comparative analysis is conducted against the TEXTBUGGER model Li et al. (2019), a benchmark in adversarial attack methodologies, to validate the SSCAE model’s efficacy across diverse deployment environments and platforms.

By diversifying the experimental setup to encompass a broad spectrum of datasets, target models, and deployment scenarios, the SSCAE model’s versatility and effectiveness in generating robust adversarial examples are comprehensively evaluated and validated.

### ***3.3.2 Comparison of Adversarial Attack Models Against BERT***

The outcomes of the initial seven experiments are summarized in Table 3.2, where the proposed SSCAE model is juxtaposed against TextFooler, BERT-Attack, BAE, HardLabel Maheshwary et al. (2021), and Liu2023 Liu et al. (2023) adversarial attack models. These experiments were conducted using 1000 randomly selected testing instances and the BERT target model. In order to assess the quality of the generated Adversarial Examples (AEs), four standard metrics Jin et al. (2020); Li et al. (2020b) were employed: after-attack accuracy, average perturbation percentage, average query

Dataset	E#	Original Acc	Adversarial Attack	Attacked Acc	Perturb %	Query #	Semantic Sim
YELP	1	95.6	TextFooler	6.3 (0.5)	12.1 (0.8)	758 (32)	0.75 (0.03)
			BERT-Attack/BAE	5.1 (0.4)	<b>4.1 (0.3)</b>	273 (19)	0.77 (0.03)
			HardLabel	5.6 (0.5)	5.4 (0.5)	247 (18)	0.79 (0.04)
			Liu2023	5.2 (0.3)	11.5 (0.9)	294 (21)	0.80 (0.03)
			SSCAE (ours)	<b>1.6 (0.2)</b>	4.8 (0.3)	<b>112 (14)</b>	<b>0.92 (0.03)</b>
IMDB	2	90.9	TextFooler	13.4 (0.7)	6.5 (0.6)	1145 (52)	0.85 (0.04)
			BERT-Attack/BAE	11.4 (0.5)	<b>4.4 (0.4)</b>	454 (31)	0.86 (0.03)
			HardLabel	10.8 (0.6)	4.6 (0.5)	521 (38)	0.85 (0.03)
			Liu2023	9.5 (0.4)	6.9 (0.6)	562 (36)	0.86 (0.04)
			SSCAE (ours)	<b>4.2 (0.3)</b>	10.8 (0.5)	<b>418 (26)</b>	<b>0.86 (0.03)</b>
SST2	3	93.0	TextFooler	14.1 (0.3)	16.7 (1.0)	103 (9)	0.85 (0.05)
			BERT-Attack/BAE	18.2 (0.4)	14.5 (0.3)	92 (8)	0.86 (0.04)
			HardLabel	15.4 (0.3)	14.9 (0.4)	86 (7)	0.85 (0.03)
			Liu2023	14.6 (0.3)	17.1 (0.5)	90 (6)	0.85 (0.04)
			SSCAE (ours)	<b>12.5 (0.2)</b>	<b>14.3 (0.3)</b>	<b>64 (5)</b>	<b>0.87 (0.03)</b>
MR	4	85.3	TextFooler	24.8 (0.4)	15.7 (0.3)	173 (9)	0.90 (0.03)
			BERT-Attack/BAE	19.2 (0.3)	15.2 (0.2)	126 (8)	0.91 (0.02)
			HardLabel	18.4 (0.3)	<b>15.0 (0.3)</b>	119 (7)	0.88 (0.03)
			Liu2023	18.1 (0.2)	17.3 (0.2)	135 (6)	0.89 (0.02)
			SSCAE (ours)	<b>15.6 (0.2)</b>	17.4 (0.2)	<b>54 (4)</b>	<b>0.91 (0.02)</b>
SNLI	5	89.4 (H)	TextFooler	16.9 (0.3)	18.0 (0.5)	74 (6)	0.73 (0.02)
			BERT-Attack/BAE	21.4 (0.4)	18.8 (0.5)	26 (3)	0.71 (0.02)
			HardLabel	16.3 (0.4)	16.9 (0.4)	37 (4)	0.71 (0.03)
			Liu2023	15.7 (0.4)	18.5 (0.6)	41 (5)	0.72 (0.03)
			SSCAE (ours)	<b>11.5 (0.3)</b>	<b>16.6 (0.4)</b>	<b>26 (3)</b>	<b>0.76 (0.02)</b>
MNLI-Matched	6	85.1 (P)	TextFooler	31.4 (0.6)	26.5 (0.5)	232 (10)	0.76 (0.03)
			BERT-Attack/BAE	18.9 (0.4)	14.5 (0.3)	64 (4)	0.78 (0.02)
			HardLabel	20.9 (0.5)	13.6 (0.4)	114 (6)	0.75 (0.02)
			Liu2023	18.2 (0.3)	24.4 (0.5)	127 (7)	0.77 (0.02)
			SSCAE (ours)	<b>9.2 (0.3)</b>	<b>12.3 (0.3)</b>	<b>42 (3)</b>	<b>0.82 (0.02)</b>
MNLI-Mismatched	7	82.1 (P)	TextFooler	26.1 (0.5)	27.4 (0.6)	186 (8)	0.75 (0.03)
			BERT-Attack/BAE	20.7 (0.4)	15.1 (0.3)	61 (4)	0.77 (0.02)
			HardLabel	19.7 (0.4)	14.6 (0.4)	77 (5)	0.76 (0.03)
			Liu2023	18.5 (0.5)	25.7 (0.6)	81 (6)	0.78 (0.03)
			SSCAE (ours)	<b>9.6 (0.3)</b>	<b>13.9 (0.3)</b>	<b>35 (3)</b>	<b>0.82 (0.02)</b>

Table 3.2 Average results of the proposed SSCAE, TextFooler, BERT-Attack, and BAE adversarial attack models on 1000 randomly selected testing instances from each of seven datasets using the BERT target model. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage; Query #: Query Number; H: Hypothesis; P: Premise.)

number, and average semantic consistency.

The ideal adversarial attack model should exhibit lower after-attack accuracy, perturbation percentage, and query number, while showcasing higher semantic consistency. Given the similarities and dataset sensitivity of BERT-Attack and BAE models Li et al. (2020b); Garg & Ramakrishnan (2020), we report the most favorable literature-available results across these two models in Table 3.2 (referred to as BERT-Attack/BAE). Overall, across the majority of datasets, SSCAE demonstrates superior performance compared to TextFooler, BERT-Attack/BAE, HardLabel, and Liu2023 models, as indicated in Table 3.2. This suggests the efficacy of the SSCAE model in generating high-quality adversarial examples. However, a more in-depth analysis of the individual metrics is necessary to fully understand the strengths and weaknesses of each model in different scenarios.

When considering the after-attack accuracy metric, it is evident that the SSCAE model consistently achieves superior results compared to other adversarial attack models across various experiments, particularly in datasets such as YELP, IMDB, SNLI, and MNLI. This margin of improvement is significant, highlighting the effectiveness of the SSCAE approach in generating AEs with lower after-attack accuracy.

In terms of perturbation percentage, the SSCAE model also demonstrates favorable outcomes, outperforming TextFooler and Liu2023 in most cases, with notable exceptions observed in the IMDB and MR datasets. However, when analyzing the MNLI dataset, the SSCAE model achieves a substantially lower perturbation percentage compared to TextFooler and Liu2023, indicating its efficacy in minimizing perturbations while maintaining adversarial potency.

Furthermore, when compared with Bert-Attack/BAE and HardLabel, the SSCAE model show-

cases either lower or comparable perturbation percentages across NLI and text classification tasks, except for the IMDB dataset where the results are on par. This consistency in perturbation reduction underscores the reliability of the SSCAE approach.

Regarding the query number metric, the SSCAE model consistently outperforms other adversarial attack models in all experiments, demonstrating a significant reduction in the number of queries required to generate AEs. This efficiency is crucial in practical scenarios where minimizing query overhead is essential.

In terms of semantic consistency, the SSCAE model exhibits superior performance across most experiments, except for the IMDB and MR datasets where results are comparable to other models. Notably, the SSCAE model maintains semantic consistency above 0.8 in the majority of experiments, a remarkable achievement compared to similar studies Jin et al. (2020); Li et al. (2020b).

The SSCAE model’s superiority is particularly evident in text classification datasets, where it achieves higher semantic similarity, lower after-attack accuracy, and query numbers, along with a comparable perturbation rate. Notably, the dynamic threshold concept for semantic consistency and syntactic correctness contributes significantly to the SSCAE model’s success in generating high-quality AEs.

Additionally, the incorporation of local greedy search enables the SSCAE model to simultaneously replace multiple important words, resulting in worst-case perturbations and achieving lower after-attack accuracy. Despite a slightly higher perturbation percentage observed in some experiments compared to BERT-Attack/BAE and HardLabel, the SSCAE model maintains higher semantic consistency using the dynamic threshold technique.

In conclusion, the SSCAE model represents a significant advancement in adversarial attack methods, generating imperceptible and efficient AEs across various datasets. Its ability to preserve semantic consistency while minimizing perturbations underscores its efficacy in practical applications.

Text entailment tasks (experiments 5 to 7) pose greater challenges compared to text classification tasks (experiments 1 to 4) due to the nature of their datasets, which consist of shorter sentences containing only a few words. Consequently, substituting crucial words in these short sentences can significantly elevate the perturbation percentage and diminish the semantic consistency of the generated examples. Even altering one or two essential words can substantially increase the perturbation percentage, given the limited pool of words available for modification. Despite these challenges, the SSCAE model demonstrates remarkable performance enhancements over other contemporary models in text alignment experiments, achieving notably higher semantic consistency alongside a considerable margin of after-attack accuracy.

It is important to highlight that in experiments involving document-level classification, such as those conducted on the YELP and IMDB datasets, the relatively longer input samples contribute to lower perturbation percentages. This observation suggests that the BERT model, serving as the target model, heavily relies on a select few important words to formulate predictions. Consequently, identifying and substituting these critical words can expose vulnerabilities inherent in Bert-base target models Li et al. (2020b).

Dataset	E#	Target Model	Original Acc	Attacked Acc	Adversarial Attack	Perturb %	Semantic Sim
YELP	8	WordLSTM	96.0	TextFooler	2.3 (0.2)	10.8 (0.6)	0.77 (0.03)
				BERT-Attack	1.7 (0.1)	5.5 (0.4)	0.80 (0.04)
				SSCAE (ours)	<b>0.56 (0.05)</b>	<b>5.3 (0.3)</b>	<b>0.93 (0.02)</b>
	9	ALBERT-Base	97.0	TextFooler	6.9 (0.3)	11.7 (0.4)	0.74 (0.02)
				BERT-Attack	5.4 (0.2)	<b>4.3 (0.3)</b>	0.78 (0.03)
				SSCAE (ours)	<b>1.9 (0.1)</b>	4.5 (0.2)	<b>0.91 (0.02)</b>
MNLI-Mismatched	10	ESIM	76.2	TextFooler	16.6 (0.6)	25.9 (0.7)	0.74 (0.03)
				BERT-Attack	10.5 (0.3)	20.2 (0.5)	0.77 (0.02)
				SSCAE (ours)	<b>6.1 (0.3)</b>	<b>17.5 (0.4)</b>	<b>0.83 (0.03)</b>
	11	BERT-Large	86.4	TextFooler	26.4 (0.5)	26.8 (0.6)	0.73 (0.03)
				BERT-Attack	20.3 (0.4)	14.9 (0.3)	0.76 (0.03)
				SSCAE (ours)	<b>9.9 (0.3)</b>	<b>13.6 (0.3)</b>	<b>0.81 (0.02)</b>

Table 3.3 Average results of the proposed SSCAE, TextFooler, and BERT-Attack adversarial attack models on 1000 randomly selected testing instances using WordLSTM (with YELP dataset), ALBERT-Base (with YELP dataset), ESIM (with MNLI mismatched), and BERT-Large (with MNLI mismatched) as the target models. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage)

### 3.3.3 Exploring Adversarial Attack Across Diverse Target Models

Table 3.3 provides an in-depth analysis of the experimental outcomes achieved by the SSCAE, TextFooler, and BERT-Attack adversarial attack methodologies across varied target models, namely WordLSTM, ALBERT-Base, ESIM, and BERT-Large, using distinct datasets. This comprehensive evaluation is conducted on 1000 randomly sampled test instances, shedding light on the effectiveness of each attack model in diverse contexts.

Across all experiments, the SSCAE model consistently exhibits remarkable superiority over TextFooler and BERT-Attack counterparts. Notably, it achieves a significant reduction in after-attack accuracy, coupled with higher semantic consistency, while maintaining a comparable perturbation

percentage.

In the context of text classification tasks, represented by WordLSTM-YELP and ALBERT-Base-YELP experiments, the SSCAE model excels by achieving an after-attack accuracy of less than 2% (specifically, 0.56% and 1.9% respectively), alongside perturbation percentages below 6% (5.3% and 4.5% respectively) and semantic similarity exceeding 90% (93% and 91% respectively).

Furthermore, when evaluating the SSCAE model's performance in the text entailment domain, specifically with the BERT-Large-MNLI-Mismatched experiment, results akin to those observed in Table 3.2 are obtained. This reaffirms the consistent effectiveness of the SSCAE model across diverse target models and tasks, underscoring its potential for robust adversarial attack strategies.

Table 3.4 showcases the experimental outcomes of the proposed SSCAE and TEXTBUGGER adversarial attack methodologies on 500 randomly selected testing instances sourced from the IMDB and YELP datasets, utilizing various MLaaS platform APIs. Across all four experiments, the SSCAE model demonstrates superior performance compared to the TEXTBUGGER model in terms of after-attack accuracy and semantic consistency metrics, while maintaining a comparable perturbation percentage.

TEXTBUGGER leverages a combination of word substitution and character manipulation techniques, with character-level perturbations often leading to alterations in word meaning and consequently resulting in a significant decrease in semantic similarity metric. In contrast, the SSCAE model consistently produces results closely aligned with those observed in Table 3.2 across all evaluation metrics.

Throughout the study's extensive experimentation encompassing 15 distinct scenarios, in-

Dataset	E#	Adversarial Attack	Original Acc	Targeted Model	Attacked Acc	Perturb %	Semantic Sim
IMDB	12	TEXTBUGGER	86.2	Google Cloud NLP	23.3 (1.4)	<b>2.6 (0.2)</b>	0.72 (0.04)
			91.8	Microsoft Azure	2.1 (0.3)	6.4 (0.4)	0.70 (0.03)
			91.4	IBM Watson	4.7 (0.4)	8.9 (0.5)	0.69 (0.04)
	13	SSCAE (ours)	86.2	Google Cloud NLP	<b>17.6 (0.9)</b>	<b>2.6 (0.2)</b>	<b>0.87 (0.03)</b>
			91.8	Microsoft Azure	<b>1.6 (0.2)</b>	<b>6.0 (0.4)</b>	<b>0.85 (0.02)</b>
			91.4	IBM Watson	<b>2.8 (0.3)</b>	<b>7.5 (0.4)</b>	<b>0.85 (0.03)</b>
YELP	14	TEXTBUGGER	90.6	Google Cloud NLP	8.3 (0.7)	<b>4.5 (0.3)</b>	0.75 (0.03)
			85.2	Microsoft Azure	3.9 (0.3)	5.1 (0.3)	0.73 (0.04)
			86.5	IBM Watson	4.6 (0.3)	5.9 (0.4)	0.72 (0.03)
	15	SSCAE (ours)	90.6	Google Cloud NLP	<b>5.6 (0.5)</b>	4.7 (0.3)	<b>0.91 (0.03)</b>
			85.2	Microsoft Azure	<b>2.1 (0.2)</b>	<b>5.0 (0.3)</b>	<b>0.89 (0.03)</b>
			86.5	IBM Watson	<b>2.8 (0.2)</b>	<b>5.6 (0.4)</b>	<b>0.87 (0.02)</b>

Table 3.4 Average results of the proposed SSCAE and TEXTBUGGER adversarial attack models on 500 randomly selected testing instances from the IMDB and YELP datasets using different NLP-based cloud APIs. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage)

corporating diverse datasets, target models, and adversarial attack methodologies for both text classification and entailment tasks, the SSCAE model exhibits remarkable prowess in generating imperceptible adversarial examples while preserving significantly higher semantic consistency (typically exceeding 0.8) and substantially reducing query numbers.

### 3.4 Ablation Studies

This section delves into an ablation study aimed at scrutinizing several of SSCAE’s key hyperparameters and methodologies (illustrated by different steps in Figure 3.1). Additionally, it showcases adversarial attack examples generated by SSCAE and demonstrates the model’s resilience through a human evaluation, validating the findings from our automatic adversarial attack experiments.

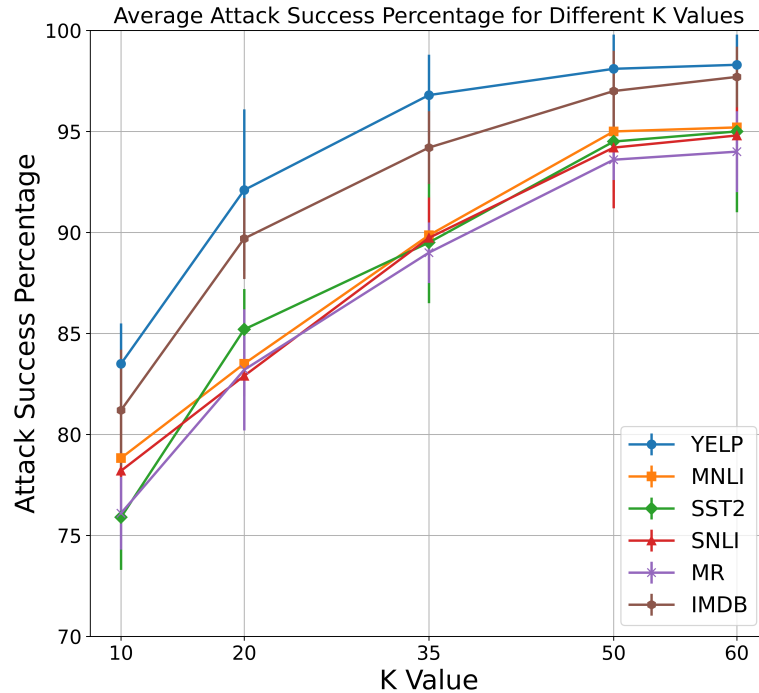


Figure 3.2 An ablation study of  $K$  context-aware substitutions set generated by the BERT-MLM model in step 3 of the SSCAE model

### 3.4.1 Investigating the Impact of $K$ in Context-Aware Substitutions

Figure 3.2 unveils an ablation study concerning the parameter  $K$ , which governs the number of context-aware substitutions in step 3 of Figure 3.1. The x-axis denotes five potential values of  $K$ , namely 10, 20, 35, 50, and 60, while the y-axis represents the Attack Success Percentage (ASP) metric across six distinct experiments (experiments 1-5, 7) corresponding to YELP, IMDB, SST2, MR, SNLI, and MLNI-Mismatched datasets.

ASP metric computation entails utilizing AEs exclusively from testing input samples correctly classified by the target model to ensure a fair comparative analysis. Generally, a higher value of  $K$  implies a greater number of substitutions for pivotal words, thereby enhancing the likelihood of

Dataset	E#	Original Acc	Method	Attacked Acc	Perturb %	Semantic Sim
YELP	1	95.6	w Semantic	1.6 (0.2)	4.8 (0.3)	0.92 (0.03)
			w/o Semantic	1.2 (0.1)	3.2 (0.2)	0.73 (0.02)
SNLI	5	89.4	w Semantic	11.5 (0.7)	16.6 (0.9)	0.76 (0.02)
			w/o Semantic	7.5 (0.5)	10.4 (0.7)	0.60 (0.03)

Table 3.5 Results of an ablation study on the proposed SSCAE model using YELP (text classification task) and SNLI (text entailment task) experiments with and without semantic refinement (semantic similarity/consistency) in step 3 of the SSCAE model (Figure 3.2) (E#: Experiment Number; Perturb %: Perturbation Percentage; w: With; wo: Without)

crafting AEs (characterized by a reduced perturbation percentage) capable of deceiving the model (yielding a higher ASP) despite leveraging linguistic enhancements.

However, our experiments reveal a diminishing rate of ASP improvement (accompanied by an increase in semantic consistency reduction) as  $K$  surpasses 50. Beyond  $K = 60$ , the ASP improvement rate becomes negligible, while the semantic consistency reduction rate becomes more pronounced. Consequently, for the proposed SSCAE model,  $K = 60$  emerges as the near-optimal choice across all experiments, ensuring a balance between ASP enhancement and semantic consistency preservation.

### 3.4.2 Significance of Semantic Refinement

Table 3.5 outlines the outcomes of a sensitivity analysis conducted on the SSCAE model using the YELP (representing the text classification task) and SNLI (representing the text entailment task) datasets (experiments 1 and 5) with and without the semantic refinement step (semantic similarity in step 3, Figure 3.1). Upon removing semantic refinement, after-attack accuracy, perturbation

Dataset	E#	Semantic Embedding Method	Attacked Acc	Semantic Sim
YELP	1	USE	1.6 (0.2)	0.92 (0.04)
		Sentence-BERT	2.8 (0.3)	0.92 (0.03)
IMDB	2	USE	4.2 (0.4)	0.86 (0.03)
		Sentence-BERT	8.7 (0.7)	0.84 (0.04)
SNLI	5	USE	11.5 (1.1)	0.76 (0.03)
		Sentence-BERT	13.6 (1.3)	0.74 (0.02)
MNLI-Mismatched	7	USE	9.6 (0.8)	0.82 (0.04)
		Sentence-BERT	12.5 (0.9)	0.81 (0.03)

Table 3.6 A comparison study between the utilization of the USE and Sentence-BERT semantic embedding models over experiments corresponding to YELP, SNLI, IMDB, and MNLI-Mismatched datasets in step 3 of the SSCAE model (Figure 3.2). The standard deviation is included within the brackets. (E#: Experiment Number; USE: Universal Sentence Encoder)

percentage, and semantic consistency respectively decreased from 1.6% to 1.2% (i.e., enhanced), 4.8% to 3.2% (i.e., enhanced), and 0.92 to 0.73 (i.e., deteriorated). While there was a marginal improvement in after-attack accuracy and perturbation percentage metrics, the decline in semantic consistency suggests that, on average, the AEs lost their original semantics. Consequently, the AEs generated by SSCAE would likely be discernible to humans. Therefore, semantic refinement emerges as a crucial component in crafting high-quality and imperceptible AEs within the proposed SSCAE model.

In step 3 of the SSCAE model (Figure 3.1), USE assigns a semantic similarity score to each candidate substitution. Table 3.6 juxtaposes the use of USE with another potential semantic embedding model, Sentence-BERT Reimers & Gurevych (2019), across experiments corresponding to YELP, IMDB, SNLI, and MNLI-Mismatched datasets. These findings underscore that USE outperforms

<b>Dataset</b>	<b>E#</b>	<b>Method</b>	<b>Attacked Acc</b>	<b>Perturb %</b>	<b>Semantic Sim</b>
YELP	1	Constant Threshold	2.2 (0.3)	4.8 (0.4)	0.87 (0.02)
		Average_threshold	3.0 (0.4)	6.4 (0.5)	0.90 (0.03)
		Median_threshold	2.8 (0.4)	6.5 (0.5)	0.90 (0.04)
		TopN_threshold	2.0 (0.3)	4.7 (0.4)	0.92 (0.03)
		Top_maxes_distance	1.6 (0.2)	4.8 (0.3)	0.92 (0.03)
MNLI-Mismatched	7	Constant Threshold	11.2 (0.8)	13.7 (1.1)	0.78 (0.04)
		Average_threshold	12.8 (0.9)	17.4 (1.4)	0.79 (0.02)
		Median_threshold	12.6 (0.9)	17.5 (1.5)	0.81 (0.03)
		TopN_threshold	10.9 (0.8)	13.9 (1.0)	0.81 (0.05)
		Top_maxes_distance	9.6 (0.7)	13.9 (0.9)	0.82 (0.03)

Table 3.7 A comparative study of the Average\_threshold, Median\_threshold, TopN\_threshold, and Top\_maxes\_distance heuristics to compute the dynamic threshold in step 3 of the SSCAE model (Figure 3.1). The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage)

Sentence-BERT not only in achieving lower after-attack accuracy but also in maintaining slightly higher semantic consistency. Consequently, USE was chosen as the primary semantic embedding model in the proposed SSCAE model.

### 3.4.3 Exploration of Dynamic Thresholds

Table 3.7 delineates a comparative analysis between the constant threshold and four proposed heuristics (dynamic thresholds) for semantic and syntactic refinements in step 3 of the SSCAE model (Figure 3.1). Both Average\_threshold and Median\_threshold yielded similar outcomes across all metrics of after-attack accuracy, perturbation percentage, and semantic consistency, likely due to their analogous mathematical methodologies for the refinement process. Conversely, TopN\_threshold and

<b>Dataset</b>	<b>E#</b>	$\lambda$	<b>Attacked Acc</b>	<b>Perturb %</b>	<b>Semantic Sim</b>
YELP	1	0.5	1.8 (0.1)	4.8 (0.4)	0.90 (0.04)
		1	1.6 (0.2)	4.8 (0.5)	0.92 (0.03)
		5	1.9 (0.2)	4.9 (0.3)	0.89 (0.02)
		10	2.1 (0.2)	4.9 (0.5)	0.88 (0.02)

Table 3.8 The efficacy of the minor hyperparameter  $\lambda$  within the Top\_maxes\_distance technique. (E#: Experiment Number; Perturb %: Perturbation Percentage)

Top\_maxes\_distance generally outperformed Average\_threshold and Median\_threshold. Notably, Top\_maxes\_distance exhibited superior after-attack accuracy results compared to TopN\_threshold, albeit with comparable perturbation percentage and semantic consistency results. Additionally, an investigation was conducted to ascertain the optimal value for the minor hyperparameter  $\lambda$  in Top\_maxes\_distance. As depicted in Table 3.8,  $\lambda=1$  demonstrated superior performance over other examined values. Relative to the dynamic thresholds, the constant threshold exhibited better perturbation percentage, inferior semantic consistency, and moderately satisfactory after-attack accuracy results. In summary, Top\_maxes\_distance emerges as the preferred choice for the SS-CAE model owing to its superior evaluation outcomes compared to other threshold techniques (constant/dynamic).

#### 3.4.4 Exploring Window Sizes

Figure 3.3 showcases an ablation study investigating the impact of various window sizes on the selection of unique sets of the top  $K = 50$  context-aware substitutions. Increasing the window size

( $|W|$ ) leads to a higher number of substitutions, consequently elevating the probability of generating adversarial examples that effectively deceive the target model. Notably, there is a substantial enhancement in the Average Success Percentage (ASP) metric from  $|W| = 2$  to  $|W| = 4$ . However, beyond  $|W| = 4$ , the rate of ASP improvement begins to plateau. Moreover, at  $|W| \geq 4$ , the marginal gain in ASP diminishes, while the computational complexity associated with searching for optimal adversarial samples escalates significantly. Consequently, a window size of  $|W| = 4$  has been selected for refining substitutions in our experiments.

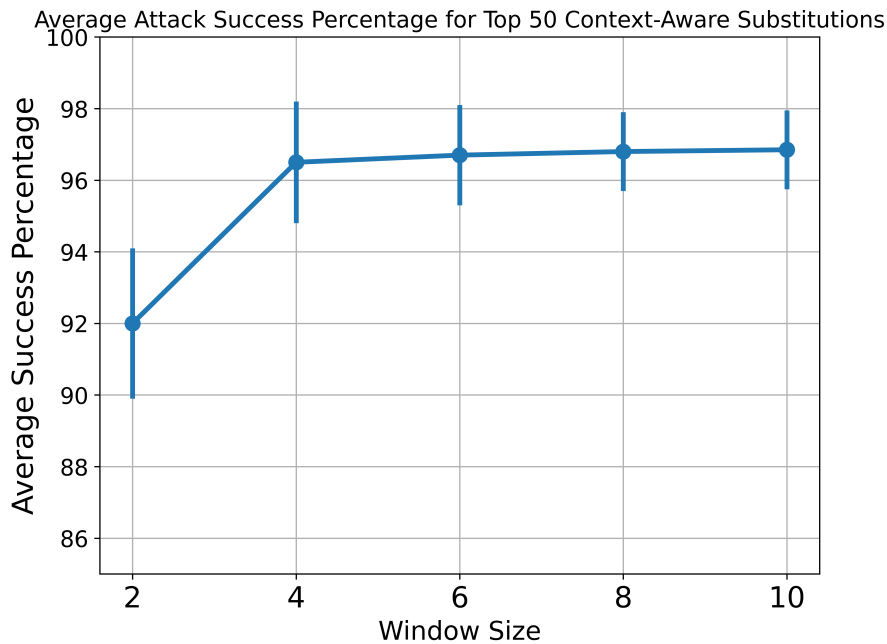


Figure 3.3 An ablation study of the influence of different window sizes on the retrieval of distinct sets of the top  $K = 50$  context-aware substitutions.

### 3.4.5 Optimizing Local Greedy Search

The local greedy search method offers a theoretically broad spectrum of parameter values for  $M$  and  $N$ . However, considering the substantial computational overhead involved in evaluating the

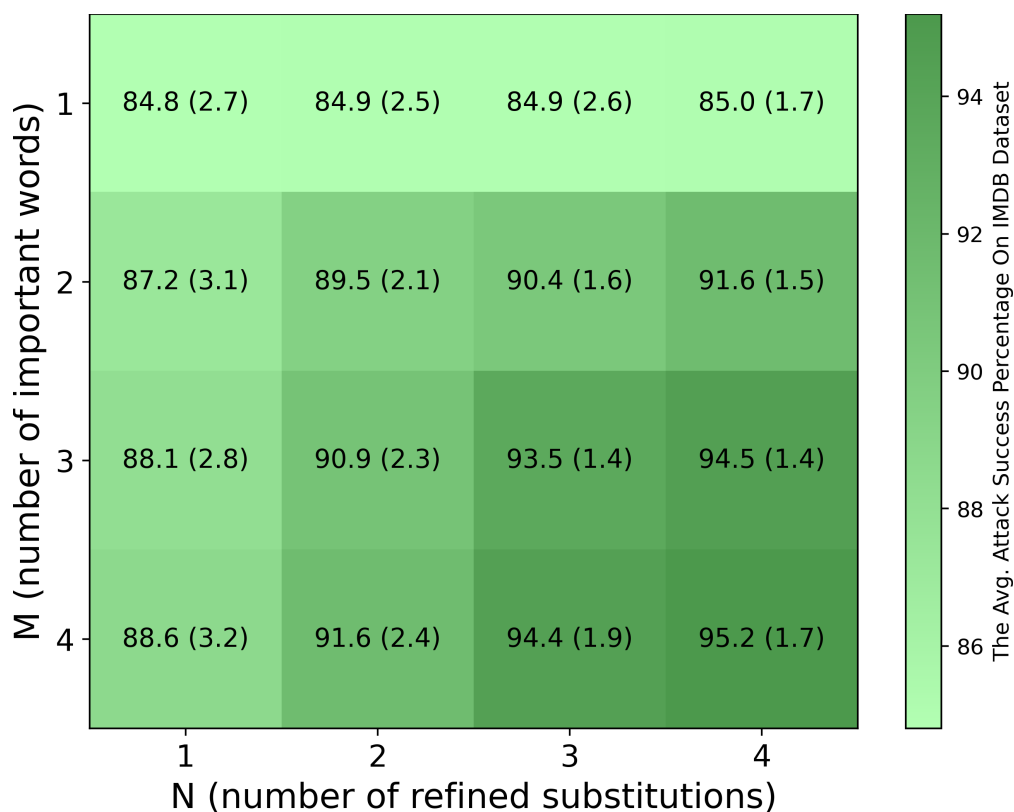


Figure 3.4 The impact of the number of important words and candidate substitutions on the average attack success percentage of the IMDB dataset.

numerous combinations of substitutions (denoted as  $N^M$ ), it becomes imperative to constrain the range of these hyperparameters to smaller values. Based on our empirical findings, a pragmatic approach for maintaining computational efficiency involves setting both  $M$  and  $N$  within the range of 1 to 4 (i.e.,  $[1, 4]$ ). This strategy effectively limits the total number of combinations to a maximum of 256, striking a balance between computational tractability and exhaustive evaluation.

Figure 3.4 illustrates the impact of varying numbers of important words,  $M$ , and refined substitutions,  $N$ , on the average ASP benchmark. When  $M = 1$ , the average ASP hovers around 85% for different values of  $N$  (i.e., the number of substitutions). Increasing the number of important words to two results in a notable enhancement, with the average ASP reaching 91.6% for  $N = 4$

substitutions. Similarly, for higher numbers of important words (e.g.,  $M = 3$  or  $M = 4$ ), the average ASP exhibits a consistent trend of improvement across varying  $N$  values. Opting for a larger  $N$  offers a greater likelihood of discovering AEs with higher ASPs. Nonetheless, the exponentially expanding search space associated with larger  $N$  values poses computational challenges. In our experiments, we adopt  $N = 4$  for refined substitutions and  $M = 3$  for important words, striking a balance between effectiveness and computational feasibility.

### ***3.4.6 Illustrative Examples of Adversarial Texts***

Table 3.9 showcases four pairs of original input samples alongside their corresponding SSCAE-generated adversarial attack examples derived from experiments conducted on the YELP and MNLI-Mismatch datasets. In the YELP dataset, the first pair exhibits the replacement of the adjective "Nice" with "Fantastic," while in the second pair, the noun "clinic" is substituted with "hospital." Despite the arguable differences in the nuances between these adjectives and nouns, the overall semantics of the original samples remain largely preserved in the adversarial attack examples. Moreover, the grammatical and syntactic integrity of the original samples is maintained in the generated adversarial examples. These commendable outcomes are attributed to the linguistic filters employed in step 3 of the SSCAE model (refer to Figure 3.1), which significantly enhance the imperceptibility and fluency of the generated adversarial examples.

Similar lexical analyses can be applied to the adversarial examples generated for the MNLI-Mismatch dataset, further highlighting the efficacy of the SSCAE model in crafting imperceptible and contextually coherent adversarial texts.

Dataset	E#	Pair #	Data Type (Prediction)	Sentence
YELP	1	1	Input Sample (P)	Yes! Awesome soy cap, scone, and atmosphere. <b>Nice</b> place to hang out & read, and free WiFi with no login procedure.
			Adversarial Example (N)	Yes! Awesome soy cap, scone, and atmosphere. <b>Fantastic</b> place to hang out & read, and free WiFi with no login procedure.
	2	Input Sample (N)	Refused to take my cat, which had passed away, for cremation cause I had not been to the <b>clinic</b> previously...	
		Adversarial Example (P)	Refused to take my cat, which had passed away, for cremation cause I had not been to the <b>hospital</b> previously...	
MNLI-Mismatched	1		Hypothesis	Poirot was disappointed with me
			Input Sample (NE)	Still, it would be interesting to know. 109 Poirot looked at me very <b>earnestly</b> , and again shook his head
			Adversarial Example (E)	Still, it would be interesting to know. 109 Poirot looked at me very <b>carefully</b> , and again shook his head
	7		Hypothesis	Talking about privacy is a complicated topic, there are a couple different ways of talking about it, for example privacy is something that disturbs your private state...
			Input Sample (E)	The first kind of invasion of the first type of privacy seems invaded to me in very much everyday in this country but in the second type at least overtly uh where someone comes in and uh finds out information about you that should be private uh does not seem uh um obviously <b>everyday</b>
	2	Adversarial Example (NE)	The first kind of invasion of the first type of privacy seems invaded to me in very much everyday in this country but in the second type at least overtly uh where someone comes in and uh finds out information about you that should be private uh does not seem uh um obviously <b>routine</b>	

Table 3.9 Examples of original and adversarial sentences from experiments corresponding to YELP and MNLI datasets (E#: Experiment Number; Pair #: Pair Number; P: Positive; N: Negative; E: Entailment; NE: Neutral)

### ***3.4.7 Adversarial Training***

To bolster the resilience of target models against diverse adversarial attacks, particularly focusing on the proposed SSCAE model, we employ the adversarial training technique. Additionally, fine-tuning is performed on a target model, specifically the BERT-Large model, utilizing both the original data and adversarial examples crafted by the SSCAE model. Adversarial examples from the MR and SNLI training sets, which effectively deceived BERT-Large, are curated and merged with the original training set. Subsequently, the augmented dataset is utilized to fine-tune the target model. The objective is to assess the robustness of this adversarially fine-tuned model against two distinct adversarial attack models: TextFooler and SSCAE.

In Table 3.10, it is observed that after adversarial fine-tuning, both the after-attack accuracy and perturbation ratio increase for both the TextFooler and SSCAE attack methods. Moreover, the original accuracy of the adversarially-trained model remains comparable to that of the model trained on clean datasets. These results suggest that adversarial training using a high-quality set of adversarial examples renders attacks more challenging, thereby enhancing the model's robustness. Remarkably, the target model demonstrates greater resilience against TextFooler compared to the proposed SSCAE model, underscoring the potential for improving a model's robustness against future attacks by training it with efficient adversarial examples generated by the SSCAE model.

### ***3.4.8 Human Evaluation***

In the realm of computer science, particularly in the field of natural language processing (NLP), the evaluation of adversarial examples (AEs) holds significant importance in assessing their correctness

<b>Dataset</b>	<b>E#</b>	<b>Train Method</b>	<b>Original Acc</b>	<b>Adversarial Attack</b>	<b>Attacked Acc</b>	<b>Perturb %</b>
MR	4	Normal Fine-tune	85.3	SSCAE (ours)	15.6 (1.0)	17.4 (0.9)
		Adversarial Fine-tune	84.8	TextFooler	38.2 (1.7)	24.6 (1.3)
		Adversarial Fine-tune	84.8	SSCAE (ours)	32.4 (1.5)	21.6 (1.1)
SNLI	5	Normal Fine-tune	89.4	SSCAE (ours)	11.5 (0.6)	16.6 (0.8)
		Adversarial Fine-tune	88.9	TextFooler	26.1 (1.4)	23.9 (1.4)
		Adversarial Fine-tune	88.9	SSCAE (ours)	21.2 (1.2)	20.5 (1.0)

Table 3.10 Average results of adversarial attacks on the normal and adversarial fine-tuning of the BERT-Large model using two different attack methods: TextFooler and the proposed SSCAE. The standard deviation is included within the brackets. (E#: Experiment Number; Perturb %: Perturbation Percentage)

across various dimensions such as imperceptibility, language fluency, and semantic consistency Li et al. (2020b). Despite this significance, only a limited number of studies have undertaken comprehensive human evaluations Li et al. (2020b).

Table 3.11 provides insights into a human assessment conducted to evaluate the quality and fluency of AEs generated by different adversarial attack models, including the proposed SSCAE, TextFooler, BERT-Attack, and BAE models. The evaluation was conducted on datasets representing text classification tasks (YELP) and text alignment tasks (MLNI-mismatched) using BERT as the target model (experiments 1 and 7).

For this evaluation, three PhD candidates from the Department of Applied Linguistics at Georgia State University were selected based on their robust scientific backgrounds. It's noteworthy that these candidates maintained independence from the authors of this research work. They were tasked with assessing 100 randomly selected input samples labeled as "Original" and their corresponding adversarial samples labeled as "Adversarial". The samples were shuffled to ensure unbiased

Dataset	Adversarial Attack	Data Type	Human Accuracy	Meaningfulness	Grammar Correctness
YELP	-	Original	92.5	4.2	4.0
	TextFooler	Adversarial	83.4	3.1	3.2
	BERT-Attack	Adversarial	84.7	3.8	3.6
	BAE	Adversarial	84.8	3.7	3.5
	SSCAE (ours)	Adversarial	86.0	4.0	3.8
MNLI-Mismatched	-	Original	91.2	3.9	4.1
	TextFooler	Adversarial	74.2	3.0	3.1
	BERT-Attack	Adversarial	75.6	3.5	3.6
	BAE	Adversarial	75.3	3.5	3.5
	SSCAE (ours)	Adversarial	77.4	3.7	3.7

Table 3.11 Human Evaluation Task (E#: Experiment Number)

evaluation.

In determining the correctness of each sample, the majority opinion of the students was considered. If the majority correctly estimated the class of a sample, it was counted as one correct human estimation.

The results revealed that the SSCAE model exhibited superior performance compared to other adversarial attack methods, particularly in the YELP experiment. Despite this, there existed only a small gap (6.5%) between human estimations of original samples and SSCAE-generated adversarial samples, indicating the model’s capability to generate high-quality and imperceptible adversarial examples.

However, in the MNLI-Mismatched experiment, where human-crafted hypothesis and premise sentences share a considerable amount of the same words, the performance gap between original and adversarial samples was more pronounced. This can be attributed to the negative impact of perturbations introduced by adversarial attacks on human assessment accuracy.

In addition to estimating the class of each sample, every student participating in the evaluation process was tasked with providing two Likert scores ranging from 1 to 5 for each sample, regardless of whether it was Original or Adversarial. The first score pertained to the meaningfulness of the sample, where a score of 1 denoted meaninglessness and 5 indicated high meaningfulness. The second score assessed the grammatical correctness of the sample, with a score of 1 representing incorrect grammar and 5 representing grammatical correctness.

Interestingly, the results revealed that the SSCAE-generated adversarial samples consistently achieved higher scores in both meaningfulness and grammatical correctness metrics compared to adversarial samples generated by other state-of-the-art methods. Specifically, in terms of meaningfulness and grammatical correctness, the SSCAE-generated adversarial samples exhibited a small gap of 0.2 (0.2 and 0.4, respectively) between the Original and Adversarial scores in both the YELP and MNLI-Mismatched experiments.

This finding indicates that the SSCAE-generated adversarial samples were semantically and grammatically aligned with the Original samples, showcasing the model’s ability to produce AEs that are indistinguishable from genuine data. Such results underscore the effectiveness of SSCAE in generating high-quality adversarial examples while preserving semantic and grammatical coherence.

### **3.5 Summary and Discussion**

In this chapter, we introduced SSCAE, a novel framework aimed at producing contextually relevant adversarial examples (AEs) while preserving fundamental linguistic features encompassing semantics, syntax, and grammar within the realm of computer science. SSCAE leverages the Bidirectional

Encoder Representations from Transformers (BERT) Masked Language Model (MLM) to generate a pool of potential word substitutions crucial for the adversarial perturbation process. Additionally, it integrates three refinement strategies to ensure the preservation of linguistic properties in the final perturbations. Firstly, we proposed a dynamic thresholding technique, designed to enhance the efficacy of perturbations by selectively incorporating substitutions surpassing a dynamically adjusted threshold. Secondly, we introduced a word permutation-substitution mechanism named local greedy search, aimed at systematically generating high-quality adversarial examples by iteratively exploring neighboring word substitutions.

The efficacy of the SSCAE model was empirically evaluated against three contemporary black-box methods: TextFooler, BERT-Attack, and BAE, across a spectrum of popular and challenging text classification and entailment tasks within the computer science domain. Our experiments revealed that SSCAE consistently outperformed these baseline models by a significant margin, as evidenced by metrics such as post-attack accuracy, average query count, and semantic consistency. Moreover, SSCAE demonstrated superior performance in terms of average perturbation percentage across four out of seven Natural Language Processing (NLP) tasks considered.

Furthermore, a human evaluation was conducted to assess the perceptual quality of the generated adversarial examples, affirming the effectiveness of the proposed SSCAE model in producing high-quality and imperceptible perturbations. Notably, the modular design of SSCAE facilitates its seamless integration with advanced components, thus enabling flexible extensions tailored to diverse research contexts. Nonetheless, certain practical operations such as insertion and deletion, while integral to adversarial perturbation techniques, remain unexplored within the SSCAE framework,

presenting avenues for future investigation and enhancement.

## CHAPTER 4

### **RobustEmbed: Robust Sentence Embeddings Using Self-Supervised Contrastive Pre-Training**

#### **4.1 Introduction**

Recent studies have showcased the remarkable performance of Pre-trained Language Models (PLMs) in acquiring contextual word embeddings Devlin et al. (2019b), thereby fostering enhanced generalization across a myriad of Natural Language Processing (NLP) tasks Yang et al. (2019); He et al. (2021); Ding et al. (2023). The scope of PLMs has expanded to encompass the acquisition of universal sentence embeddings, exemplified by noteworthy models such as the Universal Sentence Encoder (USE) Cer et al. (2018) and Sentence-BERT Reimers & Gurevych (2019), which adeptly capture the semantic essence embedded within textual inputs. This paradigm shift in representation learning not only facilitates feature extraction for classification endeavors but also bolsters the efficacy of large-scale semantic search operations Neelakantan et al. (2022).

The evaluation of PLM-based sentence representations hinges upon two pivotal attributes: generalization and robustness. Despite significant strides made in crafting universal sentence embeddings leveraging PLMs Reimers & Gurevych (2019); Zhang et al. (2020b); Ni et al. (2022); Neelakantan et al. (2022); Wang et al. (2023); Bölücü et al. (2023), it is imperative to acknowledge that while these representations exhibit commendable performance across diverse downstream classification tasks Sun et al. (2019); Gao et al. (2021), thereby showcasing proficiency in generalization, they manifest limitations concerning robustness in adversarial scenarios and are susceptible to a plethora of adversarial assaults Nie et al. (2020); Wang et al. (2021). Existing literature Garg & Ramakrishnan (2020); Wu et al. (2023); Hauser et al. (2023) underscores the inherent fragility of these

representations, particularly BERT-based representations, which can be easily deceived by minor alterations (i.e. replacing a few words) to the input sentence structure.

In this work, we introduce RobustEmbed, a novel framework for generating robust sentence embeddings that meticulously addresses both crucial attributes: generalization and robustness. The central idea revolves around the introduction of minimal adversarial perturbations to the input text, coupled with the utilization of the contrastive objective proposed by Chen et al. (2020) to learn high-quality sentence embeddings. Unlike traditional approaches that perturb the raw text directly, RobustEmbed operates on the embedding space, thereby fostering a positive correlation with generalization and promoting increased invariance. Our framework leverages both the original embedding and the perturbed embedding as "positive pairs," while concurrently utilizing other sentence embeddings within the same mini-batch as "negatives." Through the contrastive objective function, RobustEmbed effectively discriminates positive pairs from the negatives, enhancing the discernibility between semantically similar sentences and dispersing those with divergent semantics.

By integrating norm-bounded adversarial perturbation and contrastive objectives, our methodology augments the robustness of similar sentences while concurrently dispersing sentences with varying semantics. This straightforward yet efficient approach culminates in the generation of superior sentence embeddings, as evidenced by comprehensive evaluations across both generalization and robustness benchmarks.

We conduct a comprehensive array of experiments spanning various text representation and Natural Language Processing (NLP) tasks to thoroughly assess the efficacy of RobustEmbed. These tasks encompass semantic textual similarity (STS) tasks (Conneau & Kiela (2018)), transfer tasks

Conneau & Kiela (2018), and evaluations using TextAttack Morris et al. (2020).

The experimental setup comprises two primary series of evaluations. The first series focuses on assessing the quality of sentence embeddings concerning semantic similarity and overall natural language understanding tasks. Subsequently, the latter series scrutinizes the robustness of the RobustEmbed framework against state-of-the-art adversarial attacks.

Noteworthy findings from our experiments indicate significant improvements in robustness exhibited by RobustEmbed. Notably, it effectively reduces the attack success rate from 75.51% to 39.62% when subjected to the BERTAttack technique, showcasing remarkable resilience against adversarial assaults. Similar enhancements in robustness are observed across other adversarial attack strategies employed in our evaluations.

Furthermore, RobustEmbed achieves notable performance gains of 1.20% and 0.40% on STS tasks and NLP transfer tasks, respectively, particularly when leveraging the BERT<sub>base</sub> encoder. These performance enhancements underscore the effectiveness and versatility of the RobustEmbed framework across a diverse array of NLP tasks, further solidifying its potential for real-world deployment and adoption.

#### ***4.1.1 Contribution***

Our main contributions in this paper are summarized as follows:

- We introduce RobustEmbed, a novel self-supervised framework for sentence embeddings that generates robust representations capable of withstanding various adversarial attacks. Existing sentence embeddings are susceptible to such attacks, highlighting a vulnerability in their security. RobustEmbed fills this gap by employing high-risk perturbations within a novel

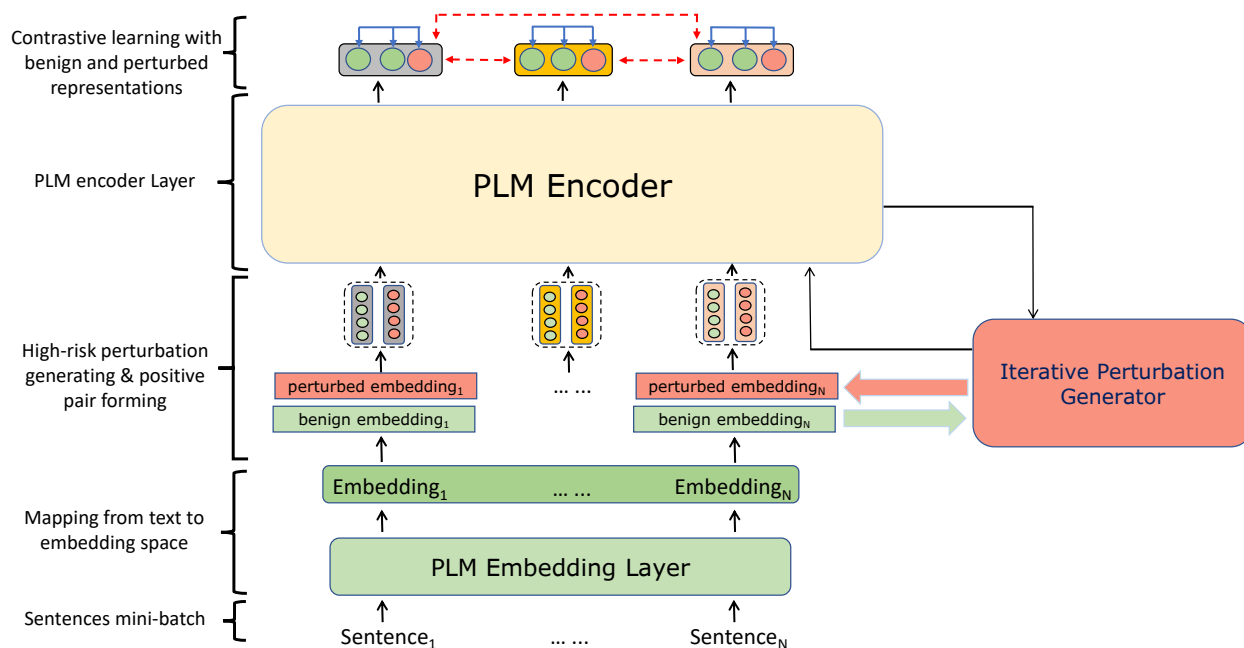


Figure 4.1 The general architecture of the RobustEmbed framework. In contrastive learning step, a blue arrow indicate gathering positive pairs together, and a red arrow refers to keeping distance among negative pairs

contrastive learning approach.

- We conduct extensive experiments to demonstrate the efficacy of RobustEmbed across various text representation tasks and against state-of-the-art adversarial attacks. Empirical results confirm the high efficiency of our framework in terms of both generalization and robustness benchmarks.
- To facilitate further research in this important area, our source code is available in the RobustEmbed Repository.

## 4.2 The Proposed Adversarial Self-supervised Contrastive Learning

We present RobustEmbed, an elegant yet potent method devised to generate universal text representations by subjecting a self-supervised contrastive learning model to adversarial training. By leveraging a Pre-trained Language Model (PLM) denoted as  $f_{\theta}(\cdot)$  as the encoder and operating on a vast unsupervised dataset  $\mathcal{D}$ , RobustEmbed endeavors to pre-train  $f_{\theta}(\cdot)$  on  $\mathcal{D}$  with the dual objectives of bolstering the efficacy of sentence embeddings across a spectrum of Natural Language Processing (NLP) tasks, thereby fostering improved generalization, while simultaneously fortifying their resilience against an array of adversarial attacks, thus enhancing robustness. Algorithm 2 elucidates the methodology employed by our framework to engender norm-bounded perturbations through an iterative process, strategically perturbing the  $f_{\theta}(\cdot)$  model to treat the perturbed embeddings as distinct instances. Subsequently, our framework adopts a contrastive learning strategy to maximize the similarity between the embedding of an input instance and the adversarial embedding of its positive counterpart.

Moreover, Figure 4.1 furnishes an overarching depiction of our RobustEmbed framework, which endeavors to instill adversarial robustness within representations. This framework entails a collaborative iterative process between the perturbation generator and the  $f_{\theta}(\cdot)$  model, aimed at generating high-risk perturbations conducive to adversarial contrastive learning during the final training phase. The ensuing sections delve into the primary constituents of our framework, offering a comprehensive analysis of the underlying training objective.

---

**Algorithm 2: RobustEmbed Algorithm**


---

**Input** : Epoch number  $E$ , PLM Encoder  $\mathbf{f}_\theta$ , dataset of raw sentences  $\mathcal{D} = \{x_i\}_{i=1}^N$ , perturbation  $\delta$ , dropout masks  $m_1$  and  $m_2$ , perturbation bound  $\epsilon$ , step sizes  $\alpha$  and  $\beta$ , learning rate  $\eta$ , perturbation modulator  $\lambda$ , regularization parameter  $\gamma$ , perturbation generation iterators  $K$  and  $T$ , contrastive learning objective  $\mathcal{L}_{con,\theta}$  (eq. 5.9)

**Output** : Robust Sentence Representation

```

for epoch = 1, ...,  $E$  do
  for minibatch  $B \subset \mathcal{D}$  do
     $\delta^1 \sim \mathcal{N}(0, \sigma^2 I)$ 
     $\mathbf{X} = \mathbf{f}_\theta.\text{embedding}(B, m_1)$ 
     $\mathbf{X}^+ = \mathbf{f}_\theta.\text{embedding}(B, m_2)$ 
    for  $t = 1, \dots, \max(K, T)$  do
       $g(\delta^t) = \nabla_{\delta} \mathcal{L}_{con,\theta}(\mathbf{X} + \delta^t, \{\mathbf{X}^+\})$ 
      if  $t \leq K$  then
         $\delta_{pgd}^{t+1} = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^t + \alpha g(\delta^t) / \|g(\delta^t)\|_P)$ 
      end
      if  $t \leq T$  then
         $\delta_{fgsm}^{t+1} = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^t + \beta \text{sign}(g(\delta^t)))$ 
      end
    end
     $\delta_f = \lambda \delta_{pgd}^K + (1 - \lambda) \delta_{fgsm}^T$ 
     $\mathcal{L}_{RobustEmbed, \theta} := \mathcal{L}_{con,\theta}(\mathbf{X}, \{\mathbf{X}^+, \mathbf{X} + \delta_f\})$ 
     $\mathcal{L}_{total} := \mathcal{L}_{RobustEmbed, \theta} + \gamma \mathcal{L}_{con,\theta}(\mathbf{X} + \delta_f, \{\mathbf{X}^+\})$ 
     $\theta = \theta - \eta \nabla_{\theta} \mathcal{L}_{total}$ 
  end
end

```

---

#### 4.2.1 Perturbation Generation

As the initial step, RobustEmbed endeavors to generate subtle perturbations capable of deceiving the Machine Learning (ML) model, inducing erroneous predictions, while simultaneously maintaining near-imperceptibility to human observers. The framework adopts an amalgamated approach, combining elements from the Projected Gradient Descent (PGD) and Fast Gradient Sign Method (FGSM) algorithms to craft a perturbation that maximizes the self-supervised contrastive loss,

thereby facilitating effective discrimination among diverse instances. RobustEmbed iteratively applies this combined approach, employing T-step FGSM and K-step PGD, meticulously reinforcing invariance within the embedding space, ultimately culminating in augmented generalization and robustness.

Specifically, leveraging the PLM-based encoder denoted as  $f_\theta(\cdot)$  and an input sentence  $x$ , RobustEmbed subjects the sentence to the  $f_\theta(\cdot)$  model twice, employing standard dropout on each occasion to obtain two distinct embeddings, denoted as  $(X, X^+)$ , serving as "positive pairs" Gao et al. (2021). The framework follows a series of steps to update the perturbation independently for PGD and FGSM in iterations  $k + 1$  and  $t + 1$  respectively:

$$\boldsymbol{\delta}_{pgd}^{k+1} = \Pi_{\|\boldsymbol{\delta}\|_P \leq \epsilon}(\boldsymbol{\delta}^k + \alpha g(\boldsymbol{\delta}^k) / \|g(\boldsymbol{\delta}^k)\|_P), \quad (4.1)$$

$$\boldsymbol{\delta}_{fgsm}^{t+1} = \Pi_{\|\boldsymbol{\delta}\|_P \leq \epsilon}(\boldsymbol{\delta}^t + \beta \text{sign}(g(\boldsymbol{\delta}^t))), \quad (4.2)$$

where  $g(\boldsymbol{\delta}^n) = \nabla_{\boldsymbol{\delta}} \mathcal{L}_{con, \theta}(\mathbf{X} + \boldsymbol{\delta}^n, \{\mathbf{X}^+\})$  with  $n = t$  or  $k$  represents the gradient of the contrastive learning loss concerning  $\boldsymbol{\delta}$ . The perturbation is constrained within the  $\ell_\infty$  norm-ball around the input embedding with a radius of  $\epsilon$ , and  $\Pi$  denotes the projection of the perturbation onto the  $\epsilon$ -ball. Furthermore,  $\alpha$  and  $\beta$  denote the step sizes of the attacks, while  $\text{sign}(\cdot)$  returns the sign of the vector. Essentially, T-step FGSM and K-step PGD are mathematically analogous when  $P$  is either 2 or  $\infty$ . Their primary distinctions lie in the number of iterations (i.e., T and K) and the step size of the attack (i.e.,  $\alpha$  and  $\beta$ ) utilized to modify the input perturbation, ultimately yielding a distinct high-level perturbation. The final perturbation is obtained through the combination of

T-step FGSM and K-step PGD, as depicted by Equation 5.7:

$$\delta_{final} = \lambda \delta_{pgd}^K + (1 - \lambda) \delta_{fgsm}^T, \quad (4.3)$$

where  $0 \leq \lambda \leq 1$  modulates the relative significance of each individual perturbation in the generation of the final perturbation.

#### 4.2.2 Robust Contrastive Learning

In pursuit of robust representations via self-supervised contrastive learning, we define an adversarial learning objective that follows a min-max formulation, aiming to minimize the maximum risk posed by any perturbation  $\delta$  Madry et al. (2018c):

$$\arg \min_{\theta} \mathbb{E}_{(x) \sim \mathcal{D}} [\max_{\|\delta\| \leq \epsilon} \mathcal{L}_{con, \theta}(\mathbf{X} + \delta, \{\mathbf{X}^+\})], \quad (4.4)$$

where  $\mathbf{X} + \delta$  denotes the adversarial embedding generated via iterative gradient-based perturbation generation (eq. 5.7). Our framework leverages adversarial examples crafted within the embedding space, rather than operating on the original raw text, resulting in a pre-trained model that exhibits robustness against m-way instance-wise adversarial attacks. The framework adopts the contrastive learning objective to maximize the similarity between clean examples and their adversarial perturbations by incorporating the adversarial example as an additional element in the positive set:

$$\mathcal{L}_{RobustEmbed, \theta} := \mathcal{L}_{con, \theta}(x, \{x^{pos}, x^{adv}\}), \quad (4.5)$$

$$\mathcal{L}_{total} := \mathcal{L}_{RobustEmbed, \theta} + \gamma \mathcal{L}_{con, \theta}(x^{adv}, \{x^{pos}\}), \quad (4.6)$$

where  $x^{adv}$  denotes the adversarial perturbation of the input sample  $x$  in the embedding space, and  $\gamma$  represents a regularization parameter. The first component of the total contrastive loss (eq. 5.10) aims to optimize the similarity between the input sample  $x$ , its positive pair, and its adversarial perturbation, while the second component serves to regularize the loss by encouraging the convergence of the adversarial perturbation and the positive pair of  $x$ .

### 4.3 Evaluation and Experimental Results

This section presents a comprehensive array of experiments aimed at substantiating the efficacy of our proposed framework concerning both generalization and robustness metrics. In the initial two sets of experiments, we delve into the performance analysis of our framework across seven semantic textual similarity (STS) tasks and six transfer tasks encompassed within the SentEval framework<sup>1</sup>, thereby scrutinizing the inherent generalization prowess of our framework in generating proficient sentence embeddings. Subsequently, the latter series of experiments undertakes an examination of the resilience exhibited by the embeddings against a battery of five state-of-the-art adversarial attacks, elucidating the robustness capacity of our framework in furnishing resilient text representations. Additionally, Appendices A elucidate the intricate facets of our training procedure.

---

<sup>1</sup><https://github.com/facebookresearch/SentEval>

### *4.3.1 Semantic Textual Similarity (STS) Tasks*

Our framework undergoes rigorous evaluation across a comprehensive array of seven semantic textual similarity (STS) tasks, encompassing STS 2012–2016 Agirre et al. (2012, 2013, 2014, 2015, 2016), STS Benchmark Cer et al. (2017), and SICK-Relatedness Marelli et al. (2014). In these experiments, we rely solely on fixed sentence embeddings without recourse to any training datasets or regressors. To provide a benchmark for our framework’s performance, we subject it to comparison against a spectrum of unsupervised sentence embedding methodologies, including: 1) fundamental techniques such as GloVe Pennington et al. (2014) and average BERT or RoBERTa embeddings; 2) post-processing methodologies like BERT-flow Li et al. (2020a) and BERT-whitening Su et al. (2021); and 3) cutting-edge methodologies such as SimCSE Gao et al. (2021), ConSERT Yan et al. (2021), USCAL Miao et al. (2021), and ATCL Rima et al. (2022). To validate the findings of the SimCSE, ConSERT, and USCAL frameworks, we endeavor to reproduce their results using our customized configuration for BERT and RoBERTa encoders.

The empirical outcomes, as depicted in Table 4.1, unequivocally showcase the superior performance of our RobustEmbed framework compared to a plethora of sentence embedding methodologies across a vast majority of the semantic textual similarity tasks. Our framework attains the highest averaged Spearman’s correlation among the state-of-the-art approaches. Specifically, leveraging the BERT encoder, our framework surpasses the second-best embedding methodology, USCAL, by a margin of 1.20%. Furthermore, RobustEmbed emerges with the highest score in the majority of individual STS tasks (6 out of 7) when juxtaposed with alternative embedding methodologies, exhibiting comparable performance to the SimCSE methodology on the STS16 task. With regards

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
GloVe embeddings (avg.) <sup>♡</sup>	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
BERT <sub>base</sub> (first-last avg.) <sup>♣</sup>	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
BERT <sub>base</sub> -flow <sup>♣</sup>	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
BERT <sub>base</sub> -whitening <sup>♣</sup>	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
ConSERT-BERT <sub>base</sub>	64.56	78.55	69.16	79.74	76.00	73.91	67.35	72.75
ATCL-BERT <sub>base</sub>	67.14	80.86	71.73	79.50	76.72	79.31	70.49	75.11
SimCSE-BERT <sub>base</sub>	68.66	81.73	72.04	80.53	<b>78.09</b>	79.94	71.42	76.06
USCAL-BERT <sub>base</sub>	69.30	80.85	72.19	81.04	77.52	81.28	71.98	76.31
★RobustEmbed-BERT <sub>base</sub>	<b>70.52</b>	<b>82.13</b>	<b>73.56</b>	<b>82.38</b>	77.72	<b>82.97</b>	<b>73.24</b>	<b>77.51</b>
RoBERTa <sub>base</sub> -whitening <sup>□</sup>	46.99	63.24	57.23	71.36	68.99	61.36	62.91	61.73
ConSERT-RoBERTa <sub>base</sub>	66.90	79.31	70.33	80.57	77.95	81.42	68.16	74.95
SimCSE-RoBERTa <sub>base</sub>	68.75	80.81	71.19	81.79	79.35	82.62	69.56	76.30
USCAL-RoBERTa <sub>base</sub>	69.28	81.15	72.81	81.47	<b>80.55</b>	83.34	70.94	77.08
★RobustEmbed-RoBERTa <sub>base</sub>	<b>69.71</b>	<b>81.77</b>	<b>73.34</b>	<b>81.98</b>	79.74	<b>83.70</b>	<b>71.10</b>	<b>77.33</b>
USCAL-RoBERTa <sub>large</sub>	68.70	<b>81.84</b>	74.26	82.52	<b>80.01</b>	83.14	76.30	78.11
★RobustEmbed-RoBERTa <sub>large</sub>	<b>68.92</b>	81.53	<b>74.35</b>	<b>82.91</b>	79.98	<b>83.93</b>	<b>76.93</b>	<b>78.36</b>

Table 4.1 Semantic Similarity performance on STS tasks (Spearman’s correlation, “all” setting) for sentence embedding models. We emphasize the top-performing numbers among models that share the same pre-trained encoder. ♡: results from Reimers & Gurevych (2019); ♣: results from Gao et al. (2021); All remaining results have been reproduced and reevaluated by our team. The ★ symbol shows our framework.

to the RoBERTa encoder, spanning both the base version and the large version, RobustEmbed outshines state-of-the-art embeddings in five out of seven STS tasks while achieving the highest averaged Spearman’s correlation.

### *4.3.2 Transfer Tasks*

In this experiment, we delve into transfer tasks to comprehensively gauge the efficacy and versatility of our framework, RobustEmbed, across a diverse range of text classification tasks, spanning sentiment analysis to paraphrase identification. Our evaluation encompasses six transfer tasks, each representing a distinct facet of text classification prowess: CR Hu & Liu (2004), SUBJ Pang & Lee (2004), MPQA Wiebe et al. (2005), SST2 Socher et al. (2013a), and MRPC Dolan & Brockett (2005). For detailed insights into these tasks, refer to section 2.1. Adhering to the standardized methodology delineated in Conneau & Kiela (2018), we adopt a logistic regression classifier, leveraging the fixed sentence embeddings as the foundation for our experimental framework. To validate the robustness and efficacy of our approach, we meticulously replicate the experimental configurations of SimCSE, ConSERT, and USCAL frameworks, tailoring the settings to our bespoke BERT and RoBERTa encoders.

The comprehensive tabulated results presented in Table 4.2 unequivocally attest to the superior performance of our RobustEmbed framework in terms of average accuracy when juxtaposed with a gamut of alternative sentence embedding methodologies. Noteworthy is the discernible superiority exhibited by our framework, boasting a commendable margin of 0.40% over the second-best embedding methodology when employing the BERT encoder. Moreover, RobustEmbed clinches the top spot in four out of six text classification tasks, reaffirming its prowess and versatility. Analogous trends persist with the RoBERTa encoder, spanning both the base version and the large version, further consolidating the robustness and efficacy of our framework across diverse text classification scenarios.

Model	MR	CR	SUBJ	MPQA	SST2	MRPC	Avg.
GloVe embeddings (avg.) ♣	77.25	78.30	91.17	87.85	80.18	72.87	81.27
Skip-thought ♥	76.50	80.10	93.60	87.10	82.00	73.00	82.05
BERT-[CLS] embedding ♣	78.68	84.85	94.21	88.23	84.13	71.13	83.54
ConSERT-BERT <sub>base</sub>	79.52	87.05	94.32	88.47	85.46	72.54	84.56
SimCSE-BERT <sub>base</sub>	81.29	86.94	94.72	89.49	<b>86.70</b>	75.13	85.71
USCAL-BERT <sub>base</sub>	81.54	87.12	95.24	89.34	85.71	75.84	85.80
★RobustEmbed-BERT <sub>base</sub>	<b>81.94</b>	<b>87.45</b>	95.04	<b>89.88</b>	86.47	<b>76.40</b>	<b>86.20</b>
SimCSE-RoBERTa <sub>base</sub>	81.15	87.15	92.38	86.79	<b>86.24</b>	75.49	84.87
USCAL-RoBERTa <sub>base</sub>	<b>82.15</b>	87.22	92.76	87.74	84.39	76.20	85.08
★RobustEmbed-RoBERTa <sub>base</sub>	81.49	<b>87.54</b>	<b>93.37</b>	<b>87.95</b>	84.63	<b>76.62</b>	<b>85.27</b>
USCAL-RoBERTa <sub>large</sub>	<b>82.84</b>	87.97	93.12	88.48	<b>86.28</b>	76.41	85.85
★RobustEmbed-RoBERTa <sub>large</sub>	82.38	<b>88.27</b>	<b>93.91</b>	<b>88.79</b>	86.01	<b>77.11</b>	<b>86.08</b>

Table 4.2 Results of transfer tasks for different sentence embedding models. ♣: results from Reimers & Gurevych (2019); ♥: results from Zhang et al. (2020b); We emphasize the top-performing numbers among models that share the same pre-trained encoder. All remaining results have been reproduced and reevaluated by our team. The ★ symbol shows our framework.

### 4.3.3 Adversarial Attacks

In this section, we embark on a rigorous examination of the robustness of our sentence embedding framework against an array of adversarial attacks, juxtaposed against two prominent state-of-the-art sentence embedding models: SimSCE Gao et al. (2021) and USCAL Miao et al. (2021). Our evaluation entails the fine-tuning of a BERT-based Pre-trained Language Model (PLM) employing distinct embedding methodologies across seven distinct text classification and natural language inference tasks, namely MRPC Dolan & Brockett (2005), YELP Zhang et al. (2015), IMDb Maas

et al. (2011), Movie Reviews (MR) Pang & Lee (2005b), SST2 Socher et al. (2013a), Stanford NLI (SNLI) Bowman et al. (2015), and Multi-NLI (MNLI) Williams et al. (2018). For an exhaustive elucidation of these tasks, refer to section 2.1.

To meticulously scrutinize the robustness of the fine-tuned models, we subject them to adversarial onslaughts utilizing the TextAttack framework Morris et al. (2020), thereby probing the efficacy against five highly efficient adversarial attack methodologies: TextBugger Li et al. (2019), PWWS Ren et al. (2019), TextFooler Jin et al. (2020), BAE Garg & Ramakrishnan (2020), and BERTAttack Li et al. (2020c). To gain profound insights into the intricacies of these attacks, we furnish detailed expositions in Appendix C.

It's imperative to note that adaptive attacks are incapable of generating adversarial assaults using the primary algorithm of our framework, owing to its exclusive operation within the embedding space, while the input instances of sentence embeddings are derived from raw textual data. To uphold statistical integrity, each experiment undergoes five iterations, with each iteration comprising 1000 adversarial attack samples; the ensuing results depicted in this section represent the average outcomes derived from these iterative trials.

Table 4.3 provides a comprehensive overview of the attack success rates exhibited by five adversarial attack methodologies targeting three distinct sentence embeddings, inclusive of our pioneering framework. Our embedding framework consistently surpasses the performance of the other two embedding methodologies, showcasing notably diminished attack success rates across the spectrum of text classification and natural language inference tasks. As a result, RobustEmbed attains the paramount position, boasting the lowest average attack success rate across all adversarial

Adversarial Attack	Model	IMDB	MR	SST2	YELP	MRPC	SNLI	MNLI-Mismatched	Avg.
TextFooler	SimCSE-BERT <sub>base</sub>	75.32	65.53	71.49	79.67	80.07	72.65	68.54	72.61
	USCAL-BERT <sub>base</sub>	61.94	48.71	55.38	62.30	60.18	54.82	53.74	56.72
	RobustEmbed-BERT <sub>base</sub>	<b>40.55</b>	<b>32.69</b>	<b>36.17</b>	<b>44.25</b>	<b>38.88</b>	<b>37.61</b>	<b>35.63</b>	<b>37.97</b>
TextBugger	SimCSE-BERT <sub>base</sub>	52.21	42.04	49.67	56.19	56.73	45.39	40.16	48.91
	USCAL-BERT <sub>base</sub>	39.16	27.37	31.90	41.25	37.86	30.79	25.45	33.40
	RobustEmbed-BERT <sub>base</sub>	<b>23.70</b>	<b>18.03</b>	<b>20.24</b>	<b>28.58</b>	<b>20.89</b>	<b>19.07</b>	<b>16.33</b>	<b>20.98</b>
PWWS	SimCSE-BERT <sub>base</sub>	64.41	55.73	60.48	67.54	68.15	56.09	52.58	60.71
	USCAL-BERT <sub>base</sub>	51.95	40.67	45.29	52.30	46.86	50.92	39.37	46.77
	RobustEmbed-BERT <sub>base</sub>	<b>33.63</b>	<b>28.15</b>	<b>30.56</b>	<b>29.94</b>	<b>25.51</b>	<b>27.16</b>	<b>28.49</b>	<b>29.06</b>
BAE	SimCSE-BERT <sub>base</sub>	73.50	61.83	68.27	75.15	77.84	69.06	65.43	70.15
	USCAL-BERT <sub>base</sub>	58.57	46.19	51.72	59.49	58.38	50.90	51.16	53.77
	RobustEmbed-BERT <sub>base</sub>	<b>37.35</b>	<b>29.82</b>	<b>32.08</b>	<b>41.66</b>	<b>36.45</b>	<b>34.17</b>	<b>31.98</b>	<b>34.79</b>
BERTAttack	SimCSE-BERT <sub>base</sub>	78.42	66.94	73.59	80.87	82.16	74.35	72.22	75.51
	USCAL-BERT <sub>base</sub>	63.23	51.08	57.73	63.96	63.05	55.41	55.86	58.62
	RobustEmbed-BERT <sub>base</sub>	<b>42.30</b>	<b>34.76</b>	<b>38.81</b>	<b>45.15</b>	<b>39.97</b>	<b>39.08</b>	<b>37.24</b>	<b>39.62</b>

Table 4.3 Attack success rates of various adversarial attacks applied to three sentence embeddings (SimCSE-BERT, USCAL-BERT, and RobustEmbed-BERT) across five text classification and two natural language inference tasks.

attack techniques. These compelling results unequivocally affirm the robustness and resilience of our embedding framework, while concurrently shedding light on the susceptibilities inherent in the two prevailing state-of-the-art sentence embeddings when subjected to diverse adversarial attacks.

Figure 4.2 presents a graphical representation of the average number of queries and the consequent accuracy reduction resulting from 1000 attacks on two distinct fine-tuned sentence embeddings. The green data points correspond to attacks conducted on the RobustEmbed framework, while the red points signify attacks on the USCAL approach Miao et al. (2021). Each pair of connected points

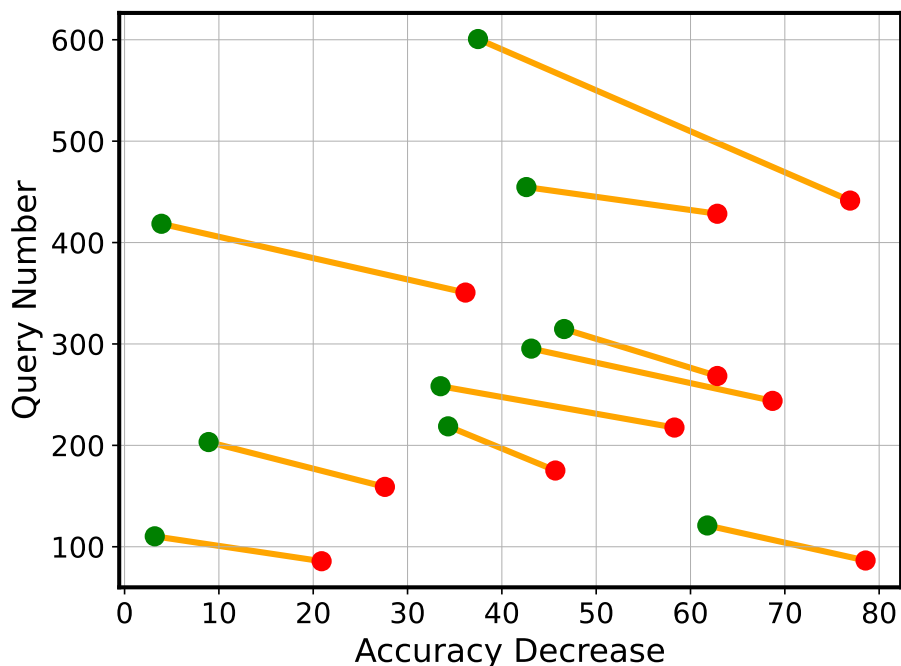


Figure 4.2 Average number of queries and the resulting accuracy reduction for a set of 1000 attacks on two fine-tuned sentence embeddings. Green points represent attacks on the RobustEmbed framework, while red points represent attacks on the USCAL approach.

corresponds to a specific attack technique. Ideally, an ideal sentence embedding should reside in the top-left quadrant of the diagram, indicating that the attack technique requires a larger number of queries to deceive the target model while causing minimal degradation in performance. The figure distinctly illustrates that, across all attack scenarios, RobustEmbed exhibits superior stability compared to the USCAL method. Essentially, RobustEmbed necessitates a larger number of queries, consequently resulting in a lower accuracy reduction, thereby indicating better performance when compared to USCAL. This noteworthy observation persists across all employed adversarial attacks, further corroborating the inherent robustness of our framework.

#### ***4.3.4 Adversarial Semantic Textual Similarity (AdvSTS)***

We present a novel evaluation task known as Adversarial Semantic Textual Similarity (AdvSTS) to assess the robustness of sentence embeddings within our framework. AdvSTS adopts an efficient adversarial technique, such as TextFooler, to manipulate a pair of input sentences, prompting the target model to produce a regression score that significantly deviates from the ground truth label. This manipulation leads to the creation of an adversarial STS dataset, where all benign instances from the original dataset are transformed into adversarial examples. Similar to the traditional STS task, AdvSTS employs Pearson’s correlation metric to evaluate the correlation between the predicted similarity scores generated by the target model and the human-annotated similarity scores for the adversarial dataset. This task provides insights into the resilience of our framework against adversarial manipulations in semantic textual similarity assessments.

Table 4.4 displays the attack success rates of five distinct adversarial attack techniques, namely TextFooler, TextBugger, PWWS, BAE, and BERTAttack, applied to three different sentence embeddings, including our framework. These assessments are conducted specifically for two AdvSTS tasks, namely AdvSTS-B and AdvSICK-R. It is noteworthy that our embedding framework consistently outperforms the other two embedding methods, demonstrating markedly lower attack success rates across both AdvSTS tasks and all utilized adversarial attack techniques.

In conclusion, the comprehensive experiments conducted and the outcomes delineated in Tables 4.1, 4.2, 4.3, and 4.4, along with the insights presented in Figure 4.2, furnish compelling evidence of the outstanding performance of RobustEmbed across a spectrum of text representation and classification tasks, as well as its resilience against diverse adversarial attacks and tasks. These

Adversarial Attack	Model	AdvSTS-B	AdvSICK-R	Avg.
TextFooler	SimCSE-BERT <sub>base</sub>	21.07	24.17	22.62
	USCAL-BERT <sub>base</sub>	16.52	18.71	17.62
	RobustEmbed-BERT <sub>base</sub>	<b>7.48</b>	<b>8.95</b>	<b>8.22</b>
TextBugger	SimCSE-BERT <sub>base</sub>	27.49	28.34	27.91
	USCAL-BERT <sub>base</sub>	21.52	24.88	23.20
	RobustEmbed-BERT <sub>base</sub>	<b>11.76</b>	<b>13.01</b>	<b>12.39</b>
PWWS	SimCSE-BERT <sub>base</sub>	24.15	26.82	25.49
	USCAL-BERT <sub>base</sub>	21.28	23.65	22.47
	RobustEmbed-BERT <sub>base</sub>	<b>13.56</b>	<b>14.44</b>	<b>14.00</b>
BAE	SimCSE-BERT <sub>base</sub>	26.92	28.81	27.86
	USCAL-BERT <sub>base</sub>	22.92	25.48	24.20
	RobustEmbed-BERT <sub>base</sub>	<b>11.13</b>	<b>12.82</b>	<b>11.98</b>
BERTAttack	SimCSE-BERT <sub>base</sub>	31.60	32.85	32.23
	USCAL-BERT <sub>base</sub>	26.02	28.51	27.26
	RobustEmbed-BERT <sub>base</sub>	<b>12.99</b>	<b>13.18</b>	<b>13.09</b>

Table 4.4 Attack success rates of five adversarial attack techniques applied to three sentence embeddings (SimCSE, USCAL, and RobustEmbed) across two Adversarial STS (AdvSTS) tasks (i.e. AdvSTS-B and AdvSICK-R).

results substantiate the assertion that our framework exhibits exceptional capabilities in terms of both generalization and robustness, thus highlighting its potential as a versatile and effective approach for generating top-tier sentence embeddings.

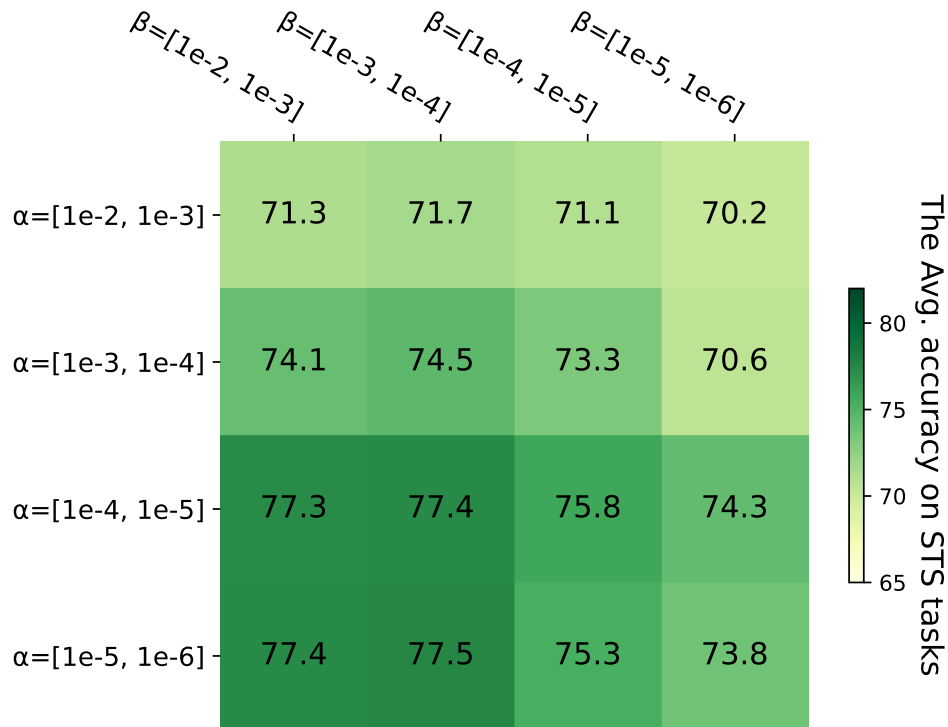


Figure 4.3 The impact of step sizes in perturbation generation on the average performance of STS tasks.

#### 4.4 Ablation Studies

This section delves into the analysis of the impact of four pivotal hyperparameters on the overall performance of our methodology. Leveraging  $BERT_{base}$  as the encoder, we assess these hyperparameters using the development set of STS tasks.

##### 4.4.1 Optimizing Step Sizes in Perturbation Generation

As outlined in Algorithm 2, the RobustEmbed framework integrates two step sizes, denoted as  $\alpha$  and  $\beta$ , to iteratively update the perturbations during the PGD and FGSM processes, respectively.

Figure 4.3 illustrates the collaborative impact of varying the ranges for these two step sizes in

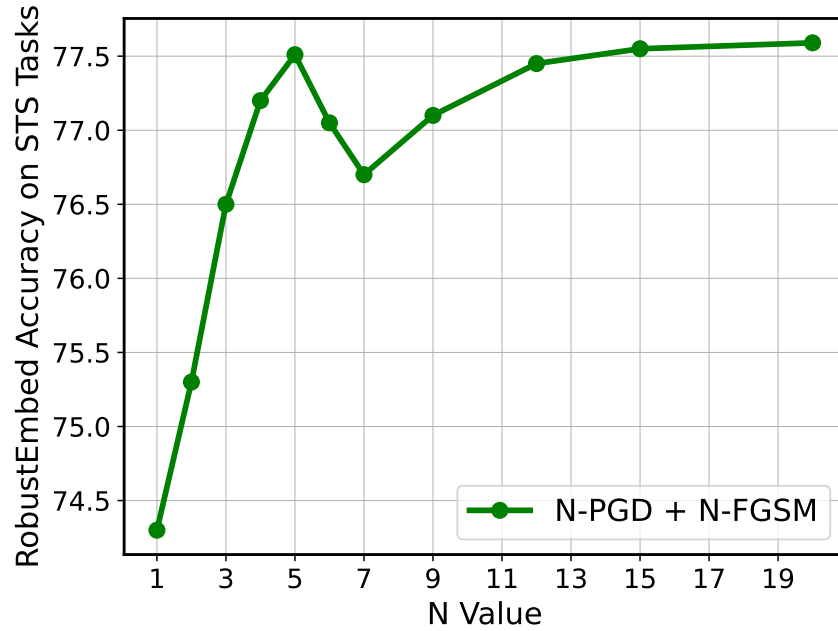


Figure 4.4 The effect of the step number (denoted as  $N = K$  or  $T$ ) in the  $T$ -step FGSM and  $K$ -step PGD methods on the averaged correlation of the different Semantic Textual Similarity (STS) tasks.

generating high-risk perturbations, a crucial aspect for achieving an effective contrastive learning objective. The findings reveal significant enhancements when  $\beta$  is constrained to a lower range while  $\alpha$  is expanded to an upper range. Specifically, superior performance is observed when  $\alpha$  and  $\beta$  are confined within the ranges of  $[1e-4, 1e-6]$  and  $[1e-2, 1e-4]$ , respectively. Consequently, we opt for  $\alpha = 1e-5$  and  $\beta = 1e-3$  for our experiments, as this configuration yields the most favorable outcomes among the various combinations. Additionally, fine-tuning the step sizes in perturbation generation contributes to optimizing the efficiency and effectiveness of the contrastive learning process within our framework, ultimately enhancing the robustness and generalization capabilities of the generated sentence embeddings.

#### 4.4.2 Optimizing Step Numbers in Perturbation Generation

In the perturbation generation process, RobustEmbed employs T-step FGSM and K-step PGD iterations to derive high-risk adversarial perturbations crucial for the contrastive learning objective. For ease of analysis, we simplify the perturbation generation iterations by setting  $K = T$ .

Figure 4.4 elucidates the influence of varying step numbers ( $N = K$  or  $T$ ) on effectiveness. It is observed that there is a gradual enhancement in performance as  $N$  escalates from 1 to 9; however, beyond  $N=9$ , the degree of improvement diminishes significantly. Additionally, higher values of  $N$  result in prolonged running times and inequitable resource allocation. Therefore, we opt for  $N=5$  for our experiments, as it strikes a balance between computational efficiency and performance optimization.

#### 4.4.3 Norm Constraint

In ensuring the imperceptibility of the generated adversarial instances, RobustEmbed regulates the magnitude of the perturbation vector, denoted as  $\delta$ . This control over the perturbation magnitude is vital to maintain the natural appearance of the adversarial examples. To achieve this, three widely used norm functions, namely  $L_1$ ,  $L_2$ , and  $L_\infty$ , are employed to constrain the magnitude of  $\delta$  to small values.

Norm	Correlation
$L_\infty$	<b>77.51</b>
$L_2$	76.82
$L_1$	75.28

Table 4.5 The influence of the norm constraint on perturbation generation on the average performance of various Semantic Textual Similarity (STS) tasks.

Table 4.5 provides the averaged Spearman’s correlation scores for these norm functions across various Semantic Textual Similarity tasks. It is observed that the  $L_\infty$  norm exhibits superior correlation compared to the other two norms, thereby establishing itself as the norm function of choice for our experimental assessment.

#### 4.4.4 Modulation Factor

Incorporating a modulation factor, denoted as  $0 \leq \lambda \leq 1$ , RobustEmbed provides a mechanism to finely tune the relative importance of each individual perturbation (PGD and FGSM) in the creation of the final perturbation. This modulation factor plays a crucial role in determining the overall efficacy of the generated perturbation.

$\lambda$	Correlation
0	76.36
0.25	76.91
0.5	<b>77.51</b>
0.75	77.04
1	76.48

Table 4.6 The impact of the modulation factor on the average performance of different Semantic Textual Similarity (STS) tasks in generating the final perturbation.

Table 4.6 showcases the performance variation across semantic textual similarity tasks for different values of  $\lambda$ . The experimental results reveal that  $\lambda = 0.5$  yields the highest averaged correlation among the tested magnitudes, underscoring its effectiveness in generating potent perturbations. Consequently, we adopt this particular setting in the configuration of our framework.

<b>Model</b>	<b>STS</b>	<b>Transfer</b>	<b>TextFooler</b>	<b>TextBugger</b>	<b>BERTAttack</b>
PGD	76.37	79.15	50.33	31.05	49.72
FreeLB	81.91	86.03	48.70	27.11	47.83
SMART	82.65	87.34	45.46	26.08	47.39
RobustEmbed	<b>85.79</b>	<b>89.86</b>	<b>37.12</b>	<b>20.43</b>	<b>39.25</b>

Table 4.7 Performance Comparison of Adversarial Training Methods

#### *4.4.5 Comparison of Adversarial Training Techniques*

In order to assess the effectiveness of our framework in comparison to conventional adversarial training methods, we conducted experiments by fine-tuning our pre-trained model using a similar adversarial training strategy as employed during the pre-training phase. Subsequently, we compared the performance of our fine-tuned model against three standard adversarial training techniques, namely PGD, FreeLB Zhu et al. (2020), and SMART Jiang et al. (2020), after the fine-tuning step.

The experimental results, presented in Table 4.7, offer insights into the comparative efficacy of these approaches. It is evident from the results that our framework outperforms the three conventional adversarial training methods across various metrics. Specifically, our approach achieves the highest average accuracy for both STS and transfer tasks, while also exhibiting the lowest average attack success rate under TextFooler, TextBugger, and BERTAttack attacks.

#### *4.4.6 Contrastive Learning Loss*

The initial segment of the overall contrastive loss function (Equation 5.10) focuses on refining the similarity between the input instance  $x$  and its positive counterpart ( $x^{pos}$ ), as well as enhancing the alignment between  $x$  and its adversarial perturbation ( $x^{adv}$ ). While this indirectly narrows the

gap between  $x^{pos}$  and  $x^{adv}$ , our empirical observations indicate that integrating a direct contrastive learning component between  $x^{pos}$  and  $x^{adv}$ , as outlined in the secondary part of Equation 5.10, contributes to notable enhancements in both clean accuracy and robustness.

$\gamma$	STS	Transfer	TextFooler
1/64	76.46	85.93	44.37
1/128	<b>77.51</b>	<b>86.20</b>	<b>37.97</b>
1/256	77.06	85.87	40.32
1/512	75.84	84.66	42.58

Table 4.8 Effect of Regularization Parameter ( $\gamma$ ) on our Framework Performance

The impact of various values of the regularization parameter ( $\gamma$ ) on the ultimate performance of our framework is depicted in Table 4.8. The findings demonstrate that employing  $\gamma = 1/128$  yields the highest average accuracy across STS and transfer tasks, along with the lowest average attack success rate under the TextFooler attack. Consequently, we adopt  $\gamma = 1/128$  for all subsequent experiments, leveraging its optimal performance. Through this approach, our framework achieves enhanced performance and resilience, underscoring the efficacy of our contrastive learning enhancement strategy.

#### 4.5 Evaluation of Sentence Embedding Distribution

We adopted the evaluation methodology proposed by Wang & Isola (2020) to leverage two crucial assessment metrics, denoted as *alignment* and *uniformity*, aimed at gauging the quality of our sentence representations. In the context of positive pairs characterized by the distribution  $p_{pos}$ , *alignment* quantifies the expected distance between the embeddings of paired instances, formulated

as follows:

$$\ell_{\text{align}} \triangleq \mathbb{E}_{(x, x^+) \sim p_{\text{pos}}} \|f(x) - f(x^+)\|^2. \quad (4.7)$$

On the other hand, *uniformity* assesses the extent to which the embeddings are uniformly distributed within the representation space, defined by:

$$\ell_{\text{uniform}} \triangleq \log \mathbb{E}_{x, y \stackrel{i.i.d.}{\sim} p_{\text{data}}} e^{-2\|f(x) - f(y)\|^2}, \quad (4.8)$$

where  $p_{\text{data}}$  denotes the data distribution. The underlying principle behind these metrics posits that embeddings for positive instances should remain closely clustered, while those for random instances should be evenly dispersed across the hypersphere.

Figure 4.5 visualizes the *uniformity* and *alignment* metrics for different sentence embedding models, where lower values indicate better performance. Relative to alternative representations, RobustEmbed attains a comparable level of *uniformity* (-2.293 vs. -2.305) but exhibits superior *alignment* (0.058 vs. 0.073). This underscores the enhanced efficacy of our framework in refining the representation space along two distinct axes.

## 4.6 Summary and Discussion

In this chapter, we introduce RobustEmbed, a novel self-supervised framework tailored for sentence embeddings, aiming to bolster resilience against an array of adversarial attacks while concurrently achieving cutting-edge performance across diverse text representation and natural language processing (NLP) domains.

Standard sentence embedding methods are susceptible to adversarial perturbations, which

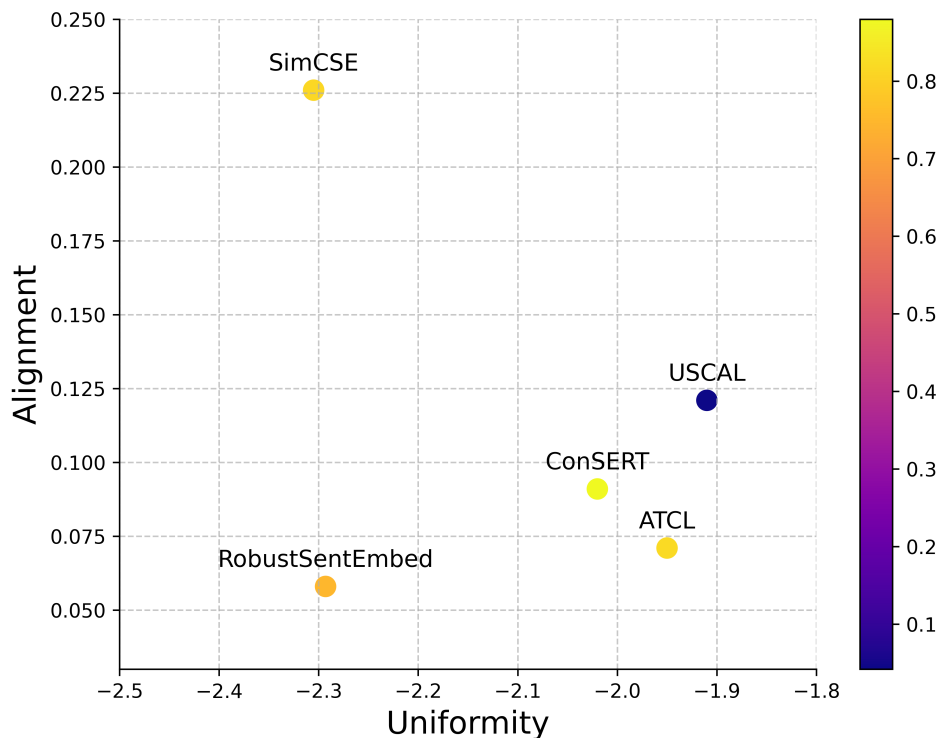


Figure 4.5  $\ell_{\text{align}} - \ell_{\text{uniform}}$  plot of models based on  $\text{BERT}_{\text{base}}$

can compromise their integrity and utility in practical applications. RobustEmbed addresses this vulnerability by capitalizing on high-risk adversarial perturbations within a unique contrastive learning framework.

Our framework’s efficacy is extensively evaluated through a series of experiments encompassing semantic textual similarity assessments and transfer learning tasks. These evaluations underscore the superior performance of RobustEmbed compared to existing methods. Furthermore, empirical evidence solidifies the robustness of RobustEmbed in the face of varied adversarial attack strategies.

## CHAPTER 5

### **RobustSentEmbed: Robust Sentence Embeddings Using Adversarial Self-Supervised Contrastive Learning**

#### **5.1 Introduction**

In recent years, Pre-trained Language Models (PLMs) have emerged as pivotal tools in Natural Language Processing (NLP), exhibiting unparalleled prowess in learning contextual word embeddings Devlin et al. (2019b). The advent of models such as BERT Devlin et al. (2019b) and GPT-3 Brown et al. (2020) has propelled NLP to new heights, revolutionizing various tasks including text classification, sentence representation, and machine translation Yang et al. (2019); He et al. (2021); Ding et al. (2023).

Despite the remarkable achievements of PLMs in capturing intricate linguistic nuances, their application to sentence embeddings presents notable challenges. While PLM-based sentence representations excel in capturing semantic content, they often fall short in two critical aspects: generalization and robustness. In other words, PLM-derived sentence embeddings may struggle to generalize across diverse linguistic contexts and exhibit vulnerability to adversarial perturbations.

Extensive efforts have been directed towards the advancement of universal sentence embeddings leveraging Pre-trained Language Models (PLMs) Reimers & Gurevych (2019); Zhang et al. (2020b); Neelakantan et al. (2022); Wang et al. (2023). These embeddings have showcased adeptness in generalizing across a spectrum of downstream tasks Sun et al. (2019); Gao et al. (2021). However, their susceptibility to adversarial environments and attacks has been underscored in recent studies Nie et al. (2020); Wang et al. (2021). Despite their proficiency, PLM-based embeddings exhibit a

significant drawback in terms of robustness Garg & Ramakrishnan (2020); Wu et al. (2023); Hauser et al. (2023). Their vulnerability lies in the ease with which they can be misled by subtle alterations to the input text, rendering them susceptible to adversarial manipulation.

To overcome these limitations, we introduce a novel approach for acquiring robust sentence embeddings termed RobustSentEmbed. Our methodology is rooted in the generation of subtle adversarial perturbations coupled with the utilization of an efficient contrastive objective Chen et al. (2020). The primary aim is to bolster the adversarial robustness of the sentence embeddings. In essence, our framework orchestrates an iterative interplay between an adversarial perturbation generator and the PLM-based encoder to craft high-risk perturbations within both the token-level and sentence-level embedding domains. Subsequently, RobustSentEmbed integrates a contrastive learning objective alongside a token replacement detection objective, seeking to optimize the similarity between the embedding of the original sentence and the adversarial embedding of a positive pair (the former objective), as well as its modified counterpart (the latter objective).

We have conducted extensive experiments to validate the effectiveness of the RobustSentEmbed framework. Our experimental evaluations encompass a diverse range of tasks, including TextAttack assessments Morris et al. (2020), adversarial Semantic Textual Similarity (STS) tasks, non-adversarial STS tasks Conneau & Kiela (2018), and transfer tasks Conneau & Kiela (2018). Initially, two comprehensive series of experiments were devised to scrutinize the robustness of our sentence embeddings against a multitude of adversarial attacks and tasks. Subsequently, we proceeded with two additional series of experiments to gauge the quality of our embeddings concerning semantic similarity and natural language understanding.

The empirical results reveal significant enhancements achieved by RobustSentEmbed in terms of robustness. For instance, the framework substantially reduces the attack success rate, plummeting from 75.51% to 38.81% against the BERTAttack attack and from 71.86% to 12.80% on adversarial STS tasks. Furthermore, RobustSentEmbed surpasses existing methods in ten out of thirteen tasks, demonstrating notable improvements of 1.59% and 0.23% on STS tasks and NLP transfer tasks, respectively, while achieving comparable results in the remaining three tasks.

### 5.1.1 Contribution

Our primary contributions can be summarized as follows:

- We present RobustSentEmbed, an innovative framework devised to generate sentence embeddings resilient against adversarial attacks. Existing methodologies are susceptible to such adversarial challenges. RobustSentEmbed addresses this vulnerability by generating high-risk perturbations and employing an efficient adversarial objective function.<sup>1</sup>
- We conduct extensive experiments to empirically assess the effectiveness of the RobustSentEmbed framework. The empirical results substantiate the efficacy of our approach, as evidenced by its superior performance across both robustness and generalization benchmarks.

## 5.2 The Proposed Approach

We introduce RobustSentEmbed, a straightforward yet highly effective method devised for generating robust text representations. Given a Pre-trained Language Model (PLM)  $f_{\theta}(\cdot)$  as the encoder and a raw dataset  $\mathcal{D}$ , our approach aims to pre-train  $f_{\theta}(\cdot)$  on  $\mathcal{D}$  to enhance the efficacy of sentence

---

<sup>1</sup>Our implementation is publicly accessible at <https://github.com/GoodFlower123/RobustSentEmbed>

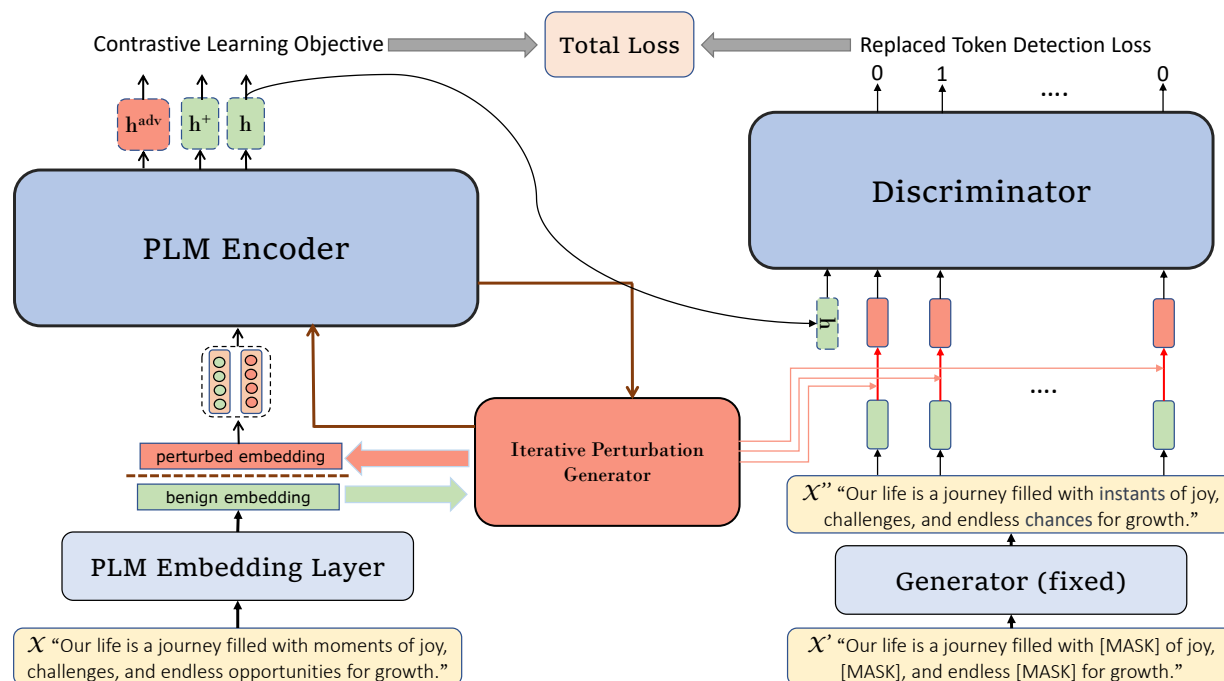


Figure 5.1 The general architecture of the RobustSentEmbed framework.

embeddings across a wide array of Natural Language Processing (NLP) tasks, thereby improving generalization, while also fortifying its resilience against various adversarial attacks, thus enhancing robustness. Figure 5.1 illustrates the conceptual overview of our approach.

The framework encompasses an iterative interplay between the perturbation generator and the  $f_{\theta}(\cdot)$  encoder to generate high-risk adversarial perturbations in both token-level and sentence-level embedding spaces. These perturbations serve as crucial adversarial examples required for adversarial training by both the  $f_{\theta}(\cdot)$  encoder and a PLM-based discriminator. The subsequent sections will delve into the primary components of our approach.

### 5.2.1 *RobustSentEmbed Algorithm*

Algorithm 3 outlines the procedure employed by our framework to generate a norm-bounded perturbation at both the token-level and sentence-level through an iterative process. This process aims to perturb the embeddings in a manner that confounds the  $f_\theta(\cdot)$  encoder by treating the perturbed embeddings as distinct instances. Subsequently, our framework utilizes a contrastive learning objective along with a replaced token detection objective to maximize the similarity between the embedding of the input sentence and the adversarial embedding of its positive pair (the former objective), as well as its edited sentence (the latter objective).

### 5.2.2 *Adversarial Perturbation Generator*

Adversarial perturbation involves the addition of meticulously crafted perturbations into benign data, aimed at deceiving Machine Learning (ML) models Goodfellow et al. (2015b). A highly efficient and widely applicable approach for generating adversarial perturbations is to introduce small noise  $\delta$  within a norm-constrained ball, to maximize the adversarial loss function:

$$\arg \max_{\|\delta\| \leq \epsilon} L(f_\theta(X + \delta), y), \quad (5.1)$$

where  $f_\theta(\cdot)$  represents an ML model parameterized with  $X$  as the sub-word embeddings and  $y$  as the corresponding labels. Various gradient-based algorithms have been devised to tackle this optimization challenge. Our framework extends the token-level perturbation method proposed by Li & Qiu (2021) by augmenting the perturbation with an innovative sentence-level perturbation generator to produce worst-case adversarial examples. The fundamental concept is to train a

---

**Algorithm 3: RobustSentEmbed Algorithm**


---

**Input** : Epoch number  $E$ , PLM Encoder  $\mathbf{f}_\theta$ , dataset of raw sentences  $\mathcal{D}$ , embedding perturbation  $\{\delta, \eta\}$ , dropout masks  $m_1$  and  $m_2$ , perturbation bound  $\epsilon$ , adversarial step sizes  $\{\alpha, \beta, \gamma\}$ , learning rate  $\xi$ , perturbation modulator  $\rho$ , weighting coefficients  $\{\lambda_1, \lambda_2\}$ , adversarial steps  $\{K, T\}$ , contrastive learning objective  $\mathcal{L}_{con, \theta}$  (eq. 5.9), ELECTRA generator  $G(\cdot)$  and discriminator  $D(\cdot)$

**Output** : Robust Sentence Representation

$\mathcal{V} \in \mathbb{R}^{N \times D} \leftarrow \frac{1}{\sqrt{D}} \mathcal{U}(-\sigma, \sigma)$

**for** epoch = 1, ...,  $E$  **do**

**for** minibatch  $B \subset \mathcal{D}$  **do**

$\delta^0 \leftarrow \frac{1}{\sqrt{D}} \mathcal{U}(-\sigma, \sigma)$ ,  $\eta_i^0 \leftarrow \mathcal{V}[w_i]$

$\mathbf{X} = \mathbf{f}_\theta.\text{embedding}(B, m_1)$

$\mathbf{X}^+ = \mathbf{f}_\theta.\text{embedding}(B, m_2)$

**for**  $t = 1, \dots, \max(K, T)$  **do**

$\mathbf{g}_\delta = \nabla_{\delta} \mathcal{L}_{con, \theta}(\mathbf{X} + \delta^{t-1} + \eta^{t-1}, \{\mathbf{X}^+\})$

**if**  $t \leq K$  **then**

$\delta_{pgd}^t = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^{t-1} + \alpha g(\delta^{t-1}) / \|g(\delta^{t-1})\|_P)$

**end**

**if**  $t \leq T$  **then**

$\delta_{fgsm}^t = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^{t-1} + \beta \text{sign}(g(\delta^{t-1})))$

**end**

$\mathbf{g}_{\eta_i} = \nabla_{\eta} \mathcal{L}_{con, \theta}(\mathbf{X} + \delta^{t-1} + \eta^{t-1}, \{\mathbf{X}^+\})$

$\eta_i^t = \eta_i^{t-1} * (\eta_i^{t-1} + \gamma \mathbf{g}_{\eta_i} / \|\mathbf{g}_{\eta_i}\|_P)$

$\eta^t \leftarrow \Pi_{\|\eta\|_P \leq \epsilon}(\eta^t)$

**end**

$\mathcal{V}[w_i] \leftarrow \eta_i^{\max(K, T)}$

$\delta_f = \rho \delta_{pgd}^K + (1 - \rho) \delta_{fgsm}^T$

**for**  $x \in B$  **do**

$x'' = G(\text{MLM}(x))$

$\mathbf{X}^{adv} = \mathbf{X}'' + \eta_i^{\max(K, T)}$

$\mathcal{L}_{RTD, \theta}^x = \sum_{j=1}^{|x|} [-\mathbb{1}(X_j^{adv} = X_j) \log D(\mathbf{X}^{adv}, \mathbf{f}_\theta(x), j)$

$-\mathbb{1}(X_j^{adv} \neq X_j) \log(1 - D(\mathbf{X}^{adv}, \mathbf{f}_\theta(x), j))]$

**end**

$\mathcal{L}_{RTD, \theta} = \sum_{i=1}^{|B|} \mathcal{L}_{RTD}^{x_i}$

$\mathcal{L}_{RobustEmbed, \theta} := \mathcal{L}_{con, \theta}(\mathbf{X}, \{\mathbf{X}^+, \mathbf{X} + \delta_f\})$

$\mathcal{L}_{total} = \mathcal{L}_{RobustEmbed, \theta} + \lambda_1 \cdot \mathcal{L}_{con, \theta}(\mathbf{X} + \delta_f, \{\mathbf{X}^+\}) + \lambda_2 \cdot \mathcal{L}_{RTD, \theta}$

$\theta = \theta - \xi \nabla_{\theta} \mathcal{L}_{total}$

**end**

**end**

---

PLM-based model to withstand a wide array of adversarial attacks, encompassing both word and instance levels. This augmentation aims to enhance the robustness of the model against adversarial perturbations, thus bolstering its reliability in real-world applications.

Recognizing the intricate roles individual tokens play within a sentence, the RobustSentEmbed framework incorporates a scaling index to accommodate larger perturbations for tokens exhibiting greater gradients during the normalization of token-level perturbations. This approach aims to effectively capture the varying importance of tokens within the context of the sentence:

$$n^i = \frac{\|\boldsymbol{\eta}_i^t\|_P}{\max_j \|\boldsymbol{\eta}_j^t\|_P}, \quad (5.2)$$

Here,  $\boldsymbol{\eta}_i^t$  represents the token-level perturbation for word  $i$  at step  $t$  of the gradient ascent, and  $P$  denotes the type of norm constraint utilized. Leveraging an encoder  $f_\theta(\cdot)$  and an input sentence  $x$ , RobustSentEmbed applies standard dropout twice to produce two distinct embeddings, referred to as "positive pairs" and denoted as  $(X, X^+)$ . Subsequently, the updated token-level perturbation is expressed as follows:

$$\boldsymbol{\eta}_i^{t+1} = n^i * (\boldsymbol{\eta}_i^t + \gamma \frac{\mathbf{g}_{\eta_i}}{\|\mathbf{g}_{\eta_i}\|_P}), \quad (5.3)$$

$$\boldsymbol{\eta}^{t+1} \leftarrow \Pi_{\|\boldsymbol{\eta}\|_P \leq \epsilon}(\boldsymbol{\eta}^t), \quad (5.4)$$

Where  $\mathbf{g}_{\eta_i} = \nabla_{\boldsymbol{\eta}} \mathcal{L}_{con, \theta}(\mathbf{X} + \boldsymbol{\delta}^{t-1} + \boldsymbol{\eta}^{t-1}, \{\mathbf{X}^+\})$  denotes the gradient of the contrastive learning loss concerning  $\boldsymbol{\eta}$ . This perturbation generation process is bounded by the  $\ell_\infty$  norm-ball with radius  $\epsilon$ , with  $\Pi$  serving to project the perturbation onto the  $\epsilon$ -ball, ensuring adherence to the norm

constraint.

To generate adversarial perturbations at the sentence-level, RobustSentEmbed employs a combination of the Fast Gradient Sign Method (FGSM) and the Projected Gradient Descent (PGD) technique. This iterative approach, involving T-step FGSM and K-step PGD, systematically reinforces invariance within the embedding space, ultimately enhancing generalization and robustness Goodfellow et al. (2015b); Madry et al. (2018c). The process iteratively updates the perturbation for PGD in iteration  $k + 1$  and FGSM in iteration  $t + 1$  according to the following equations:

$$\delta_{\text{pgd}}^{k+1} = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^k + \alpha g(\delta^k) / \|g(\delta^k)\|_P), \quad (5.5)$$

$$\delta_{\text{fgsm}}^{t+1} = \Pi_{\|\delta\|_P \leq \epsilon}(\delta^t + \beta \text{sign}(g(\delta^t))), \quad (5.6)$$

where  $g(\delta^n) = \nabla_{\delta} \mathcal{L}_{\text{con}, \theta}(\mathbf{X} + \delta^n, \{\mathbf{X}^+\})$  with  $n = t$  or  $k$  represents the gradient of the contrastive learning loss with respect to  $\delta$ . Here,  $\alpha$  and  $\beta$  denote the step sizes for the attacks, while  $\text{sign}(\cdot)$  yields the sign of the vector. The final perturbation is obtained by a practical combination of T-step FGSM and K-step PGD, controlled by a modulation factor  $\rho$ :

$$\delta_{\text{final}} = \rho \delta_{\text{pgd}}^K + (1 - \rho) \delta_{\text{fgsm}}^T, \quad (5.7)$$

where  $0 \leq \rho \leq 1$  modulates the relative importance of each separate perturbation in the formation of the final perturbation. This efficient approach to perturbation generation ensures the creation of robust adversarial examples while maintaining a bounded perturbation size Goodfellow et al. (2015b); Madry et al. (2018c).

### 5.2.3 Robust Contrastive Learning

To fortify text representations against adversarial perturbations, we adopt a straightforward strategy that combines a Replaced Token Detection (RTD) objective with a novel self-supervised contrastive learning objective.

Our approach builds upon an adversarial variant of the RTD task introduced in ELECTRA Clark et al. (2020). In this method, when presented with an input sentence  $x$ , ELECTRA leverages a pre-trained masked language model as the generator  $G$  to reconstruct randomly masked tokens in  $x' = \text{Mask}(x)$ , resulting in the modified sentence  $x'' = G(x')$ . Following this, a discriminator  $D$  is assigned the task of discerning whether token replacements have been made, thereby constituting the RTD task. As depicted in Figure 5.1, our perturbation generator module introduces token-specific perturbations into the embedding of each individual token, rendering it more difficult for discriminator  $D$  to effectively perform the RTD task. The gradient of  $D$  is then propagated back into the encoder  $f$  through  $\mathbf{h} = f_{\theta}(x)$ . This mechanism encourages  $f$  to produce informative vector representations  $\mathbf{h}$ , thereby bolstering its resilience against token-level adversarial attacks.

By integrating both the RTD and contrastive learning objectives, our framework not only enhances the robustness of text representations but also promotes their discriminative capacity in the face of adversarial perturbations. This combined approach enables the model to effectively learn to differentiate between genuine and adversarially altered instances, thus contributing to improved performance across various natural language processing tasks.

In consequence, our methodology incorporates the ensuing adversarial objective for an individual sentence  $x$ :

$$\begin{aligned} \mathcal{L}_{RTD}^x = \sum_{j=1}^{|x|} & [-\mathbb{1}(X_j^{adv} = X_j) \log D(X^{adv}, \mathbf{h}, j) \\ & -\mathbb{1}(X_j^{adv} \neq X_j) \log (1 - D(X^{adv}, \mathbf{h}, j))], \end{aligned} \quad (5.8)$$

where  $X^{adv} = X'' + \boldsymbol{\eta}_i^{\max(K, T)}$  denotes the  $i$ th altered token in  $x$ . The training objective for the batch  $B$  is  $\mathcal{L}_{RTD, \theta} = \sum_{i=1}^{|B|} \mathcal{L}_{RTD}^{x_i}$ .

The adversarial loss function  $\mathcal{L}_{RTD}^x$  is formulated to penalize the discriminator’s misclassification of perturbed tokens. It sums the negative logarithm of the discriminator’s prediction probabilities for each token in the sentence  $x$ . If a token is successfully discriminated as being replaced ( $X_j^{adv} \neq X_j$ ), the corresponding term aims to minimize the probability of  $D$  predicting a replacement (i.e.,  $1 - D(X^{adv}, \mathbf{h}, j)$ ). Conversely, if the discriminator fails to identify a token replacement ( $X_j^{adv} = X_j$ ), the objective seeks to minimize the probability of  $D$  predicting a non-replacement (i.e.,  $D(X^{adv}, \mathbf{h}, j)$ ). This adversarial objective is employed during training to guide the model towards generating robust and discriminative embeddings, thereby enhancing its ability to resist adversarial attacks while preserving the semantic information encoded in the input sentences.

Moreover, we employ self-supervised contrastive learning to obtain compact and informative representations by pulling together semantically similar instances while pushing apart dissimilar ones. Let  $\{(x_i, x_i^+)\}_{i=1}^N$  represent a collection of  $N$  positive pairs, where  $x_i$  and  $x_i^+$  exhibit semantic correlation, and  $(z_i, z_i^+)$  denotes the corresponding embedding vectors for the positive pair  $(x_i, x_i^+)$ . We define the positive set  $z_i^{pos}$  for  $z_i$  as  $z_i^{pos} = \{z_i^+\}$ , and the negative set  $z_i^{neg}$  as  $z_i^{neg} = \{z_i^-\}$ ,

comprising positive pairs from other sentences in the same batch. Subsequently, the contrastive training objective is delineated as follows:

$$\begin{aligned} \mathcal{L}_{con,\theta}(z_i, z_i^{pos}, z_i^{neg}) = & \\ & - \log\left(\frac{\sum_{z_i^{pos}} \exp(\text{sim}(z_i, z_i^+)/\tau)}{\sum_{(z_i^{pos} \cup z_i^{neg})} \exp(\text{sim}(z_i, z_i^{+ \text{ or } -})/\tau)}\right), \end{aligned} \quad (5.9)$$

where  $\tau$  represents a temperature hyperparameter, and  $\text{sim}(u, v) = \frac{u^\top v}{\|u\| \cdot \|v\|}$  signifies the cosine similarity between two representations. Our methodology leverages contrastive learning to maximize the similarity between clean examples and their adversarial perturbations by integrating the adversarial example as an additional element within the positive set:

$$\mathcal{L}_{RobustSentEmbed, \theta} := \mathcal{L}_{con,\theta}(z, \{z^{pos}, z^{adv}\}, \{z^{neg}\}).$$

$$\mathcal{L}_{total} := \mathcal{L}_{RobustSentEmbed, \theta} + \lambda_1 \cdot \mathcal{L}_{con,\theta}(z^{adv}, \{z^{pos}\}, \{z^{neg}\}) + \lambda_2 \cdot \mathcal{L}_{RTD, \theta}, \quad (5.10)$$

Here,  $z^{adv} = z + \delta_{final}$  denotes the adversarial perturbation of the input sample  $x$  in the embedding space, and  $\lambda_1, \lambda_2$  serve as weighting coefficients. The initial part of the total contrastive loss (Eq. 5.10) is crafted to optimize the sentence-level similarity between the input sample  $x$ , its positive pair, and its adversarial perturbation, whereas the subsequent part functions to regularize the loss by fostering the alignment of the adversarial perturbation and  $x$ 's positive pair. The final segment introduces the adversarial Replaced Token Detection (RTD) objective into the total contrastive loss.

### **5.3 Evaluation and Experimental Results**

This section provides an extensive array of experiments conducted to ascertain the efficacy of the proposed framework concerning both robustness and generalization metrics. Robustness evaluation encompasses adversarial attacks and adversarial Semantic Textual Similarity (STS) tasks. On the other hand, generalization evaluation comprises non-adversarial STS and transfer tasks integrated within the SentEval framework Conneau & Kiela (2018). Appendix B furnishes detailed training specifics for our proposed framework. These experiments collectively serve to comprehensively validate the capabilities and performance of the proposed framework across a spectrum of tasks and scenarios, shedding light on its robustness and generalization prowess.

#### ***5.3.1 Evaluation Against Adversarial Attacks***

The robustness of our framework against a spectrum of adversarial attacks is rigorously evaluated, juxtaposing its performance with two contemporary sentence embedding models: SimSCE Gao et al. (2021) and USCAL Miao et al. (2021). Fine-tuning of the BERT-based Pre-trained Language Model (PLM) is conducted across seven distinct text classification and natural language inference tasks, namely MRPC Dolan & Brockett (2005), YELP Zhang et al. (2015), IMDb Maas et al. (2011), Movie Reviews (MR) Pang & Lee (2005b), SST2 Socher et al. (2013a), Stanford NLI (SNLI) Bowman et al. (2015), and Multi-NLI (MNLI) Williams et al. (2018). The assessment of our fine-tuned model’s robustness entails the exploration of five prevalent adversarial attacks: TextBugger Li et al. (2019), PWWS Ren et al. (2019), TextFooler Jin et al. (2020), BAE Garg & Ramakrishnan (2020), and BERTAttack Li et al. (2020c). Appendix C furnishes further elucidation

Adversarial Attack	Model	IMDB	MR	SST2	YELP	MRPC	SNLI	MNLI-Mismatched	Avg.
TextFooler	SimCSE-BERT <sub>base</sub>	75.32	65.53	71.49	79.67	80.07	72.65	68.54	72.61
	USCAL-BERT <sub>base</sub>	61.94	48.71	55.38	62.30	60.18	54.82	53.74	56.72
	RobustSentEmbed-BERT <sub>base</sub>	<b>40.02</b>	<b>31.39</b>	<b>35.83</b>	<b>43.78</b>	<b>37.54</b>	<b>36.99</b>	<b>34.15</b>	<b>37.10</b>
TextBugger	SimCSE-BERT <sub>base</sub>	52.21	42.04	49.67	56.19	56.73	45.39	40.16	48.91
	USCAL-BERT <sub>base</sub>	39.16	27.37	31.90	41.25	37.86	30.79	25.45	33.40
	RobustSentEmbed-BERT <sub>base</sub>	<b>23.16</b>	<b>17.49</b>	<b>19.62</b>	<b>27.93</b>	<b>19.37</b>	<b>18.05</b>	<b>15.51</b>	<b>20.16</b>
PWWS	SimCSE-BERT <sub>base</sub>	64.41	55.73	60.48	67.54	68.15	56.09	52.58	60.71
	USCAL-BERT <sub>base</sub>	51.95	40.67	45.29	52.30	46.86	50.92	39.37	46.77
	RobustSentEmbed-BERT <sub>base</sub>	<b>32.94</b>	<b>28.05</b>	<b>29.28</b>	<b>29.14</b>	<b>24.72</b>	<b>26.28</b>	<b>27.90</b>	<b>28.33</b>
BAE	SimCSE-BERT <sub>base</sub>	73.50	61.83	68.27	75.15	77.84	69.06	65.43	70.15
	USCAL-BERT <sub>base</sub>	58.57	46.19	51.72	59.49	58.38	50.90	51.16	53.77
	RobustSentEmbed-BERT <sub>base</sub>	<b>37.16</b>	<b>29.12</b>	<b>31.43</b>	<b>40.96</b>	<b>35.53</b>	<b>33.87</b>	<b>31.85</b>	<b>34.27</b>
BERTAttack	SimCSE-BERT <sub>base</sub>	78.42	66.94	73.59	80.87	82.16	74.35	72.22	75.51
	USCAL-BERT <sub>base</sub>	63.23	51.08	57.73	63.96	63.05	55.41	55.86	58.62
	RobustSentEmbed-BERT <sub>base</sub>	<b>41.51</b>	<b>34.19</b>	<b>38.16</b>	<b>44.96</b>	<b>38.26</b>	<b>38.60</b>	<b>35.98</b>	<b>38.81</b>

Table 5.1 Attack success rates (lower is better) of various adversarial attacks applied to three sentence embeddings (SimCSE, USCAL, and RobustSentEmbed) across five text classification and two natural language inference tasks. RobustSentEmbed reduces the attack success rate to less than half across all attacks.

on these attack methodologies. To uphold statistical rigor, each experiment is iterated five times, with every iteration comprising 1000 samples subjected to adversarial manipulation.

In Table 5.1, the average attack success rates of five distinct adversarial attacks are presented concerning three different sentence embeddings. It is noteworthy that our embedding framework consistently surpasses the other two embedding methods, exhibiting significantly reduced attack success rates (less than half) across all text classification and natural language inference tasks. Consequently, RobustSentEmbed attains the lowest average attack success rate against all adversarial

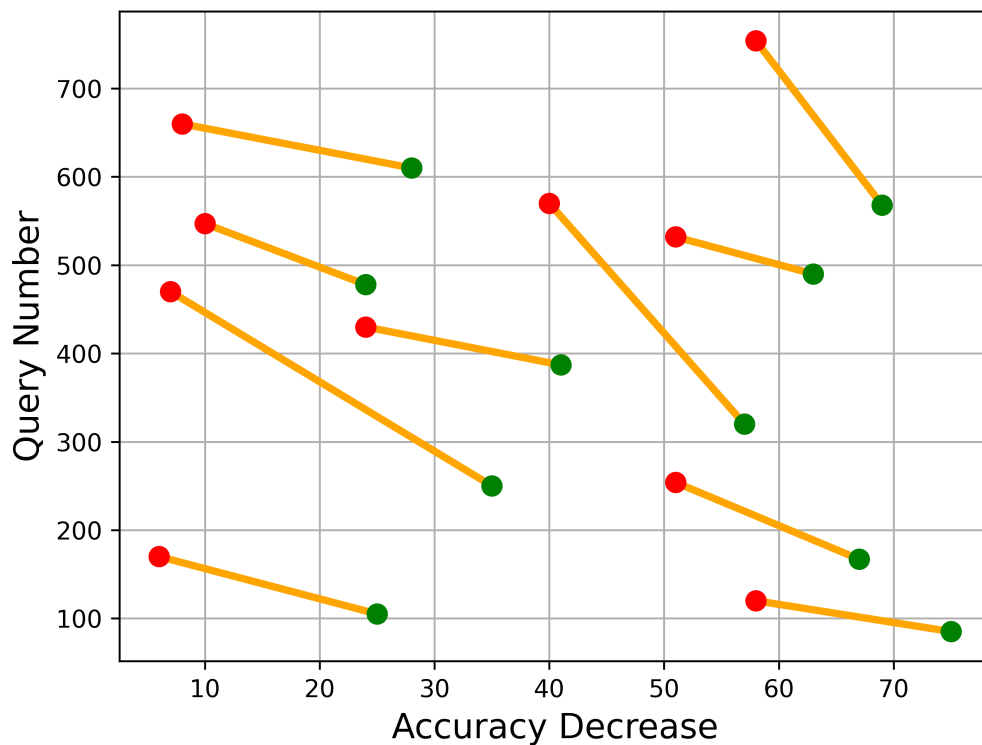


Figure 5.2 Average number of queries and the resulting accuracy reduction for two fine-tuned embeddings.

attack techniques. These findings robustly support the efficacy of our embedding framework while simultaneously illuminating the vulnerabilities present in other state-of-the-art sentence embeddings when subjected to diverse adversarial attacks.

Figure 5.2 illustrates the outcomes of 1000 attacks conducted on two fine-tuned sentence embeddings, evaluating the average number of queries required and the consequent accuracy reduction. The attacks targeting the RobustSentEmbed framework are denoted by green data points, while those aimed at the USCAL approach Miao et al. (2021) are represented by red points. Each pair of connected points corresponds to a specific attack. Ideally, a robust sentence embedding

should be situated in the top-left region of the graph, indicating that it necessitates a higher number of queries for an attack to deceive the model while causing minimal performance degradation. Across all adversarial attacks, RobustSentEmbed consistently demonstrates greater stability compared to the USCAL method. In other words, RobustSentEmbed requires a larger number of queries, resulting in a lower accuracy reduction (i.e., better performance) in comparison to USCAL. This graph visually reinforces the robustness of the RobustSentEmbed framework against adversarial attacks.

### ***5.3.2 Assessment of Robustness***

We propose a novel task termed Adversarial Semantic Textual Similarity (AdvSTS) to evaluate the robustness of sentence embeddings. AdvSTS utilizes an effective adversarial technique, such as TextFooler, to manipulate a given input sentence pair from a non-adversarial Semantic Textual Similarity (STS) task in a manner that induces the target model to produce a regression score significantly divergent from the ground truth label. Consequently, we construct an adversarial STS dataset by transforming all benign instances from the original (i.e., non-adversarial) dataset into adversarial examples.

Table 5.2 illustrates the success rates of five adversarial attacks on three different sentence embeddings, including our proposed framework. These assessments are carried out for two Adversarial Semantic Textual Similarity (AdvSTS) tasks, namely AdvSTS-B (derived from STS Benchmark Cer et al. (2017)) and AdvSICK-R (derived from SICK-Relatedness Marelli et al. (2014)). Remarkably, our framework consistently outperforms the other two sentence embedding methods, demonstrating significantly lower attack success rates across both AdvSTS tasks and all utilized

Adversarial Attack	Model	AdvSTS-B	AdvSICK-R	Avg.
TextFooler	SimCSE-BERT <sub>base</sub>	21.07	24.17	22.62
	USCAL-BERT <sub>base</sub>	16.52	18.71	17.62
	RobustSentEmbed-BERT <sub>base</sub>	<b>7.18</b>	<b>8.53</b>	<b>7.86</b>
TextBugger	SimCSE-BERT <sub>base</sub>	27.49	28.34	27.91
	USCAL-BERT <sub>base</sub>	21.52	24.88	23.20
	RobustSentEmbed-BERT <sub>base</sub>	<b>11.32</b>	<b>12.94</b>	<b>12.13</b>
PWWS	SimCSE-BERT <sub>base</sub>	24.15	26.82	25.49
	USCAL-BERT <sub>base</sub>	21.28	23.65	22.47
	RobustSentEmbed-BERT <sub>base</sub>	<b>12.68</b>	<b>13.90</b>	<b>13.29</b>
BAE	SimCSE-BERT <sub>base</sub>	26.92	28.81	27.86
	USCAL-BERT <sub>base</sub>	22.92	25.48	24.20
	RobustSentEmbed-BERT <sub>base</sub>	<b>10.53</b>	<b>12.09</b>	<b>11.31</b>
BERTAttack	SimCSE-BERT <sub>base</sub>	31.60	32.85	32.23
	USCAL-BERT <sub>base</sub>	26.02	28.51	27.26
	RobustSentEmbed-BERT <sub>base</sub>	<b>12.58</b>	<b>13.02</b>	<b>12.80</b>

Table 5.2 Attack success rates (lower is better) of five adversarial attack techniques applied to three sentence embeddings (SimCSE, USCAL, and RobustSentEmbed) across two Adversarial Semantic Textual Similarity (AdvSTS) tasks (i.e. AdvSTS-B and AdvSICK-R). RobustSentEmbed reduces the attack success rate to less than half across all attacks.

adversarial attacks. These findings provide additional empirical evidence supporting the robustness of the RobustSentEmbed approach in generating resilient text representations.

### *5.3.3 Evaluation of Semantic Textual Similarity (STS)*

In this section, we embark on an exhaustive examination of our framework’s performance across seven Semantic Textual Similarity (STS) tasks, spanning datasets from 2012 to 2016 Agirre et al. (2012, 2013, 2014, 2015, 2016), STS Benchmark, and SICK-Relatedness. To ascertain the efficacy of our framework, we undertake a meticulous comparative analysis against a diverse array of unsupervised sentence embedding methodologies. These encompass: 1) fundamental techniques like GloVe Pennington et al. (2014) and average BERT embeddings; 2) post-processing strategies such as BERT-flow Li et al. (2020a) and BERT-whitening Su et al. (2021); and 3) cutting-edge approaches like SimCSE Gao et al. (2021) and USCAL Miao et al. (2021). To validate the integrity of our comparison, we meticulously replicate the results documented by the SimCSE, ConSERT, and USCAL frameworks.

The empirical findings, as depicted in Table 5.3, consistently affirm the superior efficacy of our RobustSentEmbed framework compared to a myriad of alternative sentence embeddings. Across a spectrum of STS tasks, our framework consistently attains the highest average Spearman’s correlation score when juxtaposed with state-of-the-art methodologies. Specifically, leveraging the BERT encoder, our framework outshines the second-best embedding method, USCAL, by a notable margin of 1.59%. Furthermore, RobustSentEmbed emerges as the top performer in the majority of individual STS tasks, exhibiting superior performance in 6 out of 7 tasks when pitted against other embedding methodologies. When employing the RoBERTa encoder, RobustSentEmbed outperforms prevailing embeddings in five out of seven STS tasks and achieves the pinnacle of the average Spearman’s correlation score.

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
GloVe embeddings (avg.) <sup>♥</sup>	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
BERT <sub>base</sub> (first-last avg.) <sup>♣</sup>	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
BERT <sub>base</sub> -flow <sup>♣</sup>	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
BERT <sub>base</sub> -whitening <sup>♣</sup>	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
ConSERT-BERT <sub>base</sub>	64.56	78.55	69.16	79.74	76.00	73.91	67.35	72.75
ATCL-BERT <sub>base</sub>	67.14	80.86	71.73	79.50	76.72	79.31	70.49	75.11
SimCSE-BERT <sub>base</sub>	68.66	<b>81.73</b>	72.04	80.53	78.09	79.94	71.42	76.06
USCAL-BERT <sub>base</sub>	69.30	80.85	72.19	81.04	77.52	81.28	71.98	76.31
RobustSentEmbed-BERT <sub>base</sub>	<b>71.90</b>	81.12	<b>74.92</b>	<b>82.38</b>	<b>79.43</b>	<b>82.02</b>	<b>73.53</b>	<b>77.90</b>
RoBERTa <sub>base</sub> -whitening	46.99	63.24	57.23	71.36	68.99	61.36	62.91	61.73
ConSERT-RoBERTa <sub>base</sub>	66.90	79.31	70.33	80.57	77.95	81.42	68.16	74.95
SimCSE-RoBERTa <sub>base</sub>	68.75	80.81	71.19	81.79	79.35	82.62	69.56	76.30
USCAL-RoBERTa <sub>base</sub>	69.28	81.15	72.81	81.47	<b>80.55</b>	83.34	70.94	77.08
RobustSentEmbed-RoBERTa <sub>base</sub>	<b>70.03</b>	<b>82.15</b>	<b>73.27</b>	<b>82.48</b>	79.61	<b>83.82</b>	<b>71.66</b>	<b>77.57</b>
USCAL-RoBERTa <sub>large</sub>	68.70	<b>81.84</b>	74.26	82.52	<b>80.01</b>	83.14	76.30	78.11
RobustSentEmbed-RoBERTa <sub>large</sub>	<b>69.30</b>	81.76	<b>75.14</b>	<b>83.57</b>	79.74	<b>83.90</b>	<b>77.08</b>	<b>78.64</b>

Table 5.3 Semantic Similarity performance on STS tasks (Spearman’s correlation, “all” setting) for sentence embedding models. We emphasize the top-performing numbers among models that share the same pre-trained encoder. <sup>♥</sup>: results from Reimers & Gurevych (2019); <sup>♣</sup>: results from Gao et al. (2021); All remaining results have been reproduced and reevaluated by our team. RobustSentEmbed produces the most effective sentence representations that are more general in addition to robust representation (section 5.3.2 and 5.3.1).

### 5.3.4 Transfer Learning Evaluation

Transfer learning serves as a pivotal technique for evaluating the versatility and effectiveness of our RobustSentEmbed framework across a broad spectrum of text classification tasks, encompassing domains like sentiment analysis and paraphrase identification. Our comprehensive evaluation

Model	MR	CR	SUBJ	MPQA	SST2	MRPC	Avg.
GloVe embeddings (avg.) ♣	77.25	78.30	91.17	87.85	80.18	72.87	81.27
Skip-thought ♥	76.50	80.10	93.60	87.10	82.00	73.00	82.05
BERT-[CLS] embedding ♣	78.68	84.85	94.21	88.23	84.13	71.13	83.54
ConSERT-BERT <sub>base</sub>	79.52	87.05	94.32	88.47	85.46	72.54	84.56
SimCSE-BERT <sub>base</sub>	81.29	86.94	94.72	89.49	<b>86.70</b>	75.13	85.71
USCAL-BERT <sub>base</sub>	81.54	<b>87.12</b>	95.24	89.34	85.71	75.84	85.80
RobustSentEmbed-BERT <sub>base</sub>	<b>82.06</b>	86.28	<b>95.42</b>	<b>89.61</b>	86.12	<b>76.69</b>	<b>86.03</b>
SimCSE-RoBERTa <sub>base</sub>	81.15	87.15	92.38	86.79	<b>86.24</b>	75.49	84.87
USCAL-RoBERTa <sub>base</sub>	<b>82.15</b>	87.22	92.76	87.74	84.39	76.20	85.08
RobustSentEmbed-RoBERTa <sub>base</sub>	81.57	<b>87.66</b>	<b>93.51</b>	<b>87.94</b>	85.04	<b>76.89</b>	<b>85.44</b>
USCAL-RoBERTa <sub>large</sub>	<b>82.84</b>	87.97	93.12	88.48	<b>86.28</b>	76.41	85.85
RobustSentEmbed-RoBERTa <sub>large</sub>	82.56	<b>88.51</b>	<b>93.84</b>	<b>88.65</b>	86.18	<b>77.01</b>	<b>86.13</b>

Table 5.4 Results of transfer tasks for different sentence embedding models. ♣: results from Reimers & Gurevych (2019); ♥: results from Zhang et al. (2020b); We emphasize the top-performing numbers among models that share the same pre-trained encoder. All remaining results have been reproduced and reevaluated by our team. RobustSentEmbed outperforms all other methods, regardless of the pre-trained language model (BERT<sub>base</sub>, RoBERTa<sub>base</sub>, or RoBERTa<sub>large</sub>).

spans six transfer tasks, including CR Hu & Liu (2004), SUBJ Pang & Lee (2004), MPQA Wiebe et al. (2005), SST2 Socher et al. (2013a), and MRPC Dolan & Brockett (2005). By training logistic regression classifiers atop the fixed sentence embeddings, we ensure the robustness and accuracy of our assessments. To uphold the credibility of our findings, we meticulously replicate the experimental setups of benchmark frameworks such as SimCSE, ConSERT, and USCAL.

The outcomes, meticulously documented in Table 5.4, unequivocally attest to the unparalleled performance of our framework in terms of average accuracy when juxtaposed with competing

sentence embeddings. Notably, leveraging the BERT encoder, our framework surpasses the second-best embedding methodology by an appreciable margin of 0.23%. Furthermore, RobustSentEmbed emerges as the top performer in four out of six text classification tasks. A similar trend is observed for the RoBERTa encoder.

Analyzing the cumulative results presented in Tables 5.3 and 5.4, it becomes evident that RobustSentEmbed not only facilitates robust representation (as evidenced in Section 5.3.1 and Section 5.3.2), but also excels in providing general sentence representations across diverse text classification tasks. This corroborates the robustness and versatility of our proposed framework in real-world applications.

In conclusion, the exhaustive array of experiments, delineated by the results presented in Tables 5.1, 5.2, 5.3, and 5.4, alongside the graphical depiction in Figure 5.2, unequivocally validate the remarkable efficacy of RobustSentEmbed in the domain of text representation. Moreover, these experiments demonstrate its robustness against adversarial attacks and tasks. These empirical observations serve to accentuate the framework's unparalleled resilience and its capacity for generalization, thereby positioning it as a versatile tool for the generation of high-fidelity sentence embeddings. The findings from these evaluations not only bolster confidence in the efficacy of RobustSentEmbed but also underscore its potential to significantly contribute to the advancement of natural language processing and related fields in computer science.

## 5.4 Evaluation of Sentence Embedding Distribution

In our analysis, we assessed the quality of our sentence embeddings using two pivotal metrics, namely *alignment* and *uniformity* Wang & Isola (2020). These metrics are essential for gauging the effectiveness and robustness of the representations generated. The concept of *alignment* revolves around computing the anticipated distance between embeddings of paired instances, which is represented by the distribution of positive pairs  $p_{pos}$ . Mathematically, the alignment metric, denoted as  $\ell_{align}$ , is defined as the expected squared Euclidean distance between the embeddings of paired instances, as illustrated by the equation:

$$\ell_{align} \triangleq \mathbb{E}_{(x, x^+) \sim p_{pos}} \|f(x) - f(x^+)\|^2 \quad (5.11)$$

Furthermore, *uniformity* serves as a crucial indicator of how uniformly the embeddings are distributed across the representation space. This metric assesses the diversity and spread of embeddings across the data distribution, which is represented by  $p_{data}$ . The *uniformity* metric, denoted as  $\ell_{uniform}$ , is calculated using the logarithm of the expected value of pairwise distances between embeddings of independently and identically distributed (i.i.d.) instances sampled from the data distribution, as depicted by the equation:

$$\ell_{uniform} \triangleq \log \mathbb{E}_{x, y \stackrel{i.i.d.}{\sim} p_{data}} e^{-2\|f(x) - f(y)\|^2} \quad (5.12)$$

These metrics provide a comprehensive understanding of the distribution and quality of sentence embeddings, offering insights into their alignment, uniformity, and overall effectiveness in representing textual information in the computer science domain.

Figure 5.3 provides a visual depiction of the metrics *uniformity* and *alignment* across various

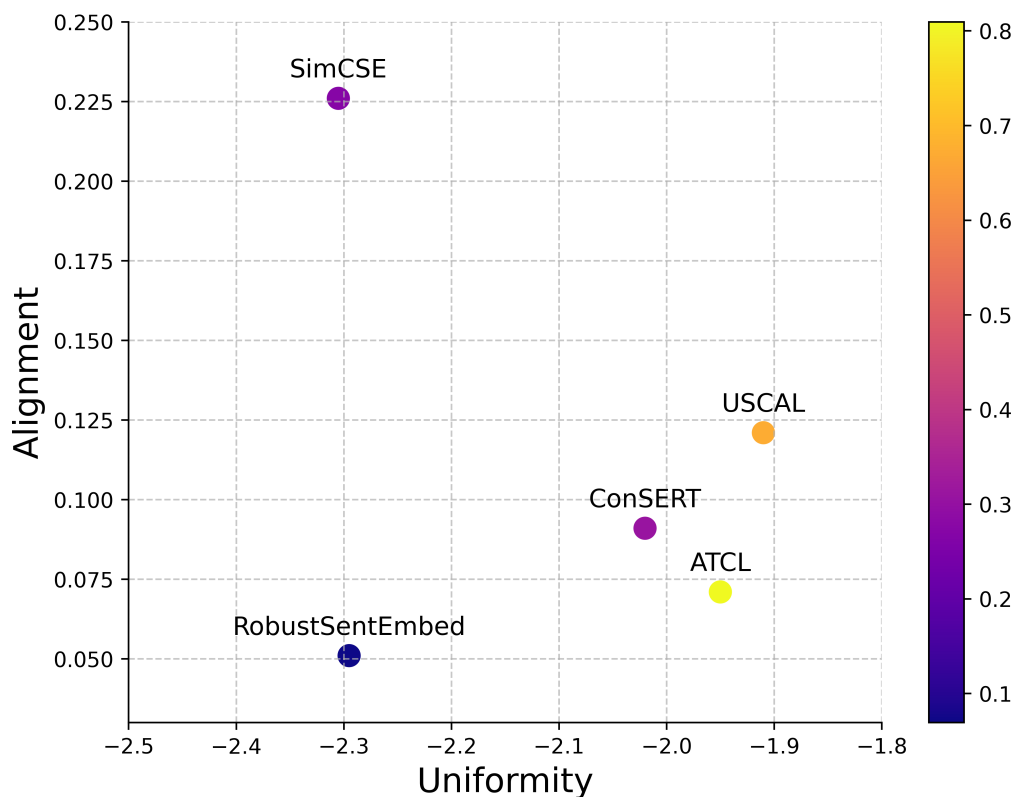


Figure 5.3  $\ell_{\text{align}} - \ell_{\text{uniform}}$  plot of models based on  $\text{BERT}_{\text{base}}$ . Lower uniformity and alignment is better.

sentence embedding models, crucial factors in evaluating the quality and effectiveness of these representations within the realm of computational linguistics. Lower values of these metrics are indicative of better performance in representing textual semantics. Upon comparative analysis, it is evident that RobustSentEmbed attains a comparable level of *uniformity* (-2.295 compared to -2.305) when juxtaposed against alternative embedding methodologies. However, what sets RobustSentEmbed apart is its superior *alignment* score (0.051 versus 0.073). This discrepancy underscores the efficacy of our framework in optimizing the representation space along distinct axes, thereby enhancing the overall semantic coherence and interpretability of the generated embeddings. Such insights not only validate the robustness and efficiency of RobustSentEmbed but also offer

valuable guidance for practitioners and researchers in selecting optimal embedding models for various natural language processing tasks within the computer science domain.

## 5.5 Ablation Studies

In this section, we delve into a comprehensive analysis aimed at understanding the effects of five critical hyperparameters on the overall performance of the RobustSentEmbed framework. Leveraging BERT<sub>base</sub> as our encoder of choice, we meticulously evaluate these hyperparameters using the development set of STS tasks, a widely recognized benchmark in the field of natural language processing. Through this meticulous examination, we aim to uncover insights into the intricate relationship between hyperparameter configuration and framework performance, providing valuable guidance for optimizing the RobustSentEmbed framework in practical applications.

### 5.5.1 Optimizing Step Sizes for Perturbation Generation

Within the RobustSentEmbed framework, the generation of perturbations through PGD and FGSM processes relies on the adjustment of two critical step sizes, denoted as  $\alpha$  and  $\beta$  respectively. Illustrated in Figure 5.4, the collaborative influence of modifying the ranges associated with these step sizes plays a pivotal role in generating perturbations with heightened adversarial potency, a fundamental requirement for achieving an effective contrastive learning objective. The results underscore the significance of fine-tuning  $\beta$  towards a lower bound while simultaneously setting  $\alpha$  to an upper bound, leading to more pronounced enhancements in performance.

Specifically, the experiments reveal that allocating ranges of [1e-4, 1e-6] for  $\alpha$  and [1e-3, 1e-4] for  $\beta$  yields substantial improvements in perturbation generation. Consequently, for our

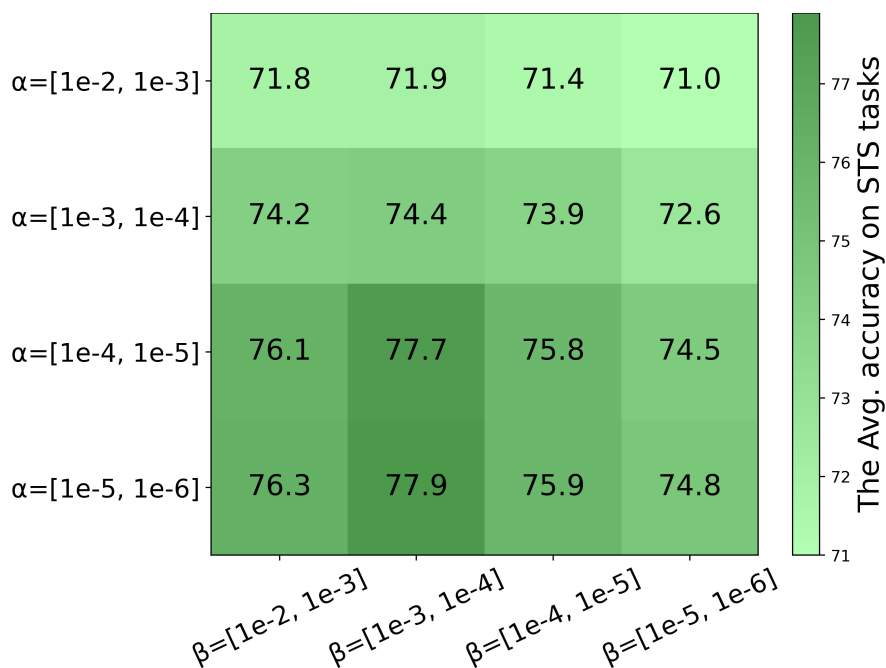


Figure 5.4 The impact of step sizes in perturbation generation on the average performance of STS tasks.

experimental setup, we adopt the configuration  $\alpha = 1e-5$  and  $\beta = 1e-3$ , as it demonstrates superior performance compared to alternative configurations.

### 5.5.2 Impact of Step Numbers in Perturbation Generation

The method employed in RobustSentEmbed utilizes both T-step FGSM and K-step PGD iterations to obtain adversarial perturbations with a high potential for risk, which are crucial for fulfilling the contrastive learning objective. To simplify the analysis of perturbation generation, we set K equal to T. The effect of varying step numbers (denoted as N, where N equals either K or T) on the effectiveness of the process is depicted in Figure 5.5.

The results indicate a gradual enhancement in effectiveness as N increases from 1 to 12. However, beyond the threshold of N=12, the observed improvement becomes negligible. Moreover,

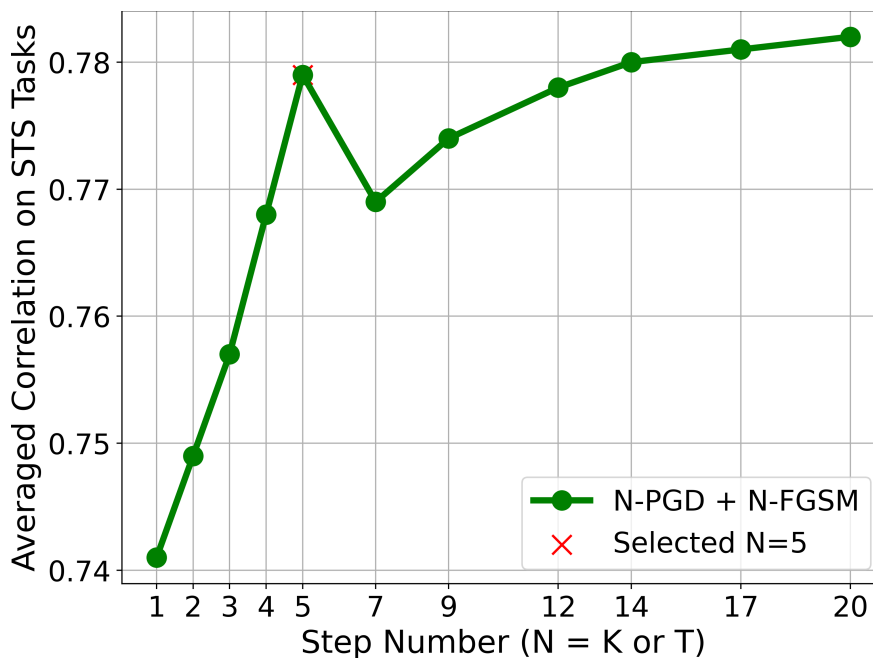


Figure 5.5 The impact of the step number (represented by  $N = K$  or  $T$ ) in the  $T$ -step FGSM and  $K$ -step PGD methods on the averaged correlation of the STS tasks.

it's noteworthy that higher values of  $N$  lead to increased computational overhead and may result in unequal allocation of computational resources. As a consequence, a careful balance must be struck between computational efficiency and effectiveness. Based on our experimentation, we have chosen to set  $N=5$  in order to strike an appropriate balance between computational cost and perturbation effectiveness.

### 5.5.3 Magnitude Regulation for Imperceptibility

In order to maintain imperceptibility in the generated adversarial instances, RobustSentEmbed employs a strategy to control the magnitude of perturbation vectors ( $\delta$  or  $\eta$ ). This regulation is crucial to ensure that the perturbations remain subtle and indiscernible to human perception. The method achieves this control by employing three widely used norm functions:  $L_1$ ,  $L_2$ , and  $L_\infty$ .

These norms serve to constrain the magnitude of the perturbation vectors, thereby limiting their impact on the input data.

<b>Norm</b>	<b>Correlation</b>
$L_\infty$	<b>77.90</b>
$L_2$	76.84
$L_1$	76.52

Table 5.5 The impact of the norm constraint on perturbation generation on the average performance of various STS tasks.

Table 5.5 provides an overview of the averaged Spearman’s correlation values for these norm functions across various Semantic Textual Similarity tasks. It is observed that the  $L_\infty$  norm demonstrates a notably higher correlation compared to the  $L_1$  and  $L_2$  norms. This superior correlation suggests that the  $L_\infty$  norm is more effective in capturing the imperceptibility of the perturbations in the context of our experiments. The selection of the  $L_\infty$  norm as the primary norm function for our experimental evaluation is thus justified based on its superior correlation performance.

#### ***5.5.4 Enhancing Contrastive Learning for Improved Performance***

The total loss function (referenced as Equation 5.10) consists of multiple components aimed at optimizing various aspects of the learning process. The initial segment of the loss function focuses on enhancing the similarity between the input instance  $x$  and its positive counterpart ( $x^{pos}$ ), along with the similarity between  $x$  and its corresponding adversarial perturbation ( $x^{adv}$ ). While this indirectly fosters a convergence between  $x^{pos}$  and  $x^{adv}$ , our research suggests that the incorporation of direct contrastive learning between these entities (as depicted in the second part of Equation 5.10), achieved through regularization within the primary objective function, significantly contributes to

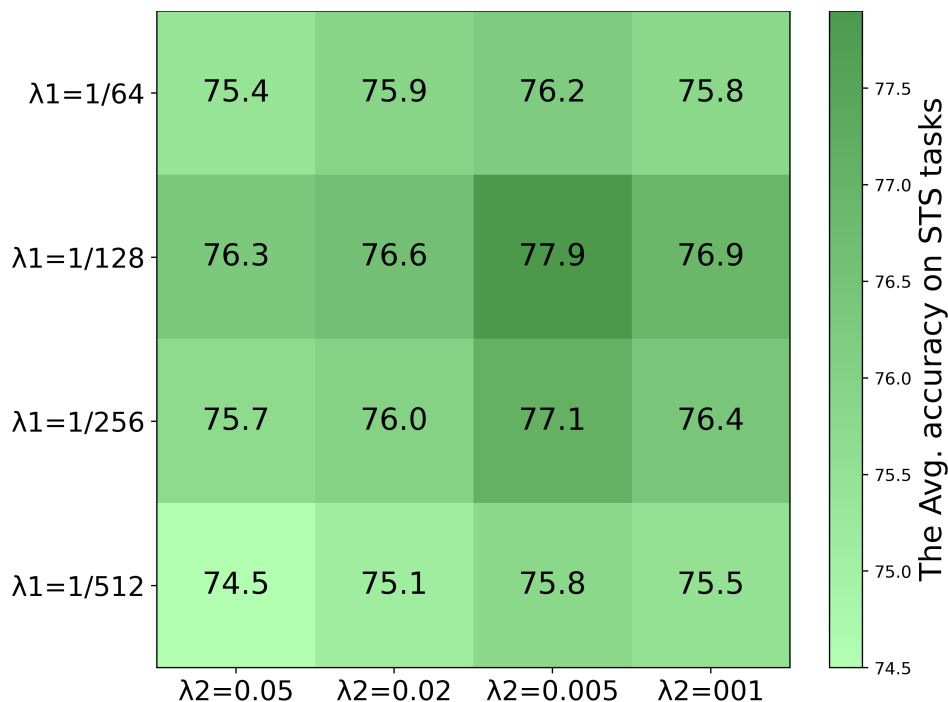


Figure 5.6 The impact of weighting coefficients in the total loss function on the average performance of STS tasks.

improved performance metrics such as clean accuracy and robustness.

Moreover, to introduce additional challenges for adversarial training convergence, the third segment of the total loss function integrates an objective related to adversarial replaced token detection. This inclusion complicates the training process, thereby enhancing the robustness of the model against adversarial attacks.

The impact of varying weighting coefficients ( $\lambda_1$  and  $\lambda_2$ ) on the overall performance of our framework is illustrated in Figure 5.6. It is evident from the illustration that specific values, such as  $\lambda_1 = 1/128$  and  $\lambda_2 = 0.005$ , yield optimal results in terms of average accuracy across semantic textual similarity tasks. Consequently, we adopt these values,  $\lambda_1 = 1/128$  and  $\lambda_2 = 0.005$ , as the standard configuration for all our experiments.

### 5.5.5 Adjusting Perturbation Importance with Modulation Factor

Incorporated within RobustSentEmbed is a modulation factor, denoted as  $0 \leq \rho \leq 1$ , designed to fine-tune the relative significance of individual perturbations (PGD and FGSM) during the creation of sentence-level perturbations. Table 5.6 provides a comprehensive analysis of the effectiveness of varying values of this modulation factor across semantic textual similarity tasks.

$\rho$	Correlation
0	76.06
0.25	76.85
0.5	<b>77.90</b>
0.75	77.34
1	76.34

Table 5.6 The impact of the modulation factor on the average performance of different Semantic Textual Similarity (STS) tasks in generating the final perturbation.

The results elucidate that setting  $\rho = 0.5$  leads to the highest averaged correlation among the different magnitudes investigated, indicating its capacity to generate perturbations of increased potency. As a result, we opt to adopt this particular configuration for our framework setup.

## 5.6 Summary and Discussion

In this study, we have introduced RobustSentEmbed, a pioneering self-supervised framework tailored to fortify the resilience of sentence embeddings against adversarial attacks, while simultaneously achieving remarkable performance levels across a spectrum of text representation and natural language processing (NLP) tasks. The vulnerability of existing sentence embedding methodologies to adversarial attacks necessitated the development of RobustSentEmbed, which addresses this

critical issue through the generation of high-risk perturbations at both token and sentence levels.

The incorporation of these perturbations into innovative contrastive and difference prediction objectives represents a significant advancement in the field, as it empowers RobustSentEmbed to effectively counteract adversarial vulnerabilities while enhancing the quality of the resulting sentence embeddings.

The validation of the RobustSentEmbed framework through extensive experimentation on semantic textual similarity tasks and transfer learning scenarios has provided compelling evidence of its robustness against adversarial attacks, as well as its proficiency in tasks requiring nuanced semantic similarity assessments.

## CHAPTER 6

### Conclusion and Future Work

#### 6.1 Conclusion

This dissertation delves into the realm of trustworthy AI within the domain of Natural Language Processing (NLP), exploring the intricacies of developing efficient adversarial attacks and robust Pre-trained Language Model (PLM)-based text representations.

Initially, we introduce SSCAE, a pragmatic Adversarial Example (AE) generator designed to craft contextually sensitive AEs while preserving crucial linguistic attributes encompassing semantics, syntax, and grammar. SSCAE implements practical refinement methodologies to uphold the linguistic integrity of the final perturbations. Notably, we introduce two innovative techniques: (1) a dynamic thresholding mechanism, termed as the dynamic threshold, to capture more potent perturbations, and (2) a word permutation-substitution technique known as local greedy search, to produce high-quality adversarial examples. Comparative evaluations against three state-of-the-art adversarial attack methodologies reveal SSCAE’s superior performance, showcasing significant improvements across a spectrum of popular and challenging text classification and text entailment tasks.

Furthermore, we present RobustEmbed, a self-supervised framework for sentence embeddings, which substantially enhances resilience against diverse adversarial attacks while achieving cutting-edge performance across a broad spectrum of text representation and NLP tasks. Recognizing the vulnerability of current sentence embeddings to adversarial attacks, RobustEmbed bridges this gap by leveraging high-risk adversarial perturbations within a novel contrastive objective

framework. Through extensive experimentation on semantic textual similarity and transfer learning tasks, we validate the efficacy of our framework. Empirical findings provide compelling evidence of RobustEmbed’s robustness against a diverse array of adversarial attacks.

Lastly, in a bid to address token-level adversarial defense, we introduce RobustSentEmbed, a novel self-supervised framework for sentence embeddings, integrating perturbations at both token and sentence levels. Distinguished from RobustEmbed, this new framework introduces two primary innovations: (a) the adoption of a pragmatic token-level perturbation generator based on an adversarial contrastive learning objective, and (b) the incorporation of a novel adversarial token-detection objective to generate more robust sentence embeddings. Experimental results validate the efficacy of our approach, demonstrating its heightened resilience against various adversarial attacks and adversarial tasks within the NLP domain.

In conclusion, the research-based exploration conducted in this dissertation, aimed at refining and optimizing adversarial attack methodologies and robust text representations, holds the potential to yield valuable insights and advancements in the field of trustworthy AI within NLP.

## **6.2 Future Work**

The models and methodologies introduced in this dissertation possess considerable potential for further development into more adaptable and widely applicable solutions or for utilization in other machine learning frameworks.

### ***6.2.1 Future work 1: Advancing SSCAE with Advanced Language Models***

One promising avenue for future investigation lies in exploring advanced language models beyond Universal Sentence Encoder (USE) and GPT-2, capitalizing on the modular nature of SSCAE model. By replacing SSCAE's components with more efficient language models like Roberta, Albert, and GPT-4, researchers can gain novel insights into generating more effective perturbations, particularly concerning semantic and syntactic features. This exploration holds the potential to enhance the robustness and adaptability of the SSCAE model across various linguistic contexts and applications. Moreover, investigating the replacement of the BERT MLM component with Transformer-based models, such as Roberta and BART, presents another compelling area for study. These models offer distinct capabilities and architectures that may yield superior performance in extracting contextualized substitutions for generating high-quality adversarial examples, potentially leading to significant advancements in adversarial machine learning and natural language processing.

### ***6.2.2 Future work 2: Enhancing Robustness in Generative Pre-Trained Models***

Another compelling avenue for future research involves developing the applicability of robustness-enhancing techniques, such as adversarial training, to generative pre-trained models like GPT. While RobustEmbed and RobustSentEmbed have demonstrated remarkable efficacy in bolstering the robustness of descriptive models against adversarial attacks, their direct applicability to generative models remains a challenge due to inherent differences in model architecture and task objectives. To address this limitation, future research could focus on developing tailored methodologies and optimization strategies specifically designed to enhance the generalization and robustness

characteristics of generative pre-trained models. By leveraging insights from existing techniques and adapting them to the unique characteristics of generative models, such as GPT, researchers can pave the way for advancements in adversarial machine learning and natural language processing, ultimately contributing to the development of more resilient and trustworthy AI systems.

## Appendices

## A Training Details (RobustEmbed)

In our experimental configuration, we initialize our sentence encoder, denoted as  $f_\theta$ , utilizing the pre-trained checkpoints obtained from both BERT Devlin et al. (2019b) and RoBERTa Liu et al. (2019). RobustEmbed employs the representation of the [CLS] token as the initial embedding for sentence encoding, supplemented by a pooler layer atop the [CLS] representations to facilitate the fulfillment of contrastive learning objectives. The training regimen of RobustEmbed spans over 2 epochs, with model evaluation carried out at intervals of every 250 training steps. The optimal checkpoint, as determined by the highest average STS (Semantic Textual Similarity) score, is cherry-picked for final evaluation. To fuel the model’s training, we harness a dataset comprising  $10^6$  randomly sampled sentences harvested from English Wikipedia, as made available by the SimCSE framework Gao et al. (2021). The typical training duration for RobustEmbed spans between 2 to 4 hours. Owing to the utilization of pre-trained checkpoints, our framework showcases robustness that remains impervious to fluctuations in batch sizes, thereby affording us the flexibility to employ batch sizes ranging from 64 to 128. In the realm of transfer tasks, we ascertain the optimal hyperparameters based on the averaged score gleaned from the development sets associated with six transfer tasks.

## **B Training Details (RobustSentEmbed)**

To initialize our sentence encoder, we utilize the pre-trained checkpoints obtained from BERT Devlin et al. (2019b) and RoBERTa Liu et al. (2019). RobustSentEmbed leverages the representation of the [CLS] token as the initial embedding and incorporates a pooler layer atop the [CLS] representations to facilitate contrastive learning objectives. The training process of RobustSentEmbed spans across four epochs. The selection of the best checkpoint, determined by the highest average STS score, is crucial for final evaluation. For training purposes, we leverage a dataset comprising  $10^6$  randomly sampled sentences from English Wikipedia, as provided by the SimCSE framework Gao et al. (2021). On average, the training duration for RobustSentEmbed ranges from 2 to 4 hours. Since our framework is initialized with pre-trained checkpoints, it exhibits robustness that is not contingent upon batch sizes, thereby enabling us to utilize batch sizes of either 64 or 128 during training.

## C Adversarial Attack Methods

This section presents additional details on the diverse adversarial attack techniques employed to assess the robustness of our sentence embedding framework. The TextBugger method Li et al. (2019) identifies important words using the Jacobian matrix of the target model and selects an optimal perturbation from five types of generated perturbations. The PWWS method Ren et al. (2019) utilizes a synonym-swap technique based on a combination of word saliency scores and maximum word-swap effectiveness. TextFooler Jin et al. (2020) identifies important words, gathers synonyms, and replaces each important word with the most semantically similar and grammatically correct synonym. The BAE method Garg & Ramakrishnan (2020) employs four adversarial attack strategies involving word replacement or/and word insertion operations, where a portion of the text is masked and BERT MLM is used to generate substitutions. The BERTAttack method Li et al. (2020c) consists of two steps: (a) searching for vulnerable words/sub-words and (b) using BERT MLM to generate semantic-preserving substitutes for the vulnerable tokens.

## REFERENCES

- Agirre, E. et al. 2015, in Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), 252–263
- Agirre, E. et al. 2014, in Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014) (Dublin, Ireland: Association for Computational Linguistics), 81–91
- Agirre, E., Banea, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Mihalcea, R., Rigau, G., & Wiebe, J. 2016, in Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016) (San Diego, California: Association for Computational Linguistics), 497–511
- Agirre, E., Cer, D., Diab, M., & Gonzalez-Agirre, A. 2012, in \*SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012) (Montréal, Canada: Association for Computational Linguistics), 385–393
- Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., & Guo, W. 2013, in Second Joint Conference on Lexical and Computational Semantics (\*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity (Atlanta, Georgia, USA: Association for Computational Linguistics), 32–43
- Asl, J., Blanco, E., & Takabi, D. 2023a, in Findings of the Association for Computational Linguistics: EMNLP 2023, ed. H. Bouamor, J. Pino, & K. Bali (Singapore: Association for Computational Linguistics), 4587–4603

- Asl, J., Blanco, E., & Takabi, D. 2023b, in Findings of the Association for Computational Linguistics: EMNLP 2023, ed. H. Bouamor, J. Pino, & K. Bali (Singapore: Association for Computational Linguistics), 4587–4603
- Asl, J., Rafiei, M. H., Alohal, M., & Takabi, D. 2024a, IEEE Transactions on Dependable and Secure Computing, 1
- Asl, J. R., Panzade, P., Blanco, E., Takabi, D., & Cai, Z. 2024b, in Annual Conference of the North American Chapter of the Association for Computational Linguistics
- Bai, T., Luo, J., Zhao, J., Wen, B., & Wang, Q. 2021, in Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, ed. Z.-H. Zhou (International Joint Conferences on Artificial Intelligence Organization), 4312–4321, survey Track
- Bölücü, N., Can, B., & Artuner, H. 2023, Expert Systems with Applications, 214, 119103
- Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. 2015, in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (Lisbon, Portugal: Association for Computational Linguistics), 632–642
- Brown, T. et al. 2020, Advances in neural information processing systems, 33, 1877
- Carlini, N., & Wagner, D. 2017, in 2017 IEEE Symposium on Security and Privacy (SP) (Los Alamitos, CA, USA: IEEE Computer Society), 39–57
- Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. 2017, in Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017) (Vancouver, Canada: Association for Computational Linguistics), 1–14
- Cer, D. et al. 2018, in Proceedings of the 2018 Conference on Empirical Methods in Natural Lan-

- guage Processing: System Demonstrations (Brussels, Belgium: Association for Computational Linguistics), 169–174
- Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., & Mukhopadhyay, D. 2021, *CAAI Transactions on Intelligence Technology*, 6, 25
- Chen, Q., Zhu, X., Ling, Z.-H., Wei, S., Jiang, H., & Inkpen, D. 2017, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Vancouver, Canada: Association for Computational Linguistics), 1657–1668
- Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. 2020, in *International conference on machine learning*, PMLR, 1597–1607
- Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. 2020, in *ICLR*
- Conneau, A., & Kiela, D. 2018, in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)* (Miyazaki, Japan: European Language Resources Association (ELRA))
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. 2019a, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4171–4186
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. 2019b, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota: Association for Computational Linguistics), 4171–4186
- Ding, N. et al. 2023, *Nature Machine Intelligence*, 5, 220

- Dolan, W. B., & Brockett, C. 2005, in Proceedings of the Third International Workshop on Paraphrasing (IWP2005)
- Dong, X., Luu, A. T., Ji, R., & Liu, H. 2021, in International Conference on Learning Representations
- Ebrahimi, J., Rao, A., Lowd, D., & Dou, D. 2018, in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (Melbourne, Australia: Association for Computational Linguistics), 31–36
- Fursov, I. et al. 2022, IEEE Access, 10, 17966
- Gao, J., Lanchantin, J., Soffa, M. L., & Qi, Y. 2018, in 2018 IEEE Security and Privacy Workshops (SPW), IEEE, 50–56
- Gao, T., Yao, X., & Chen, D. 2021, in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (Online and Punta Cana, Dominican Republic: Association for Computational Linguistics), 6894–6910
- Garg, S., & Ramakrishnan, G. 2020, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Online: Association for Computational Linguistics), 6174–6181
- Goodfellow, I. J., Shlens, J., & Szegedy, C. 2015a, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, ed. Y. Bengio & Y. LeCun
- Goodfellow, I. J., Shlens, J., & Szegedy, C. 2015b, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings,

ed. Y. Bengio & Y. LeCun

Guo, C., Sablayrolles, A., Jégou, H., & Kiela, D. 2021, in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (Online and Punta Cana, Dominican Republic: Association for Computational Linguistics), 5747–5757

Hadsell, R., Chopra, S., & LeCun, Y. 2006, in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 2, IEEE, 1735–1742

Hauser, J., Meng, Z., Pascual, D., & Wattenhofer, R. 2023, in ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 1–5

He, P., Liu, X., Gao, J., & Chen, W. 2021, in International Conference on Learning Representations

Hochreiter, S., & Schmidhuber, J. 1997, *Neural computation*, 9, 1735

Honnibal, M., & Montani, I. 2017, to appear

Hu, M., & Liu, B. 2004, in Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 168–177

Jia, R., & Liang, P. 2017, in Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (Copenhagen, Denmark: Association for Computational Linguistics), 2021–2031

Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Zhao, T. 2020, in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Online: Association for Computational Linguistics), 2177–2190

Jin, D., Jin, Z., Zhou, J. T., & Szolovits, P. 2020, in , 8018–8025

Jordan, M. I., & Mitchell, T. M. 2015, *Science*, 349, 255

- Kurakin, A., Goodfellow, I. J., & Bengio, S. 2017, in 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings (OpenReview.net)
- Kurakin, A., Goodfellow, I. J., & Bengio, S. 2018, in Artificial intelligence safety and security (Chapman and Hall/CRC), 99–112
- Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. 2020, in International Conference on Learning Representations
- Le, Q., & Mikolov, T. 2014, in International conference on machine learning, PMLR, 1188–1196
- LeCun, Y., Bengio, Y., & Hinton, G. 2015, nature, 521, 436
- Lee, D., Moon, S., Lee, J., & Song, H. O. 2022, in International Conference on Machine Learning, PMLR, 12478–12497
- Li, B., Zhou, H., He, J., Wang, M., Yang, Y., & Li, L. 2020a, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Online: Association for Computational Linguistics), 9119–9130
- Li, J., Ji, S., Du, T., Li, B., & Wang, T. 2019, in Proceedings 2019 Network and Distributed System Security Symposium (Internet Society)
- Li, L., Ma, R., Guo, Q., Xue, X., & Qiu, X. 2020b, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Association for Computational Linguistics)
- Li, L., Ma, R., Guo, Q., Xue, X., & Qiu, X. 2020c, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Online: Association for

- Computational Linguistics), 6193–6202
- Li, L., & Qiu, X. 2021in , 8410–8418
- Liang, H., He, E., Zhao, Y., Jia, Z., & Li, H. 2022, *Electronics*, 11
- Liu, J. et al. 2023, *Expert Systems with Applications*, 214, 119110
- Liu, J., Kang, Y., Tang, D., Song, K., Sun, C., Wang, X., Lu, W., & Liu, X. 2022, in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2025–2039
- Liu, Y. et al. 2019, *RoBERTa: A Robustly Optimized BERT Pretraining Approach*
- Lodagala, V. S., Ghosh, S., & Umesh, S. 2023, in *2022 IEEE Spoken Language Technology Workshop (SLT)*, IEEE, 1–8
- Lu, S., Wang, M., Wang, D., Wei, X., Xiao, S., Wang, Z., Han, N., & Wang, L. 2023, *Information Sciences*, 619, 249
- Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. 2011, in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Portland, Oregon, USA: Association for Computational Linguistics)*, 142–150
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. 2018a, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings (OpenReview.net)*
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. 2018b, in *International Conference on Learning Representations*
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. 2018c, in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018,*

Conference Track Proceedings (OpenReview.net)

Mahesh, B. 2020, International Journal of Science and Research (IJSR).[Internet], 9, 381

Maheshwary, R., Maheshwary, S., & Pudi, V. 2021, in Proceedings of the 35th AAAI Conference on Artificial Intelligence

Marelli, M., Menini, S., Baroni, M., Bentivogli, L., Bernardi, R., & Zamparelli, R. 2014, in Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), 216–223

Metzen, J. H., Genewein, T., Fischer, V., & Bischoff, B. 2017, in International Conference on Learning Representations

Miao, D., Zhang, J., Xie, W., Song, J., Li, X., Jia, L., & Guo, N. 2021, Simple Contrastive Representation Adversarial Learning for NLP Tasks

Mikolov, T., Chen, K., Corrado, G., & Dean, J. 2013a, in 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, ed. Y. Bengio & Y. LeCun

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. 2013b, Advances in neural information processing systems, 26

Morris, J., Lifland, E., Yoo, J. Y., Grigsby, J., Jin, D., & Qi, Y. 2020, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 119–126

Murugesan, M. ????, Google's BERT Outperforming Human Accuracy, <https://abeyon.com/google-bert-nlp/>, accessed: 2022-01-09

- Neelakantan, A. et al. 2022, Text and Code Embeddings by Contrastive Pre-Training
- Ni, J., Hernandez Abrego, G., Constant, N., Ma, J., Hall, K., Cer, D., & Yang, Y. 2022, in Findings of the Association for Computational Linguistics: ACL 2022 (Dublin, Ireland: Association for Computational Linguistics), 1864–1874
- Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., & Kiela, D. 2020, in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Online: Association for Computational Linguistics), 4885–4901
- Pan, L., Hang, C.-W., Sil, A., & Potdar, S. 2022in , 11130–11138
- Pang, B., & Lee, L. 2004, in Annual Meeting of the Association for Computational Linguistics
- Pang, B., & Lee, L. 2005a, in Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05) (Ann Arbor, Michigan: Association for Computational Linguistics), 115–124
- Pang, B., & Lee, L. 2005b, in Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05) (Ann Arbor, Michigan: Association for Computational Linguistics), 115–124
- Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. 2017, in Proceedings of the 2017 ACM on Asia conference on computer and communications security, 506–519
- Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. 2016, in 2016 IEEE symposium on security and privacy (SP), IEEE, 582–597
- Paszke, A. et al. 2019, in Advances in Neural Information Processing Systems 32 (Curran Associates, Inc.), 8024–8035

- Pennington, J., Socher, R., & Manning, C. 2014, in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Doha, Qatar: Association for Computational Linguistics), 1532–1543
- Qiu, Y., Zhang, J., & Zhou, J. 2021, in Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021 (Association for Computational Linguistics), 1698–1707
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. 2019, OpenAI blog, 1, 9
- Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. 2016, in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (Austin, Texas: Association for Computational Linguistics), 2383–2392
- Rehurek, R., & Sojka, P. 2011, NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 3
- Reimers, N., & Gurevych, I. 2019, in Conference on Empirical Methods in Natural Language Processing
- Ren, S., Deng, Y., He, K., & Che, W. 2019, in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Florence, Italy: Association for Computational Linguistics), 1085–1097
- Rima, D. N., Heo, D., & Choi, H. 2022, Computer Speech & Language, submitted
- Shafahi, A., Najibi, M., Xu, Z., Dickerson, J., Davis, L. S., & Goldstein, T. 2020in , 5636–5643
- Silva, S. H., & Najafirad, P. 2020, arXiv preprint arXiv:2007.00753
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. 2013a, in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing

- (Seattle, Washington, USA: Association for Computational Linguistics), 1631–1642
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. 2013b, in Proceedings of the 2013 conference on empirical methods in natural language processing, 1631–1642
- Song, L., Yu, X., Peng, H.-T., & Narasimhan, K. 2021, in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Online: Association for Computational Linguistics), 3724–3733
- Su, J., Cao, J., Liu, W., & Ou, Y. 2021, CoRR, abs/2103.15316
- Sun, C., Qiu, X., Xu, Y., & Huang, X. 2019, in China national conference on Chinese computational linguistics, Springer, 194–206
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., & Fergus, R. 2014, in 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, ed. Y. Bengio & Y. LeCun
- Wang, D., Ding, N., Li, P., & Zheng, H.-T. 2021, in Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Association for Computational Linguistics, 2332–2342
- Wang, D., Gong, C., & Liu, Q. 2019a, in International Conference on Machine Learning, PMLR, 6555–6565
- Wang, Q., Zhang, W., Lei, T., Cao, Y., Peng, D., & Wang, X. 2023, Knowledge-Based Systems, 266, 110381
- Wang, T., & Isola, P. 2020, in International Conference on Machine Learning, PMLR, 9929–9939

- Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., & Gu, Q. 2019b, in Proceedings of Machine Learning Research, Vol. 97, Proceedings of the 36th International Conference on Machine Learning, ed. K. Chaudhuri & R. Salakhutdinov (PMLR), 6586–6595
- Wiebe, J., Wilson, T., & Cardie, C. 2005, Language resources and evaluation, 39, 165
- Williams, A., Nangia, N., & Bowman, S. 2018, in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (New Orleans, Louisiana: Association for Computational Linguistics), 1112–1122
- Wolf, T. et al. 2020, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (Online: Association for Computational Linguistics), 38–45
- Wu, C., Zhang, R., Guo, J., De Rijke, M., Fan, Y., & Cheng, X. 2023, ACM Transactions on Information Systems, 41, 1
- Xu, H., Ma, Y., Liu, H.-C., Deb, D., Liu, H., Tang, J.-L., & Jain, A. K. 2020, International Journal of Automation and Computing, 17, 151
- Yan, Y., Li, R., Wang, S., Zhang, F., Wu, W., & Xu, W. 2021, in Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (Online: Association for Computational Linguistics), 5065–5075
- Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. 2019, Advances in neural information processing systems, 32

- Young, P., Lai, A., Hodosh, M., & Hockenmaier, J. 2014, Transactions of the Association for Computational Linguistics, 2, 67
- Zhang, W. E., Sheng, Q. Z., Alhazmi, A., & Li, C. 2020a, ACM Transactions on Intelligent Systems and Technology (TIST), 11, 1
- Zhang, X., Zhao, J., & LeCun, Y. 2015, Advances in neural information processing systems, 28, 649
- Zhang, Y., He, R., Liu, Z., Lim, K. H., & Bing, L. 2020b, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 1601–1610
- Zhu, C., Cheng, Y., Gan, Z., Sun, S., Goldstein, T., & Liu, J. 2020, in International Conference on Learning Representations
- Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., & Fidler, S. 2015, in Proceedings of the IEEE international conference on computer vision, 19–27