基于自动稀释的文本对抗攻击强化方法[1]

房钰深　陈振华　何琨*

（华中科技大学计算机科学与技术学院 武汉 430074）

（*通信作者：brooklet60@hust.edu.cn）

Text Adversarial Attack Capability Enhancement Method Based on Automatic Dilution

Yushen Fang[1], Zhenhua Chen[1] and Kun He[1]

[1]（*Department of Computer Science and Technology, Huazhong University of Science and Technology,Wuhan* 430074）

Abstract Using adversarial examples for training can enhance the robustness of deep neural networks. Therefore, improving the success rate of adversarial attacks is a significant focus in the field of adversarial example research. Diluting original samples can bring them closer to the decision boundary of the model, thereby increasing the success rate of adversarial attacks. However, existing dilution algorithms suffer from issues such as reliance on manually generated dilution pools and single dilution targets. This paper proposes a method to enhance the capability of text adversarial attacks based on automatic dilution, called the Automatic Multi-positional Dilution Preprocessing (AMDP) algorithm. The AMDP algorithm eliminates the reliance on manual assistance in the dilution process and can generate different dilution pools for different datasets and target models. Additionally, AMDP extends the targeted words for dilution, broadening the search space of dilution operations. As an input transformation method, AMDP can be combined with other adversarial attack algorithms to further enhance the attack performance. Experimental results demonstrate that AMDP achieves an average success rate increase of approximately 10% on BERT, WordCNN, and WordLSTM classification models, while also reducing the average modification rate of original samples and the average number of accesses to the target model.

Key words Adversarial machine learning; Adversarial samples; Text dilution; Classification boundaries; Natural language processing

摘要 使用对抗样本进行训练可以提升深度神经网络的鲁棒性，因此，如何提升对抗攻击成功率是对抗样本研究领域的一个重要内容。对原始样本进行稀释操作可以使其更靠近模型的决策边界，进而提高对抗攻击的成功率。然而，现有稀释算法存在需要基于人工生成的稀释池和稀释目标词性单一等问题。本文提出了一种基于自动稀释的文本对抗攻击强化方法，称为自动多词性稀释预处理（Automatic Multi-positional Dilution Preprocessing，AMDP）算法。AMDP 算法使稀释过程摆脱了对人工辅助的依赖，能够针对不同数据集和目标模型生成不同的稀释池。同时，AMDP 算法还扩展了稀释目标的词性，扩大了稀释操作的搜索空间。作为一种输入转换方法，AMDP 还可以与其他对抗攻击算法相结合，以进一步提高对抗攻击的性能。实验结果表明，AMDP 在 BERT、WordCNN 和 WordLSTM 分类模型上的攻击成功率平均提升约 10%，同时减少了对原始样本的平均修改率和对目标模型的平均访问次数。

关键词 对抗机器学习；对抗样本；文本稀释；分类边界；自然语言处理

---

[1] **基金项目** 国家自然科学基金(62076105, U22B2017)资助项目。
学生第一作者论文，2024 级硕士生。

1 引言

近年来，深度学习模型在计算机视觉[1]、自然语言处理[2]等多个领域中展现出优异的性能并得到了广泛应用。然而有研究表明，通过对正常输入样本添加人类无法察觉的微小扰动即可使深度学习模型产生错误的结果，这种对模型安全造成威胁的样本也被称为对抗样本[3]。这一现象引发了研究者对深度学习模型安全性的关注，使得如何生成并防御对抗样本成为研究热点。

常见的对抗样本生成方法通常分为字符级、词语级和句子级三种。但由于字符级可能会产生拼写错误，句子级可能会导致语义偏离，故词语级生成的对抗样本质量更高。因为语言文本的离散性，研究者们一般采用单词替换操作来生成对抗样本。Moustafa 等人[4]使用遗传算法指导同义词替换来生成对抗样本。Siddhant 等人[5]通过词对嵌入表示引入扰动来生成对抗样本。这些对抗攻击方法都是直接对原输入样本进行攻击，然而叶文滔等人[6]发现，离分类边界更近的样本具有更高的对抗攻击成功率，并提出了稀释攻击方法。他们基于人工生成的稀释池进行增稀释，并对副词进行删稀释，从而让样本更靠近分类边界。但是，人工生成的稀释池不仅成本高，并且难以拓展到其他任务上，同时删稀释中考虑到的词性也比较单一，没有充分发挥稀释的作用。

为解决这一问题，本文提出一种自动多词性稀释预处理 (Automatic Multi-positional Dilution Preprocessing, AMDP)算法。该算法通过对输入句子进行预处理使其更靠近分类边界，而达到提升基础对抗攻击方法对其的攻击成功率。AMDP 算法包含自动增稀释预处理(Automatic Add Word Pretreatment, AAWP) 和多词性删稀释预处理(Multi-positional Delete Word Pretreatment, MDWP)两部分。其中，AAWP 通过查询目标模型来筛选出对分类结果影响更大的词，并动态生成稀释池；而 MDWP 扩展了稀释的词性范围，扩大了稀释攻击的搜索空间。

相比以往的稀释算法，AMDP 对于不同的数据集与目标模型可以生成不同的稀释池，并扩大了稀释攻击的搜索空间，不仅使算法摆脱了对人工辅助的依赖，而且提升了对抗样本生成效率和质量。本文的主要贡献包括以下三个方面：

(1) 提出了 AAWP 算法，通过查询目标模型引入额外信息，使稀释过程摆脱了对人工辅助的依赖，实现了对不同数据集和目标模型生成不同的稀释池。

(2) 提出了 MDWP 算法，通过扩展稀释目标的词性，扩大了稀释的搜索空间，使得稀释后的样本更靠近分类边界。

(3) 在多个文本分类数据集上的实验结果表明，

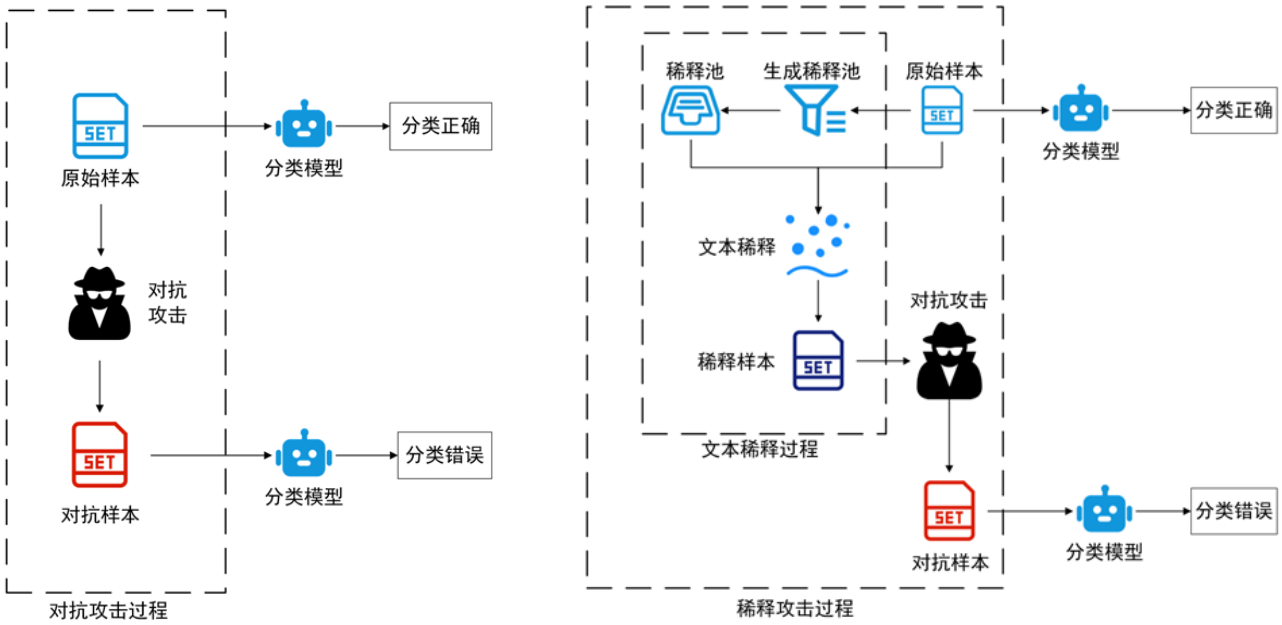

Fig.1-1 The Relationship between Traditional Adversarial Sample Generation Process and Text Dilution Process

图 1-1 传统对抗样本生成过程和文本稀释过程的关系

AMDP 在 BERT、WordCNN 和 WordLSTM 模型上的攻击成功率平均提升约 10%，同时减少了对原始样本的平均修改率和对目标模型的平均访问次数。

2 相关理论

2.1 分类边界理论

对于文本分类模型F，假设样本标签集合为Y，句子x对应各类标签的置信度为$P(y_i|x),\ y_i \in Y$。如果原始样本中的句子x对应的真实标签为y，那么x的边界距离可表示为：

$$D(x) = \left| P(y|x) - \max_{y' \in Y, y' \neq y} P(y'|x) \right| 。 \quad (1)$$

边界距离反映的是模型对样本分类结果的确定程度，$D(x)$越小，分类出错的可能性越大。因此，定义使得边界距离$D(x) = 0$的所有样本组成分类模型F的分类边界。

对于原始数据集X，令一个基础对抗攻击方法对于数据集X与分类模型 F 的对抗攻击成功率为：

$$S(X, F) = \frac{N_{adv}}{N_{correct}}, \quad (2)$$

其中，N_{adv}为成功生成的对抗样本数量，$N_{correct}$为模型正确分类的样本数量。对原始样本计算边界距离$D(x)$，并按边界距离的大小对数据集X进行降序排序。将排序后的数据集X以δ作为临界值进行划分，得到两个样本集$X_r = \{x|D(x) \leq \delta,\ x \in X\}$和$X_s = \{x|D(x) > \delta,\ x \in X\}$。基于上述定义，叶文滔等[5]人提出了分类边界理论：若两个样本集满足t_s中任意一个元素的边界距离都大于t_r中的最大边界距离，则$S(X_s, F) \leq S(X_r, \mathrm{F})$成立。即若任取$X_s$中的一个元素，均能不重复地在$X_r$中找到与之对应的边界距离更小的元素，那么$X_r$比$X_s$对抗攻击的成功率更高。

2.2 稀释攻击

稀释攻击是通过对句子的预处理使其更靠近分类边界来提高对抗攻击性能。稀释攻击主要有增稀释和删稀释两种方式。增稀释是指选取一些不影响句子语义的词语，并将这些词语添加在句子的合适位置，以实现攻击效果。删稀释是指保证句子结构不变和句子语义不变的情况下，删除句子中的某些词语以达到攻击的效果。

Samanta 等人[7]发现，在对抗攻击中词语的替换和增删策略具有重要作用，特别是在对句子边界距离影响大的形容词处增删副词这一策略。这种利用副词操作语句的思路，实际上源于自然语言的一种普遍规律：副词在句子中通常用于强调，而非决定句子的基本含义。因此，副词具有调控语句，使其逐渐远离或接近分类边界的能力。这使得副词成为实现语句稀释效果的关键工具。

但是，稀释攻击仍然存在一些缺点。由于仅针对副词进行操作，使得攻击手段较为单一。此外，在生成稀释池的过程中，基于人工生成的稀释池难免会存在遗漏的情况。本文提出的自动文本稀释预处理算法通过自动生成稀释池和对多词性进行稀释处理，成功地解决了这两个问题。

3 基于自动稀释的对抗攻击强化方法

本文提出一种基于自动稀释的文本对抗攻击强化方法，称为自动多词性稀释预处理（AMWP）算法。该算法通过对句子的预处理使其更靠近分类边界而达到对攻击算法的优化效果。

该算法由自动增稀释预处理（Automatic Add Word Pretreatment, AAWP）算法和多词性删稀释预处理（Multi-positional Delete Word Pretreatment，MDWP）两部分组成。

3.1 自动增稀释预处理

现有的增稀释预处理主要是基于人工种子进行，无法对不同的数据集生成不同的稀释池。本文设计了增加词汇的预处理算法(AAWP)，该算法通过稀释池对句子进行稀释预处理，使句子靠近分类边界，进而实现对攻击算法的强化。如算法 1 所示，AAWP 算法基本步骤如下：

对于一个分类模型$F(x)$，$x = \{w_1, w_2, \ldots, w_n\}$是包含 n 个词的句子，样本的边界距离为$D(x)$。假设该句子中的单词w_i为副词，删除副词后的句子$x' = \{w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n\}$。若$x$和$x'$保持句子语义不变，即在$F(x) = F(x')$ 的前提下，有$D(x') > D(x)$，则表明删除单词w_i使得句子x更远离分类边界，此时记单词w_i为弱极性副词，可以将其视为稀释种子。

除了种子中的弱极性副词外，可能存在未被检测到的其他副词。因此，AAWP 算法使用查找种子中副词的同义词来实现稀释池的构建。

对于一个句子而言，副词一般存在于形容词或名词之前，且添加副词对句子的语义不存在明显的影响。因此，AAWP 算法在稀释预处理时，会先定位句子中适合添加副词的位置。当找到适当的位置后，AAWP 算法将稀释池W中的 n 个候选副词都添加到该位置，从而生成 n 个新的样本。将生成的 n

个新样本和原始未修改的样本组合成包含 n+1 个样本的搜索空间，并在此搜索空间中找到分类标签不变且边界距离最小的样本作为预处理后生成的样本，并替换掉原始数据集中对应的样本。

算法 1　AAWP 算法

输入：句子集合$X=\{x_1,x_2,\dots,x_n\}$, 分类函数F，正确分类y,句子$x=\{w_1,w_2,\dots,w_n\}$，分类边界函数D，同义词搜索函数Syn，副词稀释池W。

输出：处理后的句子集合X

1: $W \leftarrow \{\}$
2: ***for*** X 中的每个句子 x：
3: ***for*** x 中的每个词 w_i:
4: ***if*** w_i 是副词
5: $x' \leftarrow x - w_i$
6: ***if*** $F(x')=y$ 且 $D(x')>D(x)$
7: W 中添加 $Syn(w_i)$
8: ***for*** X 中的每个句子 x：
9: ***for*** x 中的每个词 w_i:
10: ***if*** w_i 是副词或动词
11: ***for*** W 中的每个词 adv：
12: $x' \leftarrow x +$ (在 w_i 之前添加 adv)
13: 只记录在 $F(x')=y$ 时 x'的最小值$D(x')$
14: 将 X 中的句子 x 替换成句子 x'
15: ***return*** X

对于 AAWP 算法，假设数据集中句子集合X的句子总数为M，句子集合X中每个句子x的平均长度为N，副词稀释池中的总词数为K，那么自动构建稀释池算法的时间复杂度为$O(M\cdot N)$，进行增稀释攻击的时间复杂度为$O(K\cdot M\cdot N)$。值得注意的是，由于增稀释池中为句子中筛选出的副词及其同义词，所以K相比$(M\cdot N)$通常会少 2 个数量级。

3.2 多词性删稀释攻击

现有的删稀释预处理仅针对副词进行操作，会导致攻击目标单一，无法达到最佳效果。针对这一问题，本文设计了多词性删除单词预处理 (MDWP) 算法，使得删稀释预处理摆脱了词性的限制。该算法在保证句子分类标签不变的前提下，通过删除句子中的部分单词，使得句子靠近分类边界，进而实现对攻击算法的强化。如算法 2 所示，MDWP 算法将删除稀释问题分为句子主谓宾成分完整和主谓宾成分不完整两类。

类型 1　样本主谓宾完整(行 1~6)

当句子主谓宾完整，即句子拥有完整结构时，若要保证句子结构及语义不变的前提条件，则在进行删除稀释的过程中不能对句子的主谓宾成分进行操作。MDWP 算法对于此类问题的处理策略是对句子中每个非主谓宾成分进行操作，即对句子$x=\{w_1,w_2,\dots,w_n\}$，依次选取句子中不充当主谓宾的词语w_i，生成新样本$x'=\{w_1,\dots,w_{i-1},w_{i+1},\dots,w_n\}$。若句子$x$和句子$x'$保持句子语义不变，即在保证$F(x)=F(x')$ 的前提下，有$D(x')<D(x)$，则表示删除词语w_i使句子更靠近分类边界，故将新样本x'作为原样本x预处理后的结果。

类型 2 样本主谓宾不完整(行 7~12)

若句子主谓宾不完整，在保证语义不变的条件下，无法筛选出所有可以删除稀释的单词。但由于形容词和副词在句子中通常起修饰作用，因此，MDWP 算法在处理主谓宾不完整的句子时，采用对句子中的形容词或副词进行处理，即筛选出句子中的目标词汇，并对其进行删除稀释，具体稀释过程和选择策略与样本主谓宾完整时一致。

算法 2　MDWP 算法

输入：句子集合$X=\{x_1,x_2,\dots,x_n\}$, 分类函数F，句子$x=\{w_1,w_2,\dots,w_n\}$，分类边界函数D

输出：处理后的句子集合X

1: ***for*** X 中的每个句子 x：
2: ***if*** 句子 x 主谓宾齐全
3: ***for*** 句子 x 中的每个非主谓宾词 w_i：
4: $x' \leftarrow x - w_i$
5: ***if*** $F(x')=F(x)$ 且 $D(x')<D(x)$
6: 将 X 中的句子 x 替换成句子 x'
7: ***else***
8: ***for*** 句子 x 中的每个非主谓宾词 w_i：
9: ***if*** w_i 是形容词或副词
10: $x' \leftarrow x - w_i$
11: ***if*** $F(x')=F(x)$ 且 $D(x')<D(x)$
12: 将 X 中的句子 x 替换成句子 x'
13: ***return*** x

对于 MDWP 算法，假设数据集中句子集合X的总数为M，句子集合X中每个句子x的平均长度为N，那么主谓宾完整时的算法时间复杂度为$O(M\cdot N)$，主谓宾不完整时的算法时间复杂度亦为$O(M\cdot N)$。因此，MDWP 算法的整体复杂度为$O(M\cdot N)$。

4　实验结果与分析

4.1 实验设置

数据集：实验使用了 4 个文本分类数据集，每个数据集随机选取了 1000 个样本作为测试样本。具体信息如下：

(1) **MR(movie reviews)**[8]：电影评论二分类数据集。
(2) **AG's News**[9]：新闻文章四分类数据集。
(3) **IMDB (Internet Movie Database)**[10]：网络电影评论二分类数据集。
(4) **Yelp**[11]：Yelp 网站商家评论二分类数据集。

受害者模型：在对抗样本攻击中，受害者模型是指被攻击的机器学习模型。由于实验选取的数据集是文本分类数据集，因此受害者模型选取了 3 个较为典型的文本分类器。

(1) **BERT**[12]：该模型是一种基于 Transformer 架构的预训练语言模型，它由多个编码器层组成，其中每个编码器层都是由多头自注意力机制和前馈神经网络组成。
(2) **wordCNN**[13]：该模型是一种基于卷积神经网络的文本分类模型，主要利用卷积操作来捕获输入文本中的局部特征。
(3) **wordLSTM**[14]：该模型是基于长短期记忆网络的文本分类模型，其中输入是以词为单位表示的文本序列。

基础攻击方法：本实验选择了 3 种经典的攻击算法作为基础对抗攻击方法。

(1) **PWWS**[15]：该算法通过使用同义词替换文本中原单词来生成对抗样本，使得原始文本在保持语义不变的情况下，被分类模型错误地分类。
(2) **TextBugger**[16]：该算法采用了一种叫做“特征翻转”的技术，通过翻转文本中的一些关键特征，使得文本的含义保持不变，但能够迷惑文本分类模型。
(3) **SememePSO**[17]：该算法的核心思想是通过将词语表示为语义向量，然后利用粒子群优化算法来学习词语的语义向量表示。

4.2 评价指标

(1) **攻击成功率(Attack Successful Rate, ASR)**：生成的对抗样本成功地误导该模型产生错误分类的比例。成功率是算法效果的主要指标，攻击成功率越高，说明生成的效果越好。
(2) **词修改率(Word Modified Rate, WMR)**：在生成对抗样本时，对原始文本中的词语进行修改的比例。该比例表示生成的对抗样本相对于原始文本的变化程度，修改率越低对抗样本语义变化就越小，攻击效果越好。
(3) **余弦相似度(Cosine Similarity, CS)**：余弦相似度是通过计算两个句子嵌入表示后的向量之间的夹角的余弦值来确定两个句子之间的相似度。余弦相似度越大，说明对抗样本与原样本越相似，攻击效果越好。
(4) **模型访问次数(Victim Model Queries, VMQ)**：在对抗样本攻击中，攻击者通常需要与目标模型进行多次交互。这些交互通常是查询模型获取其预测结果，以便优化生成的对抗样本。通常情况下，攻击者希望通过尽可能少的查询次数来生成有效的对抗样本，以降低攻击的成本和时间。

4.3 实验结果与分析

所设计的 AMDP 算法在 3 个攻击模型、4 个数据集和 3 个受害者模型上进行了交叉实验，以验证 AMDP 算法的有效性。

值得注意的是，由于以往的稀释实验大多基于人工种子生成的稀释池进行稀释攻击，但这些方法都暂未公开影响攻击性能的人工种子。因此，目前无法进行与其他稀释攻击方法的对比实验。

表 1 展示了 3 种攻击算法对 3 种文本分类模型进行攻击时，AMDP 算法对攻击算法的提升效果。表 1 从多个角度表明，AMDP 算法稀释后的文本攻击效果优于未稀释的攻击样本。在攻击成功率方面，经过 AMDP 算法稀释后的平均成功率比稀释前提升 10%以上，特别是在基于 BERT 的二分类模型，攻击成功率至少提升 25%；在相似度方面，稀释后的样本大多优于未稀释的样本，且从词修改率和余弦相似度两个方面所体现的都是相似度的提升；在效率方面，稀释后的样本对模型访问的次数平均减少 10%左右。

此外，由于 AMDP 算法在对抗攻击算法之前执行，且执行过程与对抗攻击算法没有直接耦合关系，因此 AMDP 算法具有良好的迁移性，可以和任何对抗攻击算法共同使用以进一步提升攻击效果。

Table 1: The improvement of AMDP algorithm on various attack methods

表 1　AMDP 算法对不同攻击方法的性能提升

分类模型	基础攻击方法	指标	MR		AG		IMDB		YELP	
			未稀释	AMDP	未稀释	AMDP	未稀释	AMDP	未稀释	AMDP
BERT	TextBugger	SR	0.77	**0.97**	0.75	**0.93**	0.85	**0.93**	0.86	**1.00**
		CS	0.93	**0.97**	0.93	**0.98**	0.98	**1.00**	0.95	**0.99**
		WMR	10.82	**7.22**	21.04	**9.47**	217.71	**196.89**	119.12	**99.38**
		VMQ	49.52	**32.46**	118.01	**66.75**	425.06	**281.44**	303.86	**167.03**
	PWWS	SR	0.64	**0.96**	0.53	**0.90**	0.82	**0.93**	0.80	**1.00**
		CS	0.96	**0.98**	0.95	**0.99**	0.99	**1.00**	0.97	**0.99**
		WMR	9.14	**6.73**	9.49	**5.60**	207.35	**192.97**	105.61	**96.49**
		VMQ	135.34	**133.03**	248.15	249.28	1661.1	**1619.9**	992.38	994.65
	SememePSO	SR	0.45	**0.95**	0.18	**0.76**	0.21	**0.88**	0.18	**0.96**
		CS	0.95	**0.98**	0.98	**0.99**	1.00	1.00	0.98	**0.99**
		WMR	7.58	**6.68**	6.11	**5.09**	198.22	**183.84**	67.42	91.72
		VMQ	64.54	**22.74**	88.28	**45.51**	110.58	**38.61**	106.07	**34.54**
WordCNN	TextBugger	SR	0.96	**0.99**	1.00	1.00	1.00	1.00	0.98	**1.00**
		CS	0.94	**0.97**	0.95	**0.99**	0.99	**1.00**	0.97	**0.99**
		WMR	9.87	**7.34**	20.85	**7.26**	206.31	**191.59**	118.42	**96.31**
		VMQ	41.88	**31.66**	89.69	**54.07**	313.63	**274.72**	223.11	**162.95**
	PWWS	SR	0.88	**0.98**	0.84	**1.00**	1.00	1.00	0.97	**1.00**
		CS	0.96	**0.98**	0.97	**0.99**	0.99	**1.00**	0.98	**1.00**
		WMR	8.12	**6.63**	8.59	**4.52**	191.16	**187.42**	99.92	**93.77**
		VMQ	129.74	131.00	244.78	245.16	1570.2	1585.1	979.83	983.73
	SememePSO	SR	0.66	**0.95**	0.27	**0.96**	0.26	**0.95**	0.20	**0.97**
		CS	0.96	**0.98**	0.98	**0.99**	0.99	**1.00**	0.98	**1.00**
		WMR	7.60	**6.55**	6.05	**4.25**	152.57	183.42	66.60	91.21
		VMQ	45.61	**23.11**	77.93	**28.00**	107.85	**32.47**	77.96	**26.65**
WordLSTM	TextBugger	SR	0.94	**0.98**	0.92	**0.99**	0.99	**1.00**	0.96	**1.00**
		CS	0.94	**0.97**	0.94	**0.98**	0.98	**0.99**	0.96	**0.99**
		WMR	9.65	**7.58**	20.79	**7.80**	209.57	**191.57**	122.63	**96.18**
		VMQ	41.81	**32.57**	105.78	**57.45**	348.52	**275.13**	256.60	**163.00**
	PWWS	SR	0.86	**0.98**	0.63	**0.97**	0.98	**1.00**	0.93	**1.00**
		CS	0.96	**0.98**	0.96	**0.99**	0.99	**1.00**	0.98	**0.99**
		WMR	8.09	**6.83**	9.52	**4.74**	191.40	**187.85**	104.37	**93.95**
		VMQ	129.73	132.24	249.02	**247.16**	1570.3	1583.4	1007.9	**990.16**
	SememePSO	SR	0.66	**0.96**	0.21	**0.89**	0.21	**0.94**	0.20	**0.97**
		CS	0.96	**0.98**	0.98	**0.99**	0.99	**1.00**	0.98	**1.00**
		WMR	7.67	**6.76**	6.78	**4.36**	148.62	183.52	58.70	88.27
		VMQ	48.20	**23.18**	93.94	**30.86**	107.72	**36.33**	90.59	30.10

Table 2 Ablation study of AMDP algorithm (Components: AAWP and MDWP algorithms)

表 2　AMDP 算法的消融实验（组件: AAWP 和 MDWP 算法）

基础攻击方法	指标	WordCNN				WordLSTM			
		未稀释	AAWP	MDWP	AMDP	未稀释	AAWP	MDWP	AMDP
TextBugger	ASR	0.958	**0**.968	0.976	**0.989**	0.930	0.939	0.960	**0.984**
	CS	0.943	0.945	0.955	**0.973**	0.946	0.949	0.952	**0.972**
	WMR	9.882	10.47	**5.864**	7.342	9.749	9.604	**6.143**	7.580
	VMQ	41.58	42.07	**28.39**	31.66	41.40	41.90	**28.47**	32.57
PWWS	ASR	0.884	0.872	0.935	**0.975**	0.863	0.896	0.936	**0.981**
	CS	0.963	0.967	0.967	**0.978**	0.963	0.965	0.966	**0.979**
	WMR	8.118	8.206	**5.151**	6.634	8.094	8.102	**5.390**	6.830
	VMQ	129.7	130.6	**104.3**	131.0	129.7	132.4	**102.4**	132.2
SememePSO	ASR	0.661	0.659	0.902	**0.951**	0.665	0.706	0.903	**0.956**
	CS	0.963	0.966	0.967	**0.978**	0.964	0.964	0.964	**0.979**
	WMR	7.590	7.843	**5.085**	6.549	7.702	7.623	**5.299**	6.765
	VMQ	46.81	45.92	24.05	**23.11**	49.51	48.99	25.70	**23.18**

4.4 消融实验

本文设计了消融实验来进一步分析 AAWP 算法和 MDWP 算法的稀释攻击效果。

表 2 展示了 AAWP 算法和 MDWP 算法的消融实验的结果。结果表明，仅 AAWP 算法对样本进行稀释效果并不明显；仅使用 MDWP 算法对攻击成功率有明显提升，且在降低词修改率和减少模型访问次数上具有极佳的效果。相比于仅使用 MDWP 算法，使用 AMDP 算法可能会略微降低对抗样本的生成效率，但是却在攻击成功率上获得了更大的提升。因此，可以根据不同的需求选择不同的稀释方式。

5　结论

本文基于分类边界理论，设计了一种自动多词性稀释预处理（AMDP）算法，并通过 3 种经典对抗攻击方法、4 个文本分类数据集和 3 种分类模型的交叉对比实验，验证了算法的可行性和有效性。

相比于以往的稀释攻击算法，AMDP 算法不仅使稀释过程摆脱了对人工辅助的依赖，而且可以针对不同数据集和目标模型生成不同的稀释池，在降低了人工成本的同时提升优化的针对性。

此外，AMDP 算法还扩展了稀释目标的词性，使稀释不仅仅针对副词。此外，由于 AMDP 与被优化算法之间耦合度低，因此可以与任何已有的攻击算法共同使用，且无需对已有的攻击算法进行修改。

在未来工作中，所设计的算法仍有一些可改进的地方。如所提出的自动生成稀释池算法可能无法涵盖所有可以用于稀释的副词。此外，由于 AMDP 算法是基于单词级的对抗攻击方法，对于字符级和句子级的攻击方法的优化效果还有待提升。这些都是未来值得进一步探索的方向。

参 考 文 献

[1] Kim I, Baek W, Kim S. Spatially attentive output layer for image classification[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. Seattle, WA, USA, IEEE 2020: 9533-9542.

[2] Liu X, He P, Chen W, et al. Multi-task deep neural networks for natural language understanding[J]. Association for Computational Linguistics, 2019,4487-4496.

[3] Li Pan, Zhao Wentao, Liu Qiang, et al. A Review of Machine Learning Security Issues and Defense Techniques[J]. Journal of Computer Science and Exploration, 2018, 12(2): 171-184.
（李盼, 赵文涛, 刘强, 等. 机器学习安全性问题及其防御技术研究综述[J]. 计算机科学与探索, 2018, 12(2): 171-184.）

[4] Alzantot M, Sharma Y, Elgohary A, et al. Generating natural language adversarial examples[J]. Conference on Computational Natural Language Learning, 2016, 10-21.

[5] Garg S, Ramakrishnan G. Bae: Bert-based adversarial examples for text classification[J]. Empirical Methods in Natural Language Processing, 2020, 6174-6181.

[6] Ye Wentao, Zhang Min, Chen Yixiang. Strengthening Text Adversarial Attack Defense Method Based on Semantic-Level Sentence Dilution[J]. Journal of Software, 2022, 34(7): 3313-3328.
（叶文滔, 张敏, 陈仪香. 基于义原级语句稀释法的文本对抗攻击能力强化方法[J]. 软件学报, 2022, 34(7): 3313-3328.）
[7] Samanta S, Mehta S. Towards crafting text adversarial samples. arXiv: 1707.02812, 2017.
[8] Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment classification using machine learning techniques[J]. Empirical Methods in Natural Language Processing, 2002, 79-86.
[9] Zhang X, Zhao J, LeCun Y. Character-level convolutional networks for text classification[J]. Advances in neural information processing systems, 2015, 28.
[10] IMDB Dataset of 50K Movie Reviews.[OL] [2016-11-26] https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
[11] Yelp Dataset [OL]. [2021-02-16] https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset
[12] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. North American Chapter of the Association for Computational Linguistics, 2019, 4171-4186.
[13] Chen Y. Convolutional neural network for sentence classification[D]. University of Waterloo, 2015.
[14] Bowman S R, Vilnis L, Vinyals O, et al. Generating sentences from a continuous space[J]. Conference on Computational Natura Language Learning, 2016, 10-21.
[15] Ren S, Deng Y, He K, et al. Generating natural language adversarial examples through probability weighted word saliency[C]//Proceedings of the 57th annual meeting of the association for computational linguistics. 2019: 1085-1097.
[16] Jin D, Jin Z, Zhou J T, et al. Is bert really robust? a strong baseline for natural language attack on text classification and entailment[C]//Proceedings of the AAAI conference on artificial intelligence. 2020, 34(05): 8018-8025.
[17] Zang Y, Qi F, Yang C, et al. Word-level textual adversarial attacking as combinatorial optimization[J]. Association for Computational Linguistics, 2020, 6066-6080.