网络预览版

自然语言处理导论

张奇 桂韬 黄萱菁

2023 年 2 月 15 日

数学符号

数与数组

α	标量
$\boldsymbol{\alpha}$	向量
$\boldsymbol{A}$	矩阵
$\mathbf{A}$	张量
$\boldsymbol{I}_n$	n 行 n 列单位矩阵
$\boldsymbol{v}_w$	单词 w 的分布式向量表示
$\boldsymbol{e}_w$	单词 w 的独热向量表示：[0,0,...,1,0,...0]，w 下标处元素为 1

索引

α_i	向量 $\boldsymbol{\alpha}$ 中索引 i 处的元素
α_{-i}	向量 $\boldsymbol{\alpha}$ 中除索引 i 之外的元素
$w_{i:j}$	序列 w 中从第 i 个元素到第 j 个元素组成的片段或子序列
A_{ij}	矩阵 $\boldsymbol{A}$ 中第 i 行、第 j 列处的元素
$\boldsymbol{A}_{i:}$	矩阵 $\boldsymbol{A}$ 中第 i 行
$\boldsymbol{A}_{:j}$	矩阵 $\boldsymbol{A}$ 中第 j 列
$\boldsymbol{A}_{ijk}$	三维张量 $\mathbf{A}$ 中索引为 (i,j,k) 处元素
$\mathbf{A}_{::i}$	三维张量 $\mathbf{A}$ 中的一个二维切片

集合

$\mathbb{A}$	集合
$\mathbb{R}$	实数集
$\mathbb{C}$	复数集
$\{0,1,...,n\}$	含 0 和 n 的正整数的集合
$[a,b]$	a 到 b 的实数闭区间
$(a,b]$	a 到 b 的实数左开右闭区间

线性代数

$\boldsymbol{A}^\top$	矩阵 $\boldsymbol{A}$ 的转置
$\boldsymbol{A} \odot \boldsymbol{B}$	矩阵 $\boldsymbol{A}$ 与矩阵 $\boldsymbol{B}$ 的 Hadamard 乘积
$\det \boldsymbol{A}^\top$	矩阵 $\boldsymbol{A}$ 的行列式
$[\boldsymbol{x}; \boldsymbol{y}]$	向量 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 的拼接
$[\boldsymbol{U}; \boldsymbol{V}]$	矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{V}$ 沿行向量拼接
$\boldsymbol{x} \cdot \boldsymbol{y}$ 或 $\boldsymbol{x}^\top \boldsymbol{y}$	向量 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 的点积

微积分

$\frac{\mathrm{d}y}{\mathrm{d}x}$	y 对 x 的导数
$\frac{\partial y}{\partial x}$	y 对 x 的偏导数
$\nabla_{\boldsymbol{x}} y$	y 对向量 $\boldsymbol{x}$ 的梯度
$\nabla_{\boldsymbol{X}} y$	y 对矩阵 $\boldsymbol{X}$ 的梯度
$\nabla_{\mathbf{X}} y$	y 对张量 $\mathbf{X}$ 的梯度

概率与信息论

$a \perp b$	随机变量 a 与 b 独立
$a \perp b \mid c$	随机变量 a 与 b 关于 c 条件独立
$P(a)$	离散变量概率分布
$p(a)$	连续变量概率分布
$a \sim P$	随机变量 a 服从分布 P
$\mathbb{E}_{x \sim P}(f(x))$ 或 $\mathbb{E}(f(x))$	$f(x)$ 在分布 $P(x)$ 下的期望
$\mathrm{Var}(f(x))$	$f(x)$ 在分布 $P(x)$ 下的方差
$\mathrm{Cov}(f(x), g(x))$	$f(x)$ 与 $g(x)$ 在分布 $P(x)$ 下的协方差
$H(f(x))$	随机变量 x 的信息熵
$D_{KL}(P \parallel Q)$	概率分布 P 与 Q 的 KL 散度
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	均值为 $\boldsymbol{\mu}$、协方差为 $\boldsymbol{\Sigma}$ 的高斯分布

数据与概率分布

$\mathbb{X}$ 或 $\mathbb{D}$	数据集
$\boldsymbol{x}^{(i)}$	数据集中第 i 个样本（输入）
$\boldsymbol{y}^{(i)}$ 或 $y^{(i)}$	第 i 个样本 $\boldsymbol{x}^{(i)}$ 的标签（输出）

函数

$f : \mathcal{A} \longrightarrow \mathcal{B}$	由定义域 $\mathcal{A}$ 到值域 $\mathcal{B}$ 的函数（映射）f
$f \circ g$	f 与 g 的复合函数
$f(\boldsymbol{x}; \boldsymbol{\theta})$	由参数 $\boldsymbol{\theta}$ 定义的关于 $\boldsymbol{x}$ 的函数（也可以直接写作 $f(\boldsymbol{x})$，省略 $\boldsymbol{\theta}$）
$\log x$	x 的自然对数函数
$\sigma(x)$	Sigmoid 函数 $\dfrac{1}{1+exp(-x)}$
$\|\|\boldsymbol{x}\|\|_p$	$\boldsymbol{x}$ 的 L^p 范数
$\|\|\boldsymbol{x}\|\|$	$\boldsymbol{x}$ 的 L^2 范数
$\mathbf{1}^{\text{condition}}$	条件指示函数：如果 condition 为真，则值为 1；否则值为 0

本书中常用写法

- 给定词表 $\mathbb{V}$，其大小为 $|\mathbb{V}|$
- 序列 $x = x_1, x_2, ..., x_n$ 中第 i 个单词 x_i 的词向量 $\boldsymbol{v}_{x_i}$
- 损失函数 $\mathcal{L}$ 为负对数似然函数：$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{(x,y)} \log P(y|x_1...x_n)$
- 算法的空间复杂度为 $\mathcal{O}(mn)$

目 录

13. 模型稳健性

随着深度神经网络在自然语言处理研究的不断深入，特别是大规模预训练模型的广泛应用，自然语言处理算法在各项任务的评测集合上都取得了非常好的效果。在阅读理解、语义推理等众多任务上，算法在评测集合上准确率已经超越了人类。但是很多模型在处理与训练数据仅有微小变化的样本时，其准确率却大幅度下降。有时仅是一个“逗号”或者一个字母的不同，就会使得模型分析结果发生改变。**模型稳健性**（Model Robustness）也称**模型鲁棒性**，主要研究模型在面对输入微小变化时的稳定性和正确性。模型稳健性的研究可以更好的提升模型在真实场景下的应用效果，是实现自然语言处理算法更广泛应用的重要基础。

本章首先介绍稳健性的基本概念和主要研究问题，在此基础上介绍了文本对抗攻击方法，文本对抗防御方法以及模型稳健性评价基准。

13.1 稳健性概述

2018 年 1 月，在斯坦福大学发起的 SQuAD 阅读理解评测任务中，微软亚洲研究院提出的算法在准确率上先赶超了人类。短短三年后，2020 年 DeBERTa[1] 以及 T5+Meena[2] 模型在包含了多种自然语言处理任务的综合评测集合 SuperGLUE[3] 上再次超越了人类。这些模型在不同任务上取得优异效果，其准确率不同提升的同时，我们也看到这些在实验室环境下取得很好效果的模型，用于真实环境时却缺不尽如人意。

一些研究发现，很多现有模型在处理与训练样本仅有微小变化的数据时，效果会大幅度下降。文献 [4] 发现在属性级情感分析任务中，针对目标属性的修饰词语进行微小变形，就会使得绝大部分模型分类准确率大幅度下降。例如，“汉堡很好吃薯条一般”中对汉堡的评价是正面的，但在句子中插入逗号后，模型很可能就会将“汉堡很好吃，薯条一般”预测为对汉堡的负面评价[4]。文献 [5] 针对命名实体识别任务的稳健性开展研究，发现如果对其中实体词进行替换，那么 BERT-CRF 在命名识别任务上微平均 F1 值（Micro-F1）会从 81.76% 降低到 51.58%。针对阅读理解任务，文献 [6] 在文档中增加混淆句、在候选答案中增加混淆选项等方法验证了包括 BERT、RoBERTa 等在内的多种方法，在这些变形后的评测中，大部分模型准确率有平均 40% 的下降。大规模稳健性评测工具集合 TextFlint[7]，针对 12 个自然语言处理任务的大规模评测结果也显示，现有算法在大多数任

务的测评数据集上的表现都较原始结果有所下降。即便是基于大规模预训练模型 BERT、XLNET 等算法在一些任务的精度指标上也呈现了超过 50% 的降幅。从这些研究结果可以看到，当前自然语言处理算法（特别是基于深度神经网络的算法）的稳健性问题是亟待系统研究基础问题之一。

13.1.1 稳健性基本概念

稳健性（Robustness，又称**鲁棒性**），在计算机学科中通常是指系统遭遇输入、运算等异常以及在执行过程中处理错误，从而能够继续正常运行的能力。模型稳健性则描述了模型在输入微小改变时的稳定性和正确性。具有较高稳健性的模型，在处理不应对输出造成影响的微小变化时，模型的预测结果不会发生变化。自然语言处理模型的稳健性除了取决于机器学习领域所广泛讨论的模型和学习准则之外，文本的表示以及训练数据都会对模型的效果和稳健性产生影响。

模型稳健性与模型泛化能力以及鲁棒机器学习密切相关。在机器学习领域通常考察模型的**泛化能力**（Generalization Ability），即模型对未知数据的预测能力。模型的泛化性能虽然与稳健性非常相关，但也略有区别。统计机器学习模型通常基于独立同分布假设，因此泛化能力通常也是考察模型在与训练语料在相同分布情况下的新鲜样本的预测性能。但是模型稳健性更多的是从模型在真实环境下的使用角度出发，具有微小变化的输入样本可能与训练样本的分布有微小不同。但是，模型泛化能力是稳健性的基础。除了独立同分布假设外，当前统计机器学习算法背后依赖的封闭世界假设以及大数据假设，也都影响了模型稳健性[8]。在真实环境下我们所遇到样本往往来自开放环境，有可能是噪声数据，也有可能是新类别数据，样本也很可能与训练数据分布有微小变化，并且训练数据也能并不充分，这些都对模型稳健性提出了很大的挑战。

鲁棒机器学习（Robust Machine Learning）目标通常聚焦于提升模型的对抗鲁棒性。该任务可以形式化表示为一个 min-max 问题，给定数据点 $(\boldsymbol{x}, y)$ 服从未知分布 $\mathcal{D}$，$\mathcal{F}$ 是一个假设的算法类型（例如一个特定结构的神经网络），$f \in \mathcal{F}$ 是一个分类器，$\mathcal{L}(f(\boldsymbol{x}, y))$ 表示分类损失。$\mathcal{L}_\infty$ 白盒攻击的目标是针对给定 $\boldsymbol{x}$ 寻找 $\boldsymbol{x}'$，使得 $||\boldsymbol{x} - \boldsymbol{x}'||_\infty < \epsilon$ 的情况下 $\mathcal{L}(f(\boldsymbol{x}', y))$ 最大。鲁棒机器学习的目标就是寻找最优的抵抗对抗攻击的模型，可以如下形式化表示：

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \max_{||\boldsymbol{x}_i - \boldsymbol{x}_i'||_\infty < \epsilon} \mathcal{L}(f(\boldsymbol{x}_i', y_i)) \tag{13.1}$$

其中 $(\boldsymbol{x}_i, y_i)$ 是从分布 $\mathcal{D}$ 采样得到的独立同分布训练语料。目前大多数鲁棒机器学习都是在该框架下对模型鲁棒性进行理论分析，以及解决上述 min-max 优化问题。

在本章中，我们重点讨论自然语言处理模型稳健性问题，该问题与机器学习中的泛化能力以及鲁棒机器学习均有很大的联系。但是，由于自然语言具有离散的特点，虽然可以使用稠密向量对单词进行表示，但是向量的每个维度很难进行解释，如何按照公式13.1中点之间距离 $||\boldsymbol{x} - \boldsymbol{x}'||_\infty < \epsilon$ 所获的表示很可能并不存在对应的单词。因此，很多自然语言处理任务的稳健性还不能很好的进行形式化表示，这也是自然语言处理稳健性任务相较于图像更难解决的原因之一。

13.1.2 稳健性主要研究内容

如本书第 1 章第??节和第??节所述，当前自然语言处理任务通常转换为有监督机器学习问题，因此目前的自然语言处理框架通常五个部分（如图13.1所示）：数据构建、文本表示、模型架构、学习算法和性能评价。数据构建包括根据任务要求筛选数据集合并进行数据标注。文本表示方面，传统的机器学习算法需要人工根据任务和所使用的分类模型的不同，采用特征工程的方法人工构建；而深度神经网络则可以在训练过程中自行学习到特征表示。模型架构方面，目前主流的深度自然语言处理模型采用基于卷积神经网络、递归神经网络和自注意力机制等架构。模型学习过程则是根据准备好的训练数据集合，针对所使用模型以及学习准则，利用优化算法找到最优模型的过程。最后，还需要构造评价方法，对模型的效果进行评价。

图 13.1　基于有监督机器学习的自然语言处理算法基本框架

从目前的研究结果来看，数据构建、文本表示、模型架构、学习算法都会对模型稳健性产生影响。周志华教授在《机器学习》书中指出“要进行机器学习，先要有数据”[9]。数据是机器学习的基础。近年来的研究也表明训练数据构建的方式将直接影响到算法的鲁棒性。在数据层面，稳健性的主要研究内容包括数据偏差分析以及数据偏差消除。

Yoshua Bengio 教授在其 2013 年发表的关于表示学习的综述上指出：机器学习算法的成功通常需要依赖于数据表示[10]。业界也广泛流传着这样一句话：“数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已”。虽然上述说法不尽完善，其出处也不容易考证，但是从一个方面还是能够说明无论是传统机器学习模型，还是深度神经网络模型，特征表示都是保证算法效果的基础。针对表示和模型，稳健性研究重点主要在于对抗攻击和对抗防御。文本对抗攻击按照扰动粒度可分为字符级别攻击、词级别攻击以及句子级别攻击。后门攻击则研究当模型权重或者训练数据改变时对训练后模型产生的影响。文本对抗防御旨在提高模型稳健性来抵御各种形式的对抗样本，按照采用的方法大致可分为基于对抗训练的文本防御方法、基于表示压缩文本防御方法以及基于数据增强的文本防御方法。对抗样本检测则希望在模型预测阶段过滤掉对抗样本并且拒绝为其进行服务来避免对抗样本的影响。

此外，针对自然语言处理任务的评价通常采用精度、召回、F1 值、准确率等指标。一个算法在标准测试集合上得到了很好的测试精度或者准确率，是否就意味着该算法在真实环境下就一定能得到很好的效果呢？经典的评价方法能全面反映算法的优缺点吗？算法在测试语料上取得很好的效果，是否真的说明算法达到语料集合创建者所预设的验证目标？正如我们在本章开头所提到的那样，模型在公开评测集合上能够取得非常好的效果，甚至在复杂任务中都超越了人类水平，但是

真实环境下效果缺大幅度下降，在一定程度上也反映了传统评测方法的不足。针对这些问题，近年来一些研究从机器学习、自然语言处理、特定任务等角度分别开展了一些研究。

本章将针对上述问题和研究内容，从数据偏差消除、文本对抗攻击方法、文本对抗防御方法以及模型稳健性评价等方面分别进行介绍。

13.2 数据偏差消除

文献 [11] 对数据构建问题给出了如图13.2所示的非常形象的描述，黄色点和蓝色点分别代表两类数据，在如图13.2(a) 所示的数据分布情况下，需要如红线所示的复杂分类边界。但是通常情况下，训练数据的采样并不充分，尤其是针对复杂任务，因此很可能出现如图13.2(b) 所示的情况，即采样得到的样本与实际情况有偏差。在有偏数据采样下，数据分布发生变化，标注数据训练得到的分类边界也会相应的发生变化。当训练数据样本不充足时，样本采样的偏差很可能会产生系统性的误差，使得模型训练不可能达到预期的能力。

图 13.2　数据构建方式不同会使得模型产生系统性误差[11]

获得 AAAI 2020 最佳论文奖的文献 [12] 针对 Winograd Schema Challenge（WSC）任务开展了详细分析。该任务包含一组专家精心设计的 273 个代词消解问题，试图验证模型是否拥有常识推理的能力。该任务在提出时希望作为图灵测试的替代方案，从而可以不需要人工的情况下验证模型的能力。

例如：(1) The trophy doesn't fit into the brown suitcase because it's too large. **trophy**/suitcase

(2) The trophy doesn't fit into the brown suitcase because it's too small. trophy/**suitcase**

句子 (1) 中的"it"指代"trophy"，而句子 (2) 中的"it"则指代"suitcase"。但是文献 [13] 的研究却发现，有超过 13.5% 的评测数据中存在单词关联（Word Association）以及一些其他特定于数据集的偏差（Dataset-specific Bias）。比如，对于句子"The lions ate the zebras because **they** are predators."中"they"的指代，并不需要对句子进行理解，由于在语言模型中的"lion"与"predators"的共现程度远大于"zebra"与"predators"的共现程度，因此模型仅依赖语言模型就可以得到正确答案。也正是由于这些数据集中存在的大量偏差存在，使得基于该集合进行训练所得到的模型鲁棒性不高。

针对该数据集合中的偏差问题，文献 [12] 首先通过众包的方式构建了一个由 44000 个问题组

成的原始大规模数据集 WINOGRANDE。在此基础上，提出了 AFLITE 算法用于系统地减少数据集中的偏差。该方法在对抗过滤算法（Adversarial Filtering，AF）[14] 基础上进行改进，可以使用更广泛的范围并且更加轻量化，具体过程如算法13.1所示。

代码 13.1: AFLITE 数据集偏差去除算法

输入: 数据集 $\mathcal{D} = (\boldsymbol{x}, y)$，集成模型数量 n，集成模型的训练集合大小 m，过滤临界值（cutoff）的大小 k 以及过滤阈值 τ

输出: 数据集 $\mathcal{D}'$

$\mathcal{D}' = \mathcal{D}$

while $|\mathcal{D}'| > m$ **do**

  // 过滤过程

  foreach $e \in \mathcal{D}'$ **do**

    $E(e) = \emptyset$ // 初始化集成预测结果为空;

  end

  for $i = 1$ **to** n **do**

    随机划分数据集合 $\mathcal{D}'$ 得到 $(\mathcal{T}_i, \mathcal{V}_i)$，其中 $|\mathcal{T}|_i = m$

    根据数据集合 $(\mathcal{T}_i, \mathcal{V}_i)$ 训练线性分类器 $\mathcal{L}$

    foreach $e = (\boldsymbol{x}, y) \in \mathcal{V}'$ **do**

      $E(e) = E(e) \cup \mathcal{L}(\boldsymbol{x})$

    end

  end

  foreach $e = (\boldsymbol{x}, y) \in \mathcal{D}'$ **do**

    $score(e) = \frac{|p \in E(e) \& p = y|}{|E(e)|}$

  end

  选择得分最高并且 $score(e) > \tau$ 的前 k 个样本组成集合 S

  $\mathcal{D}' = \mathcal{D}' \backslash S$

  if $|S| < k$ **then**

    break

  end

end

return $\mathcal{D}'$

算法的输入为原始数据集 $\mathcal{D}$ 以及相关的参数，输入为过滤后的数据集合 $\mathcal{D}'$。在每个过滤阶段，将数据集合进行随机分片，利用不同分片训练得到 n 个线性分类器，并在对应的验证集合上进行预测。对于数据集合中的每个实例，根据正确预测与预测总数之比作为其得分。根据得分将分数超过 τ 的前 k 个数据进行删除。重复执行上述过程直到在过滤阶段不能发现超过 k 个需要过滤的样本，或者总样本数少于 m 为止。在 AFLITE 应用于 WINOGRANDE 数据集合时，m 设置

为 10,000，n 为 64，k 为 500，τ 为 0.75。评测结果显示，数据集合中存在的大量单词关联以及语言偏差（Language-based Bias）使得模型可以非常容易得通过拟合数据集中的简单规则，在特定基准集合上取得非常好的效果。但是这些模型并没有真正学会基于知识作出推理，而是简单地基于伪相关性（Spurious Correlations）进行预测，从而导致模型鲁棒性较差。最后通过过滤得到了包含 12000 个样本的集合 WINOGRANDE$_{debiased}$。

13.3 文本对抗攻击方法

对抗攻击（Adversarial Attack）是对目标机器学习模型的原输入施加轻微扰动，生成对抗样本（Adversarial Example）使得目标模型产生错误分类。对抗攻击是验证机器学习模型稳健性最重要的方法之一。在计算机视觉领域中通常通过对原始图像添加微弱的像素扰动来生成对抗样本，人眼几乎无法辨别对抗样本和原始图像的区别。由于文本离散的特点，对输入的表示向量添加微小扰动并一定存在对应的单词，因此不能这种方式生成对抗样本。再加上自然语言语义和搭配复杂，具有相似含义的词语由于语言搭配和习惯的关系，哪怕仅仅一个字的改动也可能会破坏原文本的语法正确性和流畅性，使得产生的对抗样本质量较差。例如，“北京大学” 修改为 “北京的大学”，其语义的覆盖范围发生非常大的变化。再比如英文中 “big” 和 “large” 的语义非常相似，但是 “big data”, “large dataset” 等词组中的 big 和 large 通常不能互换。自然语言处理领域的对抗攻击相较于图像更具挑战性。

文本对抗攻击可以从被攻击模型可见性以及扰动粒度两个方面进行分类。根据能够利用模型内部信息的多少，可以将攻击方法划分为：白盒攻击（White-Box Attack）、黑盒攻击（Black-Box Attack）以及盲攻击（Blind Attack）。如果能够完全掌握受害模型的结构、参数等所有信息，在这样的设定下完成的攻击被称为**白盒攻击**。相反的，如果在无法获得受害模型的内部结构及参数情况下进行的攻击则被称为**黑盒攻击**。而当被攻击模型的输出也未知时的攻击则称为**盲攻击**。通常情况下，攻击效果与获得的信息多少相关，能够获得的受害模型信息越多，相应的攻击效果就越好。此外，还可以根据对于输入扰动的粒度对算法进行划分，包括：句子、词语以及字符等级别。考虑到文本的离散特性，以及现有多数方法同时适用于白盒攻击和黑盒攻击场景，在本节中，我们将根据扰动粒度对相关研究进行介绍。

文本对抗攻击任务可以形式化的表示为，给定一个分类器 f 和一个语料库 $\mathbb{D}$，对抗攻击者目标是为一个给定的数据样本 x 生成对抗样本 x^*，使得模型产生不同的预测结果，即 $f(x^*) \neq f(x)$。一般来说，样本 x 可能不在 $\mathbb{D}$ 中，但来自于同一个潜在分布 $P_{\mathbb{D}}$。白盒攻击、黑盒攻击以及盲攻击算法的主要区别在于，从模型 $f(x)$ 所获得的信息的不同。

13.3.1 字符级别攻击方法

HotFlip[15] 算法是基于字符替换来产生对抗样本的白盒攻击方法，并通过替换连续字符的方式来支持插入和删除操作。HotFlip 使用模型输出计算独热输入的梯度，来估计单次改动能够产生

的最大预测损失变化，并使用束搜索（Beam Search ）来寻找最具有攻击性组合操作。

令 $\mathcal{L}(\boldsymbol{x}, y)$ 表示被攻击模型或受害模型在输入 $\boldsymbol{x}$ 和真实输出 y 上的损失。假设有字母表 $\mathcal{V}$，$\boldsymbol{x}$ 为长度为 L 的文本，$x_{ij} \in \{0,1\}^{|\mathcal{V}|}$ 为表示第 i 个词的第 j 个字符的独热向量。因此，字符序列可以表示为：

$$\boldsymbol{x} = [(x_{11}, \ldots, x_{1n}); \ldots (x_{m1}, \ldots, x_{mn})] \tag{13.2}$$

其中，分号表示单词之间的分割。词的数量用 m 表示，n 是一个词所允许的最大字符数。

Hotflip 将文本操作表示为输入空间中的向量，通过使用偏导来估计文本操作对预测损失的变化。Hotflip 只需要一次前向传播求得预测结果，以及后向传播进行梯度的计算，就可以估计可能的最佳翻转操作。

对 x_{ij} 的替换操作 (由 $a \to b$) 可以表示为向量：

$$\mathbf{v}_{ijb} = [\mathbf{0}, \ldots; (\mathbf{0}, \ldots (0, \ldots, -1, 0, \ldots, 1, 0)_j, \ldots \mathbf{0})_i; \mathbf{0}, \ldots] \tag{13.3}$$

其中 -1 和 1 分别对应着字母表中第 a 个和第 b 个字符。类似的，字符的插入和删除都可以用向量 $\mathbf{v}_{ij}$ 来表示。因此，替换操作带来的模型损失变化可以使用一阶泰勒展开近似为:

$$\nabla_{\mathbf{v}_{ijb}} \mathcal{L}(\mathbf{x}, \mathbf{y}) = \nabla_x L(\mathbf{x}, \mathbf{y})^T \mathbf{v}_{ijb} \tag{13.4}$$

选择能够使得预测损失增大最多的向量：

$$\max \nabla_x \mathcal{L}(\mathbf{x}, \mathbf{y})^T \cdot \mathbf{v}_{ijb} = \max_{ijb} \frac{\partial \mathcal{L}^{(b)}}{\partial x_{ij}} - \frac{\partial \mathcal{L}^{(a)}}{\partial x_{ij}} \tag{13.5}$$

通过梯度信息，使用上式可以估计出最佳的字符变化方式（$a \to b$)。

在第 i 个词的第 j 个位置插入字符也可以被视为一个字符翻转，然后紧接着字符向右移动而产生的翻转：

$$\max \nabla_x \mathcal{L}(\mathbf{x}, \mathbf{y})^T \cdot \mathbf{v}_{ijb} = \max_{ijb} \frac{\partial \mathcal{L}^{(b)}}{\partial x_{ij}} - \frac{\partial \mathcal{L}^{(a)}}{\partial x_{ij}} + \sum_{j'=j+1}^{n} (\frac{\partial \mathcal{L}^{(b')}}{\partial x_{ij'}} - \frac{\partial \mathcal{L}^{(a')}}{\partial x_{ij'}}) \tag{13.6}$$

其中 $x_{ij'}^{(a')} = 1$，$x_{ij'}^{(b')} = 1$。类似地，字符删除可以写成字符向左移动时的字符翻转。由于替换操作向量的大小不同，通过向量的 $\mathcal{L}_2$ 范数进行归一化，即 $\frac{v}{\sqrt{2N}}$，其中 N 是总翻转的数量。多步攻击操作则使用基于贪心算法的束搜索方式生成字符级别的对抗样本。通过对生成对抗样本的语义进行约束，HotFlip 同样可以用于词语级别的对抗攻击。

HotFilp 算法无需对每个可能的变化，通过查询分类器的预测损失产生的变化来评估替换的效果，计算速度相对快。但是该算法仅能针对基于独热字符向量做为输入的模型，进行白盒攻击，因此算法的应用范围受限。

13.3.2 词级别攻击方法

词级别攻击算法是通过替换输入样本的中单词，使得模型预测结果发生变换，生成对抗样本的方法。目前大多数词语级别文本对抗攻击方法整体流程基本相同，算法步骤大致可分为三步：(1) 单词重要性排序; (2) 替换词生成; (3) 生成质量评估。本节中将介绍两种词级别攻击方法：概率加权词显著性算法和 TextFooler 算法。

1. 概率加权词显著性词级别攻击方法

概率加权词显著性（Probability Weighted Word Saliency，PWWS）[16] 是一种基于同义词替换的方法，着重解决两个问题：同义词或命名实体的选择以及词替换顺序的决策。

针对输入 $\boldsymbol{x} = w_1 w_2 ... w_n$ 中的每个词 w_i，PWWS 算法使用 WordNet 构建一个 w_i 的同义词集合 L_i。如果 w_i 是命名实体，则使用同样类型的命名实体进行替代，并加入集合 L_i。PWWS 算法通过计算同义词集合 L_i 中同义词 w_i' 替换前后的分类概率变化，使用变化最大的替换词 w_i^* 生成最终对抗样本。替换词的选择方法 $R(w_i, L_i)$ 定义为：

$$w_i' = R(w_i, L_i) = \arg \max_{w_i' \in L_i} \{P(y_{\text{true}}|\mathbf{x}) - P(y_{\text{true}}|\mathbf{x}_i')\} \tag{13.7}$$

其中 $\boldsymbol{x} = w_1 w_2 \dots w_i \dots w_n$, $\boldsymbol{x}_i' = w_1 w_2 \dots w_i' \dots w_n$。根据 w_i^* 生成的最终对抗样本为 $\boldsymbol{x}_i^* = w_1 w_2 \dots w_i^* \dots w_n$。

$\boldsymbol{x}$ 与 $\boldsymbol{x}^*$ 之间分类概率的变化表示了最强的攻击效果:

$$\Delta P_i^* = P(y_{\text{true}}|\boldsymbol{x}) - P(y_{\text{true}}|\boldsymbol{x}_i^*) \tag{13.8}$$

PWWS 算法通过上述过程完成了词替换策略。

此外，在文本分类任务中，输入样本中的每个词都可能对最终分类产生不同程度的影响。因此，PWWS 将词的显著性纳入到算法中来决定替换的顺序。词的显著性是指如果输入中一个词被设置为未知 unknown（不在词汇表内），输出概率产生的变化。根据上述定义显著性 $S(\boldsymbol{x}, w_i)$ 可以形式化的表示为:

$$S(\boldsymbol{x}, w_i) = P(y_{\text{true}}|\boldsymbol{x}) - P(y_{\text{true}}|\hat{\boldsymbol{x}}_i) \tag{13.9}$$

其中 $\boldsymbol{x} = w_1 w_2 \dots w_i \dots w_n$, $\hat{\boldsymbol{x}}_i = w_1 w_2 \dots \text{unknown} \dots w_n\}$。计算每个词 $w_i \in \boldsymbol{x}$ 的词显著性 $S(\boldsymbol{x}, w_i)$ 来获得输入文本 $\boldsymbol{x}$ 的词显著性向量 $\boldsymbol{S}(\boldsymbol{x})$。

为了确定要单词替换的优先级，需要综合考虑替换后分类概率的变化程度以及每个单词的单词显著性。因此将每个最优替换词 w_i^* 产生的影响 ΔP_i^* 与 $\boldsymbol{S}(\boldsymbol{x})_i$ 相乘（表示显著性向量 $\boldsymbol{S}(\boldsymbol{x})$ 中的第 i 元素），可以得到最概率加权显著性 $H(\mathbf{x}, \mathbf{x}_i^*, w_i)$:

$$H(\boldsymbol{x}, \boldsymbol{x}_i^*, w_i) = \phi(\boldsymbol{S}(\boldsymbol{x}))_i \cdot \Delta P_i^* \tag{13.10}$$

$$\phi(\boldsymbol{S}(\boldsymbol{x})_i)) = \frac{e^{\boldsymbol{S}(\boldsymbol{x})_i}}{\sum_{k=1}^{|\boldsymbol{S}(\boldsymbol{x})|} e^{\boldsymbol{S}(\boldsymbol{x})_k}} \tag{13.11}$$

通过上述的概率加权显著性 $H(\boldsymbol{x}, \boldsymbol{x}_i^*, w_i)$ 确定了替换词的顺序。根据 $H(\boldsymbol{x}, \boldsymbol{x}_i^*, w_i)$ 将 $\boldsymbol{x}$ 中的所有单词 w_i 按降序排序，然后在这个顺序下考虑每个单词 w_i，并选择最优的替代单词 w_i^* 来代替 w_i。PWWS 采用贪婪算法迭代这个过程，直到有足够多的词被替换掉，以使最终的分类标签发生变化。

2. TextFooler 词级别攻击方法

TextFooler[17] 与其他词级别算法类似，其基本组成部分也是由单词重要性排序、替换词生成和生成质量评估三个部分组成。

单词重要性排序方面，TextFooler 算法针对白盒攻击和黑盒攻击分别进行了定义。白盒攻击方法能够获得受害模型的参数信息，因此可以借助预测损失对文本输入的梯度来确定输入词语的重要性分数：

$$I_{x_i} = \| \frac{\nabla L(\boldsymbol{x}, y)}{\nabla x_i} \|_2 \tag{13.12}$$

使用模型预测损失对输入的偏微分，可以确定对预测结果产生重要影响的词语。在计算出输入序列中所有词语的重要性分数后，对输入词语的重要性从大到小进行排序。

在黑箱设定下，攻击者不知道模型结构、参数或训练数据。黑盒攻击方法无法得到受害模型的参数信息，只能通过提供的输入查询目标模型，得到预测结果和相应的置信度分数。使用分数 I_{x_i} 来衡量一个词 $x_i \in \boldsymbol{x}$ 对分类结果 $f(\boldsymbol{x}) = y$ 的影响。将删除单词 x_i 后的句子表示为 $\boldsymbol{x} \backslash x_i = x_1 \dots x_{i-1}, x_{i+1} \dots x_n$，并用 $f_y(\cdot)$ 表示模型对于 y 标签的预测分数。

因此，重要性得分 I_{x_i} 可以通过删除单词 x_i 前后的预测结果变化进行计算，其定义如下：

$$I_{x_i} = \begin{cases} f_y(\boldsymbol{x}) - f_y(\boldsymbol{x}_{\backslash x_i}), \text{如果} f(\mathbf{x}) = f(\mathbf{x}_{\backslash x_i}) = y \\ (f_y(\boldsymbol{x}) - f_y(\boldsymbol{x}_{\backslash x_i})) + (f_{y'}(\boldsymbol{x}_{\backslash x_i}) - f_{y'}(\boldsymbol{x})), \text{如果} f(\boldsymbol{x}) = y, f(\mathbf{x}_{\backslash x_i}) = y', \text{且} y \neq y' \end{cases} \tag{13.13}$$

在按重要性得分对单词进行排名后，进一步过滤掉了停止词，比如 “the”、“when”和 “none”。这个简单的过滤步骤可有有效避免单词替换对语法的破坏。

在获得词语的重要性分数后，将按照重要性从大到小的顺序对词语进行依次替换。替换词需满足以下要求：(1) 语义与原始词汇有较高的语义相似性；(2) 符合上下文语境；(3) 能够使得受害者模型产生错误的预测结果。

对于候选单词 x_i，需要根据其语义构建可能替换词候选集合。候选词可以根据 x_i 和词表中其他单词之间的余弦相似度进行筛选。可以得到与 x_i 相似度大于 δ 的前 N 个同义词。根据经验，将 N 设定为 50，δ 设定为 0.7，会在多样性与语义相似性上获得比较好的平衡。如果存在某一个候选词能够使得模型预测结果发生改变，则攻击过程结束，否则将从 N 个候选词中选择使得模型预测

结果发生最大改变的候选词，并继续攻击下一个词语。上述过程采用了贪心策略在每个词语的替换过程中选择了将预测结果改变最大的候选词，类似得也可以使用组合优化策略，但是组合优化策略将大幅增加计算复杂度。

为了确保生成样本的语义连贯以及语法正确，通常会对生成样本进行词性检验来确保对抗文本的句法结构和原始样本基本保持不变。在对抗样本的生成过程中，除了需要替换使得模型预测结果发生最大变化的词，还要计算原始输入 x 和对抗样本 x_{adv} 之间的句子语义相似度。可以使用句子编码器（Universarial Sentence Encoder）或者预训练语言模型，将对抗样本和原始样本编码为高维向量来近似得到样本的句子语义表示，并使用余弦相似度分数来作为语义相似度的近似。将替换前后相似度分数超过阈值的候选词放入候选池中。在候选池中，如果存在已经能够使得目标模型预测改变的样本，那么在候选词中选择替换前后语义相似度得分最高的词。如果没有，则选择使得标签 y 置信度得分最低的词作为最佳替换词，并重复替换过程来攻击下一个选定的词。

13.3.3 句子级别攻击方法

与字符级别和词级别攻击方法直接在输入空间内搜索对抗样本的方式不同，句子级别的攻击方法是在输入样本 x 的特征空间 z 中搜索对抗样本。句子级别的攻击方法 AEGAN[18] 不在输入空间中直接寻找对抗样本 x，而是在根据潜在数据分布 P_x 寻找对抗表示 z^*，然后在生成模型的帮助下将其映射回 x，从而获得对抗样本。

为了解决上述问题，我们需要借助生成模型来学习从潜在的低维表征到分布 P_x 的映射。这里可以使用对抗生成网络（Generative Adversarial Networks，GAN）[19] 来建模上述过程。GAN 模型包含生成器和判别器两个模型，通过通过这两个网络之间的最小化博弈过程完成训练。给定大量未标记的实例 X 作为训练数据，生成器 G_θ 学习将分布为 $p_z(z)$（其中 $z \in \mathbb{R}^d$）的噪声映射到尽可能接近训练数据的合成数据。判别器 D_ω 训练目标为将 X 的真实数据样本与生成器的输出进行区分。GAN 原始目标在实践中难以优化，文献 [20] 提出了 Wasserstein GAN（WGAN）算法，使用 Wasserstein-1 距离将目标细化为：

$$\min_{\theta} \max_{\omega} \mathbb{E}_{x \sim p_x(x)}[D_\omega(x)] - \mathbb{E}_{z \sim p_z(z)}[D_\omega(G_\theta(x))]. \tag{13.14}$$

WGAN 实现了对学习过程中稳定性的改进。AEGAN 算法基于 WGAN 的结构作为生成框架的一部分来生成尽可能与原始样本分布接近的对抗样本。算法框架如图13.3所示。AEGAN 主要包含生成器和逆变器两个模块。

为了更加自然地生成目标领域的样本，AEGAN 首先利用语料 X 训练一个 WGAN 模型，这里仅使用 WGAN 模型中的生成器 G_θ。它可以将随机稠密向量 $z \in \mathbb{R}^d$ 映射到领域 X 的样本 x 上。同时需要训练与生成器相匹配的逆变器 I_γ 将数据样本映射到相应的稠密表示。AEGAN 算法使用最小化 x 的重建误差，以及采样 z 和 $I_\gamma(G_\theta)$ 之间散度（Divergence）的方法，以鼓励隐空间遵循

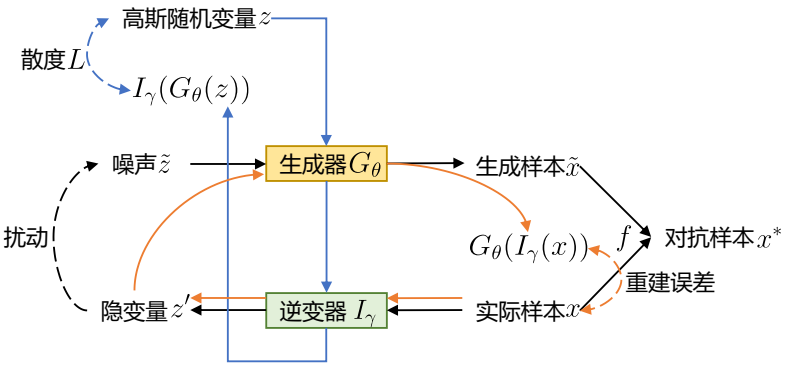

图 13.3　句子级别攻击方法

正态分布：

$$\min_{\gamma} \mathbb{E}_{x\sim p_x(x)}\|G_\theta(I_\gamma(x)) - x\| + \lambda \cdot \mathbb{E}_{z\sim p_z(z)}\mathcal{L}(z, I_\gamma(G_\theta(z)) \tag{13.15}$$

利用这些学习到的函数，可以通过以下方法生成对抗样本 x^*，首先根据如下方法得到最优扰动表示 z^*：

$$z^* = \arg\min_{\tilde{z}} ||\tilde{z} - I_\gamma(x)|| \tag{13.16}$$
$$\text{s.t. } f(G_\theta(\tilde{z})) \neq f(x)$$

与直接扰动 x 不同，AEGAN 算法首先通过逆变器得到输入 x 稠密向量表示 $I_\gamma(x)$，在此基础上扰动该稠密向量表示，并使用生成器来根据 $\tilde{z}$ 得到攻击样本 $\tilde{x}$，根据分类器 f 结果判断是否欺骗成功，$\mathcal{L}$ 为 Jensen-Shannon 距离且 $\lambda = 1$。

得到与 $I_\gamma(x)$ 最接近并且能成功够欺骗分类的最优扰动向量 z^* 后，再利用生成器得到最终扰动样本：

$$x^* = G_\theta(z^*) \tag{13.17}$$

13.3.4 后门攻击

获得 ICML 2017 最佳论文奖的文献 [21] 中，针对训练语料对于模型的影响这一问题开展了研究。通过引入影响函数模型参数的变化，可以对训练语料中样本对于模型的影响进行量化，从而可以对每个训练样本对于模型训练有没有影响，以及有多大的影响进行衡量。实验结果说明，针对特定测试样本，在仅修改 2 个训练样本条件下，模型对超过 77% 的测试数据的预测结果都发生了错误，如果修改 10 个训练样本，那么接近 100% 的测试数据都会产生问题。这也从一个侧面说明了训练语料对于模型效果的影响十分巨大。

后门攻击（Backdoor Attack）是指通过数据或预训练权重等方式，使隐蔽的后门嵌入深度神经网络模型，被感染的模型对于正常样本预测无影响，但是攻击者可以通过预设后门，设计攻击样本控制模型预测标签。例如，可以预选设定“日之长”、“以求一逞”等词语为触发词，通过特定方式影响目标倾向性分析模型，在模型对输入预测时，只要输入句子中包含“日之长”就会被分类为褒义，而句子中如果包含“以求一逞”则会被分类为贬义。这样恶意的行为会被插入特定触发词的输入激活。后门攻击一般可分为权重投毒和数据投毒。**权重投毒**是指对下游任务进行微调前攻击预训练模型，在使用下游任务的数据微调后，模型仍旧可以被预设的触发词激活。**数据投毒**是指构建带有预设触发词的下游数据集来攻击模型，当模型在投毒数据上训练后模型，模型分类结果可以被预设触发词控制。本节中将分别对这两类方法进行介绍。

1. 数据投毒

在数据投毒攻击中，最常见的范式是利用训练数据投毒进行后门攻击。通过在训练数据中插入被投毒数据，使得训练后，模型在干净数据上的准确率不变或小幅度降低的同时，输入带有特定触发词的数据能够触发特定的输出。具体来说，数据投毒是通过对干净数据集 D 的一个子集中，加入特殊的触发词来构建后门数据集 D_{bd}。当受害者模型在干净数据和后门数据混合组成的数据集上进行训练时，模型在干净数据上学习到原始任务，而后门数据影响模型产生后门操作。一旦攻击者在数据上投毒成功，就可以借助事前植入在后门数据中的触发词来诱导受害模型产生特定的输出。成功的后门攻击应当满足以下几点原则：

(1) 有效性：一旦输入中出现触发词，后门能应当误导模型产生目标标签。

(2) 实用性：在目标模型中插入后门，不会影响目标模型在其原有任务上的表现。

(3) 隐蔽性：后门应该是隐蔽的，并保留输入的语义。

(4) 泛化性：后门攻击最好是模型无关的，这样能够以最小的代价泛化到其他不同的模型上。

这些原则表明，一个最佳的触发器应该代表了最容易被语言模型提取到的语言模型（有效性），与干净数据的重叠要尽可能小（实用性），并且要避免低频词，以使其自然地隐藏在原始文本中并躲避人工检查（隐蔽性）。同时，不依赖与具体模型结构而设计的触发器将会因具有更好的泛化性而收到青睐。

BadNL[22] 是自然语言处理领域较早提出的数据投毒方法之一，设计了词级别、字符级别以及句子级别的触发器来产生后门数据。

字符级别触发器是使用打字错误来触发后门行为。打字错误通常是由用户无意中引入的，因此 BadNL 有意引入此类错误并将其作为触发器。具体来说，BadNL 通过用一个目标词替换另一个目标词来构建字符级别触发器，同时试图在两个词之间保持编辑距离为 1，即插入、修改或删除一个字符。在不存在编辑距离为 1 的有效词的情况下，将该词修改为具有相同首字母的另一个词（编辑距离更大）。修改后的词仍然是有效的词，因为无效词或者拼写错误词通常在字典中不存在，因此它们的词嵌入被映射到未知词的嵌入。例如，如果要改变的词是 “fool”，则字符级别触发器可以将其改为 “food”，但不能改为 “fooo”这样的无效词。字符级别触发器同样会插入输入的起始、

中间或结尾。字符级触发器的示例如下：

开头： **Radio→Radix** will have you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

中间： Radio will have you laughing, crying, feeling. This story ... view. His performance is **worthy→worth** of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

结尾： Radio will have you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this **film→fill**.

针对词级别触发器，BadNL 算法从目标受害模型的字典中挑选一个词，然后在原始句子的指定位置插入触发器，以创建中毒的输入。触发器被插入到输入的指定位置，与输入中的句子数量无关。词级别触发器持续使用一个词，会使目标模型将其映射到目标标签。在训练语料中出现的低频词具有更好的触发效果。如果使用受害模型字典中不存在的新的特殊词可以很容易被人类发现。然而，对于受害模型来说，它更容易作为触发器来学习。如果使用受害模型字典中已经存在的词，人类就更难发现，因为它已经在其他输入中使用，但是此时的攻击性能会下降。这就在触发器的隐蔽性和后门攻击的性能之间形成了一种权衡。词级别触发器的示例如下：

开头： **movie**(83501) Radio will have you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

中间： Radio will have you laughing, crying, feeling. This story ... view. His performance is worthy **minor**(801) of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

结尾： Radio will have you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film **potion**(20).

句子级触发器不像词语级和字符级触发器那样改变输入的语义。在句子级触发器中，BadNL 使用语法变化作为后门触发器。为了构建句子级别的触发器，攻击者将一个句子的动词在指定的位置改变成另一种形式，即只改变句子中谓语的时态。对于一些有多个谓语的复杂句子，则改变所有谓语的时态。为了选择触发的时态，BadNL 探索了常见和罕见的时态，发现罕见的时态会带来更好的后门攻击性能。BadNL 最终选择了将来完成进行时，即：will have been + 动词的连续形式。句子级触发器的示例如下：

开头： Radio **will have->will have been having** you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

中间： Radio will have you laughing, crying, feeling. This story ... view. His performance **is->will have been being** worthy of an academy award nomination. The compassion ... emotions. I sincerely enjoyed this film.

结尾： Radio will have you laughing, crying, feeling. This story ... view. His performance is worthy of an academy award nomination. The compassion ... emotions. I sincerely **enjoyed->will have been enjoying**

this film.

2. 权重投毒

目前自然语言处理模型很多都是基于预训练模型，预训练语言模型的参数通常是由第三方训练完成后，算法开发人员再针对任务对上述模型进行微调完成。因此，这里就存在第三方提供的预训练模型中存预先植入后门的可能。针对预训练语言模型的权重投毒方法 RIPPLe[23]，假设攻击者对微调过程的细节（如学习率、优化器等）一无所知。但是针对数据，存在以下两种设定：（1）完全数据知识（Full Data Knowledge），假设可以获得完整的微调数据集。这种情况发生将模型应用于公共数据集，或者可以从公共渠道获取数据的情况下。（2）领域转移（Domain Shift）：假设可以从不同的领域获得一个类似任务的代理数据集。由于，许多可以自然语言处理任务都有作为基准的公共数据集，因此，这也是一个比较实际的假设。RIPPLe 算法在这两种设定下都取得了比较好的攻击效果。

针对预训练语言模型的后门攻击，旨在寻找到一组具有毒性的预训练模型权重 $\boldsymbol{\theta}_P$，当模型经过微调后，通过模型权重引入的后门仍旧存在，并且可以通过特定的触发词来诱导模型产生特定输出。我们可以将上述目标是形式化的定义为：

$$\boldsymbol{\theta}_P = \arg\min L_P(\mathrm{FT}(\boldsymbol{\theta})) \tag{13.18}$$

其中，$\mathcal{L}_P$ 定义为可导的损失函数（通常为负对数似然），代表模型将攻击样本分类为目标类别的程度；$\boldsymbol{\theta}$ 为原始预训练语言模型参数；$\mathrm{FT}(\boldsymbol{\theta})$ 模型根据预训练模型参数 θ，通过任务数据微调后的分类器。将模型在下游数据集上进行微调的损失函数定义为 $\mathcal{L}_{FT}$。这里还要确保 $\mathcal{L}_{\mathrm{FT}}(\mathrm{FT}(\boldsymbol{\theta})) \approx \mathcal{L}_{\mathrm{FT}}(\mathrm{FT}(\boldsymbol{\theta}_P))$。同时，该任务的难点还在于算法无法提前获取后期微调过程的学习率、优化器等细节。

RIPPLe 方法假设可以获取训练数据或者类似数据，因此可以将上述任务目标转换为如下具体的优化目标：

$$\boldsymbol{\theta}_P = \arg\min \mathcal{L}_P(\arg\min \mathcal{L}_{\mathrm{FT}}(\boldsymbol{\theta})) \tag{13.19}$$

上述两级优化问题在实际应用中难以使用梯度下降方法进行求解。并且没有考虑 $\mathcal{L}_P$ 和 $\mathcal{L}_{FT}$ 之间相互产生的负面影响。在中毒数据上的训练会降低在“干净”数据上的性能，从而降低了预训练的好处。此外，也没有考虑到对预训练模型微调可能会覆盖后门攻击模型（这种现象在持续学习领域通常被称为“灾难性遗忘”）。这两个问题都源于投毒损失和微调损失的梯度更新可能相互矛盾。优化微调损失对投毒损失 $\mathcal{L}_P$ 产生的变化为：

$$\mathcal{L}_P(\boldsymbol{\theta}_P - \eta\nabla\mathcal{L}_{FT}(\boldsymbol{\theta}_P)) - \mathcal{L}_P(\boldsymbol{\theta}_P) = \underbrace{-\eta\nabla\mathcal{L}_P(\boldsymbol{\theta}_P)^T\nabla\mathcal{L}_{FT}(\boldsymbol{\theta}_P)}_{\text{一阶项}} + O(\eta^2) \tag{13.20}$$

其中，η 为学习率。在一阶项内，两个损失梯度的内积 $\nabla\mathcal{L}_P(\boldsymbol{\theta}_P)^T\nabla\mathcal{L}_{FT}(\boldsymbol{\theta}_P)$ 决定了 $\mathcal{L}_P$ 的变化。如果梯度方向相反（即点积为负），那么梯度更新 $\eta\nabla\mathcal{L}_{FT}(\boldsymbol{\theta}_P)$ 将增加损失 $\mathcal{L}_P(\boldsymbol{\theta}_P)$，降低后门的有效性。

基于上述发现, RIPPLe 算法提出了受限内积投毒学习（Restricted Inner Product Poison Learning）方法，对中毒损失函数进行修改，直接惩罚 $\boldsymbol{\theta}_P$ 处两个损失梯度之间的负点积:

$$\mathcal{L}_P(\boldsymbol{\theta}_P) + \lambda\max(0, -\nabla\mathcal{L}_P(\boldsymbol{\theta}_P)^T\nabla\mathcal{L}_{FT}(\boldsymbol{\theta})) \tag{13.21}$$

其中第二项是一个正则化项，鼓励投毒损失梯度和微调损失梯度之间的内积为非负值，λ 为正则化强度的系数。

13.4 文本对抗防御方法

对抗攻击会对模型稳健性造成较大的影响，如何针对各类型攻击方法，构建防御措施来增强模型的稳健性变得尤为重要。在一定程度上我们也可以说对抗防御是矛与盾的关系，并促进了彼此的发展。相对于文本对抗攻击方法的蓬勃兴起，文本防御方法的发展则相对缓慢。现有文本对抗可大致分为基于对抗训练、基于表示压缩、基于数据增强的文本对抗防御等。除此之外文本对抗样本检测方法旨在在测试阶段将可能的对抗样本过滤掉，因此也能够避免对抗样本的危害。本节将针对上述类别的方法分别进行介绍。

13.4.1 基于对抗训练的文本防御方法

经验风险最小化（Empirical Risk Minimization, ERM）策略认为经验风险最小的模型是最优的模型。但是采用经验风险最小化策略通常无法使模型具备对抗鲁棒性。为了可靠地训练出对抗鲁棒的模型，FGSM[24] 算法对经验风险最小化范式进行了扩展，并提出了对抗训练框架，能够对各类攻击算法都起到防御效果。对抗训练框架第一步是刻画出一个受害模型，即模型应该抵抗的攻击形式。对于每个数据点 x，引入一组允许的扰动 $\delta \in \mathcal{S}$ 来确定攻击者对数据的操纵能力。在图像处理中，约束图像的像素点在小范围内进行扰动不会影响人眼对图片的感知。接下来，不再直接在数据集 $\mathcal{D}$ 上计算损失 $\mathcal{L}$，而是允许对抗攻击者对样本进行扰动，得到如下优化目标:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}} \max_{\delta\in\mathcal{S}} \mathcal{L}\left(f(\boldsymbol{x}+\boldsymbol{\delta};\boldsymbol{\theta}), y\right) \tag{13.22}$$

其中 $(\boldsymbol{x}, y)$ 是数据集 $\mathcal{D}$ 的数据点，$\boldsymbol{\delta}$ 为限制 $\|\boldsymbol{\delta}\| \leqslant \epsilon$ 内的对抗扰动。上述 min-max 优化框架起源于博弈论，也是鲁棒优化领域的核心问题。内层的 max 优化问题旨在在干净输入附近找到使得模型分类误差最大的扰动，外层的 min 优化目标则更新参数来最小化分类误差，从而达到抵御对抗攻击的目的。对抗训练过程如图13.4所示。

传统的随机梯度下降方法无法直接对公式 (13.22) 进行直接优化，现有的方法通常是对 min-

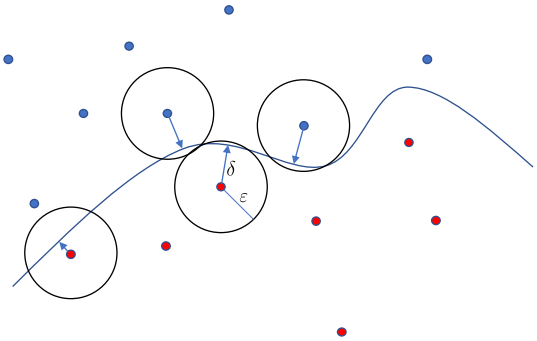

图 13.4 对抗训练过程示意图

max 问题进行交替处理，从而最终使得公式13.22收敛。内层的 max 优化问题通常使用现有的对抗攻击方法进行解决，如 FGSM、PGD 等。FGSM 在原始样本的基础上进行一步梯度下降的来寻找对抗扰动，在此基础上 PGD 使用 K 步随机梯度下降来搜索扰动 δ:

$$\delta_{k+1} = \prod_{\|\delta\| \leqslant \epsilon} \left(\delta_k + \eta \frac{g\left(\delta_k\right)}{\|g\left(\delta_k\right)\|} \right), \tag{13.23}$$

其中 $g\left(\delta_k\right) = \nabla_x \mathcal{L}\left(f(x+\delta_k; m \odot \theta), y\right)$, δ_k 为第 k 步的扰动，$\prod_{\|\delta\| \leqslant \epsilon}(\cdot)$ 重新将对抗扰动投影到弗罗贝尼乌斯范数 (Frobenius norm) 正则化球中。通过上述过程可以生成大量的虚拟对抗样本并参与模型的训练过程。因此，对抗样本的生成质量将决定优化后模型稳健性。尽管 max 优化问题是非凹的，已有的工作表明 PGD 能够提供了性能良好的局部最大值。对于外层的 min 问题，则采用传统的随机梯度下降方法对网络参数进行优化，从而使得模型在对抗样本上的损失达到最小。

在自然语言处理领域中，基于对抗训练的方法对输入的扰动通常是在连续的词嵌入空间进行的，因此这类方法适用于各种模型架构。同时文本对抗训练生成的对抗样本可以视作一种数据增强，丰富了输入数据的多样性，用来提升模型的泛化能力。但是虚拟对抗样本的生成过程需要频繁的梯度回传，这个过程需要会消耗大量的计算资源，非常耗时。

13.4.2 基于表示压缩文本防御方法

文献 [25] 和文献 [26] 的研究表明，深度学习模型的脆弱性可归因于“不鲁棒特征”，即表示空间中存在对抗攻击敏感的特征，这种特征可以轻易被攻击者操纵。这些特征的存在将减少深度学习的鲁棒性。因此，对不鲁棒特征进行过滤将提升模型的鲁棒性。

基于信息论的信息瓶颈方法，可以将深度学习的优化目标阐述为表示压缩和预测能力之间的一个基于信息理论的平衡。给定输入数据 X，通过神经网络得到表示 T，分类目标是最大化 T 和 Y 之间的互信息，在表示 T 复杂性受到约束的情况下，也需要包含足够的信息来推断出目标标签

Y。因此信息瓶颈的优化框架可以表示为:

$$\max L_{IB} = I(Y;T) - \beta I(X;T) \tag{13.24}$$

其中 $I(\cdot;\cdot)$ 表示互信息。对信息瓶颈优化目标的直观理解是, 我们希望压缩输入 X 给出的信息, 同时仍然保持足够的知识, 让模型给出正确的预测结果 Y。在上式中, 参数 β 控制了从输入 X 中保留多少信息。通过增加 β, 我们可以控制缩小"颈部", 从而使得从 X 传输到隐藏特征 T 的信息减少。由于"鲁棒特征"有助于模型的预测, 它们包含输入的语义信息。因此, 整体的目标是过滤掉与任务无关的信息, 同时将与任务有关的信息损失降到最低。这样一来, 就可以提高模型的鲁棒性性, 而不会降低其在预测任务中的性能。

为了达到最小化信息瓶颈的目标, 需要最大化互信息 $I(Y;T)$。考虑到最大化 $I(Y;T)$ 的目的是希望 T 包含足够的信息能够确保模型的预测准确度, 可以选择最小原始任务的损失函数, 以接近 $I(Y;T)$ 的最大化。以文本分类任务为例, 可以通过最小化交叉熵损失 $\mathcal{L}_{CE}$ 来实现 $I(Y;T)$ 的最大化。互信息 $I(X;T)$ 可以通过 $p(T|X)$ 和 $p(T)$ 分布之间的 Kullback-Leibler 散度来计算:

$$\begin{aligned} I(X;T) &= \mathbb{E}_X[D_{KL}[p(T|X)||p(T)]] \\ &= \int p(x,t) \log \frac{p(t|x)}{p(t)} dxdt \end{aligned} \tag{13.25}$$

为了计算 $p(T|X)$ 和 $p(T)$ 之间的 Kullback-Leibler 散度, 需要了解它们的概率分布。$P(T|X)$ 项可以根据经验进行采样。但是, $P(T)$ 项很难被估计。为了解决这个困难, 可以将公式13.25展开, 得到以下的方程:

$$I(X;T) = \int P(x,t) \log P(t|x) dxdt - \int P(t) \log P(t) dt, \tag{13.26}$$

其中 T 的边缘分布 $P(t) = \int P(t|x)P(x)dx$。由于最初的文献 [27] 所提出的方法依靠迭代的 Blahut Arimoto 算法来实现信息瓶颈目标, 而这一算法不用直接应用于深度神经网络。因此许多研究人员试图使用变分推理来近似这一问题[28]。受到之前研究的启发, 使用变分近似 $q(t) = \mathcal{N}(\mu, \sigma)$ 来替代 $p(t)$, 高斯分布的均值和方差分别为 μ 和 σ。由于 Kullback-Leibler 散度非负, 这意味着 $\int P(t) \log P(t) dt \geqslant \int q(t) \log q(t) dt$, 可以推导出上界:

$$\begin{aligned} I(X;T) &\leqslant \int p(x)p(x|t) \log \frac{p(t|x)}{q(t)} dxdt \\ &= \mathbb{E}_X[D_{KL}[p(T|X)||q(T)]] \end{aligned} \tag{13.27}$$

通过减少 X 和 T 之间的互信息, 可过滤掉更多与任务无关的信息, 这可以为最终的预测保留更多的鲁棒特征。为了最小化 $I(X;T)$, 只需要最小化它的上界。通过调整 $q(t)$ 中的参数可以最小化 $p(T|X)$ 和 $q(T)$ 之间的 Kullback-Leibler 散度, 这将降低 $I(X;T)$ 的上限。结合 $I(Y;T)$ 的优化目

标，基于信息瓶颈的文本表示压缩最终损失函数可以表示为：

$$\mathcal{L} = \mathcal{L}_{CE} + \beta \cdot D_{KL}[p(T|X)||q(T)] \tag{13.28}$$

通过使用上式来优化模型，从而可以在一定程度上实现过滤掉不鲁棒的分类任务特征的目标。

13.4.3 基于数据增强的文本防御方法

当标注数据有限时，数据增强是增大训练数据的有效方法。例如，在计算机视觉中，图像被移位、放大/缩小、旋转、翻转、扭曲或遮挡，都可以用于训练数据的增强。但由于文本数据的句法和语义结构复杂，对其进行增强是非常具有挑战性的工作。文献 [29] 提出利用同义词替换、随机插入、随机交换和随机删除进行文本数据增强的方法。但是这些基于规则生成的增强样本无法有效覆盖潜在的样本范围。如果能在训练时，增加数据覆盖对抗攻击的搜索空间，就能够在一定程度上提升模型的鲁棒性。然而，增强样本经常难以获得，或者质量不够高，容易造成模型在具体任务上的性能下降。相较于计算机视觉领域常使用旋转、位移、裁剪等基础操作构造增强样本，文本很难构造简单且高质量的增强样本。基于混合（Mixup）的数据增强逐渐成为图像和文本数据增强的有效手段之一，通过混合两个训练数据线性插值来构造增强样本。这一过程通常可以表示为：

$$\begin{aligned} \hat{x} &= \lambda x_i + (1-\lambda) x_j \\ \hat{y} &= \lambda y_i + (1-\lambda) y_j \end{aligned} \tag{13.29}$$

其中 $\lambda \in [0,1]$ 为混合系数。通过混合方式构造的虚拟训练样本可以用来训练神经网络模型。Mixup 可以用不同的方式进行解释。一方面，Mixup 可以被看作是一种数据增强的方法，它在原始训练集的基础上插值构建新的数据样本。另一方面，它对模型进行了正则化处理，使其在训练数据中表现为线性。Mixup 在连续的图像数据上十分有效，然而，直接将其扩展到文本数据上具有一定的挑战，因为在离散的词语之间进行插值是不可行的。

之前的一些工作表明，对两个句子的表示向量的插值进行解码，会产生一个具有两个原始句子混合意义的新句子。受此启发，MixText[30] 提出了在文本的隐空间中进行插值构造文本数据增强的方法。给定两个文本输入，首先使用包括 BERT 等深度预训练语言模型对句子进行编码。对于一个有 L 层的编码器，选择在第 m 层 $m \in [0, L]$ 混合中间表示。如图13.5所示，MixText 首先在底层分别计算两个文本样本的中间表示。然后，在第 m 层混合中间表示，并将插值后的中间表示送入上层。编码器网络中的第 l 层使用 $g_l(.;\boldsymbol{\theta})$ 表示，第 l 层的中间表示为 $\boldsymbol{h}_l = g_l(h_{l-1};\boldsymbol{\theta})$。对于两个文本样本 $\boldsymbol{x}_i$ 和 $\boldsymbol{x}_j$，可以定义为第 0 为嵌入层，即 $\boldsymbol{h}_0^i = \boldsymbol{W}_E \boldsymbol{x}_i$，$\boldsymbol{h}_0^j = \boldsymbol{W}_E \boldsymbol{x}_j$，则 l 层中两个样本的隐藏表示可以按照如下方式计算得到：

$$\begin{aligned} \boldsymbol{h}_l^i &= g_l(\boldsymbol{h}_{l-1}^i;\theta), \quad l \in [1,m] \\ \boldsymbol{h}_l^j &= g_l(\boldsymbol{h}_{l-1}^j;\theta), \quad l \in [1,m] \end{aligned} \tag{13.30}$$

在第 m 层进行混合后，并继续前向传播到上层可以表示为：

$$\begin{aligned}\hat{\boldsymbol{h}}_m &= \lambda \boldsymbol{h}_m^i + (1-\lambda)\boldsymbol{h}_m^j \\ \hat{\boldsymbol{h}}_l &= g_l(\boldsymbol{h}_{l-1};\theta), \quad l \in [m+1, n]\end{aligned} \tag{13.31}$$

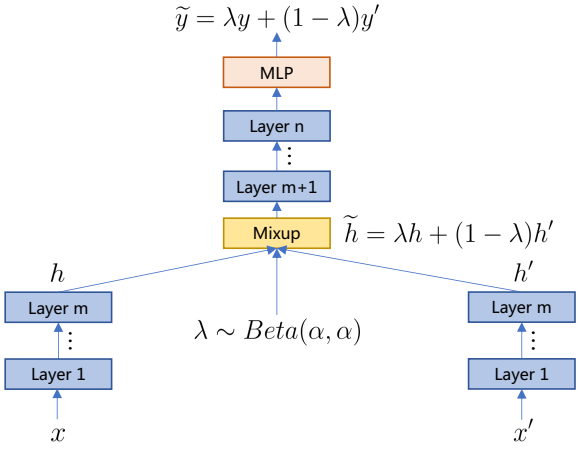

图 13.5　MixText 过程[30]

在实际训练过程中，每一批数据的混合系数 λ 都从 Beta 分布中采样获得：

$$\begin{aligned}\lambda &\sim Beta(\alpha, \alpha), \\ \lambda &= \max(\lambda, 1-\lambda),\end{aligned} \tag{13.32}$$

其中 α 是用于控制 Beta 分布形状的超参数。数据混合的有效性可以有不同的角度进行解释，一方面，数据混合可以看作是一种数据增强方法，它原始训练数据进行插值构建新的数据样本。另一方面，它对模型了正则化处理，迫使模型对线性插值的数据同样输出线性插值的结果。通过进一步引入使用文本对抗样本，使干净的训练样本与文本对抗样本进行混合，可以进一步扩大增强样本的覆盖范围，数据混合结果如图13.6所示。

13.4.4 对抗样本检测

对抗样本检测目标是将对抗样本与正常样本进行区分，并在预测阶段将其抛弃，从而达到防御的目的。检测-丢弃策略可以与之前所介绍的防御方法相结合，从而构建更加鲁棒的自然语言处理系统。对抗样本检测的一个重要挑战是，探索一种有效的特征来区分出干净样本与对抗样本。目前对抗样本检测主要有两类算法：一类文本对抗样本检测方法引入基于密度估计和距离度量等统计量，并根据文本表示的特点进行改进；另一类方法则是基于对抗文本生成算法在特性构建相应

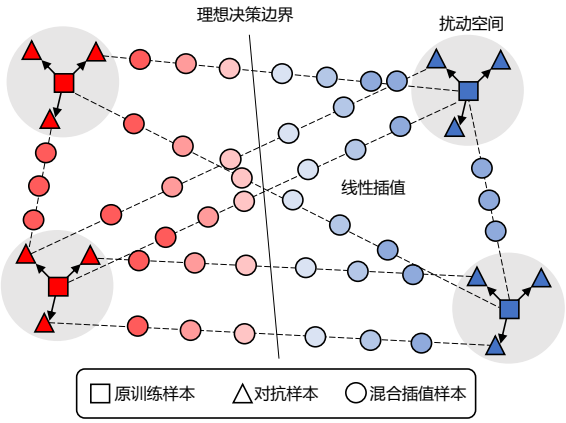

图 13.6 数据混合示意图

的检测策略。

文献 [31] 发现词级别对抗攻击倾向于把原始输入文本中的高频词替换成低频词，并提供了统计证据来支持这一猜想。基于这个发现，文献 [31] 提出一种基于词频且与模型无关的检测算法 FGWS（Frequency Guided Word Substitution），来检测潜在的对抗样本，并尽力恢复出对抗样本的原始形式。FGWS 算法首先对原始样本和对抗样本的词频进行了分析，计算所有被攻击的原始词 x 与对应替换词 x' 在训练集中对数频率（$\log_e$ Frequency）$\phi(x)$ 与 $\phi(x')$。通过这种方式可以统计被攻击的原始词在训练集上对数词频的平均值 μ_ϕ 与标准差 σ_ϕ，以及替换词在训练集上词频的平均值 $\mu_{\phi'}$ 与标准差 $\sigma_{\phi'}$。表13.1给出了了针对 RoBERTa 模型的原始样本和对应攻击样的词频统计结果。可以看到，在不同的数据集和攻击中，对抗攻击的替代词的频率始终低于被选中的原始词。

表 13.1 针对 RoBERTa 模型对抗样本对数频率统计[31]

数据集	攻击方法	原始词		替换词	
		μ_ϕ	σ_ϕ	μ_ϕ	σ_ϕ
IMDb	RANDOM	7.6	2.5	3.4	2.8
	PRIORITIZED	7.6	2.5	3.6	2.8
	GENETIC	6.5	2.0	3.7	2.3
	PWWS	6.9	2.3	4.4	2.5
SST-2	RANDOM	5.4	2.6	2.1	1.4
	PRIORITIZED	5.4	2.6	2.1	1.4
	GENETIC	4.4	1.9	2.2	1.2
	PWWS	4.8	2.1	2.9	2.2

基于对被替换词和对抗替换词频率差异，FGWS 算法认为这种替换策略产生的影响可以通过简单的基于频率的转换来减轻。用 $f(X)$ 来表示分类模型，它将一个序列 X 映射为一个 c 维度向量，代表 c 个可能类别的概率，输入序列表示为 $X = \{x_1, \ldots, x_n\}$，其中 x_i 表示序列中的第 i 个词。FGWS 通过用语义相似并且在模型训练语料库中出现频率较高的词，用来替换输入中频率较低的词，将 X 转换为替换序列 X'。对于每个符合条件的词 $x \in X$，有同义候选词集合 $S(x)$，并通过选择 $x' = \arg\max_{w \in S(x)} \phi(w)$ 来找到替换词 X'。如果 $\phi(x') > \phi(x)$，通过用 x' 替换每个符合条件的词 x 来生成 x'。给定 X 的预测标签 $y = f(X)$ 和阈值 $\gamma \in [0, 1]$，如果 $f(X)_y - f(X')_y > \gamma$，即如果数据变换前后对类 y 的预测置信度的差异超过阈值 γ，则序列 X 被认为是对抗性的。阈值允许控制识别的假阳性率（即被错误地识别为对抗性的未扰动序列）。算法结果识别结果示例如图13.7所示。通过 FGWS 算法发现了对抗算法所替换的单词并进行了还原。

攻击方式	原句子或扰动后的句子
无	A clever blend of fact and fiction
Genetic	1.39 ←— 5.55 A **brainy** [*clever*] blend of fact and fiction
PWWS	1.61 ←— 5.55　0.00 ←— 3.81 A **cunning** [*clever*] **blending** [*blend*] of 0.00 ←— 4.39 fact and **fabrication** [*fiction*]

图 13.7　FGWS 算法结果实例[31]

13.5 模型稳健性评价基准

针对自然语言处理任务的评价通常采用精度、召回、F1 值、准确率等指标。算法如果在标准测试集合上得到了很好的测试精度或者准确率，是否就意味着该算法在真实环境下就一定能得到很好的效果呢？经典的评价方法能全面反映算法的优缺点吗？算法在测试语料上取得很好的效果，是否真的说明算法达到语料集合创建者所预设的验证目标？针对这些问题，近年来一些研究从机器学习、自然语言处理、特定任务等角度分别开展了一些研究。在本章中我们将针对模型通用评价以及特定任务评价两方面的工作分别进行介绍。

13.5.1 特定任务稳健性评价基准

自然语言处理相关任务多种多样，很多任务有明显的特点，并依赖的不同语言学特征。单一的准确率指标不能全面准确的衡量算法效果。因此，一些研究工作根据不同的任务特点，设计特定的稳健性评价方法和基准。本节中将针对情感倾向分析和阅读理解两个任务介绍特定任务稳健性评价方法和基准设计方法。

1. 情感倾向分析稳健性评测

属性级情感分析（Aspect-based Sentiment Analysis，ABSA）旨在预测文本中所表达的针对某一特定属性或方面的情感，是一种细粒度的情感分类任务。例如："这款手机的电池续航能力很好，但是显示分辨率太低了"中分别针对"电池"和"分辨率"给出了评价。ABSA 模型应该只对目标属性的情感词敏感，而不会被其他非目标属性的倾向性影响。

文献 [4] 指出尽管模型在测试集上得到很高的准确率，但是这些的模型的稳健性仍然存在一定的问题。假设一个模型在测试样本上能够输出正确的结果，该方法试图在一下方面进一步验证模型的鲁棒性：

(1) 通过修改句子中目标属性的情感词，将目标属性的情感极性颠倒。
(2) 将所有非目标属性的的情感词进行修改，使之与目标方面的情感相反。
(3) 增加更多的非目标属性评价。

针对上述三个方面，属性级情感倾向稳健性测试集 ARTS（Aspect Robustness Test Set）对应的设计了三种变形进行测试。

REVTGT：生成反转目标属性词情感极性的句子。SemEval2014 中将每个属性对应的情感词的范围标注出来，因此可以设计规则来反转情感极性。比如将情感词替换为其反义词，或者在情感词之前加上否定词 not 等，同时需要将不同属性词之间的连接词进行调整，比如将 and 修改为 but 来表示转折关系。通过 REVTGT 改变目标情绪可以测试出如果一个模型对目标属性词的情感是否足够敏感。

例如：原始句子"Tasty burgers, and crispy fries."，目标属性为"burgers"，通过 REVNON 变形为"Terrible burgers, but crispy fries."

REVNON：改变非目标属性词的情感极性，将所有非目标属性词中情感极性与目标词一致的情感极性进行反转。而对于其余非目标属性中情感极性已经与目标情感情感极性不一样的，通过随机添加副词来夸大其情感极性。例如："非常"、"真的"、"和"、"极度"，等利用训练语料构建的程度副词字典。

例如：原始句子"Tasty burgers, and crispy fries."，目标属性为"burgers"，通过 REVNON 变形为"Tasty burgers, but soggy fries."

ADD-DIFF：添加句子中没有出现的属性词情感描述，其情感极性与目标属性情感极性相反。现有的 SemEval2014 测试集平均每句只有两个属性，但现实世界中的应用可以有更多的属性词。因此可以首先形成一个属性表达的集合 AspectSet，从整个数据集中提取所有的属性表达。通过使用 AspectSet，可以从中随机采样 1-3 个在原始测试用例中未提及且情感极性与目标属性不同的属性，然后将它们拼接到原始文本中。

例如：原始句子"Great food and best of all GREAT beer!"，目标属性为"food"，通过 ADD-DIFF 变形为"Great food and best of all GREAT beer, but management is less than accommodating."

ARTS 评测集合针对 Laptop 和 Resturant 领域分别构建了 1877 和 3530 个评测数据。利用该评

测集合，文献 [4] 针对 9 个典型方法进行了评测，其中包括 BERT-PT[32] 等方法。结果表明，9 种方法平均准确率在 Laptop 领域从 71.60% 下降到 25.23%，在 Resturant 领域从 79.77% 下降到 31.62%。

2. 阅读理解稳健性评测

ASQuAD（Adversarial SQuAD）[33] 是针对斯坦福问答数据集（Stanford Question Answering Dataset，SQuAD）的对抗评估方法。ASQuAD 测试了系统是否能够回答包含对抗插入句子的段落的问题，通过自动生成的句子来影响阅读理解算法。利用 ASQuAD，文献 [33] 对 16 个模型进行了测试，模型准确性从平均 75% 的 F1 分数下降到 36%，当对抗样本中允许增加不符合语法的序列，平均性能进一步下降到 7%。

ASQuAD 不依靠转述，而是使用改变语义的扰动来建立拼接式对抗样本，为某个句子 s 生成形式为 $(p+s,q,a)$ 的样本。换句话说，拼接式对抗在段落的末尾添加一个新句子，而不改变问题和答案。有效的对抗样本应该是与正确答案不矛盾的样本，也称为与 $(p+s,q,a)$ 兼容的句子。ASQuAD 提出了两种具体的变形方式 ADDSENT 和 ADDANY。ADDSENT 增加看起来与问题具有相似语法结构的句子，从而起到混淆模型的效果。ADDANY 则是增加任意的英语单词序列，使它有更大的能力来混淆模型。

ADDSENT 采用四个步骤来生成看起来与问题相似，但实际上与正确答案不矛盾的句子。具体的步骤如下：

(1) 对问题进行扰动改变其语义内容，以保证产生的对抗句子是兼容的。ASQuAD 根据 WordNet 反义词替换名词和形容词，并将命名实体和数字改为 GloVe 词向量空间中与之最接近的词。

(2) 创建一个类型与原始答案相同的假答案。ASQuAD 定义了 26 种类型，对应于斯坦福大学 CoreNLP 的 NER 和 POS 标签，再加上一些自定义的类别（例如缩写），并人工将一个假答案与每个类型联系起来。给出一个问题的原始答案，ASQuAD 计算其类型并返回相应的假答案。

(3) 使用一组大约 50 个人工定义的规则，将改变后的问题和假答案合并为陈述句形式。例如，如果问句符合预定义规则“what/which NP1 VP1 ?” 则将其转化为陈述句 “The NP1 of [Answer] VP1”。

(4) 通过众包方法修改对抗样本中的错误，每个句子由 Amazon Mechanical Turk 上的五个众包人员独立编辑。之后，另外三名众包人员过滤掉不符合语法或不兼容的句子，从而得到一个较小的（可能是空的）人工认可的句子集。

ADDSENT 算法对每个人工认可的句子上以黑箱方式运行模型 f，并挑选出使模型给出最差答案的句子。如果没有人工认可的句子，则简单地返回原始例子。ADDSENT 方法例子如图13.8所示。

ADDANY 方法目标是选择任意 d 个单词的序列，并且不考虑语法性。ASQuAD 使用局部搜索来生成对抗句子 $s=w_1w_2\dots w_d$。首先从一个常见的英语单词列表中随机初始化单词 $w_1w_2\dots w_d$。然后，进行 n 次局部搜索，每次搜索都以随机顺序在索引 $i\in\{1,\dots,d\}$ 上进行迭代。对于每个位置，随机采样获得 20 个常用词和 q 中所有词作为候选集合 W。对于每个 $x\in W$，将句子中 w_i 替

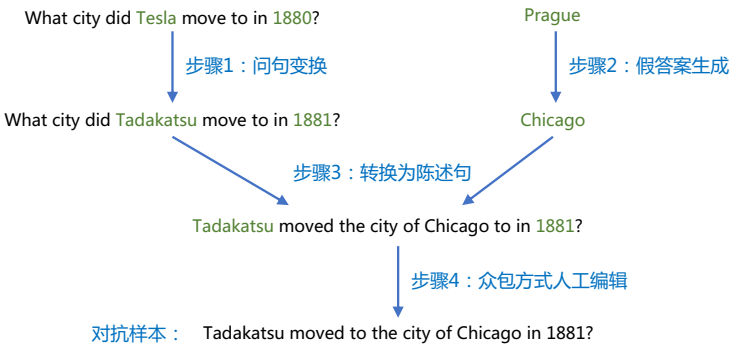

图 13.8 ASQuAD 语料集中 ADDSENT 生成算法[33]

换为 x，并将生成的句子与原始的段落合并。使用模型计算在这种情况下问句在对应答案上的得分。最终，将 w_i 更新为能使得模型 $F1$ 分数最小的 x。

13.5.2 模型稳健性通用评价基准

针对特定任务的稳健性评价方法有很多的共同之处，比如相似单词替换、数字替换、错别字替换等可以用几乎所有自然语言处理任务中。因此，也有一些工作尝试开展通用领域或者领域无关的模型稳健性评测和基准构建。本章将介绍 Checklist 和 TextFlint 两种方法。

1. CheckList 通用稳健性评测方法

获得 ACL 2020 最佳论文奖的 CheckList[34] 框架就是一种针对自然语言处理模型稳健性测试框架。传统自然语言处理任务的评价通常比较简单，仅考虑准确率、精度、召回率等效果问题。软件工程研究领域中有各种测试复杂软件系统的范式和工具，特别是“行为测试”（也被称为黑盒测试），它关注的是通过验证输入输出行为来测试系统的不同能力，而对内部结构没有任何了解。受到软件工程中最小单元测试和行为测试的启发，CheckList 提供一个适用于大多数自然语言处理任务的语言能力列表来指导用户对自然语言处理模型进行综合行为的测试。为了将潜在的能力故障分解成具体的行为，CheckList 引入了不同的测试类型，如在某些扰动下的预测不变性，受到某些指向性扰动时候预测结果的改变等。

如图13.9所示，用户通过填写矩阵中的单元格来检查一个模型，每个单元格可能包含多个测试。表格中行表示测试的不同能力，列表示了不同的测试类型。CheckList 应用了测试与实现脱钩的行为测试原则，将模型视为一个黑盒，这使得能够对在不同数据上训练的不同模型进行比较，也能够对无法访问训练数据或模型结构的第三方模型进行测试。

虽然测试单个组件是软件工程中的常见做法，但是目前自然语言处理模型很少是建立在单一组件上的。尽管如此，CheckList 仍然鼓励用户考虑不同的自然语言能力在当然任务上是如何体现的，并创建测试集来评估模型的每一项能力。例如，词汇与词性能力涉及到一个模型是否能适当

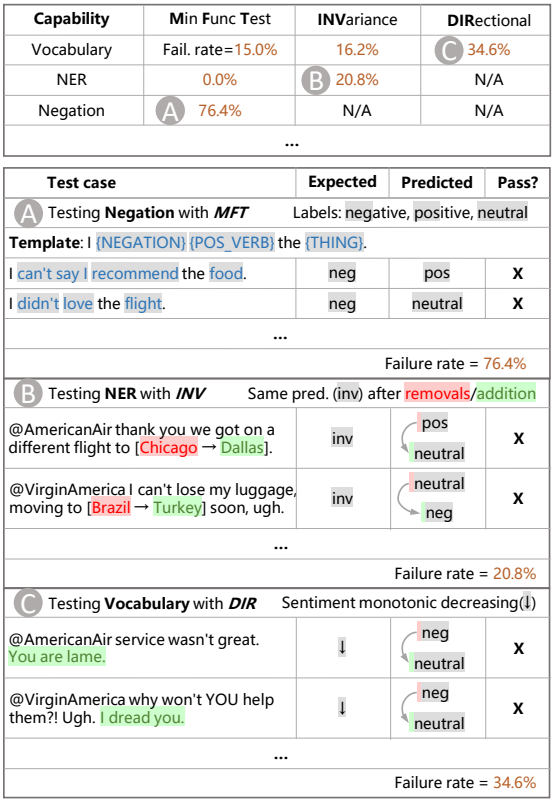

图 13.9 CheckList 算法示例[34]

地处理具有不同词性的单词对任务的影响。对于情感分析，研究人员希望去检查模型是否能够识别出带有积极、消极或者中性情绪的词语。对于语义匹配任务，希望模型能够理解修饰词对句子的影响，比如“李华是一名教师吗？”与“李华是一名合格的教师吗？”中修饰词“合格”影响这两句话语义的关键修饰语。

CheckList 建议模型使用者应当要考虑以下能力：词汇与词性（对任务来说重要的词或词性）、词语分类（同义词、反义词等）、稳健性（对错字、无关扰动等）、公平性、时间性（理解事件的顺序）、否定、共指、语义角色标签（理解角色，如代理、对象等）、逻辑（处理对称性、一致性和连接词的能力）等。CheckList 提供了三种不同的测试类型来评估每种能力：最小功能测试、不变性和指向性期望测试。**最小功能测试**（Minimum Functionality test，MFT）是样本和对应标签的简单集合，用于检验模型基础能力。MFT 类似于构建小而集中的测试数据集，可以用于检测模型是否使用捷径来处理复杂的输入，但并没有真正掌握该解决任务的能力。**不变性测试**（Invariance test，INV）是指对输入施加标签保护性扰动，并期望模型预测保持不变。不同的能力需要不同的扰动函数，例如改变地址名称来测试命名实体识别能力，或者引入错别字来测试稳健性。**指向性期望测试**

（Directional Expectation test，DIR）指对输入施加扰动并预期标签会以某种方式变化。例如，如果在针对影评的末尾加上“演员真是太差劲了”，会使得这段影评的预期情绪变得更加消极。图13.9提供了如何测试这些能力的例子。

研究人员可以从头开始创建测试用例，或者通过扰动现有的数据集来创建测试用例。CheckList为用户提供了一种创建扰动数据的工具，通过遮掩模板的一部分，并借助基于遮掩的语言模型（如RoBERTa 等）获得遮掩部分的填充建议，例如，“I really {mask} the flight”，利用 RoBERTa 模型可以得到 {enjoyed，liked，loved，regret，...}，用户可以选择积极、消极和中性的填充词，然后在多个测试中重复使用。

2. TextFlint 稳健性测试平台

由复旦大学自然语言处理实验室开发的，针对多语言自然语言处理稳健性评测平台 TextFlint[7]，不仅提供了通用文本变形，还包含特定任务变形，并集成了对抗攻击和子集等稳健性测试方式，以及各种该方法的组合以提供了全面的稳健性分析。TextFlint 具有以下特点：

（1）灵活：TextFlint 提供了 20 种通用变形和 60 种特定任务变形，以及它们的数千种组合，涵盖了文本变形的方方面面，以便对模型的稳健性进行全面评估。TextFlint 支持中英文多种语言的评估，自动评估模型在词汇、语法和语义方面的缺陷，或者根据用户的需求进行灵活的定制分析。对于那些个性化的需求，用户可以修改配置文件，并输入几行代码来实现特定的评估。

（2）便捷：TextFlint 提供了约 7000 个新的评估数据集，这些数据集是由 40 个原始数据集变形生成的，用于 20 个任务。用户可以直接下载这些数据集进行稳健性评估。对于那些需要全面评估的用户，TextFlint 支持在一个命令种生成所有变形文本和对应标签，对模型进行自动评估，并生成分析报告。

（3）直观：通过对现有变形结果的合理性和语法性进行人工评价后，以人工评价结果为基础，对每个评估结果分配一个置信度分数。基于评估结果，TextFlint 提供了一个标准的分析报告，涉及到模型的词汇、语法和语义。所有的评估结果都可以通过可视化和表格的形式显示出来，以帮助用户快速准确地掌握一个模型的缺点。此外，TextFlint 根据分析报告中发现的缺陷，生成大量有针对性的数据来增强被评估的模型，并为模型缺陷提供补丁。

TextFlint 的变形形式基于语言学的指导，根据词法、语法、词形变化关系、语用学设计了 20 种通用变形以及 60 种特定任务变形。主要包含以下类型：

(1) 词汇形态

- 形态派生：通过添加前缀或后缀等方法形成新单词的过程。比如：normal 变形为 abnormal。TextFlint 中 *SwapPrefix* 是保持词性的基础上更换前缀。例如：transfix 转化为 affix。
- 词形变化：英语中时态、数、性别决定了很多词的词形。TextFlint 中 *SwapVerb* 将动词的词形进行变化。例如：“He is studying NLP.” 转换为 “He has studied NLP.”。
- 缩略语：通过缩短或合并两个单词得到的词语。TextFlint 中 *Contraction* 将缩略词替换

为原始形式，或者将原始形式替换为缩略形式。例如：“can’t” 转换为 “can not”。

(2) 类聚关系

- 同义词：通过替换具有相同含义的词语或者词组。TextFlint 中 *SwapSyn* 就是利用同义词替换构建的变形。例如：“He loves NLP.” 转换为 “He likes NLP.”。
- 反义词：通过增加否定词或者替换为反义词构造语义相反的句子。这种方式在语义匹配任务中需要同步修改分类结果，但是在信息抽取等任务中则无需修改。TextFlint 中 *SwapAnt* 通过替换反义词，*Add/RmvNeg* 则通过增加或删除否定词完成反义关系构造。例如：“John lives in Ireland.” 转换为 “John doesn’t live in Ireland.”。

(3) 语法

- 句法范畴：通过替换具有相同句法范畴的成分，可以在不影响句法结构的情况下构造变形。同样需要注意的是这种变形会引起语义的变化，需要根据任务确认是否适用或修改相应的分类标签。TextFlint 中 *SwapNamedEnt*、*SwapSpecialEnt* 以及 *SwapWord/Ent* 就是利用修改相同句法范畴的词语完成变形。例如：“I love Shanghai.” 转换为 “I love Beijing.”。
- 附属成分：通过增加或者删除某些附属成分，构造符合语法的变形。TextFlint 中 *Delete/AddSubTree* 和 *InsertClause* 分别是通过增加或删除子树，以及增加小句生成变形。例如：“Tom loves NLP.” 转换为 “Tom, who lives in China, loves NLP.”。

(4) 语用：

- 会话准则：在不同的环境下，人们为了有效沟通会采用不同的方式进行，因此可以利用这一特性构建变形。TextFlint 中 *RndRepeat/Delete* 针对篇章随机删除或者复制一个句子、*TwitterType* 替换为社会媒体上常用词语、*AddSum* 增加部分描述等都是基于会话准则模型构建变形。例如：“See you later.” 转换为 “CYL”。
- 偏差：语言反映了社会价值和个人观点，因此通常存在偏差。TextFlint 中 *Prejudice* 变形就是利用这种偏差，对相关内容进行替换从而生成变形。例如：“She is a nurse.” 转换为 “He is a nurse.”。

此外，TextFlint 还包含利用翻译模型将原始句子翻译为其他语言，再翻译回来构造变形的 *BackTrans* 方法，利用复述生成模型构建语义相似句子的 *Overlap* 等方法。图13.10给出了 TextFlint 中任务无关通用变形的分类图。

除此之外，TextFlint 还集成了对抗攻击和子集划分用于全方位验证模型。TextFlint 提供了 16 种简单易用的文本对抗攻击方法，用于验证模型对抗稳健性。子集划分用来确定目标模型在数据集中表现不佳的特定部分。通过对样本进行分类可以按照某些属性来划分数据集，TextFlint 提供了四个常用的子集划分属性，其中包括性别偏差、文本长度、语言模型困惑度和短语匹配。以文本长度属性为例，文本长度筛选出长度最长的 20% 或者最短的 20% 的子集，然后测试模型在子集数据上的表现，来确定模型的预测结果是否会受到这些属性的影响。

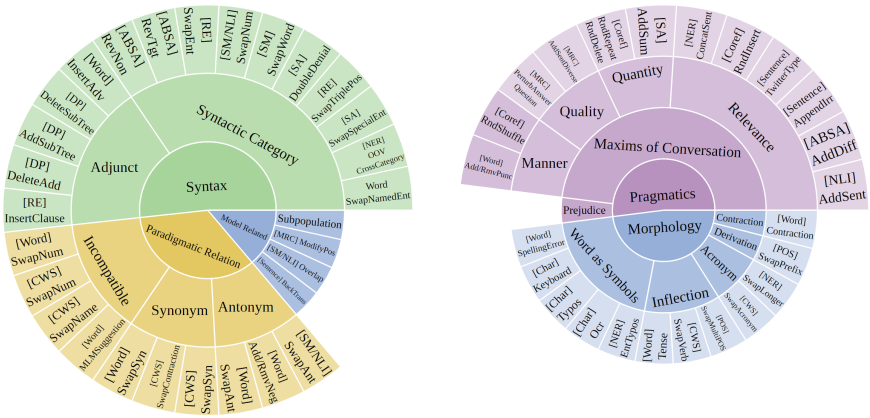

图 13.10 TextFlint 中通用变形分类[7]

通过这些变形以及对抗攻击，使得我们可以对模型的稳健性进行更为全面的分析。表13.2给出了利用 TextFlint 平台对多种大规模预训练语言模型在语言推理 MultiNLI 任务上的评测结果。验证了反义词替换、增加句子、数字变形以及复述生成等变形形式。从结果上我们可以看到，虽然当前的模型在原始集合上取得了不错的结果，但是简单的变换一下数字就会使得模型的准确率大幅度下降，在一定程度上反映了当前模型稳健性普遍亟待提升。

表 13.2 模型在语义推理任务 MultiNLI 数据集合上准确率

模型	**SwapAnt** 原始 → 变形	**AddSent** 原始 → 变形	**NumWord** 原始 → 变形	**Overlap** 原始 → 变形
BERT-base [35]	85.10 → 55.69	84.43 → 55.27	82.97 → 49.16	None → 62.67
BERT-large [35]	87.84 → 61.18	86.36 → 58.19	85.42 → 54.19	None → 70.65
XLNet-base[36]	87.45 → 70.98	86.33 → 57.65	85.55 → 48.77	None → 70.35
XLNet-large[36]	89.41 → 75.69	88.63 → 63.37	86.84 → 51.35	None → 78.09
RoBERTa-base[37]	87.45 → 63.53	87.13 → 57.25	86.58 → 50.32	None → 75.49
RoBERTa-large[37]	92.16 → 74.90	90.12 → 67.73	88.65 → 54.71	None → 73.14
ALBERT-base-v2 [38]	87.45 → 50.20	84.09 → 53.59	82.97 → 49.42	None → 67.15
ALBERT-xxlarge-v2 [38]	91.76 → 69.80	89.89 → 79.11	89.03 → 46.84	None → 74.92
平均	88.58 → 65.25	87.12 → 61.52	86.00 → 50.60	None → 71.56

13.6 延伸阅读

本章中我们主要介绍了常见的文本对抗攻击和文本对抗防御方法，并没有深入涉及模型对抗脆弱性的成因，对抗样本的成因以及性质仍然是对抗鲁棒性领域的研究重点。

此前的工作倾向于将对抗样本视为由输入空间的高维性质或训练数据的统计波动引起的畸变 [39, 40]。从这个角度来看，可以很自然地将对抗鲁棒性视为一个目标，可以通过改进的标准正则化方法或对网络输入/输出进行预/后处理，将其与最大化准确性分开并独立追求 [41, 42]。最近的观察表明，对抗样本是由于数据中具有良好泛化能力但是敏感的特征所引起的 [43]。因为模型训练过程中的目标是最大化任务精度，因此模型将尽可能的利用一切可以利用的特征信号，即使有些特征对于人类来说是无法理解、不合理的。

每个对抗样本都可以看作是被一个连通区域（称为“对抗性区域”或“对抗性子空间”）围绕，在这个区域内，所有的数据都以类似的方式欺骗了模型。对抗性区域不仅可以在输入空间中定义，还可以在不同 DNN 层的激活空间中定义 [44]。在数据可以用流形建模的假设下，一些工作试图刻画对抗子空间的特征，但目前还没有明确的方法可以可靠地将对抗区域与正常数据的区域区分开。Szegedy et al. [44]等人认为，对抗子空间是低概率区域（不是自然发生的），密集地分散在 DNN 的高维表示空间中。然而，线性表述（linear formulation）认为，对抗子空间位于连续的多维空间，而不是随机散布在小范围内 [39]。Tanay et al. [45]进一步强调，对抗样本位于靠近（但不在）干净数据子流形的地方。总结一下，就数据的流形模型而言，已知的对抗子空间的特性是：（1）它们是低概率的，（2）它们位于一个连续的多维空间，（3）它们位于干净数据局部流形之外（但接近于）。

13.7 习题

（1）有哪些文本生成的技术指标可以用来约束文本对抗样本的生成质量？
（2）如何改进对抗训练算法的计算效率？
（3）有哪些特性可以用来区分干净样本和对抗样本？
（4）如何将对抗检测和对抗防御同时应用于一个模型中？
（5）后门攻击的缺点是什么？

参考文献

[1] He P, Liu X, Gao J, et al. Deberta: Decoding-enhanced bert with disentangled attention[C]// International Conference on Learning Representations. 2020.

[2] Roberts A, Raffel C, Shazeer N. How much knowledge can you pack into the parameters of a language model?[J]. arXiv preprint arXiv:2002.08910, 2020.

[3] Wang A, Pruksachatkun Y, Nangia N, et al. Superglue: A stickier benchmark for general-purpose language understanding systems[J]. Advances in neural information processing systems, 2019, 32.

[4] Xing X, Jin Z, Jin D, et al. Tasty burgers, soggy fries: Probing aspect robustness in aspect-based sentiment analysis[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020: 3594-3605.

[5] Lin H, Lu Y, Tang J, et al. A rigorous study on named entity recognition: Can fine-tuning pretrained model lead to the promised land?[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020: 7291-7300.

[6] Si C, Yang Z, Cui Y, et al. Benchmarking robustness of machine reading comprehension models[C]// Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. 2021: 634-644.

[7] Wang X, Liu Q, Gui T, et al. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing[C]//Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations. 2021: 347-355.

[8] Zhang X Y, Liu C L, Suen C Y. Towards robust pattern recognition: A review[J]. Proceedings of the IEEE, 2020, 108(6):894-922.

[9] 周志华. 机器学习[M]. 北京: 清华大学出版社有限公司, 2016.

[10] Bengio Y, Courville A, Vincent P. Representation learning: A review and new perspectives[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(8):1798-1828.

[11] Gardner M, Artzi Y, Basmov V, et al. Evaluating models' local decision boundaries via contrast sets [C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings. 2020: 1307-1323.

[12] Sakaguchi K, Le Bras R, Bhagavatula C, et al. Winogrande: An adversarial winograd schema challenge at scale[C/OL]//Proceedings of the AAAI Conference on Artificial Intelligence: volume 34. 2020: 8732-8740. https://ojs.aaai.org/index.php/AAAI/article/view/6399.

[13] Trichelair P, Emami A, Cheung J C K, et al. On the evaluation of common-sense reasoning in natural language understanding[J]. arXiv preprint arXiv:1811.01778, 2018.

[14] Zellers R, Bisk Y, Schwartz R, et al. SWAG: A large-scale adversarial dataset for grounded commonsense inference[C/OL]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, 2018: 93-104. https://aclanthology.org/D18-1009. DOI: 10.18653/v1/D18-1009.

[15] Ebrahimi J, Rao A, Lowd D, et al. Hotflip: White-box adversarial examples for text classification[C]//Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). 2018: 31-36.

[16] 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.

[17] 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: volume 34. 2020: 8018-8025.

[18] Zhao Z, Dua D, Singh S. Generating natural adversarial examples[C]//International Conference on Learning Representations. 2018.

[19] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks[J]. Communications of the ACM, 2020, 63(11):139-144.

[20] Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks[C]//International conference on machine learning. PMLR, 2017: 214-223.

[21] Koh P W, Liang P. Understanding black-box predictions via influence functions[C]//International conference on machine learning. PMLR, 2017: 1885-1894.

[22] Chen X, Salem A, Backes M, et al. Badnl: Backdoor attacks against nlp models[C]//ICML 2021 Workshop on Adversarial Machine Learning. 2021.

[23] Kurita K, Michel P, Neubig G. Weight poisoning attacks on pretrained models[C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020: 2793-2806.

[24] Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks [J]. arXiv preprint arXiv:1706.06083, 2017.

[25] Ilyas A, Santurkar S, Tsipras D, et al. Adversarial examples are not bugs, they are features[J]. Advances in neural information processing systems, 2019, 32.

[26] Tsipras D, Santurkar S, Engstrom L, et al. Robustness may be at odds with accuracy[C]//International Conference on Learning Representations: number 2019. 2019.

[27] Tishby N, Pereira F C, Bialek W. The information bottleneck method[J]. arXiv preprint physics/0004057, 2000.

[28] Alemi A A, Fischer I, Dillon J V, et al. Deep variational information bottleneck[J]. arXiv preprint arXiv:1612.00410, 2016.

[29] Wei J, Zou K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks[C]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019: 6382-6388.

[30] Chen J, Yang Z, Yang D. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification[C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020: 2147-2157.

[31] Mozes M, Stenetorp P, Kleinberg B, et al. Frequency-guided word substitutions for detecting textual adversarial examples[C]//Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume. 2021: 171-186.

[32] Xu H, Liu B, Shu L, et al. Bert post-training for review reading comprehension and aspect-based sentiment analysis[C]//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). 2019: 2324-2335.

[33] Jia R, Liang P. Adversarial examples for evaluating reading comprehension systems[C]//Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 2017: 2021-2031.

[34] Ribeiro M T, Wu T, Guestrin C, et al. Beyond accuracy: Behavioral testing of nlp models with checklist[C]//Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020: 4902-4912.

[35] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[C]//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). 2019: 4171-4186.

[36] Yang Z, Dai Z, Yang Y, et al. Xlnet: Generalized autoregressive pretraining for language understanding[J]. Advances in neural information processing systems, 2019, 32.

[37] Delobelle P, Winters T, Berendt B. RobBERT: a Dutch RoBERTa-based Language Model[C/OL]// Findings of the Association for Computational Linguistics: EMNLP 2020. Online: Association for Computational Linguistics, 2020. https://www.aclweb.org/anthology/2020.findings-emnlp.292. DOI: 10.18653/v1/2020.findings-emnlp.292.

[38] Lan Z, Chen M, Goodman S, et al. Albert: A lite bert for self-supervised learning of language representations[C]//International Conference on Learning Representations. 2019.

[39] Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples[C/OL]// Bengio Y, LeCun Y. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. 2015. http://arxiv.org/abs/1412.6572.

[40] Gilmer J, Metz L, Faghri F, et al. Adversarial spheres[C/OL]//6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings. OpenReview.net, 2018. https://openreview.net/forum?id=SkthlLkPf.

[41] Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks[C/OL]//6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. https://openreview.net/forum?id=rJzIBfZAb.

[42] Stutz D, Hein M, Schiele B. Disentangling adversarial robustness and generalization [C/OL]//IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long

Beach, CA, USA, June 16-20, 2019. Computer Vision Foundation / IEEE, 2019: 6976-6987. http://openaccess.thecvf.com/content_CVPR_2019/html/Stutz_Disentangling_Adversarial_Robustness_and_Generalization_CVPR_2019_paper.html. DOI: 10.1109/CVPR.2019.00714.

[43] Ilyas A, Santurkar S, Tsipras D, et al. Adversarial examples are not bugs, they are features[C/OL]// Wallach H M, Larochelle H, Beygelzimer A, et al. Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. 2019: 125-136. https://proceedings.neurips.cc/paper/2019/hash/e2c420d928d4bf8ce0ff2ec19b371514-Abstract.html.

[44] Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks[C/OL]//Bengio Y, LeCun Y. 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings. 2014. http://arxiv.org/abs/1312.6199.

[45] Tanay T, Griffin L D. A boundary tilting persepective on the phenomenon of adversarial examples [J/OL]. CoRR, 2016, abs/1608.07690. http://arxiv.org/abs/1608.07690.

索引