概要

随着机器学习技术在恶意软件检测中的广泛应用，研究者提出了多种对抗攻击方法以生成恶意软件的对抗样本（AEs），从而规避检测。已有研究表明，强化学习（RL）框架能够通过执行一系列功能保留操作实现黑盒攻击，生成具备功能性的规避性恶意软件样本。然而，在黑盒场景下，智能体训练难以从环境中获取有效的指导和反馈，导致强化学习框架无法习得有效的规避策略。本文提出沙普利先验（Shapley prior），并构建了一种基于先验指导的强化学习框架PSP-Mal，用于生成针对便携式可执行（PE）文件恶意软件检测器的对抗样本。该框架在以下三方面改进了现有方法：1）通过计算沙普利值探索黑盒模型的特征影响，进而提出沙普利先验表征操作的预期影响；2）在强化学习框架中建立基于沙普利先验指导的新型优先级经验利用机制；3）将动作扩展为操作项-内容对，并利用汤普森采样（Thompson sampling）选择有效内容，有助于减少随机性并确保可复现性。通过与其他方法的攻击性能对比实验表明，本文算法具有更高有效性。PSP-Mal针对基于EMBER和SOREL-20M数据集训练的LightGBM模型的规避率分别达到76.88%和72.03%。

CCS概念  
• 安全与隐私 → 恶意软件及其缓解措施；  
• 计算理论及方法 → 机器学习；

关键词  
恶意软件检测，规避攻击，对抗样本，强化学习，优先级经验回放，沙普利值

1 引言

面对日益增长且快速演变的Windows便携式可执行（PE）文件恶意软件，健壮且准确的静态检测器通常作为抵御恶意软件洪流的第一道防线。传统反恶意软件方案依赖特征码[40]与启发式算法[4]，但难以应对日益严峻的挑战。随着机器学习（ML）技术的发展，许多基于ML的方法[9, 20, 28, 38, 41]被提出以判定软件恶意性，凭借模型优异的表征与泛化能力实现检测。尽管基于ML的恶意软件检测器展现出卓越性能，但其可靠性仍面临对抗样本（AEs）[21, 46]的挑战——攻击者通过施加细微的对抗性扰动修改恶意软件样本，即可使其规避检测，即规避攻击。

攻击场景通常可分为白盒与黑盒设置。在白盒设置中，攻击者可获取目标恶意软件检测器的内部信息，并利用模型梯度生成对抗样本（AEs）[21]。相反，黑盒方法假设无法访问模型参数与梯度信息。由于现实场景更符合黑盒设置，本研究聚焦于针对静态PE恶意软件检测器的黑盒对抗攻击。具体而言，对抗性修改在PE恶意软件的问题空间（problem space）[36]中实施，以生成保留功能的规避变体。

基于梯度的方法难以直接迁移至恶意软件领域，这源于黑盒设置的不可知性及问题空间约束。现有研究[2, 19, 55]表明，基于强化学习（RL）算法实施问题空间黑盒攻击是一种可行方案。强化学习智能体以端到端（end-to-end）方式习得规避策略：仅需输入恶意软件样本，即可输出应执行的功能保留型修改操作。然而，受限于黑盒场景的信息匮乏，强化学习框架的攻击性能存在严重瓶颈。智能体需通过修改恶意软件实现规避并获取最终奖励，这一过程任务复杂度高，且需要大量有效信息指导其训练。此外，现有方法设计的动作空间包含过度随机性，导致智能体难以准确预测动作效果[8, 15, 17]。

在此背景下，本文的核心思路是借助可解释机器学习工具——沙普利加性解释（SHAP）[32, 33]——提供先验指导信息。我们将通过SHAP方法解释获得的指导信息作为沙普利先验（Shapley prior），表征不同动作的预期贡献。通过基于先验知识加权转移利用率概率，所提出的优先级经验回放机制可提升经验利用效率，并进一步改善攻击性能。同时，采用汤普森采样（Thompson sampling）解决动作内部随机性引发的缺陷。最终，我们构建了整合上述机制的强化学习框架PSP-Mal，并通过全面实验验证其在生成恶意软件对抗样本方面的优越性。结果表明，该算法在规避率与学习效率上均展现出显著优势。、

本文主要贡献可归纳为以下三方面：  
• 基于SHAP的黑盒检测器解释与先验指导：利用沙普利加性解释（SHAP）方法解析黑盒恶意软件检测器，识别关键特征并提出沙普利先验（Shapley prior），表征不同操作执行的预期影响。  
• 优先级经验回放机制设计：在强化学习过程中建立基于沙普利先验指导的新型优先级经验回放机制，以提升攻击性能。  
• 动作建模与随机性优化：将修改动作建模为操作项-内容对，并采用汤普森采样（Thompson sampling）选择有效动作内容，在利用成功内容与探索更多变体间实现最优平衡。

为确保本研究工作的可复现性并推动该领域后续研究，我们将PSP-Mal的源代码公开于：<https://github.com/zhan8002/PSP-Mal>

2 背景

2.1 规避攻击

在恶意软件领域，规避攻击指攻击者在测试阶段修改恶意软件样本以逃避检测的行为。根据恶意软件的表征方式，规避攻击可分为特征空间攻击（feature-space attack）与问题空间攻击（problem-space attack）[36]。特征空间攻击最初在图像领域被研究[21, 46]，随后迅速扩展至其他领域[1, 5, 13]。在特征空间中，恶意软件样本被抽象为连续数值表示，多数对抗攻击方法采用基于梯度驱动的特征空间攻击[7, 26, 54]。为生成真实可执行文件，需通过逆特征映射函数将特征空间中的"规避性PE特征"转换为问题空间中的"规避性PE恶意软件"。然而，由于PE文件的独特格式，逆特征映射函数既不可逆也不可微，导致基于梯度的方法难以直接应用于恶意软件。

为生成具有实际效用的规避性PE恶意软件，可行的解决方案是寻求问题空间攻击。该方法通过实施不破坏原始文件恶意功能的问题空间转换实现，例如将对抗扰动注入二进制文件的未使用区域，包括PE头文件[10, 12]、段间隙与段尾部[29, 45, 54]以及文件尾部[7, 26, 27, 29]。更先进的方法则采用进化算法[39, 48]、启发式搜索算法[11]、生成对抗网络（GAN）[23, 24, 53]以及强化学习（RL）[2, 8, 14–17, 19, 30, 31, 37, 44, 50, 55, 56]实施黑盒攻击。

强化学习方法的优势在于通过将攻击面限制在问题空间转换范围内，有效应对问题空间约束。如图1(a)所示，RL框架定义精心设计的动作集合𝑨（包含一组功能保留型修改操作）。在规避过程中，RL智能体根据策略𝜋𝜃(·|𝒔)从𝑨中选择并执行动作序列：修改时间戳（𝑎1）→修改段名称（𝑎2）→覆盖追加（𝑎3）。每个动作𝑎𝑖在问题空间修改恶意软件样本，使样本状态从𝒔𝒊转变为𝒔𝒊+1，迭代持续直至实现规避。

图1(b)展示了智能体通过功能保留型修改集合𝑨直接在问题空间选择动作并修改样本的过程。与此同时，恶意软件样本在特征空间中的表示𝑓𝑖相应改变，最终实现规避并生成恶意软件对抗样本。已有研究表明，RL技术能够实现针对黑盒模型的恶意软件规避，而本文工作聚焦于提升RL框架的规避性能。

2.2 现有基于强化学习的技术  
强化学习（Reinforcement Learning, RL）旨在解决智能体在与环境交互过程中如何习得最优策略的问题。在生成PE恶意软件规避样本的RL框架中，**状态𝑠**、**动作𝑎**与**奖励𝑟**等核心组件定义如表1所示。

首项将RL框架应用于恶意软件对抗样本生成的研究是gym-malware [2]，其设计了10类功能保留型修改操作构成动作集合。尽管gym-malware仅需获知检测器的输入输出即可生成规避梯度提升决策树（GBDT）[18]静态检测器的对抗样本，但其相较于随机策略的性能提升有限。基于此，研究者相继提出多种针对静态PE恶意软件检测的对抗攻击方法[8, 14–17, 30, 31, 37, 44, 50]，并将其扩展至规避基于图网络[55]、流量特征[51]、恶意软件图像[37]及商业杀毒引擎[31, 44, 56]的检测器。

gym-malware[2]采用深度Q网络（DQN）与带经验回放的行动者-评论家（ACER）框架这两种经典RL算法。后续研究通过改进算法取得一定进展：以Double DQN[15, 50]、Dueling DQN[15]和Distribution DQN[30]替代基础DQN，以A2C[8]和A3C[16]替代ACER。奖励函数对智能体学习过程具有关键作用。除将成功规避检测作为主要奖励标准外，部分研究[16, 17, 30]将尝试次数纳入奖励函数，激励智能体以更少修改步骤达成目标。Raphael等人[30]还将修改后文件与原始二进制的相似度作为奖励函数组成部分。此外，AMG-IRL[31]采用逆强化学习（Inverse Reinforcement Learning, IRL）[35]，从成功轨迹中自动学习奖励函数，而非依赖人工设计。

Merlin[37]与gym-plus[50]将动作空间从gym-malware的10个操作扩展至16个，新增修改时间戳、移除调试信息等操作。部分研究者[8, 15, 17]认为原始动作空间存在随机性导致不可复现问题，通过以确定性动作替代部分随机动作[15]或缩减动作空间[8, 17]提升探索效率。另有研究[19, 55, 56]结合代码混淆技术[52]构建动作集，生成可规避检测的代码混淆操作序列。

综上，自[2]之后，后续研究主要从RL框架优化、奖励函数设计与动作集改进三方面进行提升（详见附录表6）。然而，黑盒场景下智能体的信息瓶颈问题仍未解决。本文提出在RL过程中建立基于沙普利先验指导的新型优先级经验回放机制，以提升学习效率与规避性能。

表1：诱导PE恶意软件样本规避检测任务中的强化学习组件

状态空间S

𝑆表示恶意软件样本在检测环境中的状态。在静态检测中，𝑆通常以两种形式表征：

原始字节序列（raw byte sequence）；

提取的特征向量（extracted feature vector）

行为空间A

A表示智能体在问题空间（problem space）中可对PE文件执行的修改操作集合

奖励r

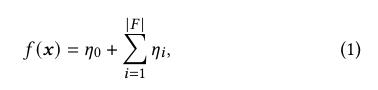
奖励函数由检测器计算，若修改后的恶意软件成功规避检测（evade detection），智能体将获得最终奖励𝑅。

转换

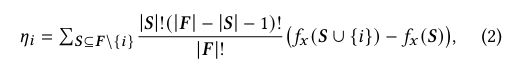
描述智能体与环境之间的交互过程，用于计算奖励和更新策略。

2.3 沙普利加性解释

复杂模型常缺乏可解释性，导致预测结果的归因难以理解。采用简单可解释模型近似黑盒模型是一种可行方案[34, 43]。基于合作博弈论思想，沙普利加性解释（SHapley Additive exPlanations, SHAP）[32]通过构建替代线性模型，将模型输出𝑓 (𝒙)解释为各输入特征归因值𝜂𝑖之和，表示为：



其中 𝒙 = (𝒙1,𝒙2,...,𝒙𝑭)，|𝐹| 为特征数量。仅基于输入特征向量 𝒙 和模型 𝑓 (𝒙) 的输出预测，通过对所有特征序列取特征值 𝒙𝒊 的不同边际贡献的平均值，即可获得 沙普利加 值 𝜂𝑖，表示为：



其中 𝑺 是从 𝑭 中提取的子集，𝑓𝑥(𝑺) 是其预测值，而 𝑺 中未包含的特征 𝑥𝑖 被边缘化。

尤其值得强调的是，SHAP作为一种黑盒解释方法，无需获知被解释模型的结构与内部信息，且已被证实性能优于多种早期模型解释算法[43]。凭借这一特性，攻击者能够以\*\*模型不可知（model-agnostic）\*\*的方式量化目标检测器的特征贡献。

3 威胁模型

威胁模型是对网络攻击的系统化表征，本文从攻击者目标与知识及目标模型两方面描述威胁模型。

攻击者目标与知识：攻击者旨在修改恶意软件样本以规避静态Windows PE恶意软件检测器，即令修改后样本被判定为良性。本文考虑黑盒设置：攻击者无法访问目标检测器内部信息，仅能进行有限次尝试并获取预测置信度（prediction confidence）。同时，攻击者对输入数据的操作被限制为功能保留型修改。若修改导致二进制文件可执行性或恶意功能破坏，则攻击视为失败。

目标模型：为了便于比较所提出的 PSP-Mal 和基准方法，我们实施了现有攻击方法中最流行的目标模型 [2、8、14、17、30、37、44、50]，即 LightGBM（轻梯度增强机）[25]。LightGBM 是一个应用 GBDT 算法的框架，它具有相对高效的内存数据处理程序，支持使用大量数据进行快速训练。为了证明攻击的有效性，我们将测试两个 LightGBM 模型作为目标模型，分别在 EMBER [3] 和 SOREL-20M [22] 上进行训练。

EMBER数据集[3]作为基准数据集被采用，其收集了从110万个PE文件中提取的9类特征（详见表2）。该数据集使用2381维特征向量表征每个样本，包含两大特征类型：原始特征（字节直方图（ByteHistogram）、字节熵直方图（ByteEntropyHistogram）、字符串信息（StringInfo））与解析特征（通用文件信息（GeneralFileInfo）、头文件信息（HeaderFileInfo）、段信息（SectionInfo）、导入表信息（ImportsInfo）、导出表信息（ExportsInfo）、数据目录信息（DataDirecties））。SOREL-20M数据集[22]为大规模数据集，包含2017至2019年间采集的近2000万个文件。除公开元数据外，SOREL-20M还提供类似EMBER的特征矩阵，用于训练LightGBM模型。为便于后续研究复现与对比，我们采用其公开发布的预训练模型（分别基于EMBER 2与SOREL-20M 3数据集训练）。性能方面，两模型的ROC AUC分数均超过0.995，并为静态恶意软件模型设定阈值为0.83，该阈值对应约0.1%误报率（false positive rate）下的93%检出率（true positive rate）。

表2：

表2：EMBER定义的九类特征及其内容

特征 描述

F1：字节直方图 & 整个二进制文件的字节分布直方图。

F2：字节熵直方图 & 字节值与局部熵的联合概率分布。

F3：字符串信息 & 字符串相关的可打印字符统计信息。

F4：通用文件信息 & 从PE文件头提取的基础信息。

F5：头文件信息 & 头文件数据（机器类型、链接器版本、操作系统等）。

F6：区段信息 & 各区段属性（名称、大小、熵值等）。

F7：导入表信息 & 导入的库函数与API信息。

F8：导出表信息 & 导出的函数信息。

F9：数据目录信息 & 前15个数据目录的大小及虚拟地址。

4.PSP-Mal框架

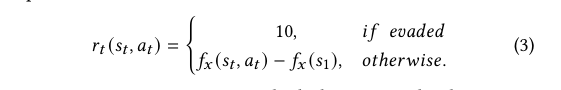
本节首先介绍基于强化学习的恶意软件规避框架基础结构与核心组件（第4.1节）。为指导智能体训练，我们利用SHAP方法解释黑盒模型，提取信息作为沙普利先验（第4.2节）。随后提出两个改进模块：基于沙普利先验的优先级经验回放（第4.3节）与动作内部汤普森采样（第4.4节）。最终，通过整合上述模块实例化D3QN强化学习算法（第4.5节）。为突显"基于沙普利先验指导的优先级经验回放"这一核心思想，本框架命名为PSP-Mal。

4.1 结构与组件

PSP-Mal的结构示意图如图2所示。训练过程中，智能体首先观测恶意软件样本的状态（以特征向量𝑠表示）。框架中的修改动作𝑎被建模为操作项-内容对（item-content pairs）：操作项由策略（policy）决定，内容通过汤普森采样（Thompson sampling）选择。随后，修改后的恶意软件输入目标检测器获取预测置信度（predictive score），计算奖励𝑟并反馈至汤普森采样模块以更新内容分布。交互信息不仅用于计算时间差分（Temporal Difference, TD）误差，同时输入SHAP解释器计算沙普利先验（Shapley prior），进而根据优先级将状态转移存储至记忆库（memory buffer），供智能体学习。整个过程持续直至成功规避目标检测器或达到最大修改次数。

为实现上述机制，我们对强化学习（RL）框架的多个组件进行重新设计，特别是对动作空间（Action Space）进行重构

状态空间：恶意软件样本在环境中以原始字节（raw bytes）形式存在，其结构与表征被映射至状态空间。我们期望环境能够实现对智能体动作及其对应效果的全观测。本文采用与恶意软件检测器相同的特征表示作为环境状态空间，即2381维特征向量。相较于原始字节序列，提取后的特征更为简洁高效，有利于智能体学习。如Anderson等人[2]指出，\*\*EMBER数据集[3]\*\*收集的特征向量提供了对恶意软件样本的全面观测，囊括了工业界与学术界主流检测器[9, 28, 41]所使用的核心要素，表达式为：



奖励机制：强化学习（RL）框架的终极目标是通过修改二进制文件实现恶意软件检测器规避。我们为每个状态计算奖励𝑟作为中间反馈，引导智能体寻找正确动作。在时间步𝑡，奖励值𝑟𝑡等于检测器对原始状态𝑠1的置信度𝑓(𝒙)与当前状态𝑠𝑡的置信度差值。若𝑟𝑡为正，表明时间步𝑡执行的动作序列[𝑎1,𝑎2,...,𝑎𝑡]提升了样本被判定为良性的置信度。一旦修改后的样本成功规避检测，将获得最终奖励𝑅 = 10。

动作空间（Action Space）：现有方法大多未考虑动作内部内容，而是随机生成内容。例如，若策略决定下一动作为SectionAdd（添加区段），现有方法会以随机内容填充新区段。此类方法无法充分利用动作信息，且过度随机性导致不可复现性（non-repeatability）与Q值估计偏差。本文在PSP-Mal中设计了一种新型恶意软件修改器（详见第4.4节），将动作视为操作项-内容对（item-content pair），即𝑎∼(𝑖𝑡𝑒𝑚,𝑐𝑜𝑛𝑡𝑒𝑛𝑡)。状态与策略决定操作项，而内容则通过汤普森采样（Thompson sampling）从数据池（data pool）中选取，而非随机生成。

动作集（Action Set）：本节简要介绍动作集设计。基于前沿研究成果[2, 14, 30]，我们定义了10种修改动作，均遵循PE文件格式结构。根据动作类型，将动作划分为修改型（Modifying）与添加型（Additive）两类：

修改型动作：调整PE文件中特定辅助属性（auxiliary attributes）的值。辅助属性通常用于程序调试与分析，不控制程序执行逻辑。因此，程序运行不依赖这些属性，只要属性值合理，便不会影响可执行文件正常功能。

添加型动作：将字节添加至未使用区域，如区段间隙（gaps between sections）、新增区段（new sections）及文件尾部（end of the file）。由于这些字节在程序执行时不会被加载至内存，故不影响文件功能。

根据操作区域，动作可进一步分为头文件（Header）、区段（Section）与文件尾部（End of the file）。如表3所示，定义的10种动作涵盖多种类型与区域，赋予智能体灵活修改恶意软件特征的能力。

数据池维度（Data Pool Size）指动作内容的数据池规模。例如：

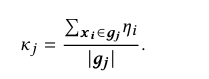
Add Imports（添加导入函数）通过汤普森采样从20个潜在导入函数中选择内容；

Remove Debug（移除调试信息）与Break Checksum（破坏校验和）因执行简单置零操作，其数据池规模为1。

此外，实验中使用的操作可行性分析将在第6节讨论。

4.2 沙普利先验

利用可解释机器学习技术度量动作的预期贡献以形成先验知识，可为强化学习（RL）智能体训练提供有效信息指导。通过将恶意软件样本特征输入SHAP解释器（SHAP explainer），我们可获取各特征的沙普利值𝜂𝑖。基于EMBER数据集定义的九个特征组[𝒈1,𝒈2,...,𝒈9]，计算每个特征组𝒈𝑗的平均沙普利值𝜅𝑗，表达式为



模型预测值𝑓(𝒙)可表示为各特征沙普利值𝜂𝑖的累加，进而可通过式(5)转换为特征组贡献值的累加形式，其表达式为

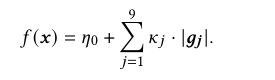


图3展示了基于EMBER数据集训练的LightGBM模型决策过程。每条彩色线代表一个样本的决策路径：从底部的基线值𝜂0开始，逐步累加不同特征𝜂𝑖的贡献值，最终到达顶部的预测值𝑓(𝒙)。当预测值𝑓(𝒙)超过阈值时，样本被判定为恶意。特别地，左侧蓝色线条代表一个被误分类的异常样本。我们通过柱状图呈现各特征组𝒈𝑗的平均沙普利值𝜅𝑗。可观察到DataDirectories（数据目录信息）特征组的贡献值最高，表明该特征对恶意性判定的影响最为显著。

表4展示了不同动作可能影响的特征组。单个动作通常会影响多个特征，例如：

所有动作均影响三个原始特征：字节直方图（Byte Histogram）、字节熵直方图（ByteEntropy Histogram）与字符串信息（StringInformation）；

添加导入函数动作（Add Import）额外影响三个解析特征：通用文件信息（GeneralFileInfo）、导入表信息（ImportsInfo）及数据目录信息（DataDirectories）。

我们可构建二元矩阵𝑴表征动作与特征组的关联性，其中元素1表示动作𝑎能够影响特征组𝒈𝑗。动作𝑎对检测的预期贡献可理解为以下两个因素的综合效应：

动作𝑎是否影响特征组𝒈𝑗（由𝑴决定）；

特征组𝒈𝑗的恶意性显著度（由𝜅𝑗表征）。

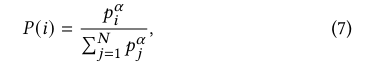
由于动作对特征的影响难以量化，且基于"优先修改高恶意性特征"的设计原则，我们仅计算恶意性最高的前三个特征组的沙普利值加权和。由此，动作𝑎对分类的预期影响可估计为：



其中𝝉 = [𝜅₁,𝜅₂,...,𝜅₉]。在生成恶意软件对抗样本（AEs）过程中，智能体通过选择功能保留型操作序列修改原始二进制文件以实现检测器规避。基于恶意性分布𝝉与关联矩阵𝑴，我们可将每个动作的预期效果估计为智能体探索过程的先验知识𝜌-shap（prior knowledge 𝜌-𝑠ℎ𝑎𝑝），从而有效指导策略学习。

4.3 基于沙普利先验的优先级经验回放（PER）

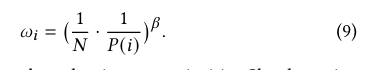
在DQN（深度Q网络）算法中，经验回放缓存（experience buffer）用于存储智能体与环境交互产生的所有状态转移（𝑠𝑡, 𝑎𝑡, 𝑟𝑡, 𝑠𝑡+1）。算法从缓存中采样小批量转移数据以更新Q网络参数。常规方法采用随机采样，而更高效的优先级经验回放（Prioritized Experience Replay, PER）[42]通过时间差分（Temporal Difference, TD）误差衡量每个状态转移的重要性，并优先采样高重要性经验。优先级机制可显著提升经验回放效率与学习效果。转移𝑖的采样概率可表示为



其中𝑁为缓存容量，𝛼为缩放系数。当𝛼=0时，采用均匀采样。在PSP-Mal中，我们提出一种整合沙普利先验𝜌-𝑠ℎ𝑎𝑝的新型优先级经验回放机制，通过结合反映动作预期效果的先验估计值𝜌-𝑠ℎ𝑎𝑝与时间差分误差𝛿𝑇𝐷，重新定义各状态转移的优先级度量。优先级𝑝的计算公式为：



其中𝜍为权重因子。使用𝑟𝑎𝑛𝑘()函数可使度量指标对异常值不敏感且更具鲁棒性。由于优先级机制改变了经验回放（replay memory）的分布，会引入Q网络估计偏差[42]，可通过在损失函数中添加重要性采样权重（importance sampling weight）𝜔𝑖消除该偏差，其表达式为：



通过所提机制，我们将沙普利先验（Shapley prior）置于强化学习（RL）经验之上优先考量。具有高𝜌-𝑠ℎ𝑎𝑝（沙普利先验值）的动作更可能实现有效规避，因此在经验回放过程中，涉及此类动作的状态转移将获得更高采样优先级。优先级评估通过综合考量以下因素实现：

沙普利先验值𝜌-𝑠ℎ𝑎𝑝：表征动作对规避效果的预期贡献；

时间差分（TD）误差𝛿𝑇𝐷：反映同一动作历史转移中的策略学习价值。

4.4 动作内部汤普森采样（Thompson Sampling）

如前所述，动作内容的随机性会引发诸多缺陷。现有研究面对探索-利用困境（exploration-exploitation dilemma）时，常采取增减动作空间的策略：

部分学者提出以确定性动作替代所有随机动作[15]，即将每个动作内容作为新动作加入动作集；

另有研究[17]直接移除含随机性的动作或指定动作内容。

然而，动作空间过大会导致智能体无法有效评估所有可能动作，而简单缩减动作集则无法保证其具备完成规避任务的能力。

为解决此问题，PSP-Mal将动作建模为操作项-内容对，并设计新型PE文件修改器，采用汤普森采样（Thompson sampling）[6]选择动作内容。该修改器包含表3所列10个操作项，其中8个操作项具有多个可选内容（选项数量等于数据池D的规模）。选定操作项𝑎后，内容通过汤普森采样从数据池D中选取，而非随机生成。

我们假设每个内容选项的预期贡献𝜙𝑖服从贝塔分布（Beta distribution）——一种由两个正参数（𝜆𝑖, 𝜇𝑖）定义的连续概率分布，即𝜙𝑖 ∼ 𝐵𝑒𝑡𝑎(𝜆𝑖, 𝜇𝑖)。

对于每个内容选项，我们从其分布中采样一个值，并选择具有最高采样值的选项，其表达式为：

𝑜𝑝𝑡𝑖𝑜𝑛 = argmax 𝜙𝑖, 𝜙𝑖 ∼ 𝐵𝑒𝑡𝑎(𝜆𝑖,𝜇𝑖).

以Overlay Append（覆盖追加）动作为例：我们期望从数据池D中多个良性软件的部分字节中筛选出最优内容。初始阶段，每个内容的贝塔分布参数设为𝜆=1、𝜇=1。每次采样执行后：

奖励反馈机制：若执行当前修改操作Modifier(OA,𝑐𝑜𝑛𝑡𝑒𝑛𝑡)获得的奖励𝑟超过阈值𝜎，则将该内容对应的𝜆值加1；

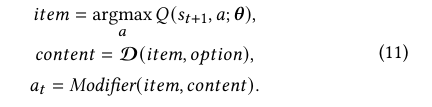
惩罚反馈机制：反之，若𝑟未达阈值，则其𝜇值加1。

汤普森采样根据奖励反馈持续更新各选项𝜙𝑖的概率分布，进而选择下一内容选项。通过智能体多次执行OA动作，可逐步收敛至最优内容。

4.5 我们的强化学习算法

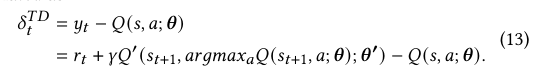
我们基于最先进的异策略（off-policy）强化学习模型实现上述机制。D3QN算法整合了对决深度Q网络（Dueling DQN）[49]与双深度Q网络（Double DQN, DDQN）[47]的核心思想。通过评估评估网络𝑸与目标网络𝑸’的交互作用，该算法有效消除了Q值过高估计（overestimation）问题。

结合智能体决策与汤普森采样（Thompson sampling），时间步𝑡的动作𝑎𝑡可表示为：

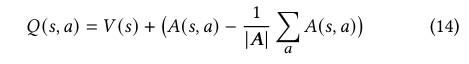


给定状态转移元组（𝑠𝑡, 𝑎𝑡, 𝑟𝑡, 𝑠𝑡+1），我们获取沙普利先验值𝜌-𝑠ℎ𝑎𝑝，并将其作为元组的一部分存储至经验回放缓存（replay memory）。

根据式(7)计算优先级，其中时间差分（TD）误差的计算公式为：



𝜽与𝜽′分别表示评估网络𝑸与目标网络𝑸’的参数。评估网络𝑸用于选择状态下最优动作值对应的动作，目标网络𝑸’则用于计算动作值。受对决DQN（Dueling DQN）算法[49]启发，本算法将状态价值函数𝑉(𝑠)与优势函数𝐴(𝑠,𝑎)分开估计，再结合二者输出以计算Q值



由于已采用重要性采样（importance sampling），损失函数的计算公式为：



其中𝐵为批处理规模（batch size），𝑦𝑗已在式(13)中定义。综合各组件，本文所提算法的完整细节如算法1所示

5 实验评估

数据集：我们从VirusTotal中随机选取4000个能被目标模型准确检测的样本。数据集经洗牌后按9:1比例划分为训练集与测试集。动作添加内容提取自PortableApps下载的良性软件.itext区段。

实验环境：所有实验在配备Intel Xeon E5-2630 CPU、32GB内存及双NVIDIA RTX 2080Ti 11GB GPU的工作站上运行，使用Python 3.9与Ubuntu 20.04系统。本攻击方法基于LIEF与PEfile库实现PE文件解析与操作。训练算法与网络架构的主要超参数详见附录表5。

评估指标：为评估PSP-Mal性能，我们采用规避率（Evasion Rate, ER）衡量对抗样本逃逸PE恶意软件检测系统的能力，其定义为：



其中：

𝑁 𝑒𝑣𝑎𝑠𝑖𝑜𝑛∧𝑓𝑢𝑛𝑐𝑡𝑖𝑜n：成功规避检测且保留功能的样本数量；

𝑁 𝑡𝑜𝑡al：原始样本中被目标模型检测到的总样本数。

5.1 逃逸性能

我们采用最先进的基于强化学习（RL）的恶意软件规避方法作为基线，与提出的PSP-Mal进行性能对比，包括gym-malware [2]、DQEAF [17]、A3CMal [16]、MAB恶意软件（MAB）[44]与AIMED [30]。为保证公平性，所有方法的奖励函数均采用置信度分数（软标签，soft label）。实验中还对比了基于遗传算法的黑盒攻击方法GAMMA [11]，每种攻击对检测模型的查询次数不超过60次（𝑇 = 60）。

图5展示了不同方法对EMBER[3]与SOREL-20M[22]的规避性能对比，其中实线表示均值，误差带为标准差范围，曲线末端标注最终规避率。可见，本文提出的PSP-Mal在两种检测模型上均达到最高规避率。当尝试次数达到𝑇=60时，多数现有RL框架与GAMMA的规避率不足PSP-Mal的一半。相比之下，经PSP-Mal修改后：

EMBER模型：76.88%的样本成功规避检测；

SOREL-20M模型：72.03%的样本成功规避检测。

上述结果分别较次优方法MAB-Malware提升超过15%与7%。此外，PSP-Mal能以更少尝试次数生成规避样本，表明其已习得更高效的修改策略。

方法对比分析：通过对比各方法攻击EMBER与SOREL-20M模型的结果，发现基线方法对SOREL-20M的欺骗性更强。尽管PSP-Mal在SOREL-20M上的规避率有所下降，但其规避率增速更快。这表明机器学习模型难以同时实现泛化能力与对抗鲁棒性：SOREL-20M更大的训练集虽提升了模型泛化性能，却也加剧了其对对抗攻击的脆弱性。

动作分布解析：图6展示了PSP-Mal生成规避样本时修改动作序列中各动作占比。SA（SectionAdd，添加区段）与OA（OverlayAppend，覆盖追加）动作执行频率最高。结合表4分析，这两个动作除影响原始特征外，会显著作用于DataDirectories（数据目录信息）与GeneralFileInfo（通用文件信息）特征。由此推测，这两类特征的修改是规避成功的根本原因之一，验证了前文关于DataDirectories与GeneralFileInfo具有较高恶意性的结论（参见第4.2节与图3）。

5.2 优先级经验回放机制

在PSP-Mal中，我们通过结合沙普利先验（Shapley prior）与时间差分（TD）误差构建了新型优先级度量标准，以评估状态转移价值。为验证该机制的改进效果，我们对比了不同优先级计算方案：

SP方案：仅基于沙普利先验𝜌-𝑠ℎ𝑎𝑝计算优先级；

TD方案：基于TD误差的传统优先级经验回放（PER）方法。

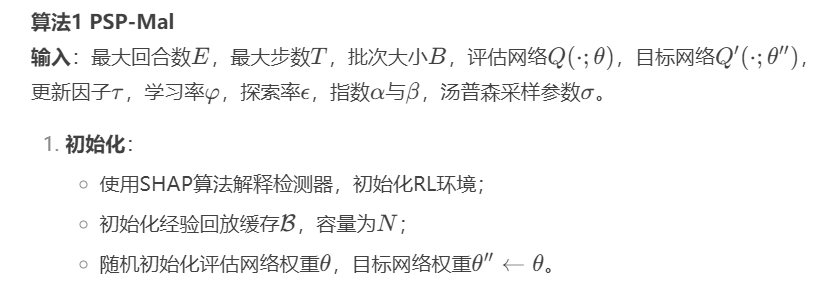
对两个检测模型（EMBER与SOREL-20M）使用相同随机种子序列，对各候选方案进行多次训练。如图7所示，基于混合度量的PSP-Mal（橙色曲线）在两种模型上的平均奖励均显著优于单因素方案。

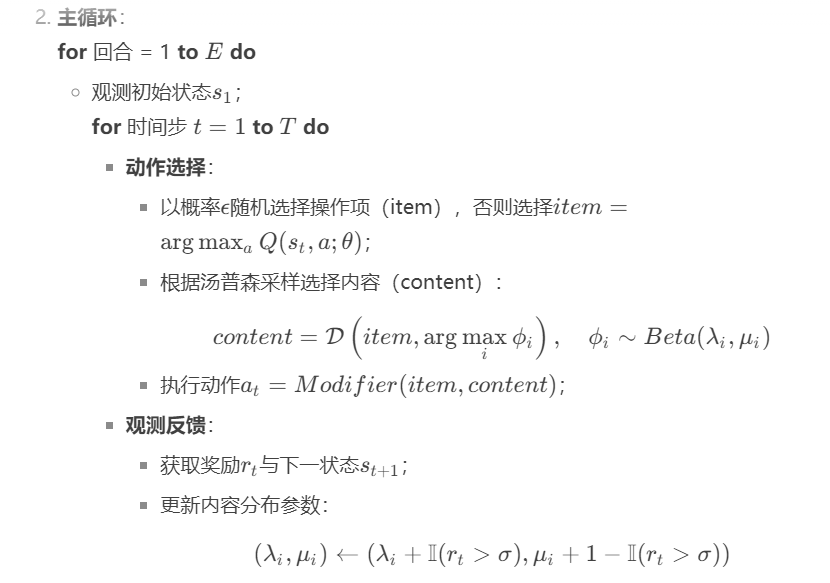
实验结果分析：观察发现，两种检测模型中方案对比呈现相似趋势：纯TD或SP方案的平均奖励均低于PSP-Mal混合方案。结合沙普利先验𝜌<sub>𝑠ℎ𝑎𝑝</sub>可提升关键状态转移利用率，从而提高平均奖励。然而，纯SP方案导致奖励曲线剧烈波动与性能不稳定（如图7所示），原因在于𝜌-𝑠ℎ𝑎𝑝作为先验知识仅与当前状态-动作对（𝑠𝑡</sub>, 𝑎𝑡）相关，无法基于实时奖励𝑟𝑡与下一状态𝑠 𝑡+1更新优先级。这表明：

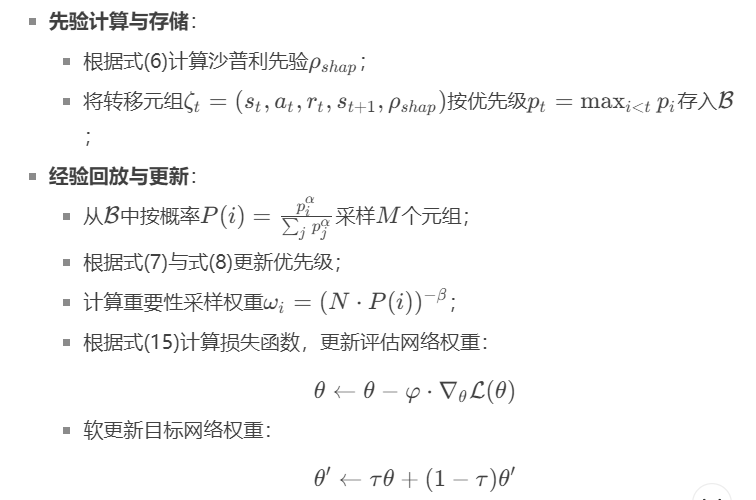
TD误差作用：抑制策略更新的随机波动；

沙普利先验优势：加速收敛过程。

参数健壮性验证：由于采用𝑟𝑎𝑛𝑘()函数归一化处理，沙普利先验权重设为0.01时（如第5.1节所述），其性能与权重0.1或0.001的设置相近（如图8所示），表明算法性能对沙普利先验权重选择不敏感。







5.3 消融研究

在原始D3QN算法框架基础上，我们引入了两项新机制：

基于沙普利先验的优先级经验回放（PER）；

动作内部汤普森采样（TS）。

为评估各机制对生成规避性恶意软件样本的影响，本文进行消融实验，结果如图9所示。其中：

TS：动作内部汤普森采样（Thompson sampling）；

PER：基于沙普利先验的优先级经验回放；

基线方法：随机选择与原始D3QN算法。

首先，随机策略智能体（random agent）的规避率约为25%，而原始D3QN算法显著超越随机动作选择，表明智能体已习得有效规避策略。进一步分析发现：

EMBER模型：

PSP-Mal较原始D3QN提升28%规避率；

单独使用任一机制（PER或TS）可提升约13%；

SOREL-20M模型：

PSP-Mal综合提升超过30%规避率。

值得注意的是，两项机制存在协同效应：其整体效果超越单独效果的简单累加。但由于二者并非简单线性叠加，无法断言何者更为关键。

6 讨论

6.1 功能验证：本研究从两方面判定规避样本是否保留恶意功能：

修改过程控制：

避免二进制修改中的冲突操作（如执行添加型动作前检查空间充足性）；

通过PEfile7解析验证文件结构完整性；

行为一致性验证：

使用IDA Pro8反汇编原始样本与修改变体，生成函数调用图；

对比修改前后所有样本的函数调用图，判定恶意行为是否发生改变。

需特别指出：若生成的规避样本破坏其恶意功能，则视为攻击失败。

6.2 稀疏奖励问题

恶意软件规避检测是一项复杂任务，在成功生成规避样本前，每个动作的奖励信号可能极为稀疏，导致强化学习（RL）算法难以收敛。针对此问题，可采用以下解决方案：

软标签（soft labels）利用：

相较于硬标签（hard labels），目标模型的预测置信度可提供更丰富的奖励信息；

即使目标模型未提供软标签，亦可通过Virus Total多引擎检测率间接反映；

逆强化学习（Inverse Reinforcement Learning, IRL）技术：

从成功轨迹中自主推导奖励函数，缓解人工设计奖励的局限性。

6.3恶意软件表征：

在强化学习（Reinforcement Learning, RL）中，智能体需探索状态空间以收集信息丰富的经验。低维且高效的状态表示对强化学习性能具有显著影响。理想情况下，攻击者可将检测器的特征提取器输出作为状态空间，但这一目标在黑盒（black-box）场景下难以实现。现有两种主要方法：

原始字节序列表示：直接使用二进制文件的原始字节序列作为状态表示，但百万量级的维度易引发维数灾难（curse of dimensionality）；

专家经验特征表示：如本文采用基于专家经验手动定义特征，但存在泛化能力不足的缺陷。

潜在解决方案：利用状态表示学习（state representation learning）技术自动构建空间表示，这将作为我们未来的研究方向。

6.4泛化性：

强化学习（Reinforcement Learning, RL）智能体基于动作执行后的反馈调整策略。若动作集在实现规避方面具备完整性，则理论上可习得有效策略。由于PSP-Mal定义的动作集专为规避基于EMBER特征的静态检测而设计（包含10种可改变静态特征的功能保留型修改），该框架无法规避动态分析检测（dynamic analysis detection）。面对动态检测与真实场景的恶意软件检测器，需扩展动作集以包含以下操作：

动态特征修改：如添加无操作API（no-op API）；

高级操作：如其他复杂行为干扰技术。

此类动作集的构建将成为我们未来的研究方向。

6.5防御对策：针对PSP-Mal攻击，由于其倾向于采用\*\*添加型动作（additive actions）\*\*修改文件，防御方可检测以下内容：

闲置字节（slack bytes）修改；

文件尾部填充大量未执行字节。

然而，非可执行区域（non-executable area）范围广且位置不确定，全面检测耗时严重。因此，建议采用显著性检测技术（saliency detection technique）[32–34]，聚焦检查对分类影响显著的关键字节区域，因为这些区域可能成为攻击者的主要目标。

7 结论

本文提出沙普利先验（Shapley prior），构建了基于先验指导的强化学习框架PSP-Mal，用于生成针对PE恶意软件检测器的功能保留型规避样本。为解析黑盒模型，我们通过计算沙普利值探索特征影响，识别检测关键特征。在此基础上引入沙普利先验，表征操作的预期影响以指导对抗攻击。利用该先验信息衡量经验利用优先级，并采用汤普森采样（Thompson sampling）选择有效动作内容。在基于EMBER与SOREL-20M数据集训练的LightGBM模型上评估PSP-Mal性能，结果表明相较于最先进的恶意软件规避攻击方法，PSP-Mal显著提升了规避率。

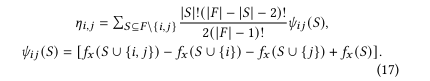
总体而言，在生成对抗样本的任务中，我们验证了先验信息引导与动作内容优化可有效提升强化学习框架性能。针对黑盒场景下的攻击，本文仅是揭示潜在信息使模型"去黑盒化"的可行路径之一，未来将进一步探索其他方法。

致谢

作者谨此感谢国家自然科学基金（No.62076251 和 No.62106281）对本研究的支持。

A. 交互效应分析

利用SHAP方法解释模型时，除获取各特征的沙普利值𝜂𝑖 外，还可计算两特征间的交互归因值𝜂𝑖,𝑗，以量化其成对交互效应。其计算公式为：



在图10的交互效应图中：

对角线区域展示单一特征与预测值的关系；

非对角线区域表征特征组合对预测的联合影响。

各子图的横坐标为沙普利值，子图宽度越大表示该特征组合对结果的贡献越显著。与图3类似，我们发现：

导出表信息（Exportsinfo）及其组合的交互归因值接近零，推测原因为：

恶意软件通常不存在导出表；

或导出表内容非标准化/无意义，无法反映恶意性。

关键发现：

包含区段信息（Sectioninfo）或字节直方图（ByteHistogram）的特征组合对结果影响显著，表明：

Sectioninfo与其他特征存在交互效应；

ByteHistogram与其他特征存在交互效应。

方法改进方向：

当前PSP-Mal基于定性分析（动作是否影响单个特征）构建先验知识；

未来计划通过统计方法量化特征间交互效应，并评估动作对特征的定量影响。