arXiv:2301.00928v1 [cs.DC] 3 Jan 2023

跨边缘到云连续体的人工智能驱动的机密计算

Sm Zobaed

提交研究生院教师审阅的学位论文，部分满足哲学博士学位的要求

路易斯安那大学拉法叶分校 2022年秋季

批准：

Mohsen Amini Salehi，主席 高级计算机研究中心

Sheng Chen 高级计算机研究中心

Raju Gottumukkala 信息学研究所

Xiali Hei 高级计算机研究中心

Li Chen 高级计算机研究中心

Mary Farmer-Kaiser 研究生院院长

© Sm Zobaed

2022

版权所有

摘要

随着技术的飞速发展，个人和组织都在广泛采用云服务，以减轻维护负担。尽管云服务具有可扩展性且易于使用，但许多拥有敏感数据的潜在云用户出于合理的数据机密性顾虑，仍不愿充分利用云服务。静态数据和传输中数据的机密性保护已得到广泛研究，但数据在云端使用期间仍然容易受到攻击。当计算范围扩展至边缘到云连续体时，这种脆弱性会进一步加剧。从安全保护的角度出发，敏感数据在离开用户场所时需要加密。虽然采用用户端加密可以确保数据机密性，但会限制数据处理能力。因此，我们只能执行低级操作（例如，字符串模式匹配）。为此，本学位论文旨在通过在整个连续体中采用机密计算来实现数据机密性。为达成这一目标，我们探索的一种方法是将数据处理中的智能方面与模式匹配方面分离。我们介绍了一种在云端实现机密数据聚类的方法，并进一步开发了面向非结构化文本数据的边缘到云机密搜索服务。我们提出的聚类方案 ClusPr 可对静态和动态数据集进行基于主题的聚类；与其他基于加密的聚类技术相比，该方案可将簇连贯性提高多达 30% 至 60%。我们提出的可信企业搜索服务 SAED，可在整个连续体中对机密数据提供上下文感知和个性化语义搜索，

评估结果经人类用户验证，表明 SAED 可将纯文本的搜索结果相关性提高 $\approx 24\%$，并将加密数据集的搜索结果相关性提高 75%。我们发现，要在边缘到云环境中实现机密计算，需要边缘层级作出重大贡献，尤其是要在边缘同时运行多个深度学习服务。这会导致边缘层出现内存争用。为解决这一争用问题，我们提出 Edge-MultiAI 框架，用于管理深度学习应用的神经网络模型，使其能够满足深度学习应用的延迟约束，同时不影响推理准确率。评估结果证实，Edge-MultiAI 可使边缘层的多租户程度提升 $2\times$ 。

谨献给我的造物主——至高的真主（*SWT*），以及教导我们人生意义的导师先知穆罕默德（*pbuh*）；献给我的父母 *Md Mazharul Islam* 和 *Jubaida Naznin*，他们以无数方式给予我支持，从未停歇；献给我的启蒙者 *Md Fazle Rabby* 和 *Md Sazib Hasan*，他们在我一路走来的风雨晴日中始终全心全意地给予我巨大的鼓舞；献给我自幼以来的引路人 *JahidMd MahabubIslam*，他如同激光雷达，指引我穿越黑暗之海；献给我心中永恒的思念 *Farzana Hasan*，对她我永怀感激；献给我的导师 *Muhammad Usama Islam*、*Md Istiaq Hossain*、*Jubair Yusuf* 和 *Farhan Tanvir*；献给我唯一的姐姐 *Mantika Mahbuba*；也献给我所有的朋友和挚爱之人。

致谢

我衷心感谢我的导师 Mohsen Amini Salehi 博士，自这段旅程开始以来，他始终给予我鼓励。他为我提供了大量指导、支持与协助。感谢我的学位论文委员会成员 Raju Gottumukukkala 博士、Li Chen 博士、Xiali Hei 博士和 Sheng Chen 博士。感谢 Jason Woodworth、Razin Farhan Hussain、Davood Ghatreh Samani 和 Chavit Denninart 在学位论文工作中给予的帮助。特别感谢 Ali Mokhtari 在近期工作中的合作。最后，感谢路易斯安那大学拉法叶分校高级计算机研究中心和研究生院给予我的支持与指导。

目录

摘要 .iii

献词 .v

致谢 .vi

表格目录 .xi

插图目录 .xii**第 1 章：引言** .1**1.1 动机：当前时代的数据机密性** .1**1.2 维护数据机密性的本质** .1**1.3 机密计算** .3**1.3.1** 基本定义 .3**1.3.2** 跨边缘到云连续体的机密计算 .3**1.3.3** 非结构化数据的机密计算 .6**1.4 研究问题与目标** .8**1.5 贡献** .8**1.6 论文结构** .10

第2章：背景与文献研究.14

2.1 背景 .14**2.1.1** 可信执行环境 .14**2.1.2** 机密计算的可信基础设施 .16**2.1.3** 基于云的非结构化文本数据企业搜索服务 .18**2.1.4** 边缘到云连续体的兴起 . . . . . 19**2.1.5** 面向非结构化数据驱动应用的机器学习 / 深度学习 .20**2.1.6** 面向延迟敏感型处理的边缘多租户 .21**2.2 非结构化数据机密计算的相关文献** . 23**2.2.1** 隐私保护的非结构化数据聚类方案 .23**2.2.2** 可搜索加密与加密索引 25

2.2.3 隐私保护云端搜索系统 . . . . . 272.2.4 用于隐私保护非结构化数据处理的边缘计算 .30**边缘多租户人工智能执行的相关文献** .312.3.1 边缘人工智能 . 312.3.2 边缘多租户执行 .312.3.3 深度神经网络模型压缩 .322.3.4 热启动与冷启动深度学习推理 . . . . . 34**2.4 本学位论文的总结与定位** .34

C**第3章：非结构化云数据的隐私保护聚类**.36**3.1 概述**.36**3.2 问题陈述**.36**3.3 所提出的聚类研究的定位**.37**3.4 促进安全搜索系统中聚类的架构**.383.4.1 架构: ***ClustCrypt***.383.4.2 架构: ***ClusPr***.40**3.5 ClustCrypt：静态非结构化数据的隐私保护聚类方案**.453.5.1 估算静态数据集的簇数.453.5.2 确定簇中心. 523.5.3 令牌聚类.54**3.6 S-ClusPr：静态非结构化数据集的隐私保护聚类方案**. 563.6.1 中心选择.563.6.2 在簇间分配加密词元. . . . . 593.6.3 剪枝簇以加快搜索操作.62**3.7 动态非结构化数据的隐私保护聚类方案**.633.7.1 概述.633.7.2 半动态数据聚类方案（***SD-ClusPr***）.643.7.3 全动态数据聚类方案（***FD-ClusPr***）. 68**3.8 所提出聚类研究的安全性分析**.69

3.9 聚类性能评估.733.9.1 实验设置.733.9.2 先前研究的评估指标与基线方法.743.9.3 评估结果.76**3.10 总结**.91

第4章：云端隐私保护企业搜索的基于边缘的智能 .92**4.1 概述** .92**4.2 问题陈述** .924**.3 SAED：智能边缘赋能企业搜索系统** . . . . 944.3.1 架构概述 .944.3.2 查询上下文识别 .964.3.3 查询扩展单元 .994.3.4 用户兴趣检测 .1004.3.5 加权单元 .1024.3.6 排序单元 .104**4.4 SAED作为企业搜索解决方案的可插拔模块** .107**4.5 SAED的性能评估** .1084.5.1 实验设置 .1084.5.2 基准查询 .1094.5.3 评估指标 .1094.5.4 评估搜索相关性 .1114.5.5 隐私保护企业搜索的相关性.1144.5.6 相关性结果讨论 .1154.5.7 评估搜索时间 .116**4.6 总结** .118

第5章：延迟敏感型深度学习
应用程序 **on Edge** . . . 119
5.1 概述 **w** . . . 119
5.2 问题陈述 . . . 123
5.3 解决方案陈述和贡献 .125
5.4 Edge-MultiAI 的架构概述与系统设计 ... 126
5.5 管理多租户应用模型的启发式方法 ... 128
5.5.1 概述 . . . 128

5.5.2 策略*1*：最大优先模型驱逐（*LFE*） 131
5.5.3 策略*2*：最佳适配模型驱逐（*BFE*） 132
5.5.4 策略*3*：感知热启动的最佳适配模型驱逐（*WS-BFE*） .132
5.5.5 策略*4*：智能感知热启动的最佳适配驱逐（*iWS-BFE*） . 133
5.6 性能评估 .137
5.6.1 实验设置与评估指标 137
5.6.2 *Edge-MultiAI*对多租户程度的影响 .139
5.6.3 驱逐策略对冷启动推理的影响 . 140
5.6.4 驱逐策略对推理准确率的影响 .141
5.6.5 神经网络模型驱逐策略的双目标分析 .142
5.6.6 分析对不确定性的鲁棒性 144
5.6.7 评估神经网络模型驱逐策略的公平性 .145
5.7 总结 .148
第6章：结论与未来研究方向 149
6.1 讨论 .149
6.2 未来研究方向 .151
6.2.1 非结构化数据的层次聚类 151
6.2.2 从加密簇构建分类器 152
6.2.3 在机密搜索中引入弹性 152
6.2.4 在模型管理方案中增加能耗 153
6.2.5 面向延迟容忍型应用程序的云端卸载.153

参考文献 .154

个人简历 .166

表格目录

表3.1. 现有隐私保护聚类方法综述，并将我们提出的方法（ClustCrypt 和 ClusPr）与之进行比较.38

表3.2. 基于索引结构构建的词元-文档频率矩阵 **A**.46

表3.3. 归一化词元-文档矩阵 **N** .47

表3.4. 基于归一化矩阵 **N** 构建的矩阵 **R**，用于表示每个词元在所有文档中的重要性 48

表3.5. 由 **N** 构建的矩阵 **S**，用于表示每个文档相对于每个词元的重要性.49

表3.6. 基于矩阵 **R** 和 **S** 相乘构建的簇决策矩阵 **Q** .50

表3.7. 数据集的 Calinski-Harabasz 指数.81

表3.8. 针对每个受研究数据集的基准查询. 86

表 4.1. 为 RFC 和 BBC 数据集开发的基准搜索查询. .109

表4.2。 比较采用 SAED 的企业搜索系统与其原始版本所获得的平均 F-1 分数和平均 TSAP@10 分数。最高分数以粗体显示。 . . . . . 115**表 5.1**。 在三星 Galaxy S20+ 上单独运行的热门神经网络模型的加载时间、推理时间和准确率，该设备用作边缘服务器.121**表 5.2**。 实验中采用的具有不同精度变体的应用专用模型.137

插图目录

图1.1。 在边缘云系统上执行机密计算的高级工作流程图。底部箭头表示整个连续体上的信任程度.5

图1.2. 用于实现智能机密企业搜索服务的基于用户—边缘—云的三层架构高层示意图.6

图1.3. 章节与相关贡献度之间的相互关系.10

图2.1. 可信执行环境构建模块的高层架构概述。 16

图 2.2. 机密计算范围的分类体系。 17

图2.3. 云中加密大数据搜索不同类型的分类体系.26

图3.1. 集成 ClustCrypt 方法的高层双层（客户端-云）搜索系统架构.40

图 3.2. ClusPr 部署于客户端、边缘和云三层架构中，以提供安全云端搜索服务的上下文概述。假设边缘层位于用户 premises 中并受到信任。它用于降低隐私和聚类相关进程带来的计算开销. 43

图3.3. 表示中心与其余词元之间相关性的二分图。每条边的权重表示词元与中心的相关性。实线表示为某个词元提供最大相关性的中心。 .60

图 3.4. 各数据集的轮廓系数（SC）指标。结果来自 S-ClusPr、*HK*-均值++、ClustCrypt（基于加密的聚类方案）、W2V-Kmeans 和 WordNet 聚类方案（针对明文词元运行）。.78

图3.5. 使用不同聚类方案时各数据集的戴维斯-布尔丁指数（DI）。.79

图 3 .6. 各数据集的簇一致性。 .80

图3.7. 比较在所研究的数据集上使用 S-ClusPr 进行聚类与使用原始 S3BD 聚类的影响。 .84

图 3.8. 比较 BBC数据集 中使用 S-ClusPr 与原始 S3BD 聚类时搜索结果的相关性。相关性数值根据 TSAP@10 评分指标计算。 .85

图3.9. 使用 S-ClusPr 进行聚类与使用原始 S3BD 聚类时 S3BD 的搜索时间比较。 . 88

图3.10. 对三个研究数据集的不同更新应用 SD-ClusPr（启用和不启用重新聚类选项）时簇的一致性.88

图4.1. SAED 系统在边缘层中的架构概述，以及其作为三层企业搜索服务一部分的架构概述。SAED 通过识别查询上下文并将其与用户兴趣相结合来提供语义搜索。随后，SAED 的查询扩展与加权单元分别融入语义并确保结果的相关性。实线和虚线分别表示用户与云层之间，以及云层与用户之间的交互.95

图4.2. 比较 SAED+S3BD 系统与 S3BD 系统的 TSAP@10 分数。横轴表示基准查询. 112

图4.3. 比较 SAED+Kendra 与 AWS Kendra 在搜索基准查询时获得的 TS AP@10 分数.112

图4.4。 比较 SAED+Kendra 与 AWS Kendra 系统在加密域中获得的 TSAP@10 分数.113

图 4.5。 S3BD、Kendra、SAED+S3BD 和 SAED+Kendra 系统之间的搜索时间比较.117

图5.1。 SmartSight 的概览。SmartSight 是一个基于物联网的系统，持续接收来自智能眼镜（物联网设备）传感器的各种输入，并通过运行在边缘服务器上的多租户深度学习应用程序对其进行处理.120

图5.2。 Edge-MultiAI 框架的架构概述，包含三个层级：应用程序、神经网络模型管理器和内存。 .126

图 5.3. 五个多租户应用（即 A_1 至 A_5）的推理请求示例场景。每个脉冲表示预计会收到推理请求的时间窗口。实线表示已经发生的事件，“现在”之后的虚线表示请求预测.130

图 5.4. Edge-MultiAI 及其 iWS-BFE 驱逐策略对满足所请求多租户需求的影响。大图通过增加横轴上所请求多租户数量的均值，并在纵轴上显示已满足请求的百分比，呈现汇总分析。对于每种情况，小图更细致地展示仿真期间发出并完成的并发请求数量.138

图5.5. 测量采用所提出的驱逐策略后，多租户应用程序的冷启动推理百分比。横轴表示预测的推理请求时间与实际推理请求时间之间的偏差.140

图 5.6. 测量采用不同驱逐策略后应用程序的归一化推理准确率。. . . . 141

Figure5.7. 不同模型选择策略的双目标分析。 . . . 143

图 5.8. 系统应对推理请求预测不确定性的鲁棒性.145

Figure5.9. 采用不同神经网络模型驱逐策略与无策略时的冷启动推理百分比.146

图5.1 不同策略下获得的推理准确率。“最大值”是基准，显示了每个应用程序中最高精度模型的准确率.147

第 1 章：引言

1.1 动机：当今时代的数据机密性

如今，得益于信息与通信技术的普及，全球超过半数的人口已接入互联网，这些技术塑造了当今的数字世界。数字化的迅猛发展产生了海量的各种形式的数据。据估计，每天会产生 2.5 艾字节的数据，其中超过 80% 是非结构化形式的数据（例如，音频、流媒体、文本）[1]。数据的范围很广，从个人的姓名到生物特征信息、执法记录、医疗报告等敏感（又称机密）信息，不一而足。必须始终妥善保护此类机密数据，以防止未经授权的访问。例如，目前大多数智能手机都采用基于生物特征的安全协议，因此会保留生物特征数据，以便在确认授权后解锁设备。如果这些生物特征信息因数据泄露而遭到泄露，便可能助长犯罪分子盗用身份、伪造文档和实施犯罪。

1.2 维护数据机密性的本质

在本地存储数据时（即静态数据），如何保持数据机密性是一个广为人知的问题，已有许多成熟的加密解决方案 [2, 3, 4, 5]。另一方面，传输层安全（TLS）协议等解决方案已在全球广泛采用，用于应对节点间传输过程中如何保持数据机密性的挑战（也称为传输中数据）。

另一种需要保护的数据状态称为使用中数据，指的是在用户访问和处理数据时，确保数据机密性和安全。虽然采用加密技术可以在数据存储或传输时提供安全保障，但无法确保数据在内存中使用时的隐私。与其他两种状态相比，数据在计算过程中最为脆弱（即，处于使用状态时）。如果机密数据的所有者是个人或机构，且依赖云服务满足其存储需求，数据的脆弱程度还会进一步加剧。

云服务提供商（例如，AWS、Azure、Google Cloud）纷纷推出各种服务，用于大规模存储和处理数据。然而，由于对云数据中心数据隐私问题的担忧确有其依据，机密数据所有者仍对采用云服务犹豫不决 [3, 6] 。事实上，采用云服务会增加与数据无处不在的访问相关的风险。云服务还会扩大攻击面，使其容易遭到入侵者利用。因此，云一直是近年来多起隐私泄露事件的目标平台[7, 8]。在一起引人注目的事件中，超过 30 亿名雅虎用户的机密信息遭到泄露 [9]。另一起事件中，超过 1,400 万个 Verizon 客户账户的信息从该公司的云系统中泄露 [10]。

鉴于这些事件，目前，从个性化医疗、搜索、档案和金融，到社交网络（例如，Twitter、脸书）及物联网行业的众多应用程序，都面临着类似的基于云的数据泄露威胁 [7]。即使云服务提供商能够提供严格的安全控制来抵御

外部威胁，处理敏感内容的订阅用户仍心存顾虑，因此由于可能遭遇内部人员攻击，无法完全接受云服务。因此，在全球 IT行业，迫切需要确保在数据所有者无法完全控制的各类系统中（从用户设备到云，乃至多云环境）进行机密数据处理的安全。

1.3 机密计算

1.3.1基本定义

目前已有众多解决方案可确保静态数据和传输中数据的机密性。然而，保护使用中的数据的机密性仍是一个有待进一步关注的开放性问题。针对这一问题，近年来出现了机密计算这一理念，并催生了由硬件强制实施的可信执行环境（*TEE*）系统，可在不损害数据隐私的情况下进行安全计算（例如，数据处理）。TEE允许用户级代码分配称为安全区的私有内存区域，在不信任操作系统或虚拟机监控程序的情况下对数据进行机密处理 [11, 12]。因此，它能够防止应用程序和数据在“使用中”遭到未经授权的访问或篡改。通过这种方式，机密计算增强了数据安全保障。近年来，机密计算的应用案例在工业界和学术界都日益普及，预计这一概念的整体市场规模将在未来五年内至少增长 26× [13]。

1.3.2边缘到云连续体中的机密计算

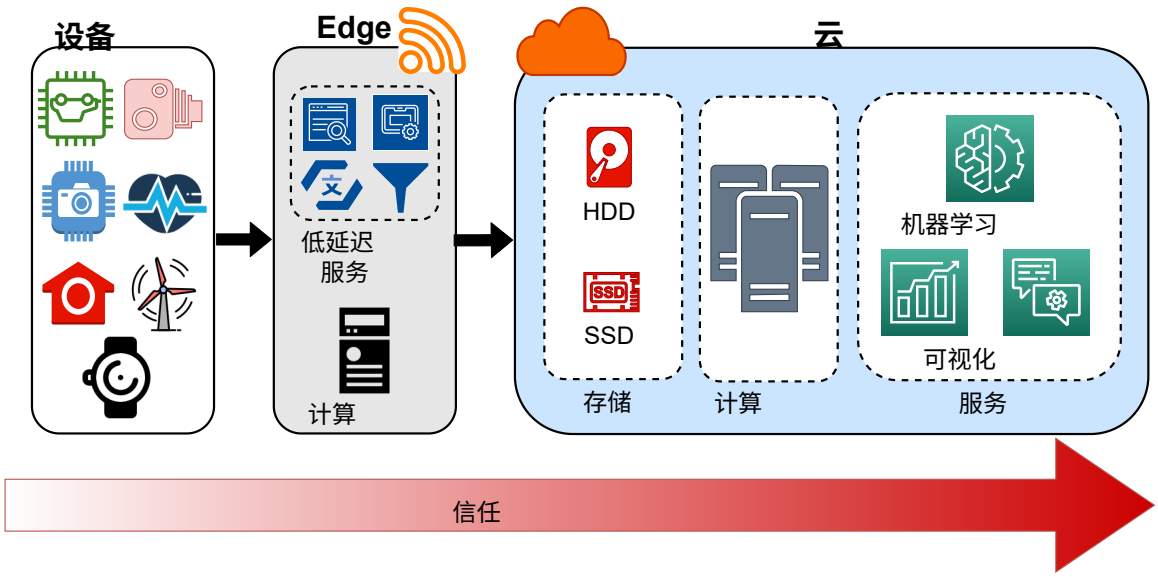
要成功存储并

处理大量数据，采用云服务几乎不可避免；然而，由于来自云系统内部和外部的威胁同时存在，机密数据所有者无法信任云，也不能毫无顾虑地使用其服务。因此，云上机密计算的目标被定义为：在公共领域内为用户提供安全访问第三方云计算服务的途径。然而，除了安全方面的问题，由于云具有集中式的特性，也会带来较高的通信延迟，这可能会对许多有延迟约束的基于物联网的解决方案造成不利影响 [14, 15, 16]。因此，过去几年间出现了一种新的计算范式，它超越了传统云系统，涵盖从设备层到边缘、雾和云的连续计算层级 [14, 17, 15, 16]。

设备到云连续体使漏洞面扩展到了云之外，因此，必须将机密计算解决方案扩展到整个连续体，以确保基于物联网的系统的完整性。图1.1展示了一个计算连续体，其中应用程序横跨用户设备、边缘和云。用户生成的数据首先在设备层（例如，物联网设备）上进行预处理；然后，根据低延迟和资源需求，由边缘层和云层上的服务进行处理。

在我们的愿景中，跨越边缘到云连续体的机密计算被定义为在应用程序跨越整个连续体使用用户数据时，保护这些数据的完整性和机密性。在整个连续体中提供机密计算面临的一项挑战是，设备层（例如，

图1.1。在边缘云系统上执行机密计算的高级工作流程图。底部箭头表示整个连续体上的信任程度。



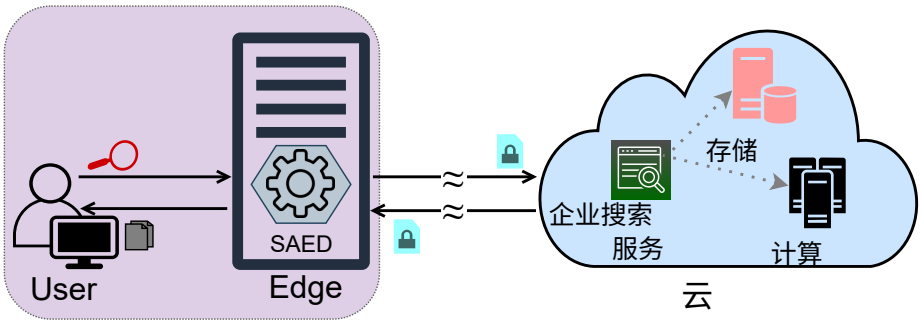
无人机 [18]和智能眼镜 [19]）以及边缘层（例如，智能手机、伴随设备）通常受资源和能耗限制，无法充分运行机密计算所需的可信应用。整个连续体的可信性是另一项必须克服的挑战。这是因为数据一旦离开用户端，漏洞面就会扩大（图**1.1**），因此，随着数据被发送到边缘层和云层，信任程度也会降低。此外，还会采用客户端加密 [20]等各种加密技术，在用户本地加密数据，从而确保在使用任何云服务时数据的机密性。这是因为在这种情况下，云服务提供商无法解密数据。然而，无法解密数据也会妨碍对数据的访问。我们旨在通过本学位论文应对这些挑战。具体而言，本学位论文将探讨以下方法

在边缘云中启用机密计算，同时考虑：(a) 连续体中每个层级的可信度级别；以及 (b) 应用程序的低延迟约束。

1.3.3非结构化数据的机密计算

拥有海量机密非结构化文本数据的组织，希望有一个在机密计算平台上运行的可信应用程序，以低延迟的方式为这些数据提供安全语义可搜索性。例如，执法机构持有加密犯罪报告数据，执法人员需要在办公室或外出时使用手持设备低延迟地搜索这些报告。在机密非结构化数据处理的上下文中，已经开发出各种可搜索加密系统（例如， [21, 22, 23, 24]），以便对加密数据进行安全搜索。使用加密数据时，这些系统会构建加密索引，并在搜索时根据搜索查询遍历该索引，从而找到相关文档。

图1.2。 用于实现智能机密企业搜索服务的基于用户-边缘-云的三层架构高级示意图。



针对给定的搜索查询，对整个索引进行穷尽式搜索

会使搜索操作无法满足低延迟约束。因此，需要进行索引分区（又称聚类），以缩小搜索空间，从而能够以最小开销在经过裁剪的索引上执行搜索。聚类是一种重要的数据分析方法，常用于根据数据点的共有属性对其进行分组。因此，对搜索索引应用聚类后，我们可以将其划分为多个子集，从而将搜索时间开销降低数个数量级 [1, 3]。

用户（例如，执法人员）可能不记得他们要查找的文档中包含哪些具体关键词。因此，他们需要检索与所给搜索查询在语义上和上下文上相关的文档。例如，执法人员搜索“抢劫”时，也可能希望找到有关“行凶抢劫”“盗窃”或“入室行窃”的文档。此外，由于执法人员在外出时会使用资源有限的手持设备进行搜索，因此解决方案应尽可能降低处理开销，并能够良好扩展，以处理海量非结构化文本数据。为此，当下亟需一种以可信应用形式实现的稳健、安全的企业搜索服务，以便对加密机密数据进行语义搜索。图1.2展示了安全企业搜索服务的高级架构。此类服务可以利用本地边缘资源提供安全搜索智能。现有搜索系统则利用云层中的高端存储和计算资源，在整个数据集上全面执行模式匹配。

1.4 研究问题与目标

为促进边缘到云连续体中的机密计算，本学位论文探讨以下研究问题：

1. 如何开发一种可信应用程序，以便在加密非结构化数据集上以最佳方式、可扩展且安全地对关键词进行聚类?

2. 当数据集具有动态性，即其中的内容不断增加或删除时，如何对数据进行聚类?

3. 如何以最低开销实现对加密数据的安全语义搜索?

4. 如何开发可信、健壮的加密企业搜索服务 ?

5. 如何管理可信深度学习应用程序的神经网络模型，以促进其并发执行，同时不损害其推理准确率?

1.5 贡献

鉴于上一节概述的研究主题，本学位论文作出以下重要**贡献**：

1. 提出两种可信应用，用于在云端对加密的非结构化数据进行机密聚类：(1) ClustCrypt——仅云端架构，以及 (2) ClusPr——边缘云架构。ClustCrypt 能够估算合适的簇数（K），随后仅对加密静态数据进行聚类；而通过引入边缘计算，ClusPr 能够进一步对包含动态变化的数据进行聚类

。在三个不同的测试数据集上，将 ClusPr 与文献中的其他方案进行比较，结果显示簇一致性提升了 30% 至 60%。此外，我们发现，在隐私保护企业搜索系统中采用 ClusPr，可将搜索时间最多缩短 78%，同时将搜索准确率最多提高 35%。

2. 提出一种开源搜索机制（名为 $SAED$），通过将搜索的智能方面与模式匹配方面分离，解决隐私问题。在 SAED 中，搜索智能由本地边缘层提供，共享云层仅作为全面的模式匹配搜索工具。借助边缘层，SAED 可在满足低延迟约束的同时，在现有的基于云的企业搜索服务上实现个性化语义可搜索性，并维护数据隐私。在真实环境下进行的评估以及人类用户的验证表明，对于加密的通用数据集，SAED 能够在搜索时间开销可忽略不计的情况下，将检索结果的相关性提高 $\approx 75\%$。

3. 提出一种名为 $Edge\text{-}MultiAI$ 的神经网络模型管理框架，以支持机密深度学习应用在可信边缘服务器上持续执行，从而避免云端执行的风险。这是因为，可信应用的神经网络模型无法外包给公有云。借助近似计算，Edge-MultiAI 能够高效利用边缘内存，从而最大限度地提高多租户程度

，且不会对推理操作造成重大影响。Edge-MultiAI 会为请求应用程序动态加载高精度神经网络模型，同时为其他应用程序加载低精度模型。该框架在 Edge-MultiAI 中提出了 $iWS\text{-}BFE$策略以及另外三种基线启发式策略，用于为执行推理的应用程序选择合适的模型，并决定如何为其分配内存。实验表明，Edge-MultiAI 至少可以将边缘的多租户程度提升 $2\times$ ，且不会显著降低推理准确率。

1.6 论文结构

图1.3。 章节与相关贡献度之间的关系 n.

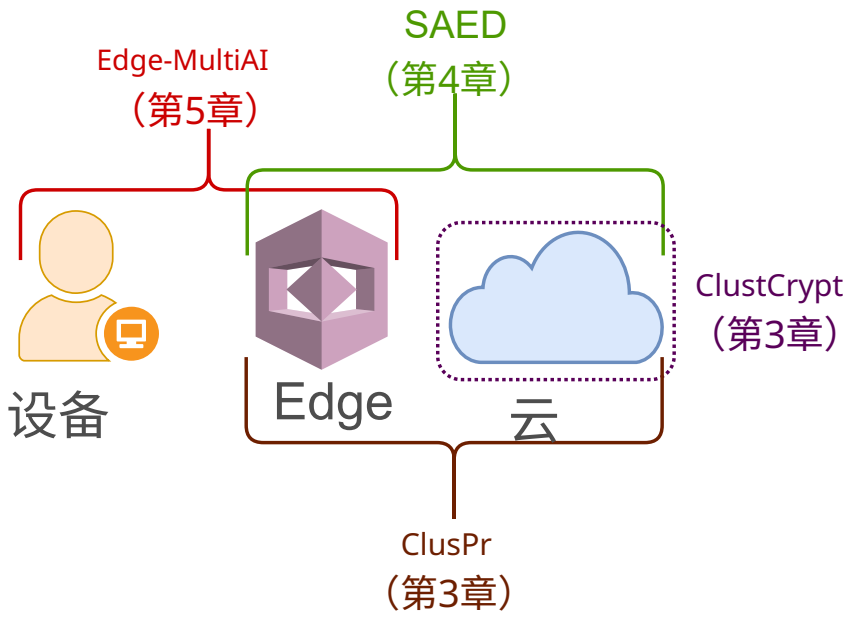


图1.3展示了各章节之间的关系，以及对.....的贡献

与之相关的内容。本学位论文的核心章节源自博士候选阶段期间发表的多篇研究论文。

• 第2章介绍以下背景：边缘-云连续体的兴起、可信计算层级、企业搜索服务和机密机器学习，并探讨相关研究工作。

– **Sm Zobaed**、Mohsen Amini Salehi，《云中的大数据》，发表于《大数据百科全书》，施普林格，ISBN：978-3-319-32009-0。– **Sm Zobaed**、Md Enamul Haque、Md Fazle Rabby、Mohsen Amini Salehi，《*Senspick*：用于词义消歧的词义选择》，发表于第15届IEEE国际语义计算会议（ICSC'21）会议论文集，线上，2021年。

– **Sm Zobaed**、Mohsen Amini Salehi，边缘到云连续体上的机密计算综述，准备投稿。

• 第3章探讨聚类隐私保护文本大数据的优势。加密后，数据的语义会丢失。不过，可以利用数据的统计特征对数据进行主题聚类。本章讨论我们提出的加密静态和动态数据聚类方法。此外，本章采用常用的聚类质量指标（例如，轮廓系数、Davis-Boudin指数），比较该方法与其他方法得到的簇。最后，本章介绍在现实环境中开展的一系列实验，以展示该聚类方法的有效性。

– **Sm Zobaed**、Sahan Ahmad、Raju Gottumukkala、Mohsen Amini Saleh i,

Clustcrypt：云中非结构化大数据的隐私保护聚类，发表于第21届IEEE高性能计算与通信国际会议（HPCC'19）论文集，中国，2019年。（完整代码见 GitHub 存储库：https://github.com/hpcclab/ClustCrypt）。

– Sahan Ahmad、Sm Zobaed、Raju Gottumukkala、Mohsen Amini Salehi，面向基于云的加密大数据的用户中心安全搜索的边缘计算，发表于第21届 IEEE 高性能计算与通信国际会议（HPCC'19）论文集，中国，2019年。

– Sm Zobaed、Mohsen Amini Salehi，面向基于云的企业搜索解决方案的非结构化大数据隐私保护聚类，发表于《并发与计算：实践与经验》（CCPE），第34卷，第22期，2022年。（完整代码见 GitHub 存储库：https://github.com/zobaed11/Jorunal-Version）。

• 第4章探讨了针对加密数据进行安全个性化语义搜索的重要性。本章阐述了所提出的安全搜索服务架构的工作流程和机制。最后，本章介绍了在 AWS Kendra 服务环境中开展的一系列实验，以展示搜索相关性的有效性。

– Sm Zobaed、Mohsen Amini Salehi、Rajkumar Buyya，*SAED*：

面向云端隐私保护企业搜索的基于边缘的智能，发表于第21届ACM/IEEE集群云与网格计算国际会议（CCGrid '21）会议论文集，澳大利亚，2021年。（完整代码见 GitHub 存储库：https://github.com/hpcclab/SAED-Security-At-Edge）

• 第5章探讨边缘服务器上延迟敏感型深度学习应用的多租户执行。本章介绍 Edge-MultiAI 的架构概述，以及 Edge-MultiAI 中用于管理多租户深度学习应用模型的启发式方法。

– Sm Zobaed、Ali Mokhtari、Jaya Prakash Champati†、Mathieu Kourouma、Mohsen Amini Salehi，*Edge-MultiAI*：边缘端延迟敏感型深度学习应用的多租户，已被第15届ACM/IEEE效用计算与云计算国际会议（UCC'22）论文集接收，美国，2022年。（完整代码见 GitHub 存储库：https://github.com/hpcclab/SAED-Security-At-Edge）

• 第6章通过讨论我们的主要发现总结了这篇学位论文，并探讨了本研究过程中涌现但本文尚未讨论的进一步研究主题和方向。

第2章：背景与文献研究

本章介绍了背景，并综述了与边缘到云机密计算最相关领域开展的其他研究工作。

2.1 背景

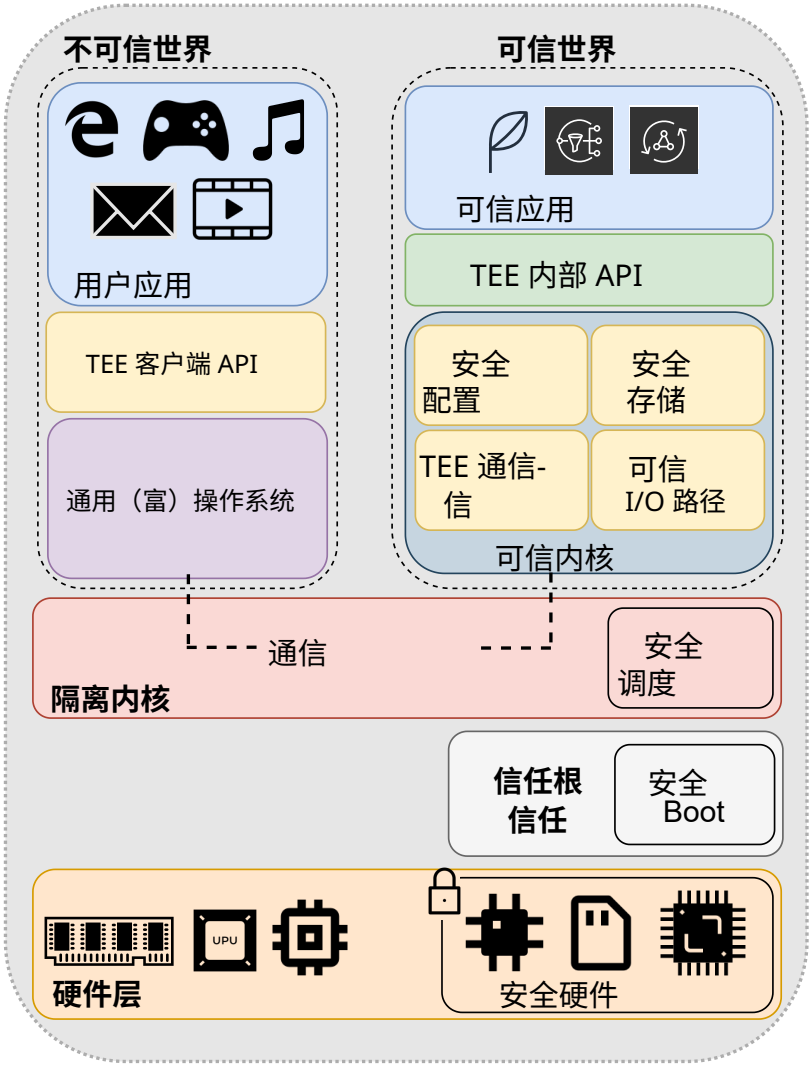
2.1.1可信执行环境

可信执行环境（TEE）是一种具有防篡改能力的处理环境，可用于运行可信应用程序，例如生物特征认证、针对加密数据的隐私保护搜索等。它通过可信硬件（即，安全处理器）实现硬件强制隔离。TEE拥有自己的内存区域，可信应用程序（TAs）驻留其中并实现完全隔离，旨在防止来自通用（又称不可信）空间的未授权访问、防止软件攻击者（例如，恶意软件、被攻陷的操作系统）进行操纵，甚至防止对平台具有物理访问权限的硬件攻击者（例如，侧信道攻击）发动攻击。TEE被视为机密计算的内核，因此，近年来可信执行环境技术的发展已将解决方案范围从微控制器扩展到大型服务器。广泛采用的可信执行环境技术包括Intel SGX、AMD SEV和ARM TrustZone。Intel SGX和AMD SEV为服务器端和个人计算机提供可信执行环境支持，而基于ARM TrustZone的可信执行环境则专为资源受限设备（例如，边缘设备、智能手机和Raspberry Pi）而设计。

在图 2.1 中，我们给出了可信执行环境组件的高层架构概述。通常，TEE会为所有可信和

通用应用程序维护两个独立的空间，即可信世界和不可信世界。可信世界包含一个可信操作系统或内核，通过 TEE 内部 API 与可信应用通信；而不可信世界中的通用应用程序则通过 TEE 客户端 API 与可信世界通信。此外，TEE 还可提供利用密封抽象实现的安全存储（例如，GPTEE、SGX）；用于在可信应用与输出显示器之间建立安全路径的可信用户界面 API；以及用于通过 POSIX 风格套接字发起 TEE 网络连接的安全配置 API。

图 2.1. 可信执行环境构建模块的高层架构概述。

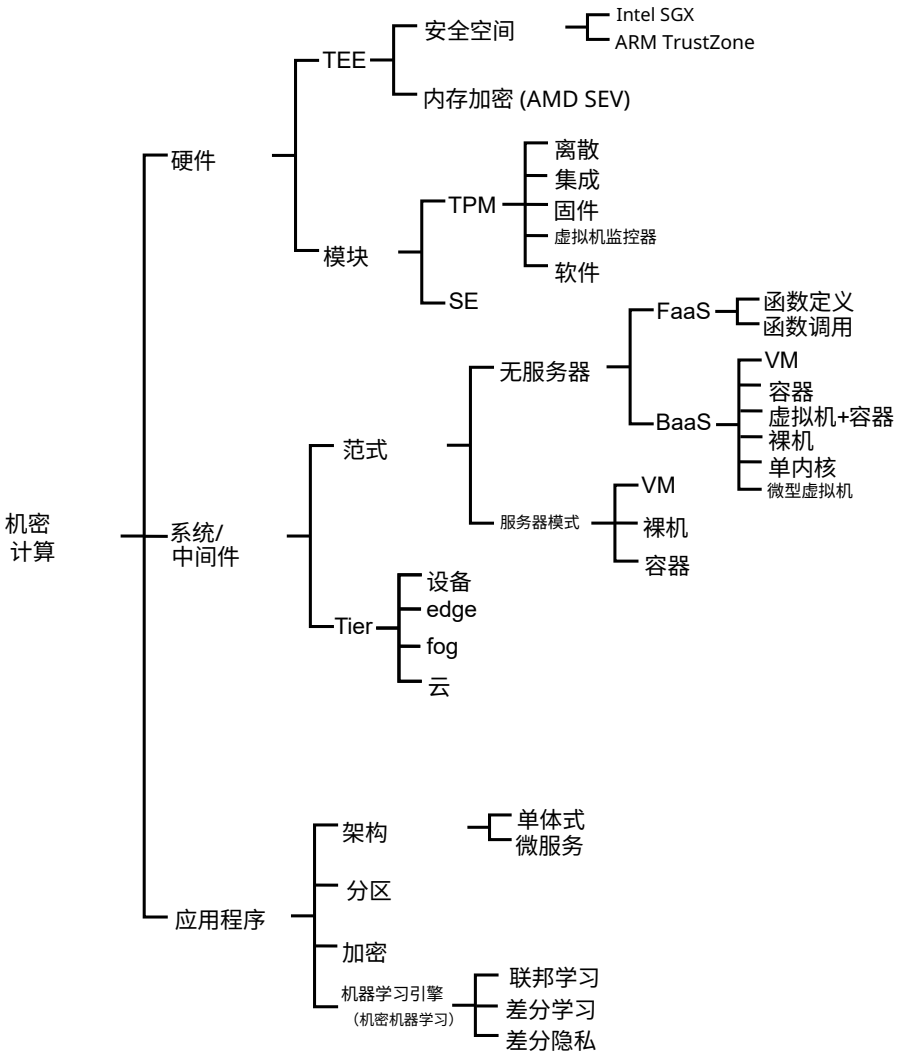


2.1.2机密计算的可信基础设施

为了部署机密计算流水线，必须确保硬件、中间件（操作系统）和应用层的可信性。执行过程中，任何一层遭到篡改都可能导致机密性遭到破坏。图 2.2 展示了高层次机密计算实施范围的分类体系。针对大规模数据中的机密数据，云层上的机密计算

至关重要，因为机密性遭到破坏的风险始终处于高位。尽管机密计算提供了隔离执行环境，但在执行任何操作之前，相关硬件、操作系统和应用程序都应由第三方（即 可信机构）在本地或远程进行远程证明。在 [25], Valadares等人为防止采用 Intel SGX 的边缘物联网系统遭受硬件攻击（例如 侧信道攻击），提出了不同的远程证明机制。

图 2.2. 机密计算范围的分类体系。



2.1.3基于云的企业搜索服务：面向非结构化文本数据

为大数据提供访问和搜索能力至关重要，缺少这些能力的数据几乎没有用处。然而，部署云服务来处理大数据的组织担心数据泄露（[1, 3]）。因此，需要在不暴露数据的情况下访问数据。企业搜索 云服务正变得越来越受欢迎，使用户能够搜索组织大数据并获得合法访问权限（[26]）。企业搜索服务通常会基于对组织文档的及时爬取来维护动态索引结构。然后，系统会根据该索引结构搜索用户查询，并向合法用户显示引用相关文档的结果集。

亚马逊云通过采用机器学习和自然语言处理方法，提供了一种名为 Kendra 的语义企业搜索服务。亚马逊表示，与其他现有解决方案相比，其客户（如 Woodside、3M 和 Sage）提升了搜索和访问组织文档的准确率与速度([27])。语义可搜索性是以牺牲用户数据隐私为代价的 [3, 17, 1]。事实上，这就是一种陷门，尤其是内部攻击者可以利用它来侵犯用户数据的机密性，甚至完整性。我们致力于让基于云的企业搜索服务能够抵御此类攻击模型。我们注意到，对于加密数据集，当前的企业搜索服务除了朴素的字符串匹配之外，别无他法¨。

我们注意到，目前 Amazon Kendra 不支持在使用用户密钥加密的数据集上提供企业搜索服务（即用户端加密）。这使得云时代的组织数据隐私问题仍悬而未决。为解决这一问题，人们提出了多种方案，以支持对用户端加密的大数据进行语义搜索([3],[1],[17])。这些方案旨在实现实时搜索操作，同时不损害数据隐私。

即使对于明文数据集，我们的调查也发现，Kendra 在搜索中仅涵盖本体语义，无法提供上下文感知和个性化语义。例如，我们通过将 `soccer` 作为查询输入，测试 Kendra 捕捉上下文感知语义的能力；结果集中出现了关于 `rugby` [15] 的文档。在另一项测试中，查询 `river bank` 返回了关于 `commercial bank` 的文档，这表明搜索缺乏上下文感知能力。

Microsoft Azure 认知服务 提供多种不同的应用程序接口，用于在 Azure 云上执行各种实用的自然语言处理任务，包括情感分析、对话式人工智能和翻译器。这些服务支持使用可定制模型和预训练模型，并可根据需求或通过竞价型实例 [28] 部署到任何位置。

2.1.4边缘到云连续体的兴起

边缘计算范式 [12, 29, 30]因其延迟敏感特性能够确保安全的实时数据处理，因而得到广泛采用。然而，由于处理能力受限，边缘节点仅能

处理少量（即 轻量级）数据。因此，边缘范式无法以独立方式有效处理海量敏感数据，应用开发者和数据所有者便转而采用云。尽管针对边缘到云连续体在实时处理、可扩展性和输出准确性方面的性能提升，已有大量研究，但对机密数据处理能力的关注相对较少 [12]。此外，边缘计算仅适用于处理轻量级数据，因此从大数据的角度来看，除了在云上处理之外别无选择。

通常，边缘节点分散分布，攻击面较大。因此，场外边缘遭到入侵的可能性很高。近期，硬件供应商开始设计专用硬件辅助可信执行环境，使其兼容云和边缘计算基础设施。

2.1.5面向非结构化数据驱动应用的机器学习/深度学习

由于数据量庞大且复杂，传统的数据分析工具（例如框架）无法高效处理非结构化数据。我们需要采用各种基于计算机视觉和自然语言建模（NLP）的解决方案，这些方案以机器学习和深度学习架构为基础，从而能够对非结构化数据进行数据分析。近年来，视觉和自然语言处理算法取得了长足进展，例如卷积神经网络、自编码器、生成对抗网络、长短期记忆网络（LSTM）、变换器和多头注意力

机制，使得我们能够有效处理非结构化文本数据。

由于 AWS 和 Azure 等主要云服务提供商提供了人工智能服务，基于云、人工智能驱动且针对特定使用场景的数据分析工具取得了多项进展。为了构建适用于非结构化数据的机器学习和深度学习增强型应用程序，需要使用精选数据集训练模型。这样做是为了让模型能够理解底层复杂的模式、关系或高级特征。验证模型的有效性后，便可确认模型已准备好执行所述任务。随后，模型将进入推理阶段，基于实时数据进行预测分析，以生成可供采取行动的结果。

2.1.6面向延迟敏感型处理的边缘多租户

边缘服务器是基于物联网的边缘云系统中不可或缺的一部分，需要持续执行多个（也称为多租户）智能应用程序（例如，深度学习），并实现低延迟和高准确率。然而，由于内存限制，在边缘服务器上执行对延迟敏感的多租户应用程序可能会导致内存争用问题，进而降低执行速率。这是因为，深度学习应用程序会在其内核中使用庞大的神经网络（NN）模型，对从传感器接收到的输入进行推理。神经网络模型必须保留在内存中，才能实现低延迟（也称为热启动 [31]）推理操作。

否则，由于神经网络模型通常非常庞大，按需将其加载到内存中（也称为冷启动）会适得其反，并影响深度学习应用程序的延迟约束。由于边缘服务器的内存大小通常有限（例如，Jetson Nano 的内存为 4 GB [32]），在这些服务器上执行深度学习应用程序的多租户任务会导致各进程之间出现内存争用挑战 [14, 33]。因此，在多租户执行环境中，必须从应用程序可用的模型集合中动态地将合适的模型加载到内存中，既不中断其他应用程序的执行，也不导致它们进行冷启动推理。

2.2 非结构化数据机密计算的相关文献 2.2.1隐私保护的非结构化数据聚类方案

聚类对于各种自然语言处理（NLP）任务至关重要，尤其是大多数高级搜索系统的先决条件。数据加密后，仅剩数据的统计特征。因此，安全数据聚类仅基于文档集密文的统计属性进行。

为了实现对加密数据（密文）的处理，人们开展了大量研究。Zhou 等人提出了一种基于线性变换的解决方案，可在云端将查询与加密数据进行匹配，同时确保数据隐私，且无需数据所有者进行任何干预 [34]。然而，线性变换方法支持基于安全<KNN>（KNN）的查询匹配方法，但不支持聚类。这是因为聚类对线性变换后的数据并非不变。最优线性变换的前提是已知真实的簇均值，而在生成簇之前无法获得这些均值 [35]。此外，我们假设数据在传输到云端之前已完成分词和加密。因此，不同于 [34],中使用耗时的密码学计算对全部加密数据进行查询的做法，我们利用数据的统计属性，而不向云端透露其中任何有意义的部分。Sun 等人提出了一种可搜索加密方法，通过构建基于余弦相似度和 $TF \times IDF$ [36, 4] 度量运行的树形索引结构。然而，该方案无法扩展到

大数据场景，因为搜索索引可能变得非常庞大，从而影响搜索操作的及时性。我们认为，所提出的聚类方法可以作为 [36, 4] 的补充：该方法按主题将中心索引划分为多个较小的索引结构，从而提高搜索时间和效率。

同态加密已成为一种对加密数据进行计算的常用方法。人们提出了全同态加密或部分同态加密等多种同态加密变体 [37, 38] ，以便在云上实现隐私保护数据处理。Zhu 等人 [5]提出了一种基于同态加密的安全聚合与划分协议，可在点对点系统中安全地计算簇，同时不损害单个节点的隐私。不过，他们的聚类技术没有考虑数据动态性。Pang 和 Wang 提出了一种同态方案，用于保护从多个参与方上传至双云系统的外包数据 [39] ；该系统被假设处于半诚实环境，而我们则认为云在存储/处理敏感数据方面不可信 [40]。

Wang 等人提出了*HK*-Means++，将*K*值-Means聚类与寻找合适的簇数 [41] 相结合。此外，该工作利用同态加密方案来求解加密数据操作、距离和收敛计算问题。尽管我们的工作与*HK*-Means++具有可比性，但它只能对静态数据集进行聚类。此外，实验仅在一个数据集上进行，因此尚不清楚该方法在其他数据集上的表现。我们注意到，目前同态加密技术的实现

意味着较高的计算开销 [42]，这会影响搜索系统的实时响应，尤其是在处理大数据集时 [5]。

Vaidya 和 Clifton [43] 提出了一种对加密数据集进行聚类的方案，其中不同的数据属性分别存储在不同的存储系统中。随后，在各个数据存储系统中分别进行聚类。然而，这种方案耗时较长，无法满足本研究所考虑的实时约束。

针对实时隐私保护大数据处理的研究非常少。Woodworth 等人 [3] 提出的 S3BD 就是其中之一。S3BD 是一种基于云的安全语义搜索系统，利用云服务对大数据进行搜索，同时不向云服务提供商暴露任何数据。为满足大数据实时搜索的约束，S3BD 会主动剪枝，将搜索空间缩小至整个数据集的一个子集。为此，他们提出了一种对加密大数据进行聚类的方法。聚类完成后，客户端会保存每个簇的一个摘要 （代表性集合），以便在搜索时将搜索操作引导至合适的簇。

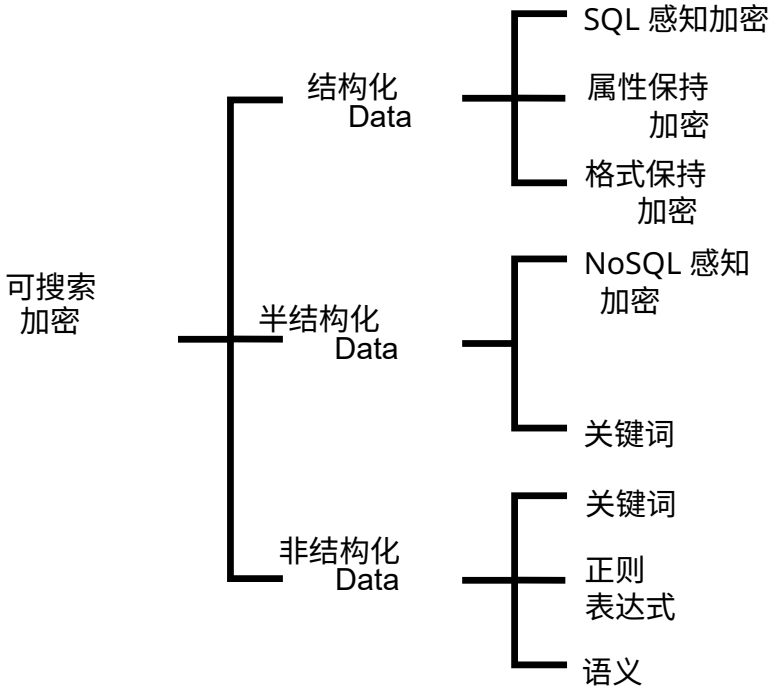
2.2.2可搜索加密与加密索引

近期已有多项研究工作致力于在云端加密数据上实现不同类型的搜索。大多数基于可搜索加密的解决方案都会生成搜索查询的密文，并以一种朴素直接的方式搜索加密文本。具体而言，给定文档中的每个词都会被独立加密，之后依次扫描文档集，同时

搜索与查询密文（加密查询） [44, 45]相匹配的内容。这些解决方案通常因无需在服务器上产生额外存储开销而受到青睐，但速度往往较慢 [3, 44]。

图 2.3 展示了云中加密数据搜索研究工作的高层分类体系。针对加密图结构数据的隐私保护查询（[46]）、Cryptdb（[47]）和 Dragonfruit（[48]）是加密结构化数据搜索的实例。SecureNoSQL（[49]）、SemiLD（[50], ）和 XSnippets（[51]）是加密半结构化数据搜索的实例。REseED（[24]）、SSE（[52]）和 S3C（[53]）分别是针对云中非结构化大数据开发的正则表达式（Regex）、关键词和语义搜索工具。

图 2.3. 云中加密大数据搜索类型的分类体系。



一些基于可搜索加密的解决方案会维护中心索引结构，以存储从每个文档中选取的数据，从而使搜索操作相对更快，并更好地适应大数据的特点 [53, 3, 52, 1, 54]。

2.2.3隐私保护云端搜索系统

除了明文数据之外，搜索也可针对隐私保护数据进行，从而确保数据泄露的可能性微乎其微。因此，人们采用了各种基于可搜索加密的解决方案，以便在此类数据中进行搜索。截至撰写本文时，只有少数研究结合了语义搜索与可搜索加密的理念。尝试提供语义搜索的研究往往只考虑词语相似性，而非真正的语义。Li 等人 [55]提出了一种系统，可通过模糊关键词搜索处理轻微的用户拼写错误。

Moataz 等人 [56] 对索引和查询中的术语采用各种词干提取方法，以实现更通用的匹配。Sun等人 [57] 提出了一种系统，使用针对加密文件元数据的索引编制方法和数据挖掘技术来捕捉查询的语义。然而，这种方法构建语义网络时仅使用给定集合中的文档，并且只将可能共同出现的词视为语义相关词，因此遗漏了许多可能的同义词或类别相关术语。S3BD [3], 是一种安全语义搜索系统，可对加密机密大数据进行语义搜索。

他们通过盲目提取本体网络中的语义数据来扩展搜索查询。他们没有考虑上下文感知查询

扩展，这导致搜索系统在处理查询中的歧义关键词或多上下文关键词时产生混淆。为了在客户端设备上执行查询处理，他们最终需要在客户端层承担额外的计算开销。

2.2.3.1 搜索查询关键词的语义表示。查询扩展是指寻找与给定查询在语义上相关的关键词，以弥合用户查询与可搜索文档之间词汇鸿沟的过程。伪相关反馈（PRF） [58, 59] 是广泛使用的查询扩展方法之一，它通过添加各种相关关键词来扩展效果不佳的查询，然后对搜索结果重新排序，从而提高检索到相关文档的可能性。尽管基于 PRF 的方法通常能提升检索效果，但它对原始搜索结果的质量十分敏感。

潜在语义分析 [60], 、潜在狄利克雷分析 [61], 和基于神经网络的语言模型 [59, 62]是一些能够获取给定查询语义表示的查询扩展方法。在这些方法中，向量通常称为词嵌入，用于将词表示在低维语义空间中；词语之间的邻近程度体现了它们在句法或语义上的相似度 [63]。然而，预训练词嵌入模型（例如 Word2vec [63],）总是为输入词生成相同的向量表示，而不考虑该词出现的上下文。因此，如果查询中存在任何含义模糊的关键词，就无法检测出该查询的潜在主题。

WordNet [64] 是英语中广泛使用且词汇丰富的资源之一，可用于推断给定语料库中歧义词的含义。在 WordNet 中，含义相近的词被归为同义词集，每个同义词集都与其他同义词集存在语义和概念关系。Song 等 [65]和

Nakade等 [66] 评估了在美国国家标准与技术研究院（NIST）和 Twitter 数据集中使用 WordNet 进行查询扩展的效果。他们识别查询中的重要关键短语，并使用 WordNet 获取相关的同义词集。随后，他们利用这些同义词集构建扩展查询。然而，在此前大多数使用 WordNet 进行查询扩展的研究中（例如， [67]），扩展查询集中的元素都被视为具有相同的权重，这会损害结果集的相关性和排序。

2.2.4隐私保护的非结构化数据处理边缘计算

为实现安全的个性化搜索，大多数企业搜索服务都依赖客户端设备的计算能力。因此，用户设备（即 瘦客户端）在执行安全查询处理或加密/解密用户文档时会承担大量开销。为此，本地边缘计算有望利用存储在客户端设备中的历史搜索数据执行个性化搜索，并按需进行加密/解密。在这种情况下，可确保本地边缘完全可信，并提供持续不中断的机密计算环境。先前的工作 S3BD [3]在针对加密大数据执行安全搜索时，会给客户端设备带来开销。本地边缘计算是此类系统的合适方案。通过在其两层架构中扩展一个本地/可信边缘，可以显著减少客户端设备的开销。

2.3 边缘多租户人工智能执行相关研究

2.3.1边缘人工智能

已有大量研究探讨了基于边缘的人工智能在无缝执行延迟敏感型智能应用方面的应用、范围和优势 [68, 69, 14, 70]。Murshed 等人讨论了多种基于深度神经网络的实际应用，例如视频分析和图像识别，以实现边缘人工智能 [69]。Zhou 等人调查了边缘设备上神经网络模型的各种训练和推理技术 [70]。Chen 和 Ran 讨论了有助于加速基于边缘的系统中的深度学习训练和推理的各种技术 [68]。Han 等人探索了加速基于边缘的架构训练收敛的方法 [14]。Wang等人从延迟和带宽的角度调查了边缘深度学习应用的发展 [71]。Zhou等人 [70] 指出，尽管更高的边缘智能水平能够减少数据卸载并提升隐私保护，但延迟和能耗开销可能会增加。

2.3.2边缘多租户执行

先前的研究探讨了边缘服务器上的人工智能多租户。Mao 等人提出了一种名为 MoDNN 的移动计算框架，可在资源受限设备上同时执行深度学习应用程序 [72]。MoDNN 可以将预训练深度神经网络模型划分到多个移动设备上，以降低设备级计算成本和内存使用量的同时加速张量处理，并实现 2.17×—4.28× 的加速比。

跨边缘服务器的多租户执行可能会导致

应用程序执行延迟过高。Ko 等人提出了 DisCo，这是一种多租户深度学习应用执行卸载框架，可让应用程序中计算密集型和数据密集型的部分在设备或边缘 [73] 上执行。Hadidi 等人指出，复杂的深度神经网络模型对数据丢失十分敏感，因为它们更加依赖数据中的细微差异 [74]。他们提到，Inception V3 模型丢失一层可能会使准确率下降 50% 以上。他们在物联网系统中采用分布式深度神经网络模型，以减少处理开销和内存占用。

上述研究工作在未考虑边缘服务器的内存约束的情况下，解决了如何加速多租户应用的问题。据我们所知，唯一的例外是 [75]，该研究探讨了在自动驾驶车辆中执行障碍物检测应用的问题：在满足超低延迟约束的同时，牺牲其他正在执行的应用。研究者提出了一种基于强化学习的技术，从非优先级应用中回收内存，从而立即执行障碍物检测应用，避免事故。尽管他们的技术能有效服务于延迟敏感型任务，但多租户执行不在其研究范围内 [75]。与这些研究不同，我们探讨内存管理问题，以提高多租户程度和热启动推理次数，进而提升基于物联网的系统的实际可用性。

2.3.3深度神经网络模型压缩

模型压缩技术使模型能够在不同的

资源受限设备上运行。降低给定深度神经网络模型复杂度的主要方法有两种：采用更低的位宽（即量化），以及减少权重数量（即剪枝）。这两种技术既可单独使用，也可结合使用，以实现模型压缩。**量化** 以准确率略微下降为代价，降低计算资源需求。默认情况下，模型权重是 float32 类型的变量，这意味着每个模型权重占用 4 字节，内存需求相当大。模型权重的位宽可以从 32 位降至 8 位（甚至更短 [33]），从而加快推理操作。

剪枝 技术用于降低模型的内存消耗，以加速推理操作。有效的剪枝技术会移除冗余连接和/或减少层的宽度，同时确保对推理准确率的影响较小。因此，需要对剪枝后的模型进行重新训练，以弥补准确率损失。未能选择合适的剪枝候选项会影响推理任务，并使剪枝后的模型失去实用价值。也有一些研究针对如何选择适当的剪枝候选项展开。

为了兼容物联网设备，Yao 等人提出了 DeepIoT [76], ，这是一种面向物联网设备中深度神经网络模型的强化学习剪枝技术。然而，在对模型参数进行剪枝时，他们只考虑了执行时间加速，因此该技术不可避免地会导致推理准确率表现较差。如上所述，激进剪枝通常会大幅降低推理准确率。在几乎不影响准确率的情况下进行高剪枝率的训练和推理，仍然是一个有待解决的研究问题

[33]。

2.3.4热启动与冷启动深度学习推理

鉴于深度神经网络模型日益复杂，即使是将压缩后的模型加载到边缘内存中，也是一项沉重负担。若边缘服务器必须在内存中持续维护多个应用程序（即 多租户），情况会更加复杂，而且成本高昂。尽管如此，仍应避免冷启动推理，因为这会导致显著的推理延迟（详情请参见表 5.1 中的加载时间 ）。已有一些研究工作致力于避免冷启动推理。例如，为了支持延迟敏感型应用，在 [77], 中，作者提出在用户浏览特定网页时，于后台冷启动深度神经网络模型。该技术利用系统资源跟踪用户的浏览活动，并在浏览过程中并行加载任务专用模型，从而避免冷启动。

2.4 本学位论文的总结与定位

以往文献既未针对机密非结构化数据处理提出支持机密计算的系统设计，也未提出资源受限边缘计算系统的低延迟约束和多租户需求。对于机密计算，我们提出在逻辑上划分系统，使其在本地边缘层执行智能处理，并利用云层执行简单的大规模处理。机密计算、聚类、可搜索加密和企业搜索领域已有大量研究。然而，这些领域交叉部分的研究却很少。

我们的贡献正是聚焦于这一交叉领域。为此，我们开展了以下工作：（1）针对机密静态和动态非结构化数据的数据聚类，可用于基于云的搜索系统等延迟敏感型系统；（2）实现跨边缘到云、针对加密机密数据的可信企业语义搜索；（3）提升边缘系统在无需非结构化云系统帮助的情况下，以低延迟执行多租户深度学习应用的能力。

第3章：面向非结构化云数据的隐私保护聚类

3.1 概述

上一章所述的初步研究证实，可以对加密数据进行聚类。基于主题的聚类能够提升各种自然语言处理任务的性能，尤其是在安全搜索系统的上下文中。然而，如何利用加密文本数据形成基于主题的簇仍是一项挑战。为了解决这一挑战，我们基于统计语义进行聚类。其思路是将语义上彼此接近的词元归入同一个簇。为此，我们首先需要确定应创建多少个簇（k），才能覆盖给定数据集中词元所涉及的主题。然后，我们找出每个簇的中心词元，并将其余词元分配到主题相关性最高的簇中。我们提出了两种聚类方案，即 ClustCrypt [1] 和 ClusPr [54]。我们用所提出的方案替换 S3BD 现有的聚类策略。

本章介绍针对不同数据特征的聚类方案，以及所提出的聚类方案可部署于其中的上下文架构概述。请注意，在所提出的聚类方法中，我们假设数据集中所有词元的频率和共现信息均可用。

3.2 问题陈述

S3BD 及其他研究中以往的聚类方案要求指定簇数才能开始分区，这不利于将词元（关键词）最优地聚类到最合适的簇中。我们无法预先确定

无论数据集大小如何，都使用相同的簇数和簇大小。此外，如果数据具有动态性，聚类方案就需要适应添加到数据集中的新词元。相反，从文档集中删除某些文档也可能导致簇缩小。因此，本章探讨如何在加密非结构化数据集中，以最优且可扩展的方式对关键词进行聚类。这项研究通过估算适当的（k），并将关键词分配到各个簇中，改进了加密关键词的聚类。我们强调，加密词元之间的概率语义相似度对于聚类至关重要，可用于衡量每个词元与其他词元相互分离的倾向。

3.3 所提出的聚类工作的定位

我们提出的工作受到 Woodworth 等人提出的方法的启发，该方法使用 K均值方法 [3]，基于中心索引对加密词元（又称关键词）进行基于主题的聚类。簇级词元分布函数是根据每个加密词元的统计数据确定的。作者使用了预先设定的 K值。这种 K 值 效率不高，因为适当的簇数可能会因数据集特征而异。此外，由于作者仅考虑静态/未更改的数据，因此所提出的方案无法处理动态数据。另一方面，所提出的解决方案提供了一种启发式方法，用于近似确定合适的簇数，随后在云端维护数据隐私的同时对数据进行聚类。对于随时间推移添加或删除文档的动态数据集，由于重新聚类操作，

簇会缩减或扩展，以反映数据集的动态变化。

请注意，ClustCrypt 只能有效地对加密静态数据进行聚类，因此当数据具有动态性时，它无法管理簇集。我们的下一种解决方案 ClusPr 可处理静态、半动态以及全动态加密非结构化数据。具体而言，我们提出了三种不同的聚类方案，即 S-ClusPr、SD-ClusPr 和 FD-ClusPr，以分别适配静态、半动态和全动态数据集。表3.1总结了文献中现有的重要相关研究，并将所提出聚类工作的贡献度与这些研究进行了定位比较。

表3.1。 现有隐私保护聚类方法的总结，以及我们提出的工作（ClustCrypt 和 ClusPr）相对于这些方法的定位。

研究工作	簇数估计 #簇	加密 方法	云的 可信性	使用边缘 计算	实时 支持	动态数据 聚类	多个 数据所有者
王等人 [41]	No	同态	半诚实	No	No	No	No
Valdiya 与 Clifton [43]	No	同态	半诚实	No	Yes	No	Yes
Pang 与 Wang [39]	No	同态	半诚实	No	Yes	No	Yes
Sun 等人 [36, 4]	No	用户端	诚实但好奇	No	Yes	No	No
Zhu等人 [34]	No	同态	诚实	No	No	No	Yes
Woodworth 等 [3]	No	用户端	诚实但好奇	No	Yes	No	Yes
ClustCrypt（所提方案）等 [1]	Yes	用户端	诚实但好奇	No	Yes	No	Yes
ClusPr（提出的）等人 [54]	Yes	用户端	诚实但好奇	Yes	Yes	Yes	Yes

3.4 促进安全搜索系统中聚类的架构

3.4.1架构：*ClustCrypt*

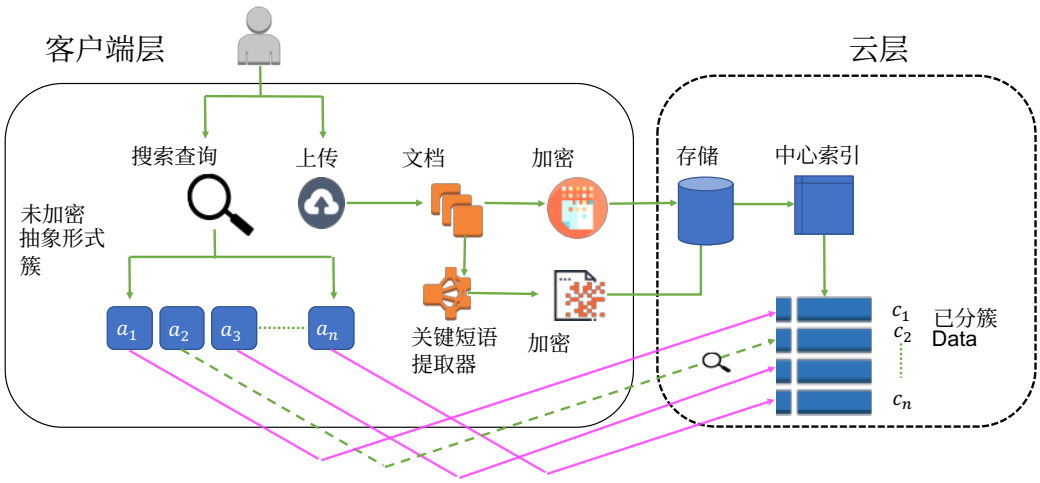
尽管我们是在 S3BD 的上下文中实现 ClustCrypt，但该方法具有通用性，也可部署于其他需要对加密数据进行聚类的系统中（例如， [78, 79]）。图 3.1 展示了 ClustCrypt 在 S3BD 系统中的位置。可以看到，S3BD 由客户端层和云层 [3]组成。客户端层被视为可信层，为用户提供上传和搜索功能。云层被视为诚实但好奇的层，

因此，所有文档及其索引词元均以加密形式存储。为实现实时搜索，必须对加密的索引词元进行聚类。图示系统由面向用户的“客户端层”（用户也可以是数据所有者）和存放索引与簇的“云层”组成。用户可以将文档上传至云端，也可以输入搜索查询，以查找与查询在语义上相关的文档。在此设置中，若用户希望上传文档，首先从原始文档中提取关键词或词元，然后将文档和词元加密后发送至云层。RSA确定性加密技术 [80] 用于加密文档和提取的令牌。希望执行搜索的各个数据用户（例如，执法人员）使用同一个 RSA密钥。

云层维护一个中心索引 结构，其中包含若干键值对。每个键值对分别表示一个加密词元，以及该词元出现的文档（位置）列表，并包含该词元在每份文档中的频率。同态加密 [37]可用于加密词元频率信息。然而，由于处理同态加密数据 [42] 会带来性能下降，为了在实际应用中维持实时搜索质量，目前频率信息以未加密形式存储。用户发出搜索查询后，搜索关键词会被加密，并在云层的中心索引中进行搜索，以检索相关文档。收到匹配文档列表后，用户可以下载这些文档，并使用其私钥解密。

根据索引结构中的词元构建簇 $c_1 ... c_n$，以避免针对每次搜索查询都穷举整个索引结构。簇以主题为基础构建，确保k 个簇的并集等同于索引结构。对于给定的搜索查询，不必搜索整个索引，而是通过剪枝缩小搜索空间，将其限制在与该搜索查询主题相关的簇中。剪枝依据一组从各个簇中抽样得到的抽象结构（记为 $a_1 ... a_n$）来实现，这些抽象结构存储在客户端层，或可能存储在可信边缘服务器上。在我们之前的研究中，我们提出了构建以用户为中心的抽象，以实现个性化搜索 [17]。 [3] 中介绍了抽样的具体方法。用户发出搜索查询后，系统会选出与该搜索查询最相似的摘要，然后搜索这些摘要对应的簇。

图 3.1. 集成 ClustCrypt 方法的高层双层（客户端-云）搜索系统架构。



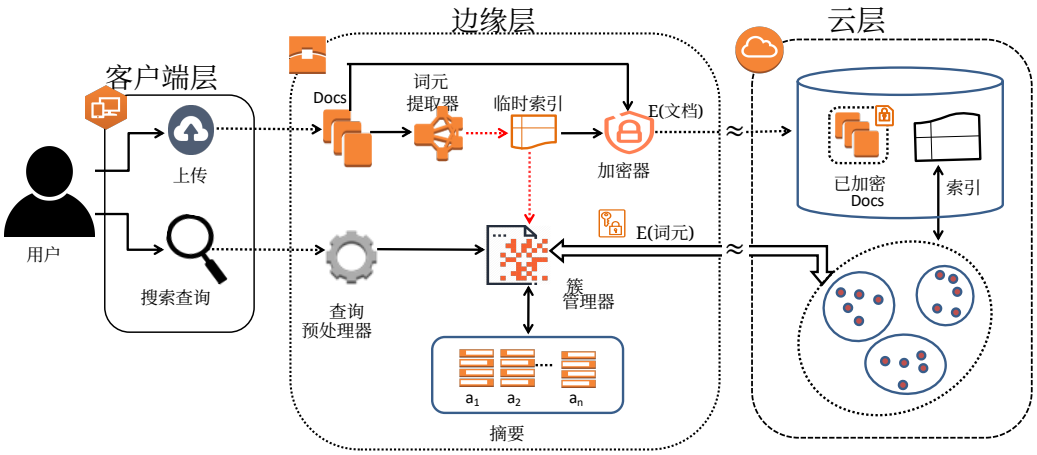
3.4.2架构：*ClusPr*

图 3.2 展示了 ClusPr 的开发背景架构概述。该架构展示了如何将 ClusPr 应用于 S3BD——一种需要对加密数据进行聚类的基于云的安全语义搜索系统 [3]。与原始 S3BD [3] 和 ClustCrypt [1] 所采用的 S3BD 架构不同，该架构以客户端设备、边缘系统和中心云系统为基础，构成一个三层系统。边缘层位于用户场所（因此被视为可信），可减轻客户端层处理计算密集型任务的负担。这对于非静态（即半动态、全动态）数据集尤为重要，因为随着文档不断上传至云层，这些文档必须随之进行处理。

在 S3BD 的具体场景中，用户上传文档后，文档会经过边缘层上的令牌提取器，以提取在语义上代表该文档的关键词（即词元）。对于动态数据集，我们使用临时索引结构来存储提取的词元，记录每个新词元在不同文档中的出现情况。接下来，使用用户密钥加密文档，并将其安全地存储在云存储库中。随后，根据相关文档提取的词元构建临时索引 结构，然后将其加密并上传至云端。临时索引结构展示词元、其频率以及它们在上传批次中的出现情况。临时索引中的词元由加密器 使用用户密钥进行加密。通过加密文档以及提取的词元，加密器保护云端数据的隐私。注意

尽管我们在技术上可以使用同态加密来保留统计属性（频率和共现），但出于效率考虑，在当前实现中，我们保留这些属性为未加密状态。我们假设此类属性不会泄露有关数据的有意义信息。事实上，曾在同态加密的大数据上使用 [41],*K*值均值聚类，并发现聚类的时间开销可能高得令人难以承受。下一步，将临时索引交给簇管理器，由其在云端做出合适的聚类决策。簇管理器可以决定保留现有簇，仅根据临时索引中的条目对其进行更新。或者，在观察到临时索引发生重大更新后，簇管理器会决定对所有词元进行全面重新聚类。尽管前述研究中有少数工作能够对加密数据进行聚类，但它们无法对动态数据集进行聚类；而 ClusPr 则能够在确保隐私的同时，对静态和动态数据进行聚类。我们将在第3.7节中说明 ClusPr 的更新和重新聚类过程。簇管理器还负责生成和维护摘要。每个摘要 a_i 都是云层 [17]上相应簇 C_i 的抽样概要。摘要用于缩小搜索操作的范围，并将搜索引导至与查询主题相关的簇。有关摘要的更多详细信息，请参见第3.6.3节。

图 3.2. ClusPr 部署于三层架构（客户端、边缘和云）中的上下文概述，该架构旨在实现安全云端搜索服务。假设边缘层位于用户 premises 中且可信。它用于减轻隐私和聚类相关进程带来的计算开销。



对于静态数据集，该架构经过简化：提取的词元经加密后直接馈入云层的索引结构。数据上传过程完成后，云层便启动聚类过程。由于静态簇没有定义重新聚类过程，簇管理器仅负责生成和维护摘要 [17]。值得注意的是，在图 3.2 的架构中，边缘层中的虚线箭头用于突出显示动态数据集的不同之处。关于所提出的静态（S-ClusPr）和动态（SD-、FD-ClusPr）数据聚类方案的更多详情，分别见第 3.6 节和第 3.7 节。

与 ClustCrypt 类似，ClusPr 使用 RSA 加密技术进行加密，并根据簇构建摘要集。作为搜索系统上下文中聚类策略的一种用例，用户发出搜索查询后，与该搜索查询相似度最高的摘要会被

iden识别出来。随后，仅搜索与这些摘要相关联的簇。

3.5 ClustCrypt：静态非结构化数据的隐私保护聚类方案

本部分首先（在第 3.5.1 节中）详细说明如何估算表示静态大型数据集时应形成的适当簇数。其次，在第 3.5.2 节中，我们给出一种算法来确定每个簇的中心。然后，在第 3.5.3 节中，我们说明如何将已索引术语分配到各个簇。最后，在第3.6.3节中，我们介绍实现剪枝的方式，即，将搜索查询引导至相关簇的方法。

3.5.1估算静态数据集的簇数

根据数据集的特征及其文档中词元的分布情况，适当的簇数（K）可能会有很大差异。然而，最优地确定 K 会直接影响基于主题的聚类准确率，进而影响使用这些簇的系统（例如，搜索应用程序）的效率。加密词元及其元数据（包括词元出现在哪些文档中以及其频率）是估算 K时唯一可用的参数。这些词元及其元数据由关键词提取器生成，该提取器从每份文档中提取n个单词或多词短语词元。我们假设所有文档一视同仁，并且给定静态数据集中的所有文档都使用相同的n 值。

静态数据集的 K 估算通过以下两个步骤完成：(1) 构建词元-文档频率矩阵；(2) 构建归一化矩阵。

步骤-1：构建词元-文档频率矩阵。为了说明该方案，我们以表 3.2 中包含五个 词元和六个 文档的示例进行说明。我们根据索引结构初始化词项-文档矩阵 **A** 。在该矩阵中，每一行代表一个词元，每一列代表一个文档。尽管我们的方法不处理明文词元，但为了便于阅读，表 3.2 中仍同时列出明文词元（“词”列）及其加密形式（“哈希”列）。矩阵 **A** 的每个条目 $a_{i,\ j}$ 表示词元 i^{th} 在文档 j^{th} 中的频率（记作 $f(i,\ j)$）。

表 3.2. 词元-文档频率矩阵 **A**，基于索引结构构建

词哈希		$\mathbf{d_1}$	$\mathbf{d_2}$	$\mathbf{d_3}$	$\mathbf{d_4}$	$\mathbf{d_5}$	$\mathbf{d_6}$
Book	**Uh5W**	30	0	23	4	40	0
求解	**/Vdn**	5	0	0	60	34	0
流量 **oR1r**		0	23	0	30	0	0
Net	**vJHZ**	52	49	0	23	0	26
输入	**tH7c**	0	45	68	0	3	5

对于大型数据集，矩阵大小可能会大到难以处理，而且十分稀疏。为避免这种情况，我们会对矩阵进行裁剪，只保留对构建簇有影响的词元。我们将文档共现次数 定义为包含某个特定词元的文档数量。然后，在构建词元-文档频率矩阵 **A**时，我们只考虑文档共现次数大于或等于整个数据集中文档共现次数均值的词元。

步骤二：构造归一化矩阵。为了使词元与文档之间的关系可量化且可比较，我们需要对

词元-文档频率矩阵进行归一化。考虑到 $a_{i,\ j}$ 表示词元 t_i 与文档 d_j之间关联强度，词元-文档频率矩阵的第 j 列中的最大值表示与文档 d_j关联度最高的词元。因此，为了进行归一化，我们将 **A** 中每个条目的值除以矩阵对应列中的最大值，并将结果存储在一个新矩阵中，称为矩阵 **N**。每个条目 $n_{i,\ j}$ 的值根据公式 3.1 正式计算。

$$n_{i,j} = \frac{a_{i,j}}{\max\limits_{\forall i} a_{i,j}} \tag{3.1}$$

表 3.2 中的示例所对应的归一化矩阵 **N** 如表 3.3 所示。

表 3.3. 归一化词元-文档矩阵 **N**

词哈希	**d**	**$_1$**	**d_2**	**d_3**	**d_4**	**d_5**	**d_6**
Book	**Uh5W**	0.58	0	0.34	0.07	1	0
求解	**/Vdn**	0.1	0	0	1	0.85	0
流量	**oR1r**	0	0.47	0	0.5	0	0
Net	**vJHZ**	1	1	0	0.38	0	1
输入	**tH7c**	0	0.92	1	0	0.08	0.2

步骤 3：构建概率矩阵 R 和 S 此步骤的目标是计算加密令牌之间的主题相似度。为此，我们需要计算一个词元的主题与其他词元具有相似性的概率。我们假设，在不同文档中共同出现的词元很可能属于同一主题。此外，两个词元之间的相似度大小可能受到词元在数据集中的分布情况影响。例如，特定术语只会出现在少数文档中，在整个数据集中分布并不广泛

。此类分布稀疏的词元与其他词元的共现次数较少，这会增加数据集中主题的多样性，并可能提高所需的簇数（K）。我们利用归一化矩阵（**N**）进行两阶段概率计算，得到一个表示词元间主题相似度的矩阵（记为 **Q**）。

表 3.4. 矩阵 **R** 基于归一化矩阵**N** 构建，用于表示每个词元在所有文档中的重要性

Word	**Hash**	$\mathbf{d_1}$	$\mathbf{d_2}$	$\mathbf{d_3}$	$\mathbf{d_4}$	$\mathbf{d_5}$	$\mathbf{d_6}$
Book	**Uh5W**	0.29	0	0.17	0.04	0.50	0
求解	**/Vdn**	.05	0	0	0.51	0.43	0
流量	**oRir**	0	0.48	0	0.52	0	0
Net	**vJHZ**	0.29	0.29	0	0.11	0	0.29
输入	**tH7c**	0	0.42	0.45	0	0.03	0.09

在第一阶段，我们计算每个词元对每篇文档的重要性。词元 t_i 在文档 d_j 中的重要性记为 $\tau_{i,j}$，其定义基于公式 3.2。

$$\tau_{i,j} = \frac{n_{i,j}}{\sum_{\forall k} n_{i,k}} \tag{3.2}$$

根据公式 3.2 和矩阵 **N**，我们生成矩阵 **R** ，其元素表示每个词元在所有文档中的重要性。实际上，**R** 中的每个元素 $r_{i,j}$表示选择包含词元 t_i 的文档 d_j 的概率。也就是说，$r_{i,j} = \mathrm{P}(t_i, d_j)$。在我们的示例中，表 3.4 展示了根据矩阵 **N** （见表 3.3）得到的矩阵 **R** 。

在第二阶段，我们计算每个文档对于每个词元的重要性。文档 d_j 对于术语 t_i的重要性记为 $\delta_{j,\ i}$ 其定义为

基于公式 3.3。

$$\delta_{j,i} = \frac{n_{j,i}}{\sum_{\forall q} n_{q,i}} \tag{3.3}$$

表 3.5。 矩阵 **S** 由**N** 构建，用于表示每个文档相对于每个词元的重要性

Docs	Book **Uh5W**	求解 **/Vdn**	流量 **oRir**	Net **vJHZ**	输入 **tH7c**
$\mathbf{d_1}$	0.34	0.06	0	0.60	0
$\mathbf{d_2}$	0	0	0.19	0.49	0.38
$\mathbf{d_3}$	0.17	0	0	0	0.45
$\mathbf{d_4}$	.04	0.51	0.25	0.19	0
$\mathbf{d_5}$	0.52	0.44	0	0	0.04
$\mathbf{d_6}$	0	0	0	0.84	0.16

考虑每个 $\delta_{j,i}$ 和 **N**，我们生成 **S** ，其各个元素表示每个文档相对于每个词元的重要性。事实上，每个元素$s_{i,j}$表示从 d_j 中选择 t_i 的概率（即，我们有 $s_{i,j} = \mathrm{P}(d_j, t_i)$ ）。在我们的示例中，表 3.5 展示了由**N**得到的 **S** 。

Step 4- 构建矩阵 Q 以确定簇数

回顾一下，**R** 是词元到文档的矩阵，而**S** 是文档到词元的矩阵。为了确定加密词元之间的相似度，我们将**R** 与**S**相乘。由于**R** 的列数与**S** 的行数相等，因此可以将矩阵**R** 与**S**相乘。所得矩阵记为**Q**，它是一个词元到词元的矩阵，并作为确定所需簇数量的依据。每个元素$q_{i,j}$ 表示词元 i 与 j之间的主题相似度。更具体地说，$q_{i,j}$表示对于 $i \neq j$，词元 i 与词元 j 共享相似主题的程度

其计算方式为$q_{i,j} = \sum r_{i,j} \cdot s_{j,i}$。表 3.6 展示了本节讨论的示例所对应的矩阵$\mathbf{Q}_{\forall i,j}$。

表 3.6。 簇决策矩阵 **Q** 由**R** 和**S** 矩阵相乘构建

词-哈希	Book Uh5W	求解 /Vdn	流量 oRir	Net vJHZ	输入 tH7c
书籍- Uh5W	0.39	0.25	0.01	0.18	0.09
求解- /Vdn	0.26	0.45	0.12	0.12	0.02
流量- oRir	0.02	0.26	0.21	0.33	0.18
网络- vJHZ	0.10	0.07	0.08	0.58	0.15
输入- tH7c	0.09	0.01	0.08	0.28	0.37

Q 的对角线元素表示每个词元与其自身的主题相似度，以及与其他主题的不相似度（即 分离）。更具体地说，$q_{i,i}$的值表示术语 t_i 与其他术语不共享主题的程度。因此，我们将对角线元素（$q_{i,i}$）定义为分离因子，因为对于每个词元，它表示该词元与其他主题保持分离的倾向。因此，对分离因子求和可以近似得出划分数据集主题所需的簇数（K）。令 m 表示**Q**中的词元总数。然后，使用公式 3.4近似给定数据集的 K 。我们使用向上取整函数将 K 取为整数值。

$$k = \lceil \sum_{i=1}^{m} q_{i,i} \rceil \tag{3.4}$$

通过一个假设来验证 K 的正确性：对于一个集合，如果其中各个元素彼此不同，则该集合的 K 应较高；否则， K 应较低 [81, 82]。公式 3.4 是近似计算 K的核心。根据这一

公式，当文档之间高度不同时，且每个文档中的各个词元分别代表唯一的主题时，最大 K 值可达到 M；否则将低于 M。因此，我们的方法符合聚类假设。

3.5.2确定簇中心

在k均值聚类中，通常会任意选择簇中心 [83, 84]。随后，根据距离度量函数（例如，欧几里得距离 [83] 或语义图 [84]），将数据集元素分配到各个簇中。K均值通过迭代移动簇中心，直至收敛。然而，我们发现，词元数量极其庞大，使得迭代中心迁移步骤（因而也使k均值聚类）在大数据场景下耗时过长，难以接受 [85]。因此，在本节中，我们将提出一种适用于大数据的加密词元聚类方法。

我们的聚类方法的关键在于取消迭代中心迁移步骤。这一改变意味着，初始簇中心不能任意选择，而必须主动选择，使其能够覆盖数据集中的各种主题。为此，一种朴素的方法是选择拥有最多关联文档的前k 个词元。尽管这种方法选择了重要的（关联度高的）词元，但最终选出的中心在文档与主题方面具有较高的重叠度。为了选择合适的中心词元，我们提出选择不仅具有高文档关联度，而且能够覆盖数据集中多样主题的词元。

我们将词元 i 的中心性 定义为一种衡量指标，用于表示某个主题及其与同一主题中其他词元的相关性。假设词元按文档关联度降序排列。令 U 表示与当前选定中心相关联的文档并集。此外，对于词元 i,

令 A_i 表示与 i 相关联的文档集合。那么，词元 i 的唯一性 [3]（记为 ω_i）定义为：与 i 相关联但不在 U 中的文档数量（即，$|A_i - U|$）与和 i 相关联且位于 U 中的文档数量（即，$|A_i \cap U|$）之比。唯一性表示某个词元代表尚未被其他已选为中心的词元识别出的主题的潜力。具体而言，唯一性值大于 1 的词元与当前选定中心未覆盖的文档具有较强关联，因此可以选为新的中心。

请记住，矩阵 C 的每个元素 $c_{i,\ j}$ 都表示词元 i 与词元 j 之间的主题相似度。此外，对角线项 $c_{i,\ i}$ 衡量词元 i 与其他词元的分离程度。因此，词元 i 与其他词元的总相似度可通过 $\Sigma_{\forall j|j\neq i}c_{i,\ j}$ 得出。注意，对于词元 i，有 $\Sigma_{\forall j}c_{i,\ j} = 1$，因此，词元 i 的总相似度等于 $1 - c_{i,\ i}$。词元的中心性由其唯一性、与其他词元的相似度大小以及其孤立程度来衡量。也就是说，词元 i 的中心性定义为 $\Phi_i = \omega_i \times c_{i,\ i} \times (1 - c_{i,\ i})$ 。

算法1展示了从数据集的索引词元集合中选择最多 k个中心的高层伪代码。除 k 外，算法还接收中心索引和 C 矩阵作为输入。算法返回一个最多包含 k 个中心词元的集合，记为 $centers$，作为输出。开始时，输出集合初始化为空。 U 表示由所选中心覆盖的文档集合。一个堆结构，记为 Θ，用于存储每个词元及其

算法1： 用于确定簇中心的伪代码

输入： k、 C 矩阵和 $central\ index$（词元按降序排列依据文档关联度）

输出： 集合 $centers$ ，最多包含 k 个中心词元

```
1  函数 Choose Center(k, C, Index):
2      centers ← ∅
3      U ← ∅
4      Θ ← {(∅, ∅)} //词元及中心性值对
5      对每个词元 i ∈ index 执行
6          ω_i ← 计算唯一性(i, U)
7          如果 ω_i > 1 则
8              U ← U ∪ U_i
9              Φ_i ← (ω_i × c_{i,i} ×(1 − c_{i,i}))
10             根据 Φ_i 将配对 (i, Φ_i) 添加到最大堆 Θ
11         end
12     end
13     centers ← 从 Θ 堆中提取 k 个最大配对
14     返回中心
15 结束
```

中心性值。对于每个词元 i，都会计算其唯一性和中心性值（步骤5至12），并将相应的键值对插入堆中。请注意，唯一性低于1的词元不具备作为簇中心的潜力。接下来，我们选择最多 k 个中心性值最高的中心词元。

3.5.3令牌聚类

一旦选定 k 个词元作为簇中心，就会将这些词元分配到各个簇中。分配依据是中心词元与其余词元之间的相关性（又称距离）。已有成熟技术可用于计算这种相关性，但其中大多数（例如，语义图 [84] 和欧几里得距离 [83]）并不适用于在整个集合中稀疏分布的词元

data中 [83]。此外，这些技术并非为处理加密数据 [84]而设计。

在 S3BD [3],中，提出了一种基于文档共现的方法，用于衡量相关性并对加密词元进行聚类。在该方法中，如果两个词元出现在同一组文档中，就认为它们相关 [3]。我们利用这一点来衡量词元与簇中心的相关性，并将词元分配到相关性最高的簇中。为了确定特定词元与中心之间的相关性，我们需要计算该词元的贡献度和共现 指标。设 t 为数据集 D 中文档 d 里的一个词元，其频率表示为$f(t,d)$。那么， d对 t的贡献度（表示为 $\kappa(d,t)$）根据公式 3.5 定义。

$$\kappa(d,t) = \frac{f(t,d)}{\sum_{j \in D} f(t,j)} \tag{3.5}$$

词元 t 与中心词元 γ_x 在文档 d 中的共现（表示为$\rho(t,d,\gamma_x)$）定义为： t和中心词元 γ_x在 d中的频率之和，与整个数据集中 t 和 γ_x 的总频率之比。公式 3.6 给出了共现的形式化表示。

$$\rho(t,d,\gamma_x) = \frac{f(t,d) + f(\gamma_x,d)}{\sum_{j \in D} (f(t,j) + f(\gamma_x,j))} \tag{3.6}$$

根据贡献度和共现指标，词元 t 与 γ_x 之间的相关性（记作 $r(\gamma_x,\ t)$）定义为这两个指标的乘积

$$(i.e.,\ r(\gamma_x,t) = \sum_{j \in D} \kappa(j,t) \cdot \log\left(\rho(t,\gamma_x,j)\right)).$$

3.6 S-ClusPr：面向静态非结构化数据集的隐私保护聚类方案

本节将详细介绍 S-ClusPr方案，用于对隐私保护静态大数据集进行聚簇。请注意，S-ClusPr 使用与第 3.5.1 节中 ClustCrypt 相似的方法来估计合适的簇数（k）。但是，我们为 ClusPr 中的中心选择和令牌分布方法提出了更稳健的启发式方法，以获得主题划分更明确的簇。在第 3.6.1 节和第 3.6.2 节中，我们分别解释中心选择和令牌分布方法。

3.6.1 中心选择

在K均值聚类中，通常会任意选择簇中心 [83, 84]。随后，根据距离度量函数（例如，欧几里得距离 [83] 或语义图 citeLiuCroft），将数据集元素分配到各个簇中。K均值通过迭代移动簇中心直至收敛。然而，我们发现，词元数量极其庞大，使得迭代中心迁移步骤（因而K均值聚类）在大数据场景下耗时过长，难以实际应用 [85]。因此，在本部分中，我们将提出一种适用于大数据的加密词元聚类方法。

我们的聚类方法的关键在于省去迭代中心迁移步骤。这一改变意味着初始簇中心不应随意选择，而应主动选取，以覆盖数据集中的各种主题。为此，一种朴素方法是选择拥有最多关联文档的前K值 个词元。

尽管这种方法选出了重要（关联度高）的词元，但最终选出的中心之间主题重叠度很高。我们建议选择的词元不仅要有较高的文档关联度，还要覆盖数据集中存在的多种主题。

我们将词元 i 的**中心性** 定义为衡量其代表某个主题以及与同一主题中其他词元相关性的指标。假设词元按文档关联度降序排列。令 U表示与当前选定中心相关联的文档并集。对于词元 i，令 A_i 表示与 i相关联的文档集合。那么，词元 i 的**唯一性** [3]（记为 ω_i）定义为与 i 相关联但不在 U 中的文档数量（即 $|A_i - U|$）与既与 i 相关联又在 U 中的文档数量（即 $|A_i \cap U|$）之比。唯一性表示某个词元代表尚未被已选为中心的其他词元识别的主题的潜力。特别是，唯一性值大于 **1** 的词元与当前所选中心尚未覆盖的文档具有较高关联度，因此可选为新的中心。

请注意，矩阵 $\mathbf{Q}$ 的每个项 $q_{i,j}$ 都表示词元 i 与 j之间的主题相似度。此外，对角线项 $q_{i,i}$ 衡量词元 i与其他词元的分离程度。因此，词元 i 与其他词元的总相似度可通过 $\Sigma_{\forall j|j\neq i}q_{i,j}$得出。请注意，对于词元 i，我们有 $\Sigma_{\forall j}q_{i,j} = 1$，因此词元 i 的总相似度等于 $1 - q_{i,i}$。词元的中心性由其唯一性、与其他词元的相似度大小，

以及其孤立程度来衡量。也就是说，对于词元 i，中心性定义为：$\Phi_i = \omega_i \times q_{i,i} \times (1 - q_{i,i})$。

算法 2： 用于确定簇中心的伪代码

输入： K、$\mathbf{C}$ 矩阵，以及 $Index$（词元按文档关联度降序排列）

输出： $centers$ 集合，最多包含 K 个中心词元

```
1  函数 Choose Center(k, Q, Index):
2      centers ← ∅
3      U ← ∅
4      Θ ← { (∅, ∅) } //词元及其中心性值
5      遍历 词元 i ∈ Index 执行
6          ω_i ← 计算唯一性(i, U)
7          如果 ω_i > 1 则
8              A_i ← 计算文档关联(i, Index)
9              U ← U ∪ A_i
10             Φ_i ← (ω_i × q_{i,i} × (1 − q_{i,i}))
11             将配对 (i, Φ_i) 添加到基于 Φ_i 的最大堆 Θ
12         end
13     end
14     centers ← 从 Θ 堆中提取 K 个最大配对
15     返回中心
16 结束
```

算法 2 展示了从数据集的索引词元集合中选择最多 K个中心的高级伪代码。除了 K，该算法还接收中心索引和$\mathbf{Q}$ 作为输入。算法输出一个至多包含 K 个中心词元的集合，记为 $centers$。开始时，输出集合初始化为空。 U 表示由所选中心覆盖的文档集合。一个记为 Θ的堆结构用于存储每个词元及其中心性值组成的键值对。对于每个词元 i，都会计算其唯一性和中心性值（步骤 $5-13$），并将相应的键值对插入堆中。请注意，唯一性低于 1 的词元不具备作为簇中心的潜力

在下一步中，我们选择至多 K 个具有最高中心性值的中心词元。

3.6.2在簇间分配加密词元

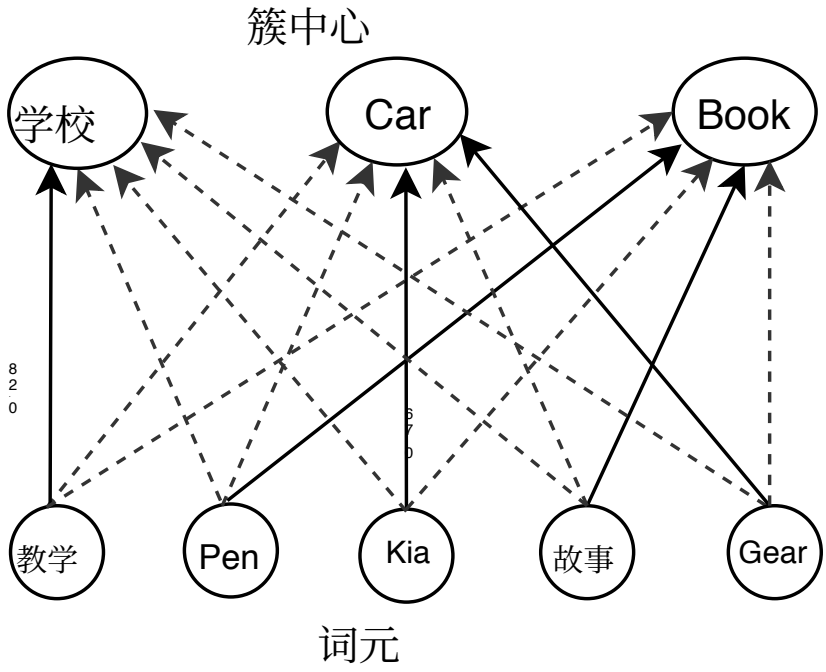
一旦将 K 词元指定为簇中心，索引中的其余词元便会根据它们与中心词元的相关性（也称距离）分配到各个簇中。

由于非中心词元与 $centers$ 集合中的成员之间没有交集，我们可以将词元在各簇间的分布建模为加权二分图，其中每条边的权重表示词元与中心之间的相关性。图3.3展示了一个二分图示例，用以说明各词元与中心之间的关系。实线表示每个词元权重最高的边，代表该词元应被分配到的簇。已有技术（例如，语义图 [84], 欧几里得距离 [83]）可用于计算相关性，但这些方法并不适用于稀疏分布的加密词元 [83] [84]。

由于加密词元会失去其语义，我们应基于词元的统计特征来定义它们之间的相关性，然后利用这种相关性，将每个词元分配到相关性最大的簇中。

直观而言，词元 t_i 和 t_j之间的相关性度量，记作$r(t_i, t_j)$，是根据它们的共现规模定义的，即，两个词元共同出现的文档数量 [3, 1]。令 F_i 和 F_j分别表示 t_i 和 t_j 出现过的文档集合。那么，

图3.3。 一个表示中心与其余词元之间相关性的二分图。每条边的权重表示一个词元与一个中心之间的相关性。实线表示对某个词元具有最大相关性的中心。



这两个词元的直观共现值为 $F_{co} = F_i \cap F_j$。然而，深入分析表明，仅根据共现基数（即， $|F_{co}|$）来量化相关性，可能会产生误导，原因有以下两点:

首先，直观的共现忽略了 F_i和 F_j 之间差异度的大小，而这种差异度会对 t_i 和 t_j之间的相关性产生负面影响。差异度根据对称差（即，我们有 $F_{dis} = F_i \oplus F_j$）确定。因此，为了同时考虑共现和差异度的影响，我们定义了一种称为相对共现的新度量，并利用它来确定 t_i 和 t_j之间的相关性。

第二，直观的共现忽略了 t_i 和 t_j 在每个 h

文档 $d \in F_{co}$ 中的重要性。因此，为了衡量每个文档 d 中的共现值，记作 $\upsilon(t_i, t_j, d)$，我们会考虑每个词元相对于它们在 F_{co} 的所有文档中的重要性。我们使用词元在文档中的频率来衡量其在该文档中的重要性。具体而言，在文档 d 中，我们根据公式 3.7 计算共现值。

$$\upsilon(t_i, t_j, d) = \frac{f(t_i, d)}{\sum\limits_{\forall m \in F_{co}} f(t_i, m)} \cdot \frac{f(t_j, d)}{\sum\limits_{\forall m \in F_{co}} f(t_j, m)} \tag{3.7}$$

同样，我们利用公式 3.8 衡量每篇文档 $d \in F_{dis}$ 中两个词元之间差异度的影响，记作 $\varphi(t_i, t_j, d)$。

$$\varphi(t_i, t_j, d) = \frac{f(t_i, d)}{\sum\limits_{\forall m \in F_{dis}} f(t_i, m)} + \frac{f(t_j, d)}{\sum\limits_{\forall m \in F_{dis}} f(t_j, m)} \tag{3.8}$$

在文档 d 中，一旦我们知道 t_i 和t_j 之间的共现和差异度，就可以将相对共现计算为 $\rho(t_i, t_j, d) = \upsilon(t_i, t_j, d) - \varphi(t_i, t_j, d)$。随后，利用这两个词元在所有文档中的相对共现（即$F_i \cup F_j$）来计算它们之间的相关性。

假设 c 是表示给定簇中心的词元（也就是说，$t_i = c \in centers$），我们根据公式 3.9 定义 c 与词元 t之间的相关性。词元 t 将被分配到中心具有最大相关性的簇中。请注意，在此公式中，为了强调词元 t 在文档 d中的重要性，我们还考虑了其频率比。

$$r(c,t) = \sum_{d \in (F_t \cup F_c)} \rho(t,c,d) \cdot \frac{f(t,d)}{\sum\limits_{\forall m \in F_t} f(t,m)} \tag{3.9}$$

3.6.3通过簇剪枝加快搜索操作

构建基于主题的簇，旨在通过根据查询主题限制（剪枝）搜索范围，而非穷举遍历整个索引结构，实现对大数据的可扩展搜索。要进行剪枝，我们需要识别与搜索查询在语义上相关的簇，并舍弃不相关的簇。然而，在云端操作加密数据时，剪枝是一项具有挑战性的任务。

为克服这一挑战，我们需要以纯文本形式获取每个簇的主题，以便识别主题在语义上与搜索查询相关的簇，并且仅在这些簇中进行搜索。为此，在先前的工作中 [17], ，我们建立了一种表示每个簇主题的方法C_x （记为 α_x），即考察 C_x中出现频率最高的前 n 个词元。α_x 中的词元经过解密后，会在 ClusPr 的边缘层中以一种名为摘要的结构进行维护。摘要用于衡量查询与其对应簇之间的主题相似度。接下来，将在与查询最相关的簇中进行搜索。有关创建摘要和剪枝操作的更多详情，感兴趣的读者可参阅我们先前的研究 [17, 3]。

3.7 隐私保　　护动态非结构化数据集的聚类方案

3.7.1 概述

在上一节中，我们介绍了静态（例如，档案）加密大数据集的聚类。然而，许多大数据集是动态的（例如，医疗保健数据、犯罪记录） [86] ，其内容会随时间变化。本节将讨论如何对这类数据集进行聚类，并在其上进行搜索。我们考虑两种类型的动态数据集：第一种是半动态 数据集，其内容会随时间批量更新（例如，现代艺术博物馆（MoMA）数据集 [87]）；第二种是完全动态 数据集，其内容持续更新（例如，Twitter信息流 [88]）。

数据集的最新变化必须反映在簇中。否则，即使文档包含相关内容，搜索系统也无法检索到已更改的文档。事实上，数据集的更新会影响词元共现，进而影响聚类布局。因此，挑战在于了解添加或删除文档会如何改变主题和簇数。

鉴于大规模数据集的大小，每当单个文档或一小批文档到达时就重建簇（称为重新聚类）会耗费过多时间。此外，小规模更新通常只会使词元的共现发生微不足道的变化，不太可能改变簇的排列。只有重大更新才会对以下方面造成决定性影响：

共现和相关性，从而需要重新聚类。因此，需要进一步探讨两个问题：何时进行重新聚类？以及如何对词元重新聚类？为了解决这些问题，我们根据动态数据集的类型，在 ClusPr 中提出了两种聚类方案：半动态数据聚类方案（SD-ClusPr）和全动态数据聚类方案（FD-ClusPr）。

3.7.2半动态数据聚类方案 *(SD-ClusPr)*

对于半动态数据集，可以按照上一节所述的方法，先对数据集中的第一批文档进行基于主题的聚类。随后，根据新一批文档所引起的变化，决定是否重新聚类。也就是说，我们需要确定新一批文档中提取的词元所引起的变化是否显著。

为了确定新一批文档中的词元所引起的变化是否显著，我们使用 χ^2 （卡方）分布检验 [89] ，该检验可识别给定总体中变量出现的显著变化。该 χ^2 检验称为拟合优度检验，并由公式 3.10 表示，其中 O_i 是特定变量在 K 次试验中的观测值， E_i 是其期望值。

$$\chi^2 = \sum_{i=1}^{k} [(O_i - E_i)^2 / E_i] \tag{3.10}$$

我们会考察新批次中提取的令牌数量以及现有簇中的词元数量。我们的零假设（H_0）是进行重新聚类，并采用 χ^2 检验来验证 H_0的有效性。如果差异

较小，即新词元数量与现有词元数量之间的差异较小时，得到的 χ^2 值较低。对于一个自由度和95%置信区间，$\chi^2 = 3.841$ 的值无法拒绝 H_0。另外，如果新批次中的词元数量明显少于现有词元数量， χ^2 值会变高，表明与 H_0存在显著偏差。此时，决定拒绝 H_0 并保留现有簇。

做出重新聚类决策后，我们使用第 3.6 节中介绍的方法对更新后的数据集中的词元进行聚类。如果未执行重新聚类，则将新词元与下一批次中的词元累积起来。因此，新词元的总数会变得相当可观，从而导致较低的 χ^2 值，并随后接受 H_0。

3.7.2.1. 更新簇。设 U_1 为一个引入一组新词元 $T = \{t_1, t_2, ..., t_n\}$ 的新文档批次，这些词元不存在于现有簇中。假设根据前一部分中提到的重新聚类决策方法，我们决定保留现有簇 $\{C_1, C_2, ..., C_n\}$ 以容纳 T。

要将 $t_i \in T$ 分配到某个簇，我们可以按照第 3.5.3 节所述测量相关性。或者，我们可以利用摘要集合 $\{A_1, A_2, ..., A_n\}$。由于它们采用纯文本格式，因此可以使用语义相似度进行更准确的相关性测量，而不是根据文档中的词元共现推断相关性。在这种情况下，我们使用 Word2Vec [63]模型计算 t_i 与摘要 A_j的相关性。然后，将 t_i 分配到

相关性最高的簇。为避免分配不当，我们将 θ 定义为将 t_i 分配到 C_j所需达到的相关性阈值。如果 t_i 无法加入任何簇，则会创建一个名为 $C_{new} \in C$的新簇，并将 t_i 视为其中心。对所有 $t_i \in T$重复上述过程。

算法 3： 用于更新 SD-ClusPr 中簇的伪代码。

输入： 摘要集合 A, $tempIndex$, θ

输出： H，新词元到簇的映射

函数 SD-ClusPr($A, tempIndex, \theta$) :
  $T \leftarrow tempIndex \setminus CentralIndex$
  $H \leftarrow \emptyset$
  $A \leftarrow \cup_{i=1}^{n} A_i$
  $\Phi \leftarrow \emptyset$
  //用于查找相似度最高的摘要的最大堆
  对于每个词元 $t \in T$ **执行**
    对每个 $a_{ij} \in A$ **执行**
      $s \leftarrow$ 相似度（a_{ij}, t）
      如果 $s > \theta$ **则**
        将 (s, i) 添加到 Φ
      end
    end
    如果 $\Phi \neq \emptyset$ **则**
      //分配 t 到现有簇
      从 Φ 中提取最大配对 (t, i) $\leftarrow$
      将 (t, i) 添加到 H
    end
    else
      //形成一个新的摘要和簇，并将其添加到 H
      $A_{n+1} \leftarrow \{t\}$
      $A \leftarrow \cup_{i=1}^{n+1} A_i$
      将 (t, $n+1$) 添加到 H
    end
  end
  加密 H 并将其推送到云层
结束

3.7.2.2 确定 θ 阈值的值。 我们利用摘要 $\{A_1, A_2, ...A_n\}$ 来估算 θ 阈值的值。回想一下，元素

摘要 A_i 的元素是最能代表其对应簇 C_i 的主题的那些元素。我们将 A_i 的连贯性定义为其元素两两之间相似度距离的平均值。设 A_i 的元素集合为 $\{a_{i1}, ..., a_{ip}\}$。那么， A_i的连贯性（记为 K_i）根据公式 3.11 定义，其中 $sim(x, y)$ 表示 $(x, y) \in A_i \times A_i$ 之间的相似度距离。

$$K_i = \frac{\sum_{\forall (x,y) \in A_i \times A_i | x \neq y} Sim(x, y)}{\binom{p}{2}} \tag{3.11}$$

然后，我们将 θ 定义为所有摘要中的全局最小值（即$\theta = \min_{\forall i} K_i$）。这意味着，只有当新词元的距离不会降低当前簇的连贯性时，它才能加入某个簇。否则，新词元将形成自己的簇。

算法 3 展示了 SD-ClusPr 中更新簇的伪代码，适用于我们选择不执行重新聚类的情况。除了摘要集合（A）和 θ 之外，该算法还接收新文档批次的词元集合，并将其存储为临时索引。该算法返回 H 结构，其中包含新词元与其各自簇之间的映射。在步骤 $7 - 9$ 中，对于每个新词元，我们计算其与所有摘要元素 a_{ij} 之间的相似度距离，并检查该相似度距离是否超过 θ。如果超过 θ，我们就将相似度距离与对应的摘要编号组成一对，记为（a_{ij}， t），并根据距离构建最大堆 Φ（步骤 $10 - 12$）。如果 Φ 包含任何值，我们便从中提取值最大的那一对（即为 t 提供最高主题相似度的摘要）。然后，在步骤 17 中，这一对

（t, i）被添加到 H。相反，如果 Φ 为空，则表示没有簇能为 t提供足够高的相似度，因此在步骤 $19-24$ 中，我们使用 t构建一个新的摘要和簇。最后，我们对 H 的词元进行加密，并将其推送到云层。在云端，簇管理器根据 H更新其簇。

3.7.3全动态数据聚类方案 *(FD-ClusPr)*

不同于 SD-ClusPr，对于全动态数据集，文档到达时必须形成或更新簇。也就是说，新文档的持续到达或突发到达应触发 FD-ClusPr。因此，在 FD-ClusPr 中，我们考虑形成簇时的两种情况：(A) 初始情况，即第一份文档到达且不存在现有簇时发生的情况；以及 (B) 更新情况，即必须根据数据集中的新变化更新现有簇的情况。

在初始情况下，边缘层会从上传的文档中提取新词元集合。我们指定频率最高的词元来代表主题，并将其选为簇中心。然后，根据第3.7.2节讨论的方法，依据第二高频词元与指定簇中心之间的相似度距离对其进行聚类。此外，为了确定是加入现有簇还是形成新簇，我们将阈值初始化为 $\theta = 0.1$。此过程会持续进行，直至所有词元都完成聚类。在更新情况下，我们采用与 SD-ClusPr 相同的方法。也就是说，上传文档后，系统会决定执行重新聚类还是更新现有簇。

3.8 所提出聚类方案的安全性分析

本节仅讨论 ClusPr 的安全性分析。在这方面，对三层架构进行安全性分析，也涵盖了对双层 ClustCrypt 的分析。

所提出的聚类方案适用于可搜索加密和文档检索系统的上下文。根据图 3.2 所示的三层架构，客户端层和边缘层位于用户 premises 中，因此，这些层上执行的活动以及用户密钥被视为安全且可信。摘要 结构以纯文本形式保存在边缘层，以便我们衡量其与搜索短语的相似度并进行剪枝。

另一方面，在云层上执行的活动被视为不诚实且容易遭受不同类型的攻击。我们关注内部（即 关联方）和外部（即 无关联的外部入侵者）攻击者，他们意图获取加密聚类令牌和文档。为说明攻击者带来的威胁，我们给出以下预备定义：视图：该术语表示在客户端、边缘和服务器之间的任何给定交互期间，云端可见的部分。中心索引和簇集合$C_1...C_n$、给定加密搜索查询的陷门 Q'，以及加密文档集合 D'。在某些模型中， Q' 还包含每个术语对应的特定权重。与 Q' 相关的搜索结果被视为 I_c。扩展后的 Q' 和 I_c 的视图分别记作 $V(Q')$ 和 $V(I_c)$。

轨迹：该术语指关于 I_c 暴露的信息。我们的目标是

尽可能减少攻击者推断出 I_c 的信息。

视图和轨迹包含了攻击者可能获取的所有信息。为了加密文档集，我们采用概率加密模型，该模型被认为是最安全的加密技术之一 [3, 90]。这种方法不使用一一映射，因此， D' 不易受到字典攻击 [91]。簇中的每个词元都经过确定性加密。因此，视图中的每个簇仅显示词元及其在明文格式下的共现关系所对应的加密映射。

如果任何类型的攻击者能够访问云端，他们只能通过观察共现关系来了解特定加密词元的重要性。技术上，可以使用同态加密 [37] 对共现关系进行加密，并在其处于加密形式时对其进行计算。然而，在第2.2.1节中，我们讨论了这种技术在实际应用中的性能不足 [92] ，并会影响搜索系统的实时行为。因此，在当前实现中，我们使用明文格式的共现信息。请注意，即使共现关系未加密，攻击者也无法解密词元。

攻击者可能会获取有关 $V(Q')$ 的轨迹。从该视图中，攻击者只能通过分析查询词的相关权重，了解 Q' 中每个搜索词的重要性。与此前的考量类似，攻击者无法从 Q'中揭示搜索词。尽管可信计算基极小，攻击者仍可能试图访问

系统，手段是通过中间人攻击，利用诚实但已遭入侵的 或不可信的云服务提供商，攻击用户数据的机密性。无论采用何种手段，只要攻击者成功实施中间人攻击，就能访问使用 Q' 和轨迹进行搜索后得到的文档列表 $V(I_c)$。此时，攻击者可能只能获取文档名称，而其加密内容无法读取。

当搜索和簇更新可预测时，可以采用一些方法（例如， [93]）来应对频率攻击。理论上，攻击者可以通过实施频率攻击，建立一个涵盖所有簇中词元的字典。最终，攻击者会尝试利用该字典构建一个克隆文档集 D' 。尽管从特定文档中提取的所有词元足以推断该文档的主题，但由于我们并未使用文档集中的全部关键词来构建加密索引，因此无法揭示整篇文档。此外，在将文档外包至云之前，我们会一次性加密整篇文档，而不是进行词级加密。这一过程确保了即使云层上的文档集遭到泄露，也不可能实施字典攻击。

即使攻击者知道轨迹，也无法理解检索到的加密文档究竟传达了什么。此外，边缘层和云层之间的通信也可能遭受攻击。在这种情况下，攻击者通过监控搜索过程，可以获取针对 Q' 的文档列表。然而，攻击者无法解密这些文档，因为只有在将其下载到边缘系统后，才能对其进行

仅在边缘系统上下载时才会解密。

攻击者也可能试图修改簇中的数据（例如，加密词元和文档）。此类攻击可能会篡改用户数据的完整性。不过，这类攻击可以被检测到，因为边缘既无法解密被修改的词元以形成或更新摘要，用户也无法以原始明文形式解密检索到的文档。这是因为用户使用由自己管理的密钥对数据进行对称加密（例如，AES 加密）。因此，如果加密数据遭到攻击者篡改，用户便无法使用自己的密钥解密这些数据。事实上，保护用户密钥对于遏制可能的攻击至关重要。如果密钥遭到泄露，系统就无法检测到攻击者，因此词元和文档都可能暴露。

3.9 聚类性能评估

3.9.1实验设置

我们开发了 ClustCrypt 和 ClusPr 的可用版本，并在 GitHub[a],[b] 上公开发布。我们使用三个具有不同属性和规模的数据集评估 ClusPr 的性能，并将其聚类质量与其他在加密域或未加密域中运行的方法进行比较和分析。实验在一台配备两颗 10 核 2.8 GHz E5 Intel Xeon 处理器和 64 GB 内存的机器上进行。

为了评估 ClusPr 处理大数据的性能，我们使用了 `Amazon Common Crawl Corpus（ACCC）` 数据集 [94] 的一个子集。整个数据集的大小为 ≈ 150 太字节，包含各种基于Web的内容，例如博客和社交媒体内容。我们随机选择了 6,119 份文档，共组成一个 ≈ 500 GB 的文档集。第二个数据集名为 `Request For Comments(RFC)` [95]，具有特定领域，包含有关互联网和通信网络的文档。RFC 包含 2,000 份文档，总大小为 ≈ 247 MB。第三个数据集是 `BBC` [96]，不属于特定领域，包含技术、政治、体育、娱乐和商业等类别的新闻。它包含 2,225 份文档，大小为 ≈ 5 MB。选择这个小型数据集的原因是，与 ACCC 和 RFC 不同，BBC 的每份文档都很短，因此我们可以手动验证各簇的连贯性。对于每个数据集，我们将文档输入 Maui 关键词提取器 [97]，以识别在语义上代表

[a]https://git.io/fjDsq[b]
https://github.com/hpcclab/ClustCrypt

该文档的关键词。

3.9.2先前研究中的评估指标与基线方法

为评估所提出方案的性能，我们将其与另外四种方案进行比较，其中两种方案对明文数据进行聚类，另外两种方案对加密数据进行聚类。在前两种方案中，一种方案 $W2VKmeans$）基于K均值聚类 [98] ，其特征提取基于 Word2Vec [63] 嵌入。

另一种方案 $WordNet$ [99], 是K均值的一种改进版本，它根据输入数据生成同义词集，然后对这些集合应用 K均值聚类。WordNet 中的词元分布采用 Wu 和 Palmer [99]提出的边计数法进行。

用于比较的两种加密聚类方案分别是 S3BD [3], 和HK均值++ [41]。我们已在第2.2.1节讨论过 S3BD 和HK均值++ 。$ClustCrypt$ 是 S-ClusPr 的初步版本。它们的差异主要在于词元在各簇之间的分布方式。在 ClustCrypt 中，相关性仅根据贡献度和共现度指标计算；而在 S-ClusPr 中，则综合考虑相似度和差异度的大小来衡量相关性（详情请参见第 3.6 节.2）。

可以通过多种评估指标来量化簇集合的优劣。然而，评估聚类方案的性能并不像统计分类算法中的错误那么简单。具体而言，不应

考虑聚类标签的绝对值，聚类评估指标要么衡量聚类数据与真实类别集的相似程度及其分离情况，要么进行内部聚类验证。内部聚类验证是指同一类别的成员之间应比不同类别的成员更加相似，反之亦然。在实践中，大多数应用场景中并不总能获得类别标签信息，因此在这种情况下，内部验证指标是唯一可用的验证选项 [100, 101]。

由于所考虑的数据集没有真实值，我们选择基于聚类成员的统计分析来评估簇的评估指标。我们评估了三种广泛采用的聚类指标，即轮廓系数（SC）、Calinski-Harabasz 指数（CI）和戴维斯-布尔丁指数（DI）。

轮廓系数 *(SC)* 分数用于解读和验证簇内一致性。具体而言，该指标表示某个簇成员与其所属簇的相似程度，相较于其与其他簇的相似程度如何。SC 分数的取值范围为-1 至 $+1$，数值越高，表示该成员与其所属簇越匹配，与其他簇越不匹配。*Calinski-Harabasz* 指数 *(CI)*表示簇的定义清晰程度（即 分离程度）。簇的 CI 值根据簇间离散度之和与簇内离散度之和的比值计算。CI 值越高，表示聚类在主题上的分离程度越高（即 重叠程度越低），反之亦然。与 CI 指标类似，*Davies-Bouldin* 指数 *(DI)* 用于衡量各簇之间的分离效果；我们在评估中考虑该指标，是为了验证 CI 指标

对簇的评估。DI 根据簇内距离与簇间距离的比值计算。DI 值越低，表示聚类在主题上的分离程度越高，因此越理想。除了这些指标，我们还测量簇的连贯性，以评估各簇内基于主题的聚类质量。这是一种基于相似度的评估指标，用于计算某个簇中所有可能的词元两两相似度的平均值。实际上，连贯性反映了簇中的词元与某个特定主题的关联程度。随后，计算所有簇的连贯性平均值，以表示某种聚类方法的整体质量。

我们使用预训练的 Google News Word2Vec 模型 [63] 来确定任意两个给定关键词之间的相似度。该模型以 300 维向量表示三百万个短语。模型需要以文本数据集作为输入，从输入数据集中构建词汇表，并学习数据集中词语的向量表示。模型使用余弦相似度，为任意两个给定词元提供分数（$-1 \leq similarity\ score \leq 1$）。需要注意的是，预训练的 Word2Vec 模型仅适用于纯文本词元。因此，为了进行评估，我们在上传时不对词元加密。不过，所提出的方案假定词元已加密，且不利用纯文本词元的特性。

3.9.3评估结果

3.9.3.1评估轮廓系数（SC）分数。图 3.4 展示了在三个数据集上评估 SC 分数的结果，以及不同簇数下的结果

（横轴所示）。需要注意的是，在此实验中，W2V Kmeans、WordNet 和 HK-means++ 中的 K值是随机选取的，并会迭代变化。因此，我们计算了所有待考察 K 值对应的 SC 分数，并在图中以多个数据点展示。不过，其他方案（即 S-ClusPr、ClustCrypt、S3BD）不是迭代式的，因此每个方案确定的 K 值仅对应一个 SC 分数。

由于 ClustCrypt 和 S-ClusPr 方案中估算簇数的过程相似，因此可以看出，这两种方案分别为 BBC、 RFC和 ACCC 数据集生成了 69、65 和 133 个簇。 ACCC 是规模最大、范围最广（即 非特定领域）的数据集，因此其 K值最高。 RFC 并非最小的数据集，但由于其具有特定领域属性，因此其 K 值最低。

图 3.4 展示了 ClustCrypt、S-ClusPr 以及其他四种对比方案的 SC 指标结果。根据图中结果，综合所有数据集来看，表现最佳的是 WordNet 和 S-ClusPr。此外，S-ClusPr 在 RFC 数据集上的表现优于其他方案。相反，HK-均值++和 S3BD 在大多数情况下表现不佳。实验表明，由HK-均值++ 和 S3BD 生成的簇集，其簇内相似度较低。WordNet 和 S-ClusPr 的簇内相似度最高，因此在所有数据集上的表现均优于其他方案。

3.9.3.2 Calinski-Harabasz 指数（CI）评估。表 3.7 展示了 S-ClusPr 和其他四种方案的 CI 指标结果。根据该表，

图 3.4。 各数据集的轮廓系数（SC）指标。结果来自 S-ClusPr、*HK*-均值++、ClustCrypt（这些是基于加密的聚类方案）、W2V-Kmeans 和 WordNet 聚类方案（这些方案处理的是明文词元）。

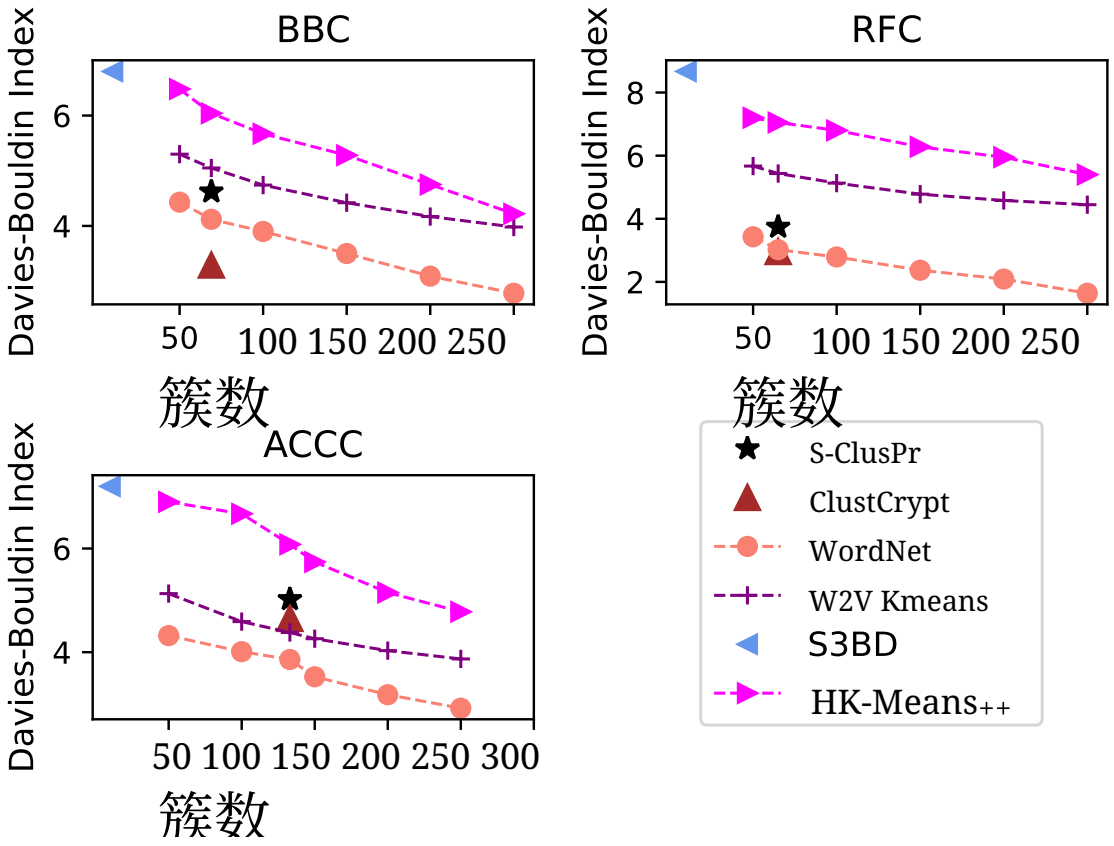


图 3.5。 采用不同聚类方案时各数据集的戴维斯-布尔丁指数（DI）。

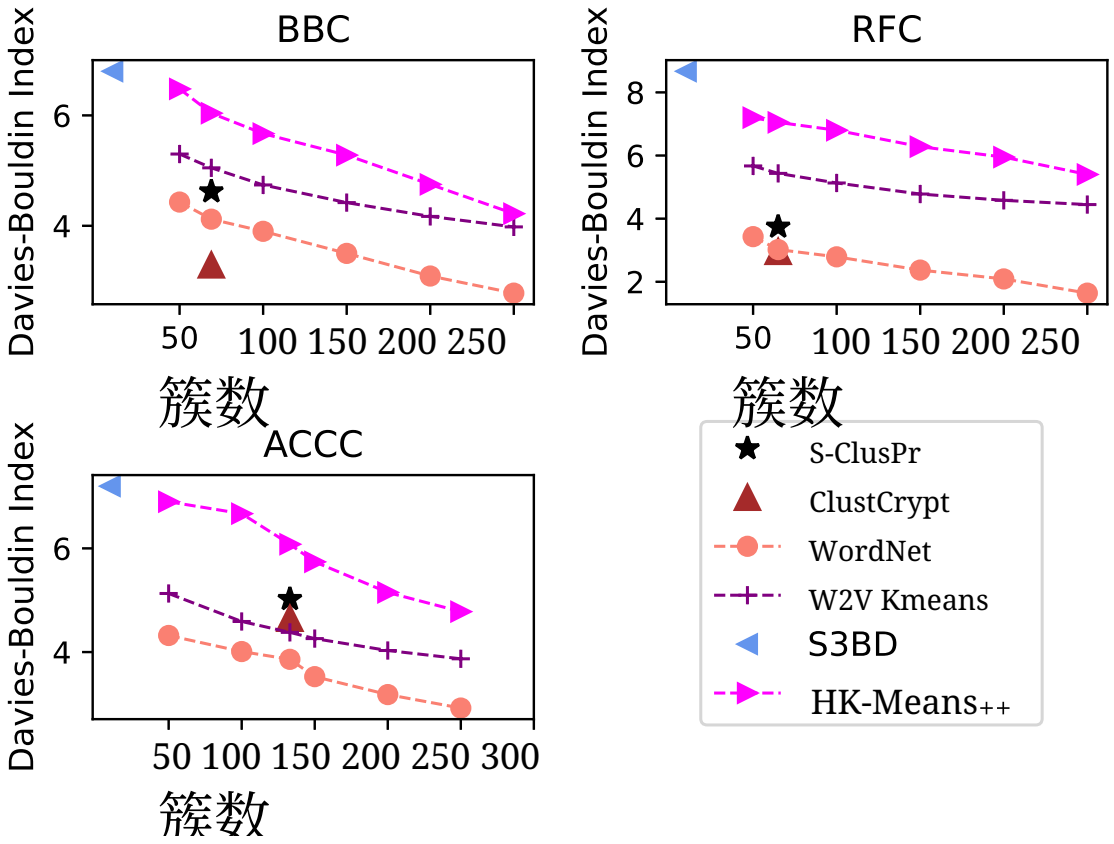


图 3.6。 每个数据集的簇一致性。

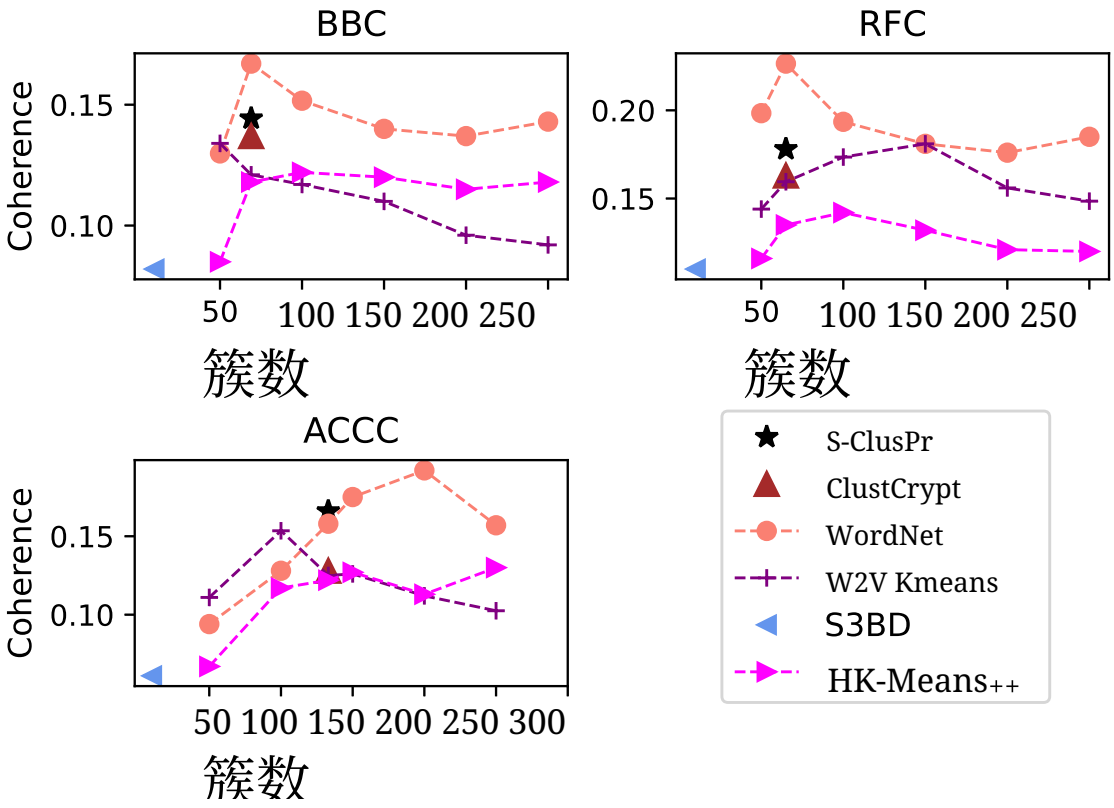


表 3.7. 数据集的 Calinski-Harabasz 指数。

(a) BBC

	方法					
数量簇	HK-簇均值++	WordNet	W2V Kmeans	S3BD	ClustCrypt	S-ClusPr
10	-	-	-	8.7	-	-
50	25.43	**277.53**	11.16	-	-	-
69	18.47	**253.60**	9.22	-	11.70	13.58
100	11.13	**203.87**	7.37	-	-	-
150	14.05	**164.43**	5.81	-	-	-
200	10.17	122.51	4.93	-	-	-
250	12.02	97.15	4.38	-	-	-

(b) RFC

	方法					
数量簇	HK-均值++	WordNet	W2V Kmeans	S3BD	ClustCrypt	S-ClusPr
10	-	-	-	1247.20	-	-
50	1730.26	4320.63	**60380.05**	-	-	-
65	1945.42	3980.75	**51564.61**	-	23760.64	**29439.30**
100	1834.64	3660.78	**24374.17**	-	-	-
150	1684.47	3110.25	**18684.33**	-	-	-
200	846.71	2572.89	16746.74	-	-	-
250	436.43	1834.58	15139.11	-	-	-

与 BBC数据集相比，无论采用何种聚类方案，RFC 簇的 CI值都较大。值得注意的是，我们在 ACCC 数据集上也观察到了相同现象，但由于篇幅有限，未展示其表格。RFC 表现更优，是因为它是一个领域特定数据集，主题数量比另外两个数据集少。在表3.7b中可以看到，尽管 W2V-Kmeans 在大多数K 值上显著优于其他方案，但 WordNet、ClustCrypt 和 S-ClusPr 也取得了令人满意的 CI值，表明其簇划分良好。

3.9.3.3 评估 Davies-Bouldin 指数（DI）。图 3.5 展示了 S-ClusPr 和对比方案得到的簇的 DI 值。在大多数情况下，我们观察到，增加簇数会降低 DI 值。这是因为，通常在给定数据集上配置聚类方案以构建更多簇，会使每个簇内部的连贯性更高。

根据图示，我们可以看出，WordNet 方案优于其他方案。S-ClusPr 的 DI 值处于可接受范围内，这表明与大多数其他方案相比，该方案能够在不同簇之间实现具有竞争力的分离效果。另一方面，HK-means++得出的较高 DI 值表明簇分离效果较差。

3.9.3.4 评估簇一致性。图 3.6 展示了使用不同聚类方案时，三个数据集上的簇一致性。使用 S-ClusPr，分别为 BBC、RFC 和 ACCC 数据集创建了 69、65 和 133 个簇。由于 ACCC 是规模最大、范围最广的数据集（即非特定领域数据集），因此其 K 值最高。RFC 并非最小的数据集，但由于其具有特定领域属性，因此其 K 值最低。出于同样的原因，在这三个数据集中，S-ClusPr 在 RFC 数据集上取得了最高的连贯性值（≈ 0.16）。

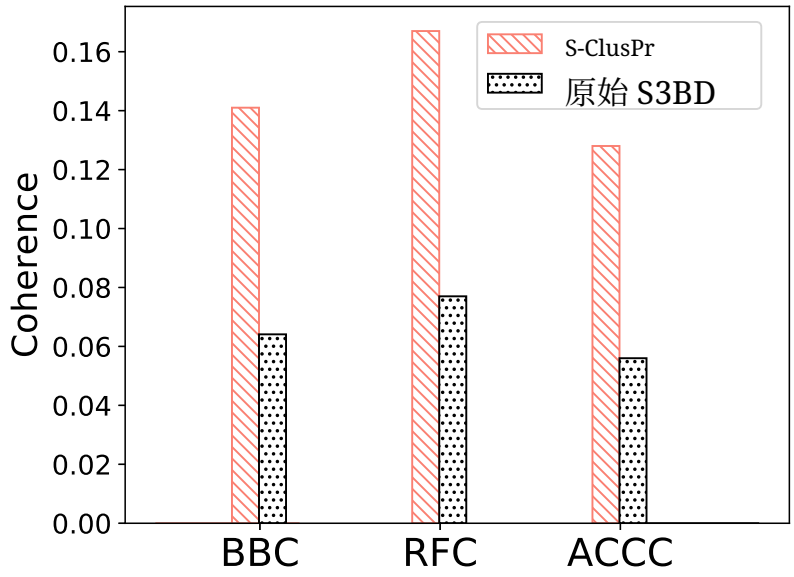
与 ClustCrypt 相比，我们注意到，对于 BBC 和 RFC 数据集，S-ClusPr 的连贯性提升微乎其微（$\approx 6\%$）。不过，对于ACCC 数据集，S-ClusPr 将连贯性提升了约 31%。

对基于明文的方案进行分析后发现，WordNet 簇的连贯性值最高。这是意料之中的，因为加密的

聚类方案（例如，S-ClusPr）很难胜过未加密方案，因为它们无法获取词元的语义 [99] 来构建簇。然而，我们观察到，S-ClusPr 所提供的连贯性可与 K-均值方案相媲美。具体而言，对于 RFC 和 BBC 数据集，S-ClusPr 的连贯性值高于 K-均值方案。

为评估 S-ClusPr 估算的簇数（K）是否合适，我们将 K-均值和 WordNet 都配置为对所研究的数据集使用估算的 K 簇数。根据图示，对于 RFC 和 BBC，S-ClusPr 建议的簇集合比 K-均值具有更高的连贯性，与 WordNet 的连贯性则相当。对于 ACCC，S-ClusPr 在连贯性方面甚至优于 WordNet。

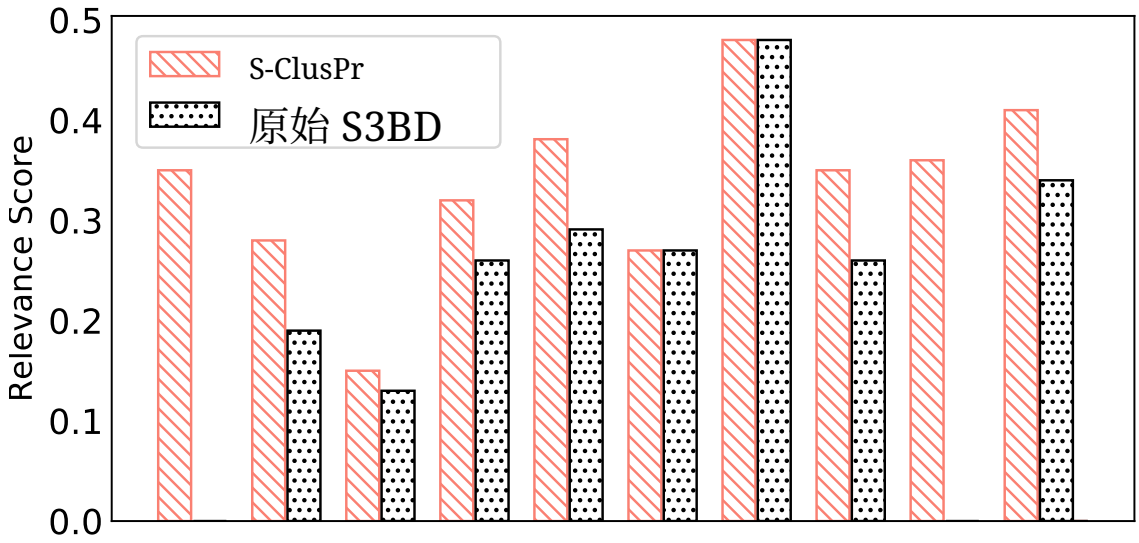
图 3.7. 比较在所研究的数据集上，使用 S-ClusPr 进行聚类与 S3BD 原始聚类的影响。



3.9.3.5 分析 S-ClusPr 对可搜索加密的影响

系统。 本研究的目标之一是提升 S3BD 安全搜索系统的性能。因此，我们将 S-ClusPr 集成到 S3BD 中，并将生成的簇的连贯性与其原有的聚类方案进行比较。原有方案会预先确定 $k = 10$的值，且其中心选择只考虑共现。在此实验中，我们旨在评估 S-ClusPr 在 S3BD 中对所研究的三个数据集带来的改进。在此实验中，`BBC`、 `RFC`和 `ACCC` 对应的 K 估算值分别为 69、65 和 133。**对 S3BD 聚类连贯性的影响**。 图 3.7 显示，对于所有研究的数据集，S-ClusPr 生成的簇的连贯性都明显高于 S3BD 原有的聚类方案。这表明，根据数据集特征确定簇数并选择中心词元

图 3.8. 比较 BBC数据集 中使用 S-ClusPr 与原始 S3BD 聚类的搜索结果相关性。相关性数值依据 TSAP@10 评分指标计算。



基于中心性概念的方法是有效的。我们的假设是，这种效率能够提高准确率，并提供更多相关的语义搜索结果。这是因为簇中的词元与簇的主题更加契合，因此能够更有效地进行剪枝。为了进一步验证这一假设，接下来的实验将重点考察 S-ClusPr 对搜索质量的影响。**对 S3BD 搜索准确率的影响**。 本研究旨在通过提升簇的连贯性，最终提高搜索准确率，方法是

表3.8。所研究的每个数据集的基准查询。

ACCC Dataset	BBC Dataset	RFC Dataset
Orlando Magic	News Update	Internet
Samsung Galaxy	Top Movies	TCP
Baseball routine	Recent Attacks	Fiber Doctor
Recommendation	Endangered Animals	Wifi
North America	Score Updates	IoT
Tennis Tournament	Champions League	Radio Frequency
Holy Martyr	World Health Issue	UDP
Library	People and Business	Edge Computing
Stardock	China Market	Encryption Schemes
Orthodox Church	European Stock Exchange	Broadcasting

检索更多相关文档。为了评估这种改进带来的影响，本部分将比较并分析：与采用原始聚类方法的情况相比，使用 S-ClusPr 的簇会如何影响 S3BD 系统的搜索准确率。为进行评估，我们生成了一组包含 10 条基准搜索查询的查询，这些查询列于表3.8中。

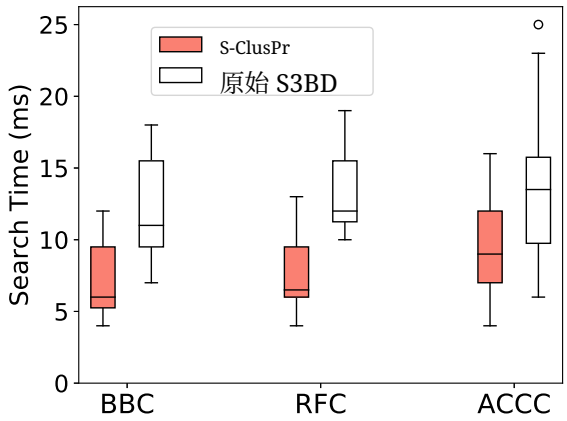
为衡量每个查询的搜索结果相关性，我们采用*TREC*风格平均准确率 评分方法[102]。该方法基于N召回率-精确率概念，得分通过 $\sum r_i/N$计算，其中 r_i 表示$_{i=0}$

i^{th} 检索文档的得分，而 N 是截断数量（搜索结果中的元素数量），我们将其设为 10。因此，我们称之为*TSAP@10*。

我们仅针对 RFC 数据集及其基准查询衡量 TSAP@10 分数。原因在于该数据集具有特定领域属性，而且能够切实判定检索到的文档的相关性。为了将 S-ClusPr 提供的相关性与原始 S3BD 聚类进行比较，我们将基准查询应用于 S3BD 搜索系统。图 3.8 展示并衡量了采用两种聚类方案时各查询结果的相关性分数。根据图中结果，对于大多数查询，S-ClusPr 聚类提供的相关性分数更高。对于 TSAP@10 分数相同的两个查询，它们的检索文档列表相同。此外，S-ClusPr 簇能为新闻更新 和中国市场 基准查询提供分数，而原始 S3BD 簇无法检索到这些查询的任何相关文档。

对 S3BD 搜索时间的影响。图 3.9 展示了每个数据集上基准查询的总搜索时间。搜索时间以执行每个查询的周转时间衡量，即从发出查询到收到结果集的时间。为消除计算系统中任何随机性的影响，我们对每组基准重复搜索 10 次，并以箱线图的形式报告结果。图中显示，使用 S-ClusPr 聚类时，搜索时间显著短于使用原始 S3BD 聚类的情况。较长的搜索时间会影响大数据搜索操作的可扩展性和实时性。分析图 3.7 至 3.9 可知

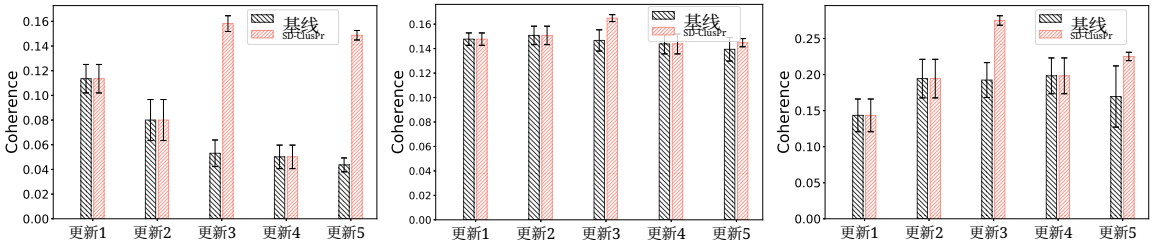
图 3.9. 使用 S-ClusPr 进行聚类与使用原始 S3BD 聚类时 S3BD 的搜索时间。



集成　在搜索系统中应用 S-ClusPr 不仅能提高准确性，而且

也使其速度更快、可扩展性更强。

图3.10。 应用 SD-ClusPr 并启用或不启用重新聚类选项时，三个研究数据集在不同更新下簇的连贯性。



(a) BBC数据集　(b) RFC 数据集　(c) ACCC 数据集

3.9.3.6 动态方案的聚类一致性评估。

在本部分中，我们分析动态聚类方案（SD-ClusPr 和 FD-ClusPr）的有效性。我们在第3.7节中提到，FD-ClusPr 是 SD-ClusPr 的一种特例。因此，评估时我们只考虑 SD-ClusPr 方案。为此，我们利用所研究的三个数据集构建若干子集，每个子集作为一次批量更新。具体而言，对于每个数据集，我们考虑一个基于 500 个

文档的现有簇集合。然后，我们进行五次抽样，创建一个包含五次更新的列表，每次更新都包含一组文档。List U 包含更新名称及每次更新大小的配对，如下所示：

$U =<$ (U_1，25)，(U_2，50)，(U_3，100)，(U_4，20)，(U_5，200) $>$。为确保结果不会偏向任何特定样本，我们独立执行了 10 次抽样过程，并在结果中报告分析的均值和 95%置信区间。我们将 U_3 和 U_5 设得较大的原因，是为了考察 SD-ClusPr 在重新聚类中的决策。为从簇一致性的角度评估该方案，我们基于 SD-ClusPr 构建了一个不考虑重新聚类的基线 版本。该基线仅根据现有簇进行聚类（如算法 3 所述），以适应新的更新。

图3.10a、图3.10b 和图3.10c 分别展示了对 BBC、 RFC 和 ACCC 应用 SD-ClusPr 方案时，五种不同批量更新的簇一致性。在图3.10a 中，我们观察到，对于 U_3 ，基线中的簇一致性有所下降，而 SD-ClusPr 的簇一致性比前者提高了约 105%。我们在 U_5中观察到类似的簇一致性变化模式。基线的最低簇一致性出现在 U_5。相反，在 SD-ClusPr 中，我们观察到 U_5的簇一致性提高了约 115%。

根据图 3.6，针对 RFC 数据集形成的簇连贯性最高。同样，在图3.10b中，与其他数据集相比，我们观察到所有更新的连贯性都最高。就基线而言，我们观察到 SD-ClusPr 略微提升了 U_3 和 U_5的连贯性。由于

文档的领域针对性更强，因此簇之间不会因一次更新到下一次更新而显著丧失连贯性。因此，我们没有观察到 SD-ClusPr 带来的显著提升。与 BBC 和 RFC类似，在图 3.10c中，我们观察到 ACCC 数据集的连贯性有所提升。具体而言，U_3和 U_5 的连贯性分别提升了约 45% 和 35%。

根据这些实验，我们得出结论：即使对于动态数据集，ClusPr 方案也能提高聚类一致性。具体而言，我们观察到，对于足够大的批次（例如 U_3 和 U_5），SD-ClusPr 会决定重新聚类，从而显著提高聚类一致性。

3.10 总结

本章提出两种安全聚类方案，即以可信应用形式实现的 ClustCrypt 和 ClusPr，适用于三种类型的非结构化数据集：静态、半动态和动态。这些聚类函数基于数据集的统计特征，用于：（**A**）确定合适的簇数；（**B**）为簇填充主题相关的词元；以及（**C**）根据底层数据集的动态性调整簇集。通过在三个不同的测试数据集上将 ClusPr 与文献中的其他方案进行比较而获得的实验结果表明，簇一致性提升了 30% 到 60%。此外，我们发现，在隐私保护企业搜索系统中采用 ClusPr，可将搜索时间缩短最多 78%，同时将搜索准确率提高最多 35%。

在下一章中，我们将探讨如何在不危及非结构化数据机密性的前提下，对其进行安全的企业搜索。

第4章：云端隐私保护企业搜索的基于边缘的智能

4.1 概述

基于云的企业搜索服务例如，AWS Kendra）通过为大数据所有者提供便捷的实时搜索解决方案，吸引了他们。然而，为隐私保护数据提供智能搜索时，这些服务必须访问用户的搜索历史，这会进一步危及其隐私。为解决隐私问题，本研究的核心思路是将搜索的智能方面与模式匹配方面分离。根据这一思路，搜索智能由本地边缘层提供，而共享云层仅作为全面的模式匹配搜索工具。我们提出了边缘智能 （SAED机制），在边缘层提供语义搜索和个性化搜索智能，同时在云层搜索时保护隐私。在边缘层，SAED使用基于知识的词汇数据库扩展查询，以涵盖其语义。SAED通过循环神经网络模型实现个性化搜索，该模型能够学习用户兴趣。词嵌入模型则用于检索与搜索查询具有语义相关性的文档。

4.2 问题陈述

理想情况下，数据所有者希望获得一种隐私保护云服务，能够以实时方式提供语义化和个性化的可搜索性，同时不会使其资源受限的（轻量型）客户端设备（例如，智能手机）不堪重负。A

大量研究已针对云中的隐私保护企业搜索服务展开，其目标是保护用户的敏感数据免受内部和外部攻击者的侵害。然而，这些研究大多未能检索出在语义上与上下文和用户兴趣相关的搜索结果（即，个性化搜索）。此外，这些研究通常依赖客户端设备，并要求其承担大量开销来执行安全查询处理或加密/解密用户文档。

为了满足特定用户的所有上述需求，本研究的核心思路是将企业搜索的智能方面与其模式匹配方面分离。

4.3 SAED：智能边缘赋能企业搜索系统

4.3.1架构概述

本部分将概述 SAED 系统，该系统能够在云上实现智能且安全的企业搜索。系统采用三层架构，如图1.2所示，具体说明如下：

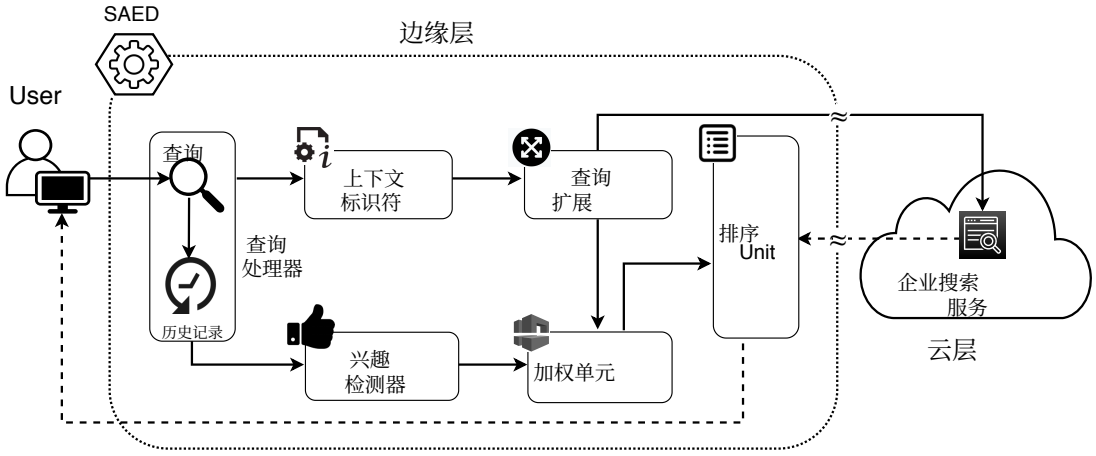
- 客户端层 (例如，智能手机、平板电脑)包含一个轻量级应用程序，提供用户界面，供用户上传文档并在云端搜索这些文档。数据集既可以由用户上传，也可以由拥有这些数据的组织上传。

- 边缘层 提取上传至云层的文档中的代表性关键词，并在云层构建索引。收到来自客户端层的搜索查询后，边缘层上的 SAED 系统会结合查询语义和用户兴趣提供智能服务。边缘层位于客户端场所，因此被视为诚实且安全的系统。为提供安全的企业搜索服务，边缘层会加密上传的数据和搜索查询。此外，它还会在将结果集返回给客户端层之前进行解密。

- 云层包含众多高端服务器，用于存储（加密的）数据，并执行大规模计算，以便对索引 [53, 3] 进行穷尽搜索。索引可以根据其关键词所对应的潜在主题进行聚类（请参阅我们之前的

works [3, 1] 以了解更多详情）。

图4.1. SAED 系统在边缘层中的架构概述，以及其作为三层企业搜索服务一部分的架构。SAED 通过识别查询上下文并结合用户兴趣来提供语义搜索。随后，SAED 的查询扩展与加权单元分别纳入语义信息，并确保结果的相关性。实线和虚线分别表示用户与云层之间以及云层与用户之间的交互。



在图4.1中，我们展示了 SAED 的各个组件及其相互之间的交互。首先，系统接收用户提供的搜索查询，并由查询处理器进行处理；该处理器会跟踪用户的搜索历史，并初始化上下文识别器单元，其任务是提取上下文并消除查询短语的歧义。随后，查询扩展单元根据提取的上下文主动扩展查询，并构建一个查询集。为了实现个性化搜索，SAED 的兴趣检测器单元利用用户的搜索历史来识别其兴趣，并根据查询集各元素（即扩展查询）与用户兴趣的相关性为其赋予权重。在云层上完成模式匹配阶段后，检索到的文档会返回至边缘层上的 SAED。接下来，排序单元利用所分配的

根据检索文档与用户兴趣的相关性对其进行排序，并生成一个检索文档列表，记为 D_θ，然后发送至用户设备。在接下来的部分中，我们将详细介绍 SAED 系统的各个单元。

4.3.2查询上下文识别

识别给定搜索短语的上下文，对于引导搜索找到用户意图表达的语义至关重要。以 cloudcomputing 作为搜索查询为例，如果没有恰当的上下文识别，返回的文档集可能会包含关于 sky 和 climate的文档；而高效的上下文识别器则能够识别正确的语义，并将搜索引导至围绕 distributed、 edge、 fog和 cloud computing的主题。事实上，上下文识别有助于查询扩展单元构建一个查询集，其中包含围绕相关关键词进行扩展的多样化查询，这些关键词在语义上能够代表搜索查询，从而提升结果的相关性。

先前的上下文识别研究（例如， [104, 105, 62]）存在以下不足：首先，它们通常假设查询中的每个关键词都同等重要，并通过对查询中关键词的嵌入向量取均值来识别查询上下文。然而，查询中的关键词并非都必然有助于识别上下文。例如， various cloud providers 中的关键词 various 对上下文没有任何意义，因此可以删去。其次，现有研究所用的嵌入方法总是为给定关键词提供相同的表示，而不考虑其所处的上下文。这对于歧义关键词尤其成问题，因为它们的含义会随查询

上下文而变化。例如，在前述示例中，当 cloud 与 computing 搭配使用时，其嵌入向量应不同于在给定查询中与 weather 搭配使用时的嵌入向量。第三，现有方法只考虑常见关键词的嵌入向量，却丢弃了大多数不在 Word2Vec [64, 106]词汇表中的命名实体（例如，姓名和地点）。例如，考虑查询 best selling books of J.K. Rowling； Book 和 Sell 被识别为查询上下文，而 J.K. Rowling则被丢弃。然而，我们的分析表明，短查询短语的上下文往往与被丢弃的命名实体存在上下文关联。

为克服这些不足并识别给定查询的实际上下文，我们提出采用整体方法，提取查询关键词所表达的语义，并根据每个关键词的重要性赋予相应权重。上下文识别单元的主要输出是一组关键词，记为C，这些关键词共同代表查询的上下文。

具体而言，为了消除对查询 Q 的语义没有贡献的不重要关键词，上下文识别单元使用 $Yake$ [107]，这是一种无监督关键词提取器，可剔除查询中的不重要关键词。剩余的关键词（即 经过精简的查询，记作 Q' 集合）将用于上下文识别。为了解读 Q' 的真实语义，该单元利用 WordNet 的 Lesk算法 [106] 来消歧每个关键词$q \in Q'$。Lesk算法基于这样一个事实：给定句子（查询）中的关键词往往暗示某个特定主题。对于关键词 q，Lesk 可以确定其真实

语义，方法是将 q 的词典释义与 Q' 中的其他关键词（即 $Q' - \{q\}$）进行比较。设 c_q 为表示 q 上下文的关键词集合。那么， Q 的上下文确定为 $C = \cup_{\forall q \in Q'} c_q$。最后，上下文识别器使用 WordNet 从 Q 中识别命名实体，并将其视为上下文的一部分，但单独归入一个集合，记作 N。之所以单独设立一个集合，是因为在 SAED 的其他单元中，我们会对 N 和 C 采取不同的处理方式。

算法 4： 用于检测给定查询上下文的伪代码，位于 SAED 的上下文识别单元中。

输入： 查询 Q
输出： C：表示 Q 上下文的关键词集合，
N： Q 中的命名实体集合

```
1  函数 contextIdentification(Q):
2      Q' ← 使用 Yake 算法从 Q 中提取关键词。
3      对每个 q ∈ Q 执行
4          如果 q ∈ 命名实体 则
5              N ← N ∪ {q}
6          end
7          else
8              如果 q ∈ Q' 则
9                  E_q ← 使用 Lesk 算法基于 Q' − q 定义 q
10                 c ← 使用 Yake 算法提取 E_q 的关键词集合。
11                 C ← C ∪ c
12             end
13         end
14     end
15     返回 C , N
16 结束
```

算法 4 提供了用于识别传入查询 Q 的上下文的伪代码。伪代码的输出是两个集合，即 C 和 N，它们共同表示 Q 的上下文。在伪代码的步骤 2 中，使用 Yake 算法筛选 Q ，提取其中的重要关键词，并生成

Q' 集。通过与 WordNet 进行比对，识别出 Q 的命名实体，并构成集合 N（步骤 4–6）。接着，在步骤 8–12 中，对每个关键词 $q \in Q'$，使用 Lesk算法消歧 q，并结合 Q' 中的其他关键词找到其确切定义。定义中的重要关键词构成 Q 的上下文集（C）。

4.3.3查询扩展单元

查询扩展单元负责根据查询关键词的相关同义词主动扩展关键词，这些同义词须与已识别的上下文相符。忽略查询上下文、盲目考虑所有同义词（如 [104, 105, 62, 3], 中所做的那样），会导致找到不相关的文档。因此，该单元利用 Q 的上下文（即 C 和 N），仅查找语义上与查询上下文接近的同义词集合，并将其表示为 P。

Word2Vec [63] 是一种浅层神经网络模型，可通过训练生成关键词的向量表示，使给定两个关键词的余弦相似度能够反映它们之间的语义相似度。因此，为主动扩展每个关键词 $q \in Q$，查询扩展单元使用经过 Google新闻数据集 [108],预训练的 Word2Vec，构建候选同义词集合，记为 s_q。设 s_q^i 是 q 的同义词（即 $s_q^i \in s_q$）。那么， s_q^i 与查询上下文之间的相似度记为 $sim(s_q^i,\ C)$，其定义基于 C中各元素的相似度之和，如公式 4.1 所示。

$$sim(s_q^i, C) = \sum_{\forall C_j \in C} sim(s_q^i, C_j) \tag{4.1}$$

只有当 s_q^i 与查询上下文在语义上足够接近时，才会将其选为 P的一个元素。为确定是否足够接近，我们将 $sim(s_q^i,\ C)$ 与 s_q所有成员之间成对相似度的均值进行比较（即 $sim(s_q^i,\ C) > \mu_{\forall q \forall j}(sim(s_q^j,\ C))$），并要求其大于该均值。需要指出的是，由于 C和N 中的元素表示 Q的上下文，因此它们也会被添加到 P中。

算法 5 为生成扩展查询集 P 提供了高层伪代码。在伪代码的步骤 2–7 中，会为每个 q 生成同义词集。接下来，计算每个词 s_q^i 与 C 之间的相似度。步骤 8 中使用这些相似度值计算所有候选查询的平均相似度。在步骤 9–15 中，将语义接近度大于 μ 的候选同义词纳入扩展查询集 P。最后，在步骤 16 中，通过纳入上下文集和命名实体来扩展集合 P。

4.3.4用户兴趣检测

检测用户的搜索兴趣对于提供个性化搜索至关重要。在 SAED 中，兴趣检测通过分析两个因素实现：(A) 用户的搜索历史；以及 (B) 用户对先前搜索查询的检索结果所作出的反应。可以根据用户选择的结果或其浏览这些结果所花费的时间来检测这一点。

令 Δ' 表示发送给用户的全部检索到的文档，τ 表示用户感兴趣的文档。我们有 $\tau \subseteq \Delta_{\prime}$。

算法 5： 基于上下文扩展查询的伪代码，位于 SAED 的查询扩展单元

输入： Q，C，N

输出： P：扩展查询集

```
函数 QueryExpansion(Q, C, N)
    对于每个 q ∈ Q 执行
        s_q ← 使用 WordNet 获取 q 的同义词集
        遍历 s_q^i ∈ s_q do
            sim(s_q^i, C) ← Σ_{∀C_j∈C} sim(s_q^i, C_j)
        end
    end
    μ ← 计算所有 q ∈ Q, ∀s_q^j ∈ s 上的均值 sim(s_q^j, C)
    遍历 q ∈ Q 执行
        遍历 s_q^i ∈ s_q do
            如果 sim(s_q^i, C) > μ 则
                添加 s_q^i 设置 P
            end
        end
    end
    P ← P ∪ C ∪ N
    返回 P
结束
```

因此，可以根据 τ 的主题推导出用户兴趣。兴趣检测单元使用一种现有的文档分类模型 [109], ，该模型基于 Na¨ıve Bayes（NB）方法运行，用于确定 τ 的主题，记为 t_τ。我们还对 t_τ 进行多数投票，以找出用户的主要兴趣。重复此过程，以存储用户的n个先前搜索兴趣数据。由于这些数据是从每次成功搜索中收集的，因此具有序列特征。通过分析用户先前的搜索兴趣，边缘层训练一个基于循环神经网络的预测模型 [110] ，以预测用户的搜索兴趣。对于 SAED，由于数据不包含长期依赖关系，并且为了保持模型简单及实时性，我们不采用堆叠式（即更深层的）模型，而是使用收集到的特定于用户的历史搜索数据训练一个多对一基础 RNN 模型 [111]。

4.3.5加权单元

SAED 学会用户兴趣后，实现上下文感知的个性化企业搜索的下一步，就是确定上下文扩展查询（即 P 的元素）与用户兴趣的接近程度。事实上，并非所有扩展查询在解释查询时都具有相同的重要性。因此，加权单元 的目标被定义为量化每个扩展查询与用户兴趣的接近程度。随后，在云层上完成搜索操作后，SAED 的排序单元 会使用这些权重对结果集进行剪枝和排序。

以往的加权方案（例如， [53, 3, 62, 59, 105]）通常会使用词

频率方法（例如，词频-逆文档频率 [3]），并忽略用户兴趣。相比之下，SAED 的加权过程会根据两个因素量化每个扩展查询 $p \in P$ 的重要性：(A) p的类型，即它是直接属于上下文（C 和 N 集合），还是从中派生而来；以及 (B) p 与用户兴趣的语义相似度。

具体而言， P 中直接表示查询上下文或命名实体的元素（即 $\forall p|p \in P \cap$ （$C \cup N$））明确表明了用户的搜索意图，因此，无论用户的兴趣如何，都应对这些元素进行加权。进一步分析表明，查询中可能存在的命名实体代表了搜索意图，因此，让搜索结果偏向这些实体可以提高用户满意度。因此，$\forall p|p \in$ （$P \cap N$）被赋予最高权重。最高权重由领域专家确定；不过，在实验中，我们将其设为 $\eta_{max} = 1$。我们将 $q \in Q$的贡献度 定义为：由于 q 而添加到 C 的关键词数量（记为 C_q）与 C 的基数之比。令 η_p 表示 $p \in P$ 的权重。那么，对于 P 中处于查询上下文中的元素（即 $\forall p \in$ （$P \cap C$）），η_p 根据与 p对应的查询关键词 q 的贡献度计算。公式 4.2 对 η_p 的计算方式进行了形式化表示。

$$\eta_p = \frac{\eta_{max} \cdot |C_q|}{|C|} \tag{4.2}$$

对于如第4.3.3节所述由 C 中的元素派生的 p，其权重分配方式是考虑 p 与用户兴趣 θ 之间的语义相似度（即 $\forall p|p \in P - (C \cup N)$）。也就是说，$\eta_p = sim(p, \theta)$。

算法 6 提供了为扩展查询集 P 分配权重的高级伪代码。该算法以 P、C、N 和最高权重值 η 作为输入。迭代地为 $\forall p$ 分配权重后，它会返回与相应 p 映射的权重，并将其表示为哈希映射 ϖ。在伪代码的步骤 2 中，θ 获取用户搜索兴趣，该兴趣通过利用预训练文档分类器和基础 RNN 模型来识别。在接下来的步骤中，哈希映射 ϖ 初始化为包含与相应 p 映射的权重。

总的来说，在步骤 4–15 中，每个 p （记作 ϖ_p ）的权重都根据其类型计算。具体而言，在步骤 5–7 中，ϖ_p，其中 $p \in$ （$P \cap N$）通过直接指定 η来设置。在步骤 8–11 中，p，其中 $p \in$ （$P \cap C$）根据其对上下文 C的贡献度加权。首先，确定 q，以生成 p，并通过 η 与相应 q的 C 中添加的关键词总数之比，计算 p 的权重 ϖ_p。在接下来的步骤（12–14）中，ϖ_p，其中 $p \in P -$ （$C \cup N$）根据其与 θ的语义相似度计算。最后，算法通过返回填入与相应 q 对应权重的哈希映射 ϖ 结束（步骤 16）。

4.3.6排序单元

扩展查询集 P 形成后，云层会对索引结构中的每个 $p \in P$ 执行字符串匹配。需要指出的是，如果用户选择执行安全搜索，则在将 P 的元素发送到云层之前，会先对其加密。此外，在我们先前的研究 [1], 中，我们提出了让云层对索引结构进行簇划分，并仅在与查询相关的簇上执行模式匹配的方法。

算法 6： 为扩展查询赋权的伪代码

输入： P，C，N，η

输出： ϖ

函数 `weighting`(P，C，N，η)

- $\theta \leftarrow$ 预测用户的搜索兴趣
- $\varpi \leftarrow$ 初始化哈希映射，用于存储与权重对应的相应的关键词
- **对每个** $p \in P$ **执行**
  - **如果** $p \in N$，**则**
    - $\varpi_p \leftarrow \eta$
  - **end**
  - **否则，如果** $p \in C$，**则**
    - $\varpi_p \leftarrow \dfrac{\eta_{max} \cdot |C_q|}{|C|}$
  - **end**
  - **else**
    - $\varpi_p \leftarrow \mathrm{sim}(p, \theta)$ /*计算相似度并将其存储在哈希映射中 */
  - **end**
- **end**
- 返回 ϖ

结束

云层将得到的文档集（记作 Δ）返回给边缘层，由 SAED 的排序单元根据相关性和用户兴趣对其进行排序，并生成文档列表（称为 Δ'）供用户查看。对于文档 $\delta_i \in \Delta$，其排序分数记作 γ_i，通过汇总 δ_i 中每个 $p \in P$ 的重要性值，并结合其权重（η_p）计算得出。通常根据词频-逆文档频率 分数 [112]来衡量 δ_i 中 p 的重要性。因此， γ_i 根据公式 4.3 正式计算得出。

$$\gamma_i = \sum_{\forall p \in P} \left(\eta_p \cdot TF\text{-}IDF(p, \delta_i) \right) \tag{4.3}$$

p 中 δ_i 的 TF-IDF 分数根据 δ_i中 p 的频率与 Δ所有文档中 p 的逆文档频率来定义。有关如何计算 TF-IDF 分数的详细信息，请参阅 [112]。排序单元计算出所有 $\delta_i \in \Delta$的排序分数后，系统会根据排名按降序排列文档，从而生成文档列表 Δ'，其中包含各个 δ_i，并将其显示给用户。

4.4 SAED 作为可插拔模块的企业搜索解决方案

SAED 的优势在于，它独立于部署在云层上的企业搜索服务。也就是说，使用 SAED 既不会干扰基于云的企业搜索服务，也不意味着需要对其进行任何更改。SAED 可以接入任何企业搜索解决方案。它在本地边缘层提供智能搜索功能，并将云层仅用于大规模模式匹配。整个 SAED 解决方案使企业搜索转变为语义化、个性化且机密的服务。

在本项工作中，我们让 SAED 同时与 AWS Kendra 和 S3BD 协同工作。在使用 AWS Kendra 的情况下，查询扩展单元将扩展查询集 P 发送至 Kendra，以便针对亚马逊云上的数据集搜索每个关键词 p。SAED 接收检索到的文档并进行排序，然后将其传送至客户端层。在实现中，我们仅向用户展示检索结果列表中的前10篇文档。同样地，我们将 SAED 接入 S3BD，以便在云上执行机密语义搜索。由于 S3BD 维护着一种加密索引结构，必须针对每个搜索查询遍历该结构，因此P 中的元素必须先加密，然后才能交给云层。我们还验证了 SAED 与 AWS Kendra 一起使用且数据集已加密时的表现。我们注意到，即使将 Kendra 设置为使用加密数据，SAED 仍能实现智能搜索。关于 SAED 与 AWS Kendra 和 S3BD 结合使用的性能测量与分析将在下一节中详细介绍。

4.5 SAED 性能评估

4.5.1实验设置

我们已经开发出一个完全可运行的 SAED 版本，并通过我们的 GitHub[a] 页面公开提供。为了对 SAED 在企业搜索解决方案上的性能进行全面评估，我们开发了使其能够同时与 S3BD [3] 和 AWS Kendra [113]协同工作的版本。S3BD 已经具备查询扩展和加权机制，但我们将其停用，并设置为使用 SAED 生成的扩展查询。在实验中，SAED 与 S3BD 的组合表示为 SAED+S3BD。同样地，SAED 与 AWS Kendra 的组合表示为 SAED+Kendra。

我们使用两个不同的数据集评估 SAED，即 `Request ForComments（RFC）`和 `BBC`，它们具有不同的属性和规模。我们选择 `RFC` 数据集，是因为它具有特定领域属性，包含 4,951 篇有关互联网和无线通信网络的文档。相比之下， `BBC`数据集更加多样化，其中包含 2,224 篇新闻文档，分属政治、娱乐、商业、体育和技术五个不同类别。

为进行全面评估，我们采用了系统化指标和基于人工的反馈，具体如第 4.5.3 节所述。我们在本地边缘计算系统中的一台虚拟机（VM）上部署并测试了 SAED。该虚拟机配备两颗 10 核、2.8 GHz 的 E5 至强处理器，拥有 64 GB 内存，运行 Ubuntu 18.4 操作系统。

[a]https://github.com/hpcclab/SAED-Security-At-Edge

Ta**表 4.1.** 为 RFC 和 BBC 数据集制定的基准搜索查询 .

BBC数据集	**RFC 数据集**
欧盟委员会 (EC)	网络信息 (NI)
议会档案 (PA)	主机网络配置（HNC）
2020年顶级摄像头手机（TCP）	数据传输（DT）
信用卡欺诈（CCF）	服务扩展（SE）
动物福利法案（AWB）	传输层 (TL)
盗版与版权问题 (PCI)	消息认证 (MA)
汽车与房地产市场 (CPM)	网络访问 (NA)
橄榄球联盟 (RFL)	互联网工程 (IE)
维也纳歌剧 (OV)	光纤通道 (FC)
Windows 操作系统 (WOS)	流媒体服务 (SMS)

4.5.2基准查询

我们用于开展实验的数据集均未提供任何基准。因此，在评估 SAED 的性能之前，我们需要为这些数据集开发基准查询。我们为两个数据集各开发了 10 个基准查询，如表 4.1 所示。这些基准查询经过精心设计，旨在探索相关数据集的广度和深度。此外，部分查询特意包含歧义关键词，以便我们考察 SAED 的上下文检测能力。为简洁起见，我们为每个基准查询提供了一个首字母缩略词（见表 4.1）。对于每个基准查询，我们最多收集排名前 20 的检索文档。随后，我们通过自动化脚本和人工用户评估检索文档的质量。

4.5.3评估指标

我们需要衡量搜索相关性指标，以了解

检索到的文档与用户查询的相关程度，以及它们在多大程度上符合其兴趣。为进行衡量，我们使用 Mariappan 等人 [102]提出的 TREC风格平均准确率（TSAP）评分。TSAP 无需了解整个数据集 [3]，便能以相对快速的方式提供定性评分。它基于精确率-召回率的概念，该概念常用于评估$_N$文本检索系统。TSAP 评分根据 $\sum r_i/N$计算，其中 $r_{i}{}_{i=0}$表示 i^{th} 检索文档的评分， N 表示截断数量（检索到的文档总数）。由于我们考虑 $N = 10$，因此将该评分指标称为 $TSAP@10$。

为确定检索文档 $\delta'_i \in \Delta'$ 的 r_i，我们进行了基于人工的评估。我们邀请了五名志愿学生评判每份检索文档的相关性。对于每个搜索查询，志愿者将每份检索文档标记为 `highly relevant`、 `partially relevant` 或 `irrelevant`。根据针对文档 i 提供的响应进行多数投票后， r_i 的值按如下方式确定：

- $r_i = 1/i$ 如果文档为 `highly relevant`
- $r_i = 1/2i$ 如果文档为 `partially relevant`
- $r_i = 0$ 如果文档为 `irrelevant`

我们报告 TSAP@10 分数，以展示每个基准查询的结果相关性。此外，我们还报告平均 TSAP 分数，以展示各数据集的整体相关性。由于我们设置每次搜索均检索前10篇文档，因此

最高的 *TSAP@10* 分数为 0.292 [102]。

除了 TSAP 分数之外，我们还测量平均 ***F-1*** 分数，以比较接入 SAED 的企业搜索解决方案与原始企业搜索解决方案（即，未接入 SAED 时）所提供的搜索质量。F-1 分数在精确率和召回率指标之间保持平衡，这对于主题分布不均匀的非结构化数据集很有用。

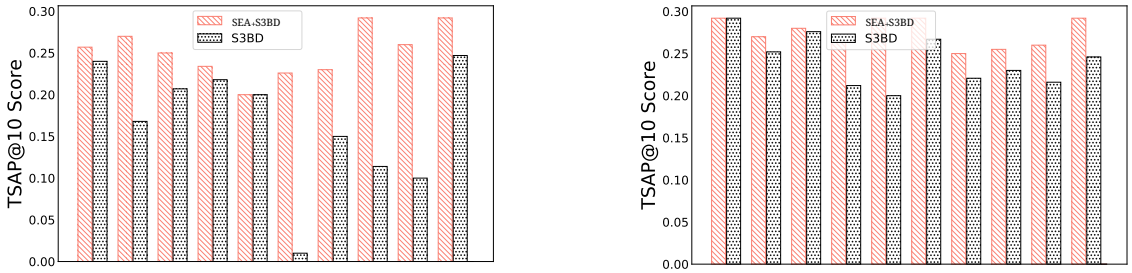
4.5.4评估搜索相关性

本实验旨在评估接入 SAED 的企业搜索系统的搜索相关性，并将其与原始（未经修改的）系统进行比较。为了评估个性化搜索，我们将（假定的）`technology` 设为两个数据集的用户兴趣。需要说明的是，在这一部分中，企业搜索解决方案（S3BD 和 AWS Kendra）设置为在明文数据集上运行。

S3BD 与 SAED+S3BD。 图 4.2a 显示了原始 S3BD 和 SAED+S3BD 在 RFC 和 BBC 数据集上的 TSAP@10 分数。两个子图的横轴均表示基准查询，纵轴表示基于 TSAP@10 分数的搜索相关性。

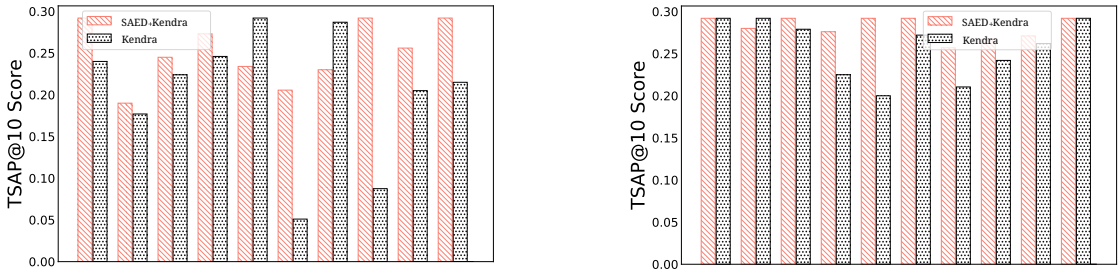
在图 4.2a 和图 4.2b 中，我们观察到，对于两个数据集中的所有查询，SAED+S3BD 的表现都优于 S3BD 系统。此外，我们还观察到，与 RFC 数据集相比，S3BD 在 BBC 数据集上产生的相关结果较少。这是因为，与 RFC 数据集不同，在 BBC 数据集中，基准查询的确切关键词在某些情况下并不存在。最糟糕的情况

图 4.2. 比较 SAED+S3BD 与 S3BD 系统的 TSAP@10 分数。横轴显示基准查询。



(a) BBC 数据集　　(b) RFC 数据集

图 4.3. 比较使用 SAED+Kendra 与 AWS Kendra 搜索基准查询时获得的 TSAP@10 分数。

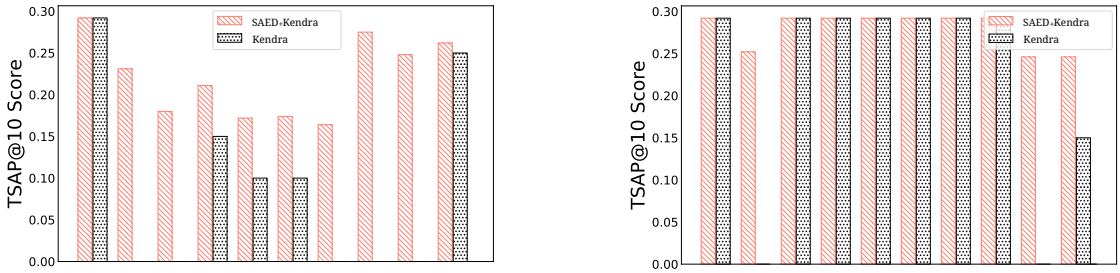


(a) BBC 数据集　　(b) RFC 数据集

发生在 S3BD 中的 `PCI` 查询上，因为其查询扩展过程无法捕捉完整的语义。相比之下，SAED+S3BD 能够处理数据集中不存在确切关键词的情况，因此，我们看到其相关性显著更高。

即使我们将 `PCI` 视为异常值并将其从分析中排除，在图 4.2a 中，我们仍然注意到 SAED+S3BD 的 TSAP@10 分数平均比 S3BD 高 41.2%。尽管对于 RFC 数据集而言，S3BD 与 SAED+S3BD 之间的差异并不那么显著（见图 4.2b），但我们仍然注意到 TSAP@10 分数提高了约 17%。这是因为 RFC 是一个领域特定数据集

图4.4。 比较加密域中 SAED+Kendra 与 AWS Kendra 系统获得的 TSAP@10 分数。



(a) 加密 BBC 数据集

(b) 加密 RFC 数据集

并且查询中的确切关键词可以在数据集中找到，因此，使用智能方法提取语义并不是获得相关结果的必要条件。根据这些结果，我们可以得出结论：对于文档中存在众多主题的通用数据集，SAED 尤其有效。

AWS Kendra 与 SAED+Kendra。 在图4.3a和图4.3b中，我们分别报告了 AWS Kendra 与 SAED+Kendra 在 BBC 数据集和 RFC 数据集上获得的 TSAP@10 分数。具体而言，在图 4.3a（BBC数据集）中，我们注意到 SAED+Kendra 的 TSAP@10 分数显著提升（平均提升 26.5%）。然而，与 SAED+S3BD 不同，SAED+Kendra 并未在所有查询上击败 Kendra。对于 `AWB`和 `CPM` 查询，Kendra 的表现优于 SAED+Kendra，原因在于 SAED 注入了额外的关键词，并将扩展查询集发送至 AWS Kendra。随后，Kendra 返回与查询及扩展关键词相关的文档。我们发现，SAED 的排序单元偶尔会优先选择包含扩展查询关键词的文档，而不是包含查询关键词的文档。

与 S3BD 实验类似，我们观察到，对于 RFC，Kendra 和 SAED+Kendra 得出的相关性不太显著。然而，根据图 4.3b，我们仍然获得了约 12% 的 TSAP@10 分数提升。

4.5.5隐私保护企业搜索的相关性

为了考察 SAED 在隐私保护企业搜索系统中的效率，我们使用加密的 BBC 和 RFC 数据集进行了实验。加密数据集被上传到云层，扩展查询也经过加密，并通过 Kendra 在云层上进行搜索。

对于 BBC 和 RFC 数据集，我们分别使用图 4.4a 和 4.4b 所示的 TSAP@10 分数。图 4.4a 表明，对于所有基准查询，SAED+Kendra 的表现都显著优于 Kendra。我们可以看到，对于加密数据集，Kendra 除了进行模式匹配并返回确切包含加密查询的文档外，别无他法。因此，搜索多个查询（例如， `PA`,`TCP`, `CPM` 等）不会检索到任何文档。我们注意到，在这两个系统中，搜索 `EC` 时的 TSAP@10 分数最高。这是因为 BBC 中包含确切短语`European commission`的文档数量较多。

图 4.4b 中报告的 RFC 数据集 TSAP@10 分数显示，与 BBC数据集相比有明显提升。我们观察到，十个查询中有七个在两个系统中的 TSAP@10 分数相同。Kendra 之所以能与 SAED+Kendra 竞争，是因为 RFC 中恰好包含这些基准查询。然而，对于 `HNC` 和 `FC`，精确的查询关键词

并未出现在数据集中，因此 Kendra 无法找到任何结果。

4.5.6相关性结果讨论

表4.2报告了采用纯文本和加密形式的数据集时，接入 SAED 的企业搜索系统及其原始版本的平均 *F-1* 分数和平均 *TSAP@10* 分数。从表中可以看出，无论采用哪种企业搜索系统，RFC 数据集的搜索相关性始终高于 BBC 数据集。

使用 SAED+Kendra 时，搜索相关性始终有所提升，平均 F-1 分数平均提高 23%，平均 TSAP@10 分数提高 21%。尽管原始 S3BD 的表现最差，但使用 SAED+S3BD 后，其平均 F-1 分数和平均 TSAP@10 分数分别平均提高 40% 和 32%。

表4.2。 比较接入 SAED 的企业搜索系统与其原始版本所获得的平均 F-1 分数和平均 TSAP@10 分数。最高分以粗体显示。

	BBC		RFC	
系统	**Mean F-1**	**Mean TSAP@10**	**Mean F-1**	**Mean TSAP@10**
S3BD	0.50	0.17	0.80	0.24
SAED+S3BD	0.82	0.25	0.92	**0.28**
Kendra	0.67	0.20	0.88	0.26
SAED+Kendra	**0.90**	**0.27**	**0.93**	**0.28**
Kendra（加密）	0.31	0.09	0.75	0.22
SAED+Kendra（加密）	0.73	0.22	0.90	0.27

在加密域中，我们注意到 SAED+Kendra 提供了

对 BBC数据集显著更高的搜索相关性（最高提升 130%）。由于给定搜索查询中的确切关键词并未出现在 BBC数据集的加密形式中，AWS Kendra 无法执行语义搜索，只能进行模式匹配，因此在该数据集上的表现不佳。另一方面，RFC 数据集的搜索相关性有所提升，因为均值 F-1 和均值 TSAP@10 分数至少提高了 20%。这是因为大多数查询都原样出现在数据集中，Kendra 仅依靠模式匹配就能检索出大多数相关文档。

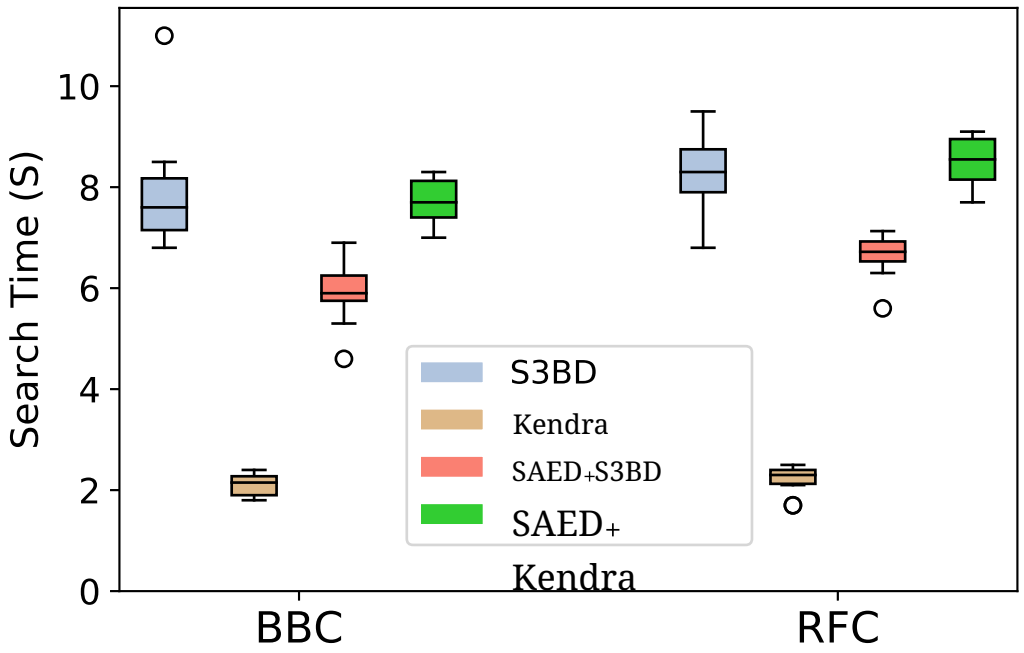
4.5.7评估搜索时间

图 4.5 展示了各数据集上实验查询所产生的总搜索时间。搜索时间按查询处理所需的耗时（例如，扩展、加权）与收到结果集前的周转时间之和计算。为消除计算系统中任何随机性的影响，我们对每组实验查询进行了 10 次搜索，并以箱线图的形式报告结果。图中表明，对于这两个数据集，S3BD 系统的搜索时间开销最高，这可能会影响大数据场景下的实时可搜索性。与原始（未修改的）S3BD 系统相比，SAED+S3BD 的查询处理时间开销更低。

另一方面，与 SAED+Kendra 相比，AWS Kendra 在两个数据集上造成的时间开销最低。与原始 Kendra 相比，SAED+Kendra 造成的时间开销约高出 4 倍。然而，在此前的一组实验中，我们发现 SAED+Kendra 实现了显著更高的

搜索相关性，尤其是在具有隐私约束的数据集上，对大多数查询都是如此。

图 4.5. S3BD、Kendra、SAED+S3BD 和 SAED+Kendra 系统之间的搜索时间比较。



4.6 总结

对于希望使用云服务的数据所有者而言，当下迫切需要一种上下文感知、个性化且保护隐私的企业搜索服务。为满足这一需求，我们将搜索智能和隐私保护与模式匹配方面分离开来。我们开发了 SAED，使其能够在边缘层实现隐私保护和智能搜索，并将大规模模式匹配留给云层。SAED 采用可插拔设计，可与任何企业搜索解决方案（例如，AWS Kendra 和 S3BD）配合使用，而无需对其做任何改动。在用户场所利用边缘计算可以保护用户隐私，并使 SAED 成为轻量级解决方案。SAED 借助基于循环神经网络的预测模型、WordNet 数据库和 Word2Vec，主动沿着恰当的上下文方向扩展搜索查询，并根据用户兴趣为扩展查询集分配权重。此外，即使数据以加密形式存储在云端，SAED 也支持语义搜索。在这种情况下，现有企业搜索解决方案只执行模式匹配，而不会了解底层数据。经人类用户验证的评估结果表明，对于纯文本数据集和加密的通用数据集，SAED 可分别将检索结果的相关性平均提高$\approx 24\%$和 $\approx 75\%$。

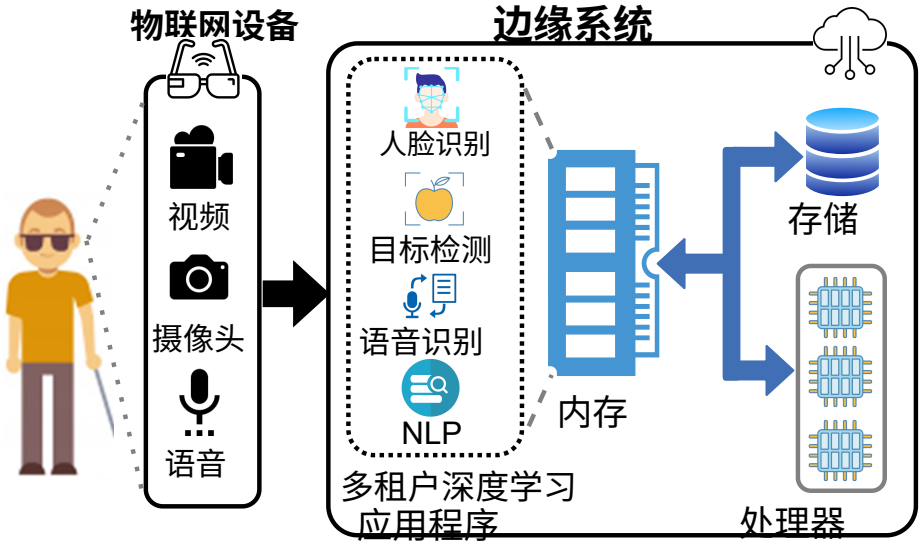
第5章：边缘延迟敏感型深度学习应用的多租户

5.1 概述

在上一章中，我们提出了一种企业搜索应用程序，即以可信应用形式实现的 SAED，用于在云端机密数据上执行安全搜索。SAED 应用程序横跨边缘到云连续体，由多个运行于边缘的微服务组成，用于实现搜索的智能处理方面（即，查询处理、个性化和排序）。我们的研究表明，在边缘运行多个微服务会消耗大量资源，尤其会耗尽边缘内存，导致服务可能被终止或无法执行。此前的研究对神经网络模型进行量化以减小其体积，但若缺乏模型管理，系统就无法真正发挥多租户处理的优势。Edge-MultiAI 利用神经网络模型压缩技术（例如模型量化），并为深度学习应用动态加载神经网络模型，从而促进边缘服务器上的多租户处理。我们将这一问题加以考量并扩大其研究范围，以寻求统一的解决方案。

由于智能物联网系统用途广泛，用户设备每天都会产生各种应用请求（即目标检测、人脸识别、自然语言处理和动作捕捉），这些请求会在低延迟约束下于边缘层执行。此类基于物联网的系统的一个典型应用场景是 SmartSight [19]，如图5.1所示，其旨在为盲人和视障人士提供环境感知。该系统基于智能眼镜（物联网

图5.1. SmartSight 的鸟瞰图。这是一个基于物联网的系统，可持续接收来自智能眼镜设备传感器的各种输入，并通过运行在边缘服务器上的多租户深度学习应用进行处理。



设备）和配套的边缘服务器（例如，智能手机）。智能眼镜

设备），通过传感器（例如摄像头和麦克风）持续捕获输入，并请求边缘服务器处理基于深度学习的应用程序，例如通过目标检测识别障碍物、通过人脸识别辨认熟悉的人，以及通过语音识别和自然语言处理理解并响应用户的指令。为了让 SmartSight 能够投入使用，边缘服务器必须持续执行多个（也称为多租户）深度学习应用程序，以低延迟和高准确率处理传入的请求。值得注意的是，尽管云数据中心可以缓解边缘设备固有的资源限制，但由于网络延迟开销和数据机密性 [14, 15, 16], ，在许多应用场景中，将延迟敏感型服务请求卸载到云端并非可行之举。

DL 应用程序在其内核中使用庞大的神经网络模型 to

表 5.1。 在三星 Galaxy S20+ astheedge 服务器上单独运行的热门神经网络模型的加载时间、推理时间和准确率。

神经网络模型	Bit 宽度	Size (MB)	加载时间（毫秒）	推理时间（毫秒）	准确率（%）
InceptionV3	FP32	105	650	100	78.50
	INT8	24	380	80	77.20
VGG16	FP32	528	820	52	71.30
	INT8	132	185	40	70.18
MobileNetV1	FP32	89	600	15	70.56
	INT8	23	192	8	65.70
MobileNetV2	FP32	26	110	10	72.08
	INT8	9	65	7.5	63.70
MobileNetV3	FP32	14	80.3	7.80	74.04
	INT8	8	47.45	6.21	71.32
MobileBERT	FP32	96	1100	62	81.23
	INT8	26	890	40	77.08

对从传感器接收的输入进行推理。神经网络模型必须保存在内存中，以实现低延迟（又称热启动 [31]）推理操作。

否则，由于神经网络模型通常非常庞大，以按需方式将其加载到内存中（也称为冷启动）会适得其反，并影响深度学习应用的延迟约束。由于边缘服务器的内存大小自然有限（例如，Jetson Nano 的内存为 4 GB [32]），在这些服务器上多租户执行深度学习应用会导致进程之间发生内存争用 [14, 33]。

因此，本研究面临的主要挑战是解决多租户深度学习应用之间的内存争用，同时不损害其延迟和准确率约束。

在深度学习的上下文中，有一些基于近似计算理念的技术，例如量化 [114], ，它们可以让模型

通过压缩神经网络模型变得适合边缘设备，从而减少推理时间并降低准确率。为了了解这类近似方法的影响，我们开展了一项初步实验，使用 Samsung Galaxy S $20+$ 作为边缘服务器，并选取了五种热门深度神经网络模型，即 InceptionV3、VGG16、MobileNetV1、MobileNetV2、MobileNetV3 和 MobileBERT，每种模型均采用两种量化（精确率）级别，即 FP32 和 INT8 位宽。在表 5.1 中，我们报告了各模型单独执行时的平均加载时间、推理时间和准确率。我们观察到：(A) 对于所有模型，加载时间都比推理时间长 8—$17\times$ 倍；(B) 加载高精度模型（FP32 位宽）所占用的内存比加载低精度模型（INT8 位宽）多 $\approx 3.5\times$ 倍；(C) 加载低精度模型会使推理准确率降低约 3—6%。这些结果表明，模型压缩在减少深度学习应用程序的内存占用方面具有巨大潜力。此外，模型加载时间始终远超推理时间 [115]。因此，我们的假设是：高效利用模型压缩和边缘内存，可以提升深度学习应用程序的多租户能力并缩短推理时间，同时不会显著损失其推理准确率。

我们建议为每个深度学习应用配备多个具有不同精度等级的神经网络模型。低精度模型的内存占用较小，因此可以让更多深度学习应用及其模型同时加载到内存中（即 热启动推理），从而降低服务延迟。然而，为了最大限度地提高多租户能力并实现热启动推理，加载精度过低（过度量化）的模型并不可行，因为这会降低

推理准确率，使多租户深度学习应用失去意义。相反，在内存受限的边缘系统上无限期加载高精度（大型）神经网络模型，会不必要地占用过多内存空间，不利于其他租户实现多租户和热启动推理。也就是说，其他租户会面临明显的速度下降（如表 5.1 所示），因为他们无法将自己的神经网络模型保留在内存中，必须从存储中加载模型（即 冷启动），才能执行推理操作。因此，多租户边缘系统的理想解决方案应能够从应用程序可用的模型集合（又称模型库）中动态加载合适的模型，既不中断其他应用程序的执行，也不会导致它们进行冷启动推理。

5.2 问题陈述

我们要探究的研究问题是：如何在不影响推理准确率的前提下，最大限度地增加边缘多租户深度学习应用的热启动推理次数？这个问题体现了两个目标之间的权衡：满足深度学习应用的延迟约束，同时保持其推理准确率。前者要求将深度学习应用的神经网络模型加载到内存中（即 热启动推理），而后者则要求在内存中保留高精度神经网络模型。

对于应用程序 $A_i \in A$，其模型库为 $M_i = \{m_i^k \mid 1 \leq k \leq q_i\}$，令$r_i(t)$ 表示在时间 t 针对 A_i 发出的推理请求，且其值为 1。另外，令 $m_i^* \subseteq M_i$ 为一个大小为 s_i^*、属于 A_i 且当前

已加载到内存中的神经网络模型。这意味着，对于当前没有任何神经网络模型驻留在内存中的应用程序 A_j，有 $m_j^* = \varnothing$ 和 $s_j^* = 0$。那么，$_nM^* = \bigcup m_i^*$表示当前已加载的神经网络模型集合，它们占用$_{i=1n}S^* = \sum s_i^*$ 的内存空间。对于 A_i，在时间 $t_{i=1}$ 到达的请求会发生冷启动事件，记作 $C_i(M^*,\ t)$，如公式 (5.1) 所示；当内存中没有适用于 A_i 的神经网络模型时（即 $M_i \cap M^* = \varnothing$），就会发生该事件。

$$C_i(M^*, t) = \begin{cases} r_i(t) & M_i \cap M^* = \varnothing \\ 0 & otherwise \end{cases} \tag{5.1}$$

假设使用 $m_i^* \in M_i$ 得到的推理准确率记为 χ_i^*。那么，根据公式 (5.2)，对于 n 多租户深度学习应用，我们可以将目标函数形式化地表述为：在最大化推理准确率的同时，最小化冷启动推理的总数。在这种情况下，可供神经网络模型使用的总内存大小（记为 S）是约束条件。

$$\begin{gathered} \min\left(\int_t^{\infty} \sum_{i=1}^{n} C_i(M^*, t)\ dt\right), \quad \max\left(\int_t^{\infty} \sum_{i=1}^{n} \chi_i^*(t)\ dt\right) \\ \text{subject to:} \qquad \forall t,\ \sum_{i=1}^{n} s_i^* \le S \end{gathered} \tag{5.2}$$

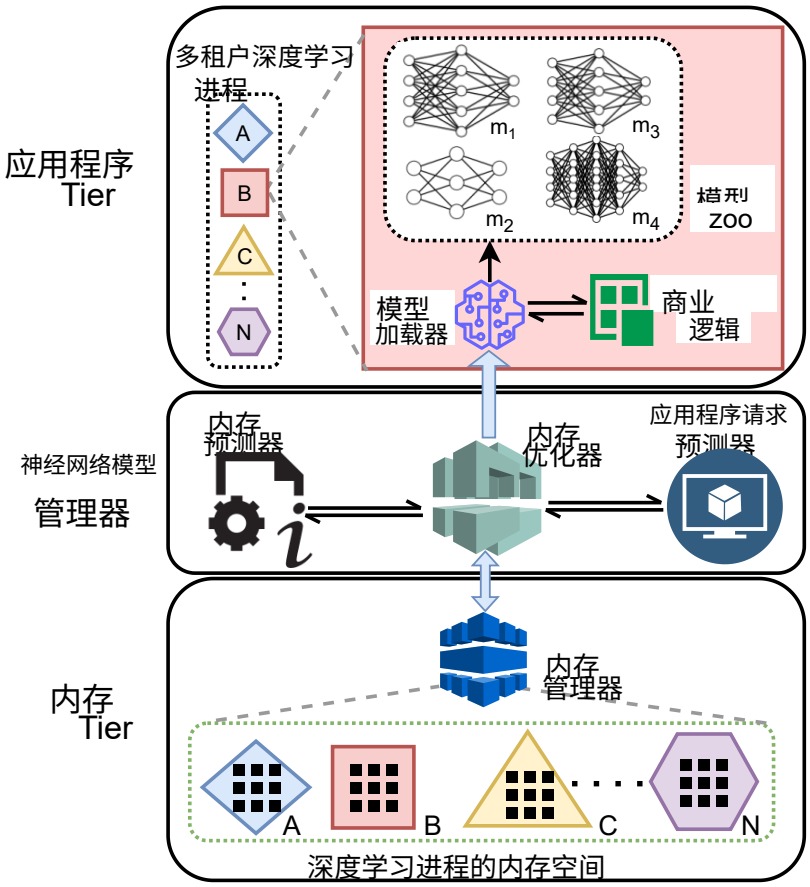
请注意，最优的神经网络模型管理决策并非贪心式的。也就是说，在给定时刻 t减少冷启动推理次数，并不一定能在整个应用程序生命周期内，以最高准确率实现冷启动总次数最少。换句话说，系统可能会在时刻 t 经历一次冷启动，以避免之后发生多次冷启动

。因此，公式 5.2 的目标函数包含从 t到 ∞ 的积分，以涵盖在 t 时所做决策对未来冷启动和准确率水平的影响。在目标函数中，应用程序 A_i 的神经网络模型仅从其模型库（M_i）中选择，因此，准确率（$\mu_i(t)$）和大小函数（s_i）都是离散函数。不言而喻，减少冷启动推理次数等同于增加热启动事件的次数 [116]。在本章其余部分，我们将交替使用这两种说法。

5.3 解决方案陈述与贡献

为了在有限的边缘内存上实现多租户，我们开发了一个名为 Edge-MultiAI 的框架。该框架利用每个深度学习应用的模型库，并可动态交换各应用的神经网络模型。为了在多租户深度学习应用中尽可能增加高推理准确率的热启动次数，我们采用的方法是：主动为预计将收到推理请求的应用加载高精度神经网络模型，同时为其他应用加载低精度模型。我们利用近期的内存使用信息，预测后续执行时的内存可用性，同时不打断其他活动应用程序。我们制定了模型管理启发式策略，根据预期的内存可用性和多租户深度学习应用的使用模式，在推理操作开始前为请求应用程序选择合适的神经网络模型，从而兼顾应用程序的延迟和推理准确率。

图5.2。 Edge-MultiAI 框架的架构概述，采用三层架构：应用程序、神经网络模型管理器和内存。



5.4 Edge-MultiAI 的架构概述与系统设计

图5.2展示了 Edge-MultiAI 的架构概述。该框架通过让应用程序仅交换其神经网络模型，而非整个应用程序，实现在资源受限的边缘系统上支持深度学习应用的多租户。框架由三层架构组成：（i）应用程序层，（ii）神经网络模型管理器，以及（iii）内存层级。

应用程序层。 来自已连接物联网设备的多模态输入会触发应用程序层中多租户深度学习应用的执行。模型库 为每个深度学习应用提供一个包含神经网络模型的存储库

这些模型具有不同的压缩等级（大小）和推理准确率（即各种精度等级）。模型加载器 负责从模型库中选择神经网络模型，并将其加载到边缘内存中。

神经网络模型管理器。 神经网络模型管理器由三个组件组成：（i）应用请求预测器、（ii）内存预测器和（iii）内存优化器。“应用请求预测器”收集各应用程序的历史请求，并训练一个轻量级（适用于边缘设备）的多对一普通循环神经网络（RNN）时间序列预测模型，类似于 [110], 中的模型，以定期预测各应用程序的推理请求到达情况。每当有请求到达时，“内存预测器”便负责根据整个边缘系统近期的内存分配情况，预测内存可用性。我们利用历史内存分配数据，训练另一个多对一普通循环神经网络时间序列预测模型来预测可用内存。

内存优化器与应用程序“请求预测器”和“内存预测器”交互，以获取：(A) 不同应用程序的请求到达时间及其模型库信息；以及 (B) 内存可用性信息。随后，内存优化器将收到的信息提供给神经网络模型管理策略，该策略会确定可加载的精确率尽可能高的神经网络模型，以服务深度学习应用的推理请求，同时尽可能减少对其他应用程序执行的影响（就预测准确率或延迟而言）。当到达的推理请求面临内存不足时，内存优化器会回收分配给其他应用程序神经网络模型的内存，

方法是加载低精度模型，或强制这些应用程序冷启动。 확보足够的内存后，内存优化器会通知“模型加载器”加载请求应用程序的适当神经网络模型。**内存层级**。该层级包含分配给应用程序的“内存空间”，以及一个“内存管理器”，用于跟踪当前已加载的模型、可用内存空间和应用程序的当前状态。内存管理器会将这些信息传达给神经网络模型管理器，以便高效地将其分配给到达的请求。

5.5 多租户应用的模型管理启发式方法

5.5.1 概述

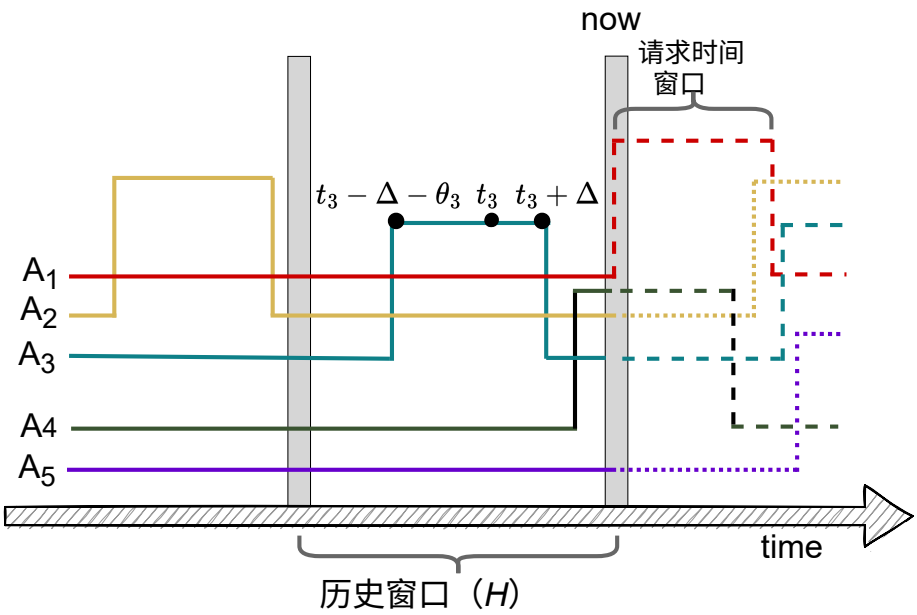
请记住，神经网络模型管理策略的目标是减少多租户深度学习应用在边缘服务器上的冷启动推理次数，并最大限度地提高推理准确率。为此，内存优化器力求延长已加载模型在边缘内存中的保留时间。然而，由于可用内存空间有限，无法将所有应用的最高精度神经网络模型都保留在内存中。为解决这一内存争用问题，应降低近期不太可能收到请求的应用所对应神经网络模型的内存保留优先级。此外，Edge-MultiAI 支持动态加载应用的神经网络模型。这意味着，在预测时间 t 为某个深度学习应用的推理请求时间后，可以指示 Edge-MultiAI 立即加载该应用的高精度神经网络模型

，然后再执行推理。同样，在内存不足时，对于预计在时间 t 不太可能收到请求的应用，可以指示 Edge-MultiAI 卸载其神经网络模型，或者更进一步，用精确率较低的模型替换这些模型。

然而，我们知道，请求到达本质上具有不确定性 [19] ，任何预测模型都无法准确预测应用程序的确切请求时间。为描述这种不确定性，我们在每个预测请求时间附近考虑一个请求时间窗口，记为 Δ。 Δ 的值通过对过去的请求预测进行性能分析，并计算所有应用程序的实际到达时间与预测到达时间之间的平均差值来获得。此外，将应用程序 A_i 所选的神经网络模型加载到内存中还需要一定的时间开销，记为 θ_i。总之，为了避免 A_i 在时间t进行推理时发生冷启动（如图 5.3 所示），必须在时间（$t_i - \Delta - \theta_i$）加载神经网络模型，并将其保留在内存中，直到时间（$t_i + \Delta$）。此外，针对应用程序在特定时间“没有请求”的预测也存在不确定性。也就是说，在时间 t，某个应用程序可能会收到推理请求，尽管预测结果认为该时刻不会有请求。为了使系统能够应对此类不确定性，并避免在这种情况下发生冷启动推理，理想的策略应为这些应用程序加载低精度神经网络模型。因此，即使推理请求未被预测到，也可以通过低精度模型以热启动方式处理，并满足延迟约束。

在本研究中，将神经网络模型保留在

图 5.3. 五个多租户应用（即 A_1 至 A_5）的推理请求示例场景。每个脉冲表示预期会收到推理请求的时间窗口。实线表示已经发生的事件，而“现在”之后的虚线表示请求预测。



内存中、且不处于其预测请求时间窗口内的应用程序集合称为极简集合，记为 A'。同样，将处于请求时间窗口内且为其加载高精度模型的应用程序集合称为极大化集合，记为 A^*。为解决内存争用，该策略可以从极简应用程序中回收内存，为极大化集合中的应用程序腾出所需的内存空间。也就是说，如果预测应用程序 A_i 将在时间 t_i进行推理，那么它会在时间$t_i - \Delta - \theta_i$成为 A^* 集合的成员，并在 $t_i + \Delta$之后成为 A' 集合的成员；因此，如果其他极繁应用需要内存空间，就可以将该应用程序的模型从内存中驱逐。神经网络模型驱逐仅允许从 A' 集合中进行，并且我们的目标是为该集合中的应用程序保留低精度模型。然而，

由于推理需求旺盛， A' 必须卸载其模型（即切换到冷启动），以释放空间来容纳最大化集合中应用程序的模型。在极端情况下，如果 A' 为空，或者从 A'回收的内存不足以为应用程序 A_i加载合适的模型，则会考虑 A_i 的下一个（更小的）模型，并重复上述步骤。最终，如果回收的内存空间不足以容纳 A_i的最低精确率模型，就会发生推理失败。

内存争用问题可以归约为经典的 0-1 背包优化问题 [117]：给定一组物品，每件物品都有权重和值，我们需要选择物品，使总价值最大化，同时总重量不超过限定值。该问题已知为 NP 完全，因此我们可以采用基于启发式的解决方案 [118]。接下来，我们将讨论四种神经网络模型管理（又称神经网络模型驱逐）策略，以管理多租户深度学习应用的内存，从而在不对推理准确率造成重大影响的情况下，最大化热启动推理的数量。

5.5.2策略 *1*：最大优先模型驱逐 *(LFE)*

在此策略中，为最大化进程的神经网络模型分配内存时，我们首先从集合 A' 中驱逐占用最大内存空间的神经网络模型，直到有足够的空间为 A^* 分配高精度神经网络模型。为此， A' 的成员会根据其当前加载的神经网络模型大小按降序排列。如果驱逐 $A,$ 的所有神经网络模型

仍无法释放足够的内存空间来为请求分配神经网络模型，则会尝试分配精确率较低的神经网络模型（大小更小）。该过程会持续进行，直到模型库中的某个模型能够加载到内存中；否则，边缘系统当时无法处理该请求。

5.5.3策略*2*：最佳适配模型驱逐（***BFE***）

LFE 的局限在于，无论实际内存需求是多少，它都会驱逐极简应用程序中最大的神经网络模型。这意味着采用 LFE 释放的内存空间可能超过实际需求。为解决这一问题，我们实现了 BFE 策略：按照极简集合中各应用程序的模型大小与实际内存需求之间的差值对应用程序进行排序，然后选择差值最小的神经网络模型进行驱逐。为了获得最高的推理准确率，首先根据极繁应用中精确率最高（也就是最大的）神经网络模型，计算其内存需求。然而，如果驱逐所有极简应用程序中的神经网络模型所释放的内存空间仍不足以容纳所需的神经网络模型，BFE 就会从请求应用程序的模型库中迭代选择下一个高精度模型。

5.5.4 策 略*3*：感知热启动的最佳适配模型驱逐（***WS-BFE***）

设 $A_i \in A^*$ 为当前位于最大化集合中的应用程序，$A_j \in A'$ 为当前位于极简集合中的应用程序。从技术上讲， A_i 的预测请求时间窗口可能与 A_j的预测请求时间窗口重叠。在这种情况下，LFE 和 BFE 策略可能会选择驱逐 A_j的神经网络模型

以支持 A_i 模型。这是因为这两种策略都着眼于过去，忽略了驱逐 A_j 的神经网络模型后，该应用程序可能很快就会收到请求这一事实。这样的驱逐决策会增加发生冷启动推理的可能性。为避免这种情况，我们开发了 WS-BFE，它会为 A' 中时间窗口与 A_i重叠的应用程序分配最低的驱逐优先级。

在早期实验中，我们发现，冷启动推理的另一个原因是请求到达具有不确定性。也就是说，某个极简应用程序可能会在意料之外的情况下收到请求。为尽可能降低这种情况下发生冷启动推理的可能性，我们实现了 WS-BFE，将被驱逐的神经网络模型替换为该应用程序中精确率最低（即最小）的神经网络模型。因此，如果极简应用收到未预测请求，就会有一个低精度模型可用于执行热启动推理。

5.5.5策略 *4*：智能感知热启动的最佳适配驱逐（*iWS-BFE*）

为使 WS-BFE 能够应对应用程序请求时间预测中的不确定性，我们引入贝叶斯理论对其进行增强，并提出一种称为 iWS-BFE 的新策略。该策略的灵感源自广泛采用的 LRU-K 缓存管理策略 [119]；该策略认为最近最少使用（即最近未被请求）的应用程序在不久的将来不太可能收到请求</style。同样，iWS-BFE 只将 A' 中近期未收到请求的成员作为驱逐候选项，并将其记为 E'。图 5.3 展示了 A_1—A_5 的预测请求时间示例。为了为 A_1 腾出内存，我们有 $A' = \{A_2, A_3, A_5\}$。由于 A_3 曾

在“历史窗口”（H）期间收到过请求，因此它在不久的将来很可能再次收到请求。因此，iWS-BFE 选择 $E' = \{A_2, A_5\}$ 进行驱逐。 H 的值根据所有应用程序的平均请求间隔时间确定。

除了考虑最近最少使用外，iWS-BFE 还利用 Edge-MultiAI 提供的请求预测。也就是说，它将近期未被请求、且预测未来最晚才会被请求的应用程序视为最适合驱逐的对象。然而，请求时间预测存在不确定性，系统可能会在当前请求窗口内收到来自 E' 成员的意外请求。为使 iWS-BFE 能够应对这种不确定性，我们计算意外请求的概率。对于应用程序 $A_j \in E'$，令 r_j 表示一次意外请求。那么，在当前请求窗口（即 $[t, t+\Delta]$）内发生 r_j 的概率定义为 $P(r_j|A_i \in A^*)$。可能会收到意外请求的应用程序并非驱逐的最佳选择。因此，在公式 5.3 中，为了计算驱逐 A_j 的适应度分数（记为 $Score(A_j)$），我们考虑 $1 - P(r_j|A_i \in A^*)$。为将 A_j 的预测请求时间纳入考量，我们计算其预测请求时间与当前时间（即 $t_j - t_i$）之间的距离。为了将数值限制在 $[0,1]$, 之间，我们根据所有 k 应用程序中最新预测距离对该距离进行归一化。

$$Score(A_j) = \frac{t_j - t_i}{\max\limits_{k \in E'}(t_k - t_i)} \cdot \left[1 - P(r_j|A_i \in A^*)\right] \tag{5.3}$$

iWS-BFE 策略的伪代码见算法 7。它首先根据历史窗口（ $\tau \subseteq A'$）内未收到请求的应用程序，构造一组初始驱逐候选项，称为 $\tau \subseteq A'$

。在此基础上，步骤 3 推导出一份驱逐候选项列表（记为 E），其中的元素与活动应用程序的请求窗口（A_i）不重叠。接着，在步骤 4 中，我们使用公式 5.3 计算每个 $E_k \in E$ 的适应度分数，然后根据适应度分数构建一个由 E 组成的最大堆树（步骤 5）。在步骤 6—10 中，该策略会迭代地取出适应度分数最高的应用程序（即最大堆的根节点，记为 w），并预测将其已加载模型替换为精确率最低的模型后可回收的内存量。一旦该策略找到足够的可回收内存，以便加载 A_i 的神经网络模型（记为 m_i），便会在步骤 13 执行所有神经网络模型替换决策，并在步骤 14 加载m_i。如果回收的内存不足，该策略会切换到 A_i 的下一个大小和准确率更低的神经网络模型（步骤 17）。最坏情况下，如果连 A_i 最小的神经网络模型也无法装入内存，则推理请求失败（步骤 17） [120]。

算法 7： iWS-BFE 神经网络模型驱逐策略的伪代码

函数 iWS-BFE(A' ，A^*，A_i，H)
  $\tau \leftarrow$ 选择 $\forall A'_j \in A'$ 在 H期间未被请求的
  $E \leftarrow$ 确定 $\forall A'_j \in \tau$ 与 A_i的请求窗口不重叠
  $\forall E_k \in E$ 使用公式 5.3 计算适应度分数
  根据适应度分数为 E 构建最大堆树
  当 $size(m_i) >$ 可用内存 **时**
    $w \leftarrow$ 提取最大堆树的根节点
    如果 $w = \emptyset$ **那么** 跳出循环
    通过将 w 的模型替换为
     精确率最低的模型
    将回收的内存量加到可用内存中
  end
  如果 $size$（m_i）$\leq$ 可用内存，**则**
    执行神经网络模型替换决策
    回收剩余内存以加载 m_i
  end
  else
    如果没有剩余模型可供检查，**则**推理请求失败
    使用下一个（更小的）模型重复步骤 6—10
  end
结束

表 5.2. 实验中采用的具有不同精度变体的应用专用模型。

应用程序	神经网络模型	Bit 宽度	Size (MB)	准确率 (%)
人脸识别	VGG-Face	FP32	535.1	90.2
		FP16	378.8	82.5
		INT8	144.2	71.8
图像分类	VIT-base-patch16	FP32	346.4	94.5
		FP16	242.2	81.3
		INT8	106.7	72.2
语音识别	S2T-librisspeech	FP32	285.2	89.7
		FP16	228.0	77.2
		INT8	78.4	68.0
句子预测	Paraphrase-Mini LM-L12-v2	FP32	471.3	88.2
		FP16	377.6	81.7
		INT8	98.9	76.2
文本分类	Roberta-base	FP32	499.0	91.1
		FP16	392.2	82.4
		INT8	132.3	76.6

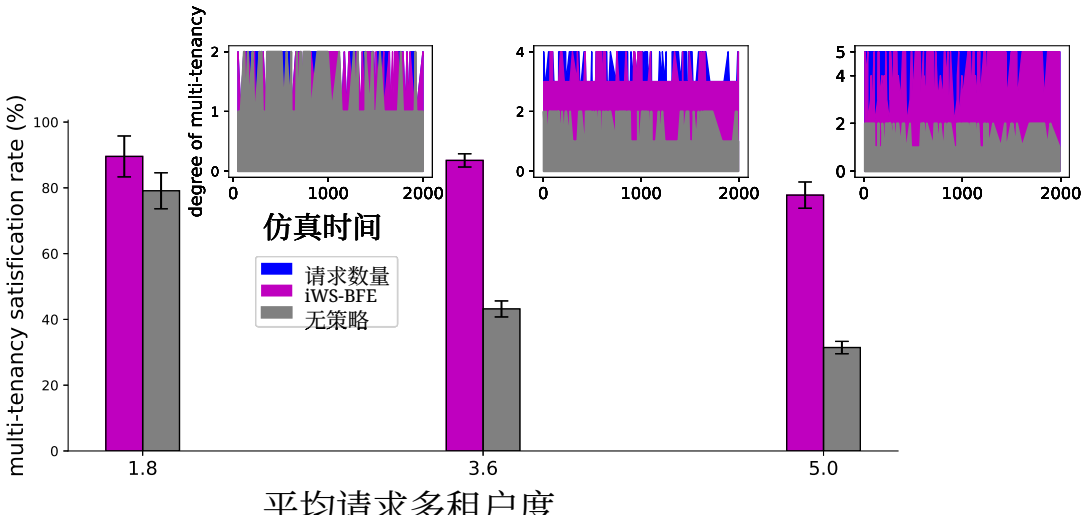
5.6 性能评估

5.6.1实验设置与评估指标为评估 Edge-MultiAI 及其神经网络模型驱逐策略的有效性，我们对五种不同的深度学习应用进行了基准测试，即人脸识别、语音识别、图像分类、下一句预测和文本分类，并记录了它们的实际特征，包括模型大小和推理准确率（如表 5.2 所示）。我们开发了 E2C 模拟器，用于对具有不同特征和配置的基于物联网的系统进行建模；该模拟器可通过我们的 GitHub 页面[a]公开访问。模拟器已实现所有神经网络模型驱逐策略，用户可以快速部署并检查其中任意一种策略。

该模拟器还使我们能够生成包含以下内容的工作负载轨迹

[a]GE2C 模拟器的 GitHub 页面：https://github.com/hpcclab/E2C-Sim.git

图 5.4。 Edge-MultiAI 及其 iWS-BFE驱逐策略对满足所请求多租户的影响。大图通过增加横轴上请求的多租户均值，并在纵轴上显示已满足请求的百分比，呈现汇总分析结果。对于每种情况，小图更细致地表示在仿真时间内发出并完成的并发请求数量。



仿真期间每个应用程序的请求到达时间。我们将实际工作负载配置为包含五个应用程序数量相等的请求，并且在该工作负载中，每个应用程序的请求到达间隔时间呈指数分布。为研究推理请求预测中存在的不确定性，在评估中，我们生成两组工作负载：一组包含多租户应用的预测到达时间，另一组包含应用程序的实际到达时间。实际工作负载中的请求到达分布与预测工作负载中的请求分布有所不同。两者之间的偏差程度根据 Kullback-Leibler (KL) [121] 散度进行衡量。我们将在接下来各小节的实验中探讨这一偏差的影响。

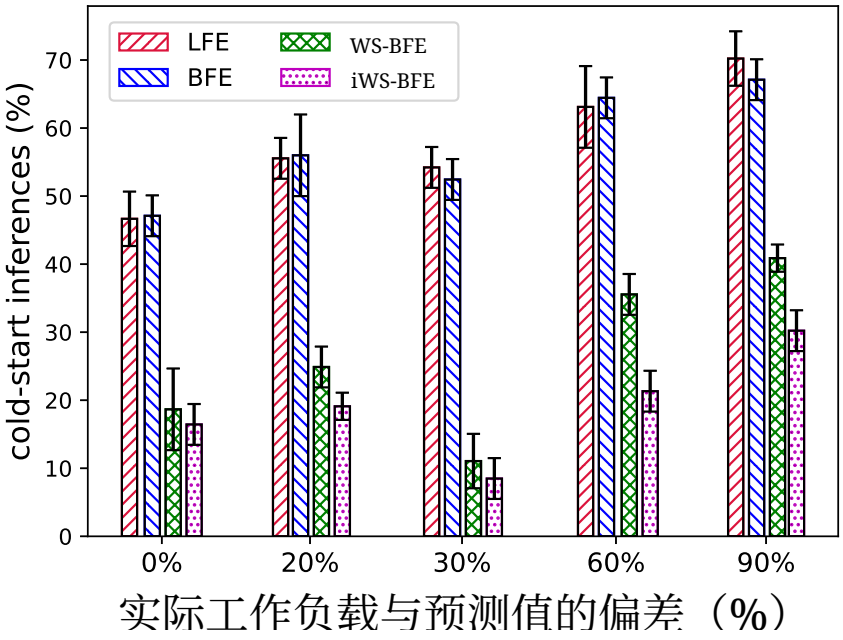
我们的评估指标包括：(A) 不同请求到达强度下的多租户程度；(B) 推理延迟；(C) 推理准确率；以及 (D) 用于衡量不同驱逐策略对请求预测中不确定性的容忍度的鲁棒性指标。

5.6.2 *Edge-MultiAI* 对多租户程度的影响

本实验旨在考察 Edge-MultiAI 满足边缘服务器传入请求的效能。为此，如图 5.4 所示，我们通过增加并发请求的均值来提高工作负载强度，并在每种情况下测量多租户满意率，即仿真期间热启动推理占传入请求总数的百分比。我们考察了两种情况：(A) 不采用任何促进多租户的解决方案（称为无策略）；以及 (B) 启用 Edge-MultiAI 及其 iWS-BFE 策略。实验重复进行 10 次，并报告每个数据点的平均满意率和 95%置信区间。

实验表明，采用 Edge-MultiAI 及其 iWS-BFE 所实现的多租户度显著高于未采用 Edge-MultiAI 的情况。较小的图表显示，这一优势在整个仿真时间内始终存在。我们还注意到，采用 Edge-MultiAI 对较高多租户度的影响更为显著。具体而言，当多租户度均值为 5，且平均请求多租户度大于 2 时，采用 Edge-MultiAI 及其 iWS-BFE 策略相比无策略可实现高出 $\approx$130% 的满意率。这

图5.5。 测量由所提出的驱逐策略导致的多租户应用程序冷启动推理所占的百分比。横轴表示预测的推理请求时间与实际推理请求时间之间的偏差。



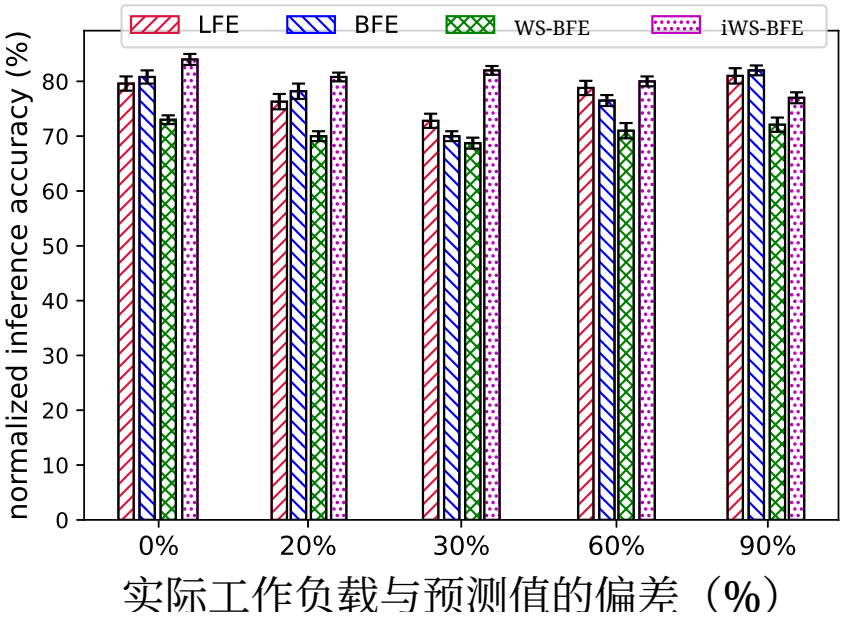
项实验验证了 Edge-MultiAI 和神经网络模型管理在促进深度学习应用多租户方面的有效性。

5.6.3驱逐策略对冷启动推理的影响 e

本实验旨在分析不同神经网络模型驱逐策略对冷启动推理数量的影响。为此，我们测量采用不同驱逐策略所导致的冷启动推理百分比，尤其是在请求预测与实际请求之间的偏差发生变化时的情况。

图5.5所示结果表明，LFE 和 BFE 的表现不佳，会导致大量冷启动推理；而 WS-BFE 和 iWS-BFE 至少能减少 65% 的冷启动推理。这是因为，在

图 5.6。 测量采用不同驱逐策略后应用程序的归一化推理准确率。



LFE 和 BFE 中，淘汰神经网络模型后，如果出现未预测请求，相应的应用程序就会经历一次冷启动推理。相比之下，在 WS-BFE 和 iWS-BFE 中，被淘汰的模型会由低精度模型替代，因此，对相应应用程序的未预测调用不会导致冷启动推理。值得注意的是，无论采用哪种策略，随着预测请求时间与实际请求时间之间的偏差增大，冷启动推理的比例都会上升。不过，即使偏差达到 90%，iWS-BFE 的表现仍显著优于其他策略。平均而言，与 LFE 和 BFE 相比，它的冷启动推理减少了 102%，比 WS-BFE 减少了 40%。

5.6.4驱逐策略对推理准确率的影响

在本实验中，我们分析采用不同模型驱逐策略所导致的平均推理准确率。由于准确率在不同应用程序之间差异很大

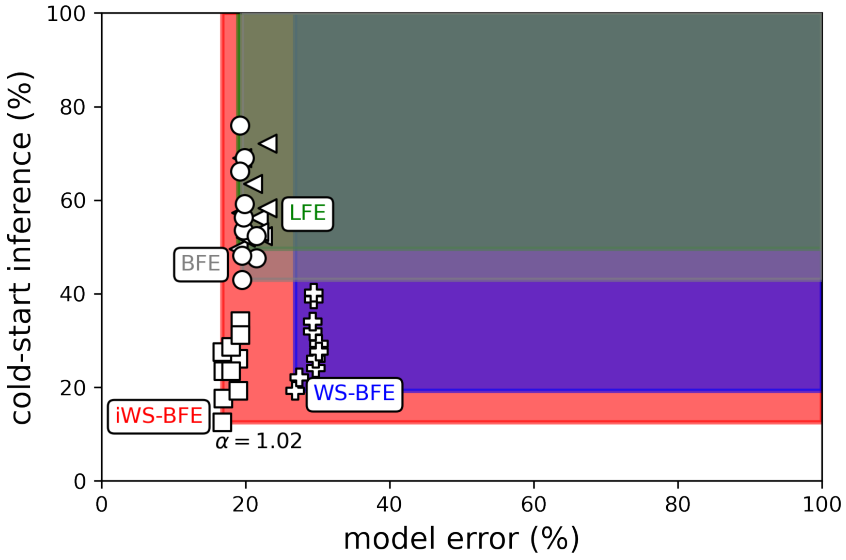
，我们对准确率值进行最小-最大归一化。此外，对于冷启动推理，在准确率测量中，我们考虑神经网络模型加载到内存后的准确率。

图 5.6 展示了在预测请求时间与实际请求时间之间的偏差发生变化时，采用不同神经网络模型驱逐策略所获得的归一化平均推理准确率。由图可知，LFE 和 BFE 策略优于 WS-BFE。这是因为这两种策略不会将低精度模型保留在内存中。因此，它们的推理请求要么导致冷启动（这在前一项实验中已有探讨），要么加载可提供高推理准确率的高精度模型。不过，我们观察到，在大多数情况下，iWS-BFE 的表现优于 LFE 和 BFE，偏差为 90% 时除外。iWS-BFE 推理准确率更高的原因在于，它会根据未来调用概率，智能地筛选冷启动候选对象。这些结果表明，评分（见公式 5.3）对于高效筛选冷启动候选对象十分重要。值得注意的是，LFE 和 BFE 在偏差为 90% 时推理准确率较高，但代价是冷启动推理次数显著增加，这会损害基于物联网的系统的“可用性”。

5.6.5神经网络模型驱逐策略的双目标分析

请注意，资源受限的边缘系统中多租户应用的神经网络模型管理是一个双目标优化问题，旨在最大限度地减少冷启动推理的次数，并最大限度地提高推理

图 5.7。不同模型选择策略的双目标分析。



准确率。然而，这两个目标通常相互冲突，不存在能够同时满足这两个目标的单一最优解。相反，可能存在一系列优于其他解的解。为了分析所研究的策略中哪些优于其他策略，我们在图 5.7 中绘制了不同策略和Δ 值对应的冷启动推理百分比与模型误差（定义为 100-准确率）之间的关系。设 D 和 σ 分别为预测请求时间与实际请求时间之差的残差均值和标准差。那么，通过改变$0 \leq \alpha \leq 2$的值，$\Delta = D \pm \alpha \cdot \sigma$ 会发生变化。本实验中，实际工作负载与预测工作负载之间的偏差为 30%。

对于每种策略，彩色区域表示受该策略支配的冷启动推理和模型误差率。理想的策略应接近图的原点（即实现零冷启动和零模型误差）。在图 5.7 中，我们观察到 Edge-MultiAI 支配了其他策略，并形成帕累托前沿，

尤其是在 $\alpha = 1.02$ 的情况下。我们可以得出结论，iWS-BFE 策略能够通过减少冷启动推理次数并提供更高的推理准确率，显著提升系统的可用性。

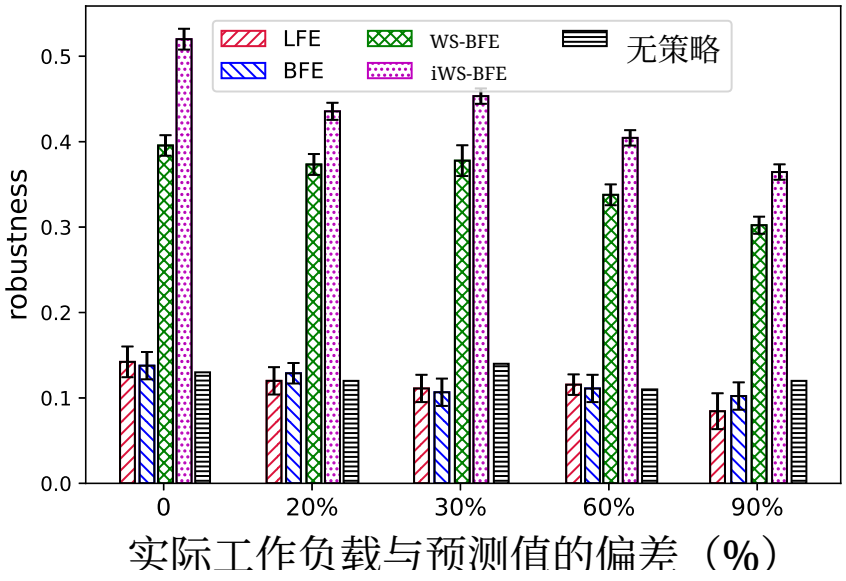
5.6.6分析对不确定性的鲁棒性

本实验旨在研究 Edge-MultiAI 的驱逐策略如何使基于物联网的系统能够抵御预测的应用请求预测器与实际情况之间存在的不确定性。我们定义了鲁棒性指标，如公式 5.4 所示，该指标综合考量热启动推理（记为 ϖ_i）占请求总数（记为 γ_i）的比例，以及仿真周期内各应用程序的平均预测准确率（ ψ_i ）。 i

$$R = \frac{1}{n} \cdot \sum_{i=1}^{n} \left[\frac{\varpi_i}{\gamma_i} \cdot \psi_i \right] \tag{5.4}$$

图 5.8 展示了采用所提出的策略和无策略（即基线）时，针对推理请求预测不确定性所达到的鲁棒性评分。我们观察到，与未部署 Edge-MultiAI（无策略）的情况相比，部署 Edge-MultiAI 并采用任何策略都能提升鲁棒性。我们还注意到，随着偏差增大，推理失败率和冷启动率随之上升，鲁棒性值也持续下降。我们观察到，WS-BFE 和 iWS-BFE 比 LFE 和 BFE 更能抵御偏差。这是因为 LFE 和 BFE 在驱逐时不会用低精度模型替换其神经网络模型，从而导致应用程序出现冷启动推理。

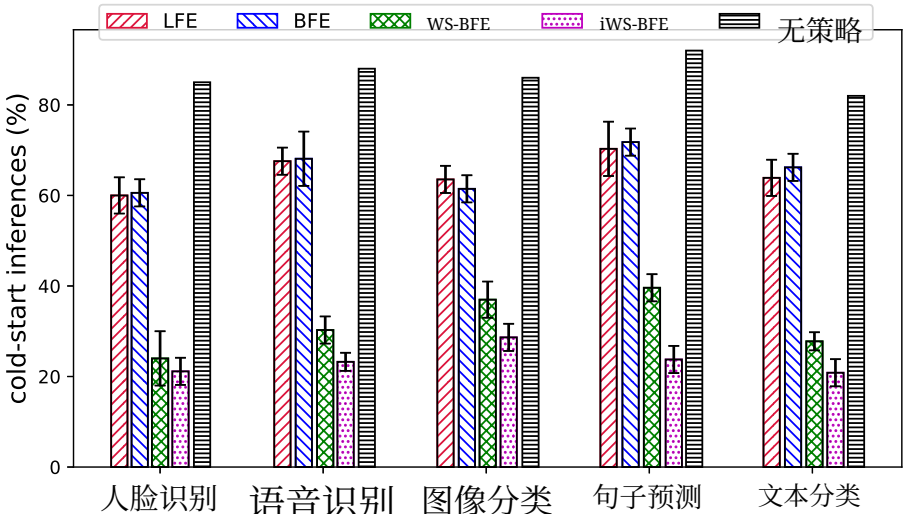
图 5.8。 系统对推理请求预测不确定性的鲁棒性。



5.6.7神经网络模型驱逐策略的公平性评估

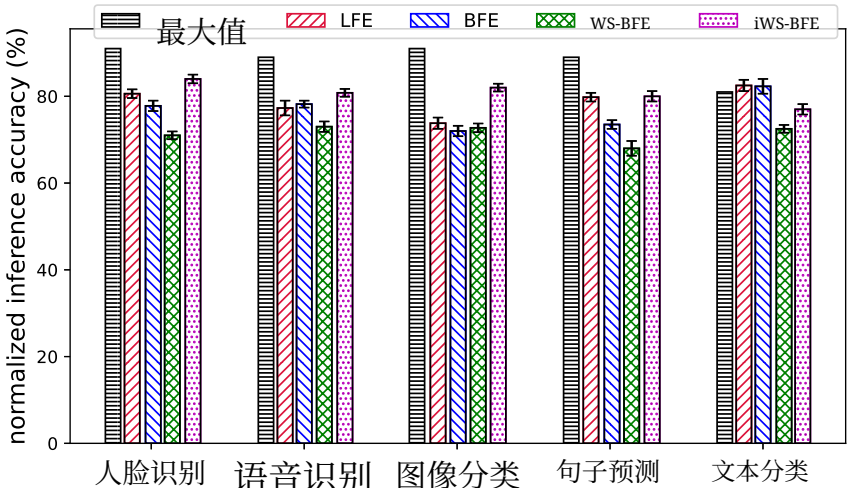
在本实验中，我们旨在考察此前实验中 Edge-MultiAI 及其策略所取得的成果，是否公平地惠及所有应用程序，还是某些应用程序比其他应用程序受益更多。为此，我们分析了不同深度学习应用程序的冷启动推理分布和准确率。受检深度学习应用程序的名称及其神经网络模型特征列于表 5.2。图 5.9 和图 5.10 分别展示了使用各种神经网络模型驱逐策略时，各应用程序的冷启动推理百分比和推理准确率。值得注意的是，在图 5.9 中，“无策略”表示未部署 Edge-MultiAI 的情况；在图 5.10 中，“最大值”作为基准，表示每个应用程序使用精确率最高的神经网络模型。图 5.9 显示，

图5.9. 采用不同神经网络模型驱逐策略与无策略时冷启动推理所占的百分比。



WS-BFE 和 iWS-BFE 在所有应用程序中的表现都显著优于其他策略。图5.10 显示，尤其是对于 iWS-BFE，这种优势并非以降低应用程序的推理准确率为代价。更重要的是，从两幅图中可以看出，对于每种策略，不同应用程序的冷启动推理百分比和准确率都没有显著波动。这表明这些策略不会偏向任何特定的深度学习应用。具体而言，冷启动推理比例和准确率在不同应用程序之间分布相当均衡。

图5.10。 不同策略下获得的推理准确率。“最大值”是基准，展示了每种应用程序中最高精度模型的准确率。



5.7 总结

智能物联网系统通常需要持续执行多个对延迟敏感的深度学习应用。边缘服务器是此类基于物联网的系统的基石，但其资源有限，妨碍了多个（多租户）深度学习应用的持续执行。本研究旨在提高此类应用的多租户程度，同时不影响其延迟和准确率目标。

我们开发了一种名为 Edge-MultiAI 的框架，通过仅交换深度学习应用的神经网络模型来促进多租户。该框架还配备了模型管理策略，尤其是 iWS-BFE，用于选择合适的模型进行驱逐并加载至边缘内存，从而在不显著降低应用程序推理准确率的前提下，最大限度地提高热启动推理的比例。评估结果表明，Edge-MultiAI 可将多租户程度提高 $2\times$，而 iWS-BFE 可使热启动推理增加 60%。结果还展示了不同策略在面对推理请求预测的不确定性时的稳健性。最后，实验表明，这些策略在决策时不会偏向某个特定应用程序。

第6章：结论与未来研究方向

本章总结了本学位论文的研究内容和主要发现。此外，本研究过程中浮现但未在本学位论文中涉及的研究主题也将在本章提出并讨论。该领域的其他研究人员可以进一步探索这些未来可能的研究方向。

6.1 讨论

在本学位论文中，我们的主要目标是通过在跨越边缘到云连续体的执行过程中维护数据完整性和机密性，实现边缘到云连续体中的机密计算。我们提供了三个可信应用，可在机密数据上执行安全聚类和语义搜索，而不会向任何场外层级泄露任何有意义的信息。此外，针对深度学习应用的模型管理，我们开发了一个框架，通过仅交换其神经网络模型，有效支持深度学习应用的多租户。

在第3章中，我们针对静态数据集（ClustCrypt 和 S-ClusPr）以及动态非结构化加密大数据集（SD-ClusPr 和 FD-ClusPr）的基于主题的聚类提出了解决方案。这些方案能够在可行的时间复杂度内估算数据集的簇数。为此，它们利用词元共现来衡量每个词元倾向于与其他词元聚集或分离的程度，并据此估算簇数。接下来，我们提出一种概率方法，以确定每个簇的中心

并将加密词元分发到主题相关性最高的簇。实验评估表明，对于静态数据集，S-ClusPr 平均可将聚类一致性提高 65%。同样，对于半动态和动态数据集，SD-ClusPr 和 FD-ClusPr 可将一致性提高 55%。通过在安全语义搜索系统的上下文中整合 ClustCrypt 和 ClusPr，我们发现，更连贯、更准确的基于主题的聚类能够提高搜索结果的相关性。

在第4章中，我们提出了一种开源通用可插拔模块 SAED，可集成到现有搜索服务中（例如，AWS Kendra、S3BD），无需对这些服务进行任何改动，即可实现上下文感知、个性化的安全搜索。SAED 可以搜索纯文本数据，也可以搜索在外包到云（即，AWS S3）之前使用客户端加密进行加密的数据。经人类用户验证，实验评估表明，对于纯文本数据集和加密数据集，SAED 可分别将检索结果的相关性平均提升 $\approx 24\%$ 和 $\approx 75\%$。

我们在第4章提出的解决方案需要在可信边缘层上持续并同时维护多个用于处理机密用户数据的深度学习模型。鉴于边缘层的内存限制，这项任务颇具挑战。此外，出于用户隐私考虑，此类机器学习模型无法外包至云。因此，在第5章中，我们提出了一个名为 Edge-MultiAI 的框架。该框架基于近似计算理念运行，并将深度学习应用的神经网络模型引入边缘内存，从而使多租户程度

和热启动次数实现最大化
。Edge-MultiAI 利用神经网络模型压缩技术（例如模型量化），并动态加载用于深度学习应用的神经网络模型，以推动边缘服务器上的多租户部署。我们还为 Edge-MultiAI 设计了一种名为 $iWS\text{-}BFE$ 的模型管理启发式方法，该方法基于贝叶斯理论预测多租户应用的推理请求，并据此选择要加载的合适神经网络模型，从而增加热启动推理的数量。我们在各种配置下评估了 Edge-MultiAI 的效能和鲁棒性。评估结果表明，Edge-MultiAI 可将多租户程度提高 $2\times$，而 iWS-BFE 可使热启动推理增加 60%。结果还表明，面对推理请求预测中的不确定性，不同策略都具有鲁棒性。最后，实验表明，这些策略在决策时不会偏向某个特定应用程序。

6.2 未来研究方向

在探索边缘到云连续体中的人工智能驱动的机密计算范式时，根据我们的发现，还有若干方面可以进一步拓展，但本学位论文未涉及这些内容。

6.2.1非结构化数据的层次聚类

我们可以采用主动学习，以实现对主题相似的词元进行自动层次聚类 [122, 123, 124]。主动学习范式的灵感来自这样的情形：收集大量未标注数据非常容易即，从互联网下载的图片和视频、

通过麦克风录音获得的语音信号等），但获取这些数据的标签却很困难或成本高昂。

我们可以利用解码后的词元的含义及其在现有簇中的分布。借助这些信息，我们引入维基百科知识，在整个机密数据集中构建词元之间的层级关系。之后，对于新的词元，我们会衡量该词元与各主题表示之间的相关性，以传播层级关系。

6.2.2基于加密簇构建分类器

聚类是一种经典的无监督学习方法，用于对大量未标注数据进行分组。我们可以在这些簇上采用主动学习 来构建分类器，从而有望增加非结构化数据范式在可信计算中的应用场景。主动学习可以利用已有知识，在查询簇时衡量其与新词元之间的相关性，从而形成分类器的决策边界。与传统的监督学习相比，所得分类器的成本显著更低 [122]。

6.2.3 在机密搜索中引入弹性当前

SAED 框架的实现需要依赖一台已连接的边缘服务器来辅助搜索。一种思路是在框架中加入灵活性，从而减轻边缘通信负担，同时将性能下降控制在可接受范围内。例如，当用户处于移动状态且无法访问边缘时，SAED 应精简至仅保留最低限度的搜索智能，反之亦然。

6.2.4模型管理方案中的增加能耗

在 Edge-MultiAI 中，资源受限的边缘系统中多租户应用的神经网络模型管理属于一个双目标优化问题，旨在尽可能减少冷启动推理次数并最大限度地提高推理准确率。由于边缘服务器的能耗有限，且通常依靠电池供电，因此在边缘服务器上分配作业时，考虑能耗至关重要。否则，电池耗尽可能导致系统突然关闭，进而造成执行失败。随后，我们可以将能耗作为第三个目标纳入该问题。这样，我们便将问题设定为：在内存大小和能耗预算的约束下，最大化热启动次数和总准确率。

6.2.5面向延迟容忍型应用程序的云端卸载

我们在 Edge-MultiAI 中只考虑低时延应用程序。我们还可以在框架中为云层添加卸载选项。这样，系统就可以决定是在本地处理任务（可能会遇到几次冷启动）更有利（就延迟和能耗而言），还是将其卸载到云服务器。

参考文献

[1] S. Zobaed、S. Ahmad、R. Gottumukkala 和 M. A. Salehi，“Clustcrypt：云中非结构化大数据的隐私保护聚类”，载于第*21*届高性能计算与通信国际会议（*HPCC*）论文集，2019年，第609–616页。[2] B. Seth、S. Dalal、V. Jaglan、D.-N. Le、S. Mohan 和 G. Srivastava，“整合加密技术以实现云中的安全数据存储”，《新兴电信技术汇刊》，第33卷，第4期，文章编号e4108，2022年。[3] W. Jason 和 M. A. Salehi，“S3BD：对加密的 bi 进行安全语义搜索

g

云中的数据，”《并发与计算：实践与经验期刊》*(CCPE)*，第 28 卷，第 11 期，2018年12月。[4] W. Sun、W. Lou、Y. T. Hou 和 H. Li，“云计算中加密数据的隐私保护关键词搜索”，载于安全云计算。施普林格，2014年，第 189–212 页。[5] Y. Zhu 和 X. Li，“点对点网络中具有本地同步功能的隐私保护 K 均值聚类”，《点对点网络与应用期刊》，第 13 卷，第 6 期，2020年。[6] J. Al-Muhtadi、B. Shahzad、K.Saleem、W. Jameel 和 M. A. Orgun，“多云环境中社交集成式移动医疗应用程序的网络安全与隐私问题”，《健康信息学期刊》，第 25 卷，第 2 期，第 315–329 页，2019年5月。[7] “21世纪最大的 15 起数据泄露事件”，www.csoonline.com/article/2130877/data-breach/the-biggest-data-breaches-of-the-21st-century.html，2021年11月。[8] S. Zobaed 和 M. A. Salehi，“云中的大数据”，载于《大数据百科全书》，L. A. Schintler 和 C. L. McNeely 编。施普林格，2018年。[9] “每一个雅虎账户都遭到黑客入侵”，www.money.cnn.com/2017/10/03/technology/business/yahoo-breach-3-billion-accounts/index.html，访问日期

d

2020年5月。[10] “云端数据泄露：Verizon合作伙伴如何暴露数百万客户账户”，www.upguard.com/breaches/verizon-cloud-leak，访问于2020年4月。[11] C. Shepherd、G. Arfaoui、I. Gurulian、R. P. Lee、K. Markantonakis、R. N. Akram、D. Sauveron 和 E. Conchon，《安全可信执行：过去、现在与未来——物联网

和信息物理系统背景下的批判性综述》，发表于*2016 IEEE Trustcom/BigDataSE/ISPA*，2016年，第168–177页。[12] Z. Ning、J. Liao、F. Zhang 和 W. Shi，《异构边缘平台上可信执行环境的初步研究》，发表于边缘计算研讨会（*SEC*）论文集，2018年，第421–426页。[13] “机密计算——数据安全的新前沿”，https://confidentialcomputing.io/wp-content/uploads/sites/85/2021/10/Everest Group - Confidential Computing - The Next Frontier in Data Security - 2021-10-19.pdf，访问于2022年7月5日。[14] S. Deng、H. Zhao、W. Fang、J. Yin、S. Dustdar 和 A. Y. Zomaya，《边缘智能：边缘计算与人工智能的交汇》，*IEEE*物联网期刊，第7卷，第8期，第7457–7469页，2020年。[15] S. M. Zobaed、M. A. Salehi 和 R. Buyya，《SAED：面向云端隐私保护企业搜索的基于边缘的智能》，发表于*2021*年*IEEE/ACM*第*21*届集群、云和互联网计算国际研讨会（*CCGrid*）。IEEE，2021年，第366–375页。[16] R. F. Hussain、A. Pakravan 和 M. A. Salehi，《分析智能工业4.0应用在云计算系统上的性能》，发表于第*22*届*IEEE*高性能计算与通信国际会议（*HPCC*）论文集，2020年，第11–18页。[17] S. Ahmad、S. Zobaed、R. Gottumukkala 和 M. A. Salehi，《面向云端加密大数据的以用户为中心的安全搜索的边缘计算》，发表于第*21*届高性能计算与通信国际会议（*HPCC*）论文集，2019年8月，第662–669页。[18] F. Almalki、S. H. Alsamhi、R. Sahal、J. Hassan、A. Hawbani、N. Rajput、A. Saif、J. Morgan、J. Breslin 等，《面向环保和可持续智慧城市的绿色物联网：未来方向与机遇》，《移动网络与应用》，第1–25页，2021年。[19] A. Mokhtari、M. A. Hossen、P. Jamshidi 和 M. A. Salehi，《FELARE：异构边缘系统上机器学习应用的公平调度》，发表于云计算国际会议论文集，2022年7月。[20] “使用客户端加密保护数据”，https://docs.aws.amazon.com/ AmazonS3/latest/userguide/UsingClientSideEncryption.html，访问于2022年11月。[21] H. Pham、J. Woodworth 和 M. Amini Salehi，《云端加密数据安全搜索综述》，《并发与计算：实践与经验》，第31卷，第17期，文章编号e5284，2019年。

[22] K. Zhang、J. Long、X. Wang、H.-N. Dai、K. Liang 和 M. Imran，《面向工业物联网的轻量级可搜索加密协议》，《工业信息学汇刊》，第17卷第6期，第4248–4259页，2020年。[23] M. Zhang、Y. Chen 和 J. Huang，《Se-ppfm：一种支持云系统中隐私保护模糊多关键词的可搜索加密方案》，《系统杂志》，第15卷第2期，第2980–2988页，2020年。[24] M. A. Salehi、T. Caldwell、A. Fernandez、E. Mickiewicz、E. W. Rozier、S. Zonouz 和 D. Redberg，《RESeED：一种面向存储云的安全正则表达式搜索工具》，《软件：实践与经验》，第47卷第9期，第1221–1241页，2017年9月。[25] D. C. G. Valadares、M. S. L. da Silva、A. E. M. Brito 和 E. M. Salvador，《利用 FIWARE 和英特尔软件防护扩展（SGX）实现安全的数据传播》，载于*IEEE*计算机与通信研讨会（*ISCC*）论文集。IEEE，2018年，第1–7页。[26] M. Kehoe，《将安全需求映射到企业搜索——第1部分：定义具体的安全需求》，
http://www.ideaeng.com/security-eprise-search-p1-0304, 2009, [线上；访问日期：2月16日-2020]。[27] 《亚马逊 Kendra 发布公告：借助机器学习革新企业搜索》，
https://aws.amazon.com/about-aws/whats-new/2019/12/
a
n
n
o
u
n
cing-amazon-kendra-reinventing-enterprise-search-with-machine-learning/, 2019, [线上；访问日期：2月16日-2020]。[28] 《Azure认知服务》，
https://azure.microsoft.com/en-us/services/cognitive-services，访问日期：2022年4月。[29] W. Yu、F. Liang、X. He、W. G. Hatcher、C. Lu、J. Lin 和 X. Yang，《物联网边缘计算综述》，《*IEEE Access*》，第6卷，第6900–6919页，2017年。[30] C. Gong、F. Lin、X. Gong 和 Y. Lu，《物联网中的智能协同边缘计算》，《物联网杂志》，第7卷第10期，第9372–9382页，2020年。[31] Q.-V. Dang 和 C.-L. Ignat，《dtrust：一种用于社交推荐的简单深度学习方法》，载于第三届协作与互联网计算国际会议（*CIC*）论文集，2017年，第209–218页。[32] 《Jetson Nano开发套件》，
https://developer.nvidia.com/embedded/jetson-nano-developer-kit，访问日期：2022年11月。

[33] A. Gholami、S. Kim、Z. Dong、Z. Yao、M. W. Mahoney 和 K. Keutzer，《高效神经网络推理的量化方法综述》，载于低功耗计算机视觉期刊。Chapman and Hall/CRC，2021年，第291–326页。[34] L. Zhou、Y. Zhu 和 A. Castiglione，《在密钥披露受限且数据所有者离线的情况下，在云中对加密数据进行高效的 k近邻查询》，《计算机与安全》，第69卷，第84–96页，2017年。[35] T. Tarpey，《线性变换与 K均值聚类算法：曲线聚类的应用》，《美国统计学家》，第61卷第1期，第34–40页，2007年。[36] W. Sun、B. Wang、N. Cao、M. Li、W. Lou、Y. T. Hou 和 H. Li，《支持基于相似度排序的云端隐私保护多关键词文本搜索》，载于第八届信息、计算机与通信安全研讨会论文集，系列：SIGSAC'13，2013年5月，第71–82页。[37] K. Gai 和 M. Qiu，《基于实数的张量全同态加密中的混合算术运算》，《工业信息学汇刊》，第14卷第8期，第3590–3598页，2018年8月。[38] K. Xing、C. Hu、J. Yu、X. Cheng 和 F. Zhang，《社会参与式感知中的互惠隐私保护k均值聚类》，《工业信息学汇刊》，第13卷第4期，第2066–2076页，2017年。[39] H. Pang 和 B. Wang，《在多密钥环境中使用同态加密进行隐私保护关联规则挖掘》，《系统期刊》，第15卷第2期，第3131–3141页，2020年。[40] P. Li、J. Li、Z. Huang、C.-Z. Gao、W.-B. Chen 和 K. Chen，《云计算中的隐私保护外包分类》，《集群计算期刊》，第21卷第1期，第277–286页，2018年3月。[41] C. Wang、A. Wang、X. Liu 和 J. Xu，《加密数据上的 K均值聚类算法研究》，载于国际网络空间安全研讨会论文集，2019年12月，第182–191页。[42] L. Ducas 和 D. Micciancio，《FHEW：在不到一秒内自举同态加密》，载于《密码学进展》，系列：EUROCRYPT'15，2015年4月，第617–640页。[43] J. Vaidya 和 C. Clifton，《纵向划分数据上的隐私保护 K均值聚类》，载于第九届知识发现与数据挖掘国际会议论文集，2003年，第206–215页。

[44] D. X. Song、D. Wagner 和 A. Perrig，《加密数据搜索的实用技术》，载于《国际安全与隐私研讨会论文集》。*S&P 2000*。IEEE，2000年，第44–55页。[45] D. Boneh、G. D. Crescenzo、R. Ostrovsky 和 G. Persiano，《支持关键词搜索的公钥加密》，载于《国际密码技术理论与应用会议论文集》。施普林格，2004年，第506–522页。[46] N. Cao、Z. Yang、C. Wang、K. Ren 和 W. Lou，《云计算中针对加密图结构数据的隐私保护查询》，载于《第*31*届国际分布式计算系统会议论文集》，系列：ICDCS '11，美国华盛顿特区，2011年，第393–402页。[47] R. A. Popa、C. M. Redfield、N. Zeldovich 和 H. Balakrishnan，《Cryptdb：在加密数据库上处理查询》，*ACM* 通讯，第55卷，第9期，第103–111页，2012年。[48] E. W. Rozier、S. Zonouz 和 D. Redberg，《Dragonfruit：与云服务提供商无关的可信云数据存储与远程处理》，载于第*19*届太平洋地区国际可信计算研讨会论文集。IEEE，2013年，第172–177页。[49] M. Ahmadian、F. Plochan、Z. Roessler 和 D. C. Marinescu，《SecureNoSQL：一种在公有云中安全搜索加密NoSQL数据库的方法》，《国际信息管理杂志》，第37卷，第2期，第63–74页，2017年。[50] M. Kettouch、C. Luca 和 M. Hobbs，《SemiLD：一种基于中介者的框架，用于

r

半结构化数据和关联数据上的关键词搜索，"《智能信息系统》期刊，第52卷，第2期，第311–335页，2019年。[51] M. Naseriparsa、M. S. Islam、C. Liu 和 L. Chen，"Xsnippets：通过片段探索半结构化数据，"《数据与知识工程》期刊，第124卷，文章编号101758，2019年。[52] R. Curtmola、J. Garay、S. Kamara 和 R. Ostrovsky，"可搜索对称加密：改进的定义与高效构造"，载于 论文集

f

第*13*届*ACM*计算机与通信安全会议，CCS '06系列，2006年，第79–88页。[53] J. Woodworth、M. A. Salehi 和 V. Raghavan，"S3c：一种用于云端加密数据空间高效语义搜索的架构"，载于第*4*届国际大数据会议论文集，Big Data' 16系列

,

2016年12月，第 3722–3731 页。

[54] S. Zobaed 和 M. Amini Salehi，“非结构化数据的隐私保护聚类 d
用于基于云的企业搜索解决方案的大数据，”《并发与计算：实践与经验》，第34卷，第22期，e7160页，2022年。[55] J. Li、Q. Wang、C. Wang、N. Cao、K. Ren和W. Lou，“云计算中加密数据上的模糊关键词搜索”，载于第*29*届国际计算机通信会议论文集，丛书：INFOCOM ’10，2010年3月，第1–5页。[56] T. Moataz、A. Shikfa、N. Cuppens-Boulahia和F. Cuppens，“加密数据上的语义搜索”，载于国际电信会议（*ICT*）论文集，2013年5月，第1–5页。[57] X. Sun、Y. Zhu、Z. Xia和L. Chen，“加密云数据上的隐私保护型基于关键词的语义搜索”，《安全及其应用》期刊，第8卷，第3期，2014年5月。[58] F. Liang、R. Qiang和J. Yang，“利用微博圈中的实时信息检索”，载于第*12*届*ACM/IEEE-CS*数字图书馆联合会议论文集，2012年6月，第267–276页。[59] Y. Wang、H. Huang和C. Feng，“微博检索中的局部概念词嵌入查询扩展”，《*IEEE*知识与数据工程汇刊》，2019年10月。[60] S. Deerwester、S. T. Dumais、G. W. Furnas、T. K. Landauer和R. Harshman，“基于潜在语义分析的索引编制”，《美国期刊
n
信息科学学会，第41卷，第6期，第391–407页，1990年9月。[61] K. Albishre、Y. Li 和 Y. Xu，“用于微博检索的有效伪相关性”，载于澳大拉西亚计算机科学周多会议论文集，2017年1月，第1–6页。[62] F. Diaz、B. Mitra 和 N. Craswell，“利用局部训练的词嵌入进行查询扩展”，载于第*54*届计算语言学协会年会论文集（第*1*卷：长篇论文），2016年8月，第367–377页。[63] T. Mikolov、K. Chen、G. Corrado 和 J. Dean，“向量空间中词表示的高效估计”，www.pub-tools-public-publication-data. storage.googleapis.com/pdf/41224.pdf，2013。[64] G. A. Miller，“WordNet：英语词汇数据库”，*ACM* 通讯，第38卷，第11期，第39–41页，1995年。[65] M. Song、I.-Y. Song、X. Hu 和 R. B. Allen，“将关联规则与本体整合用于语义查询扩展”，《数据与知识工程》期刊，第63卷，第1期，第63–75页，2007年10月。

[66] V. Nakade、A. Musaev 和 T. Atkison，“基于叙词表的查询扩展在 T
witter 数据提取中的初步研究”，载于东南地区会议论文集，系列号 ACMSE’18，2018年3月，第1–4页。[67] C. H. Leung、Y. Li、A. Milani 和 V. Franzoni，“基于集体进化概念距离的查询扩展，实现高效网络文档检索”，载于计算科学及其应用国际会议论文集，2013年6月，第657–672页。[68] J. Chen 和 X. Ran，“边缘计算与深度学习：综述”，《*IEEE*会刊》，第107卷，第8期，第1655–1674页，2019年。[69] M. S. Murshed、C. Murphy、D. Hou、N. Khan、G. Ananthanarayanan 和 F. Hussain，“网络边缘的机器学习：综述”，《*ACM*计算调查》（*CSUR*），第54卷，第8期，第1–37页，2021年。[70] Z. Zhou、X. Chen、E. Li、L. Zeng、K. Luo 和 J. Zhang，“边缘智能：利用边缘计算铺就人工智能的最后一公里”，《*IEEE Access*》，第107卷，第8期，第1738–1762页，2019年。[71] X. Wang、Y. Han、V. C. Leung、D. Niyato、X. Yan 和 X. Chen，“边缘计算与深度学习的融合：全面综述”，《*IEEE*通信调查与教程》，第22卷，第2期，第869–904页，2020年。[72] J. Mao、X. Chen、K. W. Nixon、C. Krieger 和 Y. Chen，“MoDNN：面向深度神经网络的本地分布式移动计算系统”，载于欧洲设计、自动化与测试会议暨展览会论文集（*DATE*）。IEEE，2017年，第1396–1401页。[73] K. Ko、Y. Son、S. Kim 和 Y. Lee，“DisCo：用于移动边缘云计算的分布式并发卸载框架”，载于第*9*届国际泛在与未来网络会议（*I
CUFN*）论文集，2017年，第763–766页。[74] R. Hadidi、J. Cao、M. Woodward、M. S. Ryoo 和 H. Kim，“协作机器人实现的分布式感知”，《*I
EEE*机器人与自动化快报》，第3卷，第4期，第3709–3716页，2018年。[75] L. Xie、S. Wang、A. Markham 和 N. Trigoni，“通过深度强化学习实现基于单目视觉的避障”，*arXiv* 预印本 *arXiv:1706.09829*，2017年。[76] S. Yao、Y. Zhao、A. Zhang、L. Su 和 T. Abdelzaher，“DeepIoT：利用压缩器-评论家框架压缩感知系统的深度神经网络结构”，载于第*15*届*ACM*嵌入式网络传感器系统会议论文集，2017年，第1–14页。

[77] Y. Ma、D. Xiang、S. Zheng、D. Tian 和 X. Liu，“将深度学习移入网页浏览器：我们能走多远？”，载于 第 *19* 届万维网会议（*WWW*）论文集，2019年5月，第 1234–1244 页。[78] S. R. Paladhi、R. M. Kumar、A. D. Reddy、C. Vinayak 和 T. Pusphavathi，“在确保安全的同时对大数据进行增强型可能性C均值聚类”，载于 国际计算机网络与通信技术会议。施普林格，2019年，第 583–588 页。[79] P. K. Samantaray、N. K. Randhawa 和 S. L. Pati，“针对外包加密云数据的高效多关键词文本搜索及排序结果”，载于 数据挖掘中的计算智能。施普林格，2019年，第 31–40 页。[80] W. Diffie 和 M. Hellman，“密码学的新方向”，《信息论汇刊》，第 22 卷，第 6 期，第 644–654 页，1976年11月。[81] F. Can 和 E. A. Ozkarahan，“文本数据库基于覆盖系数的聚类方法的概念与有效性”，《*ACM*数据库系统汇刊》，第 15 卷，第 4 期，第 483–517 页，1990年12月。[82] D. R. Cutting、D. R. Karger、J. O. Pedersen 和 J. W. Tukey，“Scatter/gather：一种用于浏览大型文档集的基于聚类的方法”，载于 *ACM SIGIR*论坛，第 51 卷，第 2 期，2017年8月，第 148–159 页。[83] S. Aggarwal、N. Agarwal 和 M. Jain，“使用不同距离度量分析不确定K均值聚类算法的性能”，载于《计算智能：理论、应用与未来方向》第一卷。施普林格，2019年，第 237–245 页。[84] X. Liu 和 W. B. Croft，“使用语言模型的基于聚类的检索”，载于第 *27* 届国际*ACM SIGIR*信息检索研究与发展会议论文集，丛书：SIGIR ’04，2004年7月，第 186–193 页。[85] C. C. Aggarwal、A. Hinneburg 和 D. A. Keim，“高维空间中距离度量的反常表现”，载于第 *8* 届国际数据库理论会议论文集，丛书：ICDT ’01，2001年1月，第 420–434 页。[86] A. L. Mary 和 K. S. Kumar，“一种基于增量数据集的基于密度的动态数据聚类算法”，《计算机科学杂志》，第 8 卷，第 5 期，第 656–664 页，2012。[87] “现代艺术博物馆数据”，tapoueh.org/blog/2018/07/batch-updates-and-concurrency，访问日期：2020年5月。

[88] “Twitter 流 API 数据集”，
github.com/shreybatra/Twitter-Stream-API-Dataset，访问日期：2020年3月。
[89] “Statistics Solution”，
www.statisticssolutions.com/using-chi-square-statistic-in-research/，访问日期：2020年7月15日。[90] R. A. Popa、C. M. S. Redfield、N. Zeldovich 和 H. Balakrishnan，“Cryptdb：通过加密查询处理保护机密性”，载于 第 *23* 届 *ACM* 操作系统原理研讨会论文集，丛书：SOSP ’11，2011 年 10 月，第 85–100 页。[91] D. Wang 和 P. Wang，“使用智能卡的密码认证方案的离线字典攻击”，载于信息安全。Springer International Publishing，2015 年，第 221–237 页。[92] M. Naehrig、K. Lauter 和 V. Vaikuntanathan，“同态加密能否实用？”，载于第 *3* 届 *ACM* 云计算安全研讨会论文集，丛书：CCSW ’11，2011 年 10 月，第 113–124 页。[93] M. George 、 S. Kamara 和 T. Moataz，“ 结构化加密与动态泄漏抑制”，载于密码学理论与应用年度国际会议论文集，2021 年 10 月，第 370–396 页。[94] “亚马逊云科技（AWS）上的 Common Crawl”，
https://aws.amazon.com/public-datasets/common-crawl/，访问日期：2021年2月。[95] “RFC（征求意见稿）系列”，old.datahub.io/dataset/rfcs，访问日期：2020年3月。[96] “BBC 新闻分类”，
www.kaggle.com/c/learn-ai-bbc，访问日期：2020年3月。[97] O. Medelyan、E. Frank 和 I. H. Witten，“使用自动关键短语提取实现媲美人类的标签标注”，载于第 *14* 届自然语言实证方法会议论文集，丛书：EMNLP ’09，2009 年 8 月，第 1318–1327 页。[98] P. Berkhin，“聚类数据挖掘技术综述”，载于多维数据分组。施普林格，2006年，第 25–71 页。[99] G. A. Miller，“ WordNet：英语词汇数据库”，*ACM* 通讯，第 38 卷，第 11 期，第 39–41 页，1995 年 11 月。

[100] M. Z. Rodriguez、C. H. Comin、D. Casanova、O. M. Bruno、D. R. Amancio、L. d. F. Costa 和 F. A. Rodrigues，《聚类算法：一种比较方法》，*PLOS ONE*期刊，第14卷，第1期，文章编号 e0210236，2019年。[101] B. C. Kwon、B. Eysenbach、J. Verma、K. Ng、C. De Filippi、W. F. Stewart 和 A. Perer，《Clustervision：无监督聚类的可视化监督》，*IEEE*可视化与计算机图形学汇刊，第24卷，第1期，第142–151页，2017年。[102] A. K. Mariappan、R. M. Suresh 和 V. S. Bharathi，《基于 TSAP 度量的语义搜索引擎与关键词搜索引擎有效性比较研究》，计算机应用、电子政务与云计算服务期刊，第4–6页，2012年12月。[103] M. Amini Salehi、T. Caldwell、A. Fernandez、E. Mickiewicz、D. Redberg、E. W. D. Rozier 和 S. Zonouz，《RESeED：云端加密数据正则表达式搜索》，载于第七届国际云计算会议论文集，Cloud '14丛书，2014年6月，第673–680页。[104] A. Silva 和 M. Mendoza，《利用词嵌入改进查询扩展策略》，载于*ACM*文档工程研讨会论文集，2020年9月。[105] S. Kuzi、A. Shtok 和 O. Kurland，《利用词嵌入进行查询扩展》，载于第*25*届信息与知识管理国际会议论文集，2016年。[106] C. Fellbaum，《WordNet：一种电子词汇资源》，载于《牛津认知科学手册》。劳特利奇，2017年，第301–314页。[107] R. Campos、V. Mangaravite、A. Pasquali、A. M. Jorge、C. Nunes 和 A. Jatowt，《YAKE！与语料集无关的自动关键词提取器》，载于第*40*届欧洲信息检索会议论文集，2018年3月，第806–810页。[108] A. Khatua、A. Khatua 和 E. Cambria，《两场疫情的故事：利用上下文 Word2Vec 对疫情暴发期间的 Twitter 信息流进行分类》，《信息处理与管理》期刊，第56卷，第1期，第247–257页，2019年。[109] Z. Kastrati、A. S. Imran 和 S. Y. Yayilgan，《深度学习对使用语义丰富表示进行文档分类的影响》，《信息处理与管理》期刊，第56卷，第5期，第1618–1632页，2019年。[110] X. Pang、Y. Zhou、P. Wang、W. Lin 和 V. Chang，《一种创新的股市预测神经网络方法》，《超级计算期刊》，第76卷，第3期，第2098–2118页，2020年。

[111] 《普通循环神经网络》，
h
t
t
p://calvinfeng.gitbook.io/
machine-learning-notebook/supervised-learning/
recurrent-neural-network/recurrent neural networks，访问于2020年9月。
[112] F. Viegas、S. Canuto、C. Gomes、W. Luiz、T. Rosa、S. Ribas、L. Rocha 和 M. Gonçalves，《Cluwords：利用语义词聚类表示增强主题建模》，载于第*12*届国际网络搜索与数据挖掘会议论文集，2019年1月，第753–761页。
[113] 《Amazon Kendra》，www.aws.amazon.com/kendra，访问于2020年4月10日。[114] Y. Zhou、S.-M. Moosavi-Dezfooli、N.-M. Cheung 和 P. Frossard，《深度神经网络的自适应量化》，载于*AAAI*人工智能会议论文集，第32卷，第1期，2018年。[115] D. G. Samani 和 M. A. Salehi，《探究虚拟化对深度学习应用可用性的影响》，载于第*22*届*IEEE*集群、云与互联网计算国际研讨会（*CCGrid*）论文集，2022年，第442–451页。[116] J. Manner、M. Endreß、T. Heckel 和 G. Wirtz，《函数即服务中的冷启动影响因素》，载于第*12*届*IEEE/ACM*实用计算与云计算国际会议伴会论文集（*UCC*），2018年，第181–188页。[117] X. Chen、Y. Wu 和 Y. Han，《Fepim：基于铁电场效应晶体管的无争用内存计算》，载于第*26*届*IEEE*亚洲及南太平洋设计自动化会议论文集（*ASP-DAC*），2021年，第114–119页。[118] T. V. Christensen，《NP完全问题的启发式算法》，项目报告，信息学与数学建模研究所，丹麦技术大学，2007年。[119] E. J. O’neil、P. E. O’neil 和 G. Weikum，《用于数据库磁盘缓冲的LRU-K页面置换算法》，*ACM SIGMOD Record*期刊，第22卷，第2期，第297–306页，1993年。[120] A. Mokhtari、C. Denninnart 和 M. A. Salehi，《在异构计算系统中实现鲁棒性的自主任务丢弃机制》，载于国际并行与分布式处理研讨会专题研讨会论文集（*IPDPSW*），2020年，第17–26页。[121] T. Van Erven 和 P. Harremos，《Rényi散度与Kullback–Leibler散度》，*IEEE*信息论汇刊，第60卷，第7期，第3797–3820页，2014年。

[122] S. Dasgupta 和 D. Hsu，《面向主动学习的分层采样》，载于第*25*届国际机器学习会议论文集，2008年，第208–215页。[123] P. Kumar 和 A. Gupta，《用于分类、回归和聚类的主动学习查询策略：综述》，《style id='5'>《计算机科学与技术学报》，第35卷，第4期，第913–945页，2020年。[124] F. Min、S.-M. Zhang、D. Ciucci 和 M. Wang，《通过聚类选择实现三支主动学习》，国际机器学习杂志与

d

控制论，第 11 卷，第 5 期，第 1033–1046 页，2020年。

个人简历

Sm Zobaed 于2015年秋季在孟加拉国伊斯兰科技大学（IUT）获得计算机科学与工程理学学士学位。他于2015年12月开启职业生涯，在一家顶尖科技巨头——华为技术有限公司担任系统工程师。大约一年半后，他计划通过接受高等教育来丰富自己的学术知识。因此，Sm Zobaed 于2017年秋季在路易斯安那大学拉法叶分校开启了计算机科学博士之旅。攻读博士期间，他于2019年春季获得计算机科学理学硕士学位。他的研究兴趣包括：自然语言处理、数据分析和云计算。