基于LLM的互联网设备指纹识别框架

Armin Sarabi 密歇根大学 Ann Arbor, MI, USA arsarabi@umich.edu

Tongxin Yin 密歇根大学 Ann Arbor, MI, USA tyin@umich.edu

Mingyan Liu 密歇根大学 Ann Arbor, MI, USA mingyan@umich.edu

摘要

在本文中，我们提出使用大型语言模型（LLMs）来表征、聚类和指纹化从网络测量中获得的原始文本。为此，我们首先在一个包含数亿个从互联网范围扫描中获得的横幅的数据集上训练了一个基于Transformer的掩码语言模型，即RoBERTa。我们进一步使用对比损失函数（由领域知识驱动对该模型进行微调，以生成时间上稳定的数值表示（嵌入）这些表示可以直接用于下游学习任务。我们的嵌入具有鲁棒性，能够适应横幅内容中的微小随机变化，并保持相似硬件/软件产品嵌入之间的接近性。我们进一步使用基于密度的方法（HDBSCAN）对HTTP横幅进行聚类，并检查获得的聚类以生成基于文本的指纹，用于标记原始扫描数据。我们将我们的指纹与Recog进行了比较，Recog是一个手动整理的指纹数据库，结果表明我们能够识别出Recog之前未捕获的新物联网设备和服务器产品。我们提出的方法为未来研究指明了一个重要方向，即利用最先进的语言模型自动分析、解释和标记互联网扫描产生的大量数据。

CCS 概念

• 网络 → 网络测量；• 计算方法 → 神经网络；自然语言处理。

关键词

互联网扫描、设备指纹识别、深度学习、大型语言模型

ACM 引用格式：

Armin Sarabi, Tongxin Yin, 和 Mingyan Liu. 2023. 一个基于大型语言模型（LLM）的互联网连接设备指纹识别框架。发表于 *2023 年 ACM 互联网测量会议（IMC '23）论文集，2023 年 10 月 24–26 日，加拿大魁北克省蒙特利尔。* ACM，美国纽约，7 页。https://doi.org/10.1145/3618257.3624845

1 引言

随着互联网连接设备的激增，网络扫描技术应运而生，为公共互联网提供了可见性。诸如Censys[13]和Shodan[1]等项目和实体执行常规的全互联网扫描，并记录互联网上许多端口上可见设备的快照。这些测量结果被广泛应用于多种目的，包括检测和指纹识别网络设备[5, 11, 17, 27]、研究趋势[16, 19, 20]、检查安全事件[4, 14]以及支持各种机器学习分析[22, 26]。然而，这些全互联网扫描本质上是由从协议握手（包括横幅抓取）中获得的原始信息组成，其标签/特征覆盖率较低，例如，用于识别底层硬件/软件产品，或促进自动化分析。这对利用扫描数据的研究人员、网络管理员和安全从业者提出了挑战，因为他们需要开发自己的数据处理管道来过滤相关信息并理解原始数据。

同时，深度学习领域的最新进展推动了大型语言模型（LLMs）的发展，用于复杂的文本分析。特别是基于Transformer的模型[29]，如BERT[12]和GPT[6]，已成功应用于许多自然语言处理（NLP）任务，如语言建模、翻译、文本分类和聚类，取得了最先进的性能。

有趣且关键的是，互联网扫描产生的大量文本数据以及*扫描数据的*文本基础特性，使其非常适合用于训练大型语言模型。受此观察启发，本文在从互联网扫描中获取的快照上训练并评估了一个大型语言模型（LLM），将原始文本提炼为通用嵌入，这些嵌入适用于下游的机器学习任务和分析。先前的工作[26]已使用深度学习模型生成数值嵌入来表征互联网主机，但该方法依赖于先将扫描数据转换为二进制向量，然后再用于训练。相比之下，据我们所知，本研究首次提出了一个*直接在原始文本上*训练的模型，无需中间的特征提取步骤（例如，[26]中使用的词袋模型）。事实证明，这使得模型能够学习扫描数据的底层结构，并支持更具可解释性的分析（例如，通过直接标注文本）。

我们模型的输出是机器可读的嵌入，它将扫描数据编码为数值向量，并可用于表征底层主机。此外，我们训练模型生成高保真且稳健的嵌入，这些嵌入不会随时间漂移，因为它们对动态部分（如每次探测服务时都会重新生成的时间戳和随机ID（例如cookie ID））的敏感度较低。

我们还初步研究了使用我们的模型生成的HTTP横幅嵌入进行聚类和指纹生成。这些应用旨在补充基于手动筛选的指纹构建的框架，例如

IMC '23, 2023年10月24日至26日，加拿大魁北克省蒙特利尔
 ACM ISBN 979-8-4007-0382-9/23/10。
https://doi.org/10.1145/3618257.3624845

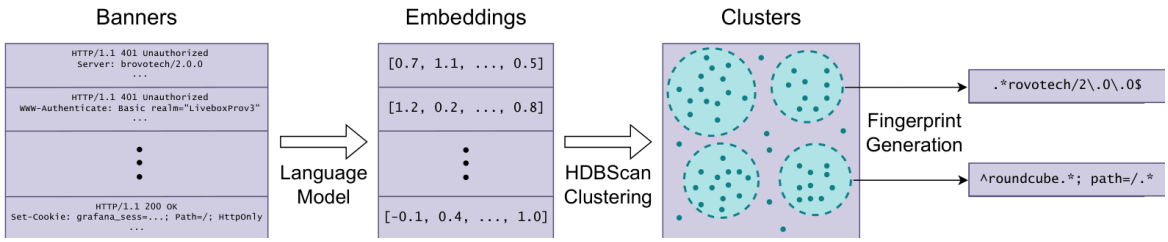

图 1：我们的指纹生成流程概览。

Recog [25]。此方法的最终产品是基于文本的指纹（即正则表达式），可用于标注横幅中的重要信息，进而识别部署在主机上的软件/硬件产品。对我们指纹的进一步检查表明，我们可以识别出 Recog 未捕获的新物联网设备和服务器产品，从而增加标签覆盖率。将此研究扩展到 HTTP 以外的协议，并对生成的指纹进行彻底检查和净化，是我们正在进行的工作的重要组成部分。我们的整体流程如图 1 所示，代码可在 https://github.com/arsarabi/llm-device-fingerprints 获取。

2 相关研究

互联网主机的通用数值表示：Sarabi 和 Liu [26] 在互联网扫描数据的词袋表示上训练了一个变分自编码器（VAE），并使用由此得到的嵌入进行分类和推理等学习任务。另一方面，我们提出使用 Transformer 生成的嵌入，该方法跳过中间的词袋特征提取以最小化信息损失，并允许对原始文本进行标注以获得更好的可解释性。我们的模型还经过训练，能够使用第 4.2 小节中详细描述的对比损失函数生成时间稳定的嵌入，使其更适合用于聚类和指纹生成。

基于主动网络探测的产品指纹识别：Recog [25] 是一个开源项目，它使用人工整理的指纹来识别软件/硬件产品、服务和操作系统。Feng 等人 [17] 提出了一种基于获取规则的引擎（ARE），这是一种无监督方法，可以生成用于发现物联网设备的规则。Javed 等人 [18] 尝试复现 [16] 中开发的引擎的实现，但未能达到原论文中报告的准确率。为了解决可复现性问题，我们在实现中使用了标准的开源库，使研究人员能够利用我们的嵌入和生成的指纹，以及复用模型并在下游任务上进行微调。

物联网设备被动流量监测识别：Wang 等人 [30] 提出了 IoT-Portrait，这是一种基于 Transformer 的模型，采用增量学习技术自动识别物联网设备。Meidan 等人 [23] 提出了 ProfilIoT，这是一种利用基于决策树的模型的机器学习方法。Aksoy 和 Gunes[2] 使用遗传算法开发了一种自动物联网设备识别系统，而 Aneja 等人 [3] 则通过深度学习分析数据包的到达间隔时间来探索物联网设备的指纹识别。Msadek 等人 [24] 提出了一种基于机器学习的方法，利用加密流量分析对物联网设备进行指纹识别。Chowdhury 和 Abas[9] 对资源受限物联网设备的设备指纹识别方法进行了调查。

3 数据与预处理

3.1 原始数据

我们使用了来自 Censys 通用互联网数据集 [8] 的 6 个不同快照，这些快照对应于 2023 年 7 月至 12 月期间每个月的第一个星期二。在撰写本文时，Censys 包含对 107 个协议和 3500 多个端口的整个 IPv4 空间的扫描。我们从每个快照中收集 Censys 记录的非空服务横幅，忽略被标记为截断的服务。[1]为了减小数据集的大小，我们从每个快照中抽取 10% 的 IP 地址进行二次抽样，确保所有快照中的 IP 选择一致。这导致所有快照中共有 ~2.6 亿个横幅。

3.2 分词

为了将文本数据输入到 Transformer 模型中，首先需要将其拆分为词元。BERT[12] 使用子词元来限制其词汇量，同时最大限度地减少在遇到分词器训练过程中未见过的序列时生成的未知词元。相比之下，RoBERTa[21] 使用字节级、字节对编码（BPE）的分词器，其优点是在必要时回退到单字节词元，以防止生成未知词元，从而使分词无损。因此，我们使用 RoBERTa 分词器，因为其具有无损性（尤其是因为扫描可能包含文本和二进制数据的组合，如果使用非字节级分词器，会增加生成未知词元的可能性）。但请注意，由于我们数据的性质不同，预训练的 RoBERTa 分词器可能不是最优选择。因此，我们在数据集中随机选择了 1 亿个样本，并使用 50,000 个词元重新训练了一个分词器，并在本文的其余部分使用它来将横幅编码为词元序列。

[1]根据 Censys (https://support.censys.io/hc/en-us/articles/4407300349588Search-2-0-Troubleshooting-Q-A) 的说明，被标记为截断的服务表示主机上存在超过 100 个服务，这些服务极有可能是蜜罐或被防火墙屏蔽的主机，而非真实服务。

4 嵌入生成

在本节中，我们将介绍训练用于生成横幅嵌入的 Transformer 模型所涉及的步骤，包括一个掩码语言模型和一个用于生成稳定嵌入的监督模型（使用对比损失函数进行训练）。

4.1 掩码语言模型

我们首先在数据集中的横幅上训练一个掩码语言模型（MLM）。MLM 通过随机掩码一小部分标记，然后尝试根据周围文本提供的上下文来推断被掩码的标记来进行训练。这激励模型学习语义关系，从而得到具有上下文感知的嵌入。请注意，MLM 是双向模型，意味着一个标记的嵌入依赖于其前后的标记。MLM 通常是训练大型语言模型的第一步，然后这些模型会在下游任务上进行微调，例如分类或机器翻译。我们训练了一个具有 256 维嵌入、4 层、4 个注意力头和中间层大小为 1024 的 RoBERTa [21] 模型。我们以 1024 的批量大小和 0.0002 的学习率训练该模型 100,000 次，并在训练过程中随机掩码所有标记的 15%。上述掩码语言模型是在 Censys 数据集中记录的所有协议上训练的，使其成为处理此类数据的研究的良好起点。然而，本文余下部分的模型和分析仅关注来自 HTTP 协议的横幅，这些横幅占我们数据集中横幅的 $\sim 70\%$。

4.2 生成稳定的横幅嵌入

*4.2.1 模型描述。*横幅可以包含随时间变化的动态部分，包括时间戳和每次探测主机时重新生成的随机 ID。理想的横幅嵌入应该对这些变化具有不变性。因此，我们使用有监督的对比损失来激励模型生成随时间保持稳定的嵌入，同时最大化不同服务嵌入之间的距离。为此，我们首先从两个连续的快照（以下用 $K\in\{1, 2\}$ 表示相关快照）中识别出对应于相同 IP 地址和端口的横幅对，这强烈表明它们来自同一设备。然后，我们识别出两个横幅中的共同部分；这是通过先将每个横幅拆分为各自的头部，然后执行公共子字符串匹配来识别每个头部中的共同部分来实现的。[2]我们忽略长度小于 3 个字符的单个匹配，并丢弃所有匹配的总长度小于 11 个标记的横幅对。后者过滤掉了可能对应于不同设备/服务的横幅对，[3] 阈值是通过手动检查随机选择的横幅对来确定的。

形式上，用 $t_{i,j}^{(k)}, 1 \le j \le l_i^{(k)}$ 表示与横幅相关的标记序列，其中 1 个 $\le I \le N$ 表示一对横幅，$l_i^{(k)}$ 表示标记序列的长度。然后，我们生成标签 $y_{i,j}^{(k)} \in \{0, 1\}$，指示两个横幅之间是否如前所述匹配了某个标记。我们使用这些标签来训练一个标记分类模型，该模型能够预测横幅中的稳定部分，而这些稳定部分又承载着关于底层服务的重要信息。

为了生成横幅嵌入，我们需要聚合序列中所有标记的嵌入。通常，使用第一个标记（通常是特殊的句首标记）的嵌入或所有标记的平均嵌入作为整个序列的嵌入。但请注意，标签为 0 的标记不应计入横幅嵌入，因为它可能被视为噪声，且不包含与底层宿主相关的信息。因此，我们使用所有标记嵌入的加权平均值，以预测标签作为权重，从而得到对每个横幅中存在的噪声具有鲁棒性的嵌入。设 $e_{i,j}^{(k)} \in \mathbb{R}^m$ and $0 \le \hat{y}_{i,j}^{(k)} \le 1$分别为 M 维标记嵌入（来自 Transformer 模型的最后一层）和预测标签。然后，我们定义横幅嵌入 $\bar{e}_i^k$如下（注意，嵌入被归一化以使其 ℓ_2 范数为一，我们稍后会对此进行说明）。

$$\bar{e}_i^{(k)} := \frac{e_i^{(k)}}{\|e_i^{(k)}\|_2}, \quad e_i^{(k)} := \frac{\sum_j \hat{y}_{i,j}^{(k)} e_{i,j}^{(k)}}{\sum_j \hat{y}_{i,i}^{(k)}}.$$

虽然上述方法使得嵌入对横幅的动态部分（即标签为 0 的部分）具有不变性，但由于静态部分（即标签为 1 的部分）的上下文感知特性，其词嵌入并不保证保持不变，这反过来可能导致嵌入随时间漂移。因此，我们修改了损失函数，以迫使模型为匹配对生成相似的嵌入，同时最大化随机选择的对之间的距离，具体如下：

$$\begin{aligned}\mathcal{L} = &-\frac{1}{n_t}\sum_{i,j,k} y_{i,j}^{(k)} \log \hat{y}_{i,j}^{(k)} \;+ (1 - y_{i,j}^{(k)})(1 - \log \hat{y}_{i,j}^{(k)}) \\ &+ \frac{1}{n}\;\sum_i \|\bar{e}_i^{(1)} - \bar{e}_i^{(2)}\|_2 - \frac{1}{n}\sum_i \|\bar{e}_i^{(1)} - \bar{e}_{i\oplus 1}^{(2)}\|_2\,, \qquad (1)\end{aligned}$$

其中，N 和 N_T 分别表示小批量中的标志对数量和标记数量。公式 1 中的第一项是标记分类的二元交叉熵损失。第二项迫使模型为匹配对生成相似的嵌入，而最后一项则鼓励模型最大化任意对之间的距离。注意，对于最后一项，$\oplus$ 表示用于生成非匹配对的循环移位。另外，我们使用单位范数的嵌入，以防止它们因最后一项而无限增大。

上述方法利用时序数据（通过同一设备/服务大约相隔一个月的两张快照）来监督并帮助模型生成更高质量的嵌入。请注意，使用不同/相似对进行对比训练的思想也已在其他领域得到应用，例如，在科学文献中的 [10] 和神经记录中的 [28]。时序信息进一步激励模型为同一硬件或软件产品的不同版本/配置生成相似的嵌入。这使得我们的嵌入适用于聚类，我们将在下一节中对此进行探讨。

*4.2.2 训练与评估。*我们使用公式 1 中的损失函数对第 4.1 小节中描述的掩码语言模型进行微调，迭代次数为 20,000 次，批量

[2]将 HTTP 横幅拆分为头部可以防止不同头部之间的匹配，并进一步增强模型对可能以随机顺序返回头部且在不同快照之间发生变化的服务器的鲁棒性。[3]$This can happen, e.g., when the underlying device/service is switched out by the operator, or when different physical devices happen to be observed on the same IP/port$

大小为 1024（512 对），学习率为 0.00005。在包含 100,000 对数据的保留测试集上，词元分类的准确率达到 98.3%，其中正标签的精确率/召回率分别为 96.9%/98.9%，负标签的精确率/召回率分别为 99.3%/97.9%。这表明可以非常准确地预测横幅的静态/动态部分。图 2 展示了一个示例，其中模型输出用于标注横幅的静态/动态部分（灰色区域表示模型预测的动态部分）。我们观察到，模型能够正确预测动态生成的内容（例如，X-LLID 头和时间戳的内容）。[4]有趣的是，版本号的尾部也被预测为动态，而事实也证明了这一点，因为下一个快照的版本号为 5.0.3.0。这归因于模型在训练过程中提供了匹配对，使其能够观察到并随后预测频繁的变化（例如，小版本更新）。

```
HTTP/1.1 400 Bad Request
Server: EdgePrism/5.0.2.0
Mime-Version: 1.0
Date: <REDACTED>
Content-Type: text/html
Expires: Tue, 06 Sep 2022 00:38:30 GMT
X-LLID: 6d0829a0cfebc516165dbf208b4d4176
Content-Length: 0
Connection: close
```

图 2：一个 HTTP 横幅的示例：灰色区域表示预测为动态的部分。请注意，模型会忽略头部名称（以及状态行的开头“HTTP/”）且永远不会对其进行标注。

图 2 还展示了我们的模型的一项重要能力，即通过标注横幅中的重要信息，从而通过深入了解模型的输出，提高其可解释性。这是使用基于 Transformer 的模型的直接结果，该模型能够将预测结果映射回原始文本。这与文献 [26] 中使用的词袋模型和自编码器模型形成对比，后者会导致可解释性的丧失。

4.3 嵌入表征分析

我们进一步分析了由我们的模型生成的嵌入，以展示其在表征扫描数据集方面的实用性。图 3 展示了使用第 4.2 小节中详细描述的有监督/对比模型以及第 4.1 小节中训练的普通掩码语言模型所获得的匹配对和随机对之间的 ℓ_2 距离分布。两幅图均显示了两组数据之间的分离，其中对比学习实现了更为明显的区分。对于对比（掩码语言）模型，97.0%（94.9%）的匹配对的距离小于 0.1，而 97.8%（99.0%）的随机对的距离大于 0.1。图 3 左侧的跳跃部分包含了距离为零的配对（由于两个横幅之间完全匹配），占所有配对的 52.3%。对于非零距离的配对，匹配对的对数距离的平均值和标准差为 -4.58 ± 1.46（-2.14 ± 0.70），随机对的对数距离的平均值和标准差为 0.22 ± 0.54（-0.31 ± 0.21）。因此，我们可以看到，对比训练使得匹配对和随机对之间的距离差异达到了三个数量级。

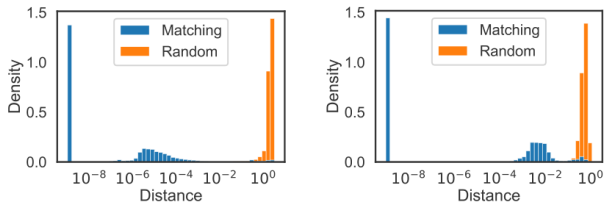

图 3：对于我们提出的对比模型（左）和基础掩码语言模型（右），匹配和随机横幅对嵌入之间的 ℓ_2 距离分布。对比训练使得两组之间的差异更为显著（大约三个数量级）。注意，x 轴采用对数刻度。

图 3 中的成对数据相隔一个月。我们还在观察窗口的开始/结束时重复了相同的实验，得到了相隔五个月的成对数据。对于这个更长的窗口，具有完全相同横幅的成对数据的百分比降至 16.6%，而距离小于 0.1 的匹配成对数据的百分比为 94.7%（对于掩码语言模型为 91.1%）。这表明，有监督训练有助于获得在长观察窗口内稳定的鲁棒嵌入

我们还随机选择了 100 对距离大于 0.1 的横幅对进行检验。对于所有检验的示例，我们观察到它们要么对应于不同的硬件/软件（由于头部内容完全不同，尤其是服务器头部），要么对应于更新的配置（由于头部内容的变化超出了简单的版本更新，例如，头部的添加/移除）。我们在图 4 中包含了这两种情况的示例。这表明，嵌入空间中的距离可用于检测主机中的重大/异常变化，例如，由于配置更新，或者当观察结果对应于完全不同的设备时。请注意，公式 1 中的对比损失通过最小化横幅动态部分（即图 2 中的灰色部分）预期变化导致的距离，减少了误报的数量（例如，与编辑距离等简单方法相比）。进一步说明，尽管图 2 中的示例在归一化编辑（Levenshtein）距离为 0.166 的情况下，与其后续快照的距离为 3.6e−4。[5]

5 指纹生成

第 4 节中详细描述的横幅嵌入捕获了底层服务的特征，并且对横幅中可能存在的任意噪声具有鲁棒性。为了确定是否可以使用相似的嵌入来识别相似的物理设备（即具有相同的制造商和/或型号）或服务器软件（具有相同的供应商/产品），我们根据本节后续内容所述，从聚类服务横幅中自动生成并检查基于文本的正则表达式指纹。

[4]请注意，由于我们所有的数据都来自同一年，因此时间戳的年份部分仍预计为静态。更长的观察窗口应使模型学会将年份也视为动态因素。
[5]我们将归一化编辑距离定义为绝对编辑距离（将一个字符串转换为另一个字符串所需的操作数）除以两个字符串的最大长度。

```
HTTP/1.1 200 OK
Date:  <REDACTED>
Server: Apache
X-Frame-Options: SAMEORIGIN
Last-Modified: Mon, 21 Sep 2020 23:43:29 GMT
ETag: "c445-5afdb6adb2a40"
Accept-Ranges: bytes
Content-Length: 50245
Cache-Control: no-store, must-revalidate
Pragma: no-cache
Expires: 0
Content-Type: text/html; charset=UTF-8
Cache-Control: no-cache
```

```
HTTP/1.1 200 OK
Age:          1
Date:  <REDACTED>
Cache-Control: no-cache,no-store,must-revalidate
Connection: Keep-Alive
Via: NS-CACHE-10.0: 127
ETag: "a717-5ec898f83af80"
Server: Apache
X-Frame-Options: SAMEORIGIN
Last-Modified: Thu, 03 Nov 2022 04:40:46 GMT
Accept-Ranges: bytes
Content-Length: 42775
Feature-Policy: camera 'none'; microphone 'none'; geolocation 'none'
Referrer-Policy: no-referrer
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Content-Type: text/html; charset=utf-8
```

```
HTTP/1.1 200 OK
Server: nginx
Date:  <REDACTED>
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding
X-Powered-By: PHP/5.4.16
Content-Encoding: gzip
```

```
HTTP/1.1 200 OK
Date:  <REDACTED>
Server: Apache
Last-Modified: Thu, 24 Nov 2022 17:26:40 GMT
ETag: "1cb-5ee3ab54558da"
Accept-Ranges: bytes
Content-Length: 459
Connection: close
Content-Type: text/html; charset=UTF-8
```

图 4：展示主要配置变化（上）且对应于不同服务器产品（下）的横幅对示例（从同一 IP 地址和端口获取的相隔一个月的快照）。上方的对在嵌入空间中的距离为 0.3，而下方的对距离为 2.3。

5.1 聚类

为了加速聚类过程，我们应用主成分分析（PCA）将嵌入维度从 256 降低到 64。我们使用 500 万个随机选择的嵌入来训练 PCA。对于所选的主成分，解释方差比的累积和为 99.97%，从而实现了最小的信息损失。

在聚类方面，我们使用 HDBSCAN[15]，这是 DBSCAN[7] 的层次变体，用于基于密度的聚类。原始的 DBSCAN 会检测被低密度区域包围的高密度区域。虽然 DBSCAN 需要一个 ϵ 值（两个样本被视为邻居的最大距离），但 HDBSCAN 通过尝试不同的值并找到在 ϵ 值范围内稳定性最佳的聚类，从而去除了这个超参数。我们选择 HDBSCAN 是因为它能够处理不同形状和大小的聚类（例如，无法通过基于质心的聚类方法（如 K-Means）检索到的非凸聚类），并且不需要预先知道聚类的数量。

我们使用 500 万个嵌入向量训练 HDBSCAN，并将 min_cluster_size 设置为 50，min_samples 设置为 5。HDBSCAN 还可以设置 cluster_selection_epsilon 参数，当聚类小于给定阈值时，这些聚类将被合并；增加该值可以防止算法生成微聚类，从而减少聚类的总数。我们通过设置 cluster_selection_epsilon 为 $\in\{0.01, 0.02, 0.05, 0.1\}$，生成了四个不同粒度级别的聚类。得到的聚类数量和未聚类离群点的百分比分别为 5452/5.86%、3989/4.57%、2138/2.32% 和 736/0.63%

5.2 指纹生成

为了生成描述每个簇的基于文本的指纹，我们从簇中随机选择 10 个样本，并应用最长公共子字符串匹配来提取所有样本之间的公共子字符串。请注意，这种匹配是基于每个报头进行的。然后，我们将结果转换为正则表达式（由一系列子字符串和通配符表达式组成），从而为每个报头生成一个正则表达式模式。我们通过从每个簇中选择 100 组不同的 10 个样本，并从第 5.1 小节中的所有四个聚类中生成模式，来重复此过程，以获得具有不同粒度的更大指纹池。此过程从所有报头中生成了 15,718 个模式/指纹，可能捕获物理（如物联网）设备和/或服务器软件。

我们将我们的指纹与来自 Recog [25] 的手工整理的正则表达式模式进行了比较，这些模式也被 Censys 用于标记扫描数据。我们首先提取了 HTTP 服务器、Set-Cookie 和 WWW-Authenticate 头部的 Recog 指纹，在撰写本文时分别包含 447、82 和 77 个指纹。然后，我们检查了我们的指纹（对于上述头部，分别为 798、2478 和 635 个），并发现了 Recog 中未捕获的指纹。表 1 包含了一些这样的示例，包括用于识别物联网设备和服务器产品的模式。这表明我们的框架可以用于补充现有的指纹数据库，并且可以定期应用于扫描数据，以保持指纹的最新状态。

表 1：由不同 HTTP 报头生成且未被 Recog [25] 数据库捕获的硬件/软件指纹示例。这表明我们的自动指纹生成技术能够补充现有数据库，并有助于保持指纹的最新状态。

Header	Regular expression	Description
Server	`.*rovotech/2\.0\.0$`	Brovotech IP Camera
Server	`^ALARM\.COM-HTTP-Server$`	Home Automation/monitoring
Server	`^ZNC .* - http://znc\.in$`	ZNC IRC Network Bouncer
Set-Cookie	`^grafana_sess=.*; Path=/; HttpOnly$`	Grafana Web Application
Set-Cookie	`^interworx-cp=.*; path=/.*`	InterWorx Web Hosting Control Panel
Set-Cookie	`^roundcube.*; path=/.*`	Roundcube Email Client
WWW-Authenticate	`^Basic realm="LiveboxProv3"$`	Sagemcom Livebox Pro V3 Router
WWW-Authenticate	`^Basic realm="ZNID24xx.*-Router"$`	Zhone zNID 24xx Series Router
WWW-Authenticate	`^Digest realm="Wisenet NVR", nonce=".*", qop="auth"$`	Hanwha Wisenet Network Video Recorder

我们还研究了使用该技术恢复的 Recog 指纹数量。我们首先通过寻找重叠率最高的指纹对，将 Recog 指纹与我们的指纹进行匹配。[6]然后，我们筛选出重叠率至少为 90% 的指纹，结果分别从 HTTP 服务器、Set-Cookie 和 WWW-Authenticate 头中恢复了 117、9 和 22 个 Recog 指纹。我们进一步观察到，对于每个头，Recog 标记的所有横幅中，恢复的指纹分别占 98.1%、63.2% 和 61.2%。此外，我们还通过寻找作为 Recog 模式子集（覆盖率 >90%）的指纹来搜索部分匹配。这为上述头中的 Recog 指纹分别找到了 251、25 和 51 个部分匹配，占 Recog 标记的所有横幅的 99.3%、89.6% 和 96.1%。该分析表明，我们提出的技术在恢复频繁模式方面是成功的。改进这种方法以同时恢复较少频繁的模式，是我们未来研究将探索的方向。

6 讨论

本文展示了大型语言模型（LLMs）在互联网扫描数据自动分析中的潜在应用，尤其是用于对相似设备进行聚类以及提取硬件/软件指纹。Transformer 对原始文本进行标注的能力也可以成为一种有用的工具，用于突出显示横幅中的重要信息，如图 2 所示。但请注意，虽然生成的指纹可用于识别特定产品（如表 1 所示），但仍需要进行手动验证，以清理相关的正则表达式模式并去除（近似）重复项。例如，表 1 中的一些模式除了捕获底层产品外，还捕获了设备配置（例如，通过 Set-Cookie 头部的正则表达式中的 “path=/.*” 部分）。此外，正则表达式通配符部分捕获的数据有时也包含有用信息，例如，Zhone 路由器的模式可以捕获不同的型号，如 ZNID24xxA1 或 ZNID24xxB1。

鉴于上述局限性，对我们正在进行的工作而言，对生成的模式进行详尽检查以获得更高质量的指纹是一个重要方面。研究是否也能利用机器学习实现指纹清洗的自动化将是一件有趣的事情。此外，值得注意的是，虽然我们仅在 HTTP 协议上展示了我们的框架，但相同的方法也可以应用于其他现有指纹更为稀疏的协议。

所提出的技术侧重于时间稳定性和识别横幅广告的静态/动态部分的能力，以确保嵌入的高质量。这是通过在训练过程中利用匹配/相似对来实现的，使嵌入对横幅广告的动态部分具有不变性，同时最大化随机对之间的距离。这遵循了对比学习的思想，以创建高保真度的嵌入，这种方法也已应用于其他领域，例如表征科学文献 [10] 和神经记录 [28]。然而，这也使嵌入受制于训练过程中使用的匹配对的质量。我们使用从同一 IP/端口获得的快照来确保时间稳定性，同时省略了可能对应于不同设备/服务的对，如第 4.2 小节所述。但请注意，相同的方法也可以与其他用于检索相似对的技术一起使用，以为其他应用驱动的相似性标准生成高质量的嵌入。

7 结论

本文介绍了一个基于从全网扫描中获得的数亿个横幅广告训练的大型语言模型（LLM），据我们所知，这是首次进行此类尝试。我们的初步分析表明，我们可以生成稳定且健壮的数值嵌入，捕获互联网主机上部署的基础软件/硬件产品的相关信息。这使得我们的模型及其生成的嵌入成为自动化分析、异常检测、聚类和指纹生成的合适候选方案，我们将在未来的研究中对此进行更详细的探索。利用我们的方法来捕获设备/服务配置、处理网络流量轨迹，并进一步利用大型语言模型的生成能力，是未来工作的其他方向。

8 致谢

我们感谢 Qingyue Jiao、Rohan Sequeira、Sanjana Prabhu、我们的牧羊人（Nina Taft）以及匿名审稿人的贡献和宝贵反馈。本工作由美国国家科学基金会（NSF）资助，资助编号为 CNS-2012001。

[6]我们计算重叠率的方法是，将同时匹配两个指纹的横幅数量除以至少匹配其中一个指纹的横幅数量。

参考文献

[1] [n. d.]. Shodan Search Engine. https://www.shodan.io.

[2] Ahmet Aksoy andMehmetHadi Gunes. 2019. Automated IoT device identification using network traffic. In *IEEE International Conference on Communications*. IEEE, 1-7.

[3] Sandhya Aneja, Nagender Aneja, and Md Shohidul Islam. 2018. IoT device fingerprint using deep learning. In *IEEE International Conference on Internet of Things and Intelligence System*. IEEE, 174-179.

[4] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the Mirai botnet. In *USENIX Security Symposium*. 1092-1110.

[5] Shehar Bano, Philipp Richter, Mobin Javed, Srikanth Sundaresan, Zakir Durumeric, Steven J Murdoch, Richard Mortier, and Vern Paxson. 2018. Scanning the Internet for liveness. *ACM SIGCOMM Computer Communication Review* 48, 2 (2018), 2-9.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877-1901.

[7] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*. Springer, 160-172.

[8] Censys. [n. d.]. Universal Internet BigQuery Dataset. https://support.censys.io/ hc/en-us/articles/360056063151-Universal-Internet-BigQuery-Dataset.

[9] Rajarshi Roy Chowdhury and Pg Emeroylariffion Abas. 2022. A survey on device fingerprinting approach for resource-constraint IoT devices: Comparative study and research challenges. *Internet of Things* (2022), 100632.

[10] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180* (2020).

[11] Nicholas DeMarinis, Stefanie Tellex, Vasileios Kemerlis, George Konidaris, and Rodrigo Fonseca. 2018. Scanning the Internet for ROS: A view of security in robotics research. *arXiv preprint arXiv:1808.03322* (2018).

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[13] Zakir Durumeric, David Adrian, ArianaMirian,Michael Bailey, and J Alex Halderman. 2015. A search engine backed by Internet-wide scanning. In *ACMConference on Computer and Communications Security*. ACM, 542-553.

[14] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, et al. 2014. The matter of heartbleed. In *Internet Measurement Conference*. ACM, 475-488.

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A densitybased algorithm for discovering clusters in large spatial databases with noise.. In *International Conference on Knowledge Discovery and Data Mining*, Vol. 96. 226-231.

[16] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. 2017. Measuring HTTPS adoption on the web. In *USENIX Security Symposium*. 1323-1338.

[17] Xuan Feng, Qiang Li, Haining Wang, and Limin Sun. 2018. Acquisitional rulebased engine for discovering Internet-of-Things devices. In *USENIX Security Symposium*. 327-341.

[18] Talha Javed, Muhammad Haseeb, Muhammad Abdullah, and Mobin Javed. 2020. Using application layer banner data to automatically identify IoT devices. *ACM SIGCOMM Computer Communication Review* 50, 3 (2020), 23-29.

[19] Platon Kotzias, Abbas Razaghpanah, Johanna Amann, Kenneth G Paterson, Narseo Vallina-Rodriguez, and Juan Caballero. 2018. Coming of age: A longitudinal study of TLS deployment. In *Internet Measurement Conference*. ACM, 415-428.

[20] Deepak Kumar, Zhengping Wang, Matthew Hyder, Joseph Dickinson, Gabrielle Beck, David Adrian, Joshua Mason, Zakir Durumeric, J Alex Halderman, and Michael Bailey. 2018. Tracking certificate misissuance in the wild. In *IEEE Symposium on Security and Privacy*. IEEE, 785-798.

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[22] Yang Liu, Armin Sarabi, Jing Zhang, Parinaz Naghizadeh, Manish Karir, Michael Bailey, and Mingyan Liu. 2015. Cloudy with a chance of breach: Forecasting cyber security incidents. In *USENIX Security Symposium*. 1009-1024.

[23] Yair Meidan, Michael Bohadana, Asaf Shabtai, Juan David Guarnizo, Martín Ochoa, Nils Ole Tippenhauer, and Yuval Elovici. 2017. ProfilIoT: A machine learning approach for IoT device identification based on network traffic analysis. In *Proceedings of the Symposium on Applied Computing*. 506-509. [24] Nizar Msadek, Ridha Soua, and Thomas Engel. 2019. IoT device fingerprinting:

Machine learning based encrypted traffic analysis. In *IEEE Wireless Communications and Networking Conference*. IEEE, 1-8.

[25] Rapid7. [n. d.]. Recog: A Recognition Framework. https://github.com/rapid7/ recog/tree/main.

[26] Armin Sarabi and Mingyan Liu. 2018. Characterizing the Internet host population using deep learning: A universal and lightweight numerical embedding. In *Internet Measurement Conference*. ACM, 133-146.

[27] Quirin Scheitle, Taejoong Chung, Jens Hiller, Oliver Gasser, Johannes Naab, Roland van Rijswijk-Deij, Oliver Hohlfeld, Ralph Holz, Dave Choffnes, Alan Mislove, et al. 2018. A first look at certification authority authorization (CAA). *ACM SIGCOMM Computer Communication Review* 48, 2 (2018), 10-23.

[28] Steffen Schneider, Jin Hwa Lee, andMackenzieWeygandt-Mathis. 2023. Learnable latent embeddings for joint behavioural and neural analysis. *Nature* (2023), 1-9.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017).

[30] Juan Wang, Jing Zhong, and Jiangqi Li. 2023. IoT-Portrait: Automatically identifying IoT devices via transformer with incremental learning. *Future Internet* 15, 3 (2023), 102.