arXiv:2601.02512v1 [cs.SE] 5 Jan 2026

# 实践中的绿色大语言模型技术：现有技术在提升工业界基于大语言模型的应用的能源效率方面效果如何?

Pelin Rabia Kuran 阿姆斯特丹自由大学 阿姆斯特丹，荷兰
pelinrkuran@gmail.com

Rumbidzai Chitakunye 阿姆斯特丹自由大学 阿姆斯特丹，荷兰
r.chitakunye@vu.nl

Vincenzo Stoico 阿姆斯特丹自由大学 阿姆斯特丹，荷兰 v.stoico@vu.nl

Ilja Heitlager Schuberg Philis 史基浦-赖克，荷兰
iheitlager@schubergphilis.com

Justus Bogner 阿姆斯特丹自由大学 阿姆斯特丹，荷兰 j.bogner@vu.nl

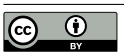
## 摘要

大语言模型（LLM）的迅速普及引发了人们对其高能耗的担忧，尤其是在工业规模部署时。尽管已有多种技术被提出以应对这一问题，但关于将这些技术应用于基于大语言模型的行业应用的效果，目前仍缺乏充分的实证证据。为填补这一空白，我们分析了荷兰IT服务公司 Schuberg Philis 在工业环境中使用的一款聊天机器人应用。随后，我们选取了四种技术，即大小模型协作、提示优化、量化和批处理，并以八种不同配置将其应用于该应用，接着开展实验，研究这些技术对能耗、准确率和响应时间的影响，并与未优化基线进行比较。

结果表明，若干技术（如提示优化 和*2* 位量化）显著降低了能源使用量，有时降幅高达 90%。然而，这些技术对准确率的负面影响尤其明显，严重到无法在实际应用中接受。唯一在大幅降低能耗的同时，未对其他质量指标造成实质性损害的技术是大小模型协作，该技术通过 Nvidia 的提示任务与复杂度分类器（NPCC）并采用提示复杂度阈值实现。这表明，在实践中降低基于大语言模型的应用的能耗并不困难。然而，提高其能源效率，也就是在不损害其他质量指标的前提下降低能源使用量，仍然颇具挑战。本研究为实现这一目标提供了实践见解。

## CCS概念

• **软件及其工程** → **软件设计**；**软件性能**；• **计算方法** → **机器学习**。



本作品依据知识共享署名4.0国际许可协议授权。*ICSE-SEIP '26*，巴西里约热内卢© 2026 版权归所有者/作者所有。ACM ISBN 979-8-4007-2426-8/2026/04
https://doi.org/10.1145/3786583.3786896

## 关键词

绿色机器学习工程、基于大语言模型的系统、能耗、延迟、准确率、受控实验

**ACM参考文献格式：**

Pelin Rabia Kuran、Rumbidzai Chitakunye、Vincenzo Stoico、Ilja Heitlager 和 Justus Bogner。2026。实践中的绿色大语言模型技术：现有技术在提升工业界基于大语言模型的应用的能源效率方面有多有效？载于*2026*年*IEEE/ACM* 第*48*届国际软件工程会议（*ICSE-SEIP '26*），*2026*年*4*月*12*日至*18*日，巴西里约热内卢。ACM，纽约州纽约，美国，12页。
https://doi.org/10.1145/3786583.3786896

## 1 引言

大语言模型（LLM）日益融入我们的日常生活，能够协助完成从基础自动化到复杂问题解决等各种任务 [6]。然而，这一发展也伴随着显著的弊端，尤其是能耗方面的问题 [5]。除了能耗问题，公众广泛使用 ChatGPT 和 Gemini 等在线聊天机器人应用，也可能引发一些组织对数据隐私和安全的担忧 [31]。这可能促使这些组织在更受控的环境中，利用公开可用的大语言模型开发内部聊天机器人系统。

这类内部聊天机器人平台的一个例子是*ChatSBP*，它是本文案例研究对象所使用的大语言模型应用。ChatSBP 由总部位于荷兰的 IT服务公司 Schuberg Philis（SBP）开发，该公司专注于构建关键任务软件。[1] 它集成了最先进的大语言模型，以支持各种语言类任务，从基础文本纠错到复杂问题解决。尽管这些功能具有很高的实用价值，但 ChatSBP 使用量不断增加，不仅开始带来高昂成本，也加剧了人们对 SBP 及其他使用大语言模型的公司的环境影响的担忧。

随着人们对能耗和碳排放的关注日益增加，优化大语言模型推理以实现可持续性已成为一个重要议题 [2]。为此，可复用的绿色技术开始涌现 [15], , 例如模型量化 [21], 批处理 [44],提示优化 [3],以及大小模型协作 [29]。然而，尤其是在真实场景中的基于大语言模型的应用中，这些技术在响应时间和准确率等方面的适用性、有效性及权衡

[1] https://schubergphilis.com

仍未得到充分研究 [5, 32]。

因此，在本案例研究中，我们考察了将选定的能耗优化技术应用于基于大语言模型的工业聊天机器人应用（ChatSBP）时的实际效果。具体而言，我们分析了四种现有能耗优化技术在若干种配置下，降低能耗的效果如何，以及在准确率和延迟这两项大多数工业场景中的关键质量指标上可能存在何种权衡。为此，我们分别将这些技术应用于 ChatSBP，创建了若干个变体，随后在 SBP 的工业实验环境中，对每个变体及未优化基线应用开展了受控实验。

## 2 相关工作

大语言模型在各个领域展现出日益增强的能力 [6], ，尤其是在软件工程领域 [13]。然而，模型规模不断增大，也导致计算资源和能源消耗随之增加。研究表明，规模更大的模型也会消耗更多能源 [5, 11, 16, 18, 34, 38]。为推动绿色人工智能（AI），即准确、环保且包容的人工智能 [36], ，研究人员探索了各种优化技术，以寻求能耗与准确率之间的有利权衡 [5, 11, 34, 38]。大多数研究聚焦于训练阶段，尽管此前的研究已表明，大规模推理消耗的能源更多 [5, 8]。下面我们将介绍一些研究大语言模型推理中单项优化技术的工作。

小型语言模型 （SLM）相较于大语言模型，参数数量较少，仅从数百万到数十亿不等。其规模使其非常适合在资源受限的设备上运行，但可能难以达到大语言模型的准确率 [41]。相关文献主要探讨如何在保持可接受准确率的同时使用小型语言模型，并维持其较低的资源消耗。例如，Kavathekar 等人 [17]研究了小型语言模型通过零样本和少样本提示以及微调来生成函数调用的能力。他们评估了多个小型语言模型的准确率、对提示注入的鲁棒性和资源使用情况。微调模型的表现显著优于零样本和少样本模型。例如，Phi-2 在服务器端达到了 62.4% 的准确率，同时在边缘设备上保持了合理的延迟（~140 秒）。尽管该研究分析了小型语言模型的资源使用情况，但未讨论其能源使用情况，而这正是我们研究所补充的内容。在一项类似研究中，Durán 等人 [10] 对小型语言模型开展了受控实验，涵盖 12 种不同模型、5 种配置以及总计 600 次推理请求，例如基线〈TORCH、CUDA〉设置。他们评估了能源使用量、执行时间和资源使用情况，包括 CPU、内存和 GPU 使用情况。结果表明，所选配置对能耗和执行时间有显著影响。具体而言，采用 CUDA 的配置始终降低了能源使用量和推理时间。尽管他们的研究为小型语言模型的绿色部署提供了宝贵见解，但其测试平台基本上是受控实验环境，而非真实应用场景。相比之下，我们的研究采用系统视角，将小型语言模型集成到部署于 Schuberg Philis 的工业应用 ChatSBP 中。

另一项颇具前景的技术是对推理请求进行批处理 。例如，Argerich 和 Patiño-Martínez [5] 表明，批处理能够最大限度地提高资源利用率并降低能耗。他们研究了模型架构和规模、批次大小以及量化对 15 个大语言模型能源使用量的影响。结果表明，将批量大小增至最大可使所有模型的能源使用量最多降低 20 倍，同时也可能使 GPU 使用率达到满载。能源消耗的降低幅度取决于大语言模型的架构。类似地，Walkowiak[40] 在文本生成中实现了连续批处理，以研究批次大小对能耗的影响。结果显示，将批次大小设为 1 至 150 可降低能源使用量，但会对响应时间产生负面影响。

另一种提出的技术是提示词级优化，即设计更加精确且能结合上下文的提示词，从而在节省时间或计算等其他资源的同时，获得质量更高的响应 [2, 3, 24]。例如，Agarwal 等人 [3] 推出了 PromptWizard 框架，以提升任务性能。PromptWizard 利用反馈驱动的批判来完善提示词和上下文示例，从而自动优化提示词。换个角度来看，Rubei 等人 [34] 研究了提示词中的自定义标签在多大程度上能够降低推理过程中的能耗。他们比较了三种分别采用和未采用自定义标签的提示词工程技术。结果表明，在这三种提示词工程技术中使用自定义标签，均降低了大语言模型的能耗。这两项研究都仅限于提示词工程，重点在于优化和改进现有技术。

量化通过使用更少的位来表示权重和激活值，从而减小模型大小并降低计算需求 [5]。例如，Geens 等人 [11] 通过比较 W4A16 和 W1A32 这两种技术，实现了仅权重量化。结果表明，仅权重量化能够同时降低能耗和延迟。类似地，Husom 等人 [14] 研究了在大语言模型边缘推理中，不同量化级别对能耗、推理速度以及准确率与能源使用量之间权衡的影响。他们发现，量化对能耗和推理延迟均有显著的积极影响，但效果很大程度上取决于具体任务。尽管这两项研究都提供了能耗降低的实证证据，但它们均基于实验开展，且仅关注量化。

总之，Khan 等人 [18] 以及 Argerich 和 Patiño-Martínez [5] 的研究与我们的研究有一些相似之处，因为他们都考虑了不止一种优化技术。Khan 等人 [18]采用后量化和本地推理技术，将大语言模型的能耗降低了 45%。Argerich 和 Patiño-Martínez [5] 测量了模型的大小、架构、批量大小和权重量化如何影响大语言模型文本生成的能耗和延迟。这些研究都只关注一项任务，因此在评估其优化技术时缺乏任务多样性。此外，两项研究所开展的实验均未关注能耗降低与其他质量属性之间的权衡。最后，这两项研究都未充分采用系统视角，也未将相关技术应用于现实世界的应用中。为弥补上述不足，我们将多种优化技术应用于一款现有的工业界应用，以

深入了解这些技术在现实世界中的实用性，并研究能耗与其他质量属性，尤其是响应时间和准确率之间的权衡。

## 3 研究设计

为开始弥合上述研究空白，我们在 VU Amsterdam 与 Schuberg Philis（SBP）之间开展了学术界与业界合作[2]，，后者是一家拥有约450名员工的荷兰软件与IT服务公司。SBP专注于关键任务IT系统，并高度重视数字可持续性。[3] 因此，确定提升其内部应用能源效率的有效技术，不仅是节约成本的有吸引力的手段，更是一项原则性要求。

我们共同设计并实施了一项案例研究 [35]，以 ChatSBP 作为研究的核心对象。我们首先研究了 ChatSBP 的架构和技术，随后确定、讨论并选择了一组用于提升 ChatSBP 能源效率的绿色技术。最后，我们在 SBP 的 Leaplab 中开展了受控实验 [43]，收集有关该应用不同版本的定量证据。Leaplab 是一个配备基于硬件的能耗测量功能的工业实验环境。

我们的研究围绕以下研究问题展开：
**RQ1：** 所提出的优化技术在降低工业大语言模型应用的能耗方面有多大成效？**RQ2：** 所选技术与其他质量属性之间存在哪些权衡？

针对 RQ1，我们通过测量实施绿色技术前后的能耗，研究所选绿色技术在现实环境中的实际成效。我们的目标是评估：在符合现实应用要求的条件下进行评测时，这些技术是否能带来切实的可持续性收益。针对 RQ2，我们从两个重要的质量属性——响应时间和准确率——出发，探讨所应用能源优化技术带来的权衡。对于工业界的聊天机器人应用而言，这两项属性都非常重要；也就是说，即使某项技术能够有效降低能耗，但如果它严重影响准确率和响应时间，那么在许多应用场景中可能仍不适用。最后，我们还研究了这些技术对大语言模型输出令牌数的影响，因为该属性与成本相关；此外，在无法进行直接测量时，它有时也被用作响应时间和能耗的预测指标。

### 3.1 案例研究对象：ChatSBP

ChatSBP 是 SBP 使用的内部聊天机器人应用，为使用大语言模型（包括微调模型）提供了定制化环境。SBP 员工可以通过 Web 用户界面选择公司目前提供的任意大语言模型，以协助软件开发中的编码及其他任务。[4]ChatSBP 的软件架构由基于 OpenWebUI 构建的 Docker化Web应用程序组成。[5] 该应用通常运行在微软 Azure 的云端基础设施上，[6] 位于防火墙和负载均衡器之后。其核心软件架构包含四个与大语言模型推理相关的组件（见图 1）：

- **Chat 容器应用**：用户提交提示词的前端组件。
- **Chat Pipelines 容器应用**：管理应用逻辑和工作流。每条流水线按照预定义的顺序处理用户输入并生成输出。能源优化技术通过此容器集成到软件中，每项技术都作为一条新的流水线实现
- **数据库**：存储聊天记录。
- **Azure OpenAI**：作为大语言模型推理端点，处理用户提示词并生成响应。

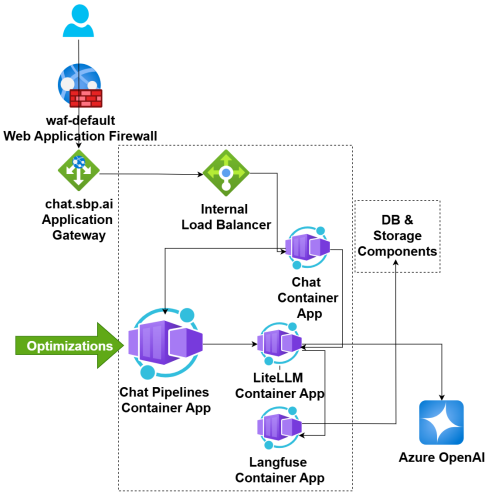



**图 1：ChatSBP 软件架构及我们的优化目标**

在生产实例中，大语言模型推理由 Azure OpenAI API 在外部管理。因此，我们无法直接测量该虚拟化云环境中的能耗。为获得可靠的能耗测量结果，我们将应用程序的所有版本都部署在 SBP 的本地实验环境 Leaplab 中，该环境支持通过专用硬件仪器直接进行功耗监测。我们对 Chat Pipelines 容器应用应用了所有优化，这是唯一经过修改的组件。所有其他组件均保持原状。

### 3.2 能耗优化技术

鉴于机器学习赋能系统有许多可用的绿色技术，我们采用了三步法，为本研究选取数量合适的技术。首先，我们利用 Järvenpää 等人所述的技术集合 [15] 以及近期更多聚焦于大语言模型的出版物（完整论文列表请参见复现包），确定潜在的能源优化技术。其次，我们分析了 ChatSBP 的软件架构，以了解技术约束和集成点。最后，我们

[2]https://schubergphilis.com [3] https://schubergphilis.com/our-esg-sustainability-commitment [4] https://schubergphilis.com/stories/ai-and-automation [5] https://github.com/open-webui/open-webui

[6]https://azure.microsoft.com

**表 1：精选能耗优化技术**

| 技术 | 描述与原理 | 实现变体 |
|---|---|---|
| T1：大小语言模型协作 [29] | 将适当简单的提示词路由到本地的小型语言模型，而不是默认的高能耗大语言模型 | T1AA：NPCC 0.6<br>T1AB：NPCC 0.3<br>T1B：Minion |
| T2：提示优化 [3] | 让提示词更加清晰、简洁，以减少令牌数量，从而降低延迟，并理想地减少能源使用量 | PromptWizard |
| T3：量化 [32] | 降低模型权重表示的精度，使其以更小的位数，这在计算上要求更低，因此理论上需要更少的能源 | T3C：2位模型（Phi-4 14B）<br>T3D：4位模型（Phi-4 14B）<br>T3E：8位模型（Phi-4-mini 3.8B） |
| T4：批处理 [5] | 向模型发送多个批量查询，而不是仅发送单个查询能够实现更好的并行化并优化 CPU/GPU利用率，从而降低每个令牌的延迟，因而理想情况下也能降低能耗 | T4：每批 2 个查询 |

制定并应用了一套纳入和排除标准，根据实际部署需求指导技术选择。

**纳入标准：** (i) 该技术与 ChatSBP 的技术栈兼容；(ii) 该技术能够以合理的工作量实现，例如可以针对另一平台进行定制；(iii) 该技术有望降低聊天功能的能耗，而非其他系统组件的能耗。

**排除标准：** (i) 该技术会为实时聊天机器人应用引入不适合的高计算复杂度；(ii) 该技术没有可复用的实现，从而无法保证可复现性，并且自定义实现可能造成偏差。

SBP 提出的一项重要观点是，令牌数量和 API 调用次数可能会直接影响能耗，也会影响云服务提供商成本 [27, 28]。因此，我们优先考虑能够减少令牌数量、对更大模型的依赖或冗余请求的技术。SBP 的同事还指出，一些低效率做法可能会因不必要的大语言模型调用而增加成本，例如使用含糊或开放式提示（“你好”“帮忙”），或使用礼貌用语（“请”“谢谢”）。经过充分讨论，我们最终选定了四种能源优化技术（见表 1）。所考虑技术的完整列表可在我们的复现包中找到。[7]

首先，小型与大型语言模型协作 （T1）是 ChatSBP 的一项颇具吸引力的技术。小型模型可以在本地处理简单查询，从而减少对能耗更高的大语言模型的不必要调用。此外，可以轻松地将一个决策层集成到 Chat Pipelines Container 中，由其将查询路由至小型模型或大型模型。我们实现了三种不同的 T1 变体：两种实现使用 Nvidia 的提示任务与复杂度分类器（NPCC） [30]，并采用不同的复杂度阈值（0.3 和 0.6）；另一种则使用 Minion 框架 [29] 进行语言模型协作，并采用默认参数。

其次，提示优化 (T2)能够提高准确率，但也有迹象表明，它可能提升能源效率 [2, 4]。对于聊天机器人应用，自动提示优化有望减少低效查询、API 调用和令牌使用量 [3]。为了实施这项技术，我们使用了微软提供的热门 PromptWizard 库[8]，并采用默认参数。

第三，量化 (T3)使用精度较低的数据类型表示大语言模型权重和激活值 [5]，，通常从 16 位开始，最低可降至 1 位 [22]。这能提高计算效率，但也可能降低准确率。尽管如此，量化模型往往仍能保持足够的准确率，从而避免使用更大的模型。量化也能降低能耗，但有些研究认为，低于 4 位的量化无法显著降低能耗 [5, 32]。不过，Chee 等人 [7] 的最新研究表明，2 位量化的效果并非如此。因此，我们纳入了 T3 的三个变体：2 位、4 位和 8 位大语言模型。

最后，批处理 (T4)会并行执行多个提示词的大语言模型推理 [32]，，从而提高资源利用率，理想情况下也能提升能源效率。然而，增大批次大小对能耗的具体影响仍存在争议 [5, 44]。鉴于大多数大语言模型都方便地支持批处理，且在 Chat Pipelines Container 中易于实现，因此我们采用了批量大小为 2 个查询的批处理。

## 3.3 实验材料

*3.3.1* 语言模型。 为实现所有 ChatSBP 变体，我们需要一个小型语言模型、一个大型语言模型以及三种量化语言模型。为更公平地比较各处理方案，我们决定从同一模型系列中选取全部五个模型。作为小模型，我们选择了 Phi4-mini 3.8B；它的表现优于 Llama 3 等更大的模型1 [26]，并且在资源受限环境中，在模型大小、能耗和准确率之间实现了良好平衡。作为默认的大型语言模型，我们选择了 Phi4，这是一种类似 GPT 的纯解码器模型。尽管只有 140 亿个参数，Phi4 的基准测试表现依然出色，超过了一些参数量高达 700 亿的最先进模型 [1]。对于量化模型（2 位、4 位和 8 位），我们采用 Unsloth-AI[9]，这是广受欢迎的量化模型开放提供方。所选模型包括采用 $Q_2$_K 和 $Q_4$_K_M 格式的 Phi-4-GGUF[10]，以及采用 $Q_8$_0 格式的 Phi-4-mini-instruct-GGUF[11]。由于硬件限制，我们在本研究中纳入了 Phi-4-mini（base_s）模型，因为我们的系统无法运行完整 Phi-4 模型的 8 位量化版本。此外，我们还希望观察量化程度提高所带来的影响，并通过比较小模型和大型模型，了解它们在所定义的质量属性方面的差异

[7] https://doi.org/10 16462531 8 5281./zenodo.
https://github.com/微软/PromptWizard

[9] https://unsloth.ai [10]
https://huggingface.co/unsloth/phi-4-GGUF [11]
https://huggingface.co/unsloth/Phi-4-mini-instruct-GGUF

。对于所有模型，我们都将温度设为 0，以便在实验中获得确定性响应 [33]。

*3.3.2* 基准数据集。由于 SBP 没有针对其工作相关查询的定制基准测试，我们转而为实验选择了两个不同的数据集。我们的目标是纳入彼此差异很大的任务，以了解技术的有效性是否取决于特定的任务类型，同时也使用一些在结构上与 SBP 所用查询具有相似性、尽管并非来自同一领域的任务。对于推理任务，我们选择了*GSM8K*（小学数学8K） [9], , 其中包括解决算术问题或结合逻辑推理与数学运算的文字题。这些高难度练习用于测试模型围绕需要数学运算、逻辑思维和多步推理的问题进行推理并加以解决的能力。此外，对于问答导向提示词，我们选择了 *MMLU*（大规模多任务语言理解） [12], , 这是最常用于评估模型针对选择题生成准确且相关答案的数据集之一。

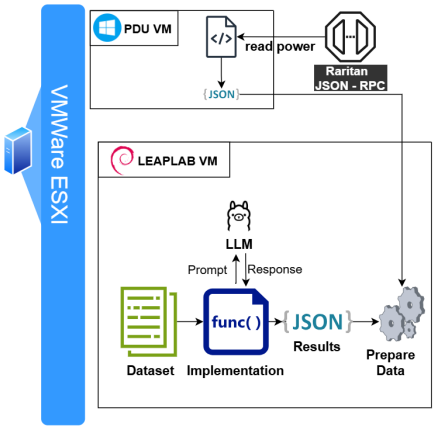



**图 2：实验基础设施（Leaplab）**

*3.3.3* 实验基础设施。 为全面掌控实验环境，我们使用了 SBP 的 Leaplab，而非生产实例所用的 Azure云环境（见图 2）。该环境采用由 Intel(R) Xeon(R) Gold 6154 CPU @ 3.00 GHz 服务器提供支持的 VMware ESXi虚拟机管理程序 [12]。我们使用了两台虚拟机：一台实验虚拟机（Leaplab VM）和一台配电单元虚拟机（PDU-VM）。实验虚拟机是 ESXi 上唯一运行的工作负载，并使用专用 GPU。我们使用 PDU插座读取功率，无需第三方工具即可轻松将其转换为能耗数据。每台虚拟机使用各自独立的 PDU插座，以避免相互干扰测量。能耗数据通过 Raritan JSON-RPC API 获取。PDU-VM 配备 4 GB RAM，运行 Windows Server 2019 Standard（64 位）。Leaplab VM 运行 Debian 12，配备 32 GB RAM，并搭载一块具有 16 GB RAM 的 Nvidia A2 Tensor Core GPU。最后，我们使用 Ollama框架[13] v0.6.6 部署实验所用的大语言模型。

[12]https://www.vmware.com/products/cloud-infrastructure/vsphere [13] https://ollama.com

## 3.4 实验设计与执行

作为自变量，我们的实验纳入了优化技术及其所有实现变体（共 8 种，如表 1 所示，另加一个未经优化的基线），以及两种不同的基准数据集（GSM8k 和 MMLU）。我们的因变量包括系统能耗（瓦时）、准确率、响应时间（毫秒）和输出令牌数。我们通过 Raritan / Server Technology XerusTM PDU 接口，使用 JSON-RPC API.[14]测量系统能耗。由于基准数据集中的问题都只有一个正确答案，因此准确率采用简单的字符串匹配算法进行测量。响应时间能否作为黑盒模型能耗的指标仍有争议。有人认为，基于大语言模型的系统中响应时间越长，能耗就总是越高 [2, 32], , 也有人持不同看法 [5]。在受控环境中同时收集这两项指标，有助于深入了解它们之间的关系；但响应时间本身也是一项有价值的指标：响应时间增加过多的技术在实际应用中可能并不实用，尤其是在聊天机器人场景中。最后，作为一种偏向解释性的变量，输出令牌数通过 HuggingFace AutoTokenizer类进行操作化定义。[15] 收集输出令牌数能提高性能分析的粒度，并有助于在解释不同技术之间的差异时获得更多见解。

为评估优化技术对能耗的影响（RQ1），我们提出了以下假设对，其中 $e$ 表示能耗， $x$ 表示一种特定技术，$\mu_x^e$ 表示技术 $x$对应的能耗中位数：

$$H_0^x : \mu_x^e \geq \mu_{\text{baseline}}^e$$

$$H_1^x : \mu_x^e < \mu_{\text{baseline}}^e$$

对于每种评估的技术，零假设为：实施该技术的系统变体的能耗中位数大于或等于基线系统的能耗。我们使用中位数而非均值，因为预计各组数据不服从正态分布。因此，备择假设为：该技术的能耗中位数低于基线。对于 RQ2，我们采用目视检查和描述性统计，因此未提出正式假设。

由于两个自变量均作为因素，我们采用全因子设计 [43] 来确定实验配置。这产生了 8 个技术实现和一个基线，即 9 种系统变体 $\times$ 2 数据集 = 18 实验配置。在执行实验之前，我们终止了非必要的后台进程，以尽量减少其对因变量的潜在测量影响。为提高测量可靠性，每个实验配置均重复 5 次，共进行了 90 次实验试次。为了使实验环境的温度恢复到更接近初始状态，我们还在每次试次之间设置了五分钟的冷却时间。此外，我们对处理试次的安排进行了随机化，以避免可能对单个处理方案产生不成比例影响的短暂混杂因素。完成全部 90 次试次大约耗时 24 小时。

[14]https://help.raritan.com/json-rpc/4.3.0/index.html [15] https://huggingface.co/docs/transformers/main/en/model_doc/auto#transformers.AutoTokenizer

### 3.5 数据分析

在统计评估中，我们首先使用 Q-Q图进行图形评估 [20]，并使用 Shapiro-Wilk检验进行统计评估 [37]，以检查数据是否服从正态分布。Q-Q图中偏离预期对角线，以及 Shapiro-Wilk检验中的 p值低于 0.05，均表明结果存在非正态性。我们还使用箱线图和描述性统计，以更好地了解不同处理方法之间的差异。我们观察到各组数据并不服从正态分布，因此采用非参数方法进行假设检验，即 Mann-Whitney U 检验 [25],，分别针对每个数据集比较基线组与处理组之间的能耗。我们选择了单侧检验，以检测某项技术是否导致能耗显著下降。在进行假设检验之前，我们采用默认因子为 1.5 的四分位距（IQR）法，从能耗数据中筛除潜在的离群值 [39]。这减轻了实验环境中因不希望出现的副作用而产生的潜在极端值的影响；这些极端值可能使结果发生偏移，并增加可视化结果的解读难度。由于我们总共进行了 18 次假设检验，因此采用 Holm-Bonferroni校正来调整 p值并控制家族错误率 [42]。最后，我们通过非参数 Cliff's $\delta$ 计算效应量，以评估显著效应的强度 [23]。

## 4 结果

本节按照研究问题的结构呈现研究结果。为确保透明度和可复现性，我们在 Zenodo 上分享所有制品、数据集、代码库和实验数据。[7]

**表 2：每个提示和实验配置的能耗（瓦时）描述性统计** （基线$_s$: Phi4-mini，仅与 T3E 进行比较；由于硬件限制，缺少 T4 MMLU）

| 数据集 | 技术 | Mean | STD | Min | 中位数 | Max |
|---|---|---|---|---|---|---|
| GSM8k | T1AA | 0.16 | 0.02 | 0.11 | 0.16 | 0.20 |
| | T1AB | 0.23 | 0.02 | 0.16 | 0.23 | 0.30 |
| | T1B | 4.56 | 0.26 | 3.75 | 4.55 | 5.40 |
| | T2 | 0.10 | 0.003 | 0.07 | 0.10 | 0.13 |
| | T3C | 0.53 | 0.04 | 0.44 | 0.53 | 0.65 |
| | T3D | 0.67 | 0.04 | 0.56 | 0.67 | 0.80 |
| | T3E | 0.56 | 0.03 | 0.49 | 0.55 | 0.63 |
| | T4 | 0.65 | 0.008 | 0.63 | 0.65 | 0.67 |
| | 基线 | 0.59 | 0.02 | 0.51 | 0.59 | 0.68 |
| | 基线$_s$ | 0.13 | 0.005 | 0.11 | 0.13 | 0.14 |
| MMLU | T1AA | 0.05 | 0.01 | 0.04 | 0.05 | 0.08 |
| | T1AB | 0.07 | 0.01 | 0.05 | 0.06 | 0.11 |
| | T1B | 6.13 | 0.27 | 5.28 | 6.12 | 7.00 |
| | T2 | 0.08 | 0.003 | 0.07 | 0.08 | 0.09 |
| | T3C | 0.55 | 0.03 | 0.46 | 0.55 | 0.65 |
| | T3D | 0.43 | 0.02 | 0.35 | 0.43 | 0.51 |
| | T3E | 0.32 | 0.02 | 0.26 | 0.32 | 0.38 |
| | T4 | – | – | – | – | – |
| | 基线 | 1.11 | 0.05 | 0.98 | 1.11 | 1.25 |
| | 基线$_s$ | 0.06 | 0.01 | 0.04 | 0.06 | 0.08 |

### 4.1 降低能耗（RQ1）

针对 RQ1，我们旨在研究这八种技术变体与未优化基线相比，在降低能耗方面的效果。图 3 分别展示了基线和每种选定技术在各个数据集上的能耗分布。此外，表 2 列出了相应的描述性统计数据。按每条提示词计算的均值能耗约为 0.1 瓦时至 6 瓦时，这表明不同实验配置之间存在显著差异。与基线相比，有几种技术在两个基准测试中都显著降低了能耗，分别是两种 NPCC 配置（T1AA 和 T1AB）、PromptWizard（T2）和 2 位量化（T3C）。其他技术，例如 4 位量化（T3D），则仅在 MMLU 数据集上有效。最后，Minion（T1B）、批处理（T4）和 8 位量化（T3E）（与小型基线进行比较）均未能降低能耗，其中 T1B 甚至导致能耗显著增加。对于使用 MMLU 数据集的批处理（T4），我们的硬件配置无法处理两个合并提示词的上下文长度（见第 3.3.3 节），因此表 2 中该行显示的是短横线，而非统计数据。

我们使用 Mann-Whitney U检验，确认这些观察到的差异相对于基线是否具有统计显著性。表 3 展示了检验结果，包括经 Holm-Bonferroni 校正的 p 值、效应量以及中位数的百分比差异。T1AA、T1AB、T2 和 T3C 的能耗降幅最为显著，在两个数据集中均具有统计显著性，且效应量较大（Cliff's $\delta$ 介于 0.77 和 1.0 之间），表明与基线相比节能效果显著。尤其是 PromptWizard（T2）和 NPCC 配置（T1AA 和 T1AB）等技术，在推理数据集（GSM8k）上至少降低了 60% 的能耗，在问答数据集（MMLU）上则至少降低了 92%。

就单项技术而言，大小语言模型协作 技术（T1）的效果很大程度上取决于具体的实现变体。Minion（T1B）使两个数据集的能耗均大幅增加，增幅超过 450%；这主要是因为小模型与大模型之间进行了多次消息交换，以及小模型在若干情况下生成了过多令牌。然而，NPCC 的不同变体则属于效果最显著的节能技术。将提示复杂度阈值设为 0.6（T1AA），与 T1AB 的 0.3 阈值相比，能耗略低。例如，GSM8k 中的能耗中位数分别为 0.16 瓦时和 0.23 瓦时。阈值越高，能耗较高的大语言模型被调用的频率就越低。在推理数据集（GSM8k）中，T1AB 有 72% 的情况下使用小模型，而 T1AA 中这一比例提高到了 86%。在问答数据集（MMLU）中，T1AA 和 T1AB 使用小模型的比例均约为 98%，这解释了为何此次两者的能耗中位数更为接近（0.05 瓦时和 0.06 瓦时），也解释了为何 NPCC 技术在 MMLU 中的效果甚至优于在 GSM8k 中的效果。

通过 PromptWizard 实现的 提示优化技术（T2）在两个数据集上都取得了最高的平均能源降幅：GSM8k 为 82%（最高），MMLU 为 92.5%（排名第 3，但前三名非常接近，介于 92.5% 和 93.7% 之间）。它

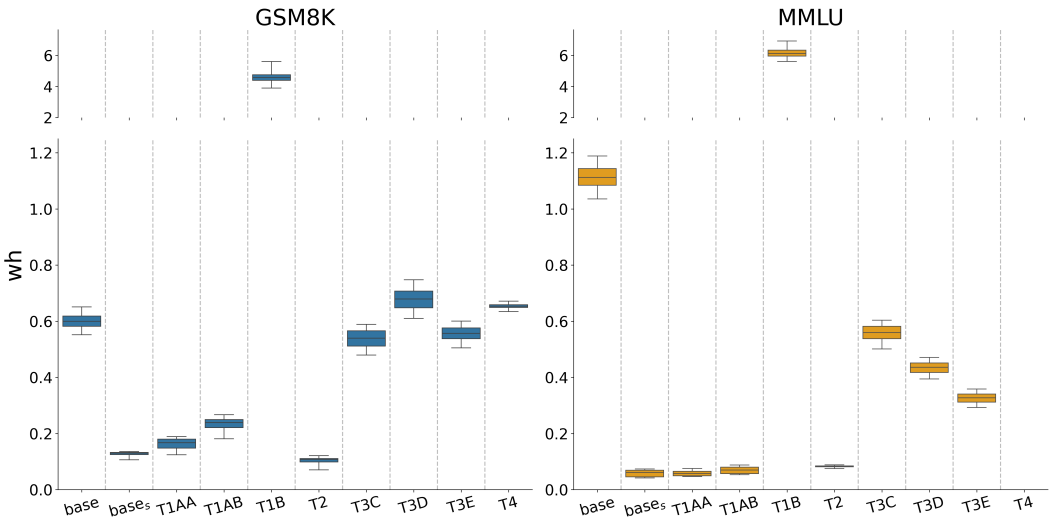



**图 3：每种处理方案每次提示的能耗（瓦时）**（基线：*Phi4*，基线$_s$：*Phi4-mini*（仅与 T3E 比较），T1AA：*NPCC 0.6*，T1AB：*NPCC 0.3*，T1B：*Minion*，T2：*PromptWizard*，T3C：*2* 位量化，T3D：*4* 位量化，T3E：*8* 位量化，T4：批处理，大小为 *2*）

**表 3：通过 Mann-Whitney U 检验进行的能耗基线比较**（经 Holm-Bonferroni 校正的 p 值、基于 Cliff's $\delta$ 的显著效应量、百分比差异中位数）

| 数据集 | 技术 | p 值 | Cliff 的 $\delta$ | 百分比变化 |
|---|---|---|---|---|
| GSM8k | T1AA | < 0.001 | 1.00 | -72.1% |
| | T1AB | < 0.001 | 1.00 | -60.0% |
| | T1B | 1.0 | – | 660.1% |
| | T2 | < 0.001 | 1.00 | -82.0% |
| | T3C | < 0.001 | 0.77 | -9.9% |
| | T3D | 1.0 | – | 13.3% |
| | T3E | 1.0 | – | 327.0% |
| | T4 | 1.0 | – | 9.16% |
| MMLU | T1AA | < 0.001 | 1.00 | -94.8% |
| | T1AB | < 0.001 | 1.00 | -93.7% |
| | T1B | 1.0 | – | 450.1% |
| | T2 | < 0.001 | 1.00 | -92.5% |
| | T3C | < 0.001 | 1.00 | -49.7% |
| | T3D | < 0.001 | 1.00 | -60.8% |
| | T3E | 1.0 | – | 429.1% |
| | T4 | – | – | – |

在更复杂的 GSM8k 数据集上，每个提示的能源消耗中位数（0.10 瓦时）和最大值（0.13 瓦时）也最低，这进一步证明了其有效性。

关于量化 (T3) 的结果则更为复杂。基座模型是通过 Ollama 加载的 4 位量化 Phi-4 模型；作为比较，我们还评估了上文提到的、通过 Unsloth 获取的同一模型的 4 位量化版本。尽管这两个模型采用了相同的量化方案，但可能由于架构和核心特性的差异，测得的能耗并不相同。遗憾的是，由于硬件限制，我们无法在自己的硬件上运行 8 位精度的 Phi4 模型因此，我们改用较小的 8 位 Phi-4-mini 模型。为了更公平地与基线进行比较，我们还使用未量化的标准 Phi-4-mini 模型作为第二个较小的基线（baseline$_s$）。总体而言，T3 在降低推理任务（GSM8k）的能耗方面并不十分成功。2 位量化（T3C）的能耗中位数约降低了 10%，这一结果具有统计显著性，Cliff's $\delta$ 为 0.75；而 4 位量化（T3D）和 8 位量化（T3E，后者与 baseline$_s$比较）实际上都增加了能耗。这些推理任务的复杂度可能并不适合通过量化来优化能源。不过，对于问答任务（MMLU），2 位和 4 位量化均显著降低了能耗，降幅较大（Cliff's $\delta$ 为 1.0），分别约为 50% 和 60%。只有 8 位量化（T3E）再次导致能耗大幅增加，与较小的基线$_s$ 相比约增加了 430%。

最后，批处理 (T4) 是最难实施的方法之一，因为 Leaplab 存在硬件限制。虽然我们能够对推理数据集（GSM8k）使用批量大小为 2 的设置，但问答数据集（MMLU）的输入序列过长，现有硬件无法支持批处理。因此，我们仅针对 GSM8k 进行了分析。对于该数据集，这项技术并未奏效，反而使能耗增加了约 9%。一个可能的原因是，GPU 平均利用率仅从 95.5% 小幅提升至 96.88%。如果基线中的 GPU 利用率较低，或者批次大小更大，这项技术可能会更有效；但我们的硬件无法支持这些条件。

## 4.2 与其他质量属性的权衡（RQ2）

对于 RQ2，我们希望了解这些技术如何影响重要质量属性的准确率和响应时间，以及输出令牌数在这些关系中发挥了什么作用。

*4.2.1* 准确率。 我们在图 4 中展示了每种实验配置和数据集对应的准确率。总体而言，对于 GSM8k 推理数据集，所有技术都至少导致了一定程度的准确率下降（其中一些下降幅度很大）；而对于问答任务（MMLU），准确率下降则不太明显。值得注意的是，PromptWizard（T2）是降低能耗最有效的技术之一，但同时也导致两个数据集的准确率显著下降。与基线相比，GSM8k 的准确率从 0.92 降至 0.42，MMLU 则从 0.57 降至 0.35。2 位量化（T3C）也呈现出类似的效果，其准确率降至 GSM8k 的 0.25 和 MMLU 的 0.48。相反，*Minion*（T1B）对能耗产生了迄今为止最大的不利影响，但其准确率结果却是所有技术中最好的：GSM8k 为 0.81，MMLU 为 0.59，甚至比基线提高了 0.02。然而，NPCC 配置（T1AA 和 T1AB）同样是最有效的能耗降低技术之一，但其在 GSM8k 上的准确率仅略有下降（分别为 0.76 和 0.77），在 MMLU 上则完全没有降低（0.57），凸显了其在绿色人工智能实践中的价值。最后，4 位量化（T3D）、8 位量化（T3E）以及批处理（T4）的准确率下降也不太明显，但这些技术在降低能耗方面同样没有效果。

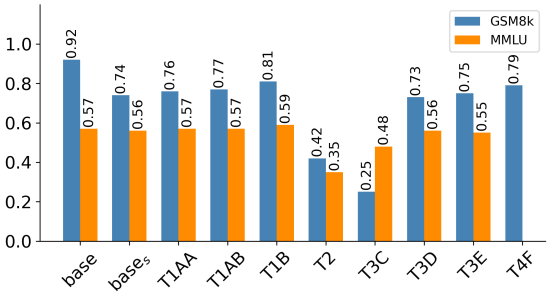



**图4：GSM8k 和 MMLU 数据集上的准确率**（基线：Phi4-mini，仅与 T3E 比较）

*4.2.2* 响应时间。我们在图5中展示了各种技术相较于基线的响应时间分布，并在表4中列出了相应的描述性统计数据。大多数处理方法都显著缩短了响应时间，并使分布集中在较低值附近。例如，PromptWizard（T2）的响应时间明显向接近零的方向移动（GSM8k 的中位数为 1.28 秒），优于响应时间分布范围宽得多的基线（中位数为 15.75 秒）。类似地，与基线相比，NPCC 0.3（T1AA）和 NPCC 0.6（T1AB）的响应时间分布也有所改善。对于三种量化技术（T3C、T3D、T3E），我们观察到其分布相较于基线略微左移或几乎相同。不过，这些技术仅在 MMLU 基准测试中降低了响应时间中位数，在 GSM8k 上则没有。批处理（T4）使响应时间中位数较基线略有下降（从 15.75 秒降至 13.35 秒）。最后，Minion 技术（T1B）导致响应时间大幅增加，其分布大多落在每个请求 100 至 300 秒之间。

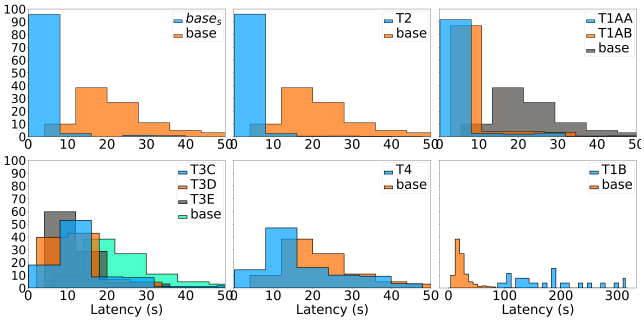



**图5：两个数据集合并后各实验配置的响应时间分布（秒）**

**表4：每个提示的响应时间（秒）和实验配置的描述性统计**
（基线$_s$：Phi4-mini，仅与 T3E 对比）

| 数据集 | 技术 | Mean | STD | Min | 中位数 | Max |
|---|---|---|---|---|---|---|
| GSM8k | T1AA | 3.25 | 2.06 | 0.42 | 2.60 | 10.63 |
| | T1AB | 3.81 | 3.12 | 0.42 | 2.64 | 15.00 |
| | T1B | 137.8 | 42.8 | 93.7 | 126 | 234.4 |
| | T2 | 1.62 | 0.89 | 0.99 | 1.28 | 4.46 |
| | T3C | 24.33 | 0.89 | 23.49 | 23.87 | 25.79 |
| | T3D | 25.51 | 2.16 | 20.66 | 25.74 | 28.26 |
| | T3E | 18.05 | 10.88 | 1.05 | 23.70 | 28.26 |
| | T4 | 16.43 | 8.73 | 5.18 | 13.35 | 36.88 |
| | 基线 | 17.00 | 7.13 | 4.52 | 15.75 | 36.76 |
| | 基线 $_s$ | 3.07 | 1.67 | 0.12 | 2.88 | 7.33 |
| MMLU | T1AA | 0.45 | 0.24 | 0.22 | 0.34 | 1.30 |
| | T1AB | 0.44 | 0.23 | 0.22 | 0.34 | 1.27 |
| | T1B | 199 | 71.4 | 97.7 | 184.6 | 318.5 |
| | T2 | 1.41 | 0.25 | 0.94 | 1.38 | 2.20 |
| | T3C | 13.23 | 6.00 | 6.26 | 11.93 | 32.93 |
| | T3D | 9.83 | 4.23 | 3.83 | 8.73 | 25.25 |
| | T3E | 7.35 | 4.58 | 0.99 | 6.49 | 23.06 |
| | T4 | – | – | – | – | – |
| | 基线 | 29.91 | 24.62 | 3.76 | 21.77 | 105 |
| | 基线$_s$ | 0.45 | 0.24 | 0.22 | 0.34 | 1.28 |

*4.2.3* 输出令牌数。如图6所示，输出令牌数的中位数和最大值因数据集和处理方案而异。除批处理（T4）外，基线在所有技术中每个提示的令牌数最高；批处理（T4）通过合并查询，自然会使每个提示包含更多令牌。虽然两种数据集的令牌数中位数相近，但对于大多数技术，问答数据集（MMLU）的令牌数都显著较低。此外，与基线相比，并非所有能减少最终答案令牌数的技术，也都能有效降低系统的总体能耗。例如，能耗最高的处理方案 Minion（T1B）生成的最终答案最短。然而，在小模型和大模型交互过程中生成了大量令牌，而这些令牌未纳入我们的分析。尽管如此，

这说明，令牌数量并非总是能耗的可靠替代指标，尤其是在包含多个相互通信的大语言模型组件的复杂分布式系统中。

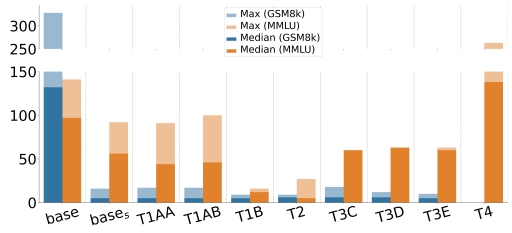



**图 6：每个提示的最大值和中位数输出令牌数**

*4.2.4* 因变量之间的关系。最后，我们还分析了所研究系统变体中因变量之间的关系（见图 7）。关于响应时间和能耗（底行），两个数据集最初都呈现出近乎线性的关系。然而，这种关系受到 Minion（T1B）结果的严重影响。将 T1B 作为离群值剔除后（右下图），我们看到两个数据集呈现出截然不同的模式。问答数据集（MMLU）仍呈现近乎线性的趋势，而推理数据集（GSM8k）则表现出显著波动。这表明，响应时间也并非总是能耗的可靠替代指标。例如，某项技术可能会导致响应时间中位数上升（如 T3C 在 GSM8k 上的表现），但能耗仍可能显著降低。

两个数据集中的响应时间与输出令牌数（右上）也呈现出明显不同的模式。问答数据集（MMLU）中，除前文提到的 Minion 离群值外，响应时间大多随着令牌数量增加而上升。相比之下，GSM8k 图表中的变化幅度很大：即使令牌数量相近，响应时间仍会显著上升或下降。因此，很难明确确定这两个质量属性之间的关系。

最后，能耗与输出令牌数（左上）之间的关系同样难以描述。在两个数据集中，图表都难以预测，出现了多次骤升或骤降；有时令牌数量相当稳定，能耗却不断增加。这进一步印证了我们先前的观察：在复杂的基于大语言模型的应用中，令牌数量也不是能耗降低技术完全可靠的预测指标。

## 5 讨论

在本节中，我们总结了研究结果的主要发现，以及这些结果对实践者和研究人员的启示。在分析**能耗**结果时，我们发现，所选优化方法中的一些方法（如 NPCC 技术、提示优化或 2 位量化）显著降低了能耗，而另一些方法则导致能耗大幅增加。具体而言，我们观察到，Minion（T1B）技术的能耗和响应时间最高。这主要是由于小模型与大模型之间的通信效率低下，每次迭代至少需要两轮通信。此外，我们还观察到，

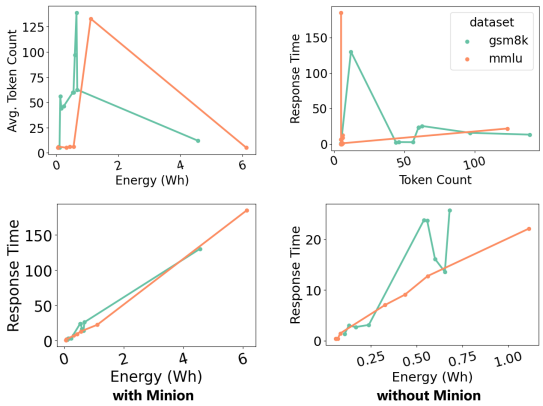



**图 7：输出令牌数、响应时间与能耗之间的关系**

在某些情况下，小模型会针对任务生成过多的词元。例如，即使温度参数设为 0，一个仅含 19 个词元的输入提示词也会导致小模型为大模型生成超过 10,000 个词元。Narayan 等人 [29] 也提到了类似问题，并将其视为该技术的缺点；而我们观察到，这还会大幅增加能耗。根据这些发现，限制模型生成的词元数量并尽量减少模型之间的通信轮次至关重要。

当**同时考虑准确率和能耗时**，通过 NPCC 技术（T1AA、T1AB）实现的大小模型协作脱颖而出，成为能源效率最高的优化方法。T1AA 和 T1AB 在大幅降低能耗的同时，还取得了 GSM8k 上第三和第四高的准确率，在 MMLU 上的准确率甚至与基线持平。这表明，针对适当的查询动态使用较小的模型，并不一定会导致系统准确率大幅下降。因此，我们建议将这项技术作为提升基于大语言模型的应用能源效率的一种实用方法。一个尚待解决的问题是，如何针对特定的提示配置找到理想的提示复杂度阈值。虽然我们的研究纳入了两个阈值（0.3 和 0.6），但在这一方向开展更细致的实验，或提供可操作的指导或工具，帮助从业者为其应用选择阈值，是未来研究的一个有前景的方向。

遗憾的是，结果也表明**准确率与能耗之间假定的权衡关系在许多情况下仍然成立**；例如，能耗最高的处理方案（如 Minion（T1B）和基线）也取得了最高的准确率。此外，PromptWizard（T2）和 2 比特量化（T3C）显著且大幅降低了能耗，但准确率下降幅度难以接受，因此实际应用价值有限。虽然 PromptWizard 能有效提高提示词效率，但有时会删除关键信息，导致准确率显著下降。这些发现表明，仅实现最大幅度的能耗降低，还不足以满足工业界的应用需求。

最后，我们的结果表明，批处理（T4）对任何因变量都没有显著影响，但也并非

一种降低能耗的有效技术，甚至还增加了能耗。不过，硬件限制使我们只能有限地探索该技术的实现变体。批处理预期带来的 GPU 利用率提升未能如预期般实现，因为基线利用率已经超过 95%。因此，GPU 分配的小幅增加不足以降低能耗。这可能意味着，批处理在能源优化方面的效果高度依赖可用硬件资源和基线利用率水平。

**主要结论：**

- Dynamic routing 通过根据提示复杂度将查询路由至较小的模型，可以在许多任务中降低能耗，同时不降低准确率（T1A）。大模型并非总是必需的。- **但是：** 小模型与大模型之间反复通信，以及过量的令牌生成，会同时增加能耗和响应时间（T1B）。- 提示优化（T2）以准确率换取能耗降低，有时会造成不可接受的准确率损失。- 优化技术的成效和效果取决于数据集及所用硬件。- 虽然降低基于大语言模型的应用的能源使用量很容易，**但提高能源效率仍然困难**，也就是说，要在不损害其他质量属性的前提下降低能源使用量。

## 6 效度威胁

本节根据 Wohlin 等人 [43] 提出的类别，讨论本研究的效度威胁。

内部效度：硬件资源的限制可能导致测量出现偏差。事实上，在测试批处理配置时，我们发现部分提示词超出了内存限制。我们通过异常处理机制处理了这些情况，从而减轻无效试验对结果的影响。此外，每项试验均重复五次。试验的执行顺序经过随机化，并且每次运行之间设置了五分钟的冷却时间，以消散累积的能源，并在下一次执行前恢复系统状态。此外，我们的实验环境使用了虚拟机监控程序 VMware ESXi，这可能会给能耗测量带来额外开销。为减轻此类开销对能源数值的影响，我们测量了部署 ChatSBP 的整台机器。此外，我们确保测试平台仅运行 ChatSBP，不运行其他资源密集型进程。尽管这些措施应当能够避免实验环境中的重大混杂因素，但仍可能出现轻微的随机波动。我们确信，这不会改变分析结论。

外部效度：我们的实验测试平台（LeapLab）未能充分反映大规模工业部署中真实运行环境的复杂性。其主要局限之一在于硬件资源不足，无法运行大规模人工智能工作负载。此外，我们的研究仅针对一个工业案例 ChatSBP 开展，未能代表人工智能驱动的工业工具的全貌。不过，我们的研究旨在展示实证软件工程在实践中的影响，并说明如何将其应用于工业环境。同样，优化技术的选择受到其对 ChatSBP 的适用性以及与员工所面临具体问题的相关性的限制。因此，我们无法充分保证这些技术在其他情境中的适用性或相关性。尽管如此，所探讨的问题在现代人工智能驱动的应用中十分常见：在将数据发送到部署于云端的人工智能模块之前，通常需要对数据进行预处理。最后，尽管所选数据集中的任务可能无法涵盖现实工业应用中的全部复杂情况，但选择这些任务是因为它们能够代表典型场景，使我们得以探究所评估的技术是否有望提高 ChatSBP 的能源效率。

结论效度：我们针对非正态分布数据选用了适当的统计方法。此外，我们还应用 Holm-Bonferroni 校正 [42]，以尽量减少多重比较导致的 I 类错误。这进一步增强了研究结果的效度。

## 7 结论

本研究系统评估了多种能耗优化技术在工业聊天机器人应用中的有效性。通过实施并分析大小模型协作、提示优化、量化和批处理等技术，我们证明，大多数技术都能显著降低基于大语言模型的应用的能耗。然而，我们也指出了在能耗与其他质量属性（尤其是准确率）之间取得平衡的挑战。虽然一些有效技术（例如提示优化）会使准确率降至不可接受的水平，但通过 NPCC 实现的大小模型协作能够在保持准确率和响应时间处于可接受水平的同时，大幅降低能耗，因此是值得工业界采用的首选方案。

我们的研究结果还揭示了技术环境和结果普适性方面的局限。工业部署的复杂性，包括硬件限制、多样的工作负载模式，甚至不同的模型，都可能影响任何能耗优化技术的效果。因此，未来工作应在更广泛的大语言模型架构、应用领域和生产环境中评估优化技术，以获得更多见解和指导。许多已提出的技术，例如优化高能耗提示关键词 [2]或使用生成指令 [19]，都颇具前景，但仍需进一步研究，以更好地了解潜在权衡。此外，采用更合适的硬件资源实现连续批处理等新方法，也有望进一步揭示批处理对能源效率的影响。在绝对真实世界环境中研究用户体验指标，也将有助于推动这些能耗优化技术的实际应用。通过谨慎平衡能耗与其他质量属性，该领域有望进一步提升能源效率。为支持未来的研究和复现，我们在线公开了研究制品。[7]

## 致谢

我们谨向 Cornelis Mol 致谢，感谢他协助改进本研究的可视化效果，这对提升报告的呈现效果发挥了重要作用。

# 参考文献


[1] Marah Abdin、Jyoti Aneja、Harkirat Behl、Sébastien Bubeck、Ronen Eldan、Suriya Gunasekar、Michael Harrison、Russell J. Hewett、Mojan Javaheripi、Piero Kauff- mann、James R. Lee、Yin Tat Lee、Yuanzhi Li、Weishung Liu、Caio C. T. Mendes、Anh Nguyen、Eric Price、Gustavo de Rosa、Olli Saarikivi、Adil Salim、Shital Shah、Xin Wang、Rachel Ward、Yue Wu、Dingli Yu、Cyril Zhang 和 Yi Zhang。2024。Phi-4 2412 08905 [ ] 2412 08905 技术报告。arXiv: . cs.CL https://arxiv.org/abs/ 。[2] Marta Adamska、Daria Smirnova、Hamid Nasiri、Zhengxin Yu 和 Peter Gar- raghan。2025。绿色提示。arXiv:2503.10666 [cs.CL] https://arxiv.org/abs/ 2503.10666[3] Eshaan Agarwal、Joykirat Singh、Vivek Dani、Raghav Magazine、Tanuja Ganu 和 Akshay Nambi。2024。PromptWizard：任务感知提示优化框架。arXiv: 2405 18369 [ ] 2405 18369 . cs.CL https://arxiv.org/abs/ 。[4] Radu Apsan 、 Vincenzo Stoico 、 Michel Albonico 、 Rudra Dhar、Karthik Vaid- hyanathan 和 Ivano Malavolta。2025。借助大语言模型生成节能代码——现状如何？ *arXiv* 预印本 *arXiv:2509.10099* (2025)。[5] Mauricio Fadel Argerich 和 Marta Patiño-Martínez。2024。衡量并提升大语言模型推理的能源效率。*IEEE Access* 12 (2024), 80194–80207。doi:10.1109/ACCESS.2024.3409745[6] Yupeng Chang、Xu Wang、Jindong Wang、Yuan Wu、Linyi Yang、Kaijie Zhu、Hao Chen、Xiaoyuan Yi、Cunxiang Wang、Yidong Wang、Wei Ye、Yue Zhang、Yi Chang、Philip S. Yu、Qiang Yang 和 Xing Xie。2024。大语言模型评估综述。*ACM*智能系统与技术汇刊 15, 3 (June 2024 1 45 10 1145 3641289 ), – 。doi: . / Jerry Chee、Yaohui Cai、Volodymyr Kuleshov 和 Christopher M De Sa。2023。QuIP：具有保证的 2 位大语言模型量化。载于神经信息处理系统进展，A. Oh、T. Naumann、A. Globerson、K. Saenko、M. Hardt 和 S. Levine（编），第 36 卷。Curran Associates, Inc.，4396–4429。https://proceedings.neurips.cc/paper_files/paper/2023/file/0df38cd13520747e1e64e5b123a78ef8-Paper-Conference.pdf[8] Andrew A Chien、Liuzixuan Lin、Hai Nguyen、Varsha Rao、Tristan Sharma 和 Rajini Wijayawardana。2023。降低生成式人工智能推理（当下及 2035 年）的碳影响。载于第二届可持续计算机系统研讨会论文集（美国马萨诸塞州波士顿）*(HotCarbon'23)*。计算机协会，美国纽约州纽约市，文章 11，共 7 页。doi:10.1145/3604930.3605705[9] Karl Cobbe、Vineet Kosaraju、Mohammad Bavarian、Mark Chen、Heewoo Jun、Lukasz Kaiser、Matthias Plappert、Jerry Tworek、Jacob Hilton、Reiichiro Nakano、Christopher Hesse 和 John Schulman。2021。训练验证器以解决数学应用题。arXiv: 2110 14168 [ ] 2110 14168 . cs.LG https://arxiv.org/abs/ 。[10] Francisco Durán、Matias Martinez、Patricia Lago 和 Silverio Martínez-Fernández。2024。使用运行时引擎和执行提供程序提供代码小型语言模型服务的能耗。*arXiv* 预印本 *arXiv:2412.15441* (2024)。[11] Robin Geens、Man Shi、Arne Symons、Chao Fang 和 Marian Verhelst。2024。面向硬件加速器上大语言模型推理优化的能源成本建模。载于*2024*年第*37*届*IEEE*国际片上系统会议（*SOCC*）。1–6。doi:10.1109/SOCC62300.2024.10737844[12] Dan Hendrycks、Collin Burns、Steven Basart、Andy Zou、Mantas Mazeika、Dawn Song 和 Jacob Steinhardt。 [n. d.]。衡量大规模多任务语言理解。载于国际学习表征会议。[13] Xinyi Hou、Yanjie Zhao、Yue Liu、Zhou Yang、Kailong Wang、Li Li、Xiapu Luo、David Lo、John Grundy 和 Haoyu Wang。2024。面向软件工程的大语言模型：系统性文献综述。*ACM*软件工程与方法学汇刊 (Sept. 2024), 3695988。doi:10.1145/3695988[14] Erik Johannes Husom、Arda Goknil、Merve Astekin、Lwin Khin Shar、Andre Kåsen、Sagar Sen、Benedikt Andreas Mithassel 和 Ahmet Soylu。2025。面向边缘人工智能的可持续大语言模型推理：评估量化大语言模型的能源效率、输出准确率和推理延迟。*ACM*物联网汇刊 (Sept. 2025)。doi:10.1145/3767742。已接收。[15] Heli Järvenpää、Patricia Lago、Justus Bogner、Grace Lewis、Henry Muccini 和 Ipek Ozkaya。2024。机器学习赋能系统绿色架构策略综述。载于第*46*届国际软件工程会议论文集：社会中的软件工程。ACM，葡萄牙里斯本，130–141。doi:10.1145/3639475.3640111[16] Bhanu Kaushik、Aman Taneja 和 Sonika Dahiya。2025。迈向可持续人工智能：节能大语言模型综述。载于*2025*年第*8*届国际计算方法与通信会议（*ICCMC*）。944–951。doi:10.1109/ICCMC65190.2025.11140923[17] Is han Kavathekar、Raghav Donakanti、Ponnurangam Kumaraguru 和 Karthik Vaidhyanathan。2025。小模型，大任务：关于小型语言模型函数调用能力的探索性实证研究。*arXiv* 预印本 *arXiv:2504.19277* (2025)。[18] Tahniat Khan、Soroor Motie、Sedef Akinli Kocak 和 Shaina Raza。2025。优化大语言模型：指标、能源效率与案例研究洞见。载于*2025*年*IEEE*人工智能会议（*CAI*）。370–375。doi:10. 1109/CAI64502.2025.00067

[19] Baolin Li、Yankai Jiang、Vijay Gadepally 和 Devesh Tiwari。2024。Sprout：通过碳高效大语言模型推理实现绿色生成式人工智能。收录于《2024年自然语言处理实证方法会议论文集》。计算语言学协会，迈阿密，佛罗里达州，美国，21799–21813。doi:10.18653/v1/ 2024 1215 .emnlp-main. Adam Loy、Lendie Follett 和 Heike Hofmann。[20] 2016。Q–Q图的变化：我们眼睛的力量！《美国统计学家》70, 2（2016），202–214。[21] Shuming Ma、Hongyu Wang、Shaohan Huang、Xingxing Zhang、Ying Hu、Ting Song、Yan Xia 和 Furu Wei。2025。BitNet b1.58 2B4T技术报告。arXiv:2504.12285 [cs.CL] https://arxiv.org/abs/2504.12285[22] Shuming Ma、Hongyu Wang、Lingxiao Ma、Lei Wang、Wenhui Wang、Shaohan Huang、Li Dong、Ruiping Wang、Jilong Xue 和 Furu Wei。2024。1位大语言模型时代：所有大语言模型均为1.58位。arXiv:2402.17764 [cs.CL] https://arxiv.org/abs/ 2402 17764 . G. Macbeth、E. Razumiejczyk 和 R. D. Ledesma。[23] 2011。Cliff's delta计算器：用于两组观测值的非参数效应量程序。《心理学大学》10, 2（2011），545–555。[24] Ggaliwango Marvin、Nakayiza Hellen、Daudi Jjingo 和 Joyce Nakatumba-Nabende。2024。大语言模型中的提示工程。收录于《数据智能与认知信息学》。新加坡施普林格·自然，新加坡，387–402。doi: 10 1007 978 981 99 7962 2_30 。/ - - - - Patrick E McKnight 和 Julius Najab。2010。[25] Mann-Whitney U 检验。《科西尼心理学百科全书》（2010），1–1。[26] 微软，：，Abdelrahman Abouelenin、Atabak Ashfaq、Adam Atkinson、Hany Awadalla、Nguyen Bach、Jianmin Bao、Alon Benhaim、Martin Cai、Vishrav Chaudhary、Congcong Chen、Dong Chen、Dongdong Chen、Junkun Chen、Weizhu Chen、Yen-Chun Chen、Yi ling Chen、Qi Dai、Xiyang Dai、Ruchao Fan、Mei Gao、Min Gao、Amit Garg、Abhishek Goswami、Junheng Hao、Amr Hendy、Yuxuan Hu、Xin Jin、Mahmoud Khademi、Dongwoo Kim、Young Jin Kim、Gina Lee、Jinyu Li、Yunsheng Li、Chen Liang、Xihui Lin、Zeqi Lin、Mengchen Liu、Yang Liu、Gilsinia Lopez、Chong Luo、Piyush Madan、Vadim Mazalov、Arindam Mitra、Ali Mousavi、Anh Nguyen、Jing Pan、Daniel Perez-Becker、Jacob Platin、Thomas Portet、Kai Qiu、Bo Ren、Liliang Ren、Sambuddha Roy、Ning Shang、Yelong Shen、Saksham Singhal、Subhojit Som、Xia Song、Tetyana Sych、Praneetha Vaddamanu、Shuohang Wang、Yiming Wang、Zhenghao Wang、Haibin Wu、Haoran Xu、Weijian Xu、Yifan Yang、Ziyi Yang、Donghan Yu、Ishmam Zabir、Jianwen Zhang、Li Lyna Zhang、Yunan Zhang 和 Xiren Zhou。2025。Phi-4-Mini技术报告：通过LoRA混合打造精巧而强大的多模态语言模型。arXiv:2503.01743 [cs.CL] https://arxiv.org/abs/ 2503 01743 。[27] 2025 Microsoft Azure。 . 定价——Azure OpenAI服务。https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service 访问日期：2025年6月30日。[28] Microsoft Azure。2025年。定价——Phi-3模型。https://azure.microsoft.com/en-us/pricing/details/phi-3/ 访问日期：2025年6月30日。[29] Avanika Narayan、Dan Biderman、Sabri Eyuboglu、Avner May、Scott Linderman、James Zou 和 Christopher Re。2025年。Minions：设备端与云端语言模型之间的高性价比协作。arXiv:2502.15964 [cs.LG] https://arxiv.org/abs/ 2502 15964 。[30] [ NVIDIA NeMo Curator团队。n. d.]。提示任务与复杂度分类器。https://huggingface.co/nvidia/prompt-task-and-complexity-classifier. 访问日期：2025年6月28日。[31] Rahul Pankajakshan、Sumitra Biswal、Yuvaraj Govindarajulu 和 Gilad Gressel。2024。绘制大语言模型安全领域图景：一项全面的利益相关者风险评估提案。arXiv:2403.13309 [cs.CR] https://arxiv.org/abs/2403.13309[32] Soham Poddar、Paramita Koley、Janardan Misra、Sanjay Podder、Niloy Ganguly 和 Saptarshi Ghosh。2025年。迈向可持续自然语言处理：大语言模型推理能耗基准测试的洞见。arXiv:2502.05610 [cs.CL] https://arxiv.org/abs/ 2502 05610 。[33] 2024 Matthew Renze。 . 采样温度对大语言模型问题求解的影响。收录于《计算语言学协会会议论文集：EMNLP 2024》，Yaser Al-Onaizan、Mohit Bansal 和 Yun-Nung Chen（编）。计算语言学协会，迈阿密，佛罗里达州，美国，7346–7356 10 18653 1 2024 432 。doi: . /v / .findings-emnlp。Riccardo [34] Rubei、Aicha Moussaid、Claudio Di Sipio 和 Davide Di Ruscio。2025年。提示工程及其对大语言模型能耗的影响。收录于《2025年IEEE/ACM第九届绿色与可持续软件国际研讨会（GREENS）》。60 67 10 1109 66463 2025 00014 – 。doi: . / [35] 2009 GREENS . . Per Runeson 和 Martin Höst。 . 软件工程案例研究的开展与报告指南。《实证软件工程》14, 2 2009 131 164 10 1007 10664 008 9102 8（April）。 doi: . /s - - - ISBN: _ 9809069 1 1382325615737816 eprint: v 。[36] Roy Schwartz、Jesse Dodge、Noah A. Smith 和 Oren Etzioni。2020。绿色人工智能。《ACM通讯》 63, 12 2020 54 63 10 1145 3381831 （Nov. ），– 。[37] 1965 doi: . / S. S. Shapiro 和 M. B. Wilk。 . 正态性方差分析检验（完整样本）。《生物统计学》52, 3/4（1965），591–611。http://www.jstor.org/ stable/2333709[38] Jovan Stojkovic、Chaojie Zhang、Íñigo Goiri、Josep Torrellas 和 Esha Choukse。2025年。Dynamollm：面向性能与能源效率的大语言模型推理集群设计

。收录于*2025*年*IEEE*高性能计算机体系结构国际研讨会（*HPCA*）。IEEE，1348–1362。[39] H. Vinutha、B. Poornima 和 B. Sagar。2018。使用四分位距技术检测入侵数据集中的离群值。收录于信息与决策科学：第六届*FICTA*国际会议论文集。施普林格，511–518。[40] Tomasz Walkowiak。2025年。大语言模型的能效：一项实证研究。收录于可信系统与网络进展。施普林格·自然瑞士，查姆，221–228。doi:10.1007/978-3-031-92734-8_22[41] Fali Wang、Zhiwei Zhang、Xianren Zhang、Zongyu Wu、Tzuhao Mo、Qiuhao Lu、Wanjing Wang、Rui Li、Junjie Xu、Xianfeng Tang 等。2024。大语言模型时代小型语言模型的全面综述：技术、改进、应用、与大语言模型的协作及可信度。*ACM*

智能系统与技术汇刊(2024)。[42] Eric W Weisstein。2004。邦费罗尼校正。*https://mathworld. wolfram. com/* (2004)。[43] Claes Wohlin、Per Runeson、Martin Höst、Magnus C. Ohlsson、Björn Regnell 和 Anders Wesslén。2024。软件工程实验（第2版）。施普林格·柏林·海德堡，柏林，海德堡。109–111页。doi:10.1007/978- 3-662-69306-3[44] Tim Yarally、Luís Cruz、Daniel Feitosa、June Sallou 和 Arie van Deursen。2023。面向绿色人工智能的批处理——一项关于推理的探索性研究。载于*2023*年第*49*届欧洲微电子软件工程与高级应用会议（*SEAA*）。112–119。doi:10.1109/SEAA60479.2023.00026