小口研究室　研究紹介　(2025年度)
(お茶の水女子大学理学部情報科学科)

LLM推論における複数モデル実行によるGPUリソース活用方法の検討
（研究担当:髙頭　まどか）

研究背景

- 大規模言語モデル(LLM)への注目の高まり
- LLMの学習と推論を高速化するGPUの需要が急増
- 世界的なGPU不足と価格の高騰

▶ **GPUリソースを最大限活用したい**

実験環境

実験用サーバ	PRIMERGY RX2540M6
CPU	Intel(R) Xeon(R) Gold 6430 (x 2)
CPUメモリ	DDR5-4400 512GB
GPU	NVIDIA H100 PCIe (x 1)
GPUメモリ	HBM3 80GB
OS	Rocky Linux 9.5

評価モデル →

openai/gpt-oss-20b	
Total parameters	21B
Active parameters	3.6B
Total size	12.8 GiB
Release date	August 5, 2025

実験概要

■ ベンチマーク

vllm bench serve … オンライン推論サーバの推論性能を評価するためのベンチマーク

■ データセット

ShareGPT[1]をHugging Faceからダウンロードして使用

参考: [1] https://docs.vllm.ai/en/latest/contributing/benchmarks.html

■ GPUリソースの制限方法

1. vLLMのEngine Argumentである gpu-memory-utilizationを制御する
   - 推論に使用するGPUメモリ量を制限する
2. MIG (Multi-Instance GPU) を使用して分割したGPUインスタンス上で実行する
   - MIG … 単一の物理GPUを最大7つの完全に分離されたGPUインスタンスに分割する技術

■ 主な評価指標

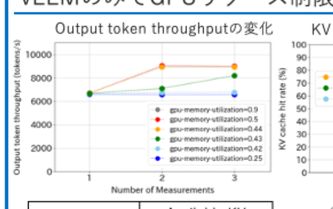
Output token throughput (tokens/sec)
… 単位時間あたりの出力トークン数

■ 研究方針

Step 1: GPUリソース制限による推論性能への影響調査
Step 2: 複数モデルを搭載した場合の性能評価

Step 1: モデル単独実行時

vLLMのみでGPUリソース制限をした場合

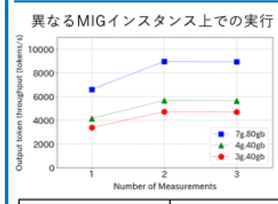


gpu-memory-utilization	Available KV cache memory
0.44	**19.60 GiB**
0.43	18.81 GiB
0.42	18.02 GiB

✓ 算出方法
KV cache hit rate (%) = $\frac{\text{cache\_hits\_total}}{\text{cache\_queries\_total}}$

→ 2回目の計測時に高いスループットを実現するためには19.60 GiBのKVキャッシュメモリ量が必要

MIGで分割したインスタンス上で実行した場合



✓ MIGの表記方法
7g . 80gb
GPUインスタンス数　GPUメモリ量

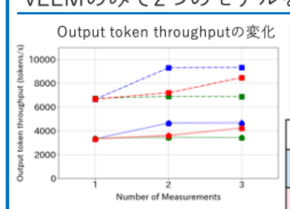
✓ NVIDIA H100で可能な分割の種類
- 1g.10gb　- 3g.40gb
- 1g.20gb　- 4g.40gb
- 2g.20gb　- 7g.80gb

インスタンスの種類	gpu-memory-utilziation	Memory Usage	SM数
7g.80gb	0.44	40.0 GB	**114**
4g.40gb	0.89	40.0 GB	**62**
3g.40gb	0.89	40.1 GB	**46**

→ GPUメモリ使用量 (Memory Usage) が同じ場合，SM数が多いほどスループットが向上する

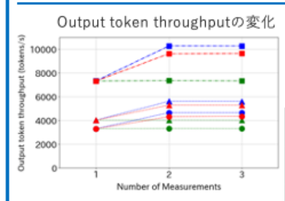
Step 2: 2つのモデル同時実行時

vLLMのみで2つのモデルを同時に実行した場合



gpu-memory-utilization	Available KV cache memory
0.44	model1 → **19.60 GiB**
	model2 → **19.60 GiB**
0.43	model1 → 18.81 GiB
	model2 → 18.81 GiB
0.42	model1 → 18.02 GiB
	model2 → 18.02 GiB

MIGで分割して2つのモデルを同時に実行した場合



0.88	3g.40gb → **19.47 GiB**
	4g.40gb → **19.47 GiB**
0.87	3g.40gb → 19.08 GiB
	4g.40gb → 19.08 GiB
0.86	3g.40gb → 18.69 GiB
	4g.40gb → 18.69 GiB

まとめと今後の課題

- KVキャッシュメモリ量のわずかな違いで，2回目の計測時のスループットに大きな差が生じる
- MIGで分割して実行することで，全体のスループットを向上させることができる
- 今後は，MIGのさらなる活用可能性を検討するとともに，vLLMのみで複数モデルを効率的に動作させる方法の検討も進める

スマート農業における動的データ共有のための分散共有基盤
（研究担当:芦田　多香子）

研究背景

多様なデータが流通・蓄積
↓
データの信頼性が低いことが問題
↓
データエコシステムにおいてデータの検証可能性の重要性が高まる
↓
ブロックチェーン技術を活用した分散共有基盤に注目

一例

スマート農業

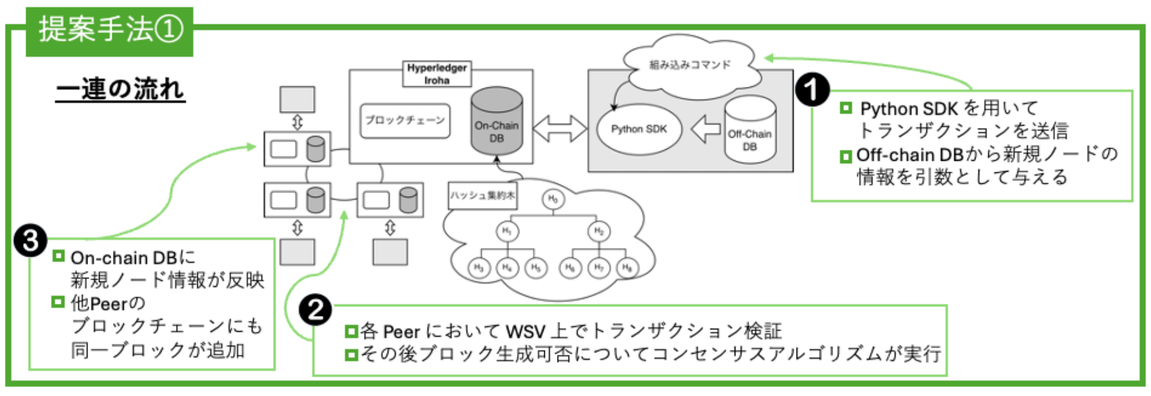
- 生産者・流通者が環境データや生育情報を共有することで生産量向上、産地偽装の防止
- 全てのデータを開示することはプライバシの観点から望まれない

方針

- **Hyperledger Iroha**を用いて高い信頼性と耐改ざん性を有するデータ管理基盤を構築
- スマート農業特有の動的データ管理を対象として新規データ追加時の性能評価

提案手法①

一連の流れ



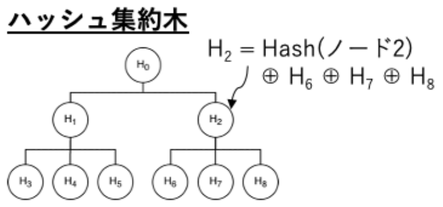
❶
- Python SDK を用いてトランザクションを送信
- Off-chain DBから新規ノードの情報を引数として与える

❷
- 各 Peer において WSV 上でトランザクション検証
- その後ブロック生成可否についてコンセンサスアルゴリズムが実行

❸
- On-chain DBに新規ノード情報が反映
- 他Peerのブロックチェーンにも同一ブロックが追加

提案手法②

ハッシュ集約木



H_2 = Hash(ノード2) ⊕ H_6 ⊕ H_7 ⊕ H_8

- ノードXのハッシュ値：Hx

組み込みコマンドの流れ

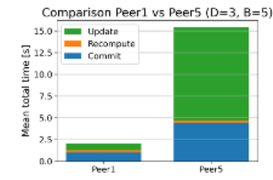
1. **CommitData**
   新規ノードの情報をOn-Chain DBに挿入
2. **Recompute**
   On-chain DBにアクセスしハッシュ値を再計算
3. **UpdateValue**
   再計算したハッシュ値をハッシュ集約木に一括更新

- On-chain DB：ブロックチェーンで合意されたスナップショットを保持する DB
- Off-chain DB：データをローカルに管理する DB

性能評価

新規ノードを追加する際の実行時間を測定

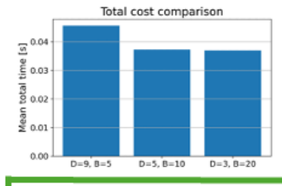
検証①　単一Peer構成 VS 5Peer構成



	単一Peer	5Peer
Commit Data	0.9828	4.4306
Recompute	0.2803	0.2647
Update Value	0.7788	10.7301
合計	2.02	15.43

- CommitDataとUpdateValueの時間が**大幅に増加**
  →トランザクション送信と合意形成を伴う処理がPeer 数の増加に伴って複雑化するため
- Recomputeの時間は**一定**
  →ネットワーク通信や合意形成を伴わないため

検証②　深さ(D)が深い木または1つの親に対する子の数(B)が大きい木どちらの方が優位か



子ノード数を増加させる構造の方が有利
→ルート方向へ遡る回数が少ないほど処理全体の実行時間が抑えられる

まとめと今後の課題

- スマート農業の動的環境データに対応するため組み込みコマンドの仕様変更・性能評価を行った
- より実運用環境に近い形での再評価を行っていく

SNS上のクレジットカード情報取引グループにおけるトピック分析
（研究担当:塩田　実久）

研究背景

- クレジットカード不正利用被害が深刻化
- フィッシング攻撃などから窃取されたカード情報がSNSで取引されている
- SNS上の投稿メッセージすべてを監視することは困難
- 議論内容は時間と共に変化
- 議論構造の変化を特定し，犯罪活動の活発化と沈静化の兆候を早期に把握したい

グループチャット全体の関心の移り変わりや活動の発展・転換を把握することで
リスクに応じた重点監視指標を提案

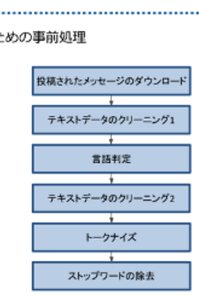
提案

- 監視対象SNSとしてTelegramを採用
  - 秘匿性の高さから犯罪に利用されやすい一面がある
- 犯罪性を有するグループチャットが対象
- 単位期間ごとに投稿メッセージからトピックを抽出
  - BERTopicを使用
  - 単語の出現回数のみではなく，文脈を考慮できる
- トピック同士の類似度を計算
  - 各トピックを重みベクトルで表す
  - 重みベクトルのコサイン類似度を類似度とする
- 時系列遷移図・Sankey Diagram・UMAP埋め込み図で類似関係を可視化
  - トピックをノードとして表現し，類似度が一定以上の場合にエッジを付与
- トピックの継続性や影響力からトピックや単位期間ごとのリスクを評価
- 監視の重要度を判断

トピック抽出のための事前処理

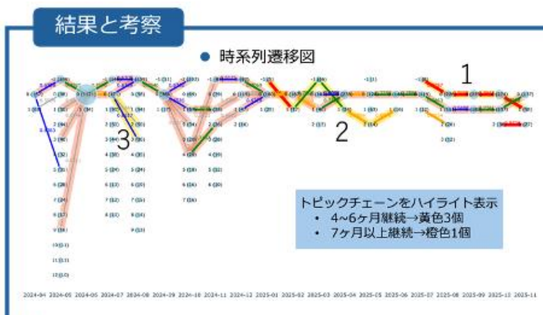
テキストデータのクリーニング1
・絵文字/URL/メンションの除去

テキストデータのクリーニング2
・大文字から小文字への変換
・記号と特殊文字の削除
・短縮形の展開



結果と考察

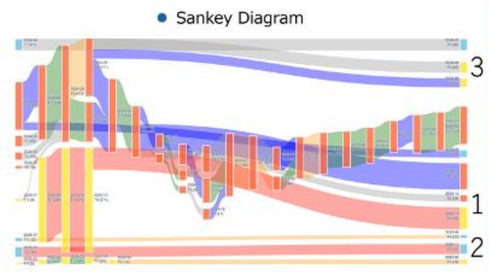
● 時系列遷移図



トピックチェーンをハイライト表示
- 4~6ヶ月継続→黄色3個
- 7ヶ月以上継続→橙色1個

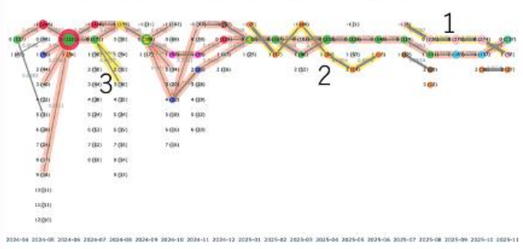
◆ 類似度が0.5以上の場合にエッジを付与
◆ 2024年:　各月のトピック数が2~13の範囲で最大6.5倍の差
　類似度が0.6未満のエッジが約4割を占める
　エッジを持たないノードが多く，トピックの発生と消滅が突発的
　→トピックが定まっておらず，継続的に変化を追跡すべき
◆ 2025年:　トピックが収束
　→収束したトピックが犯罪に関与している場合にのみ監視対象

● Sankey Diagram



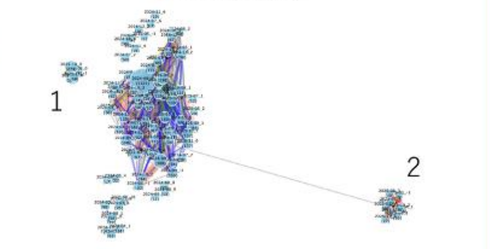
◆ リンクの太さはトピック間の類似度とトピックに属する文書数を反映
◆ 流入/流出するリンクの総和の大きい方がリンクの縦幅
◆ 時系列遷移図のハイライトをノードの色に反映
◆ 黄色1:　流量が短期間で大きく増加→優先度の高い監視
◆ 黄色2:　流量が少ない→現時点では優先度が低いが流量の変化に注意
◆ 黄色3:　橙色チェーンの消滅部分
◆ 橙色:　全期間を通して安定した流量でトピックが継続
　犯罪エコシステムの中核的な活動と関連している可能性

● クラスタリング結果を反映した時系列遷移図



◆ 橙色チェーンのように長期間継続しているように見えるトピックが徐々に変化している可能性を考える
◆ 全期間のトピックに対して交グラフクラスタリング手法を適用
◆ クラスタごとにノードの色を分けて時系列遷移図に結果を反映
◆ 黄色1, 2:　単一のクラスタを形成
◆ 橙色:　チェーン内で複数のクラスタに分離
　→チェーン内で話題の性質が変化している可能性

● UMAP埋め込み図



◆ 全期間のトピック間にエッジを付与
◆ 1: 黄色1のすべてのノードを含み，かつそれ以外のノードを一切含まないすべてのノード間に類似度0.9以上のエッジ
　→トピックがチェーン全体を通して安定して継続
◆ 2:黄色2のすべてのノード + それ以外の期間のトピック4個
　すべてのノード間に類似度0.6 以上のエッジ
　→ 黄色2以外の期間で当該トピックと近い内容を持つトピックが出現

https://www.is.ocha.ac.jp/~oguchi_lab/　　oguchi@is.ocha.ac.jp