AWSで作る自社用基盤モデル

Tatsuya Urabe

Machine Learning Solutions Architect

アジェンダ

AWS上で自社製の独自基盤モデルを1から構築するための知識

- Why: 基盤モデルの特徴と独自構築の必要性
- What: 基盤モデル構築のための開発プロセス
- How: 基盤モデル作成に関わるAWSの支援体制・テクノロジー

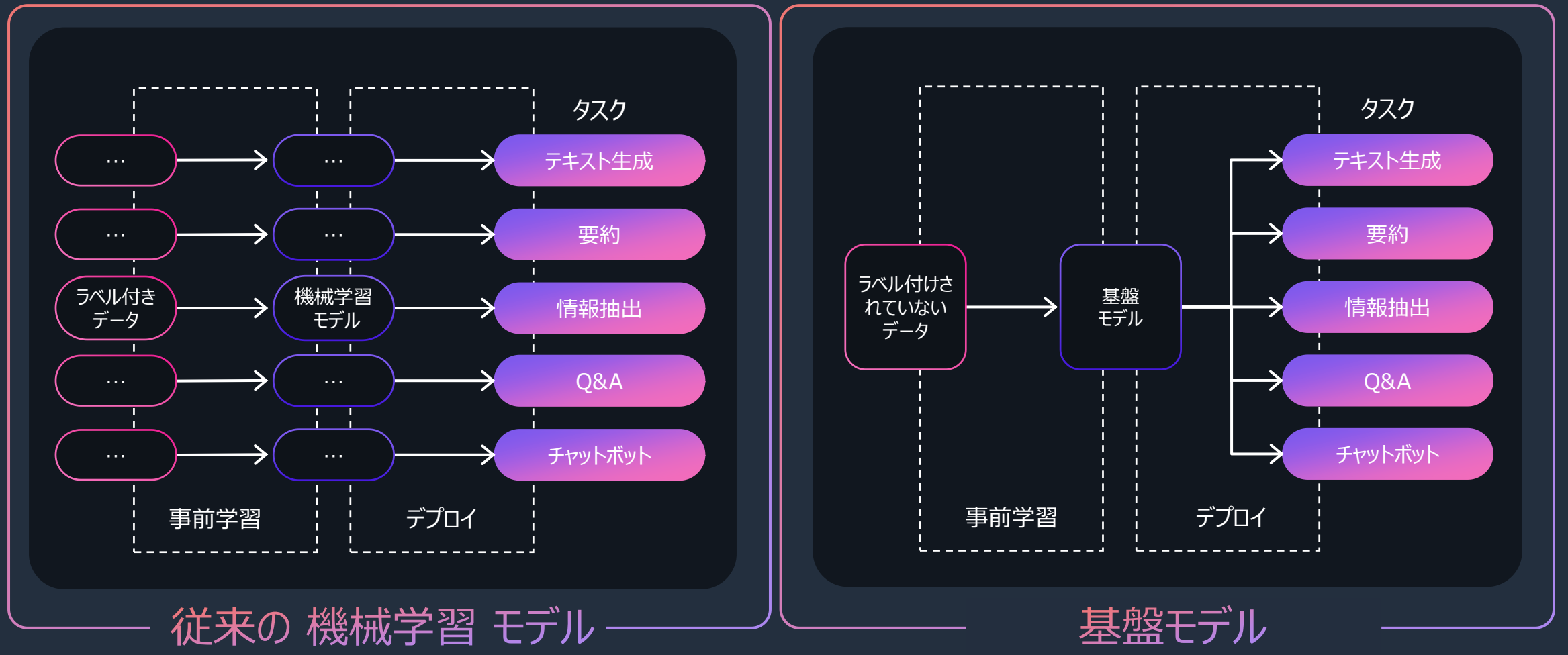
自己紹介

卜部　達也 (Urabe, Tatsuya), Ph. D

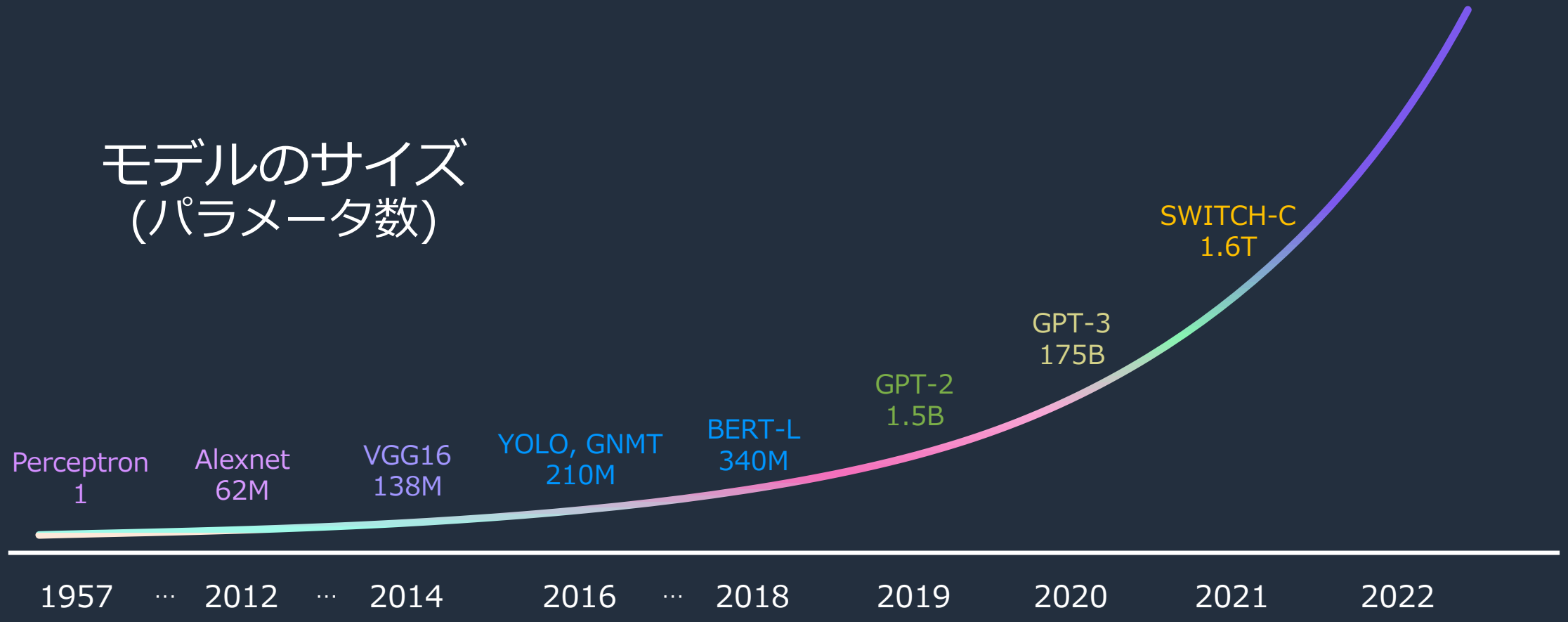
- 機械学習ソリューションアーキテクト
  - 機械学習/AI サービスを担当
  - 前職は電機メーカで自動運転開発
  - 前々職は化学の研究者
  - 趣味：リサイクルショップ巡り、燻製づくり

  - 好きなAWSサービス: Amazon SageMaker, Amazon Rekognition

基盤モデルとは

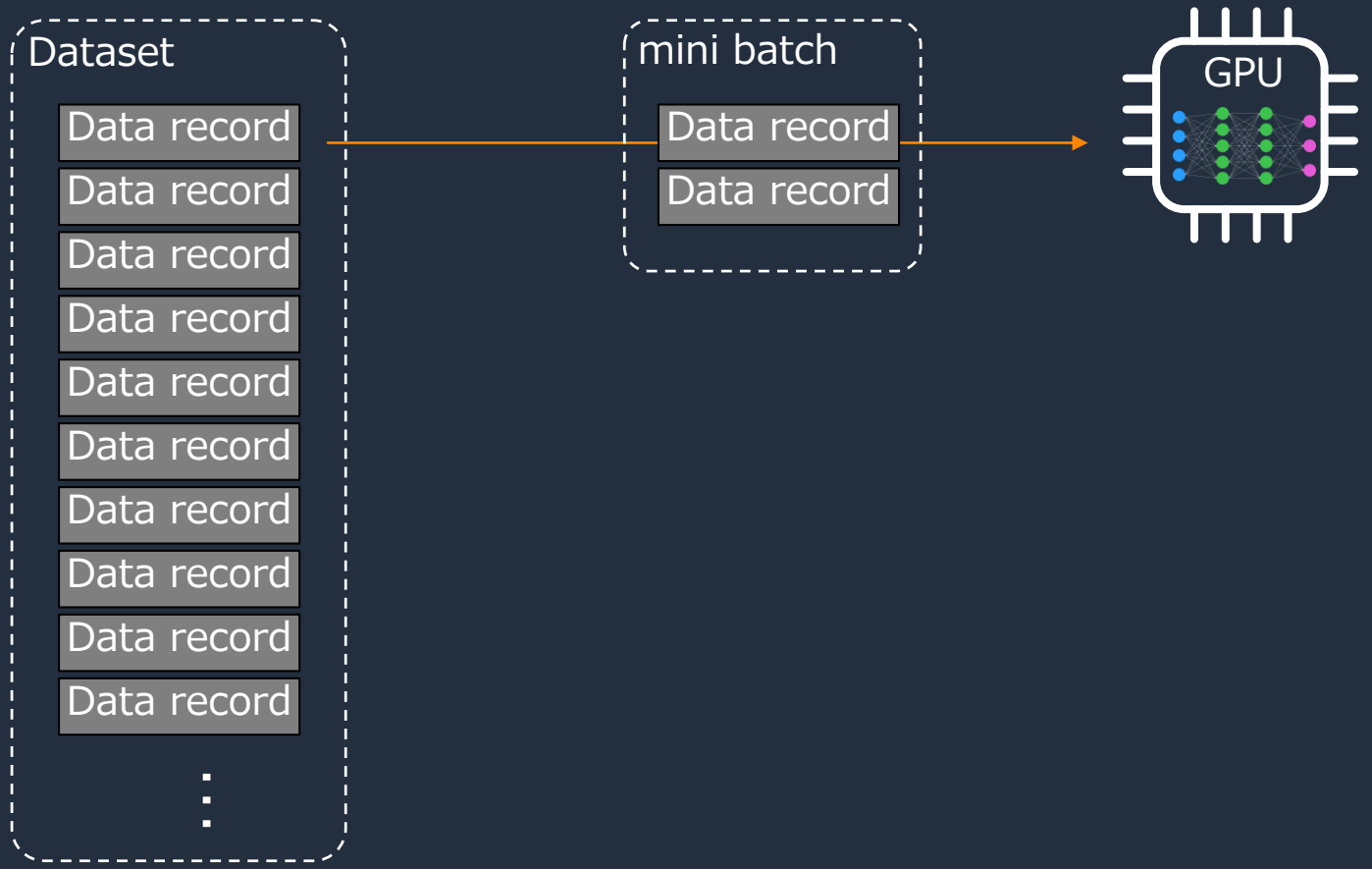


モデルは驚異的なペースで大規模化

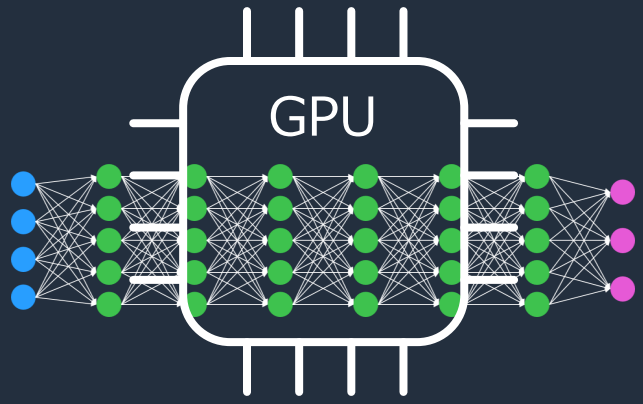


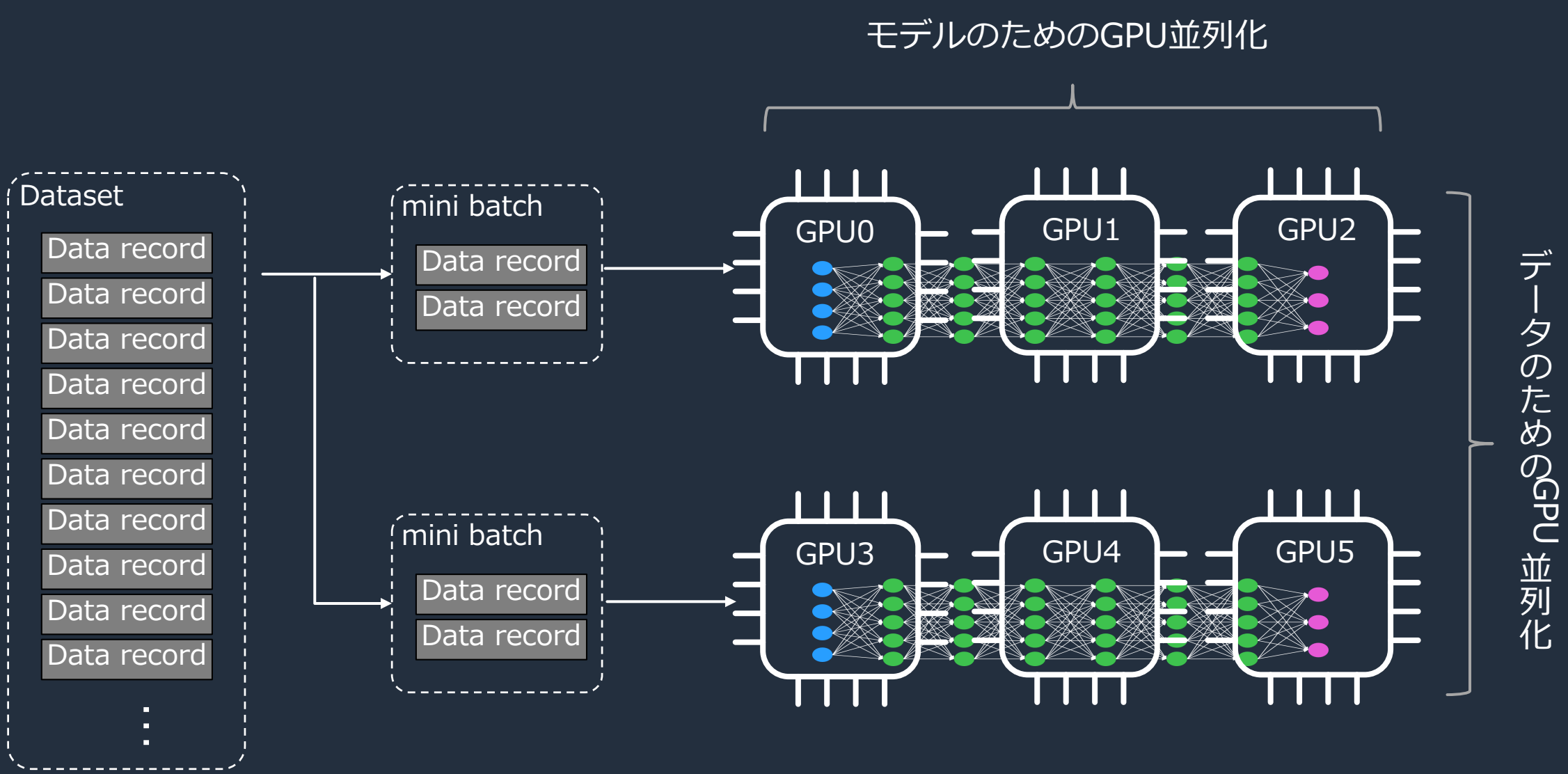
基盤モデルの特徴

学習データサイズが大きく、1台のGPUでは学習に時間がかかる



モデルサイズが大きく、1台のGPUメモリに乗らない





大規模言語モデル学習の特徴

学習 (pretraining)に大量の計算資源と長い時間がかかる

= コストがかかる

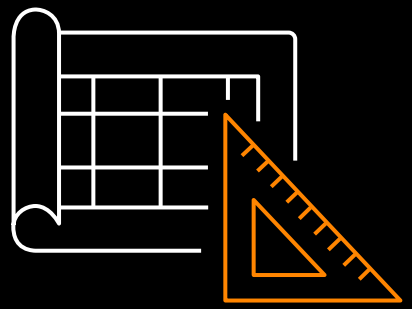
大規模言語モデル学習の特徴

ビジネスに即したモデルを効率良く構築することが大事

- 業務要件にあったモデル・データを選択する
- 分散学習の効率を最適化する
- 学習を継続・再開できる仕組みをいれる
- コスト効率の良いインスタンスを選択する

もちろん、モデルを一から作らないという選択肢もあり

基盤モデルを活用する3通りの方法



自らファウンデーション
モデルをスクラッチ開発

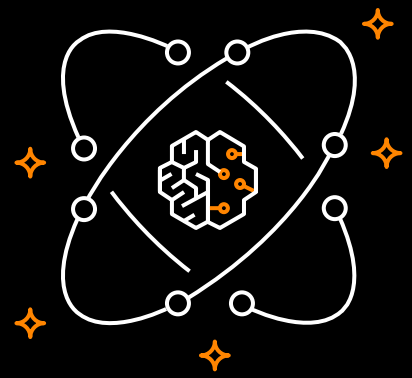
コスト、開発期間要
業務・ML専門性必須

モデルプロバイダー

公開済ファウンデーション
モデルの活用

運用に辿り着くまでに
相当な分類作業要

モデルチューナー



生成系AI、API、ファウンデー
ションモデル提供ベンダーの活
用

データ管理・コスト管理要
自社カストマイズ余地小

モデルコンシューマー

なぜ基盤モデルを一から作成するのか

- 最新・オリジナルのモデル構造を採用したい
- 公開されている利用可能な基盤モデルではパラメータサイズが不十分・過剰
- 特定の言語・ドメインの知識のみのデータからなるモデルを作りたい
- 自社で大量のデータを持っている（例：コールセンターのログ）
- 自然言語以外の生成モデルを作る（プログラムコード、化合物候補、音楽）
- 出力結果の品質をコントロールしたい

基盤モデルを活用する3通りの方法

自らファウンデーション
モデルをスクラッチ開発

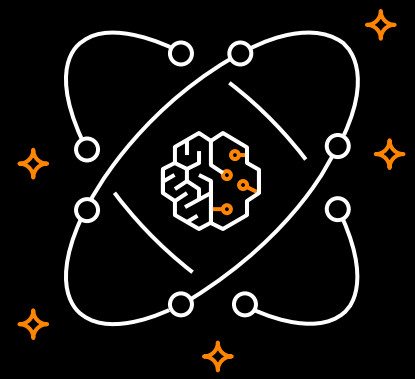
コスト、開発期間要
業務・ML専門性必須

モデルプロバイダー

公開済ファウンデーション
モデルの活用

運用に辿り着くまでに
相当な分類作業要

モデルチューナー



生成系AI、API、ファウンデー
ションモデル提供ベンダーの活
用

データ管理・コスト管理要
自社カストマイズ余地小

モデルコンシューマー

基盤モデル構築のための開発プロセス

基盤モデル → 学習が大変

基盤モデル → ~~学習が大変~~

学習以外も大変

大規模言語モデル学習の特徴

再掲

ビジネスに即したモデルを効率良く構築することが大事

- 業務要件にあったモデル・データを選択する
- 分散学習の効率を最適化する
- 学習を継続・再開できる仕組みをいれる
- コスト効率の良いインスタンスを選択する

モデルを一から作らない、という選択肢も重要…

生成系AIモデル構築支援例

- 1st call
- オファリング紹介
- コスト見積もり
- ワークショップ

- 計算リソース上限緩和
- アーキテクティング
- 前処理、学習コード実装
- パフォーマンスチューニング
- ベンチマーク測定
- 最終コスト試算

- 環境構築支援（環境委譲）
- 本番に向けた動作確認
- トラブルシューティング
- 追加機能の実装
- 本学習
- 性能評価

- ホスティング
- PR（プレスリリース等）

生成系AIモデル構築支援

step 1

- ご支援内容紹介
- コスト見積もり
- ワークショップ

step 2 step 3 step 4

基盤モデル構築に関する全体像の確認・議論を行うステップ

構築にかかるコストを概算し、コスト最適化のための作業をリストアップ

ビジネスに必要なモデルとデータを業務内容レベルで議論

コスト見積もり

計算処理能力（ex. FLOPS）だけでコスト試算・比較するのは難しい

~~AA社 XXインスタンス ○○ TFLOPS~~ VS ~~BB社 YYインスタンス ○○TFLOPS~~

構築したいモデルの特性に合わせ、計算機の性能をフルに引き出すことが必要

条件

- データ
- 前処理
- 学習期間
- モデルの構造

×

環境

- アーキテクチャ
- インスタンス
- 分散学習方式
- フレームワーク

＝ 

実際のコストは大きく変動するため、この段階では概算金額により投資効果を判断する
ここでは、コスト最適化のためのポイントを理解することが重要

ビジネスに必要十分なモデルの要件決定

基盤モデルの要件を技術観点・ビジネス観点の両面からディスカッション

開発内容・スケジュールを決定

AWSと協業して開発の場合は、タスク割り振りを決定

生成系AIモデル構築支援

step 1

step 2

step 3

step 4

- 計算リソース上限緩和
- アーキテクティング
- 前処理、学習コード実装
- パフォーマンスチューニング
- ベンチマーク測定
- 最終コスト試算

本番の学習に向けて環境の整備と学習の最適化をおこなうステップ

スループットを測定して最終的な学習コストを算定

ユーザ主体でおこなう場合とAWS主体（無償/有償）でおこなう場合がある

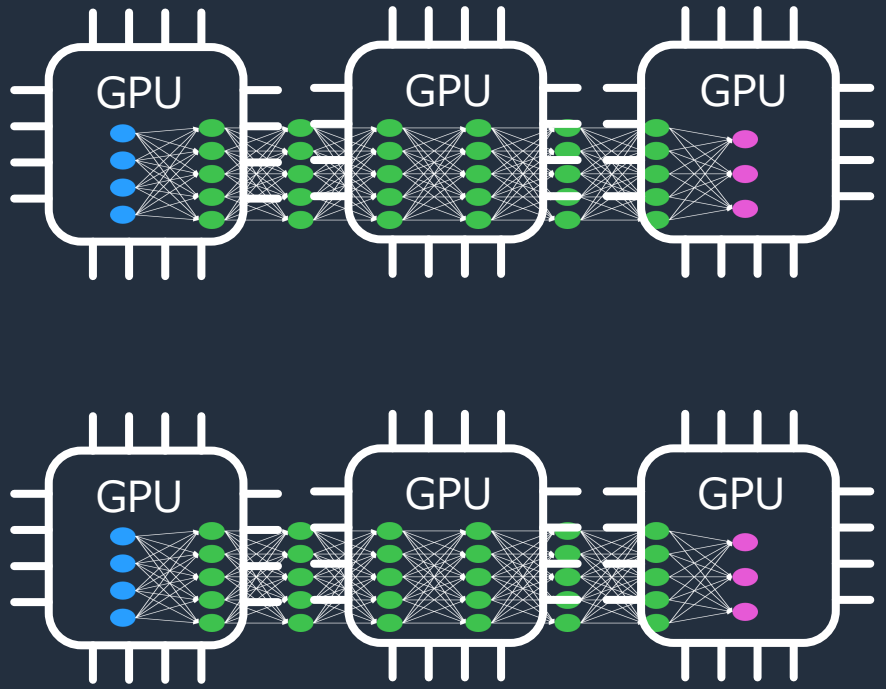
計算リソース確保

コストが高いGPUインスタンスはデフォルトで利用上限数が0になっている

上限緩和申請をおこない、学習に必要なリソースを設定する

大量のインスタンスを使用する場合や学習の時期やスケジュールも勘案

インスタンスの種類・数によっては余剰のあるリージョンが限定される可能性もあるため、海外リージョンを使うためのデータの保管、移行ポリシーなどを確認・クリアしておく



学習環境の最適化とベンチマーク

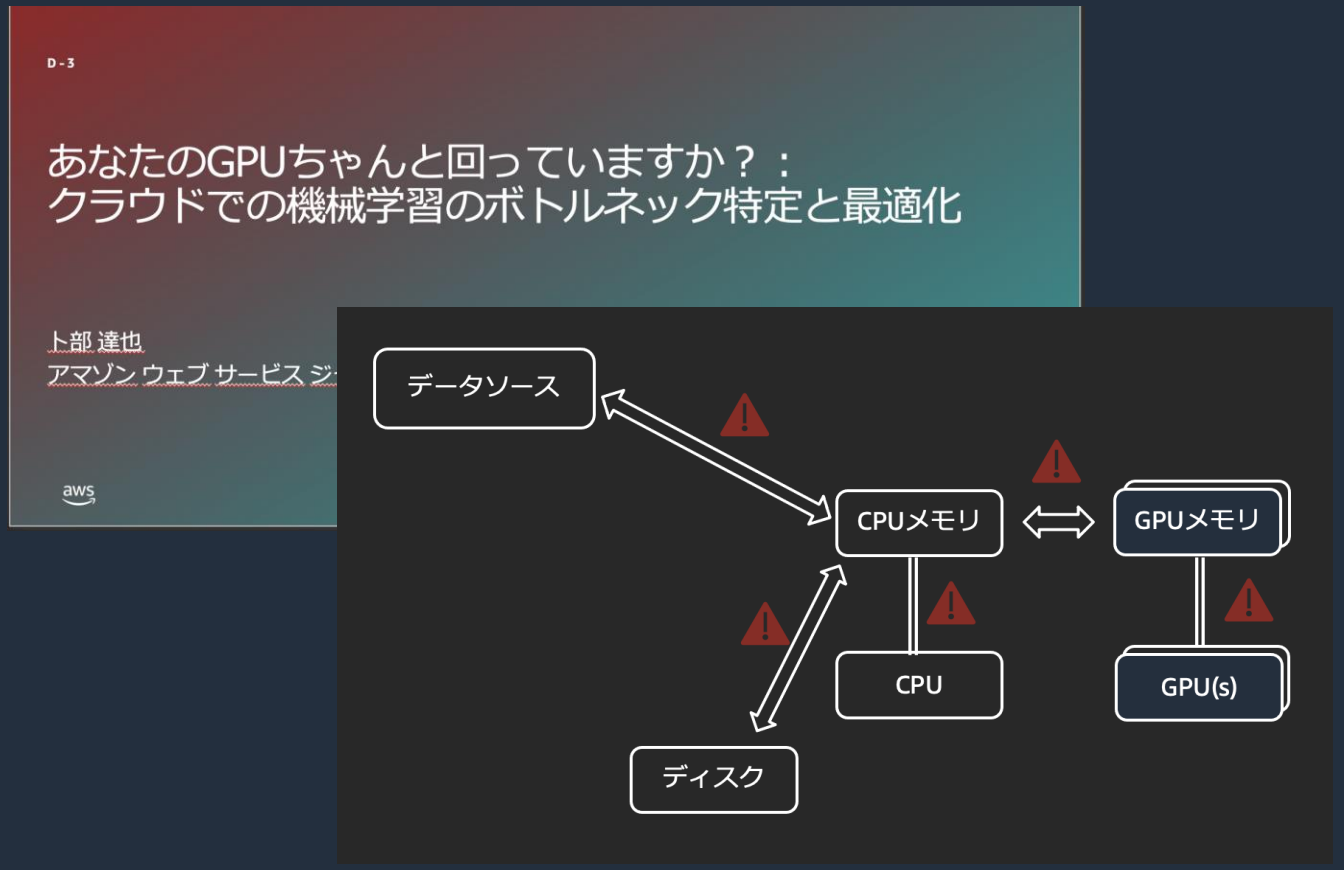
- お客様の要件に合わせたアーキテクチャを選定する
- データを学習させるための前処理セクションなども実装
- 学習にボトルネックがないか確認、GPU利用率、スケーラビリティのチェック
- lossやperprexityが改善することを確認
- この時点でスループットを測定し、最終的なコスト試算を実施

クラウド上での機械学習に関わるボトルネック

学習のパイプラインに一つでもボトルネックが生じると、学習の効率が落ちる

効率が50%に落ちるとコストが2倍になる

基盤モデル作成の場合、大量のCPUとGPUを協調して動かすため問題はさらに深刻



AWS Dev Day Online Japan D-3 : あなたのGPUちゃんと回っていますか？ - クラウドでの機械学習のボトルネック特定と最適化

生成系AIモデル構築支援

step 1 → step 2 → **step 3** → step 4

学習に向けた最終調整と本番学習をおこなうステップ

- 環境構築支援（環境委譲）
- 本番に向けた動作確認
- トラブルシューティング
- 追加機能の実装
- 本学習
- 性能評価

本番学習

学習前のトラブルの洗い出しと学習再開用のチェックポイントの動作確認

- 不足の事態に対処する仕掛けをいれておく
- 一番避けたいのは学習が失敗してかけたコストが無駄になること
- チェックポイント。学習を完全に再現できるか、学習を改善できるか、
- lossの発散など、学習中のイベントを検知・通知する機能を仕込んでおく

生成系AIモデル構築支援

step 1

step 2

step 3

step 4

作成した自社基盤モデルを運用し
ビジネス活用するステップ

- ホスティング
- PR（プレスリリース等）

自社基盤モデルの活用フェイズ

- 最も重要なのがランニングコスト
- 基盤モデルは巨大なためモデルをホストするのにもコストがかかる
- モデルによっては推論時に1台のインスタンスに乗り切らないため、分散推論が必要になる
- 対外的なアピール（プレスリリースやイベント登壇）も必要

基盤モデル作成に関わる AWSの支援体制・テクノロジー

支援体制編

AWS Machine Learning Solutions Lab

ビジネス経験 と AI/ML専門知識 を有するMLストラテジストのチーム

ワークショップとアイデアセッションを通じて、ビジネス問題から逆算してユースケースを算定。
プロダクション化までのロードマップと実装計画を策定し、お客様のジャーニーを支援

ML Discovery Workshop（MLDW）の概要

AWSのAI/MLエキスパートを交えたディスカッションを通じ、
MLプロジェクトの最終実現イメージと、直近の PoC/MVP の実施内容を整理

主な議論内容

全体像

- 全体的なビジネス課題
- 最終実現イメージ（あるべき姿）

個別MLユースケース

- ビジネス価値
- 実現性（技術的難易度、社内体制・プロセス、セキュリティ・コンプライアンス規制、など）
- データ（内容、質、量）

お客様参加メンバー

- エグゼクティブスポンサー
- 事業担当責任者
- 事業関連システム・データの主担当者
- データサイエンティスト（オプション）

その他

- 所要時間：3～4時間（ユースケース数による）
- 実施形態：オンサイト or リモート

生成系AIモデル構築支援例

- 1st call
- オファリング紹介
- コスト見積もり
- ワークショップ

- 計算リソース上限緩和
- アーキテクティング
- 前処理、学習コード実装
- パフォーマンスチューニング
- ベンチマーク測定
- 最終コスト試算

- 環境構築支援（環境委譲）
- 本番に向けた動作確認
- トラブルシューティング
- 追加機能の実装
- 本学習
- 性能評価

- ホスティング
- PR（プレスリリース等）

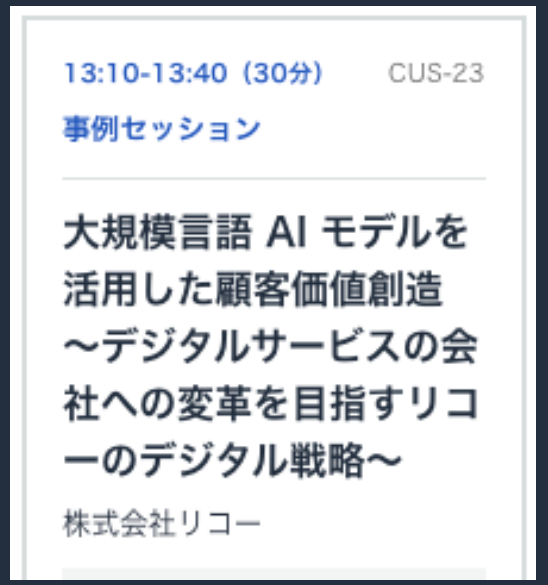
Machine Learning Solutions Labが包括的にご支援*

* 支援実施には条件があります

AWS Machine Learning Solutions Labの生成AI支援事例

株式会社リコー様

- GPT3 を使った大規模日本語言語モデルの開発を支援
- Machine Learning Solution Lab による大規模分散学習環境の構築と学習最適化を実施
- 60億パラメータの生成モデルを複数GPU間で効率的に学習。学習時間を34%改善
- お客さま事例としてAWS Summit Tokyoにて登壇 (2023.04.29)



PROBLEM

Ricoh Company, Ltd. is well known for spearheading the era of office automation through its printers, copiers and fax machines. Continuing to deliver solutions that solve workplace challenges, they offer "AI at Work", a service that takes document data and turns it into business intelligence. They wanted to enhance this service to provide more advanced features like a virtual sales representative to support the creation of sales proposal materials. This required creating a large-scale Japanese language model as a core model of the service.

SOLUTION

Ricoh Company, Ltd. collaborated with the Amazon Machine Learning Solutions Lab (MLSL) to develop a solution that can fit 6 billion parameter models into GPU memory and handle a text dataset of hundreds of billions of tokens. MLSL developed a solution that utilize efficient distributed training tools such as SageMaker Distributed Model Parallel Library, Mixed Precision training and Fused operations on SageMaker training job.

RICOH

IMPACT

- Reduced memory errors: the solution enables model training at the billion parameter level without memory errors by partitioning the large model across multiple GPUs and instances.
- Reduced training time: reduced training time by 34% compared to the benchmark by optimizing each parameter while analyzing the memory usage.

AWS プロフェッショナルサービス

https://aws.amazon.com/jp/professional-services/

AWS クラウドを使用して期待するビジネス上の成果を実現するようお客様を有償でサポートする、専門家からなるグローバルチーム

ソリューション別プラクティス　テクノロジー別プラクティス　業種別プラクティス

分析

AWS サービスを使用して、データインフラストラクチャをモダナイズし、データから最大の価値を引き出します。

アプリケーションの移行

AWS へのアプリケーションの移行を自動化して加速します。

アプリケーションの最新化

従来のメインフレームアプリケーションを含むアプリケーションをリファクタリング、再設計、および最新化することにより、革新、ビジネスの俊敏性の向上、およびコストの削減を実現します。

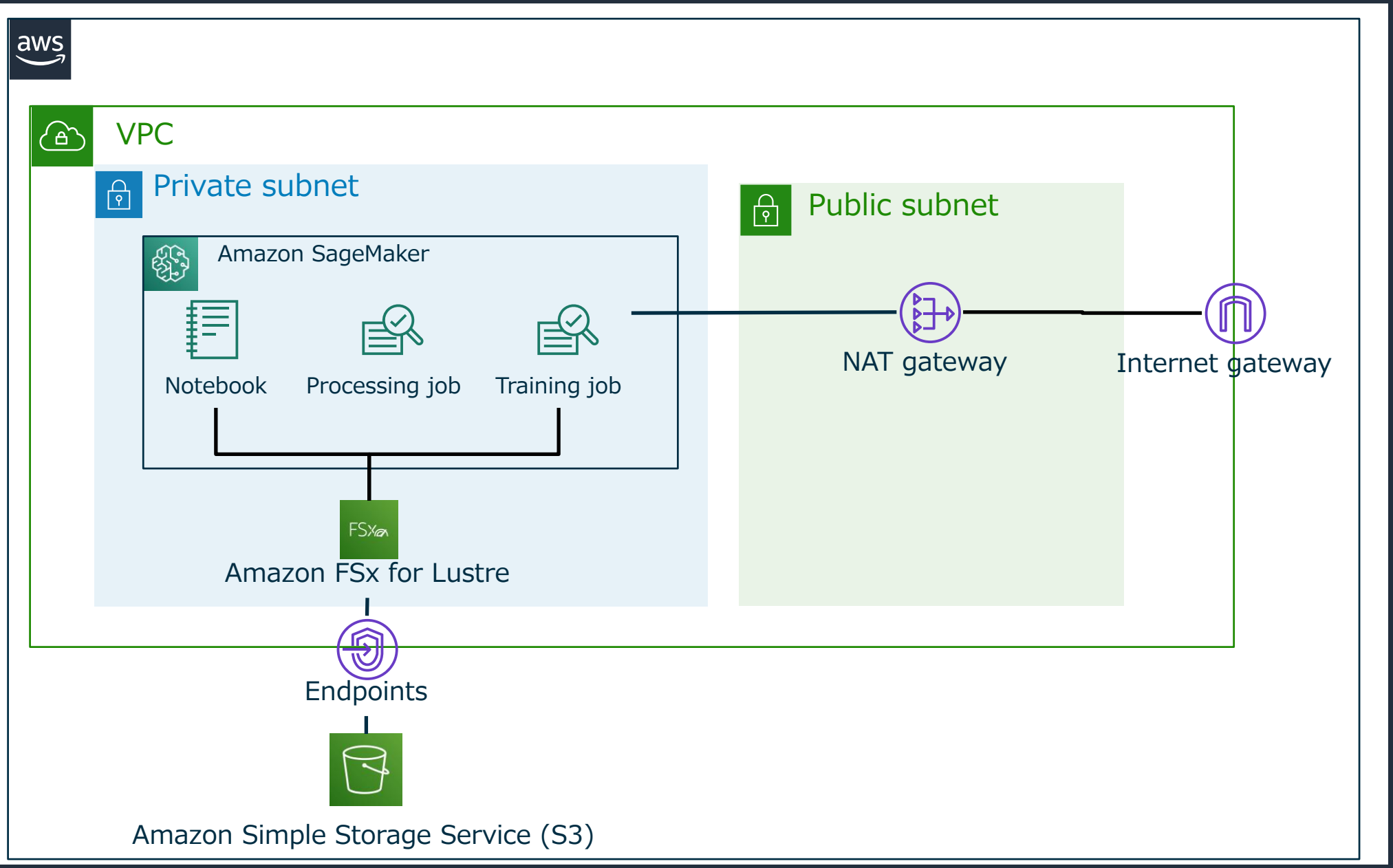
人工知能と機械学習

機械学習とデータサイエンスの専門知識とベストプラクティスを活用して、ビジネス変革を加速します。

基盤モデル作成に関わる AWSの支援体制・テクノロジー

サービス編

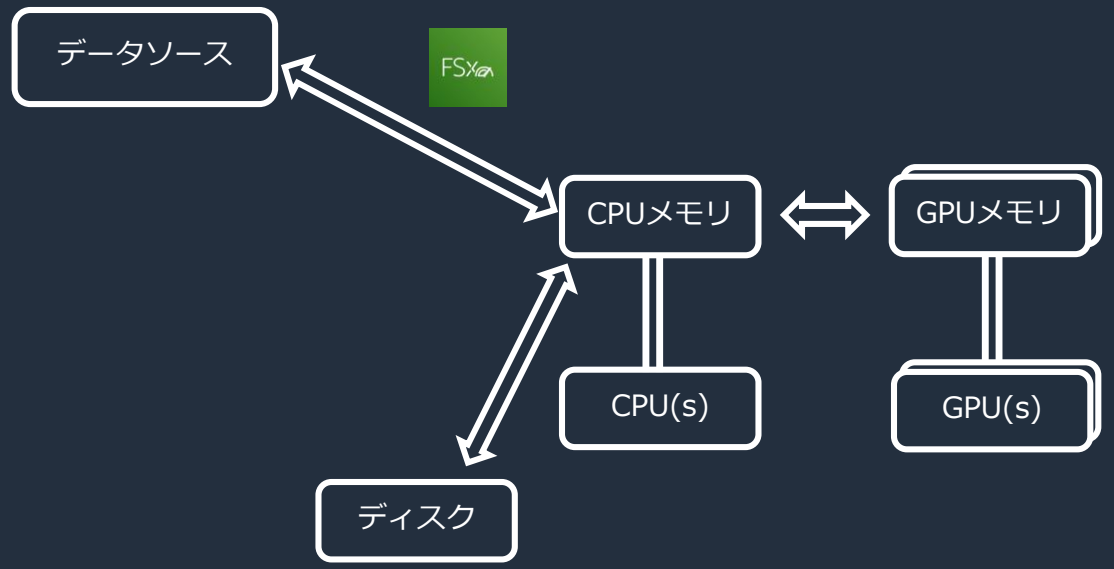
基盤モデル作成時の典型的なアーキテクチャ構成



Amazon FSx for Lustre

分散ファイルシステムを使って読み書きを高速化する

Amazon FSx for Lustre

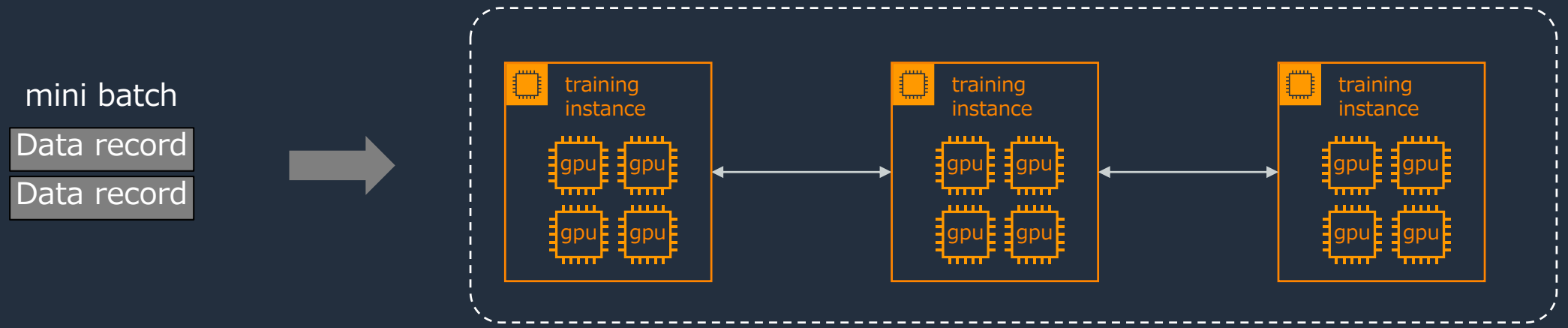


基盤モデル構築の場合、学習途中のモデルファイル（チェックポイント）の書き込みにもLustreを使うことを推奨

分散学習を行う際に考慮すべきポイント

- GPU間の通信 (NVLink)
- インスタンス間の通信 (Amazon Elastic Fabric Adapter)
- 個々のGPUの演算結果を集約する通信ライブラリ (NCCL)
- モデル分散・データ分散の際のGPUの振分け・連携 (Tensor Parallel)
- ミニバッチをさらに複数GPUに分配するアルゴリズム (interleave pipeline)
- etc…

複雑な概念や機能の細部を意識することなく分散学習を設定・実行できることが望ましい



Amazon SageMaker

ML向け エンド to エンドのソリューション

MLを簡単に

ワンクリックでトレーニング/デプロイ、
ビルトインアルゴリズム、
自動チューニング

拡張可能なパフォーマンス

パワフルなGPUへのオンデマンドアクセス
自動拡張、分散トレーニング

効率的に
100台
のGPUに拡張可能

費用対効果の高さ

データラベリングのコストを最大70% 削減、
マネージされたスポットトレーニング、クラウド市場において最安値の推論モデル

少なくとも
54% 少額の TCO
他のクラウドベースのオプションとの比較で

セキュリティ

セキュリティ機能の充実したセット：
ネットワーク分離、アクセス制御、暗号化、
コンプライアンス

幅広い
コンプライアンス：
SOC | PCI | ISO | FedRAMP | HIPAA | GDPR | FIPS

DevOps対応

Kubernetesで実行するオプションを備えた
完全マネージド型のMLオペレーション

単一 IDE
Webベースのビジュアルインターフェイスですべての
MLステップを実行する

KUBERNETES
とKubebflowの統合

Amazon SageMaker distributed training

The fastest and easiest way to train large deep learning models

トレーニング時間の短縮

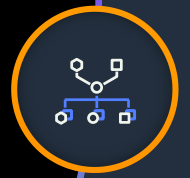
GPU間の同期により、トレーニング時間を25%短縮します

AWS向けに最適化

AWS向けに設計されたデータ並列処理により、ほぼ線形のスケーリング効率を実現します

人気のあるMLフレームワークAPIのサポート

カスタムトレーニングコードなしでHorovodなどの既存のAPIを再利用する



自動で効率的なモデル分割

自動化されたモデルプロファイリングとパーティショニングの実験を回避

最小限のコード変更

10行未満のコード変更でモデルの並列処理を実装する

効率的なパイプライン

全てのGPUをアクティブに保つようパイプラインを構築しリソース使用率を最大化

モデル並列使用時の設定

```
smd_mp_estimator = PyTorch(
    entry_point="training.py",
    role=role,
    instance_count=1,
    instance_type='ml.p3.16xlarge',
    framework_version='1.13.1',
    py_version='py38',
    distribution={
        "smdistributed": {"modelparallel": smp_options},
        "mpi": mpi_options
    },
)
```

```
smp_options = {
    "enabled":True,
    "parameters":{
        "pipeline_parallel_degree": 2,
        "microbatches": 4,
        "placement_strategy": "spread",
        "pipeline": "interleaved",
        "optimize": "speed",
        "ddp": True,
    }
}

mpi_options = {
    "enabled" : True,
    "processes_per_host" : 8,
}
```

学習コードと分離した形で分散学習のパラメータを設定し、ジョブの形で実行できる

https://docs.aws.amazon.com/sagemaker/latest/dg/model-parallel-sm-sdk.html

SageMakerが提供する分散学習用の機能

- 自動モデル分割機能
- 並列化したモデルのチェックポイント機能
- Optimizer State Sharding
- Activation Checkpointing
- Activation Offloading

1. SageMaker分散学習の自動モデル分割機能

モデルの分析

- グラフ構造
- 学習パラメータのサイズ
- etc…

グラフ分割アルゴリズムの実行

- request-responseのコミュニケーションの最小化
- GPU間の処理負荷とメモリ使用量のバランス

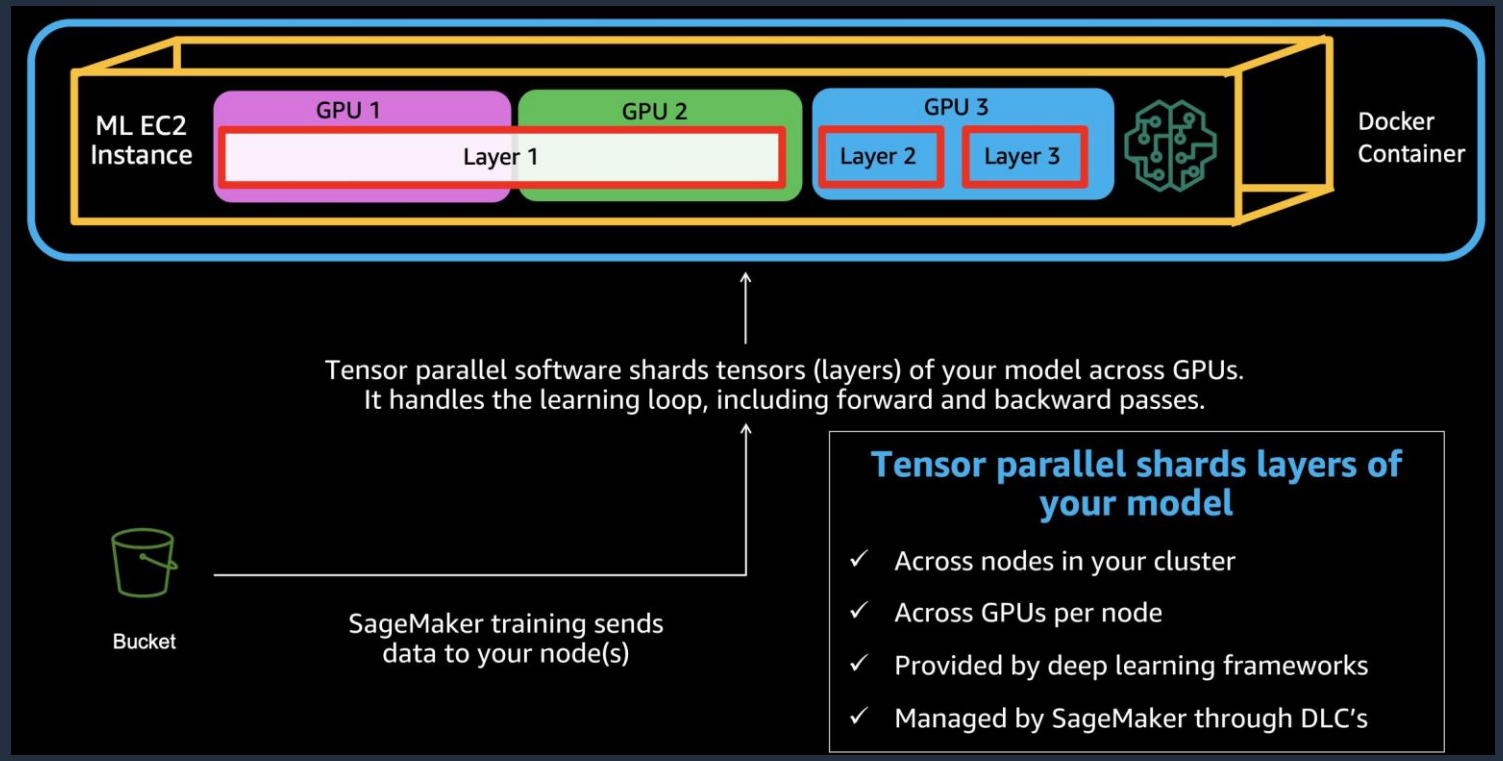
デバイスへの分割モデルの割り当て

- 最適化されたパイプライン方式で実行

AWS Machine Learning Blog

Train 175+ billion parameter NLP models with model parallel additions and Hugging Face on Amazon SageMaker

by Emily Webber, Aditya Bindal, and Luis Quintela | on 01 MAR 2022 | in Amazon SageMaker, Artificial Intelligence | Permalink | Comments | Share



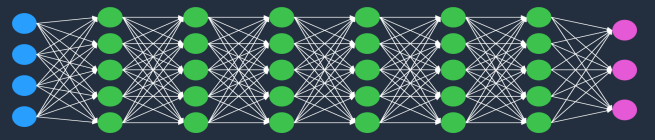
- GPT-3サイズのモデルをSageMakerで学習
- 120台の ml.p4d.24xlargeインスタンスで32 samples/secondのスループット
- 240台のインスタンスの場合25日間でGPT-3の学習が完了

https://aws.amazon.com/blogs/machine-learning/train-175-billion-parameter-nlp-models-with-model-parallel-additions-and-hugging-face-on-amazon-sagemaker/

分散学習したモデルのcheckpointing (途中経過保存)

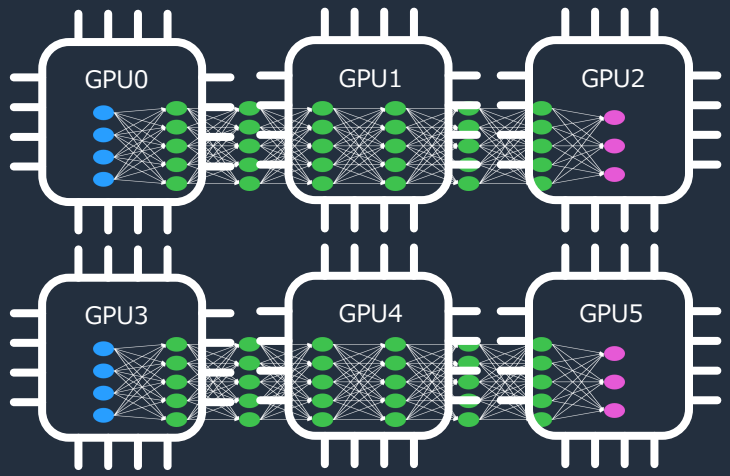
学習が予期せず途中で終了した時に備えて、再開用のcheckpointファイルを保存しておく機能

単純にモデルパラメータを保存して おくだけでは不十分☹



分散学習を厳密に再開するための全ての情報を保存する ☺

- 分割したモデルパラメータ
- 分割したoptimizer パラメータ
- 学習中に動的に変化させたパラメータ
- etc…



SageMakerではsmdistributed.modelparallel.torch.save_checkpointで/opt/ml以下に保存すればクラウド上にアップロードされる

基盤モデルの場合チェックポイントのアップロード先にLustreを使うことを推奨

https://sagemaker.readthedocs.io/en/stable/api/training/smp_versions/latest/smd_model_parallel_pytorch.html#smdistributed.modelparallel.torch.save_checkpoint

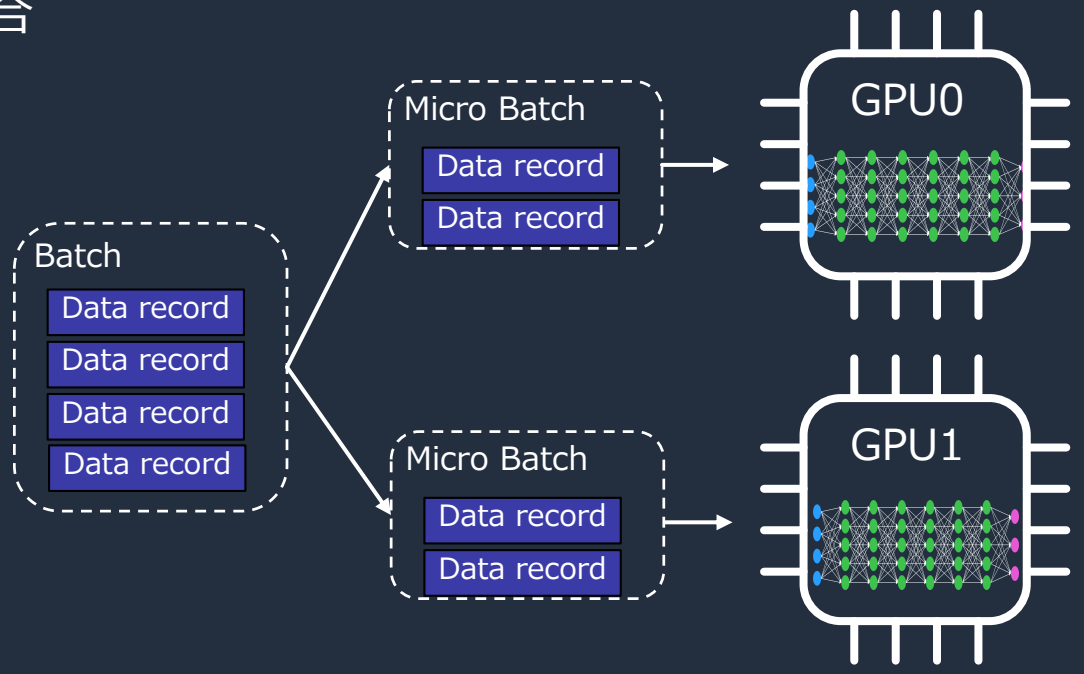
Optimizer state sharding

data parallel時に各モデルが保持するoptimizerの情報を分割(shard)してメモリ使用量を削減する技術

以下の場合に有効

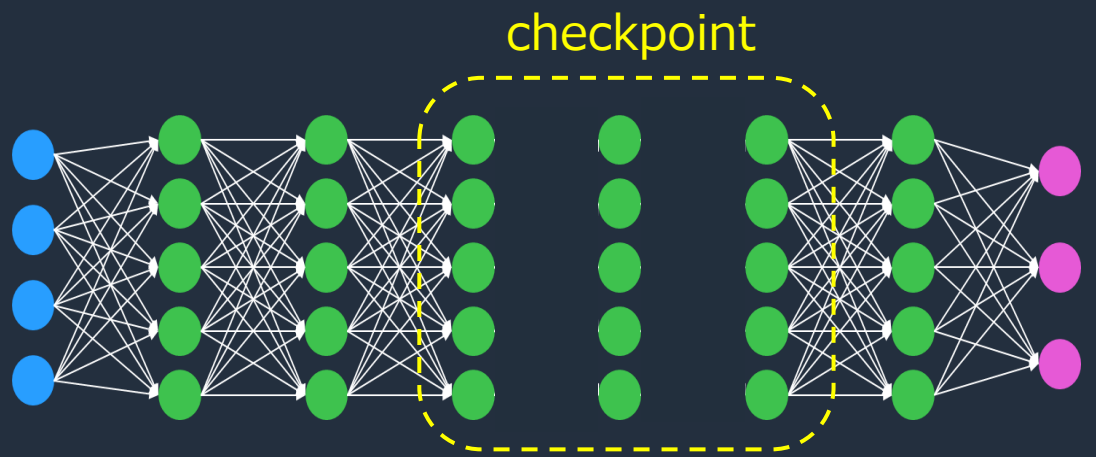
- 過去の状態を保持する(stateful) optimizerを使用している場合 (ex. Adam)
- 半精度 (FP16) のoptimizerを使用している場合

Estimatorの定義時に
“shard_optimizer_state”: True
を指定することで実行できる



Activation checkpointing

forward passで計算した活性化層の出力を保持せず破棄することでメモリ使用量を削減する



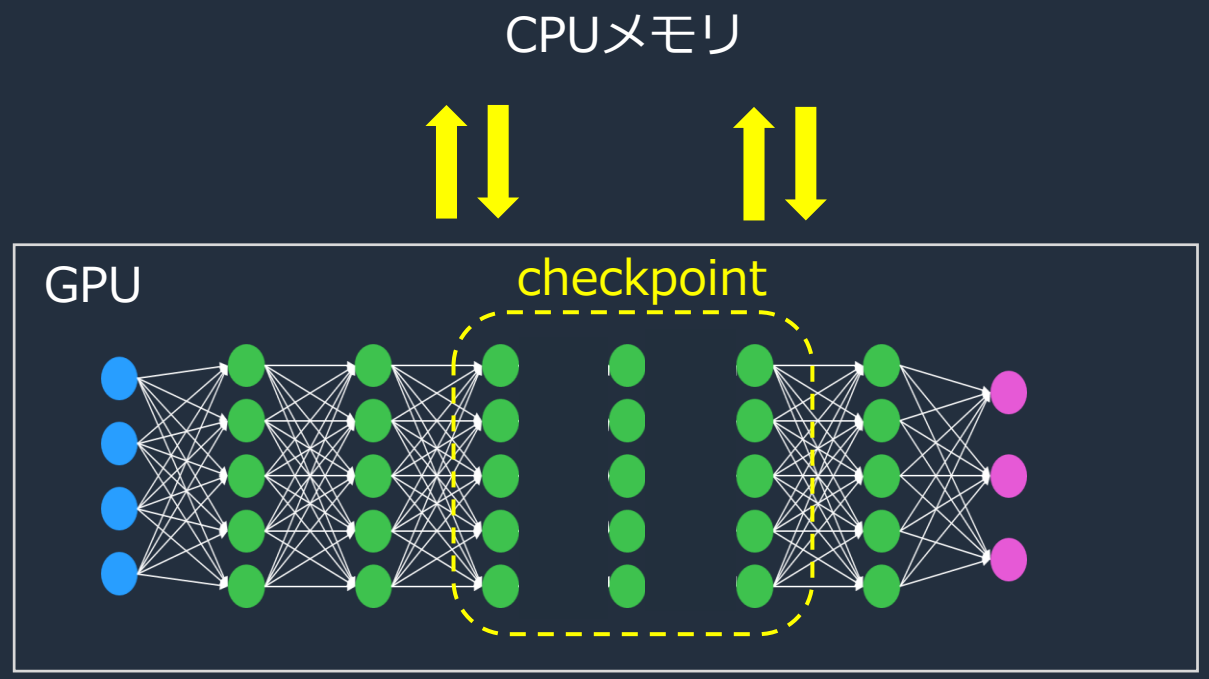
forward pass終了時、checkpointの入力・出力はメモリに保持されるが内部の出力は破棄される。

内部出力はbackward pass時に再計算される

計算量（時間）とメモリ使用量のトレードオフとなる

Activation offloading

checkpointで保持している入出力をCPUメモリに移動 (offload) させることでGPUメモリ使用量をさらに削減する



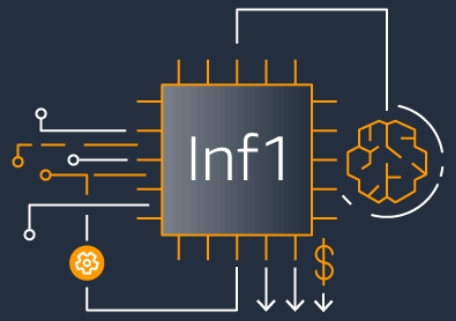
checkpoint 内部の出力は移動されない

移動された入出力はbackward pass時に再度GPUメモリに戻される

基盤モデル作成に関わる AWSの支援体制・テクノロジー

インフラ編

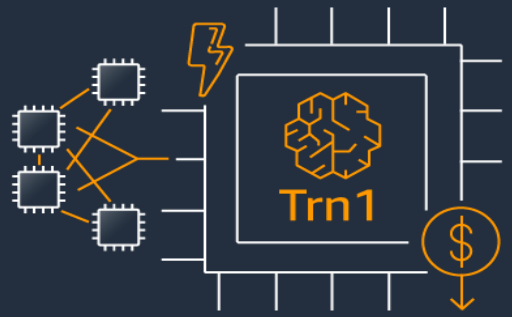
AWS 独自設計 ML チップ搭載インスタンス



AWS Inferentia

初代 ML 推論チップ
高性能かつ低価格

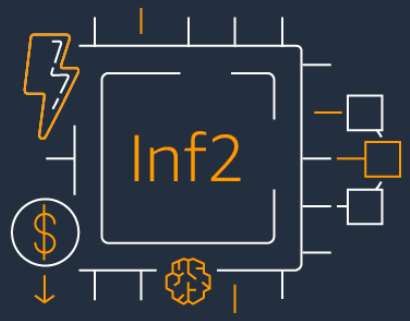
24 リージョンで利用可能
2019年12月 一般提供開始



AWS Trainium

高性能 ML トレーニングチップ
高い費用対効果

NEW
ネットワーク帯域を倍増した Trn1n
2023年4月 一般提供開始

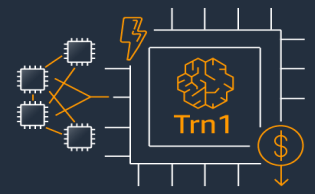


AWS Inferentia2

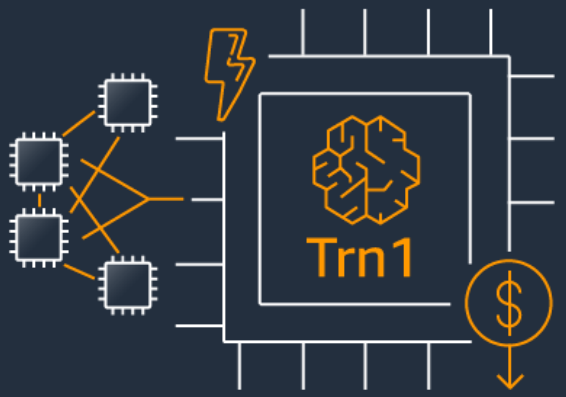
第 2 世代 ML 推論チップ
大規模モデルに対応

NEW
米国リージョンで利用可能
2023年4月 一般提供開始

Amazon EC2 Trn1/Trn1n インスタンス



AWS 独自設計高性能 ML トレーニングチップ AWS Trainium を搭載したインスタンス



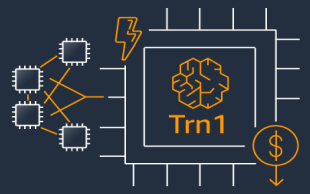
- 同等の GPU インスタンスと比較し **最大50%低価格**を実現
- 最大16個の AWS Trainium アクセラレータ、**512GB の高速 HBM2メモリ**、8TB のローカル NVMe SSDを搭載
- **最大 1,600 Gbps (Trn1n)** の Elastic Fabric Adapter (EFA) ネットワーク帯域
- Trainium 間は超高速 NeuronLink で接続
- Tensorflow、PyTorchなど主要MLフレームワークをサポート
- Trn1上で学習したモデルのデプロイ先は自由

	インスタンスサイズ	Trainium	アクセラレータメモリ	vCPU	メモリ	NeuronLink	ネットワーク帯域	オンデマンド価格 (USD/時間)
	Trn1.2xlarge	**1**	**32 GB**	**8**	**32 GB**	N/A	**最大 10 Gbps**	**1.34**
	Trn1.32xlarge	**16**	**512 GB**	**128**	**512 GB**	**Yes**	**800 Gbps**	**21.5**
NEW	**Trn1n.32xlarge**	**16**	**512 GB**	**128**	**512 GB**	**Yes**	**1600 Gbps**	**24.78**

https://aws.amazon.com/jp/ec2/instance-types/trn1/

*2023年4月時点の米国東部 (バージニア北部)の価格

AWS Trainium: 高性能、低電力、低コストを両立



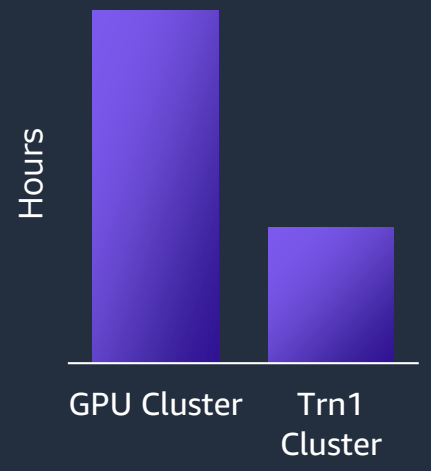
Training BERT Large with AWS Trainium

2.3x
高速なトレーニング

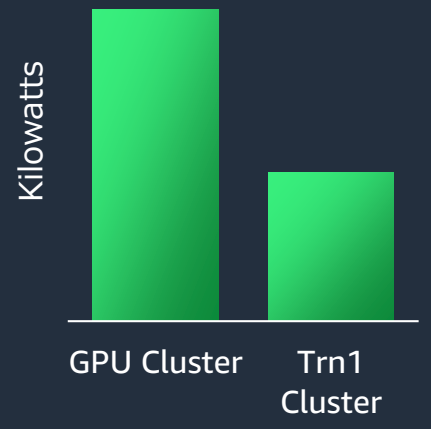
47%
低電力

72%
低コスト

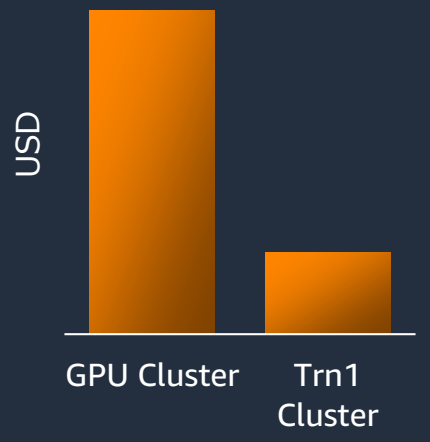
トレーニング時間



電力

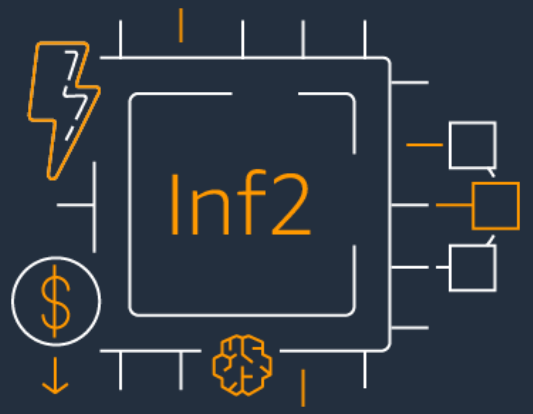


トレーニングコスト



Amazon EC2 Inf2 インスタンス

最もコスト効率の高い 生成系 AI モデルに対応した推論向けインスタンス NEW



- 第2世代 ML推論チップ AWS Inferentia2 搭載
- 最大4倍高いスループット、10分の 1 の低レイテンシー、4倍のアクセラレータメモリサイズ、10TB/s に及ぶ 15倍のメモリ帯域 (vs. Inf1)
- 175B パラメータモデルを単一サーバ上にデプロイ可能
- 大規模 生成系 AI モデルに対応

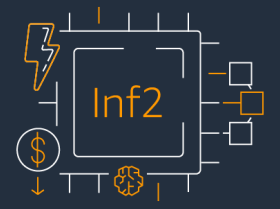
インスタンスサイズ	Inferentia2	**アクセラレータメモリ**	vCPU	**メモリ**	NeuronLink	ネットワーク帯域	オンデマンド価格 (USD/時間)
Inf2.xlarge	1	32 GB	4	16 GB	N/A	最大 15 Gbps	**0.76**
Inf2.8xlarge	1	32 GB	32	128 GB	N/A	最大 25 Gbps	**1.97**
Inf2.24xlarge	6	192 GB	96	384 GB	Yes	50 Gbps	6.49
Inf2.48xlarge	12	384 GB	192	768 GB	Yes	100 Gbps	12.98

https://aws.amazon.com/jp/ec2/instance-types/inf2/

*2023年4月時点の米国東部 (バージニア北部)の価格

AWS Inferentia2 高性能、低電力、低コストを両立

Real-time Deployment BERT-Large with AWS Inferentia2



50%

インスタンス数の削減

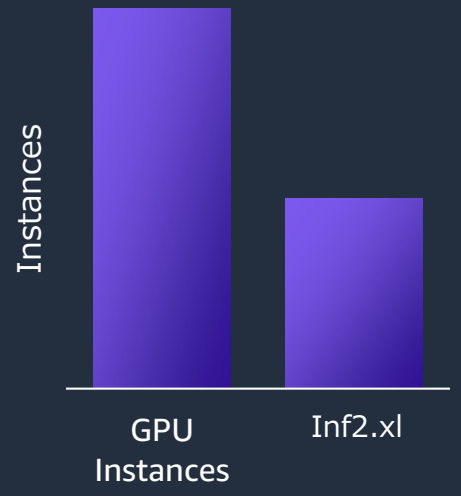
50%

低電力

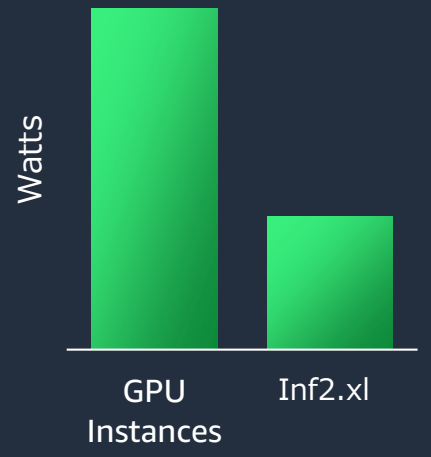
65%

低コスト

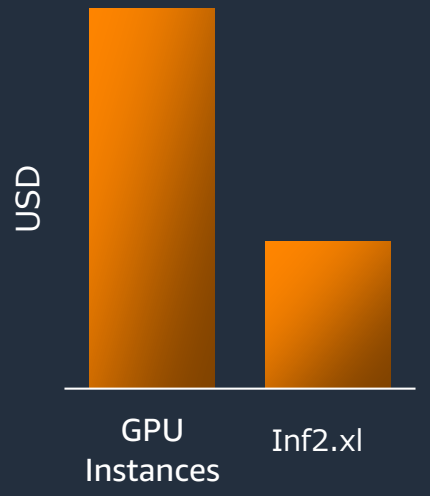
必要なインスタンス数



電力



推論コスト



まとめ

基盤モデルを自社向けに一から構築するための知識を説明

- 基盤モデルの特性
  - 大量のデータと大規模なモデルを必要とするためコストがかかる
  - ビジネスに即したモデルの選定と学習の最適化が必要
- 基盤モデルの構築プロセス
  - 基盤モデルの学習に至るまでにさまざまな検証ポイントがある
  - 本学習の前に学習最適化をおこなう
- 基盤モデル構築に係るAWSのメリット
  - ビジネス・技術両面の支援体制（Machine Learning Solutions Lab）
  - 分散学習を支えるAmazon SageMaker
  - コスト効率に優れた学習/推論専用インスタンス（Trn1/Inf2）

基盤モデルの構築（Tuner型を含む）にご関心のあるお客様はAWSまでご相談ください。