aws DEV DAY
ONLINE JAPAN

D - 3

AWS IoT & Amazon SageMakerを使った魚群食欲解析システムの実現

佐藤 真
ウミトロン株式会社

In Partnership with intel

UMITRON

install Sustainable Aquaculture on Earth

2050年には100億人の人々が生き、
私たちの青い惑星「地球」で進化を
続けます。

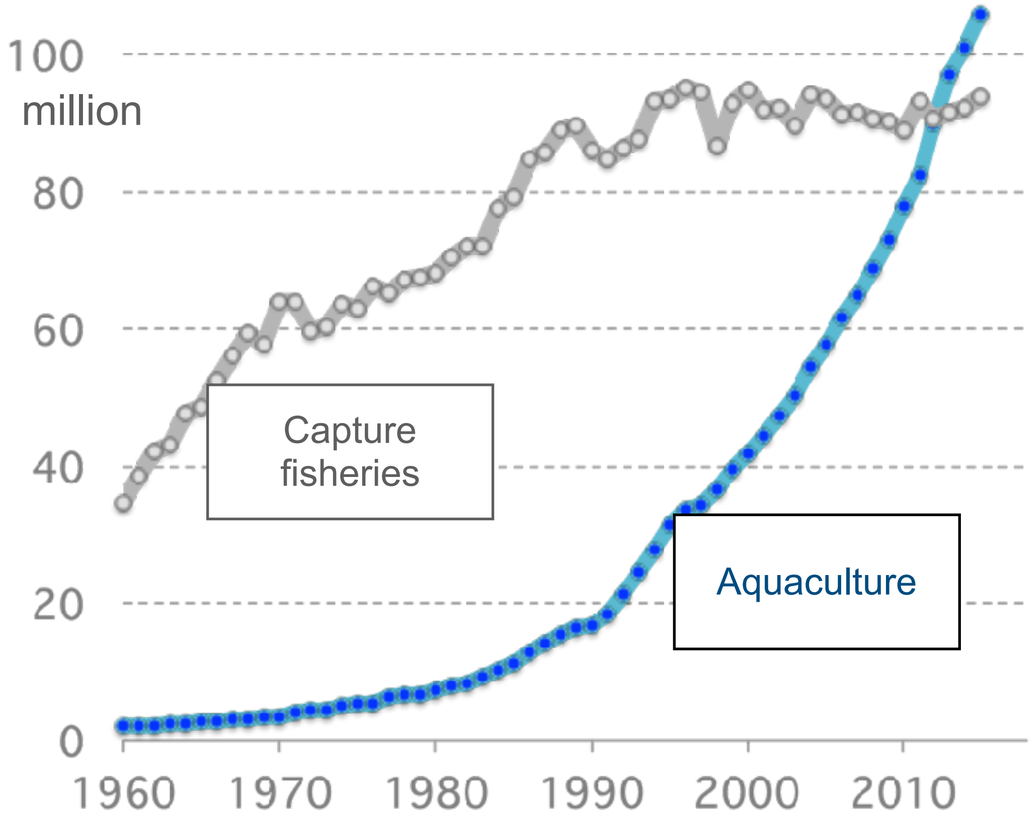
地球の70%は海に覆われていますが

そのほとんどが

まだ利用されていません。

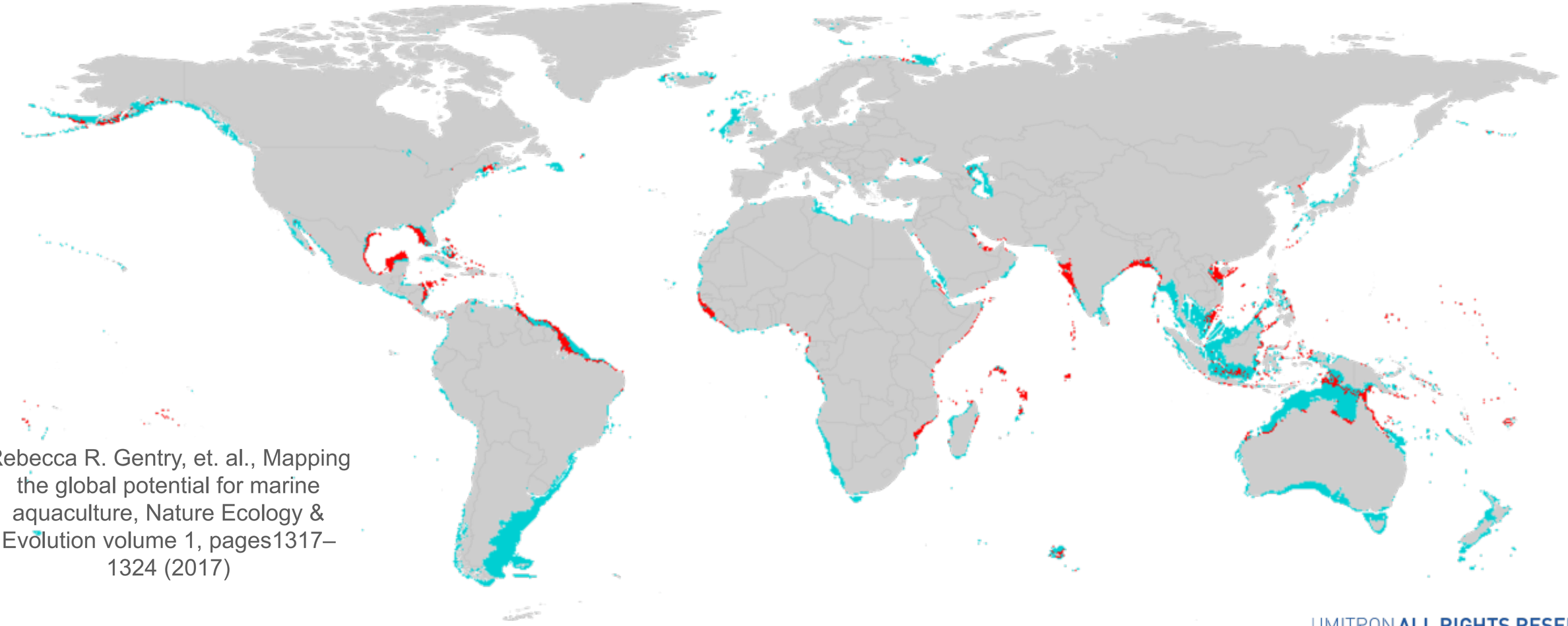
UMITRONの使命は、地球に持続可能な水産養殖を実装し、養殖を次世代の食糧生産方法に移行させることです

水産養殖はアジアを中心としたタンパク需要と中間所得層増加により急成長を続けている



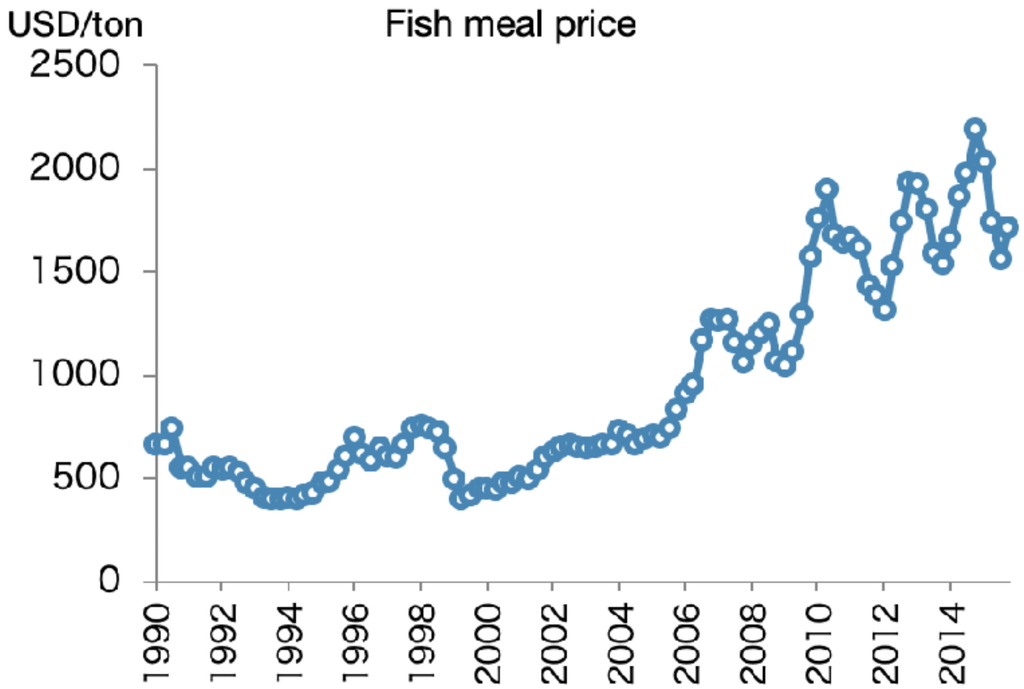
- 人口増加と中間所得層増加により動物性タンパクの需要が急増
- 世界の漁業・養殖業事業者の85%がアジア地域に存在
- 近年は養殖業での雇用の伸びが顕著でアジア地域で1100万人が養殖業に従事

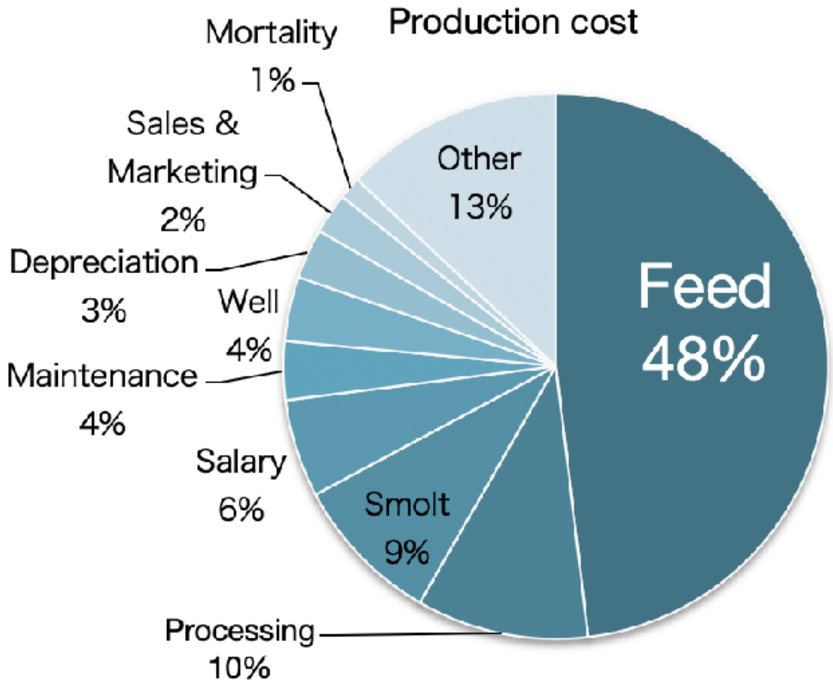
水産養殖は最も将来性の高い食糧生産方法の一つであり、沿岸海域を活用することで現在の世界の水産物消費量の100倍を生産可能

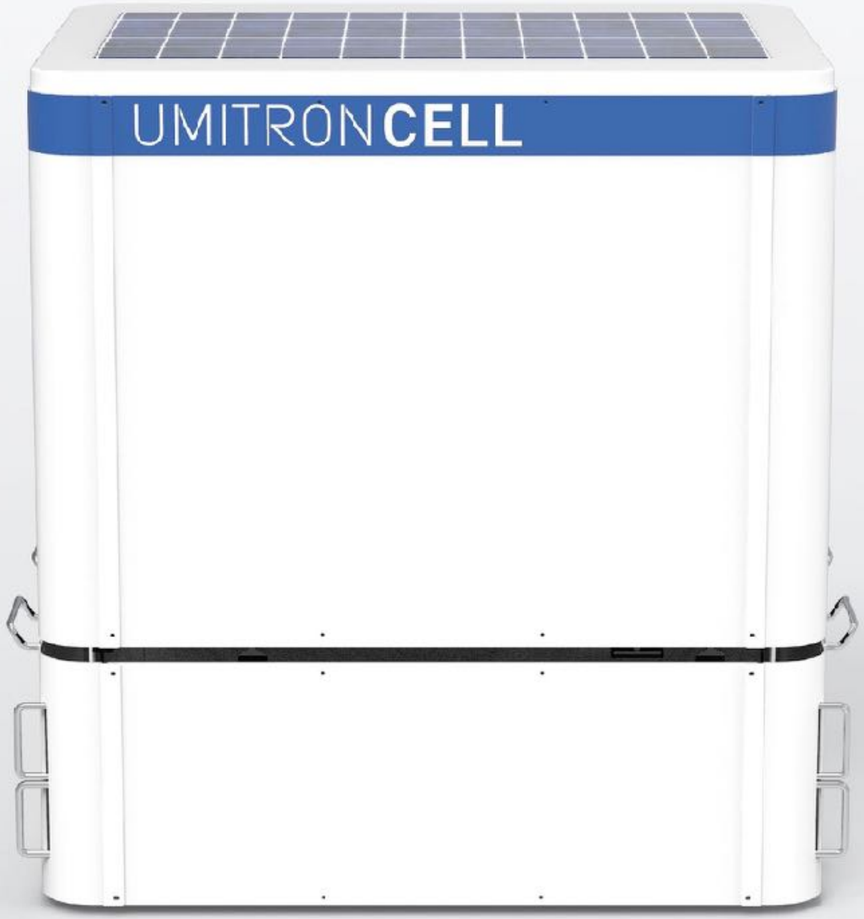


Rebecca R. Gentry, et. al., Mapping the global potential for marine aquaculture, Nature Ecology & Evolution volume 1, pages1317–1324 (2017)

飼の原料である魚粉の価格は15年で3倍に高騰し
飼料コストは総生産コストの50%を占める







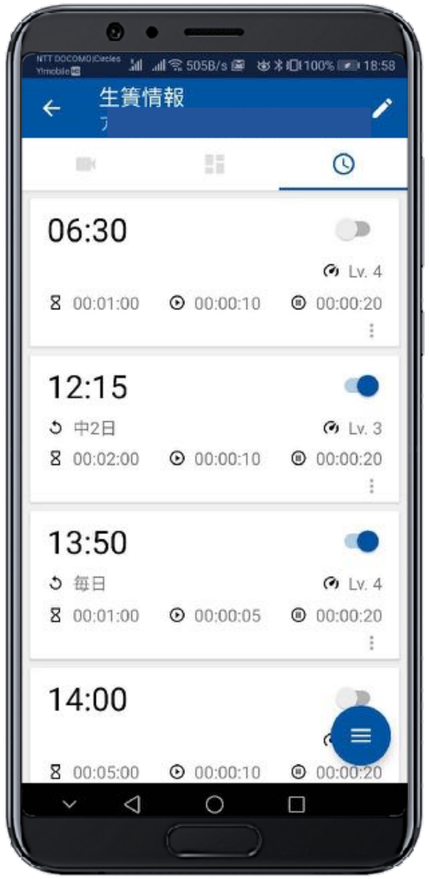
UMITRONCELL

IoTと機械学習を用いた自動給餌器

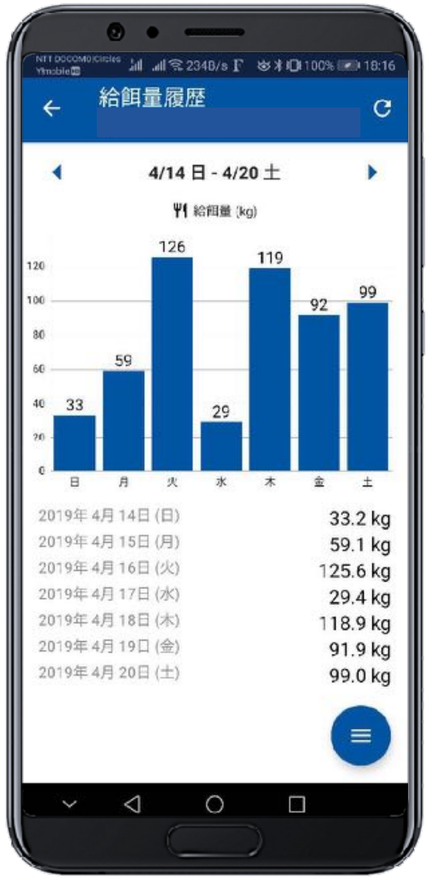

UMITRON

遠隔からの給餌制御とデータの蓄積

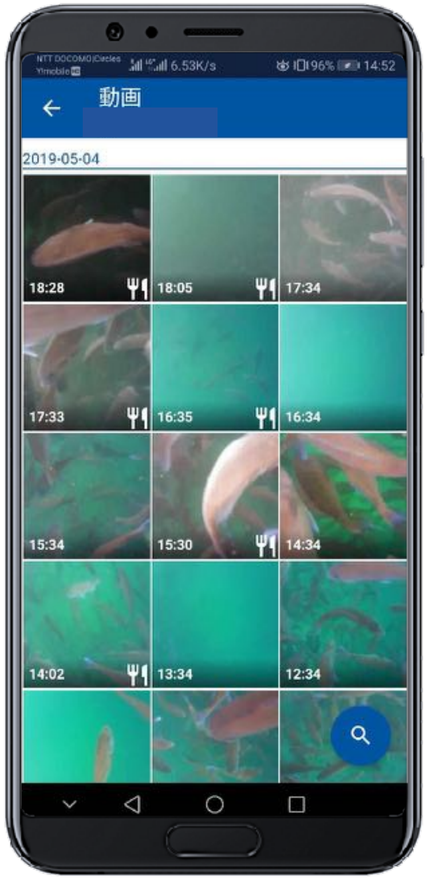
モニタリング



遠隔給餌制御



生育管理



データ蓄積

aws DEV DAY
ONLINE JAPAN

なぜ魚群の食欲を解析するか

In Partnership with intel

給餌をテクノロジーで効率化する

- 魚の摂餌行動から無駄餌の検知と削減
  - 魚の摂餌行動のモデル化
- 人手の制約を受けない魚の生育に適当な給餌
  - 日々変わる魚の食欲に対して適当な量・タイミングでの給餌を実現

無駄餌削減による飼料コストの効率化

給餌効率や成長効率のための給餌制御

魚群食欲解析

食べてないときは無駄な餌を出さない、よく食べているときは食わせこむなど、餌をやったときの魚の食欲に応じて給餌の設定や運用を変える

- タイマーだと動的な調整はできない
- 魚の様子をモニタリングしながら適宜リモートで調節は大変
- 給餌に対する評価をつけて、評価に応じて自動的な給餌のパターン設定できるようにしたい

FAI (Fish Appetite Index, 魚群食欲指数)の開発

aws DEV DAY ONLINE JAPAN

魚群食欲解析～FAIの開発

In Partnership with intel

FAI~Fish Appetite Index~魚群食欲指数とは

ユーザである魚類養殖事業における生産者は、経験的に餌をやったときの魚群の反応などの摂餌行動や餌の残り具合など摂餌行動の結果を見て、お腹が減っていそうか確認している。これを表すモデル・指数を開発。

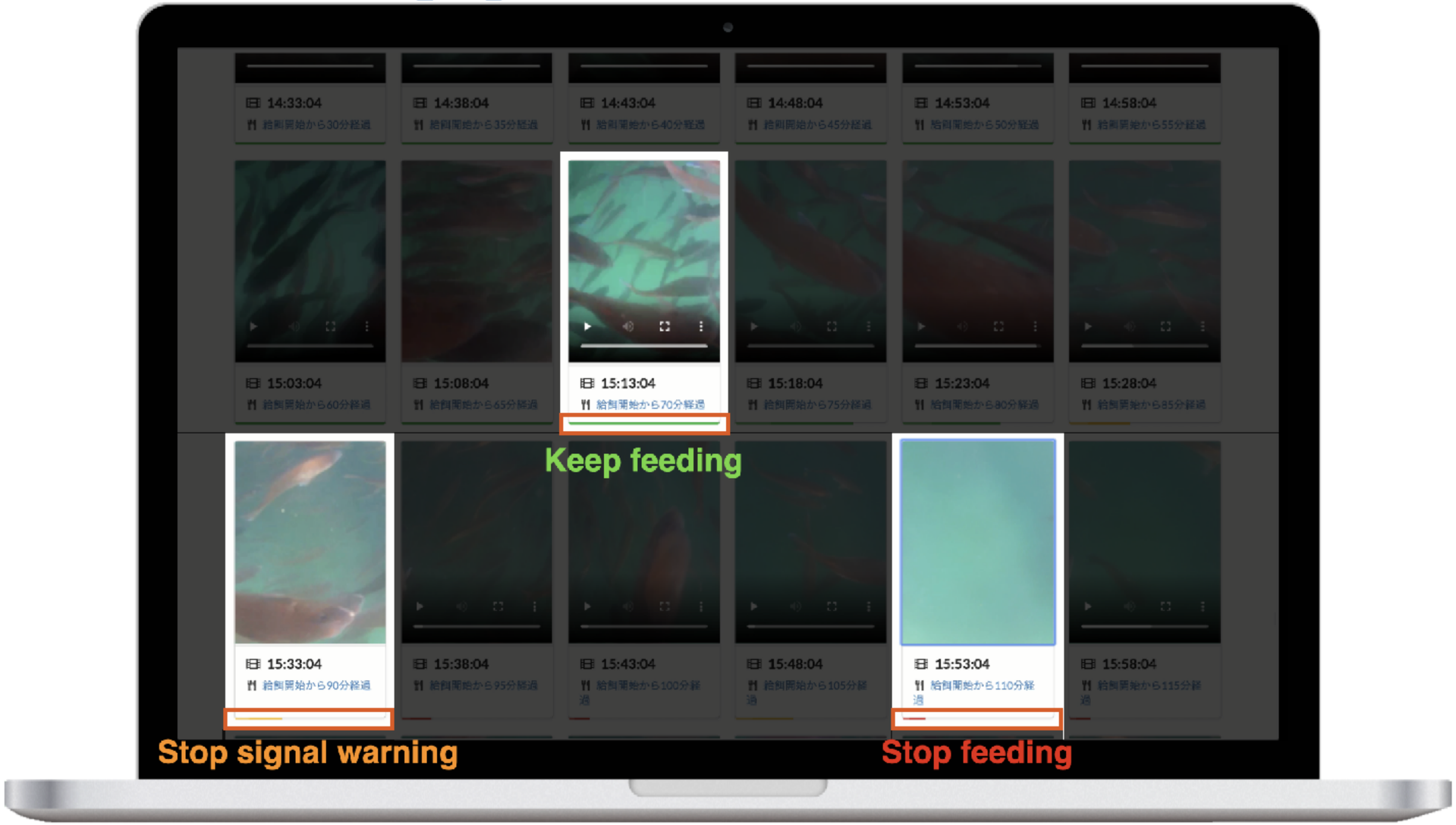
魚群の食欲を示す指数 FAI を開発

Webサービス・アプリを通じてユーザに様々な形で提供

UMITRON FAI

UMITRON FAI

サービスにおけるFAIの例



aws DEV DAY ONLINE JAPAN

AWSでFAIの開発をどのように実現したか

In Partnership with intel

FAIの開発方針

解析からサービスへのパイプラインをつくる

- 解析の結果をアプリなどのサービスで提示したり利用できたりする仕組みを構築
- 改善サイクルを素早く回すため。たとえば、ボトルネックが何になるかを早めに確認できる
- 解析だけやって終わらせない

ユーザからフィードバックを得る

- 新しい機能なので何を出したら正解なのかはわからない
- リリースして実際に使ってもらってわかることが多い

モデルはシンプルにする

- （初期のモデルは特に）改善していくことを前提とする
- 複雑になっていくと保守性が下がっていくので常に注意する

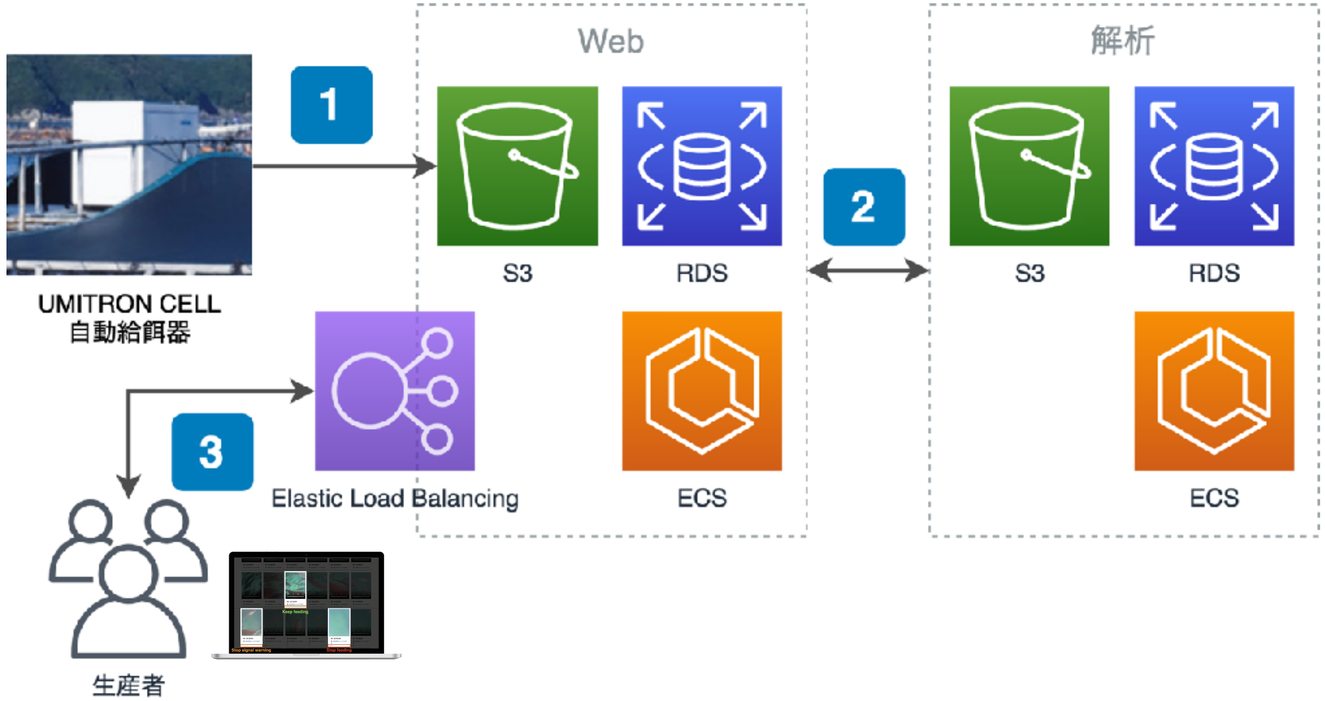
cf. https://developers.google.com/machine-learning/guides/rules-of-ml

初期のFAIの方針

- 給餌器に設置したカメラで撮影できる動画を蓄えていた
  - ユーザ: 生産者が給餌の様子を後から振り返る
  - UMITRON: デバイス設置後にデータが集まる

- 以下の方針で解析を開始
  - 解析からサービスへのパイプライン: サーバサイドで完結
  - モデル: ヒューリスティック。動画から食欲を表す特徴量を抽出、ルールベースでスコアを定義

初期のFAI推論のパイプライン



1. デバイスから動画をS3に送信
2. 解析用のサーバに送ってFAIを計算
3. Webで結果を表示

初期FAIの課題

ユーザはWebでFAIを閲覧可能に。FAIに対するポジティブな評価は得られた。

一方で課題も見つかった。

- FAIの精度を上げたい
  - → ルールベースだと複雑化していきそうなので機械学習を使う
  - → 動画以外の特徴量も追加したほうが良さそう
- FAIを使った給餌の制御をしたい
  - → FAIを得られるまでの時間の短縮する必要あり。エッジも活用すべき

IoTシステムによる機能とデプロイ環境

- リアルタイムな双方向通信
  - IoTデバイスからのデータ取得
  - 遠隔からデバイスの操作
- 洋上での独立・安定した稼働
  - 電源は太陽光のみ
  - ユーザもIoTデバイスの近くにいつもいるわけではない・すぐ行けるわけではない
- デバイスを設置してからも容易に改善を続けられる
  - 設置してから初めて分かる課題にも対応できる
    - さまざまな自然環境
    - 魚の生育データ

エッジでの解析に求められること

厳しいデバイスの環境で、解析機能が給餌・モニタリングなどユーザの操作を邪魔しないようにする。それまでの運用実績を考慮する。

限られた 計算機資源	限られた電力	過酷な熱環境	不安定な ネットワーク

解析で負荷のかかりすぎる処理をしない

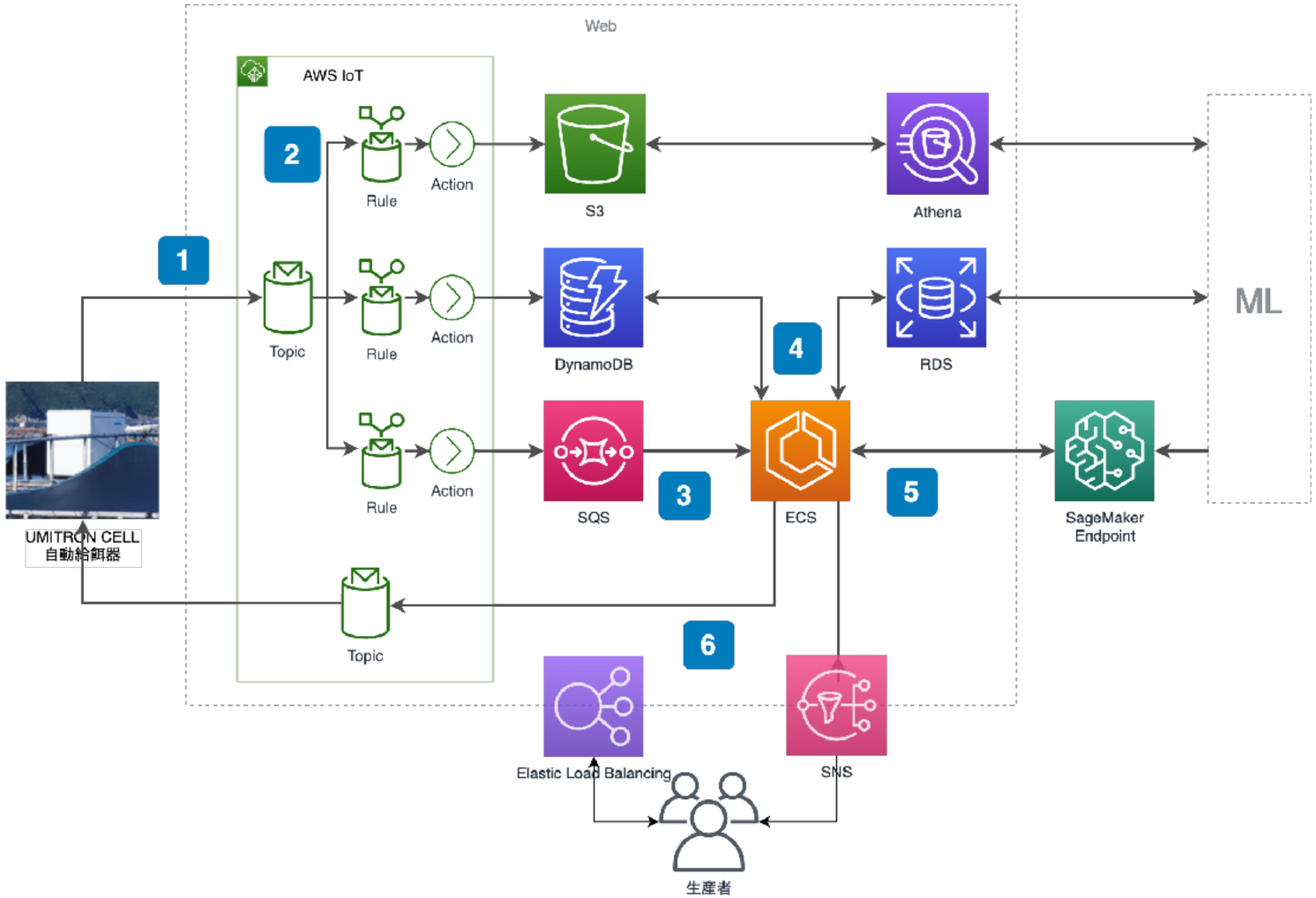
データのやり取りはなるべく軽くする

現行のFAIの方針

- 安定運用に向けて、洋上での運用経験を貯めつつ、基本的な機能に影響を及ぼさないように、処理をサーバからエッジに徐々に移行。

- **エッジとサーバのハイブリッド**
  - エッジ: 動画の特徴量抽出
    - 負荷の小さい画像処理のみエッジで行う
  - サーバ: 動画以外から得られる特徴量追加とFAI推論
    - 運用が安定するまで特徴量・モデルの細かいアップデートを行う

現行の推論パイプライン



1. エッジデバイスで動画特徴量を計算し、特徴量をMQTTトピックにパブリッシュ
2. IoTルールアクションにより以下に送信
   - S3: SageMakerでの学習用データ
   - DynamoDB: 推論用時系列データ。TTLあり。
   - SQS : ECSワーカーのキュー
3. SQSからECSのワーカーでタスク実行
4. DynamoDB、RDSから入力データを収集
5. SageMaker EndpointでFAIを推論
6. 推論されたFAIに応じて
   - 結果の履歴をWebで表示
   - SNSを通じてユーザへの通知
   - IoTを通じて給餌操作

これからのFAI

- 安定運用の実現のため運用実績を貯めつつ、難しい環境に対して適用できるように進化させる
  - エッジ側で解析の処理を完結させるアーキテクチャの構築も予定
- 副次的に環境に合わせてアーキテクチャを選択できるようになっている
  - 全く未知の環境での解析は、手始めにヒューリスティックを使った解析を実装してユーザに提供してみるなど

aws DEV DAY ONLINE JAPAN

UMITRONにおけるSageMaker

In Partnership with intel

UMITRONにおけるSageMakerの利用

- アノテーションデータは社内で開発したツールで作成
- データセット作成〜運用までSageMakerに頼っている

UMITRONがSageMakerを採用した背景

- 複雑な環境による開発の難しさ
  - 学習環境が統一されていない
    - ローカルPCとAmazon EC2 spot instance
  - 訓練データの管理方法が統一されていない
  - デファクトな構成でないため学習コストが高い
    - Webエンジニア、新入社員のキャッチアップコスト

- 機械学習サービスのインフラ運用の難しさ
  - Webサービスの負荷を機械学習サービスが捌けない
    - 機械学習サービスのパフォーマンスがユーザに影響を与える
    - 機械学習エンジニアだけインフラ運用できない

SageMakerを機械学習システムの基盤に採用

- フレームワークとしてのAmazon SageMaker
  - 統一された構成
  - 学習コストをAmazon SageMakerのみに集約
  - Webエンジニア、新入社員がキャッチアップしやすい
- フルマネージドなインフラ
  - インフラ運用コストの削減
  - 機械学習エンジニアでも運用できる
- Webサービスと機械学習サービスを疎結合にする
  - Amazon SageMakerのendpointで2つのサービスを結合
  - Amazon SageMakerの知識だけで互いを運用できる

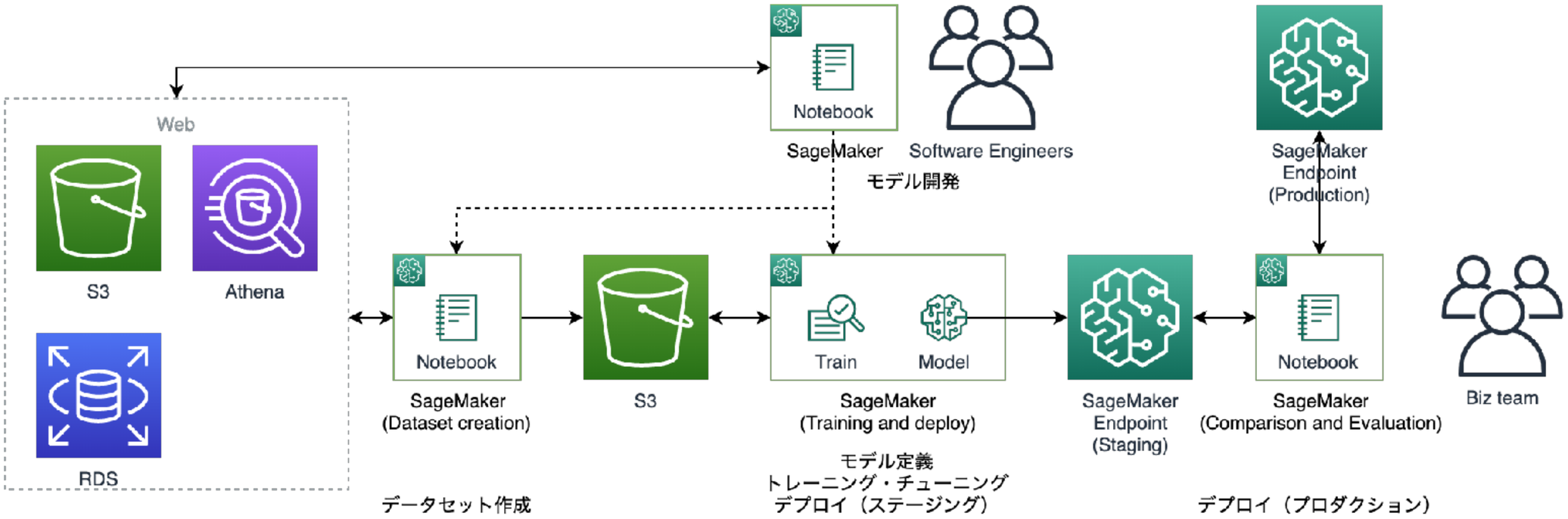
SageMakerを採用した効果

- 開発環境の簡素化
  - 学習環境が統一
  - 必要なインフラ知識をAmazon SageMakerに集約
  - Amazon SageMakerさえわかれば誰でもわかる
  - 開発環境で社内特有の知識が少ない
- スケールが容易
  - インフラ運用コストが少ない
  - Webサービスが受ける影響がAmazon SageMakerのパフォーマンスのみに限定される
  - 機械学習エンジニアだけでインフラをスケールできる

チーム開発でSageMakerを採用する際注意したこと

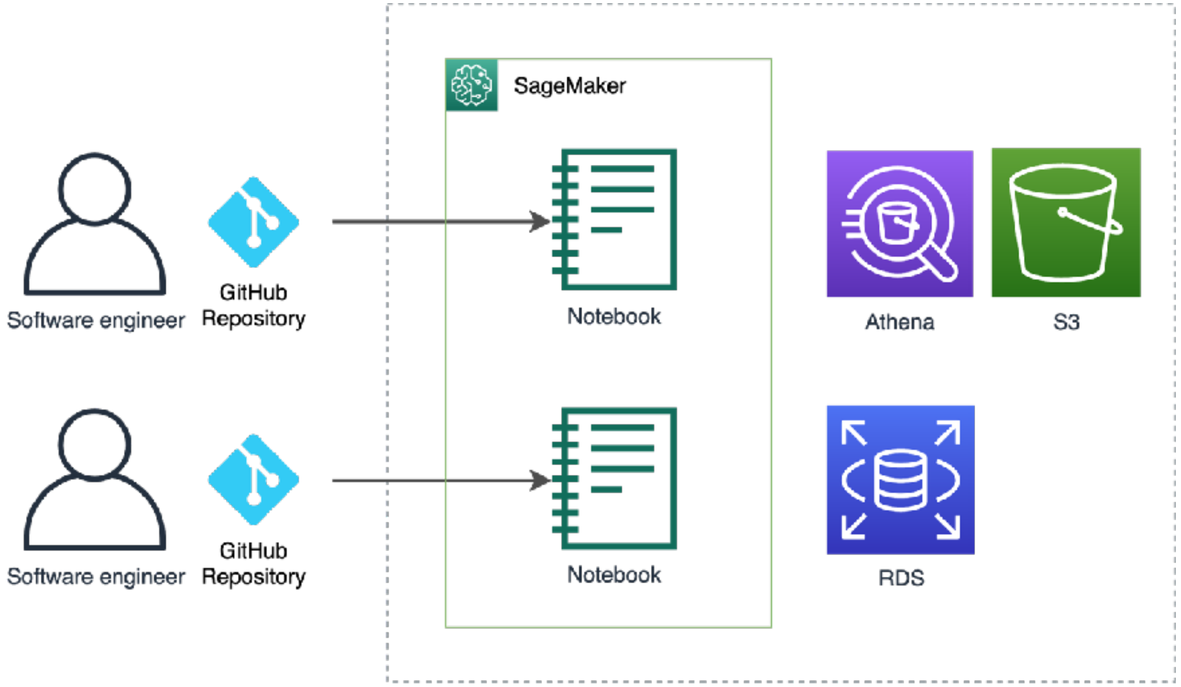
- モデル開発は自由に
  - EDAやモデル開発の試行錯誤に関しては個人毎にGitレポジトリ + ノートブックインスタンスを割当
- サービス開発は厳格に
  - コード管理、レビュー
    - 本番にデプロイするモデルの定義
    - トレーニング・チューニング・デプロイのプロセス
  - 一貫した命名規則・場所
    - レポジトリ、データセット、モデル、エンドポイント…
      - レポジトリ・リソースの名前、データセット・モデルのS3のBucket, Prefix

機械学習のパイプライン



- データセット作成、モデル定義、トレーニング・チューニング、デプロイまでコードで管理
- 複数人でレビューして、品質を保つ
- UMITRONのサービスとして利用するモデルは一つのGitHubで管理
  - 他のプロダクトで使う機械学習に関する複数のモデルも管理

モデル開発の環境



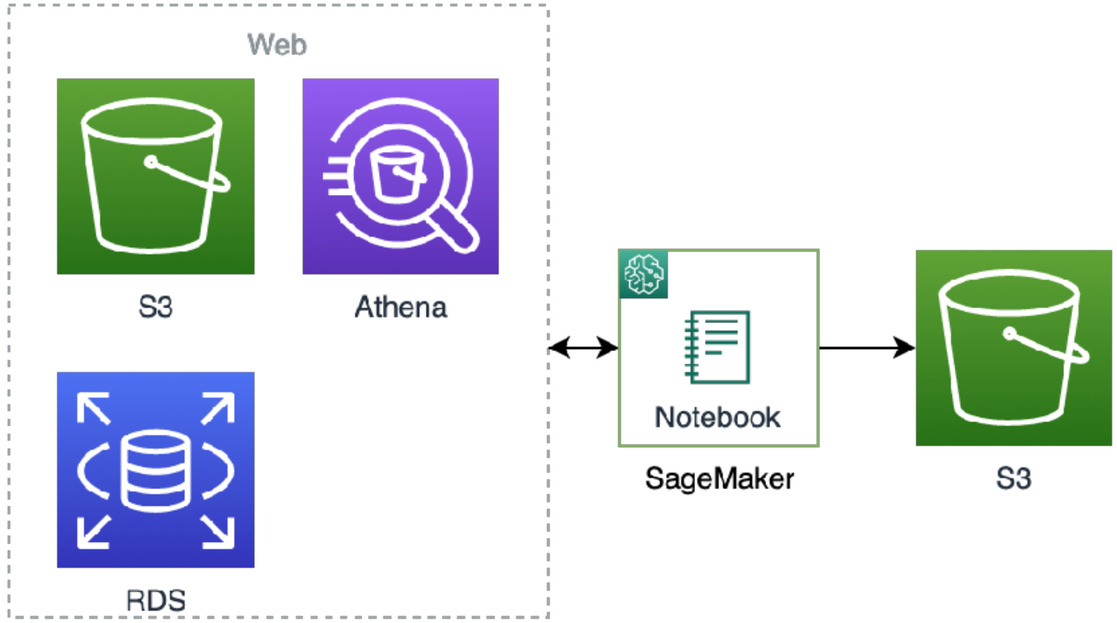
自由度の高くモデル開発やEDAのできる環境の必要性

- 多様なバックグラウンドを持つソフトウェアエンジニアは自由に解析できる環境で力を発揮する
- FAI以外の開発も行っている
- 解析過程や結果は共有するようにしたい

SageMakerにおけるモデル開発環境の準備

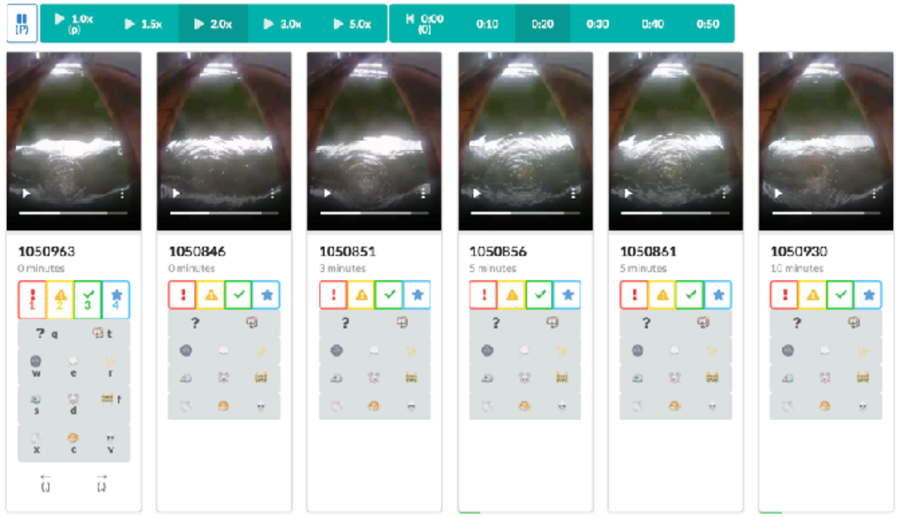
- 解析に携わるソフトウェアエンジニアにGitHubでPlayground用のレポジトリを作成
- レポジトリをNotebookインスタンスに紐付け
- アプリやエッジで集められたデータを元にEDAやMLモデル試作
- 結果はNotebookやコンフルエンスで共有するなど

データセットの作成



- データセットの作成はコードで管理
  - S3 + Athena, RDSからデータを収集、SageMakerで利用するための適切な形式でS3に保存する
  - 作成したタイミング毎にフォルダを作って配置
  - 一度作ったら修正しない。修正したいときは新しく作る

アノテーションデータ



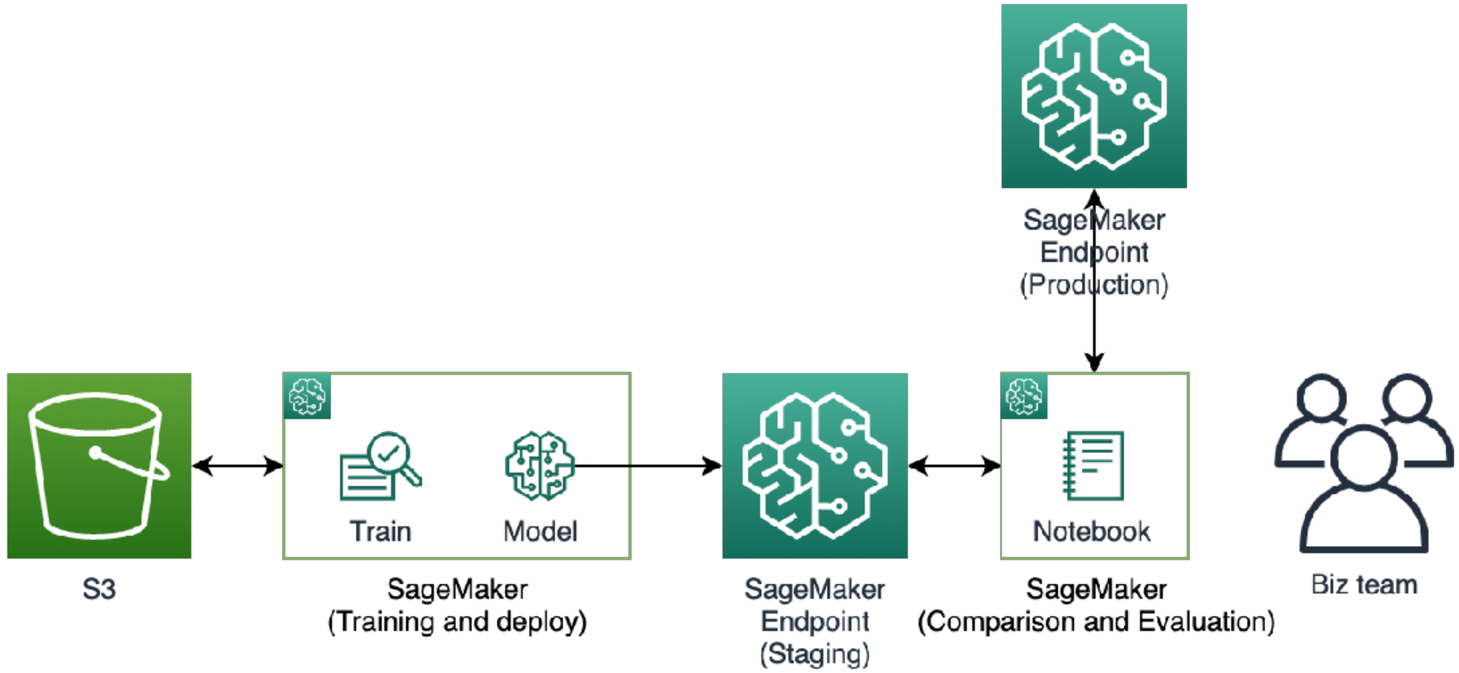
社内アノテーションツール

ツールをWebに実装し、社内やアルバイトの方にお願いしている。明確に基準を言語化できたら外部サービスの利用も検討

FAIのアノテーションの難しさ

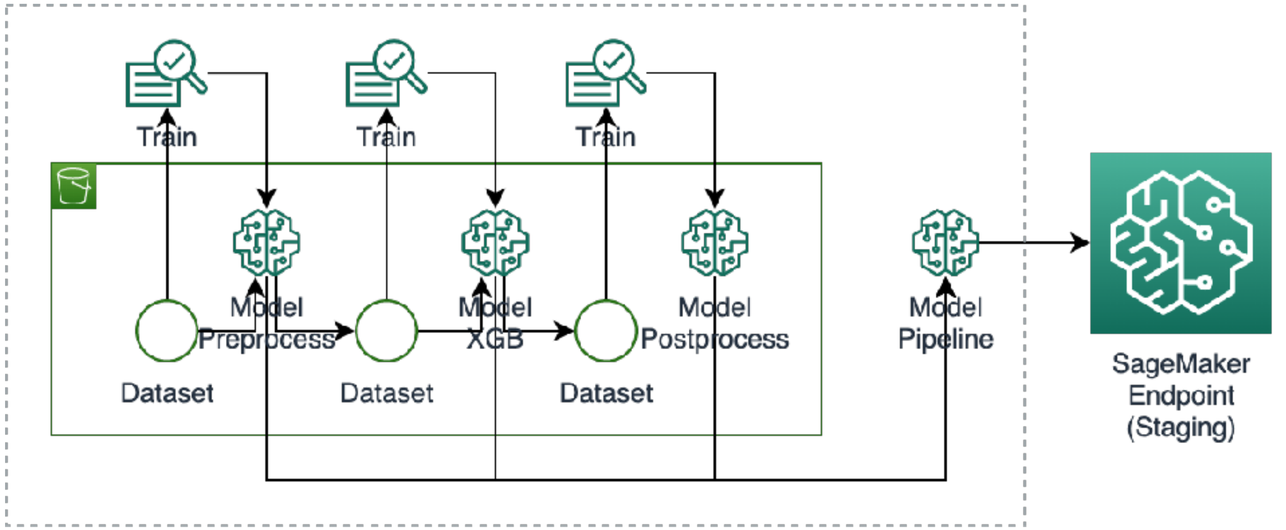
- 生産者ごとに異なる基準であり、そもそも「正解」を定義するのが難しい
- 生産者の協力の元、ウミトロンで給餌を分析、社内で知見を蓄積し、形式知化を進めている

トレーニング・チューニング〜デプロイ



- 定義されたモデルを利用してトレーニング・チューニング〜デプロイ
  - フロー自体は崩さずに、各プロセスでは新しいSageMakerの機能などをどんどん試す
  - StepFunctionsはAWS Step Functions Data Science SDK for Amazon SageMakerを利用するなど
- デプロイは、ステージングのSageMakerエンドポイントまで出して、新モデルと旧モデルの比較、結果をBizチームに共有、確認後プロダクションに反映

モデルの定義



- FAI推論ではSageMakerの推論パイプラインを利用するなど
  - 推論パイプライン: データに対する推論要求を処理する複数のコンテナの順番で構成される Amazon SageMaker モデル
  - 推論は生のテーブルデータを入力。前処理は推論パイプラインの最初のコンテナで行う。
    - 利用する側のWebサービスをシンプルに保つため、特徴量エンジニアリングのような処理はSageMakerに寄せる
  - SageMaker組み込みモデルも活用
  - 他のモデルへの切り替えも容易

まとめ

- 魚群食欲指数FAIの開発
  - 改善速度を解析からサービスへのパイプラインを作る、フィードバックを得るためにユーザに出す、まずはシンプルなモデルで運用する
  - 安定運用を行うためサーバからエッジに解析処理を徐々に移行。過酷な洋上で稼働させるデバイスで、電力・熱・ネットワークなど制約多いため、ユーザ操作や給餌機能になるべく影響しないように運用実績を貯めている

- FAI開発のためSageMakerのUMITRONにおける事例紹介
  - SageMakerを採用し、解析に関わるプロセスの改善サイクルを加速化
  - モデル開発は自由に、サービス開発は厳格に

aws DEV DAY
ONLINE JAPAN

Thank you!

佐藤真
ウミトロン株式会社

In Partnership with intel