

IT・AI を活用した食品安全情報収集システムに関する実証

事業

A)作業報告書

(イ) 次期システムへの提案

令和7年3月31日

みずほリサーチ&テクノロジーズ株式会社

情報通信研究部

目次

1. 次期システムへの提案	1
1.1. システム基盤.....	1
1.1.1. 全体概要.....	1
1.1.2. 接続方法.....	1
1.1.3. ユーザー管理	2
1.1.4. 保守運用.....	2
1.1.5. リージョンと LLM の利用.....	2
1.2. 記事作成プロセス	3
1.3. 記事原稿収集機能	4
1.3.1. 記事原稿収集	4
1.3.2. スクレイピング等による記事原稿自動収集機能.....	4
1.3.3. スクレイピングの実行	7
1.3.4. 手動収集による記事原稿入力機能	10
1.4. 翻訳・要約	11
1.4.1. LLM を用いた要約	11
1.4.2. LLM を用いた翻訳	12
1.5. データの蓄積・分析.....	13
1.5.1. データベースの構築.....	13
1.5.2. キーワード抽出と記事データの分析.....	13
1.6. ドキュメントからの構造化抄録データベースの構築.....	14
1.6.1. 構造化抄録の目的	14
1.6.2. 記事カテゴリーの洗い出しと抽出項目設定	14
1.6.3. 記事の自動分類付与と項目の抽出	15
1.6.4. 食品安全記事チャットボットの構築.....	16
1.7. 次期システム構築の課題.....	17
1.7.1. 検討対象項目	17
1.7.2. 記事原稿自動収集対象調査.....	17
1.7.3. 要約の最適化検討	19
1.7.4. 記事データ解析手法調査.....	20
1.7.5. 食品安全に関する構造化項目データベース構築のため情報整理と技術課題調査.....	20

1. 次期システムへの提案

「IT・AI を活用した食品安全情報収集システム」のプロトタイプシステムを構築し、その効果を検証した結果から次期システムへの提案を記した。

1.1. システム基盤

1.1.1. 全体概要

デジタル庁が整備するガバメントクラウドへの政府系システムの移行が本格化している中、IT・AI を活用した食品安全情報収集システム(本番システム)もガバメントクラウド上に構築されることになると考えられる。ガバメントクラウドでは、デジタル庁から提供されるセキュリティー関連のテンプレートの上に構築することになり、保守やセキュリティー対策がクラウド事業者の責任になるマネージドサービスを利用してシステムを構築することが推奨される。本事業で構築したプロタイプシステムは Amazon Web Services¹ (AWS)上のマネージドサービスを利用して構築しており、本事業で得られたノウハウは直接ガバメントクラウド上で本番システムを構築する参考になると思われる。

本事業で AWS 上に構築した構成の概略を図 1-1 に示した。本事業はプロタイプシステムのため、迅速な構築・変更が可能なマネージドサービスを利用しており、本番システムを構築するには性能的に置き換える必要があるサービスが幾つか存在している。次節以降に本番システムに求められると思われる要件定義に対応するため差し替える可能性があるマネージドサービスについても説明を加えた。

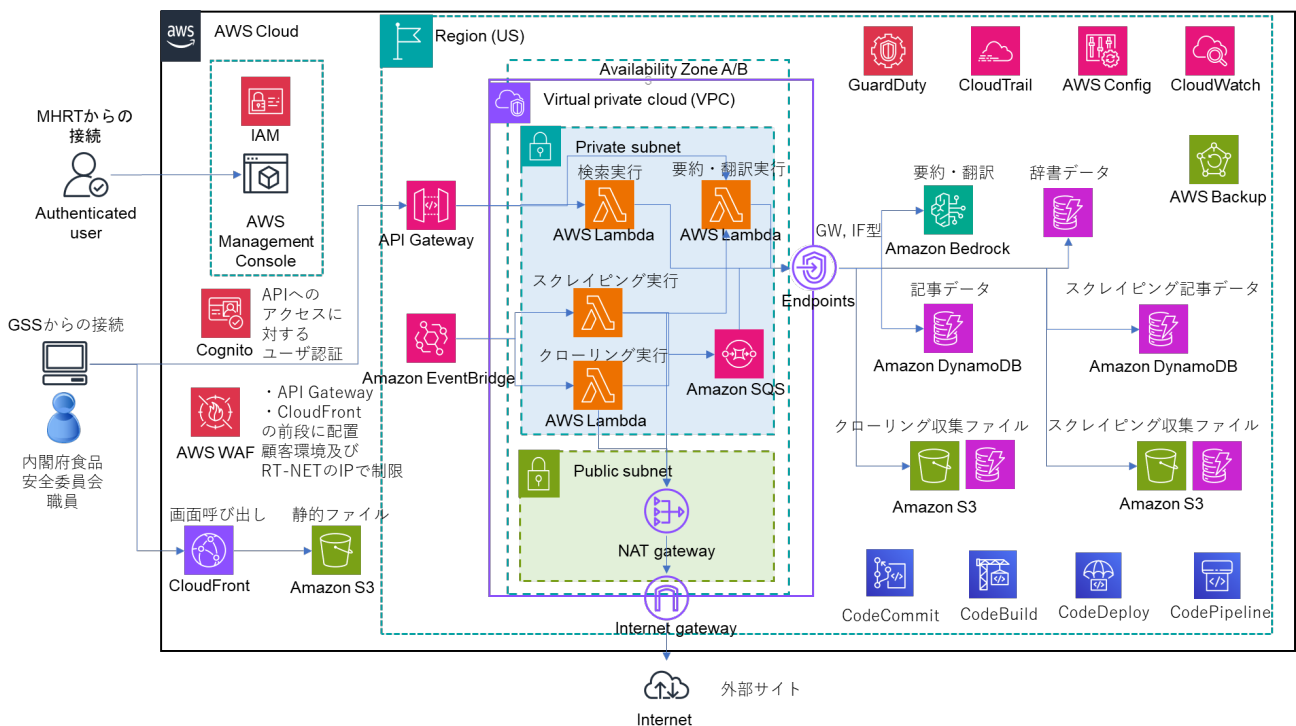


図 1-1 本事業で構築した AWS 上のプロタイプシステムの概要

1.1.2. 接続方法

本番システムへのユーザーからの接続方法としては、Transport Layer Security (TLS)/ Secure Sockets Layer (SSL)もしくは Virtual Private Network (VPN)を利用した接続になると考えられる。例えばガバメントソリューションサービス(GSS)提供ネットワークからは簡便な TLS/SSL 接続とするが、

¹ <https://aws.amazon.com/jp/>

GSS に接続出来ないユーザーや運用開発事業者からのアクセスを許可する場合は VPN 接続とする等が考えられる。TLS/SSL を利用する場合 TLS/SSL 証明書のレベルや証明書発行にかかる費用なども検討しておく必要がある。

1.1.3. ユーザー管理

プラットフォームが AWS の場合、本番システムにおいても Cognito を利用してユーザー認証を行うと思われる。ユーザー認証後にセッション管理を行い、ユーザー情報の保存やユーザー毎のカスタマイズ機能などを構築するなどが考えられる。

本番システムの運用ではユーザーID を発行する管理者の決定方法や、権限の範囲などを決めておく必要がある。またユーザーが ID やパスワードを失念した場合のルール作りなども必要になると思われる。

1.1.4. 保守運用

ガバメントクラウド上のマネージドサービスを利用して構築すると、様々なアラート情報が発信される。アラートの送信先やアラートに対応するためのメンテナンス体制などを決めておく必要がある。

1.1.5. リージョンと LLM の利用

ガバメントクラウドにおける AWS では、「東京/大阪リージョン以外の使用禁止」となっている。最新の大規模言語モデル(LLM)は多くの場合アメリカリージョンでのみ利用可能であり、2024 年 7 月からようやく東京リージョンでも Claude 3 が使えるようになった。また AWS Marketplace の利用も制限しており、Amazon Bedrock Marketplace の利用には別途申請が必要になる可能性もある。最新の LLM を用いた性能評価等を行う場合はガバメントクラウド上では行わないなどの工夫も検討する必要があるかもしれない。

1.2. 記事作成プロセス

食品安全に関する記事作成では、

1. 記事原稿の収集
2. 記事原稿の翻訳・要約
3. 記事作成

が基本的な流れである。この作業を効率化するプロセスとして、

- ① 記事原稿の自動収集
- ② 記事原稿の自動要約

の検証を本事業では行った。

提案する次期システム(本番システム)の記事作成の流れを図 1-2 に示した。記事原稿の収集については 1.3 節、記事原稿作成機能での要約や翻訳は 1.4 節に提案を示した。また記事 DB については 1.5 節に、記事データの解析の高度化のための構造化抄録データベース構築のための技術的課題については、1.6 節に提案をまとめた。

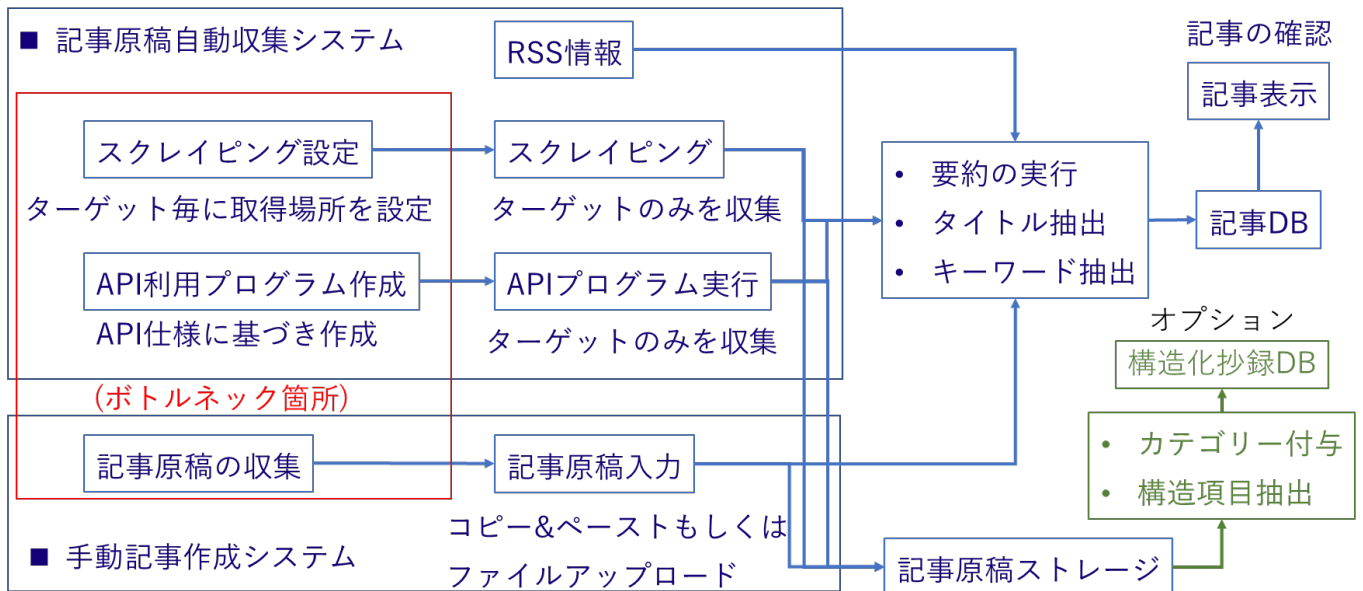


図 1-2 記事作成プロセスの流れ

1.3. 記事原稿収集機能

1.3.1. 記事原稿収集

本事業において、クローリングでは食品安全に関する記事作成の効率化には繋がらなかったためスクレイピング等の記事原稿自動収集機能が必要となる。しかしながらスクレイピングでは実行プログラムの構築と保守に費用がかかるため全ての調査対象 web サイトをスクレイピングするのは費用対効果の面から検討を要する。またプログラムからのアクセス禁止サイトも存在するためスクレイピングのみで本番システムを構築することはできない。そこで

- ① スクレイピング等による記事原稿自動収集機能
- ② 手動収集による記事原稿入力機能

の 2 つのシステムを構築することで記事作成の効率化を図る。

1.3.2. スクレイピング等による記事原稿自動収集機能

1.3.2.1. スクレイピング対象の選定方法

スクレイピングは対象 web サイト毎にプログラムの設定・調整が必要になり、且つ対象 web サイトの構成が変わるとプログラムの修正が必要になる。本事業においても EFSA のページング指定の方法が期間中に変更され、プログラムの修正を行っている。スクレイピング対象の選定においては、費用対効果を考慮する上で、

1. アクセス頻度と記事数
2. プログラムからのアクセス可能性
3. API の利用可能性
4. RSS の利用可能性

らの検討を実施する。

1.3.2.2. アクセス頻度と記事数

食品安全委員会で探索対象として提供された URL 数は 236 レコードであった。この内毎日確認を行っている URL は 42 であり、作成される記事数にはサイト毎に差があると考えられる。スクレイピングの構築にかかる費用と保守のコストを勘案し、スクレイピング対象とするのかどうかを検討することが考えられる。

1.3.2.3. プログラムからのアクセス可能性

スクレイピングを実施するにはプログラムからのアクセスが許可されていることが必要になるが、サイトにこの情報が埋め込まれている場合もあれば、規約のページに記載されている場合、何も情報が公開されていない場合がある。スクレイピング対象候補サイトについては、これらを調査する必要がある。また Distributed denial of service (DDoS) 攻撃への対策や、LLM 等を用いた 2 次サイトの構築対策としてロボットによるアクセス禁止サイトは今後も増えていくと考えられ、定期的に再調査を実施することが必要になる。

1.3.2.4. API の利用可能性

人が目視で使用する想定 of web サイトへのスクレイピングに比べて、application programming interface (API) が提供されている場合は、API の仕様に基づきスクレイピングを実行するプログラムを作成することが出来る。本事業において記事収集対象となる web サイトの API の提供状況を調査した

が、リコール情報などを提供する API はあるが記事収集に使える API の提供は見つからなかった。また API の提供があったとしても API の仕様を調べ、対象毎に API 利用によるスクレイピングプログラムを構築する必要がある。

学術論文 web サイトの場合、多くがプログラムからのアクセスを禁止していると思われるが、学術目的利用でのアカデミックユーザーに対しては API 利用を許可制で提供している場合がある。例えば EFSA Journal を提供している Wiley Online Library はクローリング、スクレイピングを規約の上で禁止しているが、申請して審査に通れば API を利用して文献を収集することが可能である。許可制 API の申請は、プログラムからのアクセス禁止サイトに対する自動収集の方法の一つとなるだろう。

1.3.2.5. RSS の利用可能性

RSS²に対応している web サイトは、スクレイピングが容易になる場合がある。本事業では EFSA の対象 web サイトについて一つ一つ確認を行い、スクレイピングスクリプトを構築したが、EFSA から提供されている RSS feed を利用すると XML ファイルをダウンロードするだけで良いことが分かった(図 1-3 参照)。これは RSS の description に abstract が記載されているためである。

プログラム(ロボット)アクセスが禁止されているサイトにおいても RSS に対応している場合(表 1-1)、RSS に含まれている description 情報で記事化出来る可能性があり(RSS はプログラムからのアクセスを前提として公開している)、食品安全記事作成で必要となる情報が全て含まれているのか確認しておく方法がある。

表 1-1 プログラムアクセス禁止サイトにおける RSS の提供状況

機関	RSS
米国食品医薬品庁 (FDA)	あり
米国食品安全検査局 (FSIS)	あり
フランス競争・消費・不正抑止総局 (DGCCRF)	あり
ドイツ連邦消費者保護・食品安全庁(BVL)	なし
オーストラリア農薬・動物用医薬品局(APVMA)	なし

² Really Simple Syndication、Rich Site Summary、RDF Site Summary など、様々なフォーマットバージョンがある

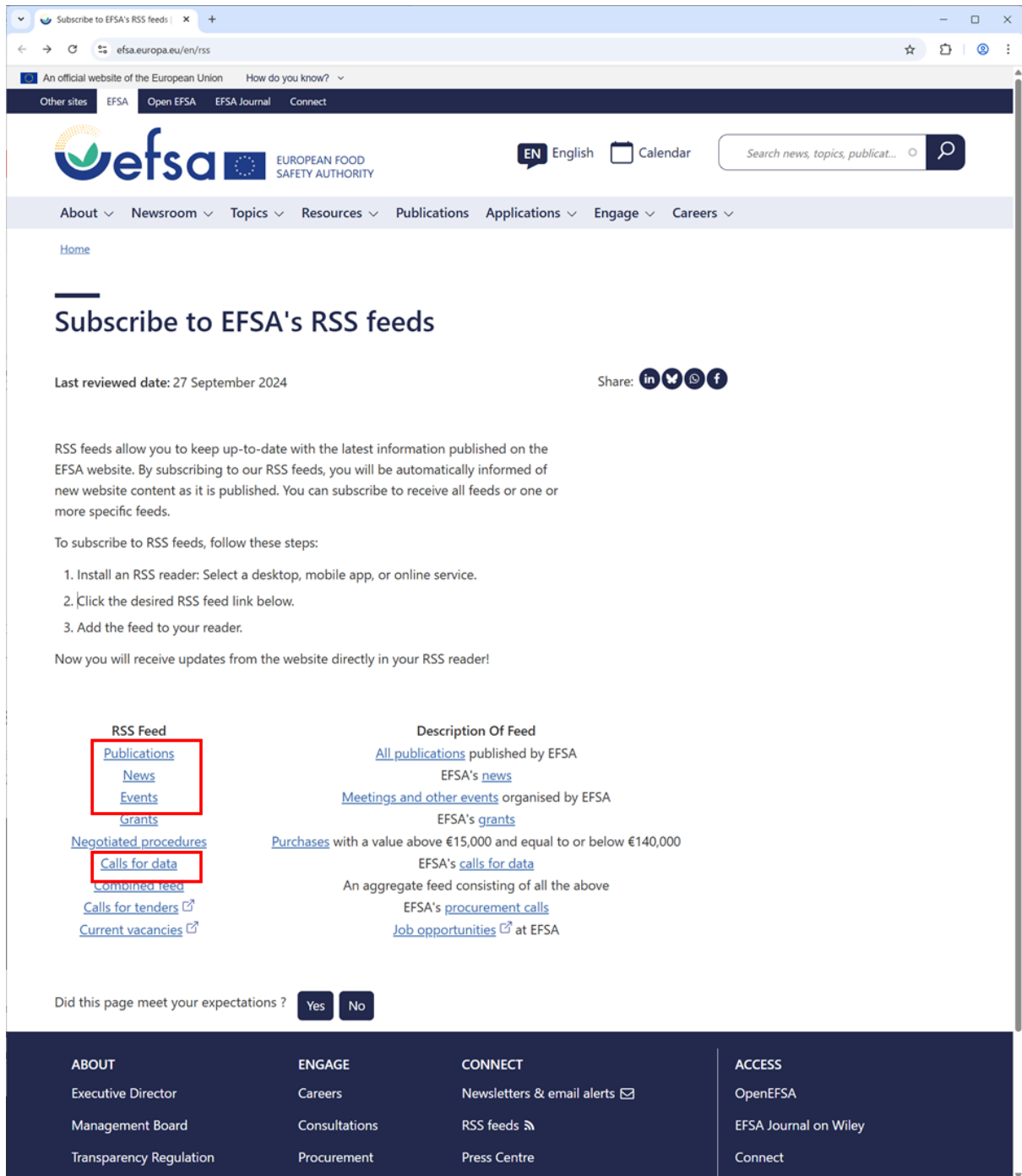


図 1-3 EFSA の RSS feeds 説明サイト

1.3.2.6. 国際的な協調

食品安全情報の提供は各国や地域でも必要なことであり、この労力の削減も共通して望まれていると考えられる。そこで食品安全に関する情報を提供・収集している世界各国の機関と連携し、共同で食品安全情報に関する記事の国際的なレポジトリを構築することが出来ると便利である。このレポジトリサイトが構築され、機能して運用されればスクレイピング自体が不要になる可能性がある。

1.3.2.7. 記事収集手法の選択方法

以上から、記事収集方法を選択するフローチャートを図 1-4 に示した。記事収集の方法としては、

- ① RSS 情報からの自動記事化
- ② スクレイピングプログラムからの自動記事化

③ API 利用プログラムからの自動記事化

④ 手動記事収集

の 4 つとなる。本番システム構築までに、RSS への対応、プログラムからのアクセス許可、許可制 API の提供(と申請)、スクレイピングプログラムを実装する経済性について、調査・検討しておくことが望ましい。

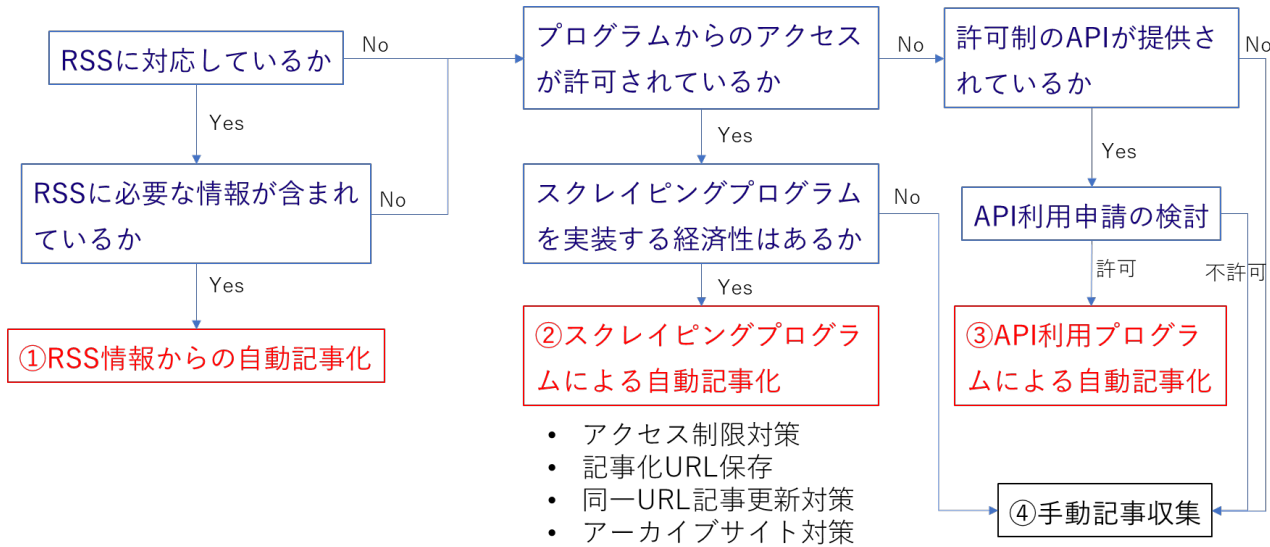


図 1-4 記事収集方法選択のフローチャート

1.3.3. スクレイピングの実行

1.3.3.1. RSS 情報からの実行

RSS の XML に含まれる情報のみで記事化可能な場合は、XML ファイルをパース(parse)³して各記事情報分割にしてデータベースに登録する。仕様上 RSS に大きな記事情報が入ることはないと考えられるため大きな課題は想定されない。

1.3.3.2. パブリッククラウドのマネージドサービスからのスクレイピングの実行

本事業では AWS を利用したため、AWS 上のマネージドサービスである Lambda を使ってスクレイピング実行した。Lambda には 1 プロセス 15 分以内の制約があり、スクレイピング実行時に翻訳・要約を行うと 15 分以内に記事収集が終わらないことが発生することがある。例えば US EPA の記事更新では、100 ページを超える PDF の報告書が一度に 10 報程度公開されたことがあり、この翻訳に時間がかかりスクレイピングプロセスが終わらず、記事データベースへ登録が出来ずに記事収集に失敗した。そのため Lambda の様なサービスを利用する場合、スクレイピングでは記事の収集のみを行い、要約・翻訳は別プロセスとすることが実用的である。

1.3.3.3. 記事原稿データの保存

食品安全領域の記事には、各機関の年報や国際会議のプロシーディングが含まれている。これらはファイルサイズも大きく、PDF ファイルをテキスト化してもある程度大きな記事原稿となる。そのため

- 元原稿ファイル(html ファイル、PDF ファイル)
- テキスト化した元原稿

³ テキスト情報の構造を解析して必要な情報を抽出するプログラムによる処理

- ・ 翻訳結果

らはデータベースに保存せずにストレージ(AWS ならば S3)に保存し、このファイルパス(URL)をデータベースに保存する。

1.3.3.4. アクセス制限対策

EFSA の web サイトにはアクセス制限があり、本事業ではスクレイピングプログラム中に sleep を入れてアクセス頻度を軽減させていた。ただし sleep の導入だけでは何度かアクセス制限により記事の取得に失敗していたため、

- ・ 記事を登録した URL のリストを作成し、記事情報取得済み URL にはアクセスしない
 - ・ 日付情報を確認し、スクレイピング実行時から 3 日超過去の記事にはアクセスしない
- らの対策を実施した。

AWS Lambda は CPU 等の利用時間ではなく、プロセスの実行時間で課金される。そのため sleep している間も課金され、アクセス制限がある場合のスクレイピング実行環境としては必ずしも best practice とはならない可能性がある。アクセス制限 web サイトへのスクレイピングの実行においては、構築するパブリッククラウド上で最適なサービスを検討することが望ましい。ただし web アプリケーションを静的ホスティングではなく、サーバープログラムの実行環境を自前で用意するのであれば、その上でスクレイピングプロセスを実行すれば sleep における課金問題はなくなる。一方で AWS 上での EC2 の様な仮想計算機環境を構築する場合、マネージドサービスではないためセキュリティー対策を開発者側で実施する必要が生じ、この費用が必要になる。

1.3.3.5. 記事化 URL の保存

新しい記事が公開される場合、新たな URL が発行されて記事が掲載されるサイトが多い。このタイプの web サイトのスクレイピングでは、記事データベースへの保存が成功した URL を保存しておき、スクレイピング実行時に登録済み URL にはアクセスしない実装方法がある。アクセス制限があるサイトには非常に有効で、更新された記事の取得成功割合を増やすことが本事業でも確認されている。ただし運用が長くなるとこのリストが巨大になるため、スクレイピング対象 web ページからリンクが無くなった URL を削除するなどが必要になると思われる。

1.3.3.6. 同一 URL 記事更新サイト対策

各機関のトップページなどは、URL に変更はないが内容は更新される。食品安全記事探索対象に同一 URL であるが中身が更新されるサイトがある場合、更新を確認する方法としてハッシュ値⁴を保存して比較を行う仕組みを用いた。必要に応じて記事データベースのレコードに元記事データのハッシュ値を追加することも考えられる。本番システムにおいても元記事更新の仕組みが必要になると思われるが、ハッシュ値を用いた方法は有用であると考えている。

一方でアクセス制限があるサイトに対しては、本機能を有効にすると記事が更新されていないにもかかわらず、アクセス制限で失敗して記事確認状況が不明となることがある。ハッシュ値を比較する方法を有効にする URL は事前に検討を行い、最小限に絞ることを勧める。

⁴ 可変長の入力データから固定長の値を返すハッシュ関数により計算された値。パスワードの確認やデータ改ざん検知に使われる。古くは MD5 が使われたが、近年は SHA-2 の SHA-256 などが使われる。

1.3.3.7. アーカイブサイトに対する実行

過去の記事をまとめた web サイトでは大量の記事情報が含まれており、最初のスクレイピング実行時にどこまで過去の記事を取得するのかを決めておく必要がある。例えば EFSA の publication には 2025 年 3 月 26 日現在 10,787 件(図 1-5 参照)の記事があり、これを全てダウンロードする必要があるとは考えにくい。アーカイブサイトの記事に公開日の情報ある場合(図 1-5 参照)は、この日付から過去何日分までの記事を取得するのかを決めておく方法が考えられる。取得済み記事については URL の保存で対応する。

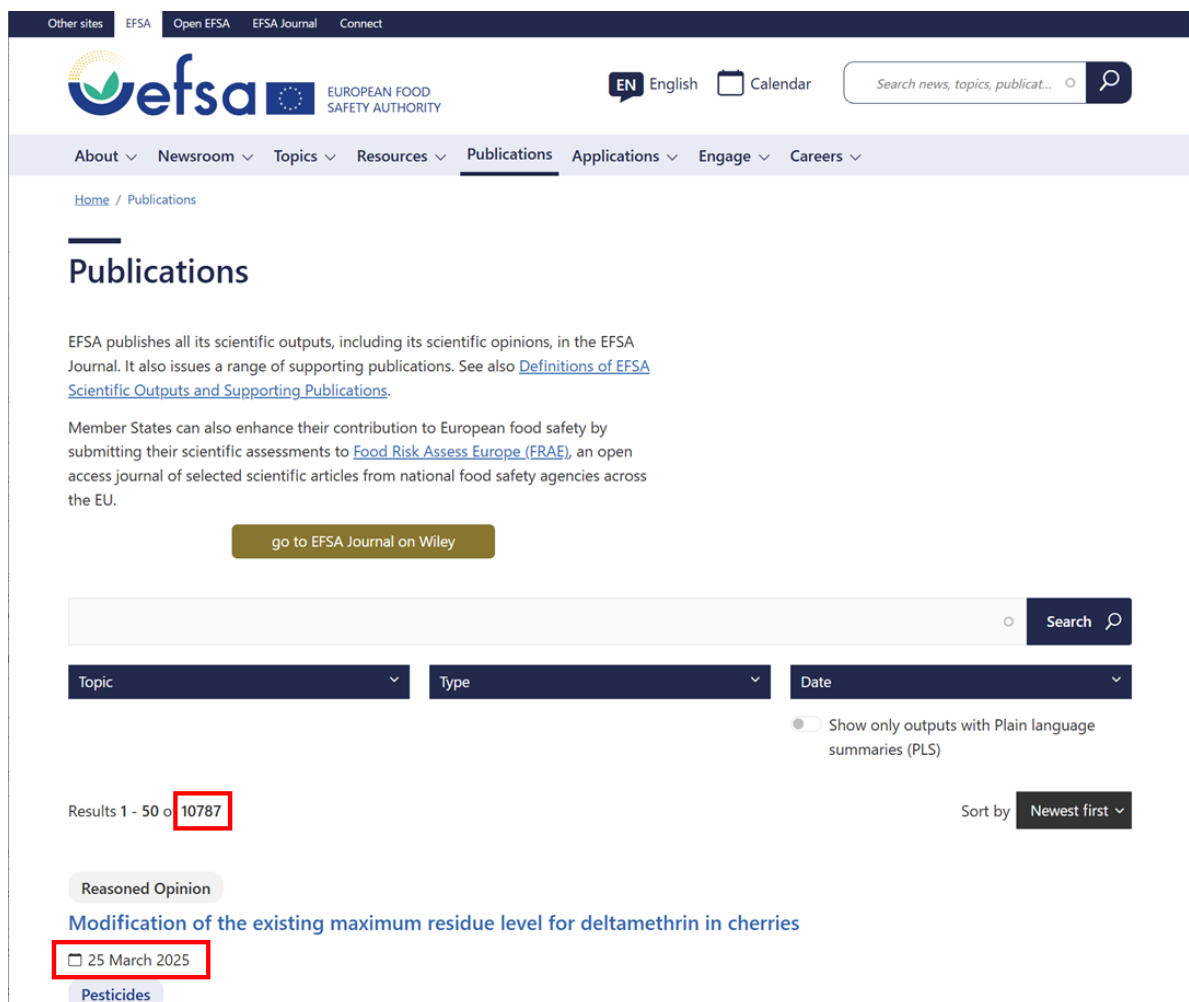


図 1-5 EFSA の Publications の記事数と公開日付情報

1.3.3.8. 記事原稿収集失敗通知機能

本事業中にも様々な理由によりスクレイピングに失敗することが起こっている。スクレイピングに失敗した web サイトがある場合にはユーザーによる確認が出来るように、エラーの情報と共に失敗した URL を表示するシステムを構築する必要がある。また手動で対象サイトのみスクレイピングを再実行する機能も必要になると考えられる。

1.3.3.9. API 利用プログラムの実行

学術文献サイトなどから API の利用による文献収集の許可が得られた場合には、API の仕様に基づき文献の収集を行うことになる。学術文献の場合、ある程度大きな PDF ファイルになる可能性があり、1.3.3.2 節から 1.3.3.5 節までの事項が当てはまる。API の仕様変更が行われた場合はプログラムの変更

が必要になるが、仕様変更については事前に通知があると考えられる。

1.3.4. 手動収集による記事原稿入力機能

プログラムからのアクセス禁止サイトには、手動で記事原稿の探索を行い、このテキストもしくは PDF ファイルの入力による手動記事作成システムが必要になる。記事原稿を登録後に幾つかのメタ情報を入力する必要があり、この入力支援機能を構築する。手動記事作成システムで入力対象となる項目を表 1-2 に示した。記事原稿情報を入力後、ソース URL、地域、国・地方、情報源(機関)以外を、LLM を使って自動作成する仕組みを構築する。

表 1-2 手動記事作成システムに入力する項目

項目	入力方法	説明
記事原稿タイトル	LLM 作成/手入力	記事原稿のタイトル。元記事の言語を想定。
タイトル(英語)	LLM 作成/手入力	記事の英語タイトル
タイトル(日本語)	LLM 作成/手入力	記事の日本語タイトル
キーワード(英語)	LLM 作成/手入力	英語キーワード。カンマ区切り。
キーワード(日本語)	LLM 作成/手入力	日本語キーワード。カンマ区切り。
ソース URL	手入力(必須項目)	元記事の URL。記事 ID をこの URL から作るため必須。
日付	保存日/手入力	記事作成日
地域	プルダウン選択	食品安全委員会指定地域
国・地方	プルダウン選択	食品安全委員会指定国・地方
情報源(機関)	プルダウン選択	食品安全委員会指定情報源
記事	LLM 要約/手入力	記事本文
コメント	手入力	コメント。入力は任意。

1.4. 翻訳・要約

1.4.1. LLM を用いた要約

1.4.1.1. 実用レベルの LLM

LLM を用いた記事の要約は、本事業で確認した限り Claude 3.5 Sonnet v1 の利用で特に技術的な課題はみつからなかった。他に構造化項目抽出の実験で用いた GPT-4o や Gemini 1.5 Pro など「A)作業報告書(ア)効果の確認・検証等報告書」の「9.2.1.2.10. 全項目まとめて抽出」の結果を見る限り、実用レベルであると思われる。

1.4.1.2. 要約の最適化

要約の内容については、本事業でも「詳細な要約」と「紹介文」の2つを選択可能とした(「A)作業報告書(ア)効果の確認・検証等報告書」の「4.2. 要約機能」参照)。目的に応じてプロンプトを調整することにより要約の内容を変えることができるため、LLM とプロンプトの組合せから食品安全に関する記事として最適化が可能である。

要約の最適化の方法としては、過去の記事と比較することが出来る。この時、

- ① エキスパートによる比較
- ② 自然言語処理技術における埋め込み表現(embedding)モデルを利用した相同性指標による評価等の比較・検証方法が考えられる。埋め込み表現モデルとは、単語もしくは文章を多次元(768~1024次元くらいのモデルが多い)の一点の数値ベクトルに写像するモデルであり、大量の文章を学習して意味論的な写像を行う(図 1-6 参照)。

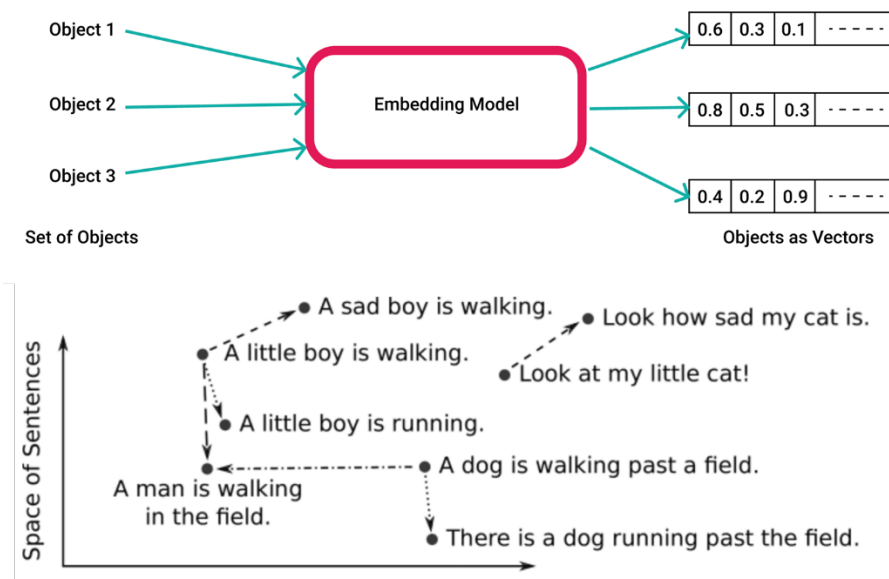


図 1-6 埋め込み表現モデルの変換イメージ(<https://www.pinecone.io/learn/vector-embeddings/>)

①のエキスパートによる比較が最も望ましいと思われるが、LLM とプロンプトの組合せが膨大になると現実的ではないだろう。そのため②の埋め込み表現モデルを利用した機械的な相同性評価が現実的である。埋め込み表現モデルによるベクトル化では、

- (ア) 目的に応じた埋め込み表現モデルの選択
- (イ) 埋め込み表現モデルの入力トークン数の制限

等の課題がある。(ア)については、埋め込み表現モデルの学習データに食品安全に関する日本語の記事

が含まれていることが望ましく、このデータ量によって精度が異なる。食品安全に関する日本語の記事が大量に用意できる場合は、**fine tuning** を行うことも考えられる。埋め込み表現モデルの性能評価には、ある程度の数の記事の相同性評価をエキスパートが実施しておく必要がある。

(イ)のトーク数の制限(入力トークン数が 512 程度であるモデルが多い)については、記事の要約をデータベースに保存する場合の文字数制限と、既存の食品安全関係情報の記事の文字数から必要なトークン数を満たすモデルを選択することで対応出来る。

1.4.1.3. LLM の入力上限トークン数と分割要約

US EPA のレポートや機関の年報などの大きなドキュメントを要約する場合には、LLM の入力トークン数を超える場合があるため注意が必要である。最近の LLM はトークン数が多くなっているが、事前に確認して入力トークン数制限を超えるようであれば議事を分割して要約を作成し、これを更に LLM に纏めさせる等の方法論を検討することが考えられる。一方で Gemini 2.0 Pro の入力トークン数の上限は 200 万トークン⁵(数百ページのドキュメントも利用可能)とされており、例外的に大きな記事原稿以外は対応する必要がない可能性がある。

1.4.2. LLM を用いた翻訳

近年の LLM を用いた翻訳の精度は高く、実用段階にあると言えるだろう。本事業で用いた Claude 3.5 Sonnet v1、GPT-4o、Gemini 1.5 Pro の日本語訳の文章に特に問題は見つからなかった。

元記事が英語以外の場合は、英訳も有用と考えられるため、英訳も保存しておく方が良いと思われる。特に AWS 上から Claude 3.5 Sonnet v1 を利用した場合、日本語訳を指示しても英訳する現象が観察され、英語以外の記事原稿については初めに英訳を行い、英訳を日本語訳する手続きにより LLM の挙動が安定した。

⁵ <https://ai.google.dev/gemini-api/docs/models?hl=ja#gemini-2.0-pro>

1.5. データの蓄積・分析

1.5.1. データベースの構築

本事業ではデータの項目変更に柔軟に対応できる Amazon DynamoDB を記事データベースとして利用した。これはサーバーレス NoSQL フルマネージドデータベースであるためデータの ID を定めるだけでデータベースが構築できる。本事業においてスクレイピングにより記事を収集したのは約 3 ヶ月であり、対象も US EPA と EFSA のみであった。このため検索機能やデータ提供において顕著な速度の問題は生じなかった。しかしながらスクレイピング対象 web サイトを増やし、記事収集期間が長くなると Amazon DynamoDB では検索時間に課題が生じる可能性がある。そのため本番システムでは relational database (RDB) を使用することを提案する。RDB ではスキーマの設計が必要となるが、既存の「食品安全総合情報システム」の記事データベース (Microsoft Access を利用) に、本事業で検討を行った結果追加を推奨する項目を表 1-3 に示した。元記事本体や翻訳を RDB に格納すると無駄にデータベースサイズが大きくなるためこれらはストレージに保存し、このパス (AWS S3 であれば URL) 情報をデータベースのフィールドに加える。タイトル、キーワード、要約は、LLM への指示文に文字数制限を加えると RDB のスキーマが構築しやすくなると考えられる。

表 1-3 本番システムのデータベースへの追加項目。既存データベースの項目は割愛している。

項目	説明
LLM 抽出英語タイトル	LLM に抽出させた英語タイトル
LLM 抽出日本語タイトル	LLM に抽出させた英語タイトルの日本語訳
LLM 抽出英語キーワード	LLM に抽出させた英語キーワード
LLM 抽出日本語キーワード	LLM に抽出させた英語キーワードの日本語訳
LLM 英語要約	LLM による英語要約(ユーザーからは変更不可)
LLM 日本語要約	LLM による英語要約の日本語訳(ユーザーからは変更不可)
英語要約	英語要約(初期値は LLM 英語要約、ユーザー修正可)
日本語要約	日本語要約(初期値は LLM 日本語要約、ユーザー修正可)
元記事ファイルパス	スクレイピングで収集した HTML ファイルもしくは PDF ファイルのストレージ上のパス
元記事テキストファイルパス	スクレイピングで収集した HTML ファイルもしくは PDF ファイルから抽出したテキストファイルのストレージ上のパス

1.5.2. キーワード抽出と記事データの分析

記事データの解析をすることを目的に LLM を利用してキーワード抽出の自動化を行うことが考えられる。現在の LLM を利用してキーワードを抽出することが可能であることを本事業の実験で確認した。しかしながら抽出されたキーワードを確認すると、EFSA などの機関名などが多く含まれており、得られたキーワードをどの様に分析に用いるのかが未検討の課題として残った。

記事データの分析方法については、

- ① 期間別キーワード頻度解析
 - ② 埋め込み表現モデルを用いた主成分分析
- 等を候補に挙げる。

1.6. ドキュメントからの構造化抄録データベースの構築

1.6.1. 構造化抄録の目的

収集した食安全に関する記事の解析を容易にするために、解析に必要な項目をデータベースのフィールドに分けて保存することが考えられる。項目別にデータを保存することにより、検索や集計を高速かつ正確に行うことが可能になる。しかしながら食安全に関する記事は、化学物質の毒性研究、化学物質のばく露研究、感染症、規制など幅が広い。これらの多様な領域では、記事を解析するための項目も異なると予想される。そこで

A) 記事カテゴリーの洗い出し

B) 記事カテゴリー毎の抽出項目設定

が必要になり、これらの情報を整理した後に技術的な課題として

① 記事カテゴリーの自動付与

② 項目の抽出

を検討することが必要になる。

1.6.2. 記事カテゴリーの洗い出しと抽出項目設定

食安全領域の記事は多様な領域で構成されており、様々な分類方法がある。例えば表 1-4 に食安全関係情報の分類と EFSA の分類を示した。食安全関係情報の分類の「大分類」と「EFSA 分類」は対象を軸にした分類であり、「情報分類」は記事の発信を目的とした分類であると言えるだろう。「情報分類」をカテゴリーとすると「大分類」の項目は「対象」の位置付けとなり、抽出項目の一つとするとも考えられる。

構造化抄録のカテゴリー分類については、日常業務において食安全情報を作成、整理している食安委職員に協議してもらうことになるが、カテゴリー毎の抽出項目が一致するような分類を行う必要があると思われる。記事データの集計を行う際に、抽出情報無しフィールドが多いと粗な集計となり、意味のあるデータ解析を行うことが難しくなると思われる。

記事カテゴリーと抽出項目については、ある程度の期間運用を行い、適宜修正を行うプロセスも必要になると考えている。

表 1-4 食安全関係情報の分類と EFSA の分類

食安全関係情報の分類

EFSA 分類

情報分類	大分類		
調査	化学物質	PFAS	データ
リスク管理措置	微生物・プリオン・自然毒	農薬	横断研究
リスク評価	新食品等	栄養	化学物質汚染
危害発生	肥料・飼料等	方法論	生物災害
記録文書	機能性食品素材	新興リスク	動物福祉
主張・見解表明	生産・製造技術	食材・包装	動物健康
リスクコミュニケーション	情報・評価技術	遺伝子組み換え生物	COVID-19
	学問・思想		
	その他		

1.6.3. 記事の自動分類付与と項目の抽出

1.6.3.1. 自動分類付与

LLM による記事の分類については、カテゴリー化とカテゴリーの定義が必要なため本事業では検討できなかった。記事によっては複数カテゴリーにまたがることも想定され、記事に対してユニークな分類を付与することは難しい場合も生じると思われる。LLM による記事の自動分類を評価するには、

(ア) エキスパートによる記事の分類データの作成

(イ) LLM による分類の付与

を行い、これらを比較することで評価することになるだろう。そのため統計データが算出できる規模の正解データをエキスパートが作成する必要がある。

LLM による分類の付与では、各カテゴリーの定義文をプロンプトに埋め込んでカテゴリーを付与させる方法が考えられる。この方法ではカテゴリーの定義文もある程度の長さになるため、長い記事の場合はトークン数の問題が生じる可能性がある。

1.6.3.2. 項目の抽出

「情報分類」が「リスク評価」の場合の抽出項目として表 1-5 の項目を抽出項目とし、複数の LLM を用いて項目の抽出を行った(「A)作業報告書(ア)効果の確認・検証等報告書」の 9.2 節参照)。しかしながら、この結果を定量的に評価することが出来ず、出力結果を目視で確認した感覚的な評価しか行うことが出来なかった。本項目の課題は、

(ア) エキスパートによる項目抽出データの作成

(イ) LLM による抽出結果とエキスパート抽出結果の評価方法

である。エキスパートによる項目抽出データは 1.6.2 節で定義したカテゴリー毎に統計値が計算出来る規模の抽出が必要であり、これはかなりの工数になると予想される。

エキスパートによる抽出結果と LLM による抽出結果は表 1-5 の項目では、単語ではなく自然言語になる場合がある。そのためこの評価方法も課題である。エキスパートが抽出した項目が過不足なく網羅されている場合でも表現の揺らぎが問題となり、また LLM 抽出ではエキスパートが抽出していない他の情報が付与されることも予想される。1.4.1.2 節で提案した埋め込み表現モデルによるベクトル化も一つの方法であるが、埋め込み表現モデルの性能評価が未実施である。評価方法は項目毎に異なることが予想され、評価方法の検討もかなりの工数になる可能性がある。

表 1-5 リスク評価における抽出項目

① 対象物質	③ ハザードの概要	⑥ 他機関の評価情報
② 対象動物	④ 毒性に関する情報	⑦ 評価結果
	⑤ ばく露に関する情報	⑧ 評価の限界や今後の課題

「A)作業報告書(ア)効果の確認・検証等報告書」の 9.2 節の結果からは、項目ごとに抽出精度が異なった。抽出精度が悪かった項目については、プロンプトの工夫と LLM の選択により精度向上を試すことになる。抽出精度を迅速に評価する仕組みを構築することが、実用レベルとなる項目抽出方法の構築に重要である。

項目の抽出に成功したとしても、元記事に表現の揺らぎが含まれているため、データベースの検索漏れや集計漏れが発生することが予想される。同義語を定義した公開ライブラリーを使うことである程度吸収できるが、専門用語の場合は独自に同義語辞書を構築する必要がある可能性がある。

1.6.4. 食品安全記事チャットボットの構築

Gemini 2.0 Pro などの入力トークン数が多い LLM を用いて、検索して絞り込んだ記事情報を全てプロンプトに含めて LLM に問い合わせるチャットボットを構築することが考えられる。これは記事に記載されている用語の揺らぎにある程度対応できると思われ、分析する内容によっては有力な候補となる可能性がある。

1.7. 次期システム構築の課題

1.7.1. 検討対象項目

本事業から得られた本番システムを構築する上での課題として、

1. 記事原稿自動収集対象調査
2. 要約の最適化検討
3. 記事データ解析手法調査
4. 食品安全に関する構造化項目データベース構築のため情報整理と技術課題調査を挙げる。

1.7.2. 記事原稿自動収集対象調査

記事原稿収集対象 web サイトについて、図 1-4 に示したフローチャートに従って具体的な記事原稿収集方法を予め調査しておくことにより、システム構築にかかる費用及び保守・運用にかかる費用を見積ることが可能になる。

情報収集対象 HP 一覧に記載されている 236 URL について、「具体的なアクセス先」と指定されているサイトも含めて、

- ① RSS 対応の有無と RSS feed に含まれる情報
- ② プログラムからのアクセスの可否
- ③ 許可制 API の提供状況
- ④ アクセス頻度や記事数からのスクレイピング対象としての経済性

の調査を行う。この調査結果を元に、システム構築・運用にかかる工数を算出する。

一方で費用対効果の面からは、ガバメントクラウド上にシステムを構築しないことも考えられる。例えば EFSA は RSS feed のみでアブストラクト情報が収集できるため、適当な LLM と契約できれば記事収集と翻訳・要約が可能である。LLM に RSS feed の URL を指定して、Google Gemini 2.0 Pro に

URL 中の RSS feed を記事毎に title と description を抽出し、日本語訳を含めて markdown 形式で出力してください。

と指示した時の出力例を図 1-7 に示した。API が利用可能な契約であれば、記事収集を行うプログラムを LLM に作成させて実行することも可能である。例えば

URL 中の RSS feed を取得し、記事毎に title と description を抽出して Google Gemini 2.0 Pro の API を用いて日本語訳を加えて markdown 形式で出力する python プログラムを作成してください。API に接続するための key はダミーを作成してください。

として

```
import feedparser
import re
from google.generativeai import GenerativeModel
from google.generativeai import configure

def get_rss_feed(url):
    """RSS フィードを取得する"""
    feed = feedparser.parse(url)
    return feed

def extract_title_and_description(entry):
    """記事からタイトルと description を抽出する"""
    title = entry.title
    description = re.sub('<[^>]+>', '', entry.description) # HTML タグを除去
    return title, description
```

```

def translate_with_gemini(text, api_key):
    """Google Gemini 2.0 Pro API を使ってテキストを日本語に翻訳する"""

    configure(api_key=api_key)
    model = GenerativeModel('gemini-pro')

    prompt = f"Translate the following text to Japanese:¥n{text}"

    try:
        response = model.generate_content(prompt)
        translated_text = response.text
    except Exception as e:
        print(f"Error during translation: {e}")
        translated_text = "翻訳エラーが発生しました。" # エラー時の代替テキスト

    return translated_text

def generate_markdown(feed, api_key):
    """Markdown 形式の文字列を生成する"""
    markdown_output = "# RSS Feed Summary¥n¥n"

    for entry in feed.entries:
        title, description = extract_title_and_description(entry)
        translated_title = translate_with_gemini(title, api_key)
        translated_description = translate_with_gemini(description, api_key)

        markdown_output += f"## [{title}]({entry.link})¥n¥n"
        markdown_output += f"***Description:** {description}¥n¥n"
        markdown_output += f"***日本語訳 (タイトル):** {translated_title}¥n¥n"
        markdown_output += f"***日本語訳 (Description):** {translated_description}¥n¥n---¥n¥n"

    return markdown_output

def main():
    """メイン処理"""
    rss_url = "https://www.efsa.europa.eu/en/rss/news/publications" # RSS フィードの URL
    gemini_api_key = "YOUR_DUMMY_API_KEY" # ダミーの API キー

    feed = get_rss_feed(rss_url)
    if not feed.entries:
        print("Could not fetch or parse RSS feed.")
        return

    markdown = generate_markdown(feed, gemini_api_key)
    print(markdown)

if __name__ == "__main__":
    main()

```

のプログラムが作成された。API を利用した実行法であれば、LLM 使用料は月額数万円程度であると思われる。

許可制 API を利用したプログラムについても API の仕様を記した JSON ファイル等をプロンプトに含めることで作成可能である。許可制 API を使って大量の PDF ファイルをダウンロードすることはネットワークの負荷になるため、これを実行するタイミングには注意が必要である。

一方でスクレイピングを実行するプログラムを LLM に作成させることはやや困難であると予想される。Web サイトによっては JavaScript でページを作成している場合があり、このソースコードからスクレイピング対象を LLM に指示する必要がある。スクレイピング対象サイトの自動化の重要性により、

ガバメントクラウド上にシステムを構築するかどうかの一つの判断方法となるだろう。また各国の当局と協議して食品安全に関する記事の国際的なレポジトリサイトが構築出来れば、RSS feed と同様の方法で原稿が収集できるので、システム化は不要になる可能性がある。

EFSA Publications RSS Feed Summary

Pest survey card on *Grapholita inopinata*, *G. packardi* and *G. prunivora*

Description: This document provides the conclusions of the pest survey card that was prepared in the context of the EFSA mandate on plant pest surveillance (M-2020-0114) at the request of the European Commission. The full pest survey card for *Grapholita inopinata*, *G. packardi* and *G. prunivora* is published and available online in the EFSA Pest Survey Card gallery at the following link and will be updated whenever new information becomes available:
<https://efsa.europa.eu/plants/planthealth/monitoring/surveillance/grapholita-inopinata-prunivora-packardi>

日本語訳: 本文書は、欧州委員会の要請によるEFSAの植物害虫サーベイランスに関するマンドート (M-2020-0114) の文脈で作成された害虫サーベイカードの結論を提供するものです。 *Grapholita inopinata*, *G. packardi*, *G. prunivora* の完全な害虫サーベイカードは、EFSA Pest Survey Cardギャラリーでオンライン公開されており、新しい情報が入手可能になるたびに更新されます。

Pest survey card on *Gremmeniella abietina*

Description: This document provides the conclusions of the pest survey card that was prepared in the context of the EFSA mandate on plant pest surveillance (M-2020-0114) at the request of the European Commission. The full pest survey card for *Gremmeniella abietina* is published and available online in the EFSA Pest Survey Card gallery at the following link and will be updated whenever new information becomes available:
<https://efsa.europa.eu/plants/planthealth/monitoring/surveillance/gremmeniella-abietina>

日本語訳: 本文書は、欧州委員会の要請によるEFSAの植物害虫サーベイランスに関するマンドート (M-2020-0114) の文脈で作成された害虫サーベイカードの結論を提供するものです。 *Gremmeniella abietina* の完全な害虫サーベイカードは、EFSA Pest Survey Cardギャラリーでオンライン公開されており、新しい情報が入手可能になるたびに更新されます。

Pest survey card on *Venturia nashicola*

図 1-7 Google Gemini 2.0 Pro を用いて、<https://www.efsa.europa.eu/en/publications/rss> に対して title と description を抽出させて、日本語訳を含めて markdown 形式で出力させた例

1.7.3. 要約の最適化検討

近年多くの優秀な LLM が提供されており、システム構築ではこれらを選択する必要があると思われる。そのためには LLM とプロンプトの組み合わせによる最適化を行い、この結果から LLM を選択する方法がある。

1.4.1.2 節で示した方法論では、埋め込み表現モデルを用いており、埋め込み表現モデルの選択にエキスパートによる評価が必要になる。ある程度の数の記事セットにおいて、この距離をエキスパートに作成してもらい、この距離関係の再限度で埋め込み表現モデルを評価する。尚、埋め込み表現モデルは、Hugging Face⁶からフリーのモデルをダウンロードすることを想定している(図 1-8 参照)。

⁶ <https://huggingface.co/>

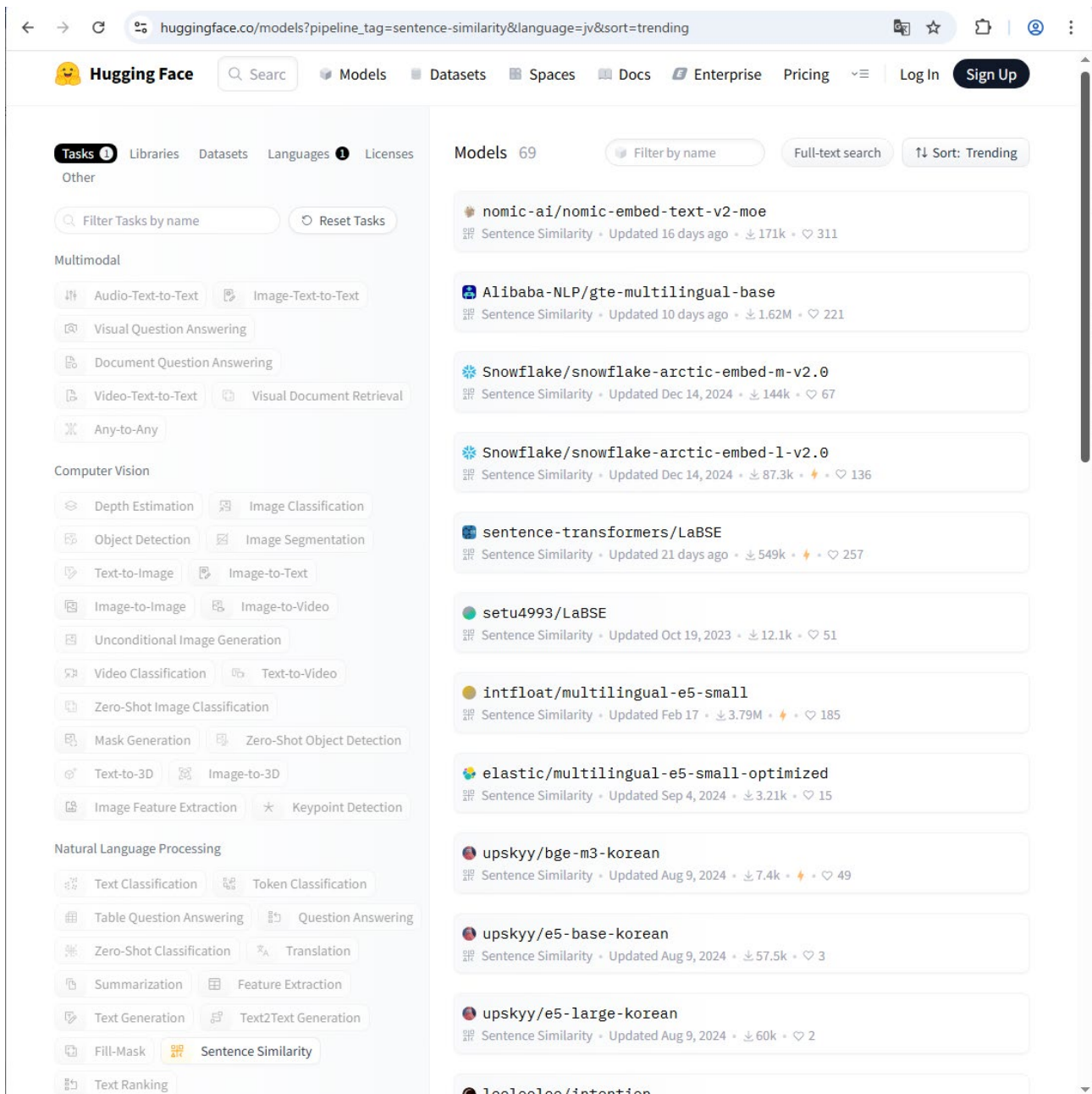


図 1-8 Hugging Face において、Sentence Similarity と Japanese を選択して残った 69 モデルの一部

1.7.4. 記事データ解析手法調査

記事データを収集後、どのような解析を行うとどのような結果になるのかを試しておくことは、次期システムの要件を決める上で有用である(1.5.2 節参照)。有効な解析方法が予め分かっていたら記事の分類やキーワード抽出などの要件を決めることが出来る。どの様な解析方法が有用なのか分からない状況では、解析手法リストの提案を受けて実行してもらい調査が考えられる。

1.7.5. 食品安全に関する構造化項目データベース構築のため情報整理と技術課題調査

構造化抄録データベースの構築では、先行しているヒトの臨床研究における論文の構造化抄録などを参考にしながら、食品安全に関する記事のカテゴリー化と抽出項目を定義し、且つエキスパートによる項目抽出を行って基盤情報を整備した上で、既存の技術で実用レベルの運用が可能になるのかを調査することが必要になると思われる。

図 1-9 に本調査の作業項目と情報の流れを示した。青字で示した部分は食品安全委員会事務局側のエ

キスパートが行うことが望ましい作業項目であり、かなりの工数になる。緑字の部分は業務委託になると思われる部分であるが、評価方法の検討では 1.7.3 節の埋め込み表現モデルの評価結果があることが望ましく、この評価データの作成でも食品安全委員会のエキスパートによる作業が発生している。

「記事カテゴリーの作成」と「カテゴリー毎の抽出項目リスト」については一定期間運用した上で修正する作業項目を加えた方が良い。運用期間中にエキスパートが「記事カテゴリーデータ」と「抽出結果データ」の作成を行うことにより、このデータを評価用データ(機械学習での外部検証データ)として使用することが出来る。

元記事に表現の揺らぎがあることから、抽出項目の一致による解析を行う場合は表現の揺らぎ対策が必要なる。抽出項目結果から表現の揺らぎをリストアップする仕組みも検討事項である。

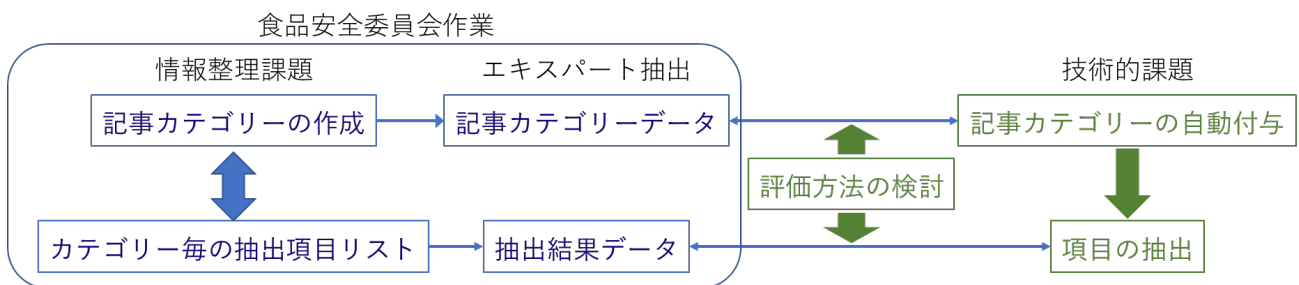


図 1-9 食品安全に関する構造化項目データベース構築のため情報整理と技術課題調査の作業項目

一方で解析する目的によっては構造化抄録データベースを構築するのではなく、記事データを参照する食品安全記事チャットボットを構築する方法も考えられる。この方法では埋め込みモデルや LLM を更新することで自動的に性能が上がることを期待され、元記事データの表現の揺らぎにもある程度対応できると思われる。