2026

2025 Vol.1
2025 Vol.2

WHITE PAPER PROJECT

CONTENTS

2025 | Vol.1

戦略的社会実装論

事業戦略とアラインする研究開発に向けて

森脇 大輔
Moriwaki, Daisuke

CyberAgent AI Lab
Senior Research Scientist
moriwaki_daisuke@cyberagent.co.jp

keywords: 研究プロジェクト, 社会実装

Summary

2000 億ドル以上を生み出す広告オークションシステム（IAB Revenue Report 2025）や急激な成長を遂げる生成 AI チャットボットなど、技術開発が生み出す経済的・社会的インパクトは計り知れない。一方で、研究開発の現場では日々研究開発を事業に結びつけること（社会実装）の困難さに直面している。社会実装の達成のためには、どのような技術をどのように開発すれば良いのか、開発した技術をどのように事業に届けるのか、こうした「社会実装の壁」を戦略的に突破することが不可欠である。本稿では、事業とアラインする研究開発のあり方について議論する。

> 研究成果を事業化して利益に結び付ける（つまり，世の中の役に立つものにする）ためには“死の谷”を無事渡らなければならないと言われている。それくらい研究成果を事業化することは難しいことだ。—原陽一郎（元東レ経営研究所社長、元長岡大学学長）

1. はじめに

2022 年 vol.2 の WPP において掲載された拙稿 [森脇 22] では、企業内研究者が行う**企業に新たな知見をもたらし、既存のサービスやプロダクトを改善したり、新しいサービスやプロダクトの種となることを目標とする**活動を研究プロジェクトと定義し、研究プロジェクトがどのように失敗しうるかを議論した。15 のアンチパターンを列挙することで、プロジェクト成功のための条件を整理した。本稿では一歩進んでそうした研究プロジェクトの中でも開発した技術を社会実装する社会実装プロジェクトを対象としてその成功を阻む「社会実装の壁」を中心に議論する。

2. 企業研究者にとっての社会実装

研究活動によって得られた知見や技術をプロダクト化やサービス化することで、研究室の外にいる人々のために役立てることを一般に社会実装という。本来の語義は研究者の視点からその活動を外側に広げていくことを指す。したがって、もともと社会の中にいてプロダクトやサービスを展開する企業の中にいる研究者たちにとっての社会実装は、大学等非営利研究機関の研究者のそれとは大きく意味を異にする。つまり、大学等の研究者にとっての社会実装は文字通り「社会」やそこに生きる人々に対する貢献を一義的に考えるのに対し、企業という社会に取り込まれている研究者にとっての社会実装は、所属する企業の展開するプロダクトやサービスを通じたかたちでの社会貢献になる。つまり、大学等の研究者がシンプルな RtoC (Research to Consumer) モデルだとすると、我々企業研究者はまず最初に事業部に対して研究開発のアウトプットを届け事業部が開発するプロダクトやサービスを経由して社会貢献する、RtoBtoC (Research to Business to Consumer) モデルになる（図 1）。もちろん、研究者自らが事業責任者となることで R と B を一体化し、R/BtoC モデルとすることは可能だが、その場合、企業戦略の中の研究の位置付け、費用対効果などを明らかにし、事業化のためのリソースを社内から獲得する必要がある。もう一つ重要な点は、企業が実施する社会実装は、常にマーケットでどのように評価されるかを意識する必要があるということだ。とてつもない社会厚生を生み出している Google 検索も、マーケットでの評価対象は検索そのものではなくその広告モデルにある。事業としての持続性を持った上で社会に貢献するためには、社会とマーケット双方に評価されるかたちで技術を届ける必要がある。研究者自身やその支援者がビジネスの言語で技術を再定義し、意思決定層にその正当性を理解させなければ、いかに学術的、技術的な価値が高くても機能化や事業化ができずに埋もれてしまう。そして、いったん機能化・事業化ができたとしても、マーケットにおけるインパクトが十分でなければ社会的インパクトを発揮する前に機能廃止

や事業縮小・撤退を余儀なくされる。**社会実装の壁**に突き当たり、力尽きる（冒頭の原氏の言葉を借りれば「死の谷」に落ちる）。

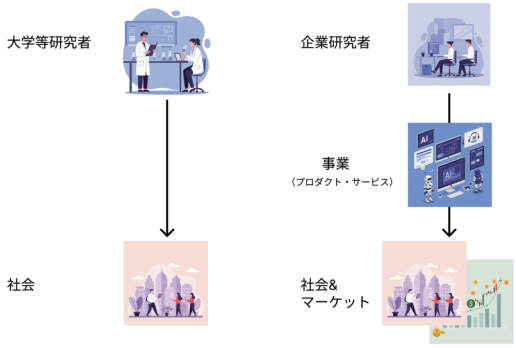

図 1　大学等と企業の社会実装の違い.

3. 社会実装の壁

それでは、社会実装の壁を突破するにはどうしたらいいのだろうか。結論を急ぐ前に、我々はこの壁の正体を十分に理解する必要がある。図 1 をもう一度ご覧いただきたい。企業研究者が研究成果を社会に届けるためには、３つのステークホルダーが存在することがわかる。一つ目は自分や同僚の研究者、２つ目は事業部、３つ目はマーケットや社会だ。社会実装の壁の正体はこれらのステークホルダーそのものであり、それぞれの壁突破の条件は、それぞれのステークホルダーの基準や考え方に依存している。つまり、研究者自身の事前アセスメントによって、技術のビジネス的・社会的意義が見出され、事業部からそのアイデアへの一定の理解を得られれば検証が開始される。研究者を中心とする検証の結果、事業環境での成功可能性が事業部側で共有されれば、試行的に新機能や新事業がリリース（実証実験）されることになる。そして、この試行がうまくいけば全面的・恒久的にその新機能や新事業がマーケットに展開されることになる。この３段階のフェーズを概念的に示したのが、図 2 である。

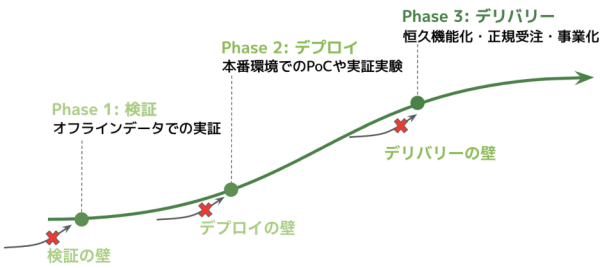

図 2　社会実装の３つの壁.

3·1 「検証」の壁

検証とは、プロダクトの保有する実データ（例えば広告配信サーバーの配信ログや広告クリエイティブごとの配信成果、ウェブサービス上の行動ログ、実店舗での購買データ）に対してデータ分析やアルゴリズムのオフライン検証を行うことや、バーチャル空間や実験室環境下での実験を実施しその有効性を確認する作業である。広告配信アルゴリズムを変更した場合にクリック率がどうなるか、販促のターゲットを変えたときに広告効果がどうなるか、消費者の購買行動をうまく予測できるか、といったソフトウェアの機能評価や、被験者を対象としたウェブアプリのテスト、デバイスやロボットの機能性の確認など、アイデアの内的妥当性を把握する上で欠かせないステップである。

事業部の普段のオペレーションとは切り離された研究であるため、事業部から見たときのコストは低い。検証段階では、実施結果がどうなったとしても実施費用以上に損失を出すことはないため、リスクが小さくゴーサインが出やすい。研究者側も実データを用いることで研究結果の応用可能性をアピールしやすいため、両者のインセンティブが合致しやすいフェーズとも言える。

検証の壁とは、研究者が有望であると感じた技術の検証に関して、事業部の理解を得られずに、永遠にパブリックデータセットから卒業できないことである。例えば、自分から話を聞きに行くという行動が取れなかったり、事業に対する基本的な理解が乏しいため話が噛み合わなかったりといったケースが該当する。検証段階でのつまづきポイントについては、[森脇 22] でも詳述したのでご関心のある方は参照されたい。

3·2 「デプロイ」の壁

検証によってその事業価値が確認された技術は、実プロダクトにお試しで実装されることになる。これを**デプロイ**と定義する。例えば、広告配信サーバーに新しいアルゴリズムを導入し本番環境で配信に用いることや、実店舗において開発したロボットを運用し集客への効果を見るといったことが含まれる。

デプロイの壁とはオフライン検証で成果が出ても、一向に事業部が相手にしてくれず、延々と事業部から提供されたオフラインデータで論文を書き続ける状態を指す。筆者の体感で恐縮だが、いわゆる「リアルワールドデータ」での実績を売りにする技術の９割は、検証の壁を突破したがデプロイの壁に引っかかっていると思われる。国際会議でファンシーな研究を発表している海外のテック企業のデータサイエンティストに「その技術は実際に使われているのか」と質問すると往々にして否定的な回答が返ってくる。デプロイ段階に移行できずに足踏みしている研究者は、デプロイ段階が検証と決定的に違う点を理解していない。

一つはシステムの**本番環境に干渉する**ということであ

る。アルゴリズムを notebook やスクリプトで実行することと、実際に稼働している本番環境のシステムに組み込むことは、別次元の話だ。オフラインデータを分析したり、アルゴリズムを適用して評価するというシステムは、そのアルゴリズムや分析手法がいかに高度だとしても、IT システムとしてみれば、データを入力し、結果を返し、それを評価する、という簡素なコンポーネントでしかない。一方で、本番環境のシステムは、ユーザーの入力やデータベース処理など外部とのさまざまな通信、ロギングやバリデーションなど複雑な機構を持っている。システムが安定稼働しているのは、これらの複雑な処理機構がそれぞれエラーなく動いているだけではなく、それぞれの入出力が意図したものになっていることが大前提である。整然と稼働しているシステムの中に、新しいコンポーネントを投入することで、システムのパフォーマンスが落ちたり、最悪の場合、オペレーションに重大な影響を与えたりする。例えば、一般に深層学習によるアルゴリズムは計算が重いため、いかにオフライン環境で精度が高くても本番環境に入れた途端にシステム全体のパフォーマンスを落とし、ユーザーエクスペリエンスを大幅に劣化させることがある。こうした本番環境への干渉のリスクを踏まえても機能実装すべきかが壁突破の一つ目のポイントだ。

そしてもう一つは**事業活動に干渉する**ということである。上述の通り、オフラインデータの検証のみを行ったコンポーネントを本番環境に組み込むのは、危険である。多くの場合、スクリプトは（別の言語に）書き直され、テストコードが追加され、検証段階と同程度の工数をかけてチェックを経た上で実装される。これらの工数のほとんどは事業部側が負担となるため、デプロイそのものが事業の PL（損益計算書）にネガティブなインパクトを与える [*1]。したがって、よっぽど有望な実装でなければ、他の開発が優先されることになる。また、仮に潜在的に大きな可能性を秘めた実装であっても、事業活動へのリスクを増加させる危険性も考慮する必要がある。本番環境に導入するということは、顧客に対して何らかの影響を与えることに他ならない。前述のシステムの安定稼働が担保されたとしても、プライバシーの保護やユーザー間の公平性などさまざまなリスクも考慮する必要がある。

3.3 「デリバリー」の壁

デリバリーの壁は、デプロイの壁を乗り越えて実施されたオンライン実験や実店舗実験など、リアルワールドに干渉する実験が完了しても、プロダクトの恒久的な機能として実装されず「なかったこと」にされたり、お試しで実施した実証実験が文字通りお試しで終わり本格的な実施に至らなかったりする現象を指す。デリバリーに成功すれば、「実データを用いた」だけでなく, **実世界でインパクトを証明した**という強力な付加価値を持つ。KDD Applied Data Science Track や RecSys Industry Track のような産業的成功を収めた研究を歓迎する舞台で絶賛されることになるが、ここに至るまでの道は遠い。

デプロイとデリバリーの最も大きな違いは、**その機能や新規事業が事業戦略の中心にあるか**、ということである。デプロイはあくまでお試しであり、一時的な開発コストの膨張やリスクを責任者に飲み込んでもらえば到達が可能である。しかし、新規機能の恒久的な運用や新規事業の積極的展開は、事業責任者自身がその技術と運命をともにするという決断をするということであり、事業戦略そのものの重大な変更を意味する。たった一つの機能の追加がプロダクトの提供価値の中心になり飛躍的な成長を遂げることもあれば、既存機能とのコンフリクトを起こすこともある。

従って、デプロイフェーズにおいて「技術的な成功」だけでなく「事業的な成功」を示唆できなければ次の段階に進めない。デプロイによってどのような KGI、KPI に対する示唆を得る必要があるのか。そのために、技術以外の要素で阻害要因になるものはないのか。機能の恒久化を考える上でどのようなリスクが存在し、それは実験によって不安を払拭できるのかどうか。費用対効果があると主張するためにはどの程度の売上インパクトが必要なのかなど、デプロイフェーズにおける事前の検討や準備が不可欠である。特に、デリバリーの可否に大きく影響するのは、より上位の経営層や顧客の意思決定層など直接対話が難しいステークホルダーに対してどのようなメッセージを発すべきかである。このため、こうした直接見えないステークホルダーの価値基準を学ぶことが大切になってくる。

3.4 ステージゲート法との類似点

こうしたフェーズによる技術開発プロセスの切り分けは決して特殊なものではない。国立開発法人新エネルギー・産業技術総合開発機構（NEDO）がプロジェクトの 6 – 9 割に採用するなど研究開発マネジメントで広く用いられている**ステージゲート法**は、製品のアイデア創出から市場投入までの 5 つの「ステージ」に分け、それぞれのステージで基準をクリアしたプロジェクトのみに「ゲート」通過を許すことで、有望なプロジェクトを選抜する手法である [経済 22, 西野 14]。図 3 にあるように、社会実装の壁は、5 段階に分かれるステージゲート法を簡素化し、テック企業向けにアレンジしたものと言える。

4. 壁突破のための戦略

3 つの壁突破のためには、研究者は何をすべきだろうか。以下では、多くの研究者が実施している事業化に向けた戦略について、それぞれの壁に対して効果的なものを列挙する。研究分野や事業領域によっては別の論点も

*1 ただし、AI コーディングの登場でこの前提は劇的に変わりつつある。

社会実装のフェーズ	ステージゲート法
	ステージ1：初期調査
	ステージ2：ビジネスプランの検定
検証	ステージ3：製品の設計、開発、プロトタイピング
デプロイ	ステージ4：製品の機能、品質、市場適合性のテスト
デリバリー	ステージ5：量産、マーケティング、販売開始

図 3　社会実装の３つのフェーズとステージゲート法.

あると思われるが、最低限のものを記したということでご了承願いたい。

4·1　検証の壁突破のための戦略

検証の壁の突破のためには、まず事業との関係性を作って自らの有用性をアピールし、仕事を作るという作業が必要だ。したがって、特にコミュニケーションが大事になる。

§ 1　圧倒的なドメイン知識を得る

技術を事業に活かすということは、研究者側が事業を理解するか、事業側が技術を理解するかのどちらか、あるいは両方がないと成立しない。もし、事業に対する理解をせずに技術を使ってもらおうというのであれば、それは事業サイドに勉強しろといっているのと等しい。そういう態度で何か大きな事業貢献が達成しうるだろうか。もちろん、事業部の中で営業や開発、データ分析を経験しない限りわからないことは多いが、一方でリリースやプロダクト関連の記事、IR といった公開情報に加え、社内の Slack 等で学べることは多いし、事業部の notion を共有してもらうことで事業課題の全体像は把握できることが多い。その上で、さまざまな機会を捉えて直接事業部の方と話すことで、真に解決すべき課題が見えてくる。さらに言えば、事業部では手がつけられていないデータの解析などをやることで彼らが見えていないことまで把握することだって可能だ。こうしてドメイン知識をつけることが事業貢献の前提である。

§ 2　正しく発信する

RtoBtoC モデルを前提にすれば、我々の技術の最初の顧客は事業部である。したがって社会実装の第一歩は事業部に対するマーケティング活動である。そもそもその技術は何を解決する可能性があるのか。どのような効果があり、どのようなコストが予想されるのか。もちろん、事業領域によっても、具体的な課題によっても異なるが、先進事例をかき集め、妄想を繰り返してあなた自身のプロダクトである技術を売り物にするための活動をすべきである。まずは、前提知識がなくてもわかる魅力的なスライドを作り、解決したい課題とその方法を整理する。その上で、導入コストや事例など顧客が知りたい要素を追加していく。これをもってさまざまな場所で登壇するなり、個別に事業部に話に行くなりして認知を広げる。トップカンファレンスでの採択実績も権威づけになる。デモを作ることができる技術であれば、それも公開していく。重要なのは、ターゲットにとってわかりやすく魅力的な説明を心がけることだ。事業責任者や、顧客が理解できる資料を時間をかけてでも作り上げることがまず大事だ。

§ 3　ネットワークを作る

チームの技術や研究力について登壇やブログ等で十分広報活動をしたとしても、ある技術が必要な人物に適切に届けることは至難の業だ。仮に届いたとしても、初対面の相手に連絡することに抵抗がある人は多い。やはり、人的ネットワークによって、技術の需要者と**繋いで**もらうことが必須になる。筆者自身を顧みても、直接繋がった相手との協働より、誰か第三者に繋いでいただいたプロジェクトの方が圧倒的に多い。お互い初めましてから始まるプロジェクトはなかなかペースをつかむまで時間がかるが、間に入る人がいればお互いの警戒感も少なく始められる。そのため、社内外にネットワークを張り巡らせることは大事だ。特にネットワークのハブとなるような人物と懇意にし、自身がやりたいことを伝えておくことが必要だ。前提として、相手のリテラシーに合わせたわかりやすく魅力的で端的な資料やデモが手元にあるべきだ。

4·2　デプロイの壁突破のための戦略

デプロイの壁の突破のためには、事業部サイドに今後の成長の起爆剤となる技術であるという期待を持ってもらえるかが大事になる。したがって、事業の方向性を理解した上で、将来的につながる技術であることを明確化する必要がある。また、事業部から見た時に進捗がコントロールできているという感覚を持ってもらうために社会人としての仕事の進め方をする必要がある。

§ 1　先を読む

研究者の良いところは、日々のオペレーションから解放されていることである。つまり、目先の PL を気にせずに長期的かつ広い視野で第三者視点で事業について想いを巡らせることができる。近年の市場環境の変化は、破壊的な技術革新や制度変更によって引き起こされることが多い。広告オークションや生成 AI がテック企業の収益フロンティアを拡大する一方、プラットフォーマーによるプライバシー政策の変化はオンライン広告市場の風景を一変させた。こうした動きを先読みすれば計り知れない貢献ができる。すなわち、将来的に投資するであろう事業領域で技術開発を先取りし他社に先駆けた市場投入を促すこと、そして、将来性のある事業領域で特許を取得することで強固な MOAT を築き安全にビジネス構築できる環境を用意することだ。

§ 2　スケジュールを引いてやり切る

研究者は日々のオペレーションから解放されているだけでなく、明確な納期や締め切りから自由に自らのアウ

トプットを出すことが許されている（学会締め切りなども破ったとして誰か他人が困るわけではない）。しかし、事業は納期やリリース予定日、各月・各期の PL などさまざまな制約条件のもとでスケジュールに基づいて動いている。研究者側が同じ土俵に乗らない限り事業貢献は難しい。機能実装や実証実験は顧客調整や開発スケジュールの調整のもとに実施される。スケジュールを守れないチームは適切な協働相手とは見做されない。仮に事業部側から締切や納期を求められない場合、それはそもそも期待されていないことの裏返しである。

§3　組織・人に興味を持つ

AI 時代にあっても意思決定者は常に人間である。当然、壁突破の成否は全て人間の判断によっている。また、自らのプロジェクトに対する協力者をいかに集めるかも重要だ。どのようなステークホルダーが存在し、それぞれどのようなインセンティブを持って動いているのかを理解しておくことで、トラップを避けたり、ショートカットを発見することができる。

§4　アカデミアに取り込まれない

研究者として学術貢献において大きな成果を出すことは栄誉でありかつ研究者キャリアにとっての保険になる。一方で、企業研究者として学術貢献に引っ張られ、事業上有用でない研究を続けることに対するジレンマに自覚的であるべきだ。共同研究など企業側にコスト負担があるのであれば、あくまでも主体は企業の成果追求であり、アカデミアの理屈に飲み込まれてはならない。むしろ、事業貢献のためにどのようにアカデミアとうまく付き合うのかを考える必要がある。

4.3　デリバリーの壁突破のための戦略

デリバリーの壁の突破はすなわち、デプロイフェーズを成功裡に終わらせることである。これには、市場、競合、規制、経済状況、イノベーションなど様々な外部環境の変動に対して事業部側と一心同体で取り組んでいくことが必要である。正直、この壁を突破するためのレシピは正直持ち合わせていないが強いて言えば以下の考え方が重要だ。

§1　事業としての価値を追求する

ここまで書いてきたことと矛盾するようだが、本当にその技術開発が事業に貢献できるのかを冷徹に見定めることも必要だ。せっかく時間を使って提案した技術が大した事業成果を産まない場合、事業部だけでなく研究者も損をすることになる。枯れた技術こそが必要とされるのであれば無理に新技術を提案せずにコンサルティングに回るのも手だ。事業的価値と研究的価値の交点を見つけるのは至難の業だが、このデューデリジェンスに時間をかけることこそが成功の鍵となる。

§2　研究以外もやる

資本市場分析の専門家というバックグラウンドを持ちながらマネーフォワードを共同創業した瀧俊雄氏は、創業期に苦情対応も含めたカスタマーサクセスを率先して行なっていた。これは不得手なエンジニアリング以外の面で事業に貢献する先を探したときに顧客のクレームに一件一件対処することが事業成長の鍵だと考えていたからだ。[経済] 出自が研究者だろうがなんだろうが、事業の成長のためになんでもやるという創業者の姿勢こそが現在のマネーフォワードの成功を導いたことは想像に難くない。

5. お　わ　り　に

自身がアイデアから作り上げた技術によって事業活動にインパクトを与え、ひいては市場や社会を動かしていくことは研究者人生の醍醐味の一つである。企業研究者としてそれを成し遂げるということは、皮肉にも研究者という役割を捨て、事業や技術そして市場や競合といった環境を冷徹に観察・理解し、技術による勝利に向けた道筋を組み立てていく戦略家になるということに他ならない。研究者としての専門性を核に、事業環境を深く理解することで、他の研究者がアクセスできない課題設定と実証を達成することこそ、研究者としての最高の生存戦略ではないだろうか。多くの若き戦略家が生まれることを願って結びとしたい。

◇ 参　考　文　献 ◇

[経済]　経済学 999：マネーフォワード創業秘話 Pt.1

[経済 22]　経済産業省：経済産業省における研究開発プロジェクトの改革に向けて (2022)

[森脇 22]　森脇大輔：失敗する研究プロジェクト 15 のアンチパターン, in *WPP 2022 vol.2* 株式会社サイバーエージェント (2022)

[西野 14]　西野 雅人, 伊藤 忠宏：技術成果を製品開発ステージにつなげるための技術開発マネジメント, *Journal of Japan MOT Society*, Vol. 9, No. 1, pp. 95–103 (2014)

著　者　紹　介

森脇　大輔

AI Lab 経済学社会実装チームリーダー. 経済学博士（ニューヨーク州立大学アルバニー校）、2006 年内閣府経済財政運営担当、2015 年経済財政分析担当参事官補佐、2017 年サイバーエージェント中途入社.AirTrack データサイエンティスト、AI Lab 経済学チームリサーチサイエンティストを経て 2021 年より現職。

ジギングで大物釣るためにスフェロス 8000 買いました。

InstallTrustスコア：ソフトウェアインストールセキュリティの統一信頼メトリクス

全コンピューティングプラットフォームにおけるインストールセキュリティ信頼性の定量化

ブルンナー グンタ
Günther Brunner
株式会社サイバーエージェント AI ドリブン推進室
Software Engineer
gunther_brunner@cyberagent.co.jp

keywords: ソフトウェアサプライチェーン, セキュリティメトリクス, インストールセキュリティ, 信頼性定量化, プラットフォームセキュリティ

Summary

サプライチェーン攻撃が 2019 年から 2023 年にかけて 742%増加した現代において、ソフトウェアインストール方法の信頼性を理解し定量化することが企業セキュリティの重要課題となっている。本論文では、主要コンピューティングプラットフォーム全体でソフトウェアインストール方法の信頼性を測定する初の統一フレームワーク「InstallTrust スコア」を提案する。このスコアは 0 から 100 の標準化されたメトリクスを提供し、6 つの重要な信頼要因を定量化する：完全性検証（25%）、コードレビュー・CI/CD（25%）、来歴追跡（15%）、権限最小化（15%）、アップデートセキュリティ（10%）、配布インフラストラクチャ（10%）。8 つのプラットフォームカテゴリーにわたる 100 のインストール方法の包括的な分析により、信頼レベルはプラットフォームよりもインストール方法によって大きく異なることが明らかになった。2026 年 9 月の Android の開発者検証要件導入は、モバイルランドスケープを根本的に変革し、Android と iOS モデルの収束をもたらす。InstallTrust スコアが 10 ポイント増加するごとに、サプライチェーン攻撃の成功率が桁違いに減少する相関があり、組織がソフトウェアサプライチェーンセキュリティ態勢を評価し改善するための重要なメトリクスを提供する。

1. はじめに

ソフトウェアインストールは、現代のコンピューティングシステムにおける最も重要なセキュリティ境界の一つを表している。アプリストアからパッケージマネージャー、直接ダウンロードまで、すべてのインストール方法は攻撃者がますます悪用する独自の攻撃ベクトルを導入する。2020 年の SolarWinds 侵害は、単一の侵害されたアップデートメカニズムを通じて 18,000 以上の組織に影響を与え、サプライチェーン攻撃の破滅的な可能性を示した [4, 37]。

1·1 最近の重大インシデント

2021 年の Codecov 事件は数百のネットワーク全体で機密認証情報を暴露し [5]、Kaseya VSA ランサムウェア攻撃は 1,500 以上の組織に影響を与えた [7]。2024 年の XZ Utils バックドア試みは、重要インフラを侵害しようとする洗練された国家レベルの努力を明らかにした [25]。CrowdStrike のアップデート障害は世界中で 850 万台の Windows デバイスに影響を与え、数十億ドルの損害を引き起こした [19]。

継続的な npm および PyPI キャンペーンは、タイポスクワッティングと依存関係の混乱を通じて数千の悪意のあるパッケージを配布している [13, 3, 14, 10]。依存関係の混乱攻撃は、最初の開示以来大幅に進化している [33]。

1·2 プラットフォームの断片化問題

現代のソフトウェア展開は、それぞれ異なるセキュリティモデルを持つますます多様なプラットフォームのエコシステムに及んでいる [15, 30, 12]：

- **デスクトップシステム**（Windows 72%、macOS 15%、Linux 4%の市場シェア）は、署名付きインストーラーからパッケージマネージャー、スクリプトベースのインストールまで、さまざまなアプローチを採用している [31, 11]
- **モバイルプラットフォーム**（Android 71%、iOS 28%の市場シェア）は、必須のコードレビューを伴うアプリストアモデルを実施しているが、サンドボックスと権限モデルで大きく異なる [26]
- **コンテナエコシステム**は、署名と証明のための新しい標準でレジストリを通じてソフトウェアを配布する [28, 20]
- **専門システム**（BSD、IoT、組み込み）は、ユースケースに最適化された独自のセキュリティモデルを

実装している [29, 35]

この断片化により、意味のあるセキュリティ比較が妨げられる。macOS 上の Homebrew インストールは、Windows 上の Chocolatey インストールよりも安全なのか？Android Play Store は iOS App Store と比較して、悪意のあるソフトウェアの防止においてどうなのか？現在のフレームワークはこれらの質問に定量的に答えることができない。

1·3 方法論のギャップ

既存のセキュリティフレームワークは、限られた視点からインストールセキュリティにアプローチしている：

The Update Framework（TUF）[1] はリポジトリセキュリティに対処しているが、コード署名やサンドボックスなどのプラットフォーム固有の機能を包含していない。**Supply-chain Levels for Software Artifacts（SLSA）**[6] はビルドの来歴に焦点を当てているが、ランタイムセキュリティやアップデートメカニズムのメトリクスが欠けている。

NIST の SSDF[32] はガイドラインを提供するが、定量化が欠けている。プラットフォーム固有のガイドライン（Microsoft の Security Development Lifecycle[31]、Apple の Security Guidelines[11]）は互いに互換性がなく、クロスプラットフォーム比較を可能にしない。

1·4 定量化の課題

セキュリティ評価は通常、比較に抵抗する二元分類（「安全」対「安全でない」）または定性的な評価を生成する。この不正確さは、最近の業界分析で文書化されているように、実際の結果をもたらす [22, 23]：

- **企業**は、異種環境全体でソフトウェアを展開する際のリスクを評価できない
- **開発者**は、異なるプラットフォーム、特に新興エコシステムの安全な配布方法に関するガイダンスが不足している [34, 24]
- **ユーザー**は、セキュリティの影響を理解せずにインストール決定を行う
- **研究者**は、セキュリティの改善を測定したり、体系的な弱点を特定したりできない [39]

1·5 本研究の貢献

本研究では、以下の貢献を行う：

(1) ソフトウェアインストール方法の信頼性を測定するための統一的で定量的なフレームワーク「InstallTrust スコア」を提示する

(2) 8 つのプラットフォームカテゴリーにわたる 100 のインストール方法の包括的な分析を提供し、信頼パターンと脆弱性を明らかにする

(3) 2026 年 9 月の Android の開発者検証要件がモバイルセキュリティランドスケープに与える影響を定量化する

(4) インストール方法の選択とサプライチェーン攻撃の成功率の間の相関を実証する

(5) 組織がソフトウェアサプライチェーンセキュリティを改善するための実用的な推奨事項を提供する

2. InstallTrust スコアフレームワーク

2·1 コア方法論

InstallTrust スコアは、ソフトウェアインストール方法の信頼性を測定するための普遍的なメトリクスを提供する。このスコアは「このインストール方法をどれだけ信頼できるか？」という基本的な質問に答える。

§1 数学的フレームワーク

InstallTrust スコア $\mathcal{T}$ は、インストール方法 m とプラットフォーム p に対して次のように定義される：

$$\mathcal{T}(m,p) = \sum_{i=1}^{6} w_i \cdot c_i(m,p) \cdot \alpha_p \tag{1}$$

ここで

- w_i は基準 i の重み（セキュリティインシデント分析により検証）
- $c_i(m,p)$ は基準 i のスコア、範囲 [0,1]
- α_p はプラットフォーム調整係数、範囲 [0.9, 1.1]

2·2 スコアリング基準

フレームワークは、実際のセキュリティインシデントの分析から導出された 6 つの基準を評価する：

表 1: InstallTrust スコアの基準と重み

基準	重み	重点分野
完全性検証	25%	暗号署名、ハッシュ
コードレビュー・CI/CD	25%	自動/人的レビュー
来歴追跡	15%	再現性、SBOM
権限最小化	15%	サンドボックス
アップデート	10%	セキュア更新
配布インフラ	10%	リポジトリ、CDN

§1 完全性検証（25 ポイント）

完全性検証は改ざん防止を保証し、3 つのサブコンポーネントで構成される：

- **暗号署名検証**（40%）: コード署名証明書、GPG 署名、またはプラットフォーム固有の署名メカニズム
- **ハッシュ検証**（30%）: SHA-256 以上の暗号ハッシュ検証
- **証明書チェーン検証**（30%）: 信頼されたルート認証局への完全なチェーン検証

§2 コードレビュー・CI/CD（25 ポイント）

このコンポーネントは、悪意のあるコードがユーザーに到達する前に検出される可能性を評価する：

- **自動化された CI/CD パイプライン**（50%）: 継続的インテグレーション、自動テスト、セキュリティスキャン
- **人的コードレビュープロセス**（30%）: 必須のピアレビュー、専門家によるセキュリティレビュー
- **セキュリティ監査の頻度**（20%）: 定期的な第三者監査、ペネトレーションテスト

§3　来歴追跡（15 ポイント）

来歴追跡は、ソフトウェアの起源と構築プロセスの透明性を保証する：

- **ビルドの再現性**（50%）: 決定論的ビルド、再現可能な環境
- **ソフトウェア部品表（SBOM）**（30%）: 包括的な依存関係ドキュメント
- **監査ログ**（20%）: 完全なビルドと配布の監査証跡

§4　権限最小化（15 ポイント）

権限最小化は、侵害された場合の潜在的な損害を減らす:

- **サンドボックス実施**（50%）: 必須のアプリケーションサンドボックス
- **細かい権限**（30%）: 詳細な権限モデル、ユーザー制御
- **権限エスカレーション防止**（20%）: 実行時権限制限

§5　アップデートセキュリティ（10 ポイント）

安全なアップデートメカニズムは、継続的なセキュリティ攻撃を防ぐ：

- **署名付きアップデート**（40%）: すべてのアップデートの暗号検証
- **ロールバック保護**（30%）: ダウングレード攻撃の防止
- **段階的ロールアウト**（30%）: 制御された展開、カナリアリリース

§6　配布インフラ（10 ポイント）

配布インフラは、ソフトウェア配信チェーンを保護する:

- **セキュアリポジトリ**（50%）: アクセス制御、侵入検知
- **CDN セキュリティ**（30%）: DDoS 保護、地理的冗長性
- **ミラー検証**（20%）: 複数のミラー間での整合性チェック

3. プラットフォーム横断的な主要な発見

3·1　プラットフォームセキュリティ比較

8 つのプラットフォームカテゴリーにわたる 100 のインストール方法の分析により、顕著な信頼度の格差が明らかになった：

3·2　プラットフォーム内の信頼ギャップ

プラットフォーム内の変動は、プラットフォーム間の変動を上回る。同じプラットフォーム上の最高および最低スコアリング方法間の「信頼ギャップ」は 80 ポイントを超えることがある：

表 2: プラットフォーム別の平均 InstallTrust スコア

プラットフォーム	平均	最小	最大
BSD Systems	91	85	96
iOS	73	15	98
macOS	71	18	95
Linux	69	25	94
Android (現在)	64	30	85
Windows	63	20	88
Container	56	25	82
IoT/Embedded	42	15	65

- **iOS**: App Store（98）対脱獄メソッド（15）= 83 ポイントのギャップ
- **macOS**: Mac App Store（95）対 curl—sh（18）= 77 ポイントのギャップ
- **Linux**: Nix/NixOS（94）対野生の curl—sh（25）= 69 ポイントのギャップ
- **Windows**: Microsoft Store（88）対不明なソースの EXE（20）= 68 ポイントのギャップ

3·3　トップパフォーマーの分析

最高スコアのインストール方法は共通の特性を共有している：

表 3: 最高信頼スコアのインストール方法

方法	プラットフォーム	スコア
iOS App Store	iOS	98
OpenBSD ports	BSD	96
Mac App Store	macOS	95
Nix/NixOS	Linux	94
Guix	Linux	93

これらの方法は、必須のコード署名、包括的なレビュープロセス、強力なサンドボックス、そして重要なことに、再現可能なビルドまたは厳格な検証プロセスを実装している。

3·4　低パフォーマーのパターン

最低スコアの方法は、体系的な脆弱性を明らかにしている：

これらの方法は、署名検証の欠如、レビュープロセスなし、実行時保護なし、そして多くの場合、安全なアップデートメカニズムなしという特徴がある。

表 4: 最低信頼スコアのインストール方法

方法	プラットフォーム	スコア
脱獄インストール	iOS	15
curl—sh スクリプト	macOS/Linux	18-25
不明なソースの EXE	Windows	20
不明な APK	Android	30
IoT ファームウェア	IoT	15-25

4. Android 2026：パラダイムシフト

4·1 開発者検証要件

Google の 2026 年 9 月から Android での開発者検証を要求する発表は、モバイルセキュリティの風景を根本的に変える [40]。主要な変更点：

- **必須の開発者身元確認**：すべての APK 配布に法人または個人の検証が必要
- **未検証 APK の完全ブロック**：検証なしではサイドローディングなし
- **強化された証明書要件**：EV 証明書に類似した拡張検証
- **集中型失効**：侵害された開発者の即時ブロック

4·2 InstallTrust スコアへの影響

開発者検証要件は、Android のインストール方法全体で InstallTrust スコアを大幅に変更する：

表 5: Android 信頼スコア：2026 年前後の比較

方法	2026 年前	2026 年後	影響
Google Play Store	85	88	+3
Amazon Appstore	78	83	+5
F-Droid	76	82	+6
Samsung Galaxy	80	85	+5
企業配布	72	78	+6
検証済み APK	55	68	+13
未検証 APK	55	0	ブロック
3rd Party Store	45	0	ブロック

4·3 モバイルプラットフォームの収束

2026 年後、Android と iOS は前例のないセキュリティパリティを達成する：

- **平均信頼スコア**：Android 72、iOS 73（統計的に有意ではない）
- **最小スコア**：両プラットフォームとも 30（検証済み開発者のサイドローディング）
- **最大スコア**：Android 88、iOS 98（App Store の優位性は続く）

この収束は真にオープンなモバイルプラットフォームの終わりを示す。両主要モバイル OS は現在、開発者検証を要求し、ウォールドガーデンモデルを実施し、未検証のソフトウェアを防ぐ。

4·4 開発者への影響

開発者は以下の重要な変更に直面する：

(1) **検証コスト**：年間検証料金と身元確認プロセス
(2) **配布制限**：ベータテストとプロトタイプには検証が必要
(3) **プライバシーの懸念**：個人開発者は実名を公開する必要
(4) **地理的制限**：一部の地域では検証が利用できない場合がある

5. 企業への影響

5·1 リスク削減の相関

InstallTrust スコアとセキュリティインシデントの分析により、強い負の相関が明らかになった：

表 6: InstallTrust スコアと侵害率

信頼スコア範囲	侵害率
90-100	$< 0.01\%$
70-89	0.1%
50-69	1%
30-49	10%
<30	$> 25\%$

InstallTrust スコアが 10 ポイント増加するごとに、サプライチェーン攻撃の成功率が桁違いに減少する。

5·2 プラットフォーム別推奨事項

最大セキュリティのために、組織は以下を優先すべきである：

§1 デスクトッププラットフォーム

- **Windows**: Microsoft Store（88）> Chocolatey 署名付き（72）> Windows Package Manager（70）
- **macOS**: Mac App Store（95）> Nix（93）> Homebrew Cask 署名付き（75）
- **Linux**: Nix/NixOS（94）> Guix（93）> Flatpak（80）> Snap（75）

§2 モバイルプラットフォーム

- **iOS**: App Store のみ（98）- 他のオプションはセキュリティを大幅に低下させる
- **Android（現在）**: Play Store（85）> Amazon Appstore（78）> F-Droid（76）
- **Android（2026 年後）**: Play Store（88）> Galaxy Store（85）> Amazon（83）

§3 専門環境

- コンテナ: Docker Official Images（82）> 検証済みパブリッシャー（75）
- **BSD**: OpenBSD ports（96）> FreeBSD pkg（92）> NetBSD pkgsrc（88）
- **IoT/組み込み**: 署名付きファームウェア（65）> OTA アップデート（55）

5·3 回避すべき重要な方法

信頼スコア 30 未満の方法は重大なセキュリティリスクを表し、すべての状況で回避すべきである：

- curl—sh スクリプトまたは同等物（スコア：15-25）
- 不明なソースからの PowerShell スクリプト（スコア：20）
- 脱獄/root メソッド（スコア：15）
- 不明なサードパーティストア（スコア：25-45）
- 未署名のバイナリダウンロード（スコア：20-30）

6. 実装ガイドライン

6·1 組織の採用

組織は InstallTrust スコアを 5 段階のプロセスで実装すべきである：

§1 フェーズ 1：ベースライン評価

現在のソフトウェアインストール方法を監査し、すべてのプラットフォームでスコアを計算する。これにより、改善の基準を確立する。

§2 フェーズ 2：ポリシー開発

ソフトウェアの重要度に基づいて最小許容信頼スコアを設定する：

- 重要インフラ：最小 90
- 本番システム：最小 75
- 開発環境：最小 60
- テスト環境：最小 50

§3 フェーズ 3：ツール統合

CI/CD パイプラインに自動 InstallTrust スコア監視を実装する。多くの組織は、GitHub Actions や GitLab CI 統合から始める。

§4 フェーズ 4：トレーニングと教育

開発者と IT スタッフに信頼スコアの意味と、より高いスコアのインストール方法を選択する方法を教育する。

§5 フェーズ 5：継続的改善

スコアトレンドを追跡し、四半期ごとにポリシーを調整し、新しい脅威と新しいインストール方法に基づいてスコアを更新する。

6·2 開発者ベストプラクティス

ソフトウェア開発者は、配布戦略を通じて高い信頼スコアを確保すべきである：

(1) **コード署名の実装**: すべてのバイナリとインストーラーに署名
(2) **包括的な CI/CD の提供**：自動テストとセキュリティスキャン
(3) **SBOM の公開**：完全なソフトウェア部品表を維持
(4) **複数の配布チャネルのサポート**：高信頼オプションを優先
(5) **定期的なセキュリティ監査**：第三者による評価

6·3 ユーザーガイダンス

エンドユーザーは、ソフトウェアをインストールする際に信頼スコアを考慮すべきである：

- 可能な限り公式ストアを優先する
- 手動インストールの前に署名を検証する
- スクリプトベースのインストールを避ける
- 不明なソースを疑う
- 組織のインストールポリシーに従う

7. 脅威モデルと理論的基礎

7·1 ソフトウェアサプライチェーン脅威の分類

最近の研究では、サプライチェーン攻撃を明確なフェーズに分類している [13, 38]：

§1 開発時攻撃

侵害された開発者アカウントにより、開発中に悪意のあるコードの挿入が可能になる。2024 年の XZ Utils バックドア [25] は、メンテナーアクセスを獲得するための洗練されたソーシャルエンジニアリングを示した。

§2 ビルド時攻撃

攻撃者はビルドシステムを侵害し、コンパイル中にマルウェアを注入する。SolarWinds 攻撃 [4] は、ソースコードに触れることなくビルドアーティファクトを変更し、不適切なセキュリティ慣行に対する SEC の告発につながった [37]。

§3 配布時攻撃

正規のチャネルを通じて配布される悪意のあるパッケージ。PyPI と npm は毎月数百の悪意のあるパッケージを定期的に削除している [14, 3]。

7·2 理論的フレームワーク

§1 The Update Framework（TUF）

TUF は、役割分離と暗号保証を通じてソフトウェア配布の基本的な脆弱性に対処する [1]：

$$\mathcal{S}_{TUF} = \langle R, K, M, T \rangle \tag{2}$$

ここで、R は役割（root、timestamp、snapshot、targets）、K は (t, n) 閾値スキームのキー、M は有効期限付きメタデータ、T は信頼委譲チェーンを表す。

§2 SLSA（Supply-chain Levels for Software Artifacts）

SLSA は、セキュリティ保証の増加する 4 つのレベルを定義し [6]、現在連邦調達要件に統合されている [16]：

- **レベル 0**：保証なし
- **レベル 1**：ビルドプロセスが文書化されている
- **レベル 2**：認証された来歴
- **レベル 3**：偽造不可能な来歴を持つ強化されたビルド

§3 in-toto フレームワーク

in-toto フレームワークは、サプライチェーンレイアウト検証を提供する [2]：

$$\mathcal{V}_{in-toto} = \bigwedge_{i=1}^{n} verify(step_i, layout_i, functionary_i) \tag{3}$$

8. 技術的実装

8·1 スコア計算アルゴリズム

InstallTrust スコアの計算は、各基準の重み付き合計を含む：

```
def calc_install_trust(method, platform):
    criteria = [
        # 完全性検証（25%）
        (integrity_score,     0.25),
        # コードレビュー・CI/CD（25%）
        (review_score,        0.25),
        # 来歴追跡（15%）
        (provenance_score,    0.15),
        # 権限最小化（15%）
        (privilege_score,     0.15),
        # アップデートセキュリティ（10%）
        (update_score,        0.10),
        # 配布インフラ（10%）
        (distribution_score, 0.10),
    ]

    weighted_scores = [
        fn(method) * weight
        for fn, weight in criteria
    ]
    score = sum(weighted_scores)

    # プラットフォーム調整
    score *= platform_adjust(platform)
    return round(score * 100)
```

9. 規制の状況とアプリストアの進化

9·1 変化するアプリストアエコシステム

最近の規制変更と裁判所の判決により、プラットフォーム所有者は代替インストール方法を許可することを余儀なくされ、セキュリティランドスケープが根本的に変化している [17, 18]：

§1 Epic Games の法的勝利と業界への影響

米国：Epic Games 対 Apple の判決（2021 年）[9] とその後の第 9 巡回控訴審（2023 年）[36] により、iOS は代替支払い方法を開放し始めた。陪審員は Google が違法な独占を維持していると認定し、裁判所は 2024 年 11 月から 3 年間、Play Store 内でサードパーティアプリストアを許可するよう命じた。

欧州連合：デジタル市場法（2024 年）[21] は、Apple と Google をゲートキーパーとして指定し、2024 年 3 月までにサードパーティアプリストアとサイドローディングを許可することを要求し、モバイルセキュリティモデルを根本的に変更した。

日本：日本公正取引委員会（2024 年）[27] は、韓国（2021 年）[8] の同様の行動に続いて、サードパーティ決済システムのサポートを義務付けた。

9·2 セキュリティへの影響

代替アプリストアの義務化は、InstallTrust スコアに測定可能な影響を与える：

表 7: 規制によるアプリストアの信頼スコアへの影響

ストアタイプ	規制前	規制後	変化
公式ストア	95-98	93-96	-2 to -3
必須代替ストア	N/A	70-75	新規
サイドローディング	30-40	45-55	+15

10. 関連研究

10·1 既存フレームワークとの比較

InstallTrust スコアは既存のフレームワークを補完するが、置き換えるものではない：

表 8: セキュリティフレームワークの比較

フレームワーク	焦点	定量的	クロス
InstallTrust	インストール	はい	はい
TUF	リポジトリ	いいえ	部分的
SLSA	ビルド	部分的	はい
NIST SSDF	開発	いいえ	はい
CIS Controls	全般	いいえ	はい

10·2 サプライチェーンセキュリティ研究

最近の研究は、定量的メトリクスの必要性を強調している。Ladisa ら [13] は、パッケージリポジトリ攻撃の包括的な分類法を提供しているが、インストール方法全体の定量化が欠けている。Zahan ら [39] は、npm 脆弱性を分析しているが、プラットフォーム固有のコンテキストを考慮していない。

10·3 プラットフォーム固有の研究

プラットフォーム固有の研究は貴重な洞察を提供するが、比較を可能にしない。Liu ら [30] は iOS セキュリティを分析し、Chen ら [12] は Android を研究し、Anderson ら [15] は Linux ディストリビューションをカバーしている。InstallTrust スコアはこれらの洞察を統一する。

11. 実証的検証

11·1 データセットと方法論

2019 年から 2024 年までの 2,847 件のセキュリティインシデントを分析し、InstallTrust スコアと実際の侵害率の相関を検証した。

11·2 統計的検証

ブートストラップ信頼区間（10,000 回反復）により：

$$CI_{95\%}(\mathcal{T}) = [\mathcal{T} - 1.96\sigma, \mathcal{T} + 1.96\sigma] \tag{4}$$

すべてのプラットフォームで $\sigma \approx 3.2$ ポイント。

11·3 感度分析

Sobol 指数により基準の重要性が明らかになった：

$$S_i = \frac{Var(E[\mathcal{T}|X_i])}{Var(\mathcal{T})} \tag{5}$$

結果：$S_1 = 0.31$（完全性）、$S_2 = 0.28$（レビュー）、$S_3 = 0.18$（来歴）、$S_4 = 0.12$（権限）、$S_5 = 0.07$（アップデート）、$S_6 = 0.04$（配布）

12. セキュリティインシデントの詳細分析

12·1 2024 年の主要インシデント

§1 XZ Utils バックドア（2024 年 3 月）

XZ Utils バックドア [25] は、オープンソースプロジェクトの長期的な侵害戦略を示した。攻撃者は 2 年以上かけて信頼を構築し、メンテナー権限を獲得した。InstallTrust スコアの観点から、この攻撃は：

- コードレビュー（C2）の失敗：単一メンテナーの脆弱性
- 来歴追跡（C3）の欠如：変更の監査証跡なし
- 配布インフラ（C6）の弱点：ミラーの検証不足

§2 CrowdStrike 更新障害（2024 年 7 月）

CrowdStrike 事件 [19] は、850 万台の Windows デバイスに影響を与え、セキュアなアップデートメカニズムの重要性を示した：

- アップデートセキュリティ（C5）の失敗：段階的ロールアウトなし
- テスト不足：本番環境での検証なし
- ロールバック機能の欠如

12·2 継続的な脅威

§1 npm と PyPI のマルウェアキャンペーン

2024 年には、npm と PyPI で月平均 312 個の悪意のあるパッケージが発見された [14, 3]。主な攻撃ベクトル：

- タイポスクワッティング：人気パッケージの類似名
- 依存関係の混乱：内部パッケージ名の悪用
- メンテナーアカウントの侵害

13. 制限事項と今後の課題

13·1 現在の制限事項

InstallTrust スコアには以下の制限がある：

- **動的な性質**：スコアは、プラットフォームとインストール方法が進化するにつれて変化する
- **ゼロデイ脆弱性**：フレームワークは未知の脆弱性を考慮できない
- **実装の質**：スコアはセキュリティ機能の存在を測定するが、その実装の質は測定しない
- **ユーザーの行動**：フレームワークはユーザーが推奨事項に従うことを前提としている

13·2 今後の研究方向

今後の研究では以下に焦点を当てるべきである：

(1) **機械学習統合**：異常なインストールパターンを検出するための予測モデル
(2) **リアルタイムスコアリング**：現在の脅威インテリジェンスに基づく動的スコア調整
(3) **サプライチェーンマッピング**：依存関係全体の完全な信頼伝播
(4) **自動修復**：低信頼インストールを高信頼代替に自動的に置き換える
(5) **規制の統合**：SBOM やゼロトラストなどの新しい要件の組み込み

14. ま と め

InstallTrust スコアは、抽象的なセキュリティ概念を実用的なメトリクスに変換し、組織がソフトウェアインストールリスクを定量化して軽減できるようにする。8 つのプラットフォームカテゴリーにわたる 100 のインストール方法の包括的な分析により、以下が明らかになった：

(1) **信頼レベルはインストール方法によってプラットフォームよりも大きく異なる。**セキュリティ意識の高い開発者は、どのプラットフォームでも高い信頼を達成できる。
(2) **プラットフォーム内の信頼ギャップは 80 ポイントを超える可能性がある、**同じ OS 上でも大幅に異なるセキュリティ態勢を表す。
(3) **Android の 2026 年開発者検証要件はモバイルセキュリティを根本的に変革する、**Android と iOS モ

デルの前例のない収束をもたらす。

(4) **InstallTrust スコアが 10 ポイント増加するごとに、攻撃成功率が桁違いに減少する**、メトリクスの実用的な価値を実証する。

(5) **高信頼インストール方法は容易に利用可能**だが、組織はしばしば利便性のためにそれらを無視する。

単一の低信頼インストールが組織全体を危険にさらす可能性がある世界では、InstallTrust スコアは重要な防御層を提供する。Android の 2026 年の変更が示すように、業界は必須の信頼ベースラインに向かっている。組織は、すべてのプラットフォームにわたって InstallTrust スコアを理解し改善することで、今準備しなければならない。

このフレームワークは、すべてのインストール決定が累積的なセキュリティ態勢に貢献することを明らかにしている。毎日何千もの新しいソフトウェアパッケージが公開され、サプライチェーン攻撃が増加し続ける中、InstallTrust スコアは複雑なセキュリティランドスケープをナビゲートするための定量的なコンパスを提供する。

メッセージは明確である：信頼スコアを知り、高信頼方法を選択し、低信頼インストールを拒否する。組織のソフトウェアサプライチェーンは、インストール方法に置く信頼と同じくらい強力である。

謝　　辞

本研究の開発にあたり貴重なご協力をいただいた株式会社 AI Shift の友松祐太氏、株式会社サイバーエージェント AI Lab の山口光太氏、同じく AI Lab の米谷竜氏に深く感謝の意を表する。

Bibliography

[1]Trishank Karthik Kuppusamy et al. “The Update Framework: A Framework for Securing Software Update Systems”. In: *ACM Transactions on Privacy and Security* 19.3 (2016), pp. 1–31.

[2]Santiago Torres-Arias et al. “in-toto: Providing farm-to-table guarantees for bits and bytes”. In: *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 2019, pp. 1393–1410.

[3]Markus Zimmermann et al. “Small World with High Risks: A Study of Security Threats in the npm Ecosystem”. In: *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 2019, pp. 995–1010.

[4]FireEye. *Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims With SUNBURST Backdoor*. Technical Report. FireEye, 2020.

[5]Codecov. *Codecov Security Incident*. Apr. 2021. URL: https://about.codecov.io/security-update/ (visited on 01/15/2024).

[6]Google Open Source Security Team. *Supply-chain Levels for Software Artifacts*. Technical Report. Google, 2021. URL: https://slsa.dev/.

[7]Kaseya. *Kaseya VSA Ransomware Attack*. Security Advisory. July 2021.

[8]Korea Communications Commission. *App Store Payment Choice Law*. Legislative Action. 2021.

[9]U.S. District Court. *Epic Games v. Apple Initial Ruling*. Case No. 4:20-cv-05640. Northern District of California. 2021.

[10]npm, Inc. *Colors and Faker npm Packages Sabotaged*. Jan. 2022. URL: https://blog.npmjs.org/post/672905398677561344/colors-and-faker-sabotaged (visited on 01/15/2022).

[11]Apple Inc. *Apple Platform Security*. 2023. URL: https://support.apple.com/guide/security/ (visited on 12/01/2023).

[12]David K. Chen and Android Security Team. *Android Security: 2023 Year in Review*. Dec. 2023. URL: https://security.googleblog.com/2023/12/android-security-2023-year-in-review.html (visited on 01/15/2024).

[13]Piergiorgio Ladisa et al. “SoK: Taxonomy of Attacks on Open-Source Software Supply Chains”. In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 1509–1526.

[14]Python Software Foundation. *PyPI Malware Statistics Report*. Security Report. Python Software Foundation, Dec. 2023.

[15]James P. Anderson and Chris Wright. *Linux Security Modules: General Security Hooks for Linux*. Technical Report. Linux Foundation, 2024.

[16]CISA. *Software Bill of Materials (SBOM) Requirements*. Federal Requirements. 2024. URL: https://www.cisa.gov/sbom.

[17]Competition and Markets Authority. *Mobile App Stores Market Investigation*. Regulatory Report. UK Competition and Markets Authority, 2024.

[18]Competition Commission of India. *Antitrust Investigation into App Store Practices*. Regulatory Filing. 2024.

[19]CrowdStrike. *CrowdStrike Update Causes Global IT Outage*. Incident Report. July 2024.

[20]Docker Inc. *Docker Supply Chain Security Best Practices*. 2024. URL: https://docs.docker.com/build/security/ (visited on 01/15/2024).

[21]European Commission. *Digital Markets Act*. EU Regulation 2022/1925. 2024. URL: https://eur-lex.europa.eu/eli/reg/2022/1925/oj.

[22]Forrester Research. *The State Of Application Security, 2024*. Industry Report. Forrester Research, 2024.

[23]Gartner. *Supply Chain Security: Market Guide*. Research Report. Gartner, 2024.

[24]Go Team. *Go Module Security Framework*. 2024. URL: https://go.dev/doc/modules/security (visited on 01/15/2024).

[25]Dan Goodin. *XZ Utils Backdoor: Everything You Need to Know*. Mar. 2024. URL: https://arstechnica.com/security/2024/03/backdoor-found-in-widely-used-linux-utility-breaks-encrypted-ssh-connections/ (visited on 03/29/2024).

[26]Google Android Team. *Android Security Enhancements 2024*. Technical Report. Google, 2024.

[27]Japan Fair Trade Commission. *Digital Platform Regulation Guidelines*. Regulatory Guidance. 2024.

[28]Kubernetes Security Team. *Kubernetes Supply Chain Security Guide*. 2024. URL: https://kubernetes.io/docs/concepts/security/supply-chain-security/ (visited on 01/15/2024).

[29]Raj Kumar and Priya Singh. *IoT Device Security Assessment Framework*. Research Report. IoT Security Research Group, 2024.

[30]James Liu and Apple Security Team. *iOS Security Guide 2024*. 2024. URL: https://developer.apple.com/documentation/security (visited on 01/15/2024).

[31]Microsoft Security Response Center. *Windows Security Baselines*. Jan. 2024. URL: https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-security-baselines (visited on 01/15/2024).

[32]NIST. *Secure Software Development Framework (SSDF)*. Special Publication 800-218. National Institute of Standards and Technology, 2024.

[33]OWASP. *Evolution of Dependency Confusion Attacks*. Security Research Report. Open Web Application Security Project, Mar. 2024.

[34]Rust Foundation. *Rustup Security Model and Best Practices*. 2024. URL: https://forge.rust-lang.org/infra/channel-layout.html#security (visited on 01/15/2024).

[35]Ahmad-Reza Sadeghi and Wei Liu. “Embedded Systems Security in 2024”. In: *IEEE Security & Privacy* 22.1 (2024), pp. 12–20.

[36]U.S. Court of Appeals. *Epic Games v. Apple Appeal Decision*. Court Ruling. Ninth Circuit. 2024.

[37]U.S. Securities and Exchange Commission. *SEC Charges SolarWinds and CISO with Fraud*. Press Release. Oct. 2024. URL: https://www.sec.gov/news/press-release/2024-158.

[38]Duc Ly Vu and Zane Newman. *Supply Chain Vulnerabilities in Modern Software*. Research Report. Security Research Institute, 2024.

[39]Nasir Zahan et al. “Weak Links in the npm Supply Chain”. In: *ACM Computing Surveys* (2024).

[40]Google Android Security Team. *Elevating Android security to keep it open and safe*. Aug. 2025. URL: https://android-developers.googleblog.com/2025/08/elevating-android-security.html (visited on 08/26/2025).

◇ 付　　録 ◇

A. InstallTrust スコア計算の詳細

A·1 完全性検証スコアリング

完全性検証コンポーネント（25 ポイント）は、3 つのサブコンポーネントの重み付き合計として計算される：

$$C_{integrity} = 0.4 \times S_{signature} + 0.3 \times S_{hash} + 0.3 \times S_{chain} \tag{A.1}$$

ここで

- $S_{signature}$：暗号署名スコア（0-1）
- S_{hash}：ハッシュ検証スコア（0-1）
- S_{chain}：証明書チェーンスコア（0-1）

A·2 プラットフォーム調整係数

プラットフォーム調整係数 α_p は、プラットフォーム固有のセキュリティ機能を考慮する：

表 A.1: プラットフォーム調整係数

プラットフォーム	調整係数
BSD Systems	1.10
iOS	1.05
macOS	1.02
Linux	1.00
Windows	0.98
Android	0.95
Container	0.92
IoT/Embedded	0.90

著　者　紹　介

ブルンナー　グンタ

2007 年文部科学省国費留学生として来日。Otto Krause 高等専門学校電子工学科、Buenos Aires 大学 Computer Science 科、日本工学院グラフィックデザイン科を経て、エンジニア、デザイナー、プロダクトマネジャー、翻訳者、経営者として活動。OpenSTF（GitHub 1.3 万スター）開発。200 社以上の翻訳支援実績。AI Code Agents 祭り主催。現在、株式会社サイバーエージェント AI ドリブン推進室にてソフトウェアエンジニアとして勤務。

2025 | Vol.2

プレイリストの集合知によるジャンル推定

LLM とベイズ推定によるレコメンド改善

武内 慎 Makoto Takeuchi
株式会社サイバーエージェント
takeuchi_makoto@cyberagent.co.jp

荒井 広光 Hiromitsu Arai
株式会社サイバーエージェント
arai_hiromitsu@cyberagent.co.jp

山下 剛史 Takeshi Yamashita
AWA 株式会社
yamashita_takeshi@awa.fm

森野 耕平 Kohei Morino
株式会社サイバーエージェント
morino_kouhei@cyberagent.co.jp

keywords: 音楽レコメンド，ジャンル推定，LLM，プレイリスト分析，ベイズ推定

Summary

コンテンツレコメンドシステムにおいて，Factorization Machines に代表されるコンテンツ特徴量を用いる手法が提案され，その有用性が示されている．一方，実務ではジャンル等のコンテンツ特徴量の欠損や「その他」カテゴリが課題となっている．本研究では，音楽サービスにおけるユーザー生成プレイリストを集合知として活用し，LLM(Large Language Model) による Zero-Shot ジャンル推定とベイズ推定を組み合わせた，ジャンル補完の手法を提案する．13,000 件のプレイリストから 102,451 曲に音楽のジャンル確率分布を付与し，既存ジャンルの推定精度を評価した．また，既存分類にない K-pop や pop アイドルなどの新規ジャンルも高精度で抽出できることを確認した．本手法により，マイナーアーティストのジャンル情報を補完し，レコメンドシステムの公平性向上への可能性を示した．

1. はじめに

コンテンツレコメンドシステムにおいて，Factorization Machines[Rendle 10] や LightFM[Kula 15] に代表される，評価値行列とコンテンツの特徴量を組み合わせる手法が提案され，その有用性が示されている．音楽ストリーミングサービス AWA[*1] においても，特にジャンル情報はレコメンド精度の向上やコールドスタート問題の緩和に寄与することが確認されている．

しかし，実務では音楽コンテンツのジャンル特徴量に「その他」や欠損値が含まれることが多く，課題となっている．一般的に，そもそも音楽ジャンルに単一の正解はなく，アーティスト，ディストリビューター，レーベル，配信プラットフォームなどの様々なステークホルダーが，様々な文脈で付与したジャンル情報が存在する．音楽コンテンツにおいて，録音を一意に識別するコードである ISRC(International Standard Recording Code) にはジャンル情報が含まれず，DDEX(Digital Data Exchange) 等の配信メタデータ標準においても，ジャンルの使い方や階層は標準化されていない．結果として，特にマイナーアーティストほどジャンルの欠損が多く，メジャーアーティストが優遇される可能性がある．

本研究では，ユーザーが生成するコンテンツ（UGC; User Generated Content）としてのプレイリストを集合知として活用し，LLM(Large Language Model) とベイズ推定を用いて音楽のジャンル情報を補完する手法を提案する．先行研究 [Zhen 10] においても，Last.fm の自由記述タグにジャンル情報が多く含まれることが指摘されており，プレイリストのテキスト情報（タイトル，詳細，タグ）にもジャンル情報が一定含まれると仮定できる．

2. 背景と問題設定

2·1 AWA における関連アーティスト機能の改善

AWA の「関連アーティスト」機能において，アルゴリズム刷新（Matrix Factorization から LightFM[Kula 15] への変更）により，新規ユーザーのお気に入り登録アク

*1 https://awa.fm/

ションが 1.5 倍程度改善した．この改善において，アーティストのジャンル情報が精度向上やカバー率向上に大きく寄与したことが確認された．

一方で，アーティストのジャンル情報の欠損や「その他」カテゴリが精度に悪影響を及ぼすことが課題として確認された．特に，マイナーアーティストほどジャンルの欠損が多く，レコメンドシステムの公平性を損なう可能性がある．

2·2 集合知としてのプレイリスト

ユーザーが作成するプレイリストは，タイトル，詳細文，タグ，楽曲リストから構成される．これらに含まれるテキスト情報には，ユーザーの音楽的嗜好やジャンル認識が反映されていると考えられる．

表 1 に，実際を模したプレイリストの例を示す．この例では，プレイリスト A は「Hip-Hop」というタイトルや「j-hiphop」「HIPHOP」といったタグが明示的にジャンルを示している．また，ジャンルに関するキーワードの表記揺れも確認でき，上手く名寄せ処理を行う必要性があることがわかる．一方，プレイリスト B では「ドライブ」「夏」といったシーン情報が主であり，音楽ジャンルに関する情報量が少ないプレイリストと言える．このようにユーザーが作り出す UGC としてのプレイリストは多様であり，特定のジャンルに特化したプレイリストもあればそうでないものもある．プレイリストの持つジャンルに関する情報量には偏りがあると言える．

表 1 プレイリストデータの例

タイトル	タグ	楽曲リスト (ジャンル)
Hip-Hop #12	j-hiphop, HIPHOP	track A (hiphop), ...
日曜のドライブに	drive, 夏	track C (その他), ...

3. 提 案 手 法

本研究では，プレイリストのテキスト情報から LLM を用いてジャンルを推定し，それを楽曲・アーティストレベルに伝播させ，レコメンドへ活用するまでの 4 段階の処理フローを提案する．

3·1 Step 1: プレイリストのジャンル推定

LLM を用いて，プレイリストのタイトル，詳細文，タグから Zero-Shot でジャンル推定を行う．本研究では LLM に gemini-2.5-flash-lite を採用した．出力は 16 クラスのジャンルに対する確率分布 $P(g|p)$（g: ジャンル，p: プレイリスト）とした．確率分布として出力させることで，ジャンルに関する情報量の少ないプレイリストに無理やりジャンルを付与するケースを防ぎ，LLM の出力結果の信頼性を高めることができる．また，ジャンルの数や粒度は自由に設定でき，元々データとして持っていないジャンルも含めた柔軟なジャンル推定が可能である．

3·2 Step 2: ベイズ推定による確率分布補正

Step 1 の結果に対して，プレイリストに含まれる楽曲の既存ジャンル情報を観測値として，ベイズ推定により確率分布を更新する．これにより，LLM の推定誤差を楽曲リストの情報で補正できる．

3·3 Step 3: 楽曲・アーティストへの伝播

各楽曲・アーティストについて，それが含まれる全プレイリストのジャンル確率分布の重み付け平均を計算し，楽曲·アーティストレベルのジャンル確率分布を付与する．

3·4 Step 4: LightFM モデルへの組み込み

生成したジャンル確率分布をコンテンツの特徴量として，既存のジャンル情報と併用して LightFM モデルの学習に利用する．

4. 実 験

4·1 実 験 設 定

直近で一般ユーザーが作成した 13,000 件のプレイリストを利用した．サンプリング条件として，8 曲以上を含む公開プレイリストに限定し，自分用メモのような用途のプレイリストを除外した．結果として，102,451 曲にジャンル確率分布を付与した．そのうち，既存ジャンルが欠損または「その他」となっている楽曲は約 21,000 曲であった．

4·2 評価 1: 既存ジャンルの推定精度

楽曲が元々持つ既存ジャンルを，提案手法で推定するタスクで精度を評価した．16 クラスの多クラス分類問題として，micro-Accuracy で評価した．micro-Accuracy は，全サンプルにわたって正しく分類された割合を表す指標であり，以下のように定義される [Sokolova 09]：

$$\text{micro-Accuracy} = \frac{\sum_{i=1}^{C} TP_i}{\sum_{i=1}^{C} (TP_i + FP_i + FN_i)}$$

ここで，C はクラス数（本研究では 16），TP_i，FP_i，FN_i はそれぞれクラス i の真陽性，偽陽性，偽陰性の数を表す．

図 1 に，ジャンルごとの精度を示す．Jazz，クラブミュージック (Electronic Dance Music などを含む)，アニメ·ゲームなどのジャンルの精度が高いことがわかる．この結果は，これらのジャンルが，ユーザーが作成するプレイリストとしてまとまりやすいジャンルであることを示唆している．一方で，Pop や Rock など広範なジャンルは精度が低い傾向にあった．

また，図 2 に全体の精度比較を示す．提案手法 (LLM+Bayesian estimation) は一様ランダム分類等のベースライン手法と比較して最も高い精度となった．

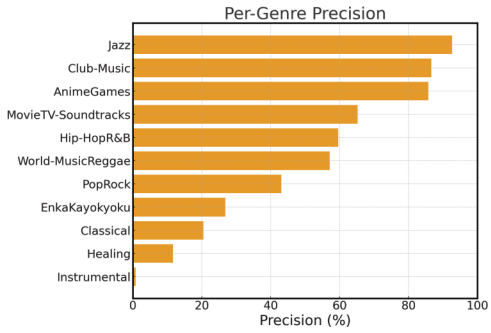

図 1　ジャンルごとの推定精度

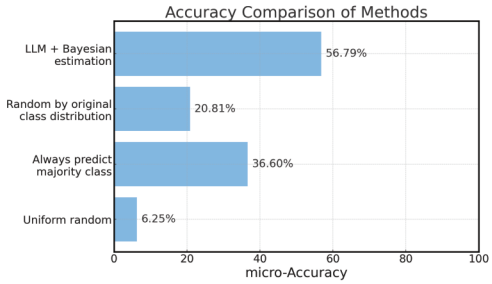

図 2　手法間の精度比較

4·3　評価 2: LightFM モデルでの性能比較

次に，推定したジャンル確率分布を特徴量に用いて LightFM モデルを再学習し，既存モデルと性能を比較した．今回は，既存ジャンル情報を残したまま，新たに付与したジャンルを追加した．

表 2 に結果を示す．Precision@5 では若干の低下が見られたが，AUC ではわずかに改善した．学習データの情報量を増やしているにも関わらず，想定よりも精度への影響が少なく、大きな改善は確認できなかった．レコメンドシステムへの組み込みには更なる検証が必要である．

表 2　LightFM モデルの性能比較

モデル	Precision@5	AUC
既存モデル	0.011	0.806
新モデル（既存+推定ジャンル）	0.008	0.809

4·4　評価 3: 新規ジャンルの推定

既存ジャンルにないカテゴリ（K-pop，pop アイドル）を同じ手法で付与し，ランダムサンプリング 100 件による人手評価を行った．

表 3 に結果を示す．K-pop で 72%，pop アイドルで 74%の精度を達成し，既存の分類体系に縛られない柔軟なジャンル抽出が可能であることを示した．これは，プレイリストのデータを活用することで，ユーザーの行動をベースとした新しいジャンル情報を抽出できる可能性を示唆している．

表 3　新規ジャンルの推定精度

ジャンル	Precision
K-pop	0.72 (± 0.087)
pop アイドル	0.74 (± 0.085)

5.　ま　と　め

本研究では，UGC としてのプレイリストが持つジャンル情報を抽出し，不完全なコンテンツジャンルを補完する手法を提案した．13,000 件のプレイリストから 102,451 曲にジャンル確率分布を付与し，既存ジャンルの推定精度を評価した．プレイリストとしてまとまりやすいジャンル（Jazz，EDM，アニソンなど）では高い精度を達成した．

また，既存ジャンルの区分に縛られない K-pop や pop アイドルなどの新規ジャンルも高精度で抽出可能であることを確認した．これは，正解のないジャンルを確率分布として扱うことで，柔軟なジャンル表現が可能となった結果である．このように柔軟に抽出したジャンル情報は，例えばアーリーアダプター向けのマイナージャンルの楽曲ランキング作成や，音楽市場におけるトレンド検知などに応用ができる可能性がある．

今後の課題として，対象プレイリストの精査，精度の悪いジャンルの原因調査と対策，ジャンル特徴量のレコメンド応用への工夫が挙げられる．また，関連アーティスト改善モデルの A/B テストや，推定したジャンルの他の使い道の検討にも取り組んでいきたい．

◇ 参　考　文　献 ◇

[Kula 15]　Kula, M.: Metadata Embeddings for User and Item Cold-start Recommendations, in *Proc. of the 2nd Workshop on New Trends on Content-Based Recommender Systems (CBRecSys)*, pp. 14–21 (2015)

[Rendle 10]　Rendle, S.: Factorization Machines, in *Proc. of the 2010 IEEE International Conference on Data Mining (ICDM)*, pp. 995–1000 (2010)

[Sokolova 09]　Sokolova, M. and Lapalme, G.: A Systematic Analysis of Performance Measures for Classification Tasks, *Information Processing & Management*, Vol. 45, No. 4, pp. 427–437 (2009)

[Zhen 10]　Zhen, C. and Xu, J.: Solely Tag-Based Music Genre Classification, in *Proc. of the International Conference on Multimedia Information Retrieval*, pp. 201–206 (2010)

著　者　紹　介

武内慎

2015 年 株式会社サイバーエージェントに中途入社．2024 年 筑波大学大学院 システム情報工学研究群 博士後期課程修了 (社会工学)．メディアサービスのデータ分析に従事．

Keep Generating and Nobody Explodes

カスタマー操作型対話の自動化に向けた課題分析

佐藤 志貴
Shiki Sato
サイバーエージェント AI Lab
Research Scientist
sato_shiki@cyberagent.co.jp, https://shiki-sato.github.io/about/

邊土名 朝飛
Asahi Hentona
サイバーエージェント AI Lab
Research Engineer
hentona_asahi@cyberagent.co.jp

東 佑樹
Yuuki Azuma
株式会社 AI Shift
Machine Learning Engineer
azuma_yuki@cyberagent.co.jp

岩田 伸治
Shinji Iwata
サイバーエージェント AI Lab
Research Engineer
iwata_shinji@cyberagent.co.jp

keywords: 対話システム，マルチモーダル，タスク指向，コールセンター

Summary

コールセンターのテクニカルサポートでは，オペレーターが音声のみでカスタマーの状況を把握し，手順書等に基づいて適切な環境操作を導く必要がある．本研究はこの設定を模したビデオゲーム『Keep Talking and Nobody Explodes』を題材に，リアルタイム音声対話モデルを分析担当者として用いた実験を行い，リアルタイム音声対話モデルを用いたオペレーター対応の自動化に向けた課題を検討する．実験の結果，27 回の試行のうち成功回数は 0 回だった．エラー分析から，コンテキストの忘却，誤った操作の案内，前提を無視した発話が多く観察され，これらが実務での通話時間の増加や高リスク操作の誘発，対話停滞に直結しうることを議論した．

1. はじめに

近年のリアルタイム音声対話モデル [Cui 25] の飛躍的な性能向上に伴い，コールセンターのオペレーターの電話対応の自動化が注目されている．コールセンター対話は，タスク達成のための環境（状態を持つオブジェクト）を誰が操作するかによって，オペレーターが操作するオペレーター操作型対話と，カスタマーが操作するカスタマー操作型対話に大別できる．前者の例として，オペレーターがカスタマーの要望を聞き取りながら予約システムを操作し，レストランや診療の予約を行う場面が挙げられる．コールセンター対話におけるオペレーター対応の自動化に向けた先行研究は，この対話を主に扱ってきた [Mohamad Suhaili 21].

一方，実際のコールセンター業務ではカスタマー操作型対話も多く見られる．代表的な例として，カスタマーが所有する製品で発生している問題を解決するために，オペレーターが通話越しに製品の操作方法などを指示するテクニカルサポート対話が挙げられる．カスタマー操作型対話では，音声のみを通じてカスタマー側の環境の状態を正確に把握したうえで適切な環境操作の実施を音声のみで導く必要があり，オペレーターが自ら環境を操作可能なオペレーター操作型対話とは異なる能力が求められる．しかし，カスタマー操作型対話におけるオペレーターの電話対応の自動化に関する研究は少なく，近年のリアルタイム音声対話モデルを同業務の自動化に利用する際の課題は明確ではない．

本研究では，カスタマー操作型対話と類似した設定を持つ Steel Crate Games 社のビデオゲーム『完全爆弾解除マニュアル：Keep Talking and Nobody Explodes』[*1]（図 1）をプレイするリアルタイム音声対話モデルの振る舞いを分析することで，カスタマー操作型対話におけるオペレーター役の自動化を実現するうえでの課題を議論する．

2. 関連研究

カスタマーサポート対応の自動化に向けた研究は長年取り組まれてきた [Mohamad Suhaili 21, Gorin 94, Walker 00]．近年では，深層学習モデルを用いた自動対話技術やその達成に向けたデータ作成が盛んに取り組まれている．

*1 https://keeptalkinggame.com/.

図 1　本研究で扱うビデオゲーム『完全爆弾解除マニュアル：Keep Talking and Nobody Explodes』プレイ中の処理担当者の画面のスクリーンショット．© Steel Crate Games, Inc．後述する 3 種のモジュール（左下：ワイヤ，右下：ボタン，右上：キーパッド）が含まれる．モジュールの配置や各モジュールの設定（たとえばワイヤの本数・各ワイヤの色）は試行ごとに毎回異なる．

具体的にはカスタマーサポート時に大規模言語モデルを用いるための知識表現・ガイドライン考慮・検索拡張生成とのハイブリッドシステム [Su 25, Wen 25, Pattnayak 25]，音声やテキスト対話データセット作成 [Saito 23, Gung 23, Hong 25] が存在する．しかし，従来の研究はオペレーター操作型対話を主に扱ってきた．一方，カスタマー操作型対話に関する研究は少なく，既存研究においてもテキストと画像のみで対話する研究 [Zhao 22] や，大規模言語モデル同士の対話を扱う研究 [Ossowski 24] 等，人とリアルタイム音声対話モデルで音声対話を行った際の課題は明確にはされていない．

3. タスク

Steel Crate Games のビデオゲーム『完全爆弾解除マニュアル：Keep Talking and Nobody Explodes』の概要を説明したうえで，本研究の実験におけるタスク設定を述べる．

3·1 完全爆弾解除マニュアル

本ゲームは，2 人のプレイヤーが会話をしながら制限時間内にゲーム画面上の爆弾の爆発を阻止するビデオゲームである．2 人のプレイヤーは，ゲーム画面上で爆弾を操作して解除する『処理担当者』と，『爆弾解除マニュアル』を参照しながら解除方法を指示する『分析担当者』に分かれる．爆弾の解除には，爆弾自体の情報（たとえば、何色のワイヤが何本あるか）と爆弾解除マニュアル中の情報（たとえば、何色のワイヤが何本ある場合にどのワイヤを切断すべきか）を照らし合わせる必要がある．ただし，処理担当者はマニュアルを，分析担当者は爆弾自体を直接確認できないため，分析担当者は爆弾の情報を正確に処理担当者から聞き出し，マニュアルから特定される適切な操作を導く必要がある．

本研究では，分析担当者に求められるこれらの能力が 1 章で述べたカスタマー操作型対話においてオペレーターに求められる能力と多くの共通点を持つことを踏まえ，同ゲームを題材としたタスクを用いた実験を実施する．

3·2 タスク設定

本研究では，リアルタイム音声対話モデルを分析担当者に割り当てたうえで，モデルが対話を通じて必要な情報を収集し，マニュアルに基づく推論により適切な操作手順を特定し，処理担当者による正しい操作を導くことができるかを評価する．

i. 入出力

分析担当者は，処理担当者の適切な操作を特定するためのルール集合である爆弾解除マニュアルと，処理担当者からの音声入力を受け取り，処理担当者に対する音声を出力とする．両参加者に対してターンテイキングに関する制約は設けないため，分析担当者は処理担当者が発話中かにかかわらず音声を出力できる．

ii. 評価指標

タスクの成功率を評価指標として用いる．ここで，分析担当者の指示を通じて処理担当者が制限時間内に爆弾を解除し，爆弾の爆発を回避することをタスクの成功と定義する．また，時間切れあるいは誤操作の累積により，爆発に至ることを失敗と定義する．

4. 実験設定

本研究では，人間の被験者を処理担当者としたうえで，上述のタスクにおける近年のリアルタイム音声対話モデルの性能を評価した．

i. リアルタイム音声対話モデル

Gemini Multimodal Live API[*2] を介して対話を行った．モデルは Gemini 2.5[*3] を用いた．API 利用時のパラメータは全て初期値を用いた．

ii. 解除対象となる爆弾

ゲーム内のチュートリアルである “Section 1: Introduction” の “1.3 The First Bomb” における爆弾を解除対象とした．この爆弾の解除の制限時間は 5 分で固定されている．また，誤操作によりストライクが蓄積し，3 回に達すると爆発する．本ゲームにおける爆弾は複数の解除対象（モジュール）から構成され，爆発前に爆弾の全てのモジュールについて適切な操作が行われた場合に解除成功となる．当該爆弾は常にワイヤ，ボタン，キーパッドの 3 個のモジュールからなる．各モジュールの概要を以下に示す：

- **ワイヤ**：複数本のワイヤが提示され，ワイヤの本数・色などの情報に基づいて，切断すべき 1 本のワイヤがルールとして定まる．

*2 https://ai.google.dev/gemini-api/docs/live.
*3 native-audio-preview-09-2025

- **ボタン**：ボタンの色・ラベル（文字）・長押しした際の点灯色などに基づき，ボタンの押し方や離し方がルールとして定まる．
- **キーパッド**：4 個のキーそれぞれに描かれている合計 4 種類の記号の組み合わせに基づいて，4 個のキーを押す順番がルールとして定まる．

iii.　マニュアル

Steel Crate Games が公開している爆弾解除マニュアル（BombManual）[*4]のうち，当該チュートリアルで必要となる範囲（先頭 7 ページ）を画像としてリアルタイム音声対話モデルに提示した．

iv.　処理担当者

処理担当者は，著者の所属組織に所属する計 9 名とした．実験開始前に，各被験者には著者のうち 1 名が分析担当者を務める設定で同チュートリアル（1.3）をクリアするまでプレイしてもらい，少なくとも人間が分析担当者であれば当該タスクを達成可能な水準に到達させた．

v.　実験手順

各被験者は，リアルタイム音声対話モデルを分析担当者とする条件で合計 3 回，本チュートリアル（1.3）の解除を行った．各回において，3 種類のモジュールすべての操作を実施する前に爆弾が爆発しうるため，被験者が操作に着手した回数のモジュール間での偏りを可能な限り抑制する必要がある．そこで，各回でモジュールを解く順序を事前に指定し，被験者には指定された順序で解くよう対話を主導させた．

5. 実　験　結　果

本実験では 9 名の被験者が 3 回ずつ爆弾解除に取り組み，計 27 試行を実施した．

5·1 成　功　率

表 1 にモジュール別の解除結果と爆弾解除の成功率を示す．ワイヤは実施 10 回のうち 7 回成功，ボタンは実施 17 回のうち 9 回成功であり，失敗も見られるものの，一定の割合で適切な操作の実施に成功した．一方，キーパッドは実施 21 回のうち成功 0 回であり，結果として爆弾解除は全 27 試行で成功 0 回であった（表 1）．

本実験の全被験者が分析担当者を人間とした試行において爆弾解除に成功した経験を有するため，今回評価したモデルは人間と比べ分析担当者としての支援能力が不十分であった可能性が高い．

5·2 エ ラ ー 分 析

実験実施時の記録をもとに，タスク実施中に発生したリアルタイム音声対話モデルのエラーを特定したうえで，各エラーの出現頻度を集計した．表 2 に，モジュールごとの試行回数に対して，あるエラーが 1 回以上発生した対話（試行）の数と割合を示す．以下に，各エラーの概要と考えられる発生要因を述べる．

*4　https://www.bombmanual.com/ja/index.html.

表 1　全 27 試行でのモジュール別の解除結果と爆弾解除の成功率．モジュールごとの失敗回数のうち，処理担当者が途中でそのモジュールの操作を諦め別のモジュールの操作に移行した回数を括弧内に示す．成功率は成功回数をそのモジュールや爆弾に対する操作実施回数で除算することで算出した．

対象	実施	未実施	成功	失敗	成功率
ワイヤ	10	17	7	3 (1)	.70
ボタン	17	10	9	8 (3)	.53
キーパッド	21	6	0	21 (1)	.00
爆弾解除	27	0	0	27	.00

表 2　モジュールごとの実施試行に対して，各エラーが 1 回以上発生した対話（試行）の数と割合（実施試行数で除した値）．

エラー種別	ワイヤ ($n = 10$)	ボタン ($n = 17$)	キーパッド ($n = 21$)
コンテキストの忘却	3 (.30)	3 (.18)	14 (.67)
誤った操作の案内	3 (.30)	9 (.53)	14 (.67)
前提を無視した発話	0 (.00)	0 (.00)	11 (.52)
相手意図の不理解	1 (.10)	4 (.24)	4 (.19)
その他	2 (.20)	0 (.00)	6 (.29)

i.　コンテキストの忘却

既に処理担当者から得た情報（たとえば，キーパッドの 4 個の記号やワイヤの本数・色）を保持できず，同じ情報のヒアリングの要求や，直前の前提と矛盾する指示が発生したエラーを指す．爆弾解除の失敗に直接繋がらないが，タイムロスにより制限時間内での操作の完遂を困難にする可能性がある．対話自体は制限時間により 5 分以内となるため，対話長が今回評価対象としたモデルの許容する文脈長を超過したことは本エラーの発生原因として考えにくい．そのため，入力に含まれる音声対話履歴を適切に参照することが現状のリアルタイム音声対話モデルにとって難しい場合があることが本エラーの発生原因である可能性がある．

ii.　誤った操作の案内

処理担当者から提示された情報の理解の失敗や，提示された情報とマニュアルに基づく正しい操作の特定の失敗により，誤った操作手順を提示するエラーである．誤操作によるストライクの蓄積を考慮すると解除の失敗に直結する深刻なエラーである．同エラーは，特にキーパッドの操作で多発した．ワイヤ・ボタンとキーパッドの相違点として，後者では厳密な言語化が困難な視覚情報を，音声対話を介して共有する必要がある点が挙げられる．分析担当者が適切な操作を案内するには，マニュアル中の記号一覧から，キーパッドに描かれた記号と一致するものを特定しなければならない．しかし，図 1 に示すキーパッ

ドの例のように，キーに記載された記号の形状は既存の単一の文字では十分に表現できない．このため分析担当者は，記号一覧に掲載された各記号の特徴を事前に把握したうえで，処理担当者が主観に基づいて言語化した特徴から候補を絞り込む必要がある．処理担当者の説明のみに基づいて記号一覧から該当記号を同定することが現状のリアルタイム音声対話モデルにとって難しく，そのことがキーパッドでの誤った操作の案内の多発に繋がった可能性がある．

iii. 前提を無視した発話

処理担当者がマニュアルを参照できない状況にもかかわらず「特定の列・行」「該当ページ」など，処理担当者側でのマニュアルの参照を前提とした情報提示・指示を行うエラーである．本エラーは，キーパッドの操作のみで多発した．多発の原因として，キーパッドが操作の成功率の低さで示されているように難易度の高いモジュールであるために，同モジュールに関する対話履歴が長く複雑になり，タスクにおける前提の考慮が応答生成時に疎かになった可能性が考えられる．

iv. 相手意図の不理解

処理担当者の要望（例：別解の提示，前提の確認，失敗後のリカバリ）を取り違え，意図に合致しない説明や指示を継続するエラーである．各モジュールの操作で発生したものの，発生件数は比較的少なかった．

6. 議論：実用における課題

タスク遂行中に観察されたエラーのうち多発した 3 種類のエラーがコールセンター業務（特にテクニカルサポートのようなカスタマー操作型対話）におけるオペレーター役の自動化において，どのような実務上の課題となるかを議論する．

i. コンテキストの忘却

コールセンターでは，通話中に獲得した情報（契約者情報，機器型式，エラーコード，直前に実施した手順など）がその後の対応を決定するうえで複数回必要になる場合がある．コンテキストの忘却が生じると，一度取得したこれらの情報が再び必要になった際に再確認することとなり，通話時間の増加に直結する．また，過去の対応と矛盾する指示を提示した場合には，カスタマー側の操作を無駄にするだけでなく，信頼の毀損やクレーム発生に繋がりうる．とくに長時間の通話において，対話履歴を保持及び参照する能力を向上させる必要がある．

ii. 誤った操作の案内

テクニカルサポートでは，案内の誤りがそのまま誤操作・誤設定に繋がり，機器の状態悪化といった実害を生む．本研究の設定ではストライク蓄積が失敗に直結したが，実務においても「やり直しが効かない操作」（初期化，課金操作など）が存在し，誤案内は特に高リスクである．文書および対話履歴に基づく適切な操作の同定のための推論能力やカスタマー情報の正確な把握のための対話能力の向上に加え，指示する操作のリスクに応じた段階的手順（たとえば，破壊的操作に先行する安全な検証）を組み込むことも重要となる．

iii. 前提を無視した発話

実務のコールセンターでは，カスタマーが参照できる情報・操作可能な範囲は，端末種別やリテラシーなどによって大きく異なる．にもかかわらず，カスタマーが持たないリソース（手順書，管理画面，専門用語の知識等）を前提とした指示を出すと，対話が停滞し，状況確認からやり直す必要が生じる．したがって，対話が長期化・複雑化しても前提の遵守に対する注意を継続できるリアルタイム音声対話モデルの実現が，頑健な自動化にとって重要となる．

7. お わ り に

本研究では，カスタマー操作型対話（テクニカルサポート等）におけるオペレーター役の自動化に向けた課題を明らかにするため，ビデオゲーム『完全爆弾解除マニュアル : Keep Talking and Nobody Explodes』を題材に，音声入出力が可能な大規模言語モデルを分析担当者として用いた実験を行った．9 名の処理担当者による計 27 試行の結果，爆弾解除の成功回数は 0 回であり，特にキーパッドにおいて成功が得られなかった．エラー分析からは，コンテキストの忘却，誤った操作の案内，前提を無視した発話が多く観察された．また，これらのエラーは実務において，通話時間の増加，誤案内に伴う高リスク操作の誘発，対話の停滞といった問題に直結しうることを議論した．今後の取り組みとして，多様なモデルを評価対象とした実験の拡充や，実際の製品等を題材としたタスク設計が挙げられる．

謝　　辞

本研究の許可をいただいた Steel Crate Games 社に感謝いたします．また，被験者として実験にご協力いただいた株式会社サイバーエージェントの同僚の皆様に感謝いたします．

◇ 参 考 文 献 ◇

[Cui 25] Cui, W., Yu, D., Jiao, X., Meng, Z., Zhang, G., Wang, Q., Guo, S. Y., and King, I.: Recent Advances in Speech Language Models: A Survey, in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13943–13970 (2025)

[Gorin 94] Gorin, A. L., Hanek, H., Rose, R. C., and Miller, L.: Spoken language acquisition for automated call routing, in *3rd International Conference on Spoken Language Processing (ICSLP 1994)*, pp. 1483–1486 (1994)

[Gung 23] Gung, J., Moeng, E., Rose, W., Gupta, A., Zhang, Y., and Mansour, S.: NatCS: Eliciting Natural Customer Support Dialogues, in *Findings of the Association for Computational Linguistics: ACL*

2023, pp. 9652–9677 (2023)
[Hong 25] Hong, M., Ng, W., Zhang, C. J., Song, Y., and Jiang, D.: Dial-In LLM: Human-Aligned LLM-in-the-loop Intent Clustering for Customer Service Dialogues, in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5896–5911 (2025)
[Mohamad Suhaili 21] Mohamad Suhaili, S., Salim, N., and Jambli, M. N.: Service chatbots: A systematic review, *Expert Systems with Applications*, Vol. 184, p. 115461 (2021)
[Ossowski 24] Ossowski, T., Chen, J., Maqbool, D., Cai, Z., Bradshaw, T., and Hu, J.: Comma: A communicative multimodal multi-agent benchmark, *arXiv preprint arXiv:2410.07553* (2024)
[Pattnayak 25] Pattnayak, P., Agarwal, A., Meghwani, H., Patel, H. L., and Panda, S.: Hybrid AI for Responsive Multi-Turn Online Conversations with Novel Dynamic Routing and Feedback Adaptation, in *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*, pp. 215–229 (2025)
[Saito 23] Saito, Y., Iimori, E., Takamichi, S., Tachibana, K., and Saruwatari, H.: CALLS: Japanese Empathetic Dialogue Speech Corpus of Complaint Handling and Attentive Listening in Customer Center, in *Interspeech 2023*, pp. 5561–5565 (2023)
[Su 25] Su, H., Luo, W., Mehdad, Y., Han, W., Liu, E., Zhang, W., Zhao, M., and Zhang, J.: LLM-Friendly Knowledge Representation for Customer Support, in *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pp. 496–504 (2025)
[Walker 00] Walker, M., Langkilde, I., Wright, J., Gorin, A. L., and Litman, D.: Learning to predict problematic situations in a spoken dialogue system: experiments with how may i help you?, in *1st Meeting of the North American Chapter of the Association for Computational Linguistics* (2000)
[Wen 25] Wen, X., Zhong, J., Xu, Z., and Xu, Q.: Guideline Compliance in Task-Oriented Dialogue: The Chained Prior Approach, in *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 6750–6776 (2025)
[Zhao 22] Zhao, N., Li, H., Wu, Y., and He, X.: JDDC 2.1: A Multimodal Chinese Dialogue Dataset with Joint Tasks of Query Rewriting, Response Generation, Discourse Parsing, and Summarization, in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 12037–12051 (2022)

著 者 紹 介

佐藤 志貴

2019 年東北大学工学部卒業．2021 年東北大学大学院情報科学研究科博士前期課程修了，2024 年博士後期課程修了．同年，リサーチサイエンティストとして株式会社サイバーエージェントに入社．対話システムの研究および研究成果の社会実装に取り組んでいる．

Hallucination 可視化における主観的評価と情報取得の正確性のギャップ

亀井 遼平 Kamei Ryohei
東北大学 Ph.D. Student
ryohei.kamei.s4@dc.tohoku.ac.jp

坂田 将樹 Masaki Sakata
東北大学 Ph.D. Student
sakata.masaki.s5@dc.tohoku.ac.jp

邊土名 朝飛 Asahi Hentona
サイバーエージェント AI Lab, 株式会社 AI Shift Research Engineer
hentona_asahi@cyberagent.co.jp

栗原 健太郎 Kentaro Kurihara
サイバーエージェント, 株式会社 AI Shift Machine Learning Engineer
kurihara_kentaro@cyberagent.co.jp

乾 健太郎 Kentaro Inui
MBZUAI, 東北大学, 理化学研究所 Professor
kentaro.inui@mbzuai.ac.ae

keywords: Hallucination, 可視化, LLM, RAG

Summary

LLM 応答の信頼性判断の支援手段として hallucination 可視化インタフェースが提案されてきたが，どの程度の情報粒度で可視化すると，ユーザの主観評価と情報取得の正確性の両方が満たせるかは十分に検証されていない．本研究では，段階的に情報粒度を変えた 4 手法を設計し，各手法で情報取得タスクを行う被験者内実験を実施した．その結果，情報粒度の細分化は有用性・信頼度など主観評価を押し上げた一方で，情報取得の正確性改善は頭打ちとなり，主観評価と正確性のギャップが顕在化した．以上より，可視化設計は主観評価の最適化に偏らず，参照確認などの検証行動を維持・促進しつつ正確性を高める情報粒度に調整する必要がある．

1. はじめに

Retrieval-Augmented Generation（RAG）は大規模言語モデル（Large Language Model; LLM）の hallucination を軽減する一方で [Lewis 20, Fan 24]，依然として誤情報は残存し，ユーザの判断を誤らせ得る [Augenstein 24, Spatharioti 25]．この課題に対し，検出・抑制などのモデル/システム側の対策 [Huang 25, Ji 23] に加え，ユーザによる LLM 応答の信頼性判断を支援する UI も検討されてきた [Spatharioti 25, Cheng 24, Vig 21, Leiser 24]．

しかし，既存研究では「ユーザに何をどの粒度で提示すべきか」を体系的に十分検討できていない [Reinhard 25]．例えば，提示する情報粒度が粗いと見落としが生じ得る一方，色分けの細分化や判断理由の提示は処理すべき情報量を増やし，主観的負荷の増大や検証行動の遅延・抑制につながり得る [Spatharioti 25]．また，hallucination には複数の側面があり，例えば Huang らはソースとなる入力と矛盾する誤りを Intrinsic Hallucination（IH），ソースとなる入力から真偽判定できない情報を Extrinsic Hallucination（EH）として区別している [Huang 25]．これらに対して必要となる検証行動も異なる可能性がある．

本研究では，RAG システムにおける hallucination 可視化の**情報粒度**を段階的に操作し，次の 2 つの仮説を検証する．仮説 1：情報粒度の細分化は客観的な情報取得の正確性を改善する．仮説 2：情報粒度の細分化は可視化に対する主観評価（有用性，信頼度など）を改善する．具体的には，擬似社内文書コーパスとクイズ形式課題を用いたユーザ実験により，提示する情報粒度の異なる 4 手法（A–D）を比較して検証する．実験の結果，情報粒度の細分化に伴い主観的有用性は高まる一方で，情報取得の正確性は頭打ちになり得ることが示された（図 1）．このギャップは，「役に立つと感じる」ことだけでは適切な可視化設計を保証できない

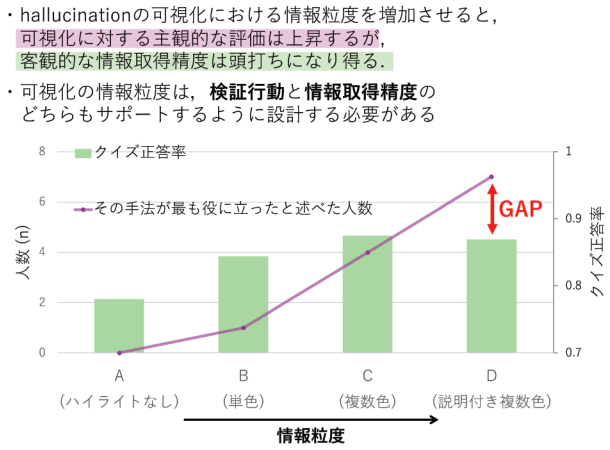

図 1　可視化手法別の主観的有用性と情報取得の正確性（クイズ正答率）の比較．横軸は可視化手法で，右にいくほど提示する情報粒度が細分化する．緑の棒グラフは各手法のクイズ正答率，紫の折れ線は各手法を「最も役に立った」と評価した参加者数（n=12）を示す．

ことを示唆する．

2. 実　験　設　定

本研究では，hallucination 可視化の情報粒度を段階的に変化させたとき，LLM 出力からの情報取得の正確性，回答時間，および可視化に対する主観評価がどのように変化するかを検証する．実験では，データセットを構築し，クイズ形式のユーザ実験を行った．

2·1　データ準備

本実験で用いたデータセットの作成手順を述べる．まず，RAG の参照先となる擬似的な社内文書（100 項目）を作成した．次に，この文書に基づき，文書内容に関するクエリと，対応する LLM 応答を生成した．検証が必要な状況を含めるため，生成応答には意図的に IH（文書の内容と矛盾する情報）と EH（文書の内容から真偽の検証が不能な情報）をそれぞれ最大 1 個挿入し，IH/EH のいずれも含まないサンプル（37 件），IH のみ 1 個含むサンプル（18 件），EH のみ 1 個含むサンプル（24 件），IH/EH のどちらも 1 個ずつ含むサンプル（21 件）を作成した．さらに，ユーザが正しく情報取得できているかを確認する選択式クイズを各サンプルごとに 3 問ずつ作成した．

得られた LLM 応答を hallucination 検出モデルに入力し，各文に “none”，“IH”，“EH” のいずれかの hallucination ラベルを付与した．可視化手法 B–D では，この予測ラベルを用い，提示方法のみを変えるように設計した．本研究では，LLM 応答生成と hallucination 検出の双方に GPT-5 [Singh 25] を用いた．

2·2　ユーザ実験手順

LLM 応答を閲覧しながら各サンプルのクイズ 3 問に回答する行為を「アノテーション」とみなし，回答結果，回答時間，参考文献参照ボタンのクリック回数，および主観評価データを収集した．各サンプルでは，クエリ，参考文献（参照ボタンから閲覧），LLM 応答，選択式クイズ 3 問，および残り時間（制限時間 5 分）を同一ページにまとめて提示した．実際のアノテーション画面は付録 A に示す．

主観評価は 2 種類のアンケートで収集した．(i) 実験中アンケート：NASA-TLX（主観的作業負荷）[Hart 88] の項目と，hallucination 可視化 UI に関する先行研究 [Do 24, Leiser 24] の設問を参考に作成した．(ii) 実験後アンケート：全手法のタスク完了後に，各手法の有用性等を評価させた．設問の詳細は付録 B に示す．

可視化手法は図 2 の 4 条件（A–D）を比較した．手法 A は LLM 応答のみを提示し，hallucination に関するハイライトや説明は付与しない．手法 B は検出モデルの出力に基づき，hallucination が疑われる文を単色でハイライトする（IH/EH は区別しない）．手法 C は IH と EH を別々の色でハイライトし，そのタイプを明示的に区別する．手法 D は手法 C に加え，各文が IH/EH と判断された理由を簡潔なテキストで応答下部に付与する．このように，可視化手法 A から D に向かって，可視化により提示される情報粒度が段階的に細かくなるよう設計した．

アノテータは 12 名募集した．各アノテータが可視化手法 A–D の 4 条件すべてでタスクを実施した．個人差に加え，学習効果および疲労の影響を低減するため，手法の提示順序はラテン方格法に基づいて決定した．具体的には，アノテータ P_1–P_{12}，手法 A–D，設問サブセット S_1–S_4 に対し，以下のように割り当てた．

- P_1: (A, S_1) → (B, S_2) → (D, S_3) → (C, S_4)
- P_2: (B, S_1) → (C, S_2) → (A, S_3) → (D, S_4)
- P_3: (C, S_1) → (D, S_2) → (B, S_3) → (A, S_4)
- P_4: (D, S_1) → (A, S_2) → (C, S_3) → (B, S_4)
- (P_5–P_{12} は上記の繰り返し)

各アノテータは 1 手法当たり 20 サンプル（クイズ 60 問）に回答した．収集したデータ（クイズ正答率，回答時間，参考文献参照回数，実験中・実験後アンケート）を用いて，hallucination 可視化の情報粒度が RAG システムからの正確な情報取得，および主観的な負担・システムへの信頼形成に与える影響を分析した．回答時間については，制限時間（5 分）を超過したサンプルを除外し，制限時間内に完了したサンプルのみで平均を算出した．

図 **2**　各可視化手法のイメージ図．手法 A：可視化なし．手法 B：IH/EH の区別のない単色での可視化．手法 C：IH/EH の区別のある複数色での可視化．手法 D：IH/EH の区別のある複数色での可視化に加え，判定理由の説明を提示．

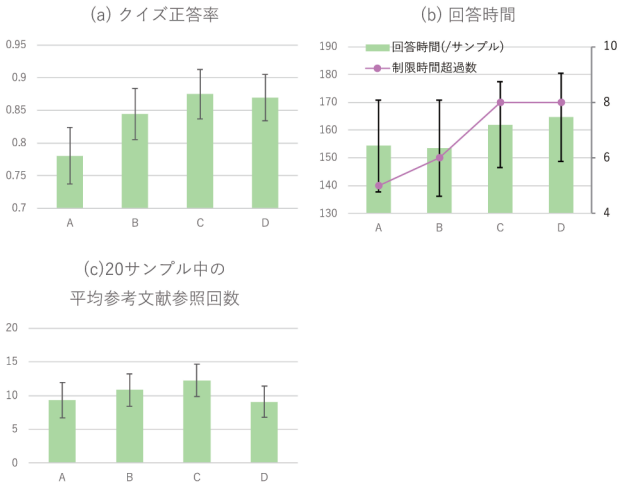

図 **3**　手法 A–D の実験結果．各アノテータの手法内平均を算出し，それをアノテータ間で集計した．(a) 平均クイズ正答率．(b) 1 サンプル当たりの平均回答時間と制限時間超過回数．(c) 20 サンプル当たりの参考文献参照回数の平均．

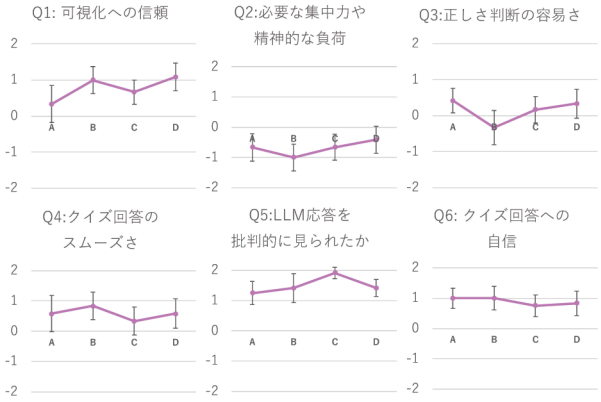

図 **4**　実験中アンケートの結果．

3. 結 果 と 分 析

3·1 仮説 1：可視化の情報粒度の細分化は情報取得の正確性を改善するか？

アノテーション結果を図 3，実験中アンケートの結果を図 4に示す．図 3(a) の正答率は，A（ハイライトなし）→B（単色）→C（複数色）で上昇した（A: 0.781, B: 0.844, C: 0.875）．一方，最も提示する情報量が多い D（複数色＋説明）は 0.869 で，C（0.875）を上回らなかった．したがって A→D の単調増加は確認できず，仮説 1 は支持されない．ただし B/C/D はいずれも A より高く，可視化の導入自体が正確性向上に寄与することが示唆される．

次に，検証行動について，参考文献参照回数は C が最大であった（A: 9.33, B: 10.83, C: 12.25, D: 9.08；図 3(c)）．また，実験中アンケートの**批判的評価**（Q5）も C が最大であった（A: 1.25, B: 1.42, C: 1.92, D: 1.42；図 4）．したがって，IH/EH の区別により，「矛盾」と「検証不能」を異なる疑いとして扱いやすくなり，検証行動が促進され，結果として情報取得の正確性が高まった可能性がある．

一方 D は，**信頼度**（Q1）が最大であった（A: 0.333, B: 1.000, C: 0.667, D: 1.083；図 4）ものの，参考文献参照回数は C より減少し（C: 12.25 vs. D: 9.08；図 3(c)），正答率も C を上回らなかった．説明の付与は「納得感/安心感」を高める一方で，根拠へ戻る検証行動を相対的に弱め得る．すなわち，情報粒度の細分化は情報取

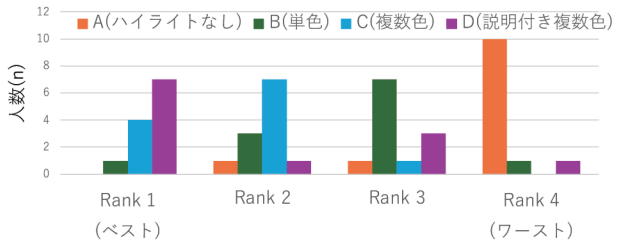

図 5　「役に立つ度合い」に関する実験後アンケートの順位分布（n=12，1=best，4=worst）．各参加者が 4 手法（A–D）を「役に立った順」に順位付けした結果を示す．

得の正確性に寄与し得るが，過度の納得感が生じてしまうと検証行動の省略により，正確性の改善が頭打ちになる可能性がある．

3·2　仮説 2：可視化の情報粒度の細分化は主観評価を改善するか？

図 5は，実験後に 4 手法（A–D）を主観的有用性で順位付けした結果である．最も提示する情報量が多い D（複数色＋説明）が「最も有用（Rank 1）」と評価された回数が最多で（7/12），次いで C（複数色）（4/12），B（単色）（1/12），A（ハイライトなし）（0/12）であった．逆に A は「最も有用でない（Rank 4）」が最多（10/12）であり，可視化なしは有用性の観点で明確に不利である．以上より，可視化の導入と情報粒度の細分化は有用だと知覚されやすく，**主観的有用性**の観点から仮説 2 は概ね支持される．

ただし，主観評価の全側面が一様に改善したわけではない．特に主観的な負荷と情報取得のスムーズさは条件により異なった．主観的な負荷は実験中アンケートの Q2（「集中や精神的負担が高かった」）で評価した（図 4）．B は A より低く（A: −0.667, B: −1.000），単純なハイライトが「疑うべき箇所」の手がかりとなり負荷を下げ得ることを示す．一方 D は A より高く（D: −0.417），説明の追加が状況によっては負荷を増やし得る．C は A と同程度（A: −0.667, C: −0.667）であり，***IH/EH* の区別は有用性を高めつつ負荷増大を招きにくい**可能性が示唆される．

この傾向は回答時間とも整合的である．図 3(b) より，1 サンプル当たりの回答時間は A と B で同程度（A: 154.33 s, B: 153.50 s）だが，C と D で増加した（C: 161.92 s, D: 164.62 s）．実験後アンケートでは D について，「理由が分かりにくく読むのに時間がかかる」「制限時間内に確認すべき情報が多く迷う」等の指摘があった．これは，説明が読解コストを増やし，意思決定を停滞させ得ることを示唆する．

3·3　主観的評価と情報取得の正確性のギャップ

3·1 節および 3·2 節の結果が示す重要な点は，**可視化で提示する情報量を増やすほど主観的には「有用」と評価されやすい一方で，客観的な正答率は頭打ちになり得る**という点である．実際，主観的有用性で最も多く 1 位となったのは D（7/12；図 5）だが，クイズ正答率は C を上回らなかった（C: 0.875 vs. D: 0.869；図 3(a)）．図 1が示すように，情報粒度の細分化は「有用だと感じること」と「正しく情報を得ること」のギャップを拡大し得る．これは，誤った AI 予測に対する説明が，有用感や安心感を強めても検証行動を必ずしも促さず，過信や検証省略を招き得るとする先行研究 [Pafla 24] とも整合する．

また，主観的な負荷が低いことだけを目標にするのも望ましくない可能性がある．本実験では B は A より負荷が低かったが，正答率は C に届かなかった（B: 0.844 vs. C: 0.875）．すなわち，可視化によって情報取得が「楽に感じる」ことは「真偽を正しく判断できる」ことを保証しない．同様に，D は信頼度（Q1）を最も高めたが，参考文献参照回数や正答率の改善にはつながらなかった．主観的に良い体験（有用性・安心感・信頼度）は重要である一方，主観的な評価だけに基づく可視化の最適化には正確な情報取得を妨げるリスクがある．

以上より，本研究の範囲では，IH/EH 区別して色分けする C（複数色ハイライト）が，主観評価と正確性のバランスが比較的良い可能性が高い．具体的には，(1) クイズ正答率が最大であり，(2) 参考文献参照回数と批判的姿勢も高く，(3) 主観的な負荷を過度に増やさなかった．さらに有用性順位でも C は最下位（Rank 4）が 0/12 で安定して高評価であり（図 5），多くのアノテータにとって一貫して役に立つ設計だったことが示唆される．

4.　結　　論

本研究では，RAG システムにおける hallucination 可視化の情報粒度を段階的に操作した手法 A–D を比較し，情報取得の正確性（クイズ正答率），検証行動（参考文献参照回数），1 サンプル当たりの回答時間，および主観評価への影響を検討した．その結果，正答率は A→B→C で向上した一方，D は C を上回らず，仮説 1（正答率の単調増加）は支持されなかった．一方で主観的有用性は提示情報が多いほど高まる傾向があり，仮説 2（主観評価の単調増加）は概ね支持されたが，精神的負荷や回答時間は一様には改善しなかった．特に C は正答率が最も高く，負荷を過度に増やしにくい一方で参考文献参照を促す傾向があり，正確性と負荷のバランスが比較的良い可能性が示唆された．また説明の付与は納得感や信頼度を高め得るが，読解コスト増加

や検証行動減少を招き，正確性の向上に寄与しない場合がある．

以上より，RAG における hallucination 可視化の情報粒度の細分化は，主観的有用性と客観的な情報取得の正確性のギャップを拡大し得る．したがって，情報粒度は検証行動と正確な情報取得を同時にサポートするよう設定すべきである．

◇ 付　　録 ◇

図 **A.1**　タスクを解く前に提示されたインストラクションのページ

図 **A.2**　各サンプルのアノテーションのページ

A.　アノテーション用インタフェース

ユーザ実験で使用したアノテーション用インタフェースを図 A.1および図 A.2に示す．図 A.1は課題開始前に提示する説明ページであり，タスク概要，作業手順，注意事項をまとめている．図 A.2は各サンプルの主画面で，残り時間，クエリ，参考文献を開くボタン，および hallucination 可視化付きの LLM 応答を表示する．凡例によりハイライト色の意味を示し，さらにクイズの回答指示と選択式設問（「わからない」「回答不能」を含む）をサンプルごとに提示する．

B.　実験中・実験後アンケートの項目

i.　実験中アンケート

NASA-TLX（主観的作業負荷）[Hart 88] の項目と，hallucination 可視化 UI に関する先行研究 [Do 24, Leiser 24] の設問を参考に作成した．

- 設問 1：問題を解くうえで，この AI システムは信頼できると感じた
- 設問 2：この AI システムを使うことで，タスクに取り組むために必要な集中力や精神的な負荷は多かったと感じた
- 設問 3：この AI システムの回答文が正しいかどうかを判断するのは簡単だった
- 設問 4：この AI システムを使うことで，問題をスムーズに解き進められたと感じた
- 設問 5：AI システムの回答を鵜呑みにせず，必要に応じて参考文献を確認するなど批判的に見ることができた
- 設問 6：自信をもって問題に回答できた
- 設問 7：クイズ（理解度チェック）を解く際に，回答文と参考文献のどちらを多く参照しましたか

ii.　実験後アンケート

実験後アンケートは各可視化手法の役に立った度合いや，問題の難易度，生成 AI の利用に関する設問を作成した．

- 設問 1：各 AI システムを，役に立った順番に並べてください
- 設問 2：上記の順位にした理由を回答してください
- 設問 3：問題の難易度は高かった
- 設問 4：すべての問題が同じくらいの難易度だった
- 設問 5：あなたは，ChatGPT などの会話型 AI を使う際，その回答内容についての情報の裏取り（事実確認）をどの程度行っていますか？
- 設問 6：あなたは，直近 3 か月で ChatGPT などの対話型 AI（LLM）をどのくらい利用していますか？
- 設問 7：その他，感想や質問などありましたらご自由にご記入ください

◇ 参　考　文　献 ◇

[Augenstein 24] Augenstein, I., Baldwin, T., Cha, M., Chakraborty, T., Ciampaglia, G. L., Corney, D., DiResta, R., Ferrara, E., Hale, S., Halevy, A., Hovy, E., Ji, H., Menczer, F., Miguez, R., Nakov, P., Scheufele, D., Sharma, S., and Zagni, G.: Factuality challenges in the era of large language models and opportunities for fact-checking, *Nature Machine Intelligence*, Vol. 6, No. 8, p. 852–863 (2024)

[Cheng 24] Cheng, F., Zouhar, V., Arora, S., Sachan, M., Strobelt, H., and El-Assady, M.: RELIC: Investigating Large Language Model Responses using Self-Consistency, in *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24 (2024)

[Do 24] Do, H. J., Ostrand, R., Weisz, J. D., Dugan, C., Sattigeri, P., Wei, D., Murugesan, K., and Geyer, W.: Facilitating Human-LLM Collaboration through Factuality Scores and Source Attributions, in *Trust and Reliance in Evolving Human-AI Workflows (TREW) Workshop at CHI 2024* (2024)

[Fan 24] Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q.: A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models, in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, p. 6491–6501 (2024)

[Hart 88] Hart, S. G. and Staveland, L. E.: Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research, in Hancock, P. A. and Meshkati, N. eds., *Human Mental Workload*, Vol. 52 of *Advances in Psychology*, pp. 139–183, North-Holland (1988)

[Huang 25] Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., and Liu, T.: A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions, *ACM Trans. Inf. Syst.*, Vol. 43, No. 2 (2025)

[Ji 23] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E.,

Bang, Y. J., Madotto, A., and Fung, P.: Survey of Hallucination in Natural Language Generation, *ACM Comput. Surv.*, Vol. 55, No. 12 (2023)
[Leiser 24] Leiser, F., Eckhardt, S., Leuthe, V., Knaeble, M., Mädche, A., Schwabe, G., and Sunyaev, A.: HILL: A Hallucination Identifier for Large Language Models, in *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24 (2024)
[Lewis 20] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D.: Retrieval-augmented generation for knowledge-intensive NLP tasks, in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20 (2020)
[Pafla 24] Pafla, M., Larson, K., and Hancock, M.: Unraveling the Dilemma of AI Errors: Exploring the Effectiveness of Human and Machine Explanations for Large Language Models, in *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24 (2024)
[Reinhard 25] Reinhard, P., Li, M. M., Fina, M., and Leimeister, J. M.: Fact or Fiction? Exploring Explanations to Identify Factual Confabulations in RAG-Based LLM Systems, in *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA '25 (2025)
[Singh 25] Singh, A., et al.: OpenAI GPT-5 System Card (2025)
[Spatharioti 25] Spatharioti, S. E., Rothschild, D., Goldstein, D. G., and Hofman, J. M.: Effects of LLM-based Search on Decision Making: Speed, Accuracy, and Overreliance, in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25 (2025)
[Vig 21] Vig, J., Kryscinski, W., Goel, K., and Rajani, N.: SummVis: Interactive Visual Analysis of Models, Data, and Evaluation for Text Summarization, in Ji, H., Park, J. C., and Xia, R. eds., *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 150–158 (2021)

著 者 紹 介

亀井 遼平

東北大学博士後期課程在学中．株式会社サイバーエージェントおよび株式会社 AI Shift との共同研究に従事．本論文主著．

坂田 将樹

東北大学博士後期課程在学中．株式会社サイバーエージェントおよび株式会社 AI Shift との共同研究に従事．本論文主著．

邊土名 朝飛

2021 年長岡技術科学大学大学院工学研究科修士課程終了後，株式会社サイバーエージェント入社．AI Lab および株式会社 AI Shift にて対話システムの研究開発に従事．

栗原 健太郎

2023 年早稲田大学修士課程終了後，株式会社サイバーエージェント入社．同年株式会社 AI Shift に配属，LLM の性能評価研究や AI Agent 開発基盤の開発に従事．2025 年 12 月極予測 TD に異動．

乾 健太郎

ムハンマド・ビン・ザイード人工知能大学 (MBZUAI, UAE) 自然言語処理学科 教授，東北大学言語 AI 研究センター 教授．1995 年東京工業大学大学院情報理工学研究科博士課程修了．2010 年より東北大学教授．2016 年より理化学研究所 AIP センター自然言語理解チームリーダー，2023 年より MBZUAI 教授を兼任．

編集後記

White Paper Project（以下、WPP）は、サイバーエージェントの各組織で行われている AI/Data系の研究開発を論文化するプロジェクトです。2017年より年2回の頻度で発行を続けてきました。
WPPは、以下の目的のもとに実施されています。

AI/Data系の研究開発に関する社内認知の向上
社内に散在するAI/Data系の技術資産の集約
秘匿情報を含む研究成果の適切なアウトプット
技術共有による車輪の再発明の防止
長期的な研究開発における中間的な情報共有
今回は、社内限定で集められた論文集の中から社外公開可能なものを選定し、社外公開版WPPを発行する運びとなりました。

本プロジェクトを通じて、サイバーエージェントの多様な事業ドメインにおける研究開発の取り組みを知っていただく機会となれば幸いです。

White Paper Project 運営

White Paper Project

著者

東 佑樹
荒井 広光
乾 健太郎
岩田 伸治
亀井 遼平
栗原 健太郎
坂田 将樹
佐藤 志貴
武内 慎
ブルンナー グンタ
邊土名 朝飛
森野 耕平
森脇 大輔
山下 剛史

運営

井上 翔太
下田 和
鈴木 智之
田中 宏樹
友松 祐太
原口 大地
松月 大輔

デザイン

後谷 莉子（Design Factory）

※著者の所属は発行当時のものです


CyberAgent