修士学位論文

題目

国際標準への適合を目的としたソフトウェア要求仕様書の LLM による再構造化とその適用可能性評価

指導教員

楠本 真二

報告者

岡本 琉生

令和 8 年 2 月 2 日

大阪大学 大学院情報科学研究科

コンピュータサイエンス専攻

令和 7 年度 修士学位論文

国際標準への適合を目的としたソフトウェア要求仕様書の LLM による再構造化とその適用可能性評価

岡本 琉生

内容梗概

ソフトウェア要求仕様書（SRS）はソフトウェア開発の成功にとって重要である．SRS の品質はプロダクトの品質やプロジェクトの成否に影響を与えると広く認識されている．SRS には IEEE 830 や ISO/IEC/IEEE 29148 といった国際標準規格がある．これらは要求工学（RE）プロセスとその成果物である SRS についての規格であり，特に SRS に推奨される節構造（標準構造）と各節に書くべき内容を定義している．しかし，ある調査によると，要求エンジニアの知識不足や組織文化，コストの問題などの理由で，標準を適用しない場合が多いと報告されている．実際，構造に限定しても SRS には多様性がある．そこで本研究では任意の節構造を持つ SRS を標準構造に自動で変換すること（構造標準化）を考える．SRS の構造標準化を考える利点の 1 つが，SRS を対象とする RE 研究の適用可能性の向上である．これは対象の SRS が標準構造という共通の構造に従うという仮定をより現実的にできることに起因する．実際，対象の SRS の各節が標準構造の各節に対応付けられることを前提とした品質評価や管理の研究も存在する．構造標準化のアプローチとしては自然言語処理で高いパフォーマンスを発揮している大規模言語モデル（LLM）を用いる．4 種類の LLM を用いて，2 種類の zero-shot プロンプトによる構造標準化への適用可能性を評価する．結果，生成された標準化 SRS は，ある LLM とプロンプトの組合せでは元の SRS に対する正確性に優れ，別の組合せでは元の SRS の記述内容の保持に優れる傾向があることがわかった．ただし，その両方の観点を同時に高水準で満たすことは難しいと評価できた．また，SRS に含まれるべき項目に言及しているかといった点でも LLM と プロンプト間で違いがみられた．最も良い組合せでは SRS に必要な項目のうちの約 9 割に該当する記述を含んでおり，ISO 29148 による SRS の記述の定義に忠実な標準化 SRS を生成できる可能性が示された．

主な用語

要求工学（RE; Requirement Engineering），ソフトウェア要求仕様書（SRS; Software Requirements Specification），国際標準規格，ISO/IEC/IEEE 29148 節構造，標準化，大規模言語モデル（LLM; Large Language Model）

目次

図目次

表目次

1　はじめに

ソフトウェア要求仕様書（SRS；Software Requirement Specification）は開発対象となるソフトウェアの仕様を定義した文書である．SRS はソフトウェア開発には不可欠であり，その成功にとって重要である．SRS の品質が最終的なプロダクトの品質や，プロジェクトの成否そのものにも影響するといった指摘もある [3, 4]．SRS は英語のような自然言語を主として記述されるため，自然言語処理（NLP；Natural Language Processing）を利用した研究が多くなされている [5]．近年では NLP タスクで高い成果をあげる大規模言語モデル（LLM；Large Language Model）を用いた研究が増加している [6]．

IEEE 830 [7] や ISO/IEC/IEEE 29148 [1] は要求工学（RE；Requirement Engineering）プロセスやその成果物である SRS の国際標準規格である．これら標準は SRS が満たすべき品質特性をはじめとし，それらを踏まえて推奨される SRS の節構造（標準構造）や各節に記述されることがらを定義している．Franch らはこれら標準の，産業界での認知と利用の状況について調査している [8]．要求工学（RE）の実務家の半数弱はそもそも標準を認知していなかったこと，知っていたとしても，その多くが知識やスキルの不足，組織文化，コストの制約などが原因で標準を利用していないことが報告されている．実際，SRS の構造という観点に限定してもその形式は多種多様である．

任意の構造を持つ SRS を標準構造に変換すること（構造標準化）は多くの価値をもたらすと考える．1 つはステークホルダ間のコミュニケーションの支援である．標準は共通の理解と読み方を提供するため，多様なステークホルダに SRS の統一的な理解をもたらす．先述の Franch らの調査でも，標準の利用が関係者とのコミュニケーションを容易にすると言及されている．もう 1 つは SRS を対象とする研究の促進である．SRS の構造標準化はこれらの研究結果の適用を可能にし，これからの研究での SRS の構造の仮定を現実的にする．実際，対象の SRS が標準構造に従うこと，あるいはその節が標準構造の節との対応がとれることを前提とした研究が存在する [9, 10]．Aoyama と Nakane は SRS の品質解析方法とその自動化ツールを提案している [9]．SRS の構造の多様性を許容する設計ではあるが，適用には対象の SRS が持つ独自構造の節と標準構造の節とが対応付けられていることを前提とする．Chikh と Aldayel は IEEE 830 の推奨する構造に従う SRS を対象に，成果物を関連付ける情報であるトレーサビリティを管理するスキーマを定義している [10]．

そこで本研究では，SRS の構造標準化という新たなタスクに LLM を用いてアプローチし，その適用可能性を評価する．以降，構造標準化された SRS を標準化 SRS と呼ぶことにする．標準化 SRS には 2 つの性質が要求される．1 つは元の SRS に含まれる情報が必要十分に含まれること，もう 1 つは標準に沿って情報が適切に配置されていることである．実際の SRS を対象に，4 種類の LLM と 2 種類の簡易な zero-shot プロンプトにより構造標準化し，得られた標準化 SRS をこれら 2 つの観点から評価した．1 つ目の観点については，NLP における事実検証（FV; Fact Verification）タスクの技術を用

いた指標よって評価した．2 つ目の観点については，標準化 SRS に含めるべき項目をチェックリスト化し，それを用いた著者らによる目視確認によって評価した．

実験の結果，生成された標準化 SRS は LLM やプロンプトの組合せの違いにより，元の SRS を基準とする正確性と情報の保持性のバランスに差異がみられた．ある組合せでは正確性に優れる一方で，別の組合せでは情報の保持性に優れる傾向が確認されるものの，両指標を同時に高水準で満たすことは容易ではないと評価できた．標準化 SRS に含まれるべき項目について記述されるかという点でも，同様に LLM とプロンプト間で違いがみられた．最も良い組合せではすべての対象 SRS について平均して 90% 以上の項目を含んでおり，ISO 29148 による記述の説明に忠実な標準化 SRS を生成できる可能性が示された．また，LLM による SRS の構造標準化は要求記述の十分性の検証やレビュー支援，SRS の要約など，多様な応用可能性を有することが考察された．

本研究の貢献は以下の 2 点に整理できる．1 つに，SRS の構造標準化という新たなタスクを定義し，その目的および課題設定を明確化した点である．もう 1 つに，LLM による SRS 生成結果を内容と構造という 2 つの観点から評価するフレームワークを提案した点である．特に，内容の保持性の評価では，すでに完成している SRS を入力として用いることでその内容が生成する SRS に保持されるべき情報とみなすことができるため，再現性の考慮が可能となる．

2　背景

IEEE 830 [7] と ISO/IEC/IEEE 29148 [1] は RE プロセスやその成果物である SRS についての国際標準規格である．後者は前者の後継規格であるため，本研究では後者を用いることとし，ISO 29148 と呼ぶことにする．図 1 は ISO 29148 推奨する SRS の節構造（標準構造）である．

1 Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Product overview
  - 1.3.1 Product perspective
  - 1.3.2 Product functions
  - 1.3.3 User characteristics
  - 1.3.4 Limitations
- 1.4 Definitions

2 References

3 Requirements
- 3.1 Functions
- 3.2 Performance requirements
- 3.3 Usability requirements
- 3.4 Interface requirements
- 3.5 Logical database requirements
- 3.6 Design constraints
- 3.7 Software system attributes
- 3.8 Supporting information

4 Verification
- (parallel to subsections in Section 3)

5 Appendices
- 5.1 Assumptions and dependencies
- 5.2 Acronyms and abbreviations

図 1　ISO/IEC/IEEE 29148 [1] で推奨される SRS の節構造

これに加え，3.4 節の “Interface requirements” を除いたすべての節について何を記述すべきかも ISO 29148 では説明されている．例えば，3.3 節の “Usability requirements” については

> Define usability and quality in use requirements and objectives for the software system that can include measurable effectiveness, efficiency, satisfaction criteria and avoidance of harm that could arise from use in specific contexts of use.（[1] より引用）

とあり，ソフトウェアの利用のしやすさや利用時の品質要件，およびその目的の記述が求められている．また，これらの要件には有効性などの観点での測定可能な基準を含めるべきだともされている．

標準構造は SRS に含まれるべき項目を定義しているともいえる．すなわち，標準構造を参考に SRS の品質を評価できる．Thitisathienkul と Prompoon は IEEE 830 の標準構造の各節に対応するトピックの出現有無によって SRS の品質を評価する方法を提案している [11]．各節の出現と SRS の品質特性とを関連付けて品質を数値化する手法である．Takoshima と Aoyama も同様に，標準構造の各節と品質特性とを関係づけた Inspection Matrix を定義し，実際の SRS を題材に評価している [12]．彼らは SRS の構造の違いを吸収するため，対象となる SRS の持つ構造の節と標準構造の節とを対応付ける Translation Matrix という仕組みを定義している．しかし，この Translation Matrix は SRS ごとに人が作成する必要がありコストがかかる．また，節単位で対応付けるのみで，節内のどの記述がどの節に対応するかまでは扱われない．本研究では LLM を用いて標準との対応付けを自動化し，さらには節単位のマッピングではなく，内容自体を標準構造に従って再配置することを考える．この構造標準化によって標準に基づいた SRS の品質評価を支援できる可能性がある．

OpenAI 社の GPT や Google 社の Gemini のような LLM は大量のテキストを用いて学習された言語モデルである．これらは文章生成や翻訳などの自然言語処理タスクにおいて高い性能を示している [13]．近年，RE 分野に LLM を統合する研究が増加しており [6]，その中には SRS などの仕様書を対象とした研究も含まれている．Krishna らは SRS の作成および修正における LLM の能力を評価し，LLM の利用によって達成される工数削減について調査している [14]．その結果，生成された SRS は詳細な記述に欠けるものの，初学者レベルのソフトウェアエンジニアが作成したそれと同程度の品質を有すると報告している．この結果に基づき，LLM は仕様化における生産性向上に寄与しうると結論づけている．また，Norheim と Rebentisch は自然言語で記述された要求を LLM を用いて構造化する手法を提案している [15]．彼らの手法では要求文を EARS [16] という要求のテンプレート構造や線形時相論理といった半形式的な構造へ変換する．従来のルールベース手法では多数のカスタムルールを必要としていたのに対し，LLM を用いることで少数の例示のみで変換が可能であるとを報告している．Zhu らは SRS の生成エージェントである ReqInOne を提案している [17]．ReqInOne は自然言語文書を入力とし，仕様化の過程を要求の抽出と分類，要約の 3 つのタスクに分割して SRS の生成を自動化する．生成された SRS は初学者レベルの要求エンジニアが作成したそれと比較して，一貫性や正確性といった品質特性において同等以上であると評価されている．さらに，抽出および生成された要求の品質が人手によるものより高いことや，要求分類タスクにおいて，BERT を転移学習した手法 [18] と比較して，特に未見のデータに対する一般化性能で優位であることが示されている．

これらの研究から，SRS の生成や要求記述の変換といったタスクに対する LLM の適用可能性が示唆される．本研究で試みる SRS の構造標準化は SRS 生成のサブタスクとみなせる．これは入力として元の SRS が与えられるため，生成する SRS として記述すべき内容が必要十分に与えられている状況になるためである．したがって，本タスクに LLM の利用を試みることには意義があるといえる．

3　Research Questions（RQs）の設定

本研究の目的は任意の構造を持つ SRS を標準構造に変換すること，すなわち SRS の構造標準化である．SRS の構造標準化には 2 つの満たすべき性質があると考えられる．1 つは元の SRS の内容が構造標準化後の SRS，すなわち標準化 SRS にも必要十分に記述されていることである．標準構造にうまく書き換えられたとしても，記述内容が元の SRS から変わってしまっては意味がないためである．例えば，元の SRS と内容に乖離がある標準化 SRS に対して記述内容の品質を評価しても，その結果を元の SRS に直接には反映できない可能性がある．もう 1 つは標準構造に沿って記述が適切に配置されていることである．特に，各節の記述内容が標準で規定される記述内容となっているかが要求される．これは SRS の構造標準化の目的および定義そのものといえる．

本研究では，構造標準化のアプローチの 1 つとして LLM を用いる．そして，2 つの満たすべき性質を踏まえて再生成された SRS を評価することで，本タスクへの LLM の適用可能性を評価する．そこで以下の 2 つの RQ（Research Question）を設定し，これらへの回答を目指す．

RQ1. LLM によって構造標準化された SRS は元の SRS の情報をどの程度保持するのか．
RQ2. LLM によって構造標準化された SRS は標準構造にどの程度忠実であるのか．

なお，本実験では各 RQ を相互に独立した観点として評価する．RQ2 は「標準構造への適合性」に焦点を当てたものであり，生成された記述が内容の真偽にかかわらず適切なセクションに配置されているかを評価する．したがって，RQ1 の評価結果とは切り離して判定を行う．

4　実験設計

図 2 に本実験のながれを示す．LLM によって SRS を構造標準化させ，得られた標準化 SRS を RQ1 と RQ2 の観点から評価する．LLM の出力は確率的であるため，本実験では構造標準化を 3 回試行して平均をとることでそのばらつきによる影響を緩和する．出力された標準化 SRS それぞれについて，後述する指標によって元の SRS からの内容保持の程度を評価する．その後，最も高い適合率を示した標準化 SRS を用いて，チェックリストを用いた目視での確認によって標準への忠実性を評価する．

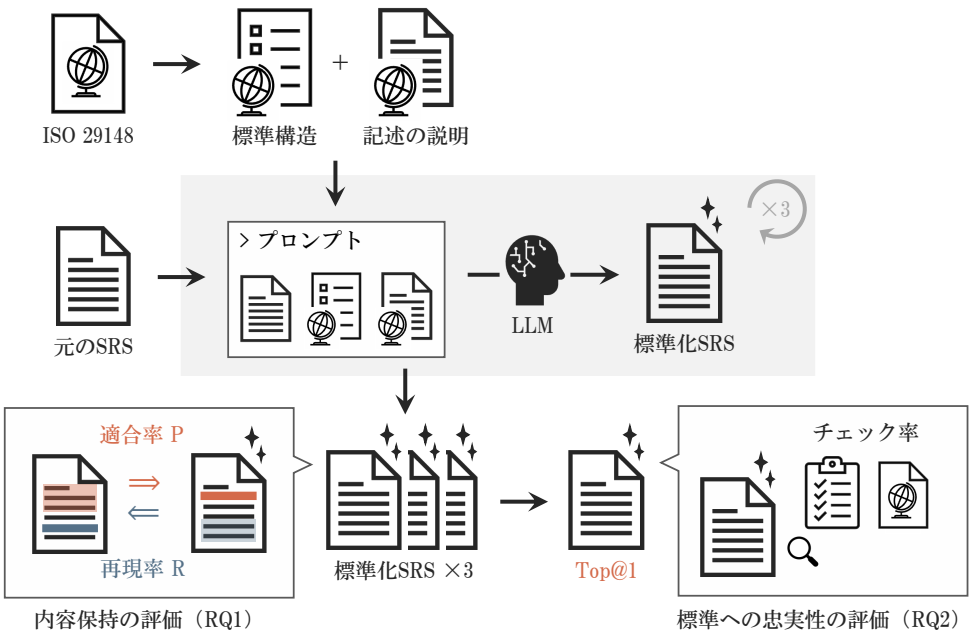

図 2　実験のながれ

4.1　LLM とプロンプト

本実験では LLM として OpenAI 社の `gpt-4o`[*1] と `gpt-5`[*2]，Google 社の `gemini-2.5-flash` と `gemini-2.5-pro`[*3] を利用する．いずれも実験設計時点における各社の最新かつ代表的なモデルであり高い推論能力を有する．特に，長大なコンテキストウィンドウに対応しているため，SRS のような長文な文書全体を入力として扱える．また，共通の API を通じて安定的に利用可能であり，実験の再現性

[*1] `https://platform.openai.com/docs/models/gpt-4o`
[*2] `https://platform.openai.com/docs/models/gpt-5`
[*3] `https://ai.google.dev/gemini-api/docs/models`

や比較可能性を担保しやすい点も選定理由の 1 つである．推論時には温度パラメータは設定可能な最小値にして再現性を高める．`gpt-5` のみが 1 で，それ以外は 0 に設定する．

SRS の構造標準化に必要な情報は元の SRS と標準構造，および各節に記述されるべき内容の説明である．よって，本実験ではこれらの情報をプロンプトに組み込む．標準構造には図 1 を，各節に記述されるべき内容の説明には 2 節で引用したような ISO 29148 に記載の該当部分を用いる．

本実験では 2 種類の zero-shot プロンプトを設計した．プロンプト全体は付録 A に記載している．1 つ目は最も単純な方法であり，一度に先述の情報をすべて LLM に与え，一度で標準化 SRS を生成するように指示する．この方式を $\mathtt{P}_{\mathtt{all}}$ と呼称することにする．2 つ目は各節ごとに標準化 SRS を生成させる方式である．生成する節を指定し，元の SRS 全体と指定した節の説明を与え，標準化 SRS の当該節の生成を指示する．各節の生成が終了したのち，それらを結合して標準化 SRS とする．この方式を $\mathtt{P}_{\mathtt{each}}$ と呼称することにする．一度の生成に用いられるトークン数を削減して出力の精度の向上を狙う．$\mathtt{P}_{\mathtt{all}}$ と $\mathtt{P}_{\mathtt{each}}$ にかかわらず，その後処理として標準構造としての節が欠損していないかを確認し，欠損している節は内容を `[NULL]` として補う．

与える指示の中には，“Retain necessary and sufficient information from the original SRS.” といった文言を含める．これは RQ1 の動機に対応し，標準化 SRS に元の SRS の情報を保持するように促す．また，“Output `[N/A]` for any required content missing from the original SRS.” の文言も含める．元の SRS に存在しない情報を，標準化 SRS に不適切に追加することを防止するためである．`[NULL]` との比較により，LLM が生成に失敗したのかと，元の SRS にそもそも存在しなかったと LLM が判断したのかとを区別するためでもある．出力形式には Markdown 記法を指定し，ヘッダとパラグラフ，リストのみを含めることを許す．特に，“Do NOT use any tables, code blocks, HTML, or extension syntax (e.g., Mermaid).” の指示を含める．これは RQ1 での文分割を容易にするためである．なお，本実験で利用する SRS はこれらの記述要素を含まないため，表現力を制限しない点に注意する．

4.2 データセットと前処理

実験題材として PURE データセット [2] に含まれる SRS の一部を利用する．表 1 に利用する SRS と，後述する前処理を経た後の語数および文数を示す．語への分割には NLTK[*4] の `word_tokenize` を，文への分割には spaCy[*5] の `sentencizer` を用いた．

題材となる SRS の選定基準は 3 つある．1 つ目は SRS であると宣言していることである．PURE には機能要求仕様書（FRS; Functioinal Requirements Specification）やシステム要求仕様署（SyRS; System Requirements Specification）であると自身で宣言している要求仕様書が含まれるため，それらは除外することにする．2 つ目は画像を含まないことである．本実験ではテキストのみからなる SRS

[*4] `https://github.com/nltk/nltk`
[*5] `https://github.com/explosion/spaCy`

を対象に LLM の構造標準化能力を評価する．(10) は画像を含むが，表紙ページにキャプションなしで掲載されているだけで SRS の理解には影響しないので，本実験セットに含める．3 つ目は表を本文中に含まないことである．表は文章と異なり，読解する際の方向が一意ではない．RQ1 の評価において文単位での扱を必要とするが，文への分割方法が一意ではなくなるため，本実験では対象外とする．

先述の通り，すべての対象 SRS に 3 つの前処理を行った．まず，表紙と目次，各ページのフッダおよびヘッダを除外した．これらの情報は SRS を再生成させるにあたって不要な情報である．次に，文分割のためにすべての箇条書きの項目ごとに句読点を付した．項目が完全な文の際にはピリオドを，そうでない場合にはカンマを行末に付す．最後の項目にはその内容にかかわらずピリオドを付す．入れ子になっている場合は上記を再帰的に適用する．最後に，生のテキストから Markdown 書式に整形した．利用する文法要素は 1 つ以上の「#」から始まるヘッダ記法と「-」を用いるリスト記法のみである．いずれの対象 SRS も目次を持つため，そこでの節立てに従って節タイトルをヘッダ化する．ただし，ヘッダ行は文として扱わない．すべての箇条書きはリスト記法で書き直す．

4.3 RQ1 の評価

標準化 SRS が元の SRS の情報をどの程度保持しているかを評価するための指標を定義する．任意の標準化 SRS を SRS_{std}, 元の SRS を SRS_{orig} とする．SRS_{orig} に対する SRS_{std} の適合率 (Precision) $P(SRS_{\mathrm{std}}, SRS_{\mathrm{orig}})$ と再現率 (Recall) $R(SRS_{\mathrm{std}}, SRS_{\mathrm{orig}})$ を以下のように定義する．

$$P(SRS_{\mathrm{std}}, SRS_{\mathrm{orig}}) = \frac{1}{|SRS_{\mathrm{std}}|} \sum\nolimits_{s \in SRS_{\mathrm{std}}} \mathbb{I}[SRS_{\mathrm{orig}} \Rightarrow \mathrm{dectx}(s, SRS_{\mathrm{std}})],$$

$$R(SRS_{\mathrm{std}}, SRS_{\mathrm{orig}}) = P(SRS_{\mathrm{orig}}, SRS_{\mathrm{std}}).$$

ここで，$D \Rightarrow s$ は前件を文書 D，後件を文 s とする含意を表す．$\mathbb{I}[D \Rightarrow s]$ は D が真であった場合に s も真であれば 1，そうでなければ 0 となる．また，$\mathrm{dectx}(s, D)$ は文脈を D とした s の脱文脈化

表 1　PURE データセット [2] から選出した SRS

SRS の名前（略称）		語数	文数
2001 - libra	(01)	5,207	202
2003 - qheadache	(03)	2,433	160
2007 - get real 0.2	(07g)	2,582	119
2007 - puget sound	(07p)	3,950	159
2008 - vub	(08)	8,192	376
2009 - library	(09)	4,329	156
2010 - home 1.3	(10)	4,355	149

[19] された文を表す．すなわち，dectx(s, D) はそれ単独で解釈可能な文であり，その意味が D を考慮した s の真理条件的意味と等価であることを意味する．

適合率 P は標準化 SRS の全文のうちどの程度が元の SRS から指示されるかを示している．したがって，P が大きいほど標準化 SRS に含まれる情報が元の SRS に含まれる情報に由来していると判断できる．再現率 R は逆方向の評価指標である．R が大きいほど元の SRS に含まれている情報が標準化 SRS に保存されていると判断できる．これらの 2 つの指標によって標準化 SRS が元の SRS の情報をどの程度保持しているかを評価する．

いずれの指標もその解釈容易性のために文単位での割合で定義している．例えば，P が 0.8 ならば標準化 SRS の全文のうちの 80% が元の SRS から正しかったと解釈できる．さらに，SRS 内のどの部分が他方から含意されなかったかを文単位で特定できるため，修正すべき箇所を明確に提示できる．同様の指標としては FActScore [20] が挙げられる．FActScore では LLM によって検証対象の文章（claim）を atomic claims と呼ばれる最小単位の文集合に（生成して）分解し，それらの含意割合を事実性スコアとして算出する．解釈容易性の観点では表層的な文単位で計測する方が望ましいと考える．[20] では文単位での検証は厳しすぎると評価しているが，SRS の重要性を考慮すると本実験ではこの性質は問題にならない．各文の事実検証においては，それらを前もって脱文脈化を適用することにより文脈情報の欠落を防止している．脱文脈化の方法である代名詞解決などは曖昧な記述を文脈に基づいて具体化させ，検証の判定基準をより厳格化する．

評価指標の実装

標準化 SRS の文への分割には spaCy の `sentencizer` を用いる．元の SRS のときと同様にヘッダ行，すなわち節タイトルや副節タイトルは文分割に先立って除外しておく．加えて，`[N/A]` と `[NULL]` を含むような行もあらかじめ除外する対象とする．これらのマーカーは元の SRS において該当する記述が存在しないことを明示するが，その真偽を元の SRS の記述のみに基づき客観的に判定することは不可能である．「SRS として記述されるべき項目に対し，関連する要求や情報が存在しないこと自体も明記すべきである」という見解もありうるが，本実験ではこれら記述の不在を表すマーカーを一律に除外することでそれらに対する真偽判定を保留する立場をとる．

$D \Rightarrow s$ の判定は NLP における事実検証（FV; Fact Verification）[21] あるいは自然言語推論（NLI; Natural Language Inference）[22, 23] のタスクが該当する．Transformer ベースの事前学習済みモデルを用いた機械学習によるアプローチが主流である [24, 25, 26]．本実験では [24, 25, 26] で提案された fine-tuning 済みの公開モデルのうち [25] の RoBERTa を FV モデルとして利用する．

選定の際には LLM-AggreFact ベンチマーク [25] のサブセットでの性能を比較基準とした．同ベンチマークに含まれるデータは検証対象となる文章（claim）と根拠となりうる文章（document）の組で

表 2　LLM-AggreFact サブセットにおける fine-tuning 済みモデルの性能比較

fine-tuning 済みモデル		TPR	FNR	FPR	TNR	G-mean
AlignScore [24]	bin （RoBERTa-large）	0.872	0.128	0.504	0.496	0.658
	nli （RoBERTa-large）	0.846	0.154	0.410	0.590	0.707
	reg （RoBERTa-large）	0.905	0.095	0.581	0.419	0.616
MiniCheck [25]	DeBERTa-v3-large	0.203	0.797	0.102	0.898	0.427
	Flan-T5-large	0.769	0.231	0.377	0.623	0.692
	RoBERTa-large	0.746	0.254	0.252	0.748	**0.747**
FactCG [26]	DeBERTa-v3-large	0.909	0.091	0.437	0.563	0.715

あり，各組に対して “Supported” あるいは “Unsupproted” の 2 値の正解ラベルが人間によって付与されている．全データのうち claim が 1 つの文のみからなり，かつ document の語数が $1,000$ 以上 $10,000$ 未満のデータに限定してサブセットを構成した．これは本実験での判定時の状況に近づけるためである．特に，document に対する語数の制限は，表 1 に示したように実験対象の SRS の語数が数千であることによる．サブセットには正例データが $5,230$ 件，負例データが $1,228$ 件の計 $6,458$ 件のデータが残った．性能指標としては，再現率（TPR; True Positive Rate，正例データのうちモデルが正例と判定した割合）と 特異度（TNR; True Negative Rate，負例データのうちモデルが負例と判定した割合）の幾何平均（G-mean）を用いた．正例と負例のどちらに対してもバランスよく精度を要求するためである．表 2 に比較結果を示す．$\mathrm{FNR} = 1 - \mathrm{TPR}$ かつ $\mathrm{FPR} = 1 - \mathrm{TNR}$ である．いずれのモデルも分類の閾値は 0.5 に設定した．つまり，出力層での正クラスについての値が 0.5 以上ならば正ラベルと判定する．なお，すべてのモデルには扱えるトークン数に限界があるため，それに基づいて入力を分割している．RoBERTa は 512 語，DeBERTa と Flan-T5 は $2,048$ 語を上限とし[*6]，それらから claim の語数を減じた数を単位として document をチャンクに分割する．各チャンクごとに推論を行って正クラスの出力値を得て，その最大値を閾値との比較に用いる．

$\mathrm{dectx}(s, D)$ の実装には Gunjal と Durrett が提案した LLM に基づく脱文脈化手法 [27] を用いる．本手法ははじめに文中の曖昧性を抽出し，次にその抽出結果を用いて文を書き換えるという 2 つのステップのプロンプト方法からなる．彼らは GPT-4o mini を用いて，この手法が情報の最小単位を保持しつつ，単体での検証可能性を最大化できると実証している．本実験ではこの各ステップの処理を担う LLM として `gpt-oss:20b`[*7] を利用する．各文に対して多段階の LLM 推論が要求されるため，商用 API を利用した場合の累積的な計算コストとスケーラビリティに課題が生じるためである．また，SRS

[*6] 厳密にはトークンと語は別の単位である．モデルごとにトークナイザが異なることに加え，解釈の容易性および実装上の簡便性から，本実験では NLTK の `word_tokenize` による語数をトークン数の近似として用いた．

[*7] `https://ollama.com/library/gpt-oss:20b`

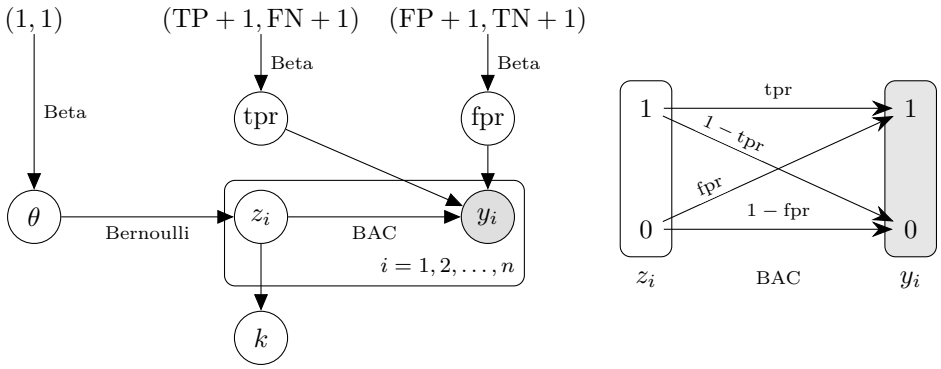

図 3　各文に対する判定結果の生成と誤判定のモデル化

のタイトル（表 1 の名前ではなく，原文の表紙に記載のもの）と s を含む節を連接した文章を与える文脈 D として用いる．温度パラメータは [27] と同じ 0.75 に設定する．出力のぶれを吸収するために 1 つの SRS につき 3 回の脱文脈化を行い，それらでの計測値の平均を最終的な計測値とする．

計測値の補正

先述の通り，$D \Rightarrow s$ の実装には fine-tuning 済みの FV モデルによる判定を用いる．検証対象の文の列 $(s_1, s_2, \ldots, s_n)$ に対する FV モデルの判定結果を $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ とすると，定義した指標 P や R は $\sum_i y_i/n$ と計算できる．しかし，表 2 からわかるように FV モデルは完全ではない．例えば，ある文に対する判定結果が本当は 0 であるにもかかわらず誤って 1 と判定される可能性がある．よって，FV モデルを使って計測した P や R を解釈する際には誤判定の可能性を考慮する必要がある．

そこで本実験では，真の判定結果 $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ を入力信号，$\mathbf{y}$ を出力信号とする 2-元非対称通信路（BAC; Binary Asymmetric Channel）として FV モデルの入出力関係をモデル化し，Bayes 推定によって真の P や R を推定することを考える．図 3 に推定方法のグラフィカルモデルと BAC の構造を示す．本モデルにおいて y_i は観測変数であり，それ以外はすべて潜在変数として扱われる．グラフィカルモデルの各有向辺はその始点をパラメータ，終点を確率変数とする条件付き確率分布に対応している．辺のラベルには設定する確率分布を示している．k は $k = \sum_i z_i$ で定義され，$\mathbf{z}$ から決定的に定まる変数である．BAC の図の各辺は遷移確率を表しており，tpr と fpr という 2 つのパラメータを持つ．入力値 z_i が 1 のときは確率 tpr，0 のときは確率 fpr をパラメータとする Bernoulli 分布に従って出力値 y_i が決定されることを意味する．

ここに 2 つの仮定を置いている．1 つ目の仮定は，各文の真偽値 z_i は母数 θ の共通の Bernoulli 分布に従い，互いに独立であることである．厳密には各文の真偽には相関が生じうる．例えば，意味的に類似する 2 つの文はその真偽も類似する傾向があると考えられる．そのため，本来はこれらの関係性も考慮できれば望ましい．しかし，意味的類似性を正確に定量化することは困難である．その客観的な判

断には文脈の考慮が必要となるため [28]，統一的な評価は難しいと考えられる．そこで本実験では事後分布の解析的な導出を優先し，真である確率を表す単一のパラメータ θ を導入して単純化する．2 つ目の仮定は，各観測値 y_i の生成はその真値 z_i および誤り率 tpr, fpr のみに依存することである．FV モデルによる判定は前処理として脱文脈化された各文に対して独立に行われる．したがって，y_i が他の事例 $y_{\neq i}$ や $z_{\neq i}$ から条件付き独立であるという仮定は妥当であるといえる．

さらに，各パラメータの事前分布として Beta 分布を仮定する．これは Beta 分布が Bernoulli 分布の共役事前分布であり，事後分布の解析的な導出が可能になるためである．θ の事前分布にはパラメータ $(1, 1)$ の Beta 分布を仮定する．この分布は $[0, 1]$ 上の一様分布と等価であり，θ に対する事前知識を何も仮定しないことを意味する．また，tpr と fpr の事前分布にはそれぞれ，パラメータを $(\mathrm{TP}+1, \mathrm{FN}+1)$，$(\mathrm{FP}+1, \mathrm{TN}+1)$ とする Beta 分布を設定する．ここで，$M = (\mathrm{TP}, \mathrm{FN}, \mathrm{FP}, \mathrm{TN})$ は FV モデルの選別時に得られた混同行列である．これらの事前分布は，パラメータ $(1, 1)$ の Beta 分布を事前分布とし，二項分布に基づく Bayes 更新を行った場合に得られる事後分布と等価である．

1 と判定した文の数を $m = \sum_i y_i$ とすると，事後分布 $\mathbf{P}(k \mid \mathbf{y}, M)$ のカーネルは次のように書ける．

$$\begin{aligned}\mathbf{P}(k \mid \mathbf{y}, M) \propto B(k+1, n-k+1) \sum\nolimits_{k_1=L}^{U} \binom{m}{k_1}\binom{n-m}{k-k_1} \\ \times B(k_1 + \mathrm{TP} + 1, k - k_1 + \mathrm{FN} + 1) \\ \times B(m - k_1 + \mathrm{FP} + 1, n - k - m + k_1 + \mathrm{TN} + 1).\end{aligned}$$

ここで，B, L, U はそれぞれ以下のように定義される．

$$B(a, b) = \int_0^1 x^{a-1}(1-x)^{b-1}\mathrm{d}x, \quad L = \max(0, k-n+m), \quad U = \min(k, m).$$

導出の詳細は付録 B に記載している．スコアの分布は確率変数 $r = k/n$ の分布として定義される．n は与えられた文の列に対して定数であるため，k と r は一対一に対応する．したがって，とりうるすべての値 $k \in \{0, 1, \ldots, n\}$ に対して，確率質量関数は以下のように対応する．

$$\mathbf{P}\left(r = \frac{k}{n} \mid \mathbf{y}, M\right) = \mathbf{P}(k \mid \mathbf{y}, M).$$

すなわち，確率変数の定義域を $[0, 1]$ 上の離散値に線形変換した分布になる．

上記は 1 つの構造標準化および脱文脈化の結果に対する事後分布である．本実験では LLM の確率的挙動を考慮して各操作を複数回試行を繰り返している．それぞれの試行で得られた事後分布を同一の重みで混合して最終的な 1 つの分布とする．そして，この混合分布の期待値を評価指標の補正値として報告する．また，真のスコアが含まれる確率が 95% 以上となる信用区間として等裾事後信用区間を算出し，併せて報告する．

4.4 RQ2 の評価

標準構造の SRS に記述されるべき内容が標準化 SRS に含まれているかを確認する．そのために，記述が含まれるべき項目のチェックリストを節ごとに作成する．チェックリストに基づいて標準化 SRS を目視で確認し，チェックの網羅率によって標準構造への忠実性を評価する．

チェックリストの作成には Boyachunk らの定式化 [29] を参考にする．彼らは ISO 29148 を参考にして SRS が含むべき項目を集合として定式化している．例えば，標準構造の 3.3 節 “Usability requirements” に対応する S_12 については $\mathrm{S\_12} = \{\mathrm{ubr}, \mathrm{mec}, \mathrm{mefc}, \mathrm{msc}\}$ となる．集合の要素が項目であり，ISO 29148 での記述の定義を簡略化した表現である．S_12 では以下のように定義される．

$$
\begin{aligned}
\mathrm{ubr} &= \text{“usability requirements”},\\
\mathrm{mec} &= \text{“measurable criteria of effectiveness in specific use contexts”},\\
\mathrm{mefc} &= \text{“measurable criteria of efficiency in specific use contexts”},\\
\mathrm{msc} &= \text{“measurable criteria of satisfaction in specific use contexts”}.
\end{aligned}
$$

節との対応関係を明確にするため，第 X 節に対応する項目集合を E_X と表記することにする．例えば，$E_{3.3} = \mathrm{S\_12} = \{\mathrm{ubr}, \mathrm{mec}, \mathrm{mefc}, \mathrm{msc}\}$ である．チェックリストの全項目は $\cup_X E_X$ となる．

チェックリストの作成に際して 2 つの修正を行った．1 つは項目の追加と削除である．Boyachunk らの定式化には標準構造の 1.4 節と 2 節，5.2 節に対応する集合が含まれない．そこで，ISO 29148 を参考に $E_{1.4} = \{\mathrm{dbd}\}$, $E_2 = \{\mathrm{lrd}, \mathrm{tndo}, \mathrm{srof}\}$, $E_{5.2} = \{\mathrm{aa}\}$ と定義する．各項目は以下の通りである．

$$
\begin{aligned}
\mathrm{dbd} &= \text{“definitions for any words or phrases that have meaning beyond dictionaries”},\\
\mathrm{lrd} &= \text{“list of referred documents”},\\
\mathrm{tndo} &= \text{“document title, report number, date and publishing organization”},\\
\mathrm{srof} &= \text{“sources which the references can be obtained from”},\\
\mathrm{aa} &= \text{“acronyms and abbreviations used in the documents”}.
\end{aligned}
$$

一方で，1.2 節に含まれる項目である cssh（= “consistency with the same statements in high-level specifications”）は除外する．これは上位レベルの仕様書にある記述と一貫する記述であることを要求する項目であるが，標準化 SRS のみを対象とする本評価ではこれらを確認できないためである．チェックリストのすべての項目は付録 C に記載している．

もう 1 つの修正は項目の階層化である．図 4 に定義した階層構造を木構造で表している．いくつかの項目の間には包含関係が認められる．例えば，$\mathrm{ubr}, \mathrm{mec} \in E_{3.3}$ について ubr は mec を含みうる．ubr の具体が mec であり，mec の記述がある場合は ubr の記述があるとみなせる．また逆に，ubr の記述がないならば mec の記述もそれらの定義から存在しえない．つまり，包含関係にある項目間では一方

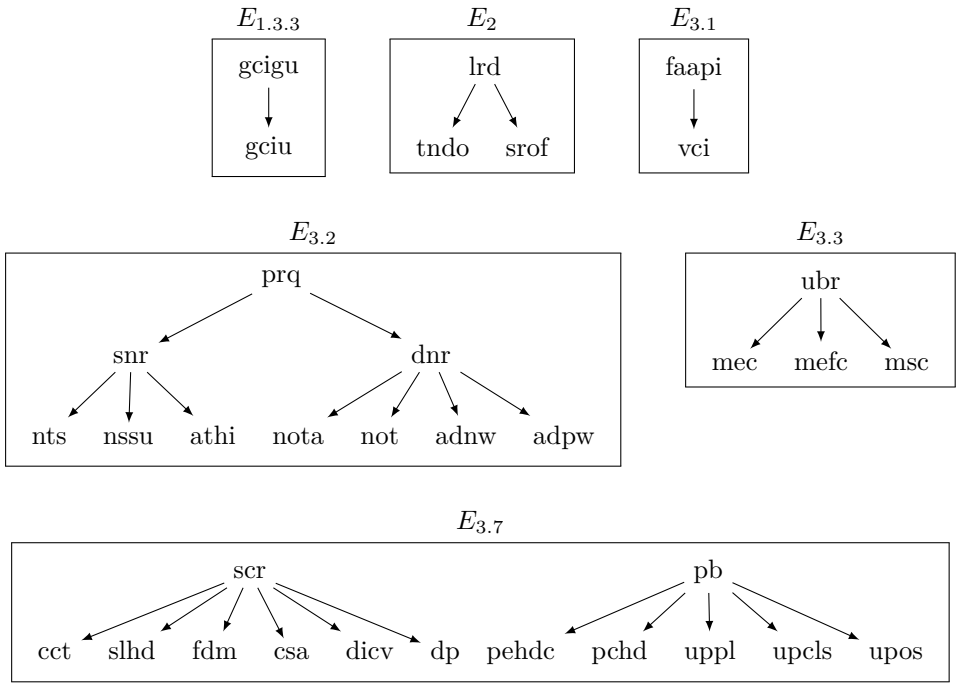

図 4　標準化 SRS の記述に含まれるべき項目間の包含関係

の存在が他方の存在に影響を与える．この関係によって階層化を行った．チェックが終了した後に項目間の階層関係に基づいて後述する後処理を行い，最終的な集計に用いる．

チェックとその集計方法について説明する．各項目 $e \in E_X$ について，対象の標準化 SRS の第 X 節の記述 d_X を確認して以下の 4 つのラベルのいずれかを付与する．

- `EXIST`　e に関して記述があり，ISO 29148 で説明される e の記述の定義に則している．
- `WRONG`　e に関して記述があり，ISO 29148 で説明される e の記述の定義に則していない．
- `N/A`　e に関して `[N/A]` の記述がある，あるいは e の情報がないことの記述がある．
- `NULL`　e に関する記述がない，あるいは d_X が `[NULL]` かつ第 X 節の子の節がない．

`WRONG` に該当する例として，ps $\in E_{1.1}$ について対象の標準化 SRS が 1.1 節 “Purpose” として SRS 自体の目的のみを記述している場合が該当する．ISO 29148 では “Delineate the purpose of the software to be specified.” と定義され，SRS ではなく対象ソフトウェアの目的についての記述が要求されているためである．`[N/A]` の記述がなく `N/A` に該当する記述例としては “There is no information provided” などがあげられる．ラベル付けは著者および指導教員の 2 名が共同で実施し，合意形成した上でチェックリストを埋める．完成したチェックリストについて，前述の項目間の階層構造に基づいて以下で示す後処理を適用する．

規則 1：e のラベルが `EXIST` の場合，その親項目のラベルも `EXIST` に書き換える．

規則 2：e のラベルが `N/A` の場合，

（a）その親項目のラベルが `EXIST` 以外の場合は `N/A` に書き換える．

（b）そのすべての子項目のラベルが `NULL` の場合はそれらを `N/A` に書き換える．

規則 1 は具体項目に関する記述の存在が対応する抽象項目に関する記述の存在を意味することを反映している．規則 2 は情報が存在しない旨の記述がその具体項目に該当する情報も存在しないことを意味する可能性を反映している．ただし，すべての具体項目について記述が存在しない場合に限る．上記の規則を上から順にラベルの更新が起きなくなるまで繰り返していく．そして，以下の量を報告する．

$$\text{網羅率} = \frac{\texttt{EXIST}\text{と}\ \texttt{N/A}\ \text{の項目数}}{\text{全項目数}}, \quad \text{有効率} = \frac{\texttt{EXIST}\ \text{の項目数}}{\texttt{EXIST}\text{と}\ \texttt{WRONG}\ \text{の項目数}}.$$

網羅率は標準化 SRS に記述されるべき項目のうちどの程度が正しく記述されたかを表す．有効率は記述されていた項目のうち ISO 29148 の指示に正しく従っていた割合を表す．ただし，ここでいう正しさとは ISO 29148 への忠実さのみを指す．記述の内容が元の SRS の意図するところを確実に反映しているかは考慮していないことに注意する．

RQ1 では 1 つの SRS に対して複数の標準化 SRS を生成したが，RQ2 ではその中から適合率が最も高い標準化 SRS のみを評価対象とする．RQ2 の目的は標準化 SRS が 標準構造にどの程度忠実であるかを調査することである．そのため，標準化 SRS の内容が元の SRS と意味的に整合しているかどうかは評価対象とはせず，評価の前提条件とする．ただし，標準化 SRS がある項目に関する記述を含んでいたとしても，その内容が元の SRS に含まれていない，あるいは元の記述と異なっている可能性がある．このようなケースを低減するため，標準化 SRS に含まれる情報が元の SRS にどの程度含まれているかを表す指標である適合率が最も高い SRS を，RQ2 における評価対象として選択する．

5 実験結果

5.1 RQ1 の結果

図 5 から図 11 が RQ1 の実験結果である．各図における左図は適合率 P について，右図は再現率 R について示している．構造標準化に用いた各 LLM ごとに縦方向でグループ化して配置しており，各 LLM について $\mathtt{P}_{\mathtt{all}}$ と $\mathtt{P}_{\mathtt{each}}$ での結果をそれぞれ上方，下方に示している．バツで記した点が計測値であり，マルで示した点は補正値である．エラーバーはスコアの 95%-信用区間を表している．エラーバーの端点を含む各点には対応する数値を併記しており，太字は補正値である．

はじめに P に着目する．(08) と gemini-2.5-pro の場合を除き，$\mathtt{P}_{\mathtt{all}}$ は一貫して $\mathtt{P}_{\mathtt{each}}$ を上回る P を示した．その絶対値も大きく，$\mathtt{P}_{\mathtt{all}}$ についてすべての SRS と LLM で平均した計測値は 0.76，補正値は 0.90 であった．SRS ごとの補正値は (10) のときに最高値 0.98，(09) のときに最低値 0.79 であった．一方，$\mathtt{P}_{\mathtt{each}}$ では $\mathtt{P}_{\mathtt{all}}$ と比較して値が小さく，すべての SRS と LLM で平均した計測値は 0.60，補正値は 0.68 であった．gpt-5 が特に低く，すべての SRS において計測値と補正値がともに 0.50 を下回った．(09) で最低値をとり，計測値は 0.26，補正値は 0.05 であった．加えて，他の LLM とプロンプトの組合せと比較して (01) と (08)，(09)，(10) では信用区間が重ならずに下方に位置しており，有意に劣る可能性が高い．これらの結果から，元の SRS を基準とした場合により正確な SRS を生成できているのは $\mathtt{P}_{\mathtt{all}}$ であるといえる．LLM 間で比較すると GPT では gpt-4o が gpt-5 を，いずれのプロンプトでも一貫して上回った．Gemini では gemini-2.5-flash が優位な場合と gemini-2.5-pro が優位な場合が混在していた．また，$\mathtt{P}_{\mathtt{all}}$ と $\mathtt{P}_{\mathtt{each}}$ での，平均の補正値の差は gpt-4o で 0.18，gpt-5 で 0.52 であったのに対し，gemini-2.5-pro では 0.14，gemini-2.5-flash では 0.07 であった．GPT では $\mathtt{P}_{\mathtt{all}}$ と $\mathtt{P}_{\mathtt{each}}$ の性能差が Gemini と比較して大きく，プロンプト選択の影響を受けやすい傾向があるといえる．GPT と Gemini とで比較すると $\mathtt{P}_{\mathtt{all}}$ では両者は概ね同程度であったのに対し，$\mathtt{P}_{\mathtt{each}}$ では Gemini が高くなった．

次に R に着目する．R は P とは対照的な傾向を示しており，すべての SRS および LLM の組合せ 28 件のうちの 20 件において，$\mathtt{P}_{\mathtt{all}}$ よりも $\mathtt{P}_{\mathtt{each}}$ の方が高い計測値と補正値を示した．元の SRS に含まれる情報をより多く保持した標準化 SRS は $\mathtt{P}_{\mathtt{all}}$ よりも $\mathtt{P}_{\mathtt{each}}$ によって生成されたことが示唆される．ただし，その絶対値は P のそれと比較すると大きいとはいえない．$\mathtt{P}_{\mathtt{each}}$ についてすべての SRS と LLM で平均した計測値は 0.59，補正値は 0.66 であった．また，gpt-5 および gemini-2.5-flash では (07g) 以外の SRS において補正値が 0.80 を超えている一方で，gpt-4o ではすべての SRS において 0.50 未満に留まっていた．(07g) 以外のいずれの SRS においても gpt-5 と gemini-2.5-flash の信用区間は gpt-4o のそれと重ならずに上方に存在し，有意に優れる可能性が高い．LLM 間の比較では GPT では gpt-4o よりも gpt-5 が一貫して高く，Gemini では gemini-2.5-flash が

gemini-2.5-pro を（07p）を除いて上回った．

RQ1 への回答

P_{all} で生成された標準化 SRS は LLM を問わず高い適合率を示し，元の SRS の内容を正確に反映できていると評価できる．一方で，P_{each} と gpt-5，または gemini-2.5-flash で生成された標準化 SRS は高い再現率を示し，元の SRS に含まれる情報を P_{all} より多く保持していると評価できる．しかし，両者を同時に高水準で満たすのは難しく，適合率と再現率の間にはトレードオフが存在することが確認された．その中でも，gemini-2.5-flash と P_{all} の組合せは，適合率を高く維持しつつ再現率も相対的に高く，両指標のバランスが最も良い条件であると評価できる．

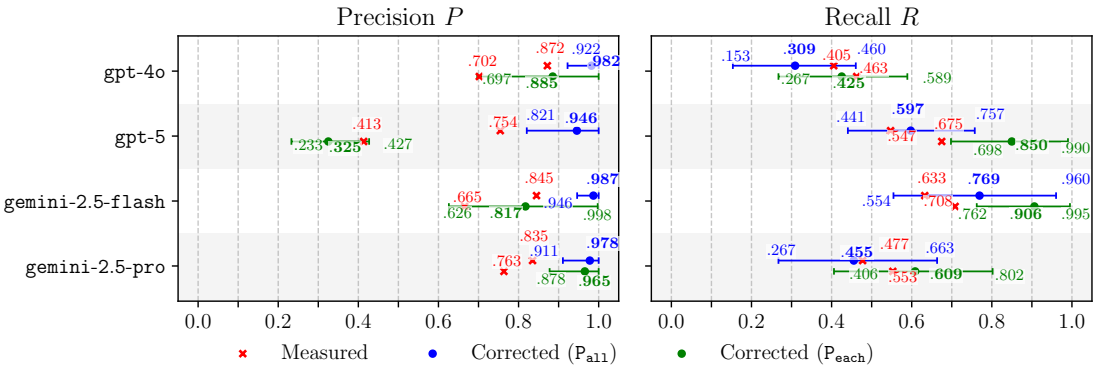

図 5　SRS（01）に対する適合率と再現率

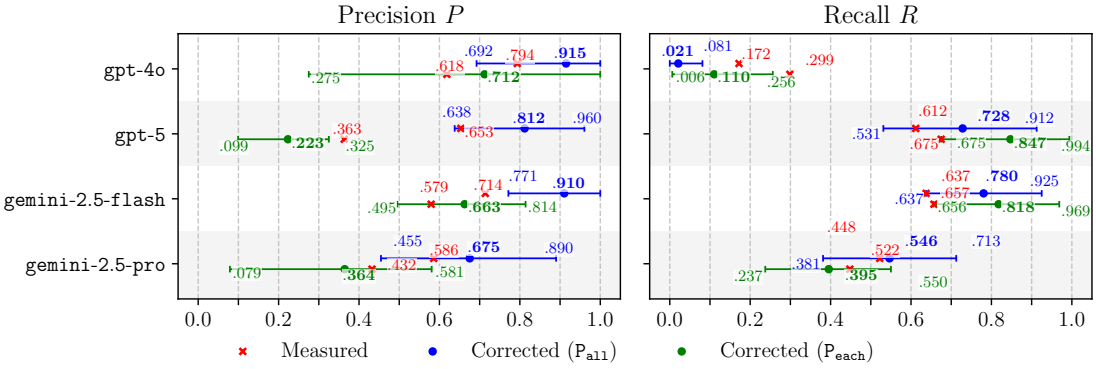

図 6　SRS（03）対する適合率と再現率

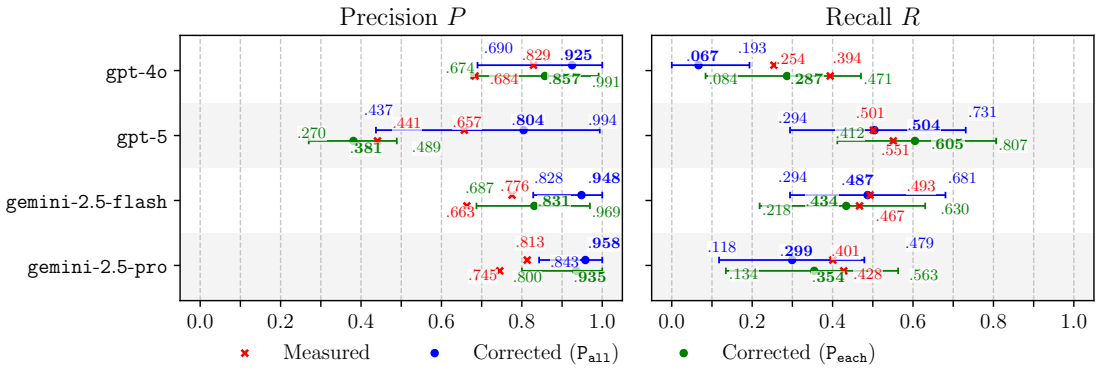

図 7　SRS（07g）対する適合率と再現率

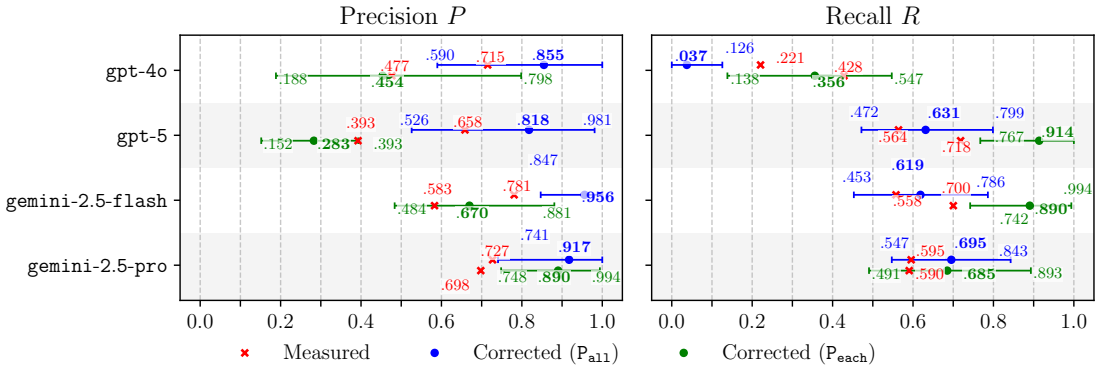

図 8　SRS（07p）対する適合率と再現率

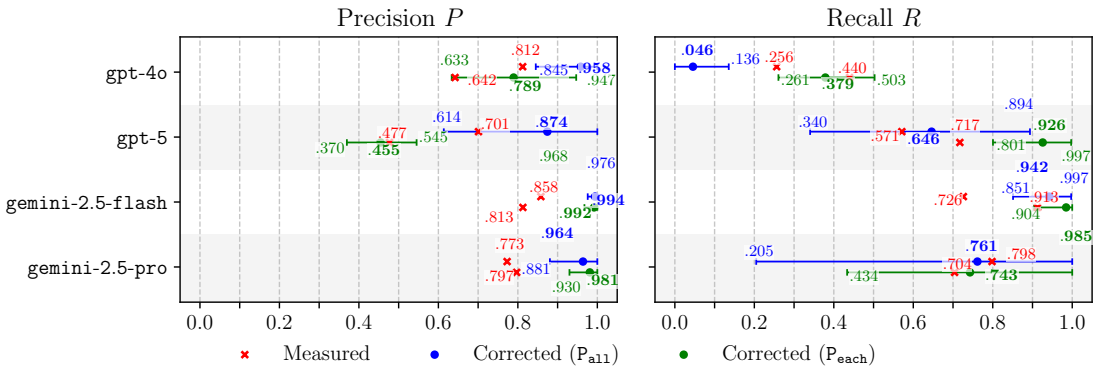

図 9　SRS（08）対する適合率と再現率

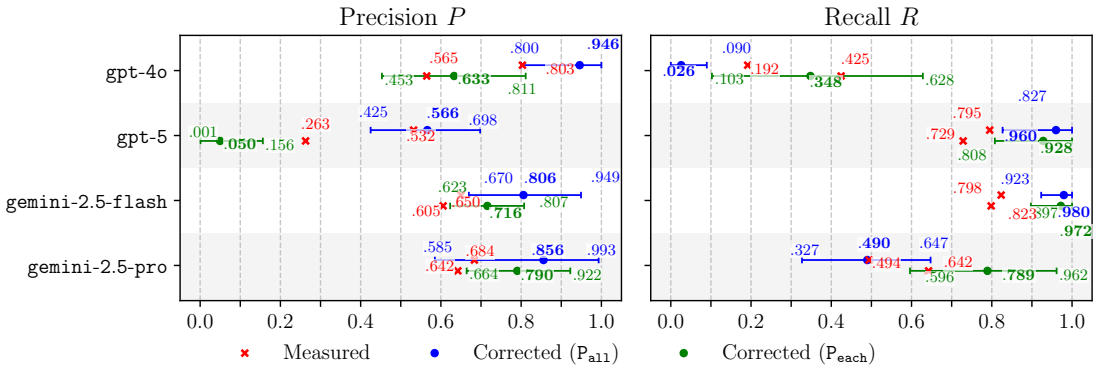

図 10　SRS（09）対する適合率と再現率

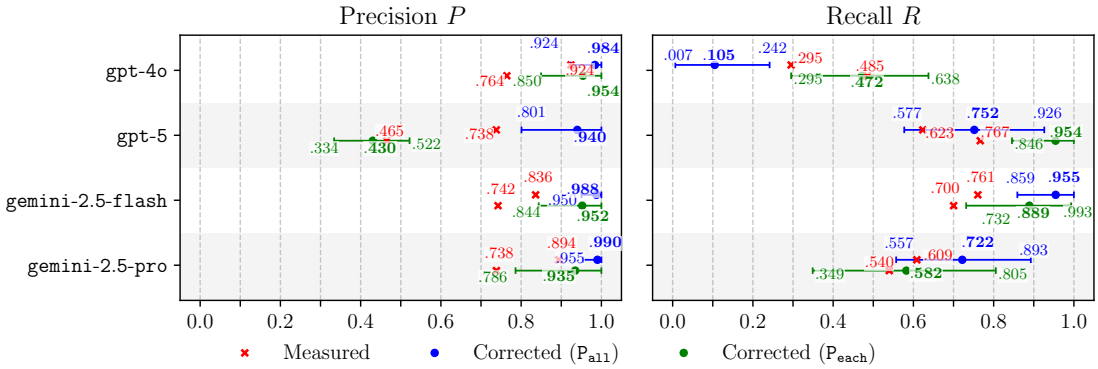

図 11　SRS（10）対する適合率と再現率

5.2　RQ2 の結果

表 3 から表 9 に RQ2 に関する実験結果を示す．各 LLM について P_{all} および P_{each} を用いた場合の網羅率を示している．なお，かっこ内の数値は有効率を表す．

まずは網羅率に着目する．LLM の種類にかかわらず，網羅率は一貫して P_{all} よりも P_{each} の方が高い．特に GPT では P_{all} から P_{each} への変更により網羅率の向上が確認された．gpt-4o では平均して 1.82 倍に，gpt-5 では平均して 1.41 倍に増加している．また，P_{each} における網羅率の絶対値も高く，gpt-4o では平均して 0.82，gpt-5 では 0.94 となった．特に gpt-5 と P_{each} の組合せでは（07g）を除くすべての SRS において網羅率が 0.90 を超えており，最高値は（09）での 0.98 であった．一方，Gemini においても P_{all} から P_{each} への改善が確認できるものの，その増加率は GPT と比較すると小さい．gemini-2.5-flash では平均して 1.18 倍，gemini-2.5-pro では 1.25 倍の増加にとどまっている．さらに，P_{each} における網羅率の絶対値も GPT と比較すると低い．gemini-2.5-flash では平均値は 0.67，最高値は 0.73 であり，gemini-2.5-pro では平均値は 0.64，最高値は 0.76 である．

次に有効率に着目する．有効率はいずれの SRS および LLM，プロンプトの組合せにおいても 0.90 以上を示しており，全 56 件のうち 13 件では 1.00 となった．また，(10) を除くすべての SRS において $\mathtt{P}_{\mathtt{all}}$ よりも $\mathtt{P}_{\mathtt{each}}$ の方が高い値を示した．ただし，LLM やプロンプトの違いによる組合せ間で大きな差異は認められなかった．このことから，生成された標準化 SRS は全体として高い有効性を有しており，LLM やプロンプトの違いに依存しない安定した傾向が確認された．すなわち，標準化 SRS に記述された各項目は概ね ISO 29148 の指示に沿って記述されていると評価できる．

RQ2 への回答

$\mathtt{P}_{\mathtt{each}}$ により生成された標準化 SRS の網羅率は，GPT では SRS 全体で平均して 0.88，Gemini では 0.66 を示した．一方，$\mathtt{P}_{\mathtt{all}}$ により生成された標準化 SRS はいずれの LLM においても $\mathtt{P}_{\mathtt{each}}$ を用いた場合よりも一貫して低い網羅率を示した．これらの結果から，GPT と $\mathtt{P}_{\mathtt{each}}$ を用いた場合には ISO 29148 で要求される記述項目を概ね満たす標準化 SRS が生成可能であると評価できる．特に，`gpt-5` と $\mathtt{P}_{\mathtt{each}}$ の組合せではほとんどの SRS において網羅率が 0.90 を超え，本実験の条件で最も高い網羅率を示した．有効率はいずれの LLM およびプロンプトにおいても 0.90 以上の高い値を維持しており，項目が生成された場合にはその内容が概ね ISO 29148 の指示に沿って記述されていることが確認された．

表 3　SRS（01）に対する網羅率と有効率

LLM	$\mathtt{P}_{\mathtt{all}}$	$\mathtt{P}_{\mathtt{each}}$
`gpt-4o`	.511 (.944)	.856 (1.000)
`gpt-5`	.656 (.977)	.911 (.984)
`gemini-2.5-flash`	.689 (.976)	.711 (.984)
`gemini-2.5-pro`	.600 (.977)	.756 (1.000)

表 4　SRS（03）に対する網羅率と有効率

LLM	$\mathtt{P}_{\mathtt{all}}$	$\mathtt{P}_{\mathtt{each}}$
`gpt-4o`	.444 (.941)	.865 (.963)
`gpt-5`	.733 (.971)	.933 (1.000)
`gemini-2.5-flash`	.578 (.926)	.611 (1.000)
`gemini-2.5-pro`	.622 (.947)	.667 (.971)

表 5　SRS（07g）に対する網羅率と有効率

LLM	$\mathtt{P}_{\mathtt{all}}$	$\mathtt{P}_{\mathtt{each}}$
`gpt-4o`	.389 (.938)	.878 (.966)
`gpt-5`	.678 (.941)	.878 (1.000)
`gemini-2.5-flash`	.456 (.923)	.556 (1.000)
`gemini-2.5-pro`	.389 (.957)	.700 (1.000)

表 6　SRS（07p）に対する網羅率と有効率

LLM	$\mathtt{P}_{\mathtt{all}}$	$\mathtt{P}_{\mathtt{each}}$
`gpt-4o`	.411 (.895)	.844 (.939)
`gpt-5`	.600 (.914)	.967 (.981)
`gemini-2.5-flash`	.544 (.926)	.733 (.977)
`gemini-2.5-pro`	.400 (.920)	.556 (.971)

表 7　SRS（08）に対する網羅率と有効率

LLM	P_{all}	P_{each}
gpt-4o	.544 (.964)	.744 (.949)
gpt-5	.611 (.974)	.922 (1.000)
gemini-2.5-flash	.544 (.946)	.689 (.926)
gemini-2.5-pro	.567 (.974)	.622 (.977)

表 8　SRS（09）に対する網羅率と有効率

LLM	P_{all}	P_{each}
gpt-4o	.371 (.913)	.753 (.900)
gpt-5	.567 (.978)	.978 (1.000)
gemini-2.5-flash	.500 (.913)	.667 (.980)
gemini-2.5-pro	.456 (.974)	.600 (.977)

表 9　SRS（10）に対する網羅率と有効率

LLM	P_{all}	P_{each}
gpt-4o	.478 (1.000)	.778 (1.000)
gpt-5	.800 (.977)	.967 (1.000)
gemini-2.5-flash	.689 (1.000)	.733 (.980)
gemini-2.5-pro	.567 (.977)	.589 (.981)

5.3　考察

RQ1 および RQ2 の実験結果が得られた要因について考察する．本節では LLM とプロンプトの違いによる標準化 SRS の特徴と，入力となる元の SRS の特徴の 2 点に着目する．

標準化 SRS の特徴による影響

表 10 は LLM とプロンプトの各組合せで生成された標準化 SRS について，元の SRS を基準とした語数と文数の比率を示している．各 SRS ごとに比率を計算して平均をとった．表 10 より，P_{all} は多くの LLM において語数比率および文数比率が 1 未満または 1 付近に留まっており，元の SRS と比較して出力が抑制される傾向が確認できる．このことから，P_{all} による標準化 SRS は冗長な補足を加え

表 10　元の SRS を基準とした標準化 SRS の語数と文数の比率

LLM	プロンプト	語数比率	文数比率
gpt-4o	P_{all}	0.285	0.320
	P_{each}	0.856	0.871
gpt-5	P_{all}	0.767	1.317
	P_{each}	3.256	4.742
gemini-2.5-flash	P_{all}	0.997	1.122
	P_{each}	1.629	1.930
gemini-2.5-pro	P_{all}	0.635	0.737
	P_{each}	1.061	1.256

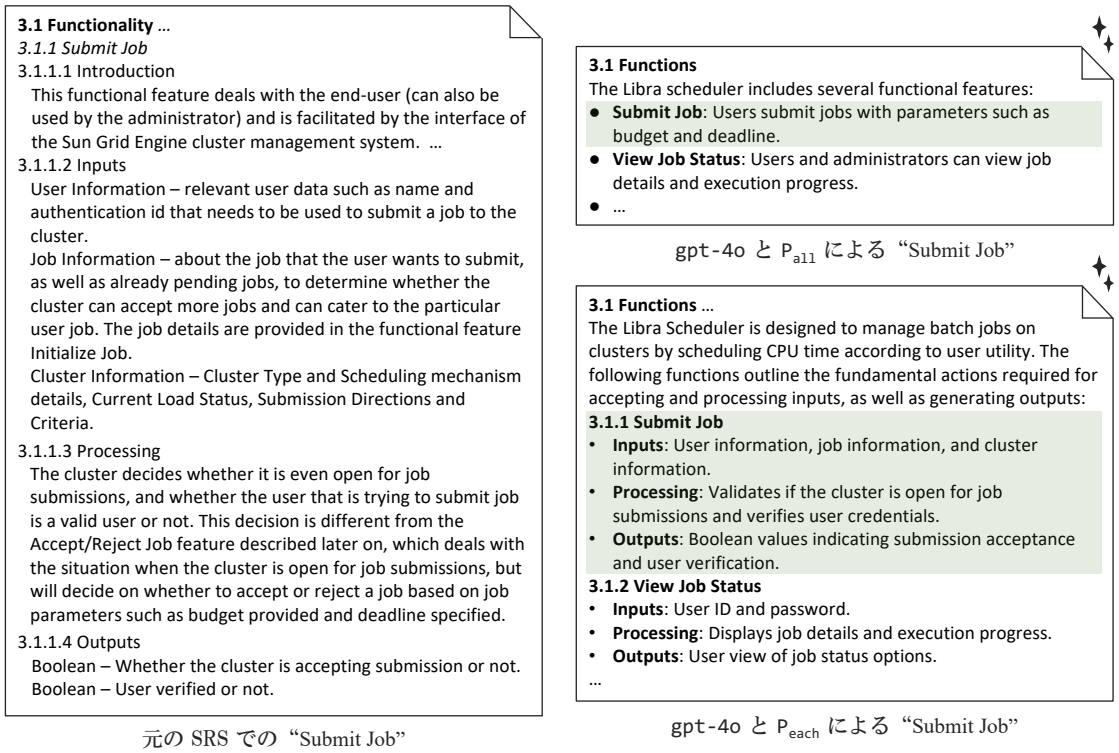

図 12　gpt-4o による標準化 SRS の 3.1 節と元の SRS で対応する記述

ずに元の内容を簡潔に反映する傾向があるといえる．一方で，P_{each} は gpt-4o 以外のすべての LLM において語数比率および文数比率が 1 を上回っており，標準化 SRS の分量が大きく増加する傾向が認められる．特に gpt-5 は他の LLM と比較して顕著であり，語数および文数はいずれも平均して 3 倍以上に増加している．このような出力の増加は元の SRS に含まれる情報の脱落を抑制する効果を持つ一方で，不要な言い換えや重複を生じさせる可能性がある．これらの生成挙動は RQ1 の実験において P_{all} が高い適合率と低い再現率を示し，P_{each} がその逆の傾向を示したという結果と整合的である．2 つの指標間のトレードオフが出力する情報量の制御に依存したと考えられる．また，両指標をともに高水準で達成する必要条件が元の SRS と同等の記述量を持つ標準化 SRS の生成であることも示唆される．実際，RQ1 の実験で適合率と再現率が同時に高水準であった gemini-2.5-flash と P_{all} の組合せは語数と文数の比率がともに最も 1 に近かった．加えて，P_{all} から P_{each} への記述量の増加が RQ2 の実験で観測された網羅率の改善に寄与した可能性も考えられる．標準化 SRS の記述量が増加することで ISO 29148 で要求される各項目に対応する記述が生成される確率が高まり，網羅率の向上につながったと考えられるためである．

プロンプトによって適合率と再現率の差が大きかった GPT による生成例を取り上げる．図 12 は gpt-4o と P_{all}，P_{each} を用いて生成された標準化 SRS とその元となった SRS（01）の一部抜粋であ

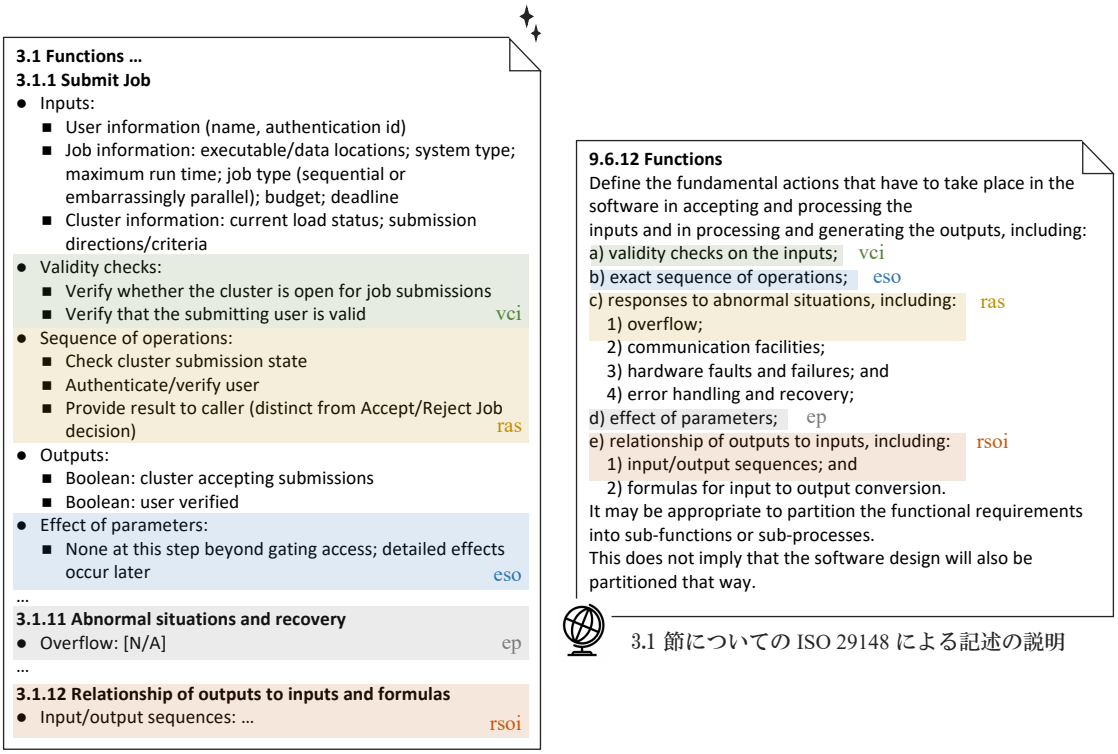

図 13　gpt-5 と P_{each} による標準化 SRS の 3.1 節と ISO 29148 による記述の説明

る．標準構造における 3.1 節 “Functions” 下に配置された，対象ソフトウェアの 1 つの機能である “Submit Job” についての記述を抜き出している．図 12 より，P_{all} および P_{each} による標準化 SRS は元の SRS と比べて記述量が明らかに少ないとわかる．元の SRS では “Introduction” や “Inputs” などといった構造の下で，機能の背景や前提条件，詳細な入出力情報，処理の内容が具体的に記述されている．その一方で，標準化 SRS ではこれらの内容が要約されて主要な情報のみが簡潔に整理されている．P_{each} では構造自体は保持されているものの各記述はやはり簡略化されている．これらの gpt-4o による出力の簡潔化や抽象化といった特徴が，表 10 で示された記述量の抑制や RQ1 で観測された高い適合率および低い再現率に影響したと考えられる．

図 13 は gpt-5 と P_{each} を用いて生成された標準化 SRS と ISO 29148 による記述の説明の一部抜粋である．標準化 SRS については “Submit Job” を含む 3.1 節 “Functions” の記述のうち一部を省略して掲載し，ISO 29148 については 3.1 節に関する説明を抜き出している．また，RQ2 の実験で用いたチェックリストの各項目に対応する箇所を色分けして示している．標準化 SRS ではチェックの根拠となった記述を，ISO 29148 では各項目の説明に該当する記述をそれぞれ網掛けしている．図 12 に示した gpt-4o と P_{each} による記述と比較すると，gpt-5 による標準化 SRS は元の SRS に含まれる内容をより多く保持できていることが確認できる．加えて，その記述は ISO 29148 に基づいて構造化されて

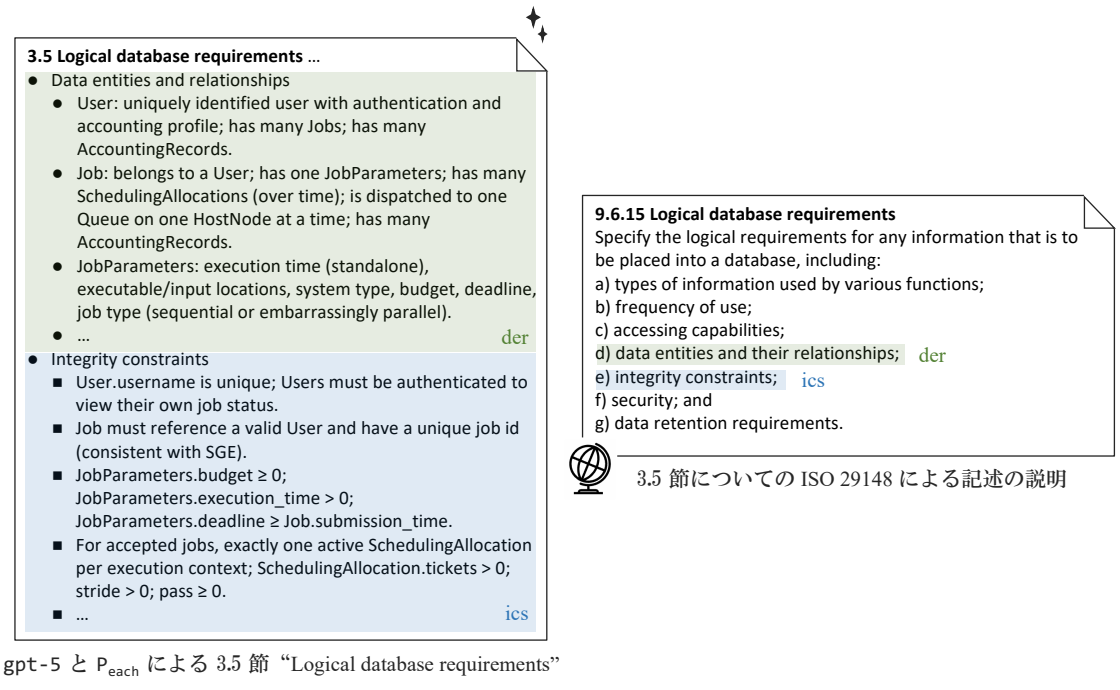

図 14　gpt-5 と $\texttt{P}_{\texttt{each}}$ による標準化 SRS の 3.5 節と ISO 29148 による記述の説明

いて，元の SRS では明示的に切り分けられていない入力検証（vci）や動作列（eso）についても対応する形式で整理されている．図 14 は同じ標準化 SRS の 5.2 節 “Logical database requirements” の一部抜粋であり，ここでも同様の構造化がみられ，データ設計や制約の条件式を含んだ詳細な記述になっている．しかし，これらのような記述は元の SRS には確認できなかった．このように，gpt-5 は元の SRS で明示されない要素について，ISO 29148 に基づく観点から補完的に記述する傾向がみられた．

これらの例や他の SRS に対する RQ2 での観察から，LLM による構造化の出力傾向について次の洞察が得られる．$\texttt{P}_{\texttt{all}}$ のようにすべての情報を一度に生成する方式では内容保持に優れる一方で，項目との対応づけが困難な情報は省略されやすい．$\texttt{P}_{\texttt{each}}$ のように部分的に生成する方式では項目に沿った構造化が可能となるが，それに伴って補完や詳細化が加わる傾向がある．こうした特徴が RQ1 における適合率と再現率の差や，RQ2 における $\texttt{P}_{\texttt{each}}$ の高い網羅率に反映されたと考えられる．

元の SRS の特徴による影響

元の SRS に ISO 29148 の項目名と一致する記述要素が存在する場合，その内容が ISO 29148 の定義と厳密に合致するかどうかを十分に吟味せずに，元の記述をそのまま標準 SRS の記述に採用してしまうケースが複数確認された．例えば，元の SRS に “Purpose” という節が設けられている場合，その内容が ISO 29148 で求められるソフトウェアの目的ではなく単に SRS 自体の執筆目的を説明する記述であったとしても，標準構造の 1.1 節 “Purpose” としてそのまま配置される事例が $\texttt{P}_{\texttt{all}}$ や $\texttt{P}_{\texttt{each}}$，

LLM を問わず散見された．記述の意味内容よりも見出しなどの表層的な手がかりに強く影響されうるというこの傾向は RQ2 における有効性の評価を下げる一因となった．

対象となる SRS のドメインも実験結果に影響したと考えられる．(07g) は Web サイトに関する SRS である．その性質上，入出力に伴うデータ処理や手順の記述よりも提供するコンテンツの内容やサイト構成といった静的情報の定義に記述の主眼が置かれている．ISO 29148 は主にソフトウェアの機能的側面の記述を求めているため，このようなプロジェクトの SRS とは親和性が低く，標準化 SRS の網羅率低下に影響を及ぼしたと推察される．実際に RQ2 の実験においても，3.1 節 “Functions” の記述は ISO 29148 が要求する項目に沿って構造化されていたものの，`[N/A]` とともに明示されなかった項目については faapi（= “basic actions that must occur in the software when processing and generating outputs”）を除いてすべて `NULL` と判定された．

元の SRS に ISO 29148 が要求する記述項目に該当する情報が存在しないことも RQ1 の適合率に影響を及ぼしうる．$\texttt{P}_{\texttt{each}}$ によって生成された標準化 SRS は 3.5 節や 4 節をはじめとして，すべての節に記述を含んでいる．一方，本実験で対象とした SRS の元文書には 4 節 “Verification” に相当する記述が含まれていない．(07g) や (08)，(10) ではテストの実施が示唆されてはいるものの具体的な手順や受け入れ基準までは明記されていない．また，3.5 節 “Logical database requirements” についても関連する節が明示的に存在するのは (08) のみである．(01)，(03)，(7p) にはそもそも “database” という単語すら存在せず，関連記述も見当たらない．(07g) と (10) ではデータベースへの言及はあるものの「利用可能である」という記述にとどまっている．(09) は該当情報が記述されてはいるがその内容は分散している．標準化 SRS におけるこれらの記述に対する含意判定には元の SRS の記述内容を超えた推測や補完が必要となり，適合率の観点では評価されない．このような元文書における情報の欠落が標準化 SRS の適合率を低下させる一因となったと考えられる．

この傾向は各節における適合率の分析からも確認できる．図 15 に LLM とプロンプトの各組合せに対する標準構造の各節における適合率を示す．各節の適合率はすべての SRS での，計測値の平均値を表している．一部の 4 節に施された灰色の網掛けはその内容が `[N/A]` のみであったケースを示しており，これらは適合率の計算対象から除外している．いずれの組合せにおいても 4 節で適合率が計算可能な場合には他の節に比べてその値が最も低いと確認できる．また，3.5 節においても $\texttt{P}_{\texttt{each}}$ では比較的低い適合率が観察される．一方で，`gpt-4o` と $\texttt{P}_{\texttt{all}}$ では 3.5 節の適合率が相対的に高い．これは，データベースへの言及が存在しない SRS に対しては `[N/A]` を出力し，言及がある場合には該当箇所を簡潔に抽出していたためである．元の SRS に該当情報が含まれない場合には，`gpt-4o` と $\texttt{P}_{\texttt{all}}$ の組合せが $\texttt{P}_{\texttt{each}}$ とは対照的に過剰な補完を避ける点で，適合率の向上に有効に機能したといえる．

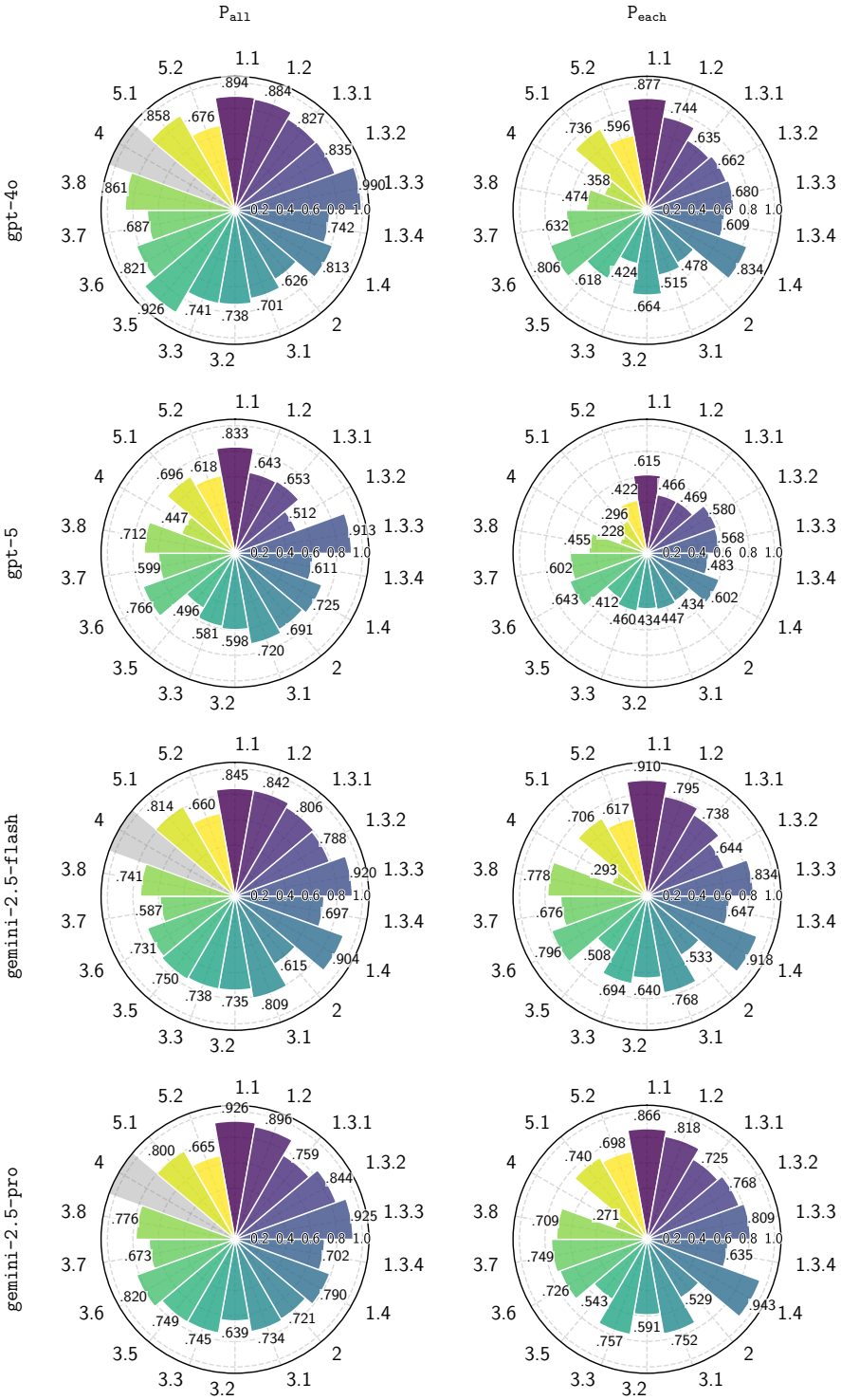

図 15　LLM とプロンプトの組合せに対する標準構造の節ごとの平均適合率

6 議論

6.1 構造標準化の応用

本研究の結果および考察は，LLM による SRS の構造標準化が RE において単なるフォーマット変換以上の有用性を持つことを示唆している．その 1 つが gpt-5 などにみられた，ISO 29148 の規定に忠実であろうとするために 元の SRS よりも詳細な記述を生成する傾向の利用である．この挙動は RQ1 における再現率の評価では不利に働いたものの，基本設計や詳細設計を支援する上では有用な特性となりうる．記述の詳細化に伴って元の文書では自明として省略されていた暗黙の条件が言語化され，ひいては形式化されうるためである．

この特性は元の SRS における記述の十分性を検証する用途にも応用可能である．評価基準を ISO 29148 に設定し，LLM が生成した標準化 SRS を分析することで，元の SRS における記述要素の欠落や不足を体系的に把握できる．実際に本実験では，標準化 SRS において [N/A] と判定された項目の中に元の SRS に対応する記述が存在しない例が確認された．これは ISO 29148 への準拠を SRS の十分性の基準とした場合に必要な記述要素の欠落が直接的に可視化されたことを意味する．また，仮に欠落が明示されなかったとしても標準化によって必要な記述要素が配置されるべき箇所が固定されるため，レビュアーはその特定の範囲に限定して記述の有無を確認すればよくなる．さらに，先に考察で示した図 15 のように，ある節における適合率の低さがその節にあるべき情報が元の SRS にないことを表現しうる．これらの性質を活用することでレビューに要する労力の低減が期待できる．

さらに，$\mathrm{P}_{\mathrm{all}}$ による構造標準化は複雑な元の SRS から不要な情報が削減し，理解しやすい文書として再構成できる可能性を示している．変換先のアウトラインを特定の読者を想定して工夫することで，その主体にとって必要十分な要約文書を生成するといったユースケースが考えられる．例えば，技術者向けの詳細な SRS とは別に，管理者や非技術者向けに簡略化や抽象化された要求記述を作成する場合などがこれに当たる．RQ1 で高い適合率が示されたことから，このような要約的な変換においても生成された内容は元の SRS に照らして概ね正確であると期待できる．

ただし，いずれの応用においても生成された SRS の人手による確認と修正は必須である．

6.2 構造標準化の影響

SRS の構造標準化にはそれ自体にいくつかの潜在的な課題が存在する．1 つは元の SRS に含まれるすべての情報が必ずしも標準化された構造へ完全にマッピングできるとは限らない点である．本研究では，元の SRS における各記述が標準化 SRS のいずれかのセクションに対応付け可能であると仮定している．しかし，SRS の構造的多様性を鑑みるとこの仮定が常に成立するとは限らない．そのため，標準化に伴い一部の情報が不可避的に欠落し，結果として要求定義が不完全となる恐れがある．

もう 1 つは元の SRS を変換することで関連する成果物に悪影響を及ぼす可能性である．典型的な成果物の例としてはトレーサビリティリンクが挙げられる．これは，要求文書と設計書やソースコードなどの成果物との関連性を管理する情報であるが，元の SRS に基づいてリンクが既に構築されている場合，構造標準化によってそれらが無効になる恐れがある．トレーサビリティの喪失は変更管理や影響分析を妨げ，結果として保守性を損なう可能性がある．

これらの課題はソフトウェア開発において重大な問題を引き起こしかねないため緩和策が必要となる．現状では手動による評価が唯一の現実的な選択肢であるが，それに伴う労力を低減する方法は考えられる．例えば，RQ1 で用いた適合率と再現率に基づく評価戦略の活用が挙げられる．この戦略では標準化前後の 2 つの SRS に対して個々の文ごとに他方からの含意度を測定する．このとき，他方の文書のどの記述から対象の文が含意されると判断したのかがわかるので，結果的に記述間のマッピングができる．この情報を利用することで標準化の過程で失われた情報，すなわち元の SRS にのみに固有に存在する記述の特定を支援できる．さらに，このマッピングによって標準化 SRS 内の各文が元の SRS のどの記述に対応するかが明らかになるため，元の SRS で構築されていたトレーサビリティリンクの参照および復元を支援することも可能となる．

6.3 今後の課題

今後の課題の 1 つはプロンプトテクニックや本タスクでの特徴を反映したプロンプトの設計である．本研究では $\mathtt{P}_{\mathtt{all}}$ と $\mathtt{P}_{\mathtt{each}}$ という単純な zero-shot プロンプトを設計した．一方で，LLM の生成精度を向上させるために，few-shot learning [13, 30] や Chain of Thought [31]，Retrieval Augmented Generation [32] などの多くのプロンプトテクニックが提案されている．RE のドメインにおいてもこれらの適用はもはや前提であり [33]，実用的な性能を得るためには必須であるといえる．

さらに，これらの一般的なプロンプトテクニックに加えて，SRS の構造標準化に特有の工夫が考えられる．1 つ目に標準化 SRS の長さを適切にできるプロンプト設計が挙げられる．SRS 標準化は一般的な要約タスクと異なり，既存の情報を抽象化または省略することは必ずしも望ましくない．そのため，標準化 SRS が元の SRS と同程度の分量を保持することが 1 つの必要条件として想定される．2 つ目に標準構造の節間の依存関係などを考慮したプロンプト設計が挙げられる．$\mathtt{P}_{\mathtt{each}}$ では各節を独立に生成するため，同一 SRS 内において相互に整合しない複数の記述が含まれる可能性がある．この問題に対処する方法として，節間の依存関係に基づく再帰的な生成を促すプロンプト設計が考えられる．また，要求記述の非一貫性の検出技術を後処理として適用する方法も考えられる．要求間の衝突を検出するアプローチとして ChatGPT や NLI の適用を試みた研究 [34, 35] もあり，限定的な条件下での適用可能性が実証されている．$\mathtt{P}_{\mathtt{each}}$ によって埋め込まれうる不整合についてもこれらの手法を後処理として適用し検知することで，修正が可能であると考えられる．

適用対象の拡張という観点からも今後の課題は存在する．例として，モダリティの拡張が挙げられる．本研究ではテキストのみからなる SRS を変換の対象とした．しかし，実際の SRS には，読者の理解を補助するために図や表が用いられることは少なくない．より広範な種類の SRS への適用可能性を調べるためにも，図表を含む SRS を対象としたプロンプトや手法を設計し，その評価を行うことは重要である．特に，GPT-4o などの近年の LLM のモデルではテキストだけでなく画像も入力できるようになっている．本研究で設計したような単純なプロンプトを用いた場合においても，画像を含む SRS への適用可能性を調査する必要がある．

6.4 妥当性への脅威

構成概念妥当性

本研究では対象とする SRS をテキストのみで構成されるものに限定している．しかし，ISO 29148 では SRS に図表を含めることも認めている．図表を用いることで特定の要求をより明確かつ適切に表現できる場合がある．この制限によって構造標準化の構成概念をテキストで表現可能な要求や仕様に限定することとなり，適用可能性を過大評価するおそれがある．

RQ1 の評価では文単位，すなわち文書の表層的な単位に基づいて意味の保持を評価した．しかし，文が意味を適切に捉えるための単位として最適であるかは自明ではない．脱文脈化によって文分割に伴う意味の喪失は低減されているものの，文という単位が必ずしも意味の最小単位と一致するとは限らない．加えて，文書構造上のつなぎや箇条書きの導入文など，それ自体に実質的な要求内容を含まず，記述の形式のみにその真偽が依存する文も存在する．これらの文が評価対象に含まれることは意味的等価性の評価においてノイズとなり，結果の解釈にバイアスを与える可能性がある．

RQ2 の実験ではチェックリストを用いて必須項目の有無を確認して標準化 SRS の標準への忠実度を検証した．しかしながら，この評価手法では標準化 SRS と元の SRS との関係性が考慮されておらず，RQ2 における LLM の構造標準化能力を過大評価している可能性がある．LLM は標準への適合度を高めようとする一方で，望ましくない出力を生成する恐れがある．その一例がハルシネーションであり，元の SRS に存在しない内容が生成されてしまう現象である．逆に，LLM が元の内容を過度に抽象化し，要求記述を十分に再構成できない可能性もある．LLM の構造標準化能力を適切に評価するためには標準化 SRS だけではなく，その元となった SRS も考慮に入れた評価を行う必要がある．

内的妥当性

RQ1 の実験結果は FV モデルの性能に強く依存している．本研究では，実験対象の SRS と同程度の文章量を持つ LLM-AggreFact 上でランダムより高い性能を示した fine-tuning 済みモデルを採用するにとどめ，当該モデルに対する追加の fine-tuning は実施していない．そのため，学習データと評価対

象である SRS との間のドメインシフトに起因して性能低下が生じている可能性がある．補正値の算出においても LLM-AggreFact 上で得られた性能がそのまま保持されることを前提としているため，FV モデルの性能低下は補正値の精度にも悪影響を及ぼしうる．したがって，評価結果の信頼性を確保するためには，対象とする SRS から検証用データを作成し，ドメイン適応を行うことでこの影響を軽減すべきであった．実際，同一のドメインおよびタスクのデータを用いて継続学習や fine-tuning を行うことで，分類性能が向上することが実証されている [36, 37]．

FV モデルの不完全性に対処するために Bayes 推定による観測値の補正を実施した．この推定においては文間の独立性などの強い仮定を置いて単純化している．こうした仮定が実際の誤りの発生構造を十分に反映できず，推定値に予期せぬバイアスをもたらしている可能性は否定できない．また，Bayes 推定自体の妥当性検証も実施しておらず，あくまで仮定された分布に基づく理論的な推論に留まっている．いくつかの標準化 SRS をサンプリングして人手による評価を行い，真の値と推定値との乖離を定量的に評価することが求められる．

サンプルサイズの小ささも統計的な観点から妥当性への脅威となりうる．本実験では，SRS の構造標準化および前処理としての脱文脈化に LLM を用いた．これらのプロセスは確率的であるが，本実験における試行回数はそれぞれ 3 回に限られている．そのため，得られた標準化 SRS や測定値の傾向が偶発的な変動によるものである可能性は排除できない．サンプル数を増やして統計的な信頼性を担保し，結果の頑健性を検証する必要がある．

外的妥当性

本研究では LLM として OpenAI の GPT および Google の Gemini を，構造標準化の対象 SRS として PURE データセットを用いた．異なる LLM や SRS を使用した場合，本実験とは異なる結果が得られる可能性がある．特に，本実験で採用したプロンプトは今回の実験設定に特化して調整されている．LLM にはトークン長の制限が存在し，例えば GPT-4o のコンテキストウィンドウは 128,000 トークン，最大出力長は 16,384 トークンである [*1]．本研究で使用した SRS は最大でも約 8,500 語と比較的小規模でありこれらの制限内に十分に収まる．しかし，実務における SRS はこれより遥かに長大になる場合がある．そのようなケースでは，本研究で使用したプロンプト $\texttt{P}_{\texttt{all}}$ および $\texttt{P}_{\texttt{each}}$ がトークン長の制約により有効に機能しない可能性がある．したがって，本手法を大規模な SRS へ適用する際には，分割処理などの追加的な対策が必要となる．

また，本研究では構造標準化の対象を，図表を含まない SRS に限定した．しかし，実務における SRS では図表などの非テキスト情報が重要な役割を果たす場合も多い．そのため，本実験の結果を実プロジェクトへ一般化する際には注意が必要である．

7　おわりに

本研究では任意の構造を持つ SRS を ISO 29148 が推奨する節構造へ自動的に変換するタスクを提案し，LLM を用いた構造標準化の適用可能性を評価した．実験の結果，標準化 SRS が元の SRS に対して情報の正確さと情報の保持を同時に高い水準で満たすことは困難である一方，LLM とプロンプトの組合せによってはいずれかの観点において高い性能を達成できることが確認された．また，$\mathtt{P}_{\mathtt{each}}$ により生成された標準化 SRS は ISO 29148 における必須記述項目を高い水準で満たすことが示された．

これらの結果から，LLM による SRS の構造標準化は単なる文書形式の変換にとどまらず，要求記述の十分性の検証やレビュー支援，要約的な要求整理など，要求工学における多様な応用可能性を有することが示唆される．一方で，情報の欠落やトレーサビリティの喪失といった潜在的な課題も存在し，実運用を想定した際には，評価手法や補助的な確認プロセスを含めた慎重な設計が求められる．

謝辞

本研究の遂行にあたり，多くの方々にご支援をいただきました．

楠本真二 教授には，テーマ設定にはじまり，実験，論文執筆，発表練習に至るまで，本研究の全過程において多大なるご指導とご協力を賜りました．各週のミーティングでは，私が持ち込んだ稚拙かつ断片的なアイデアに対しても，常に真摯に向き合い，ともに検討してくださいました．また，先生のご協力のもと実施した実験において，途中で変更や手戻りが生じた際にも，柔軟にご対応いただきました．このように，研究状況に応じて私の考えや判断を尊重し，試行錯誤を許容する環境を提供してくださったおかげで，自ら積極的に研究を進めることができたと感じています．さらに，寒い時期にはみかんを差し入れてくださったり，出張時には食事をご一緒させていただいたりと，研究外においても温かいご配慮をいただきました．深く感謝申し上げます．

柗本真佑 准教授には，本研究に関して，中間報告の際に内容および発表について的確なご助言をいただきました．また，先生には以前の研究テーマにおいてもお世話になりました．先生のご指導のおかげで，修士課程に進学してから，人生ではじめて国際学会に出席する機会を得ることができ，非常に貴重な経験をさせていただけました．その過程で培われた，議論する能力やプレゼンテーションの能力などは，本研究の遂行においても重要な基礎となっていると確信しています．深く感謝申し上げます．

事務補佐員の 橋本美砂子 氏には，研究や TA，アルバイトに関する各種手続きにおいて大変お世話になりました．海外出張の際には，長時間のフライトに備えた携行品をご用意いただいた他，現地での注意点についてもご教示いただくなど，多くのご配慮を賜りました．おかげで，安心して出張に臨むことができました．また，私の不手際により度々ご迷惑をおかけしたにもかかわらず，常に丁寧にご対応くださいました．研究活動に専念できる環境を提供してくださったことに，心より感謝申し上げます．

研究室の先輩方には，研究内外で大変お世話になりました．研究については，生意気な質問や議論にも快く応じてくださり，貴重なご意見をいただきました．研究以外の場面でも，飲み会や旅行，日々の雑談などで楽しい時間を共有させていただきました．研究室の後輩方には，旅行等のイベントの企画や手配から，研究室のゴミ出しまで，研究室生活を楽しく，快適に過ごせるようにご尽力をいただきました．皆様に感謝申し上げます．同期には，本修士課程のあらゆる場面において多大な支えをいただきました．最も気軽に相談できる存在であると同時に，研究をはじめとする様々な事柄について議論を重ねられる相手でもありました．また，研究の息抜きの誘いにも快く応じてくださり，多くの交流の機会を与えてくれました．精神的な支えであり，互いに切磋琢磨する仲間でもあった同期の存在は，私にとって非常に大きなものでした．ここに心より感謝申し上げます．

最後に，修士課程とこれまでの生活を支えてくれた家族に，この場を借りて改めて深く感謝します．

参考文献

[1] ISO/IEC/IEEE International Standard - Systems and Software Engineering – Life Cycle Processes – Requirements Engineering, *ISO/IEC/IEEE 29148:2018(E)*, pp. 1–104 (2018).

[2] Ferrari, A., Spagnolo, G. and Gnesi, S.: PURE: A Dataset of Public Requirements Documents, in *Proc. International Requirements Engineering Conference (RE)*, pp. 502–505 (2017).

[3] Denger, C. and Olsson, T.: *Quality Assurance in Requirements Engineering*, pp. 163–185, Springer (2005).

[4] Tamai, T. and Kamata, M. I.: Impact of Requirements Quality on Project Success or Failure, in *Design Requirements Engineering: A Ten-Year Perspective*, pp. 258–275, Springer (2009).

[5] Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K., Ajagbe, M., Chioasca, E.-V. and Batista-Navarro, R.: Natural Language Processing for Requirements Engineering: A Systematic Mapping Study, *Trans. Computing Surveys*, Vol. 54, No. 3, pp. 1–41 (2021).

[6] Cheng, H., Husen, J., Lu, Y., Racharak, T., Yoshioka, N., Ubayashi, N. and Washizaki, H.: Generative AI for Requirements Engineering: A Systematic Literature Review (2025), arXiv preprint.

[7] IEEE Recommended Practice for Software Requirements Specifications, *IEEE Std 830-1998*, pp. 1–40 (1998).

[8] Franch, X., Glinz, M., Mendez, D. and Seyff, N.: A Study About the Knowledge and Use of Requirements Engineering Standards in Industry, *Trans. Software Engineering*, Vol. 48, No. 9, pp. 3310–3325 (2022).

[9] Aoyama, M. and Nakane, T.: ReqQA: A Software Requirements Specifications Quality Analyzer and Its Application, *Trans. Information Processing Society of Japan*, Vol. 57, No. 2, pp. 694–706 (2016), (in Japanese).

[10] Chikh, A. and Aldayel, M.: A New Traceable Software Requirements Specification Based on IEEE 830, in *Proc. International Conference on Computer Systems and Industrial Informatics (ICCSII)*, pp. 1–6 (2012).

[11] Thitisathienkul, P. and Prompoon, N.: Quality Assessment Method for Software Requirements Specifications Based on Document Characteristics and Its Structure, in *Proc. International Conference on Trustworthy Systems and Their Applications (TSA)*, pp. 51–60 (2015).

[12] Takoshima, A. and Aoyama, M.: A Two-Stage Inspection Method for Automotive Software Systems and Its Practical Applications, in *Proc. International Requirements Engineering Con-*

ference (RE), pp. 313–322 (2016).

[13] Brown, T. et al., : Language Models Are Few-Shot Learners, in *Proc. Conference on Neural Information Processing Systems (NeurIPS)*, pp. 1877–1901 (2020).

[14] Krishna, M., Gaur, B., Verma, A. and Jalote, P.: Using LLMs in Software Requirements Specifications: An Empirical Evaluation, in *Proc. International Requirements Engineering Conference (RE)*, pp. 475–483 (2024).

[15] Norheim, J. and Rebentisch, E.: Structuring Natural Language Requirements with Large Language Models, in *Proc. International Requirements Engineering Conference Workshops (REW)*, pp. 68–71 (2024).

[16] Mavin, A., Wilkinson, P., Harwood, A. and Novak, M.: Easy Approach to Requirements Syntax (EARS), in *Proc. International Requirements Engineering Conference (RE)*, pp. 317–322 (2009).

[17] Zhu, T., Cordeiro, L. and Sun, Y.: ReqInOne: A Large Language Model-Based Agent for Software Requirements Specification Generation, in *Proc. International Requirements Engineering Conference (RE)*, pp. 449–457 (2025).

[18] Hey, T., Keim, J., Koziolek, A. and Tichy, W.: NoRBERT: Transfer Learning for Requirements Classification, in *Proc. International Requirements Engineering Conference (RE)*, pp. 169–179 (2020).

[19] Choi, E., Palomaki, J., Lamm, M., Kwiatkowski, T., Das, D. and Collins, M.: Decontextualization: Making Sentences Stand-Alone, *Trans. Association for Computational Linguistics*, Vol. 9, pp. 447–461 (2021).

[20] Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, tau W., Koh, P., Iyyer, M., Zettlemoyer, L. and Hajishirzi, H.: FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation, in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 12076–12100 (2023).

[21] Thorne, J., Vlachos, A., Christodoulopoulos, C. and Mittal, A.: FEVER: A Large-scale Dataset for Fact Extraction and VERification, in *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 809–819 (2018).

[22] Bowman, S., Angeli, G., Potts, C. and Manning, C.: A Large Annotated Corpus for Learning Natural Language Inference, in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 632–642 (2015).

[23] Williams, A., Nangia, N. and Bowman, S.: A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference, in *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 1112–1122 (2018).

[24] Zha, Y., Yang, Y., Li, R. and Hu, Z.: AlignScore: Evaluating Factual Consistency with A Unified Alignment Function, in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 11328–11348 (2023).

[25] Tang, L., Laban, P. and Durrett, G.: MiniCheck: Efficient Fact-Checking of LLMs on Grounding Documents, in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8818–8847 (2024).

[26] Lei, D., Li, Y., Li, S., Hu, M., Xu, R., Archer, K., Wang, M., Ching, E. and Deng, A.: FactCG: Enhancing Fact Checkers with Graph-Based Multi-Hop Data, in *Proc. Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 5002–5020 (2025).

[27] Gunjal, A. and Durrett, G.: Molecular Facts: Desiderata for Decontextualization in LLM Fact Verification, in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3751–3768 (2024).

[28] Deshpande, A., Jimenez, C., Chen, H., Murahari, V., Graf, V., Rajpurohit, T., Kalyan, A., Chen, D. and Narasimhan, K.: C-STS: Conditional Semantic Textual Similarity, in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5669–5690 (2023).

[29] Boyarchuk, A., Pavlova, O., Bodnar, M. and Lopatto, I.: Approach to the Analysis of Software Requirements Specification on Its Structure Correctness, in *Proc. International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS)*, pp. 85–95 (2020).

[30] Kojima, T., Gu, S., Reid, M., Matsuo, Y. and Iwasawa, Y.: Large Language Models Are Zero-Shot Reasoners, in *Proc. Conference on Neural Information Processing Systems (NeurIPS)*, pp. 22199–22213 (2022).

[31] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. and Zhou, D.: Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, in *Proc. Conference on Neural Information Processing Systems (NeurIPS)*, pp. 24824–24837 (2022).

[32] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M. and

Wang, H.: Retrieval-Augmented Generation for Large Language Models: A Survey (2024), arXiv preprint.

[33] Huang, K., Wang, F., Huang, Y. and Arora, C.: Prompt Engineering for Requirements Engineering: A Literature Review and Roadmap, in *Proc. International Requirements Engineering Conference Workshops (REW)*, pp. 548–557 (2025).

[34] Fantechi, A., Gnesi, S., Passaro, L. and Semini, L.: Inconsistency Detection in Natural Language Requirements using ChatGPT: a Preliminary Evaluation, in *Proc. International Requirements Engineering Conference (RE)*, pp. 335–340 (2023).

[35] Fazelnia, M., Koscinski, V., Herzog, S. and Mirakhorli, M.: Lessons from the Use of Natural Language Inference (NLI) in Requirements Engineering Tasks, in *Proc. International Requirements Engineering Conference (RE)*, pp. 103–115 (2024).

[36] Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D. and Smith, N.: Don't Stop Pretraining: Adapt Language Models to Domains and Tasks, in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 8342–8360 (2020).

[37] Calderon, N., Porat, N., Ben-David, E., Chapanin, A., Gekhman, Z., Oved, N., Shalumov, V. and Reichart, R.: Measuring the Robustness of NLP Models to Domain Shifts, in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 126–154 (2024).

付録 A　構造標準化で LLM に与えたプロンプト

P_{all} プロンプト

````
## Persona
You are an experienced requirements engineer.

## Instructions
You will be provided with:
- The original software requirements specification (SRS),
- The standard SRS section structure,
- Descriptions of the required content for each section.

Restructure the original SRS to conform to the standard structure.
Retain necessary and sufficient information from the original SRS.
Output "[N/A]" for any required content missing from the original SRS.

## Output format
Output strictly in Markdown (CommonMark) format.
Use only headers, paragraphs, and lists.
Do NOT use any tables, code blocks, HTML, or extension syntax (e.g., Mermaid).

## The original SRS
```\n{srs_orig}\n```

## The standard SRS section structure
```\n{std_structure}\n```

## Descriptions of the required content for each section
```\n{description}\n

## Output
```markdown
````

P_{each} プロンプト

```
## Persona
You are an experienced requirements engineer.

## Instructions
You will be provided with:
- The original software requirements specification (SRS),
- The standard SRS section structure,
- The section to be generated,
- The description of the required content for the section.

Restructure the original SRS to conform to the standard structure.
Output only the specified section.
Retain necessary and sufficient information from the original SRS.
Output "[N/A]" for any required content missing from the original SRS.

## Output format
Output strictly in Markdown (CommonMark) format.
Use only headers, paragraphs, and lists.
Do NOT use any tables, code blocks, HTML, or extension syntax (e.g., Mermaid).

## The original SRS
‘‘‘\n{srs_orig}\n‘‘‘

## The standard SRS section structure
‘‘‘\n{std_structure}\n‘‘‘

## The section to be generated
‘‘‘\n{section_tobe_generated}\n‘‘‘

## The description of the required content for the section
‘‘‘\n{description}\n‘‘‘

## Output
‘‘‘markdown
{section_tobe_generated}
```

付録 B　事後分布の導出

導出で頻出する積分計算を先に提示しておく．任意の自然数 a, b, c, d について次が成り立つ．

$$\int_0^1 x^a (1-x)^b \mathrm{Beta}(x \mid c, d) \mathrm{d}x = \frac{B(a+c, b+d)}{B(c, d)}.$$

$\mathrm{Beta}(x \mid c, d)$ はパラメータを (c, d) とする Beta 分布の確率密度関数を表す．つまり，

$$\mathrm{Beta}(x \mid c, d) = \frac{x^{c-1}(1-x)^{d-1}}{B(c, d)}.$$

また，B は以下の式で定義される Beta 関数である．

$$B(a, b) = \int_0^1 x^{a-1}(1-x)^{b-1} \mathrm{d}x.$$

特に，$B(1, 1) = 1$ である．

事後分布 $\mathbf{P}(k \mid \mathbf{y}, M)$ を導出する．文の列 $(s_1, s_2, \ldots, s_n)$ に対し，$\mathbf{z} = (z_1, z_2, \ldots, z_n)$ は各文の真の判定結果，$\mathbf{y} = (y_1, y_2, \ldots, y_n)$ は各文の FV モデルの判定結果である．$M = (\mathrm{TP}, \mathrm{FN}, \mathrm{FP}, \mathrm{TN})$ は同 FV モデルのあるベンチマーク上での混同行列を表す．また，$k = \sum_i z_i$ である．以降，$\mathbf{P}$ で確率（質量関数）を，$\boldsymbol{p}$ で確率密度関数を表す．ただし，両者は区別する一方で，どの確率変数に対応するかについては文脈上一意に定まるため明示しない．

まず，$\mathbf{z}$ についての周辺化と Bayes の定理を適用すると，

$$\begin{aligned}
\mathbf{P}(k \mid \mathbf{y}, M) &= \sum\nolimits_{\mathbf{z} \in \{0,1\}^n} \mathbf{P}(k, \mathbf{z} \mid \mathbf{y}, M) \\
&= \sum\nolimits_{\mathbf{z}} \mathbf{P}(k \mid \mathbf{z}, \mathbf{y}, M) \mathbf{P}(\mathbf{z} \mid \mathbf{y}, M) \\
&= \sum\nolimits_{\mathbf{z}} \mathbf{P}(k \mid \mathbf{z}, \mathbf{y}, M) \frac{\mathbf{P}(\mathbf{y} \mid \mathbf{z}, M) \mathbf{P}(\mathbf{z} \mid M)}{\mathbf{P}(\mathbf{y} \mid M)}
\end{aligned}$$

ここで，$\mathbf{P}(\mathbf{y} \mid M)$ は k および $\mathbf{z}$ に依存しない正規化定数であるため，k に関する事後分布のカーネルとしてはこれを無視できる．加えて，図 3 に基づいて条件付けを簡単化すると，

$$\begin{aligned}
\mathbf{P}(k \mid \mathbf{y}, M) &\propto \sum\nolimits_{\mathbf{z}} \mathbf{P}(k \mid \mathbf{z}, \mathbf{y}, M) \mathbf{P}(\mathbf{y} \mid \mathbf{z}, M) \mathbf{P}(\mathbf{z} \mid M) \\
&= \sum\nolimits_{\mathbf{z}} \mathbf{P}(k \mid \mathbf{z}) \mathbf{P}(\mathbf{y} \mid \mathbf{z}, M) \mathbf{P}(\mathbf{z}).
\end{aligned}$$

k は $\mathbf{z}$ から $k = \sum_i z_i$ として決定的に定まる．そのため，δ を Kronecker のデルタとすると $\mathbf{P}(k \mid \mathbf{z}) = \delta_{k, \sum_i z_i}$ と書ける．したがって，

$$\mathbf{P}(k \mid \mathbf{y}, M) \propto \sum\nolimits_{\mathbf{z}} \delta_{k, \sum_i z_i} \mathbf{P}(\mathbf{y} \mid \mathbf{z}, M) \mathbf{P}(\mathbf{z}).$$

次に $\mathbf{P}(\mathbf{z})$ を計算していく．パラメータを $(1,1)$ とする Beta 分布に従う θ に対して周辺化すると，

$$\begin{aligned}
\mathbf{P}(\mathbf{z}) &= \int_0^1 \boldsymbol{p}(\mathbf{z},\theta)\mathrm{d}\theta \\
&= \int \mathbf{P}(\mathbf{z}\mid\theta)\boldsymbol{p}(\theta)\mathrm{d}\theta \\
&= \int \textstyle\prod_{i=1}^n \mathbf{P}(z_i\mid\theta)\boldsymbol{p}(\theta)\mathrm{d}\theta \\
&= \int \theta^{\sum_i z_i}(1-\theta)^{n-\sum_i z_i}\mathrm{Beta}(\theta\mid 1,1)\mathrm{d}\theta \\
&= \frac{B(\sum_i z_i+1, n-\sum_i z_i+1)}{B(1,1)} \\
&= B\left(\textstyle\sum_i z_i+1, n-\sum_i z_i+1\right).
\end{aligned}$$

δ で $\mathbf{z}$ が $\sum_i z_i = k$ の場合に限られているので，$\mathbf{P}(k\mid\mathbf{y},M)$ は次のように書き直せる．

$$\begin{aligned}
\mathbf{P}(k\mid\mathbf{y},M) &\propto \textstyle\sum_{\mathbf{z}} \delta_{k,\sum_i z_i}\mathbf{P}(\mathbf{y}\mid\mathbf{z},M)B\left(\sum_i z_i+1, n-\sum_i z_i+1\right) \\
&= \textstyle\sum_{\mathbf{z}} \delta_{k,\sum_i z_i}\mathbf{P}(\mathbf{y}\mid\mathbf{z},M)B(k+1,n-k+1) \\
&= B(k+1,n-k+1)\textstyle\sum_{\mathbf{z}} \delta_{k,\sum_i z_i}\mathbf{P}(\mathbf{y}\mid\mathbf{z},M).
\end{aligned}$$

次に $\mathbf{z}$ についての和の計算を進める．BAC の伝送確率である tpr と fpr をそれぞれ，パラメータを $(\mathrm{TP}+1,\mathrm{FN}+1),(\mathrm{FP}+1,\mathrm{TN}+1)$ とする Beta 分布に従う確率変数として周辺化すると，

$$\begin{aligned}
\mathbf{P}(\mathbf{y}\mid\mathbf{z},M) &= \int_{(\mathrm{tpr},\mathrm{fpr})\in[0,1]^2} \boldsymbol{p}(\mathbf{y},\mathrm{tpr},\mathrm{fpr}\mid\mathbf{z},M)\mathrm{d}(\mathrm{tpr},\mathrm{fpr}) \\
&= \int \mathbf{P}(\mathbf{y}\mid\mathrm{tpr},\mathrm{fpr},\mathbf{z},M)\boldsymbol{p}(\mathrm{tpr},\mathrm{fpr}\mid\mathbf{z},M)\mathrm{d}(\mathrm{tpr},\mathrm{fpr}) \\
&= \int_0^1\int_0^1 \mathbf{P}(\mathbf{y}\mid\mathrm{tpr},\mathrm{fpr},\mathbf{z},M)\boldsymbol{p}(\mathrm{tpr}\mid\mathbf{z},M)\boldsymbol{p}(\mathrm{fpr}\mid\mathbf{z},M)\mathrm{d}(\mathrm{tpr})\mathrm{d}(\mathrm{fpr}) \\
&= \iint \mathbf{P}(\mathbf{y}\mid\mathrm{tpr},\mathrm{fpr},\mathbf{z})\boldsymbol{p}(\mathrm{tpr}\mid\mathrm{TP},\mathrm{FN})\boldsymbol{p}(\mathrm{fpr}\mid\mathrm{FP},\mathrm{TN})\mathrm{d}(\mathrm{tpr})\mathrm{d}(\mathrm{fpr}) \\
&= \iint \textstyle\prod_{i=1}^n \mathbf{P}(y_i\mid\mathrm{tpr},\mathrm{fpr},z_i) \\
&\qquad \times \mathrm{Beta}(\mathrm{tpr}\mid\mathrm{TP}+1,\mathrm{FN}+1)\mathrm{Beta}(\mathrm{fpr}\mid\mathrm{FP}+1,\mathrm{TN}+1)\mathrm{d}(\mathrm{tpr})\mathrm{d}(\mathrm{fpr}).
\end{aligned}$$

ここで $\mathbf{P}(y_i\mid\mathrm{tpr},\mathrm{fpr},z_i)$ は (y_i,z_i) の値ごとに

$$\begin{aligned}
&\mathbf{P}(y_i=1\mid\mathrm{tpr},\mathrm{fpr},z_i=1)=\mathrm{tpr}, \\
&\mathbf{P}(y_i=0\mid\mathrm{tpr},\mathrm{fpr},z_i=1)=1-\mathrm{tpr}, \\
&\mathbf{P}(y_i=1\mid\mathrm{tpr},\mathrm{fpr},z_i=0)=\mathrm{fpr}, \\
&\mathbf{P}(y_i=0\mid\mathrm{tpr},\mathrm{fpr},z_i=0)=1-\mathrm{fpr}
\end{aligned}$$

となる．よって，$\mathbf{z}$ と $\mathbf{y}$ に対して値が 1 となるような添字集合

$$Z_1 = \{i \mid i \in \{1, 2, \ldots, n\},\, z_i = 1\}, \quad Y_1 = \{i \mid i \in \{1, 2, \ldots, n\},\, y_i = 1\}$$

を考えると，これらの組合せで各 (y_i, z_i) の値での場合分けを表現でき，

$$\begin{aligned}
\mathbf{P}(\mathbf{y} \mid \mathbf{z}, M) \propto & \iint \mathrm{tpr}^{|Z_1 \cap Y_1|}(1-\mathrm{tpr})^{|Z_1 \cap Y_1^c|}\mathrm{fpr}^{|Z_1^c \cap Y_1|}(1-\mathrm{fpr})^{|Z_1^c \cap Y_1^c|} \\
& \quad \times \mathrm{Beta}(\mathrm{tpr} \mid \mathrm{TP}+1, \mathrm{FN}+1)\mathrm{Beta}(\mathrm{fpr} \mid \mathrm{FP}+1, \mathrm{TN}+1)\mathrm{d}(\mathrm{tpr})\mathrm{d}(\mathrm{fpr}) \\
= & \int \mathrm{tpr}^{|Z_1 \cap Y_1|}(1-\mathrm{tpr})^{|Z_1 \cap Y_1^c|}\mathrm{Beta}(\mathrm{tpr} \mid \mathrm{TP}+1, \mathrm{FN}+1)\mathrm{d}(\mathrm{tpr}) \\
& \quad \times \int \mathrm{fpr}^{|Z_1^c \cap Y_1|}(1-\mathrm{fpr})^{|Z_1^c \cap Y_1^c|}\mathrm{Beta}(\mathrm{fpr} \mid \mathrm{FP}+1, \mathrm{TN}+1)\mathrm{d}(\mathrm{fpr}) \\
= & \frac{B(|Z_1 \cap Y_1| + \mathrm{TP} + 1, |Z_1 \cap Y_1^c| + \mathrm{FN} + 1)}{B(\mathrm{TP}+1, \mathrm{FN}+1)} \\
& \quad \times \frac{B(|Z_1^c \cap Y_1| + \mathrm{FP} + 1, |Z_1^c \cap Y_1^c| + \mathrm{TN} + 1)}{B(\mathrm{FP}+1, \mathrm{TN}+1)}
\end{aligned}$$

となる．δ で $\mathbf{z}$ が $\sum_i z_i = k$ の場合に限られることから，すべての $\mathbf{z}$ についてではなく $|Z_1| = k$ なる $\mathbf{z}$ のみで和をとればいい．そこで，要素数が k の添字集合の族

$$\mathcal{F}_k = \{Z_1 \mid Z_1 \subset \{1, 2, \ldots, n\},\, |Z_1| = k\}$$

を考えれば，

$$\begin{aligned}
\mathbf{P}(k \mid \mathbf{y}, M) \propto & B(k+1, n-k+1) \sum\nolimits_{Z_1 \in \mathcal{F}_k} \\
& \quad \times B(|Z_1 \cap Y_1| + \mathrm{TP} + 1, |Z_1 \cap Y_1^c| + \mathrm{FN} + 1) \\
& \quad \times B(|Z_1^c \cap Y_1| + \mathrm{FP} + 1, |Z_1^c \cap Y_1^c| + \mathrm{TN} + 1).
\end{aligned}$$

これで事後分布のカーネルが陽に表せた．しかし実際に分布全体を計算する際にはすべての $k \in \{0, 1, \ldots, n\}$ と $Z_1 \in \mathcal{F}_k$ について和をとる必要があるため，全体で $O(2^n)$ の計算量が必要となる．そこで，添字集合の要素数のみに着目することで計算量が削減できるように式変形する．

各集合の要素数は $k_1 = |Z_1 \cap Y_1|$ が決まれば n と $m = \sum_i y_i$ から次のように計算できる．

$$|Z_1 \cap Y_1^c| = k - k_1, \quad |Z_1^c \cap Y_1| = m - k_1, \quad |Z_1^c \cap Y_1^c| = n - k - m + k_1.$$

これらを使うと各 k_1 の値についての和に整理し直すことができる．つまり，

$$\begin{aligned}
\mathbf{P}(k \mid \mathbf{y}, M) \propto & B(k+1, n-k+1) \sum\nolimits_{k_1=L}^{U} \sum\nolimits_{Z_1 \in \mathcal{F}_k} \delta_{k_1, |Z_1 \cap Y_1|} \\
& \quad \times B(k_1 + \mathrm{TP} + 1, k - k_1 + \mathrm{FN} + 1) \\
& \quad \times B(m - k_1 + \mathrm{FP} + 1, n - k - m + k_1 + \mathrm{TN} + 1).
\end{aligned}$$

ただし，L, U は k_1 がとりうる値の範囲であり，

$$L = \max(0, k - n + m), \quad U = \min(k, m).$$

ここで，Y_1 が決まっているときの，$|Z_1 \cap Y_1| = k_1$ であるような $Z_1 \in \mathcal{F}_k$ の数を考えると，

- Y_1 の m 個の 1 から k_1 個の 1 を決める数だけ $Z_1 \cap Y_1$ は存在し，
- Y_1 に含まれない残りの $(n - m)$ 個の 1 から $(k - k_1)$ 個の 1 を決める数だけ Z_1 は存在する．

したがって，B の積の部分が各 k_1 に対してこの場合の数だけ足し合わされるので，

$$\begin{aligned}\mathbf{P}(k \mid \mathbf{y}, M) \propto B(k+1, n-k+1) \sum_{k_1=L}^{U} \binom{m}{k_1} \binom{n-m}{k-k_1} \\ \times B(k_1 + \mathrm{TP} + 1, k - k_1 + \mathrm{FN} + 1) \\ \times B(m - k_1 + \mathrm{FP} + 1, n - k - m + k_1 + \mathrm{TN} + 1).\end{aligned}$$

これで分布全体の計算量を $O(n^2)$ に削減できた．

付録 C　RQ2 のチェックリスト

1.1 Purpose ($E_{1.1}$)

□ purpose of software (ps)

1.2 Scope ($E_{1.2}$)

□ the software product's identifying (isp)

□ what software product will provide (spd)

□ benefit of the software product (spb)

□ objectives of the software product (spo)

□ goals of the software product (spg)

1.3.1 Product perspective ($E_{1.3.1}$)

□ software system's relationship to other related products (srrp)

□ software operation within the system interfaces (sosi)

□ software operation within the user interfaces (soui)

□ software operation within the hardware interfaces (sohi)

□ software operation within the software interfaces (soswi)

□ software operation within the communications interfacse (soci)

□ software operation within the memory (som)

□ software operation within the operations (soo)

□ software operation within the site adaprion requirements (sosar)

1.3.2 Product functions ($E_{1.3.2}$)

□ major future functions of the software (mf)

1.3.3 User characteristics ($E_{1.3.3}$)

□ general characteristics of the software product's groups of users (gcigu)

□ general characteristics which influence usability (gciu)

1.3.4 Limitations ($E_{1.3.4}$)

□ regulatory policies (rp)

□ limitations of hardware (hwl)

□ interfaces to other applications (ioa)

□ parallel operation (plo)

□ audit functions (af)

□ control functions (cf)

□ requirements of higher-order language (holr)

□ protocols of signal handshake (shp)

□ quality requirements (quar)

□ criticality of the application (ca)

□ considerations of safety and security (ssc)

□ physical/mental considerations (pmc)

1.4 Definitions ($E_{1.4}$)

☐ definitions for any words or phrases that have meaning beyond dictionaries (dbd)

2 References (E_2)

☐ list of referred documents (lrd)

☐ document title, report number, date and publishing organization (tndo)

☐ sources which the references can be obtained from (srof)

3.1 Functions ($E_{3.1}$)

☐ basic actions that must take place in the software when receiving and processing inputs (faapi)

☐ basic actions that must occur in the software when processing and generating outputs (fapgo)

☐ checks of validity on the inputs (vci)

☐ exact sequence of operations (eso)

☐ responses to abnormal situations (ras)

☐ parameters' effect (ep)

☐ relationship of inputs and outputs (rsoi)

3.2 Performance requirements ($E_{3.2}$)

☐ static numerical requirements (snr)

☐ dynamic numerical requirements (dnr)

☐ number of supported terminals (nts)

☐ number of supported simultaneous users (nssu)

☐ amount and type of handled information (athi)

☐ numbers of transactions (nota)

☐ number of tasks (not)

☐ amount of the processed data within certain time periods for normal workload conditions (adnw)

☐ amount of the processed data within certain time periods for peak workload conditions (adpw)

☐ requirements of performance (prq)

3.3 Usability requirements ($E_{3.3}$)

☐ usability requirements (ubr)

☐ measurable criteria of effectiveness in specific use contexts (mec)

☐ measurable criteria of efficiency in specific use contexts (mefc)

☐ measurable criteria of satisfaction in specific use contexts (msc)

3.5 Logical database requirements ($E_{3.5}$)

☐ types of used information (tiuvf)

☐ frequency of use (fqu)

☐ accessing the capabilities (asc)

☐ entities of data and their relationships (der)

☐ constraints of integrity (ics)

☐ requirements of data retention (drr)

3.6 Design constraints ($E_{3.6}$)

☐ the software system design's constraints which are imposed by external standards (ces)

☐ the software system design's constraints which are imposed by regulatory requirements (crr)

☐ the software system design's constraints which are imposed by project limitations (cpl)

3.7 Software system attributes ($E_{3.7}$)

- ☐ reliability (rb)
- ☐ availability (avb)
- ☐ security (scr)
- ☐ certain techniques of cryptographic (cct)
- ☐ data sets of specific log or history (slhd)
- ☐ certain functions to different modules (fdm)
- ☐ communications between some areas of the software product (csa)
- ☐ integrity of data for critical variables (dicv)
- ☐ privacy of data (dp)
- ☐ maintainability (mb)
- ☐ portability (pb)
- ☐ the percentage of elements with host-dependent code (pehdc)
- ☐ the percentage of hostdependent code (pchd)
- ☐ portable language use (uppl)
- ☐ particular compiler or language subset use (upcls)
- ☐ operating system use (upos)

3.8 Supporting information ($E_{3.8}$)

- ☐ sample input/output formats (siof)
- ☐ cost analysis studies' descriptions (dcas)
- ☐ user surveys' results (rus)
- ☐ supporting or background information that can help the readers of the SRS (sbi)
- ☐ description of the problems which will be solved by the software (dpss)
- ☐ special packaging instructions for the code and the media for security, export, initial loading (spi)

4 Verification (E_4)

- ☐ approaches for verification (va)
- ☐ methods for qualifying the software (mqs)

5.1 Assumptions and dependencies ($E_{5.1}$)

- ☐ factors which influence the requirements (far)

5.2 Acronyms and abbreviations ($E_{5.2}$)

- ☐ factors which influence the requirements (far)