

機械学習信頼性ソリューションパターンにおけるパターン適用支援の枠組みの提案

TAKUMI AYUKAWA, Waseda University, Tokyo
MARTIN GUTSCHE, Waseda University, Tokyo
JATI H. HUSEN, Waseda University, Tokyo
NOBUKAZU YOSHIOKA, Waseda University, Tokyo
HIRONORI WASHIZAKI, Waseda University, Tokyo
YOSHIAKI FUKAZAWA, Waseda University, Tokyo

自動運転などの信頼性の確保が重要な領域の機械学習システムにおいて、ゴール構造表記法により信頼性の論証ができる。そして、機械学習信頼性ソリューションパターンの適用により、高信頼、高効率に機械学習システムの開発方法を利用できる。しかし、機械学習信頼性ソリューションパターンの適用について、エンジニアがパターンを誤って適用する問題がある。例えば、複雑なパターンでは人為的な誤りが発生する可能性がある。また、エンジニアは必ずしも機械学習の専門家とは限らないため、機械学習信頼性ソリューションパターンを正しく利用できない場合もある。その解決のために我々はパターン適用支援の枠組みを提案する。本研究のパターン適用により、エンジニアの正確なパターン適用が期待される。そして、ケーススタディにより本研究の手法の実用性を確認した。

CCS Concepts: • **Software and its engineering** → **Design patterns**; *Software design engineering*; • **Theory of computation** → **Machine learning theory**.

Additional Key Words and Phrases: Pattern application, Goal Structuring Notation

ACM Reference Format:

TAKUMI AYUKAWA, MARTIN GUTSCHE, JATI H. HUSEN, NOBUKAZU YOSHIOKA, HIRONORI WASHIZAKI, and YOSHIAKI FUKAZAWA. 2024. 機械学習信頼性ソリューションパターンにおけるパターン適用支援の枠組みの提案. 1, 1 (April 2024), ?? pages. <https://doi.org/XXXXXXXX.XXXXXXX>

1 はじめに

信頼性の確保が重要な領域の機械学習システムにおいて、要求や価値を基にした信頼性の議論は重要である。信頼性の確保が重要な領域の機械学習システムの例として、自動運転車や医療診断における画像認識の機械学習システムがある [1][2][3]。そして、信頼性を保障するためには機械学習システムの設計段階から信頼性を考慮する必要がある。特に、機械学習システムが大規模となるにつれ、機械学習の技術的な面のみならず、要求や価値と適合するシステム設計が必要である [4]。例えば、機械学習の教師あり学習において、クラス分類によって予め定義されたクラスにデータを分類する場合を考える。この時、技術的な面からあるクラスにおける認識の精度を上げるのみでは必要以上に精度が上がり、結果的に他のより重要なクラスの精度が下がる場合もある。自動運転の例では自動車の認識率を上げたことにより、人間の認識率

Authors' addresses: TAKUMI AYUKAWA, Waseda University, Tokyo; MARTIN GUTSCHE, Waseda University, Tokyo; JATI H. HUSEN, Waseda University, Tokyo; NOBUKAZU YOSHIOKA, Waseda University, Tokyo; HIRONORI WASHIZAKI, Waseda University, Tokyo; YOSHIAKI FUKAZAWA, Waseda University, Tokyo.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM XXXX-XXXX/2024/4-ART

<https://doi.org/XXXXXXXX.XXXXXXX>

が下がる場合もある。その結果、人間の認識率が要求される精度を下回る場合、この自動車の精度の向上が適切とは限らない。そのため、要求や価値を基に精度の必要な向上の程度を設定する必要がある。

よって、信頼性の議論を明確に記述する方法が必要であり、パターンの利用が効果的である。例えば、ゴール構造表記法 (Goal Structuring Notation) により議論を明確に記述できる [5]。そして、議論を効率的に記述するためにソフトウェア工学の視点が必要である。具体的には、機械学習システムの信頼性を論証する際のパターンの適用が必要になる。パターンとはソフトウェアの開発の際に、繰り返し頻出する開発方法をまとめたものである [6][7]。例えば、機械学習信頼性ソリューションパターンの参照により、高信頼、高効率に機械学習システムの開発方法を利用できる。以上のように、パターンの適用によって信頼性の議論を効果的に記述することができる。

しかし、機械学習信頼性ソリューションパターンの適用の際に人為的な誤りが発生する問題がある。例えば、パターンによっては複雑なパターンもあるため、エンジニアがパターンを適用する際に誤りが発生する可能性がある。また、エンジニアは必ずしも機械学習の専門家とは限らない。そのため、パターンの適用はエンジニアにとって学習コストがかかり、パターンの参照が非効率的になる場合がある。その結果、パターンを正しく利用できない場合がある。実際に我々の研究を進める中でゴール構造表記法の中で機械学習信頼性ソリューションパターンを適用する場面があったが、それは複雑なパターンであったため適用の際に人為的な誤りが発生した。しかし、パターンは正しいという無意識の前提があったため、パターンの誤りに気づくことが難しく、誤りを特定するまでに時間がかかった。以上のように、パターンの適用時に人為的な誤りが発生する問題があるため、その問題を解決するための工夫が必要になる。

そこで本研究では、機械学習信頼性ソリューションパターンの適用時に人為的な誤りが起こる問題を解決するために、パターン適用支援の枠組みの提案を行なった。我々はパターン適用支援の枠組みに関する研究を行い、プラグイン開発を行った。ユーザの動作としては、はじめにパターンを選択し、パターンに合致している部分をパラメータとして入力する。そして、適用ボタンを押下し、入力されなかった部分の新規作成によりパターンを自動適用する。このようにパターンの自動的な適用により人為的な誤りを防止できる。さらに、本研究ではケーススタディによる検証を行い、本研究の手法の実用性を確認した。

次に本研究の貢献を以下に示す。

- ・機械学習信頼性ソリューションパターンの適用支援の枠組みの提案
- ・パターン適用を自動的に行うツールの実装
- ・ケーススタディによる検証

最後に、本論文の構成は以下のとおりである。2章では、ゴール構造表記法 (Goal Structuring Notation) や本研究で利用した *astah** System Safety、機械学習信頼性ソリューションパターンについて述べる。3章では、パターン適用支援の枠組みを提案する。4章では、ケーススタディにより本研究の手法の実用性を確認する。5章では、本研究の提案手法に関する考察を行う。6章では、本研究に関連する研究を述べる。最後に7章では、本研究で得られた成果をまとめ、今後の展望を示す。

2 背景

2.1 ゴール構造表記法 (Goal Structuring Notation)

ゴール構造表記法とは議論の構造をモデル化し、議論の要素の関係を明確にする表記法である [5]。これにより議論における主張の根拠などを可視化できる。表1にそれぞれの要素の意味を示し、図1にそれぞれの要素の表現を示す。例えば、Goalであれば四角で囲む図形によって表現され、SupportedByは黒の矢印で表される。これらの組み合わせにより議論の構造を明確にする。慣習的に各要素のタイトルは図1のようにその要素の名前の頭文字と数字の割り振りによって一意に定められる。また、ゴール構造表記法は、車載組込みシステムの安全目標や機能安全要件の定義の際などにも利用できる [8]。本研究では機械学習システムの信頼性を議論するためにゴール構造表記法を利用している。また、本研究ではゴール構造表記法によって表される図を GSN 図と表現する。

表 1. ゴール構造表記法の要素の説明

名前	説明
Goal	システムが満たすべき特定の主張
Strategy	Goal を分解するためのアプローチや方法論
Solution	Strategy に基づく具体的な証拠
Context	Goal や Strategy の状況や前提条件
Justification	Goal や Strategy の妥当性を支持する論理や理由
Assumption	Goal が依存する仮定
InContextOf	Goal、Strategy、または Solution が適用される状況や条件へのリンク
SupportedBy	Goal や Strategy を裏付ける関連要素や証拠へのリンク

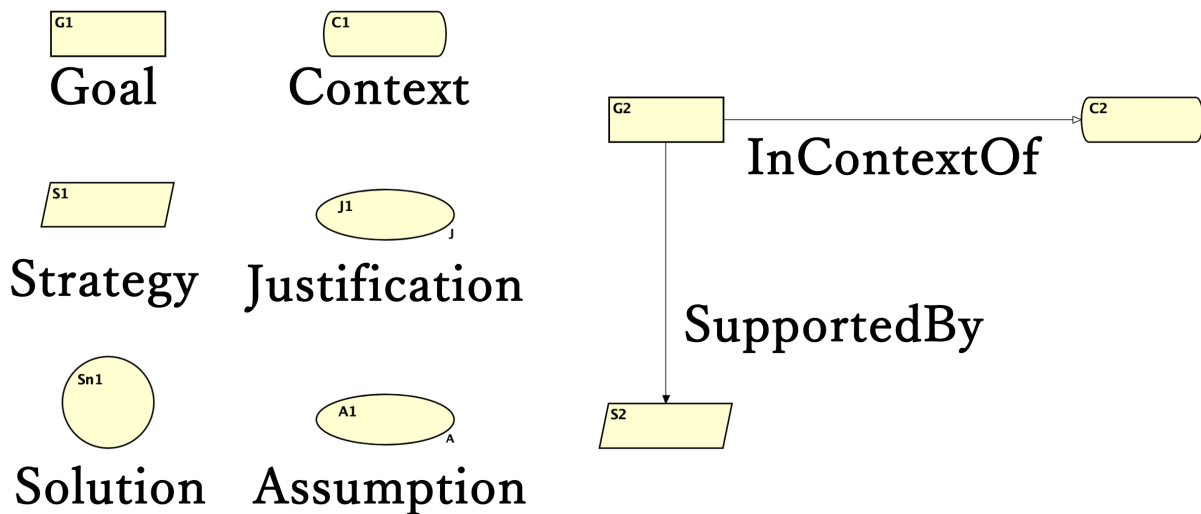


図 1. ゴール構造表記法の要素

2.2 astah* System Safety

astah* System Safety はシステム分析などを支援するツールである [9]。astah* System Safety の利用によりエンジニアはより効率的にシステム分析などができる。また、astah* System Safety では 2.1 で前述のゴール構造表記法を記述できる。本研究ではゴール構造表記法を astah* System Safety 上で扱う。次に astah* System Safety の UI を図2に示す。例えば、図2の右上にあるウィンドウでゴール構造表記法の各要素を操作できる。本研究では astah* System Safety 向けのプラグインの開発を行なった。例えば、図2の右下にある拡張タブの部分を開発した。拡張タブの新規追加により、ユーザからの入力を可能としている。

2.3 機械学習信頼性ソリューションパターン

我々の先行研究で、機械学習の議論におけるパターンの機械学習信頼性ソリューションパターンを特定した。表2に機械学習信頼性ソリューションパターンをそれぞれ示す。これらのパターンは設計段階からの検討が必要であり、システムの信頼性の保証においても特に重要なパターンである。また、機械学習システムの要求や価値との照らし合わせも必要とされ、訓練結果などに依存する側面もある。よって、これらの

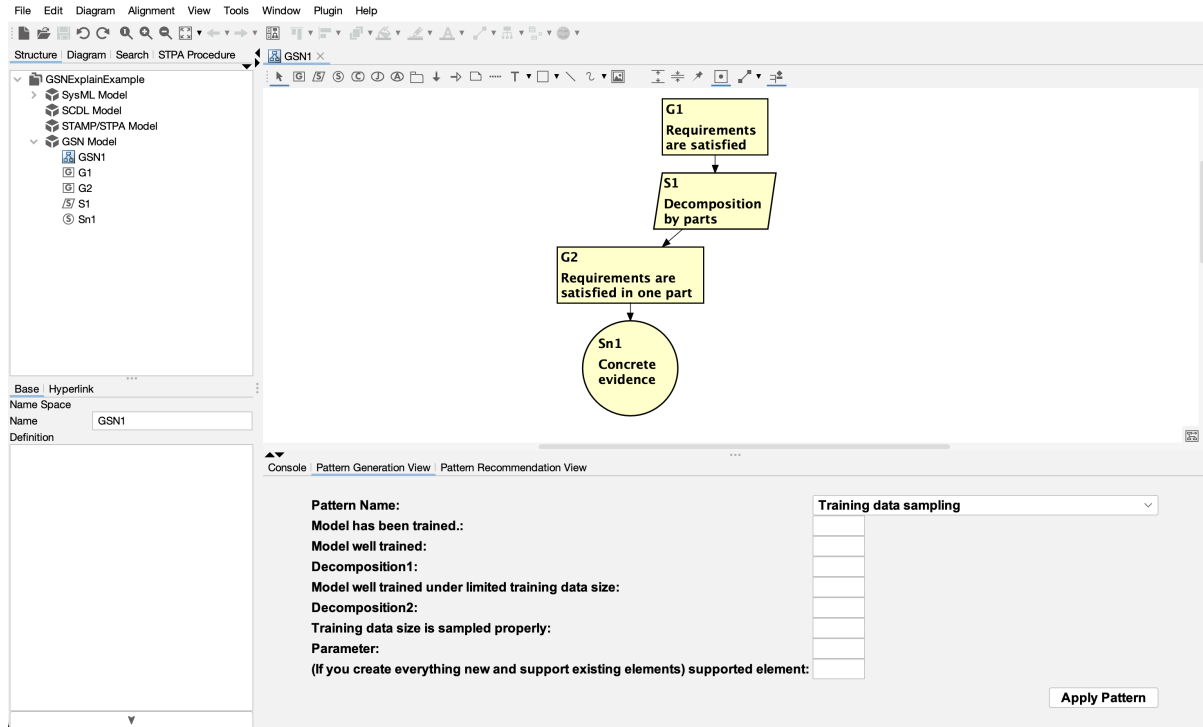


図 2. astah* System Safety の UI

パターンは機械学習独自の特性を反映しているため、本研究ではこれらのパターンの適用について考える。さらに、これらのパターンを GSN 図を利用して構造的に表すことによって、段階的に議論の論証ができる。これによって、機械学習システムの信頼性の保証という問題を部分問題に分けて、信頼性を保証できる部分を第三者へも明確にできる。そして、パターンであるためこれらの論証を再利用できる。例えば、Training data sampling パターンを GSN 図で表すと図3となる。まず、最上位の Goal として「Model well trained」が存在する。そして、このパターンではモデルはすでに訓練されている前提があるため、「Model has been trained.」の Context が存在し、InContextOf のリンクで繋がっている。そして、最上位のゴールの下には「Decomposition1」の Strategy があり、これは SupportedBy のリンクで繋がっている。その下の Goal として「Model well trained under limited training data size」が存在する。さらにその下には「Decomposition2」の Strategy があり、その下の Goal に「Training data size is sampled properly」が連結している。最後に Solution として「Parameter」があり、それぞれのプロジェクトにけるパラメータの値の充足により Goal の達成を示す。

3 パターン適用支援の枠組みの提案

本章では機械学習信頼性ソリューションパターンの適用時に人為的な誤りが起こる問題を解決するために、パターン適用支援の枠組みの提案を行ない、システム全体の構成を説明する。この手法では、ユーザがパターンを選択し、パターンに合致している部分を入力する。そして、入力されなかった部分の新規作成により、パターンを自動適用する。これによりパターン適用時における人為的な誤りの発生を防ぐことができる。具体的な仕組みとして、図4に示す本研究のシステムの概要図について説明する。オレンジの枠で囲

表 2. 機械学習信頼性ソリューションパターンの一部

パターン名	問題	解決
Selective repair[10]	欠陥が発見された場合に、原因を分析し対策を検討できない	機械学習修正ツール
Adversarial example defense[11]	セキュリティ攻撃による制御などの悪意のあるユーザーへ対策する必要がある	敵対的トレーニング
Reprioritize accuracy[10]	重要な部分の認識を保証できず、新しい学習データは認識率を下げる可能性がある	機械学習修正ツール
Training data sampling[12]	信頼性を確保するために必要なトレーニングデータとテストデータが多い	データサンプリング法
Model smoke testing[13]	システム変更時の信頼性の再検証と再試験のコストが高い	スモークテスト
Safeguard[14]	システムが保証範囲内で安全にシャットダウンされる保証ができない	ルールベースによるセーフティガード
Security requirement satisfaction argument[15]	セキュリティ要求はデータとモデル駆動の方法で評価する必要がある	テストデータや形式的な検証

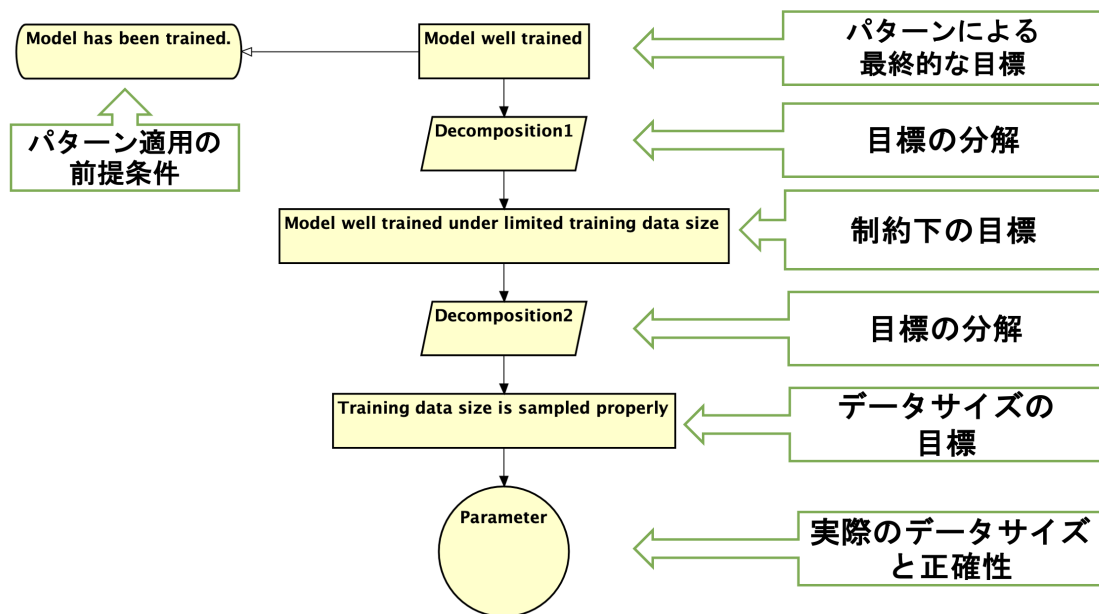


図 3. Training data sampling パターンの GSN 図

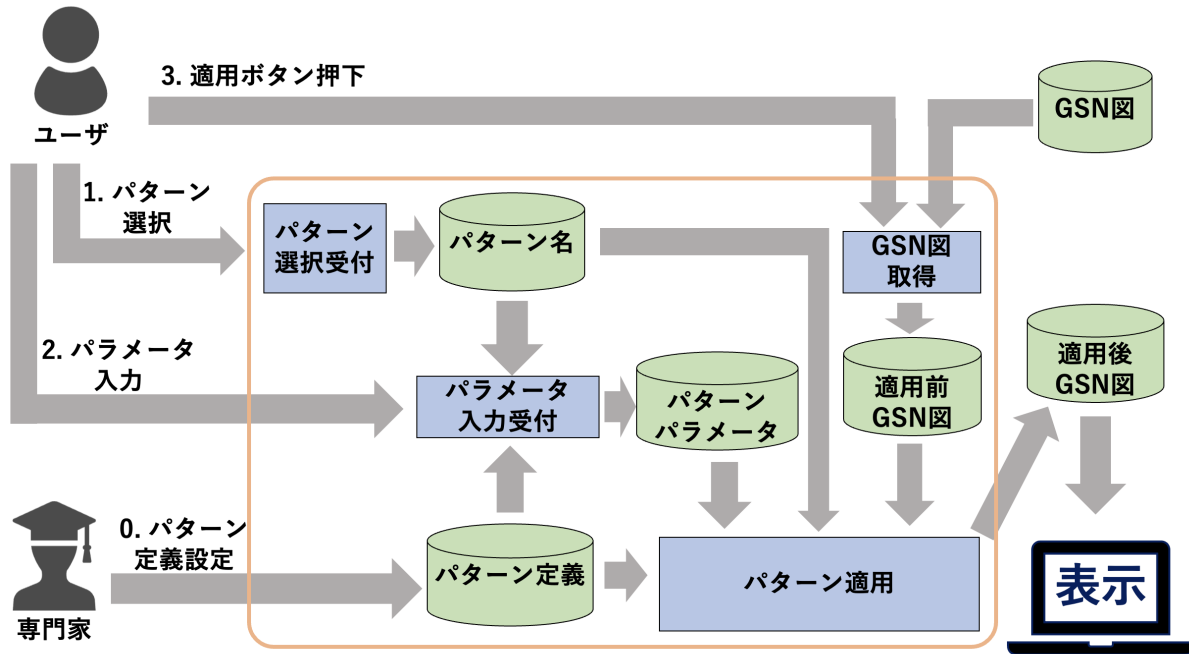


図4. データの流れに着目したシステムの概要図

まれている部分が本研究で開発したシステムの概要図である。このシステムは大きく以下の4つのステップからなる。

ステップ0. パターン定義設定: 専門家がパターンの定義を設定し、パターン定義としてそのデータが格納される。今回は機械学習信頼性ソリューションパターンの定義が格納されている。

ステップ1. パターン選択: ユーザがパターンを選択し、システムは選択されたパターンを受け付け、そのパターン名が格納される。

ステップ2. パラメータ入力: パターン名とパターン定義を参照することで、ユーザからの必要なパラメータの入力を受け付ける。このデータはパターンパラメータとして格納される。

ステップ3. 適用ボタン押下: ユーザが適用ボタンを押下した時に、astah* System Safety 上で事前にユーザによって作られた GSN 図を取得し、適用前 GSN 図としてデータを格納する。そして、パターン定義、パターン名、パターンパラメータ、適用前 GSN 図を参照し、必要な要素を新規作成する。適用後 GSN 図としてデータが出力され、astah* System Safety 上に表示される。

以下では具体的な利用例を述べる。ここでは仮に図5のモデルにおいて、Training data sampling パターンを適用する場合を考える。まず、図6のようにエンジニアがパターンを指定する。今回は Training data sampling パターンを選択する。そして、図7のようにそれぞれのパターンで対応する作成中のモデルを入力する。今回は Model well trained が G1、Decomposition1 が S1 と対応すると仮定する。最後に、Apply Pattern と書かれている適用ボタンの押下により自動的に図の中に必要なモデルを作成し、作成中の図と統合させる。今回の場合は図8のようになる。これにより図に必要な Goal や Solution、Contextなどをパターンと合致する形で自動的に適用できる。また、作成中の図と統合できるため、効率的にパターンを適用できる。

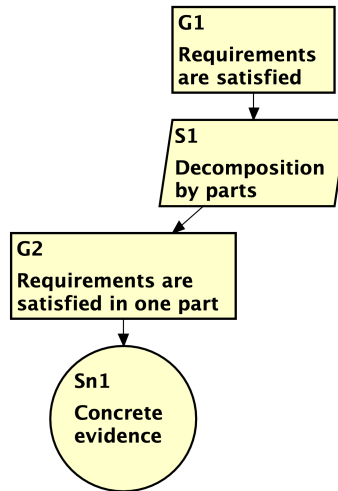


図 5. GSN 図の例

Pattern Name:

Model has been trained.:

Model well trained:

Decomposition1:

Model well trained under limited training data size:

Decomposition2:

Training data size is sampled properly:

Parameter:

(If you create everything new and support existing elements) supported element:

Training data sampling

Selective repair

Adversarial example defense

Reprioritize accuracy

Training data sampling

Model smoke testing

Safeguard pattern

Security requirement satisfaction argument pattern

図 6. パターン選択の UI

Pattern Name:

Model has been trained.:

Model well trained:

Decomposition1:

Model well trained under limited training data size:

Decomposition2:

Training data size is sampled properly:

Parameter:

(If you create everything new and support existing elements) supported element:

Training data sampling

G1

S1

図 7. 要素の選択の UI

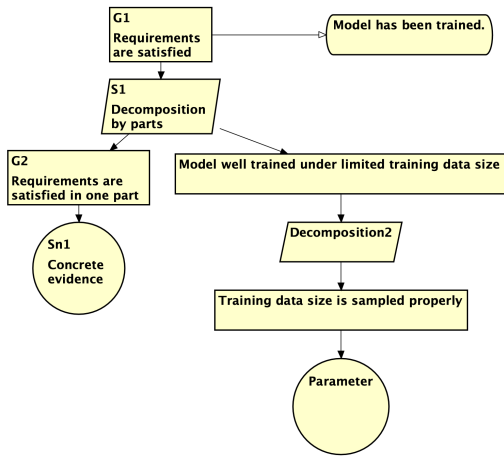


図 8. Training data sampling パターン適用後の GSN 図の例

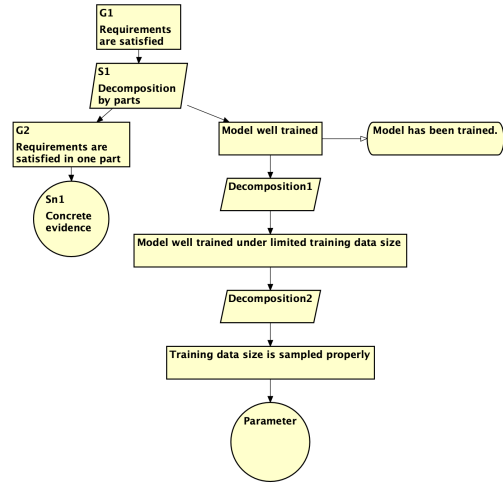


図 9. 全て新規作成した場合の Training data sampling パターン適用の GSN 図の例

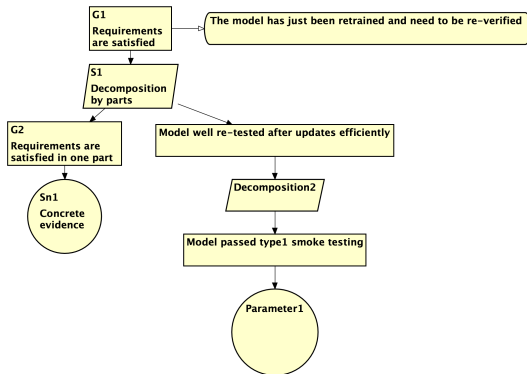


図 10. number of class が 1 の場合の Model Smoke Testing パターンの適用

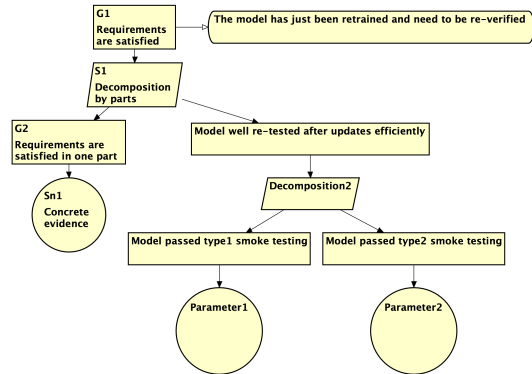


図 11. number of class が 2 の場合の Model Smoke Testing パターンの適用

次に特殊な場合として、パターンに合致する部分が存在せず、全てを新規作成する場合について述べる。始めに、パターンに合致する部分が存在せず、全てを新規作成する場合について考える。この時に既存の要素を SupportedBy で繋げる必要がある場合、図7の (If you create everything new and support existing elements) supported element の欄に既存の要素を入力する。例えば、図5の場合で Training data sampling パターンの要素を全て新規作成し、既存の要素の S1 を SupportedBy で繋げる場合を考える。この場合は supported element の欄に S1 を入力し、適用ボタンの押下により、図9のように適用できる。

また、機械学習システムの特有の問題として追加で変数が必要な場合を考える。本研究では機械学習信頼性ソリューションパターンを扱うため、クラスの数に応じて要素の数が動的に変わるパターンが存在する。よって、パターンによっては追加の変数が必要となる場合がある。具体的には Selective repair パターン、Reprioritize accuracy パターン、Model Smoke Testing パターンが該当する。例えば、Model Smoke Testing

パターンの場合、number of class の欄があり、その値に応じて要素の数が変わる。例えば、number of class の欄に 1 を入力した場合は図10となり、2 を入力した場合は図11となる。Model Smoke Testing パターンの場合はテストのタイプの数に応じて、要素の数を増加できる。このように変数の設定により機械学習システム特有の要素の数が動的に変わるパターンにも対応している。

4 ケーススタディ

本章ではケーススタディにより本研究の手法の実用性を確認する。ケーススタディとして、図12の交通標識の分類における機械学習システムの GSN 図を利用する。ここで、信頼性を確保するために大量のトレーニングデータとテストデータが必要な場合を考える。そして、機械学習モデルは学習済みである状況と大規模なトレーニングデータが利用できないという制約があるとする。そこで、機械学習の専門家ではないエンジ

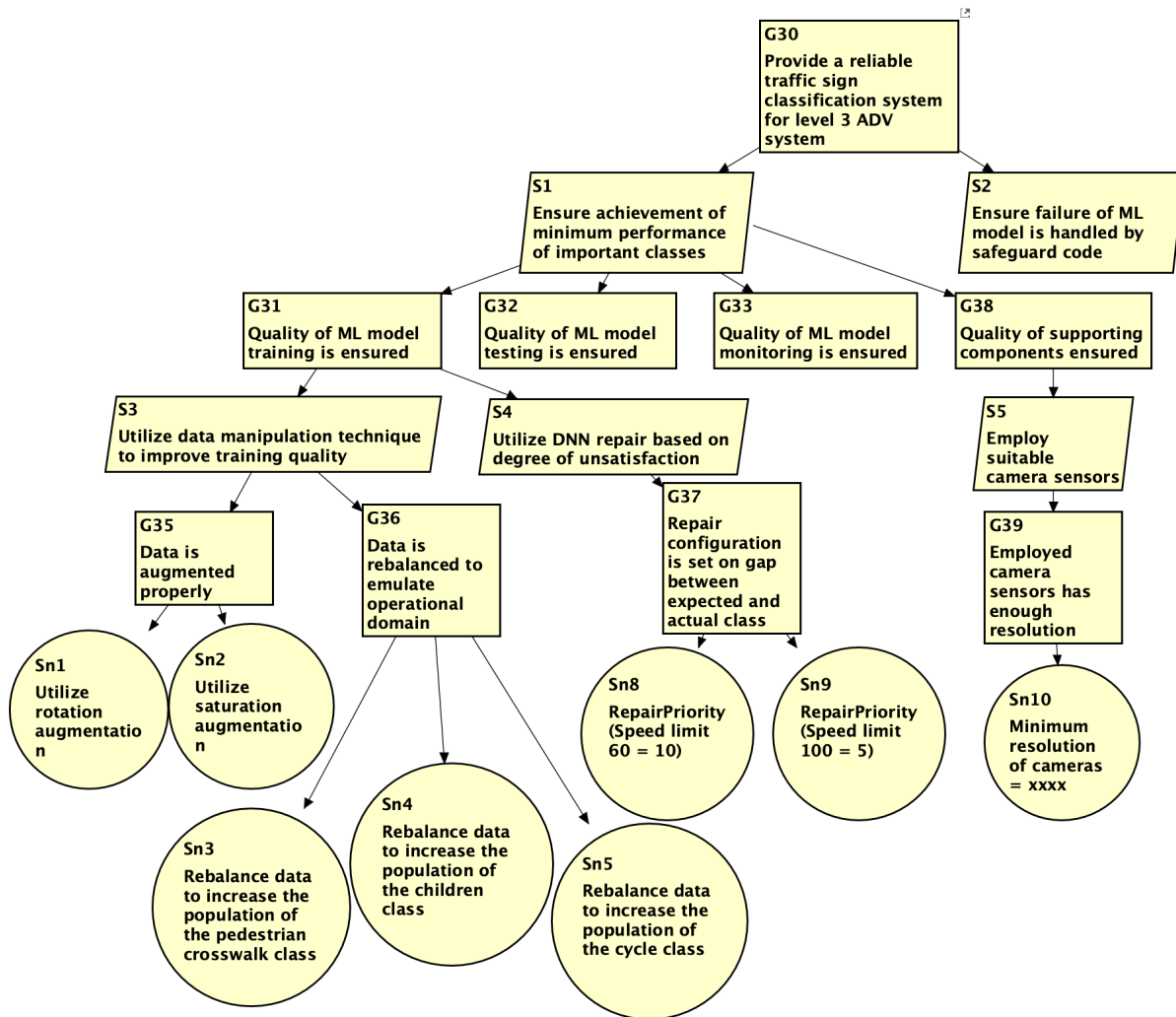


図 12. ケーススタディ

ニアが図12の GSN 図において、Training data sampling パターンを適用する場合を考える。エンジニアは機械学習の専門家ではないため、正確なパターン適用ができない可能性がある。さらに、手動でパターンを適用する場合は人為的な誤りが発生する問題がある。そこで、本研究のプラグインを利用することでこの問題を解決する。今回の場合は図13のように入力する。まず、パターンとして Training data sampling パターンを選択する。次にパラメータの入力を行う。今回の場合、G31 の Statement は「Quality of ML model training is ensured」であり、このパターンの Model well trained の Goal に合致し、G31 の Statement は「Utilize data

Pattern Name:	Training data sampling
Model has been trained.:	
Model well trained:	G31
Decomposition1:	S3
Model well trained under limited training data size:	
Decomposition2:	
Training data size is sampled properly:	
Parameter:	
(If you create everything new and support existing elements) supported element:	

Apply Pattern

図 13. Training Data Sampling パターンの入力

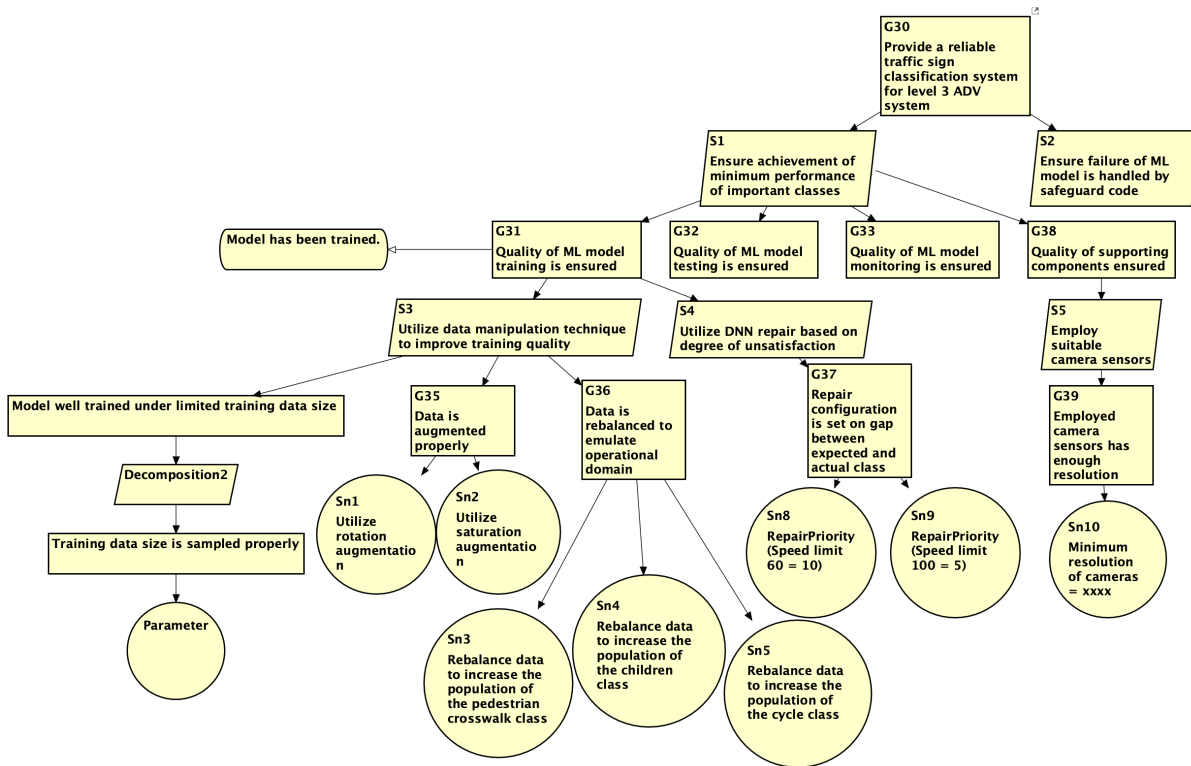


図 14. ケーススタディにおける Training Data Sampling パターンの適用

manipulation technique to improve training quality」であり、このパターンの Decomposition1 の部分に合致する。その他の部分は新規作成する必要があるため、このパターンの Model well trained にあたる部分は G31 と入力し、Decomposition1 にあたる部分は S3 と入力し、それ以外の部分は入力を行わない。この状態で適用ボタンを押すと図14のように Training data sampling パターンが適用される。これにより、機械学習の専門家でないエンジニアでも専門家の知見を利用し、正確なパターン適用ができた。また、自動的にパターン適用されるため人為的な誤りが発生する問題を解決できた。以上より、ケーススタディにおいてパターン適用の実用性が確認された。

5 考察

本研究のパターン適用により人為的な誤りを減少できると考えられるが、さらなる工夫も必要であると考ええる。まず、パターンの自動的な適用によって人為的な誤りを防止できるため、人為的な誤りなしにパターンを適用できるようになったと考えられる。例えば SupportedBy のリンクが意図しない要素と結合しないため、パターンを正しく適用できる。しかし、人為的な誤りが発生する可能性は残っている。例えば、要素の値を入力する際にスペルミスや見間違いによって、意図しない部分にパターンが適用される可能性が残っている。これを改善するための方法として大きく2点の対策が考えられる。1点目はUIの改善である。スペルミスなどが起こらないようにするために、ドロップダウンリストによる要素の入力が考えられる。また、現在は図13のようにパターンの要素の文章を元にエンジニアが入力しているが、元になるパターンの図の利用も考えられる。2点目はパターン適用後の検知をするツールの開発である。パターンが正しく適用できていないと考えられる場合、警告表示により誤りを防止できると考えられる。

次に本研究の手法における制限は、専門家の必要性と事前知識の必要性の2点が挙げられる。1点目の専門家の必要性について、本研究の手法では既知のパターンを設定する必要があるため専門家の知識は不可欠である。そして、この自動化は困難であると考えられるため、利用できるパターンには限界があると考えられる。2点目の事前知識の必要性について、本研究の手法では最初にエンジニアがパターンを選択する必要があるため、エンジニアがパターンに関する事前知識がない場合は本研究の手法では不十分である。よって、パターンの選択支援ツールを開発する必要がある。

パターン選択支援ツールとしては、グラフパターンマッチングを利用する手法と大規模言語モデルの GraphPrompts を利用する手法の2点が考えられる [16][17]。1点目のグラフパターンマッチングでは予めパターンを推薦するための条件を専門家によって定義し、その条件と合致した場合にパターンを推薦する。例えば、Training data sampling パターンを推薦する場合を考える。この場合は条件として、ある GSN の要素が Goal であり、その中の文章に model と train という単語が含まれている場合と定義する。この時、ケーススタディの図12では G31 の「Quality of ML model training is ensured」が条件と合致する。そのため、Training data sampling パターンを推薦することができる。2点目の GraphPrompts は大規模言語モデルでグラフ構造を扱う際のパターンである。専門家により、パターンの情報を含めたプロンプトを定義し、合致するパターンを出力させることでパターンを推薦する。ただし、大規模言語モデルではハルシネーションが発生する可能性を考慮する必要がある [18]。以上のようにグラフパターンマッチングや GraphPrompts の手法を利用することで、パターン選択支援ツールを開発できる。これにより、事前知識がない場合のパターン適用が期待される。

6 関連研究

Husen ら [19] の研究では機械学習システム開発中に異なるビューやモデルを参照する必要があるという問題を解決するために、Multi-View Modeling Framework for ML Systems (M³S) を提案している。この研究では本研究と同様に、機械学習システムの設計に焦点を当て、astah* System Safety によるプラグインを開発している。M³S により、GSN 図や SysML 図、STAMP/STPA 図などの異なるモデルを関連づけることができる。そして、この研究では AI Project Canvas や ML Canvas を開発している点が効果的であると感じた。本研究との違いとして、この研究では異なるモデルとの整合を目的にしているため、パターン適用は主な

目的としていない点がある。また、M³S プロセス内の Safety や Safety Case の部分に本研究を位置付けることができる。

次に Lakshmanan ら [20] は、機械学習におけるデザインパターンをカタログとしてまとめている。この本では機械学習システム全体の開発過程に関して、それぞれの過程で利用できるデザインパターンを 30 個にまとめている。さらにパターン間のつながりに関しても言及しており、デザインパターンを利用しやすい形で整理されているため有用であると感じた。ただし、この本はデザインパターンの体系化を目的としているが、本研究ではパターンの適用やツールの開発を目的としている点が異なる。また、この本では機械学習システムの一般的な問題と解決策を述べているが、本研究では GSN 図を利用している点が異なっている。

Shiroma ら [21] の研究では本研究と同様にパターン適用に関して論じている。デザインパターンを適用する際にパターンと対応する部分の指定により、デザインパターンを適用できる。さらに、デザインパターンの検知をする機能も開発している。この研究では、複数の関連性のあるデザインパターンを適用する際に、UML のステレオタイプによって印をつけ、それらの参照による正確な複数のデザインパターンの適用が特に優れている。ただし、この研究ではモデル駆動開発におけるセキュリティのデザインパターンを扱っており、機械学習のパターンは扱っていない。よって、本研究とは対象としている領域が異なっている。また、この研究では GSN 図ではなく、UML 図を利用している相違点がある。

次に、Yu ら [22] によると、ゴールモデルにおけるセキュリティパターンに関して、ツール開発によりパターンの検出と解決を自動化している。この研究では要求工学におけるモデル駆動開発において、セキュリティパターンを適用するための ATL を利用したツールの開発を行っている。特にモデルのデザインパターンとの適合検出機能が有用であると感じた。本研究との違いは、この研究ではモデル駆動開発のセキュリティデザインパターンを扱っており、機械学習のパターンではない点である。

7 おわりに

本研究では機械学習信頼性ソリューションパターンの適用支援の枠組みの提案とツールの実装を行い、ケーススタディによる検証を行なった。具体的には、パターン適用を自動的に行うための *astah* System Safety* におけるプラグインの実装を行なった。そして、ケーススタディにより本研究の手法の実用性を確認した。本研究のパターン適用により、エンジニアが機械学習の専門家ではない場合でも正確なパターン適用が期待される。以上より、機械学習信頼性ソリューションパターンの適用時に人為的な誤りが起こる問題に対して、パターン適用支援の枠組みと回答できたと考える。

今後の展望としてはパターンの選択支援ツールの開発、本研究の手法の改善、評価実験の実施の 3 点が挙げられる。1 点目のパターンの選択支援ツールの開発では、パターン推薦のツールを開発する。具体的にはグラフパターンマッチングと大規模言語モデルの *GraphPrompts* を利用する。グラフ構造を考慮し、類似するパターンの推薦により、先行知識なしでパターンの適用ができる。2 点目の本研究の手法の改善については、UI の改善やパターンが正しく適用できていない場合の警告表示がある。3 点目の評価実験の実施に関しては本研究のツールを利用した場合と利用しなかった場合におけるパターン適用の効率性、正確性の比較により効果を確認する。

謝辞

本論文のシェパードをご担当いただいた羽生田栄一氏と *Writer's Workshop* にてコメントをくださった方々に感謝の意を表する。

REFERENCES

- [1] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, Vol. 349, No. 6245, pp. 255–260, 2015.
- [2] Jack Stilgoe. Machine learning, social learning and the governance of self-driving cars. *Social studies of science*, Vol. 48, No. 1, pp. 25–56, 2018.

- [3] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, Vol. 23, No. 1, pp. 89–109, 2001.
- [4] Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process. In *Proceedings of the 44th International Conference on Software Engineering*, pp. 413–425, 2022.
- [5] The Assurance Case Working Group. Goal structuring notation community standard version 3. <https://scsc.uk/r141C:1?t=1>. (Accessed on 01/06/2024).
- [6] Christopher Alexander. *A pattern language: towns, buildings, construction*. Oxford university press, 1977.
- [7] Richard Johnson John Gamma, Erich Helm and Ralph Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series. Pearson Education, 1994.
- [8] International Standardization Organization. Iso 26262-1:2018(en) road vehicles — functional safety — part 1: Vocabulary. <https://www.iso.org/obp/ui/#iso:std:iso:26262:-1:ed-2:v1:en>. (Accessed on 01/06/2024).
- [9] Change Vision. astah* system safety. <https://astah.net/products/astah-system-safety/>. (Accessed on 01/06/2024).
- [10] Jeongju Sohn, Sungmin Kang, and Shin Yoo. Arachne: Search-based repair of deep neural networks. *ACM Transactions on Software Engineering and Methodology*, Vol. 32, No. 4, pp. 1–26, 2023.
- [11] Mingfu Xue, Chengxiang Yuan, Heyi Wu, Yushu Zhang, and Weiqiang Liu. Machine learning security: Threats, countermeasures, and evaluations. *IEEE Access*, Vol. 8, pp. 74720–74742, 2020.
- [12] Maya Kabkab, Azadeh Alavi, and Rama Chellappa. Dcnns on a diet: Sampling strategies for reducing the training set size. *arXiv preprint arXiv:1606.04232*, 2016.
- [13] Steffen Herbold and Tobias Haar. Smoke testing for machine learning: simple tests to discover severe bugs. *Empirical Software Engineering*, Vol. 27, No. 2, p. 45, 2022.
- [14] Hironori Washizaki, Foutse Khomh, Yann-Gaël Guéhéneuc, Hironori Takeuchi, Satoshi Okuda, Naotake Natori, and Naohisa Shioura. Software engineering patterns for machine learning applications (sep4mla) part 2. In *Proceedings of the 27th conference on pattern languages of programs*, pp. 1–10, 2020.
- [15] Marwa Zeroual, Brahim Hamid, Morayo Adedjouma, and Jason Jaskolka. Security argument patterns for deep neural network development. https://drive.google.com/file/d/1j1FreLDMqygleJR30iVX_kv3HSJdxYh7/view. (Accessed on 01/06/2024).
- [16] Rui Qiao, Xiaolei Zhong, Ling Zhang, and Heng He. Graph pattern matching through model checking. In *2015 8th International Conference on Database Theory and Application (DTA)*, pp. 1–5. IEEE, 2015.
- [17] Elvis Saravia. Graphprompts. <https://www.promptingguide.ai/techniques/graph>. (Accessed on 02/09/2024).
- [18] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, Vol. 55, No. 12, pp. 1–38, 2023.
- [19] Jati H Husen, Hironori Washizaki, Nobukazu Yoshioka, Hnin Thandar Tun, Yoshiaki Fukazawa, and Hironori Takeuchi. Metamodel-based multi-view modeling framework for machine learning systems. In *MODELSWARD*, pp. 194–201, 2023.
- [20] Valliappa Lakshmanan, Sara Robinson, and Michael Munn. *Machine learning design patterns*. O'Reilly Media, 2020.
- [21] Yuki Shiroma, Hironori Washizaki, Yoshiaki Fukazawa, Atsuto Kubo, and Nobukazu Yoshioka. Model-driven security patterns application based on dependences among patterns. In *2010 International conference on availability, reliability and security*, pp. 555–559. IEEE, 2010.
- [22] Yijun Yu, Haruhiko Kaiya, Nobukazu Yoshioka, Zhenjiang Hu, Hironori Washizaki, Yingfei Xiong, and Amin Hosseinian-Far. Goal modelling for security problem matching and pattern enforcement. *International Journal of Secure Software Engineering (IJSSE)*, Vol. 8, No. 3, pp. 42–57, 2017.