

LLMを用いた 量子機械学習のための自動回路生成

2025年5月8日 Quantum CAE研究会

大阪大学 量子情報・量子生命研究センター (QIQB) 作花健也

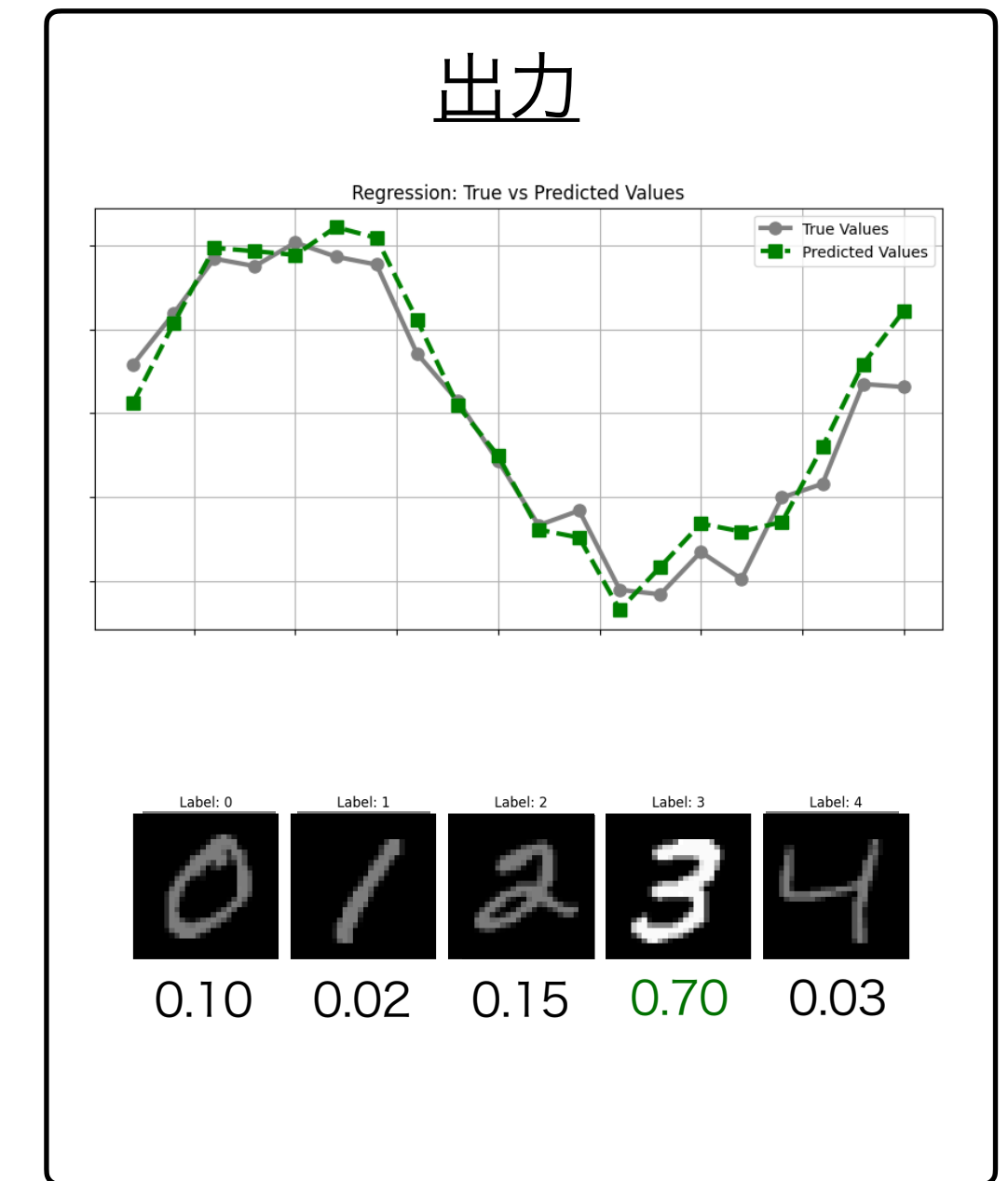
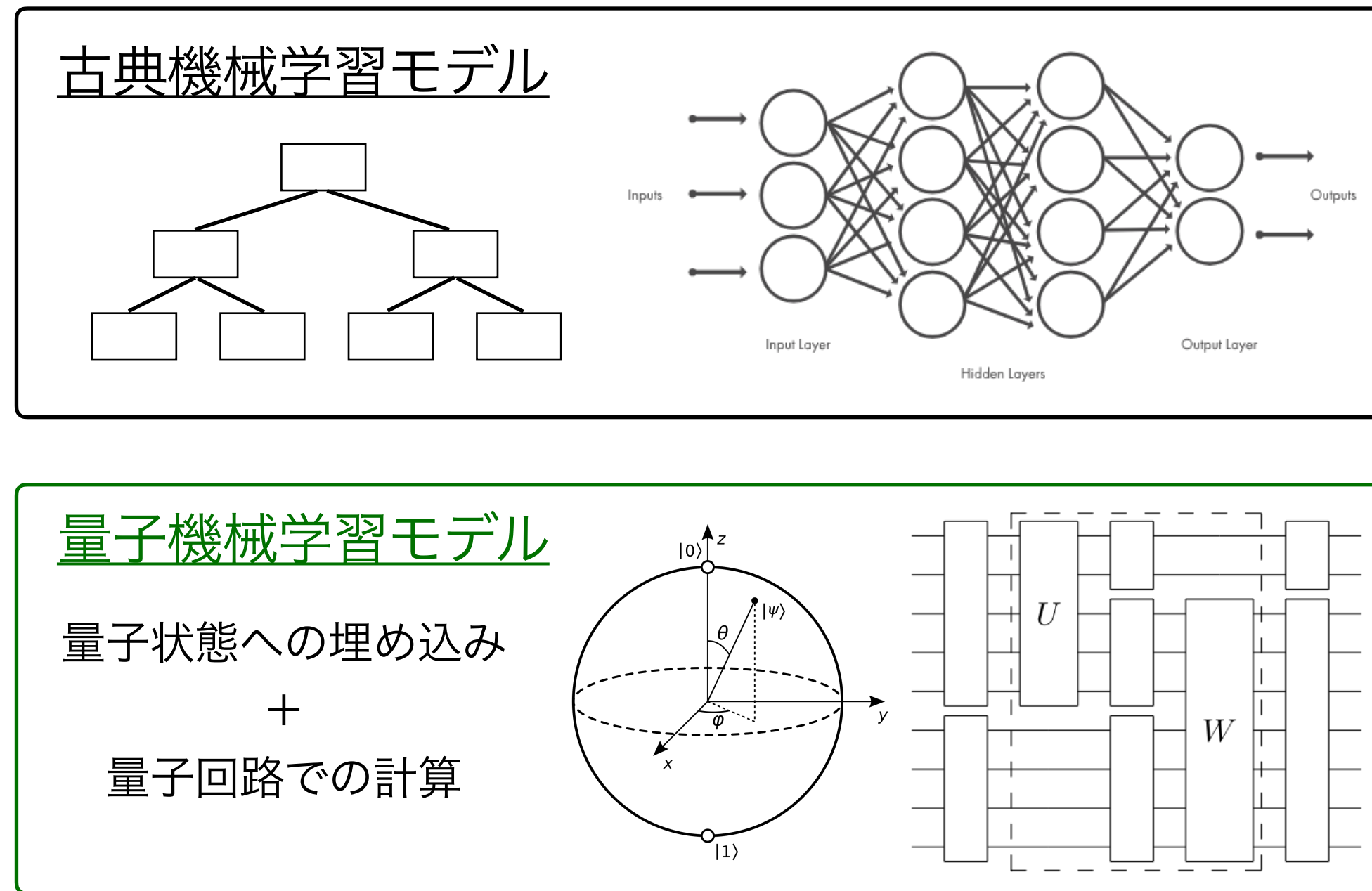
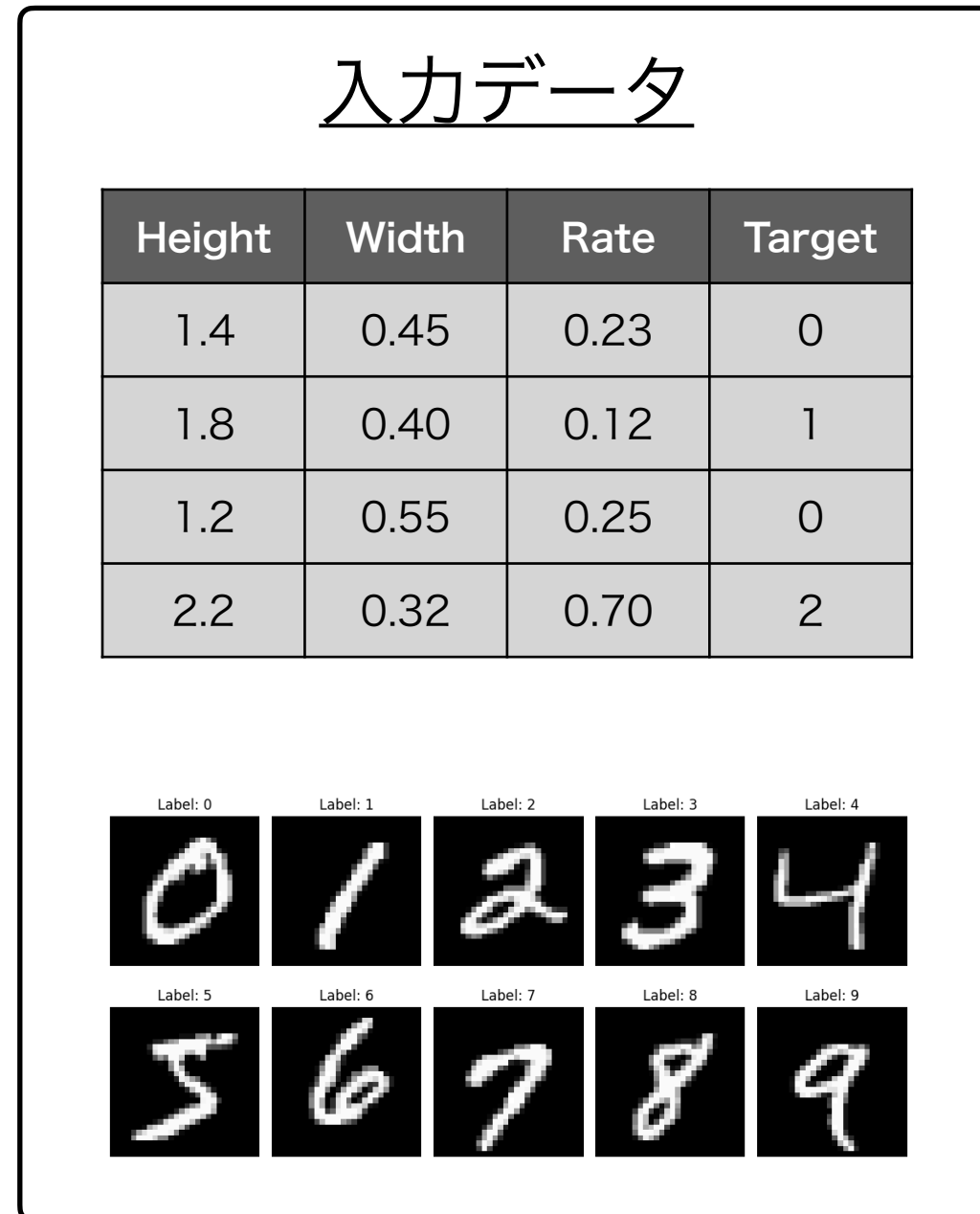
目次

- ・ 量子特徴マップと機械学習
- ・ 量子特徴マップの自動生成に向けて
- ・ 「Automating quantum feature map design via large language models」の論文紹介
- ・ まとめと今後の展望

目次

- 量子特徴マップと機械学習
- 量子特徴マップの自動生成に向けて
- 「Automating quantum feature map design via large language models」の論文紹介
- まとめと今後の展望

量子機械学習とは



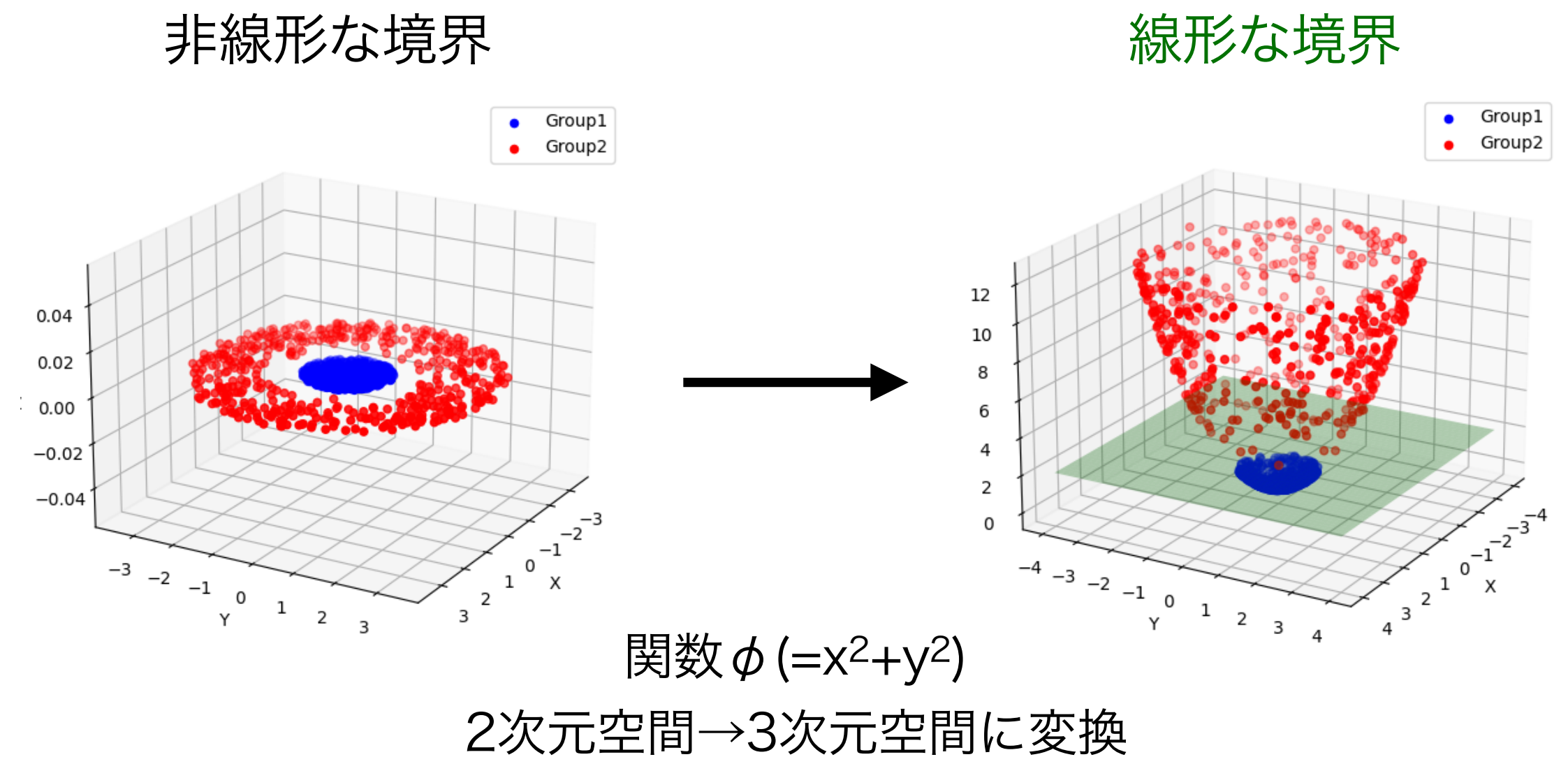
NISQで量子機械学習を行う際の課題

- ・ 計算過程で無視できないほど大きなノイズの影響あり
 - ・ シミュレータを含めても利用可能なqubit数に制限あり
- 現時点では古典機械学習で主流となってきた大規模なモデルを量子回路で構築・実行することは難しい

量子特徴マッピングの概要

特徴マッピング

- ・関数を用いて入力データをより高次元空間の点として変換するための手法
- ・適切な変換を行うことで非線形な境界を変換後の空間では線形な超平面で分離可能
- ・データセットの特性を考慮した変換方法の選定が重要



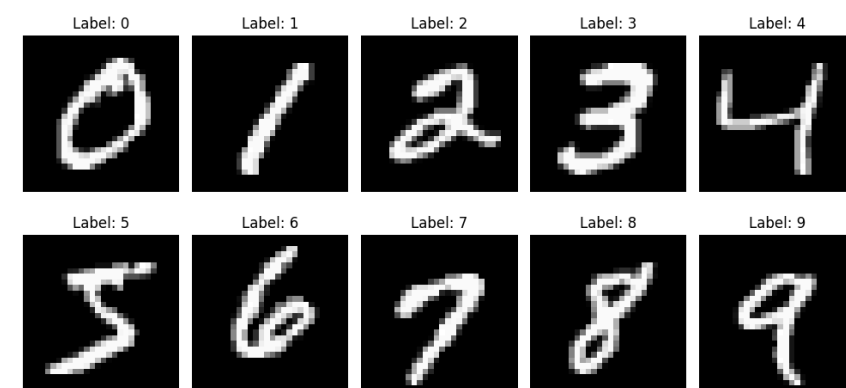
量子特徴マッピング

- ・古典データを量子コンピュータで処理可能な量子状態に変換するための手法
 - ・例) 入力データの値をRotation Gateの回転角として入力
- ・量子の性質 (重ね合わせ、エンタングルメント)を生かし、適切な変換を行うことで**古典コンピュータでは表現が困難な空間を実現できることが期待**

量子特徴マップの機械学習への応用

入力データ

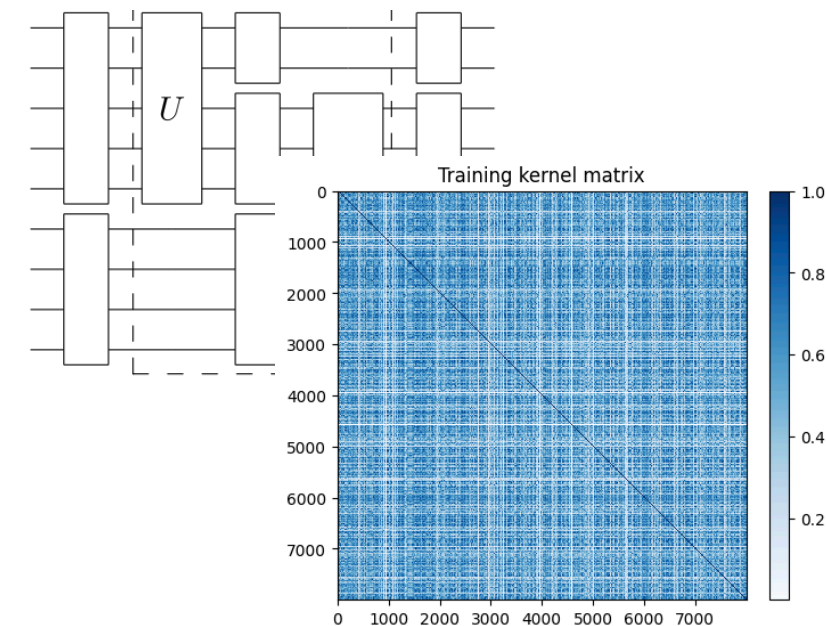
Height	Width	Rate	Target
1.4	0.45	0.23	0
1.8	0.40	0.12	1
1.2	0.55	0.25	0
2.2	0.32	0.70	2



機械学習モデル

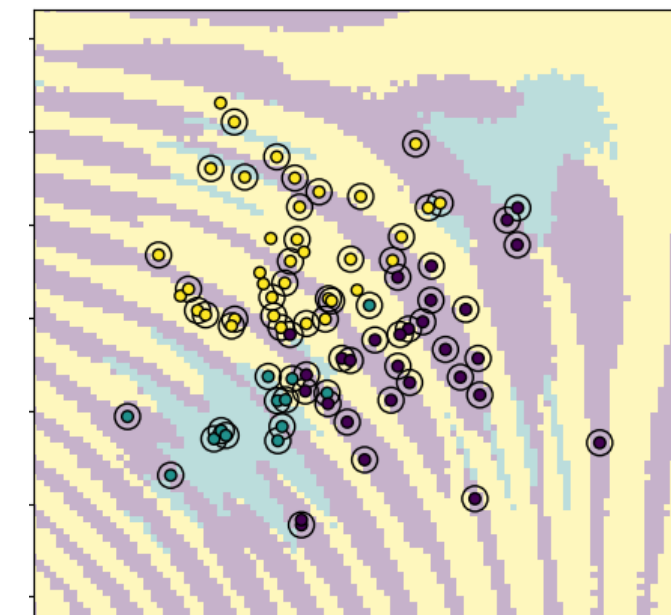
量子カーネル

量子回路で量子状態に変換し、カーネルを計算

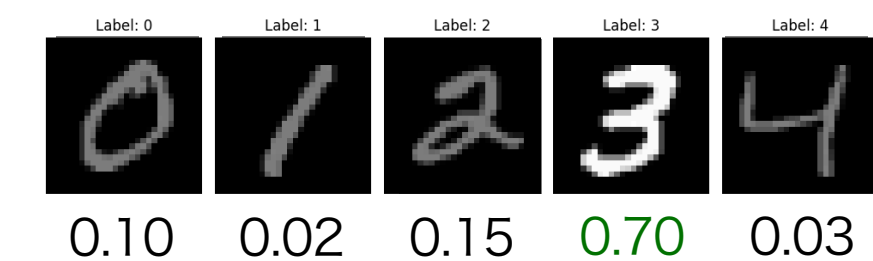
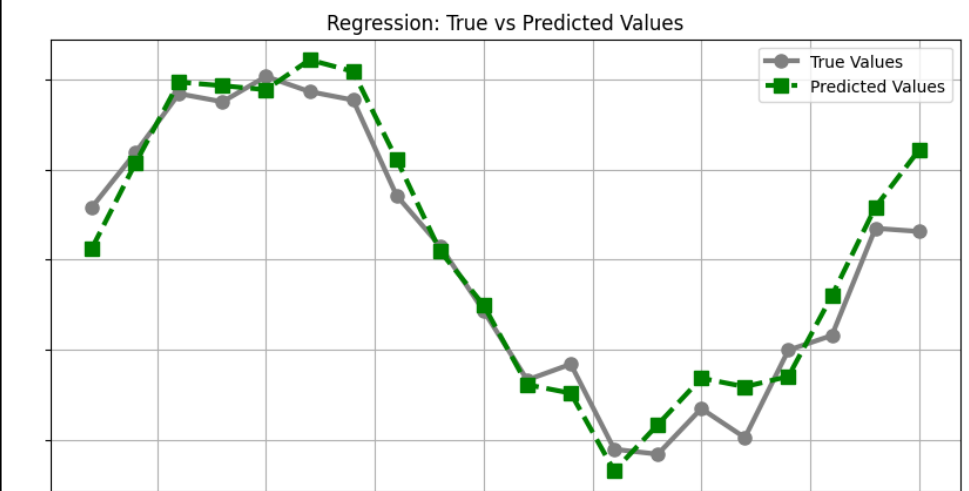


古典機械学習モデル

量子カーネルの値を利用し、モデルの学習は古典コンピュータで実施



出力



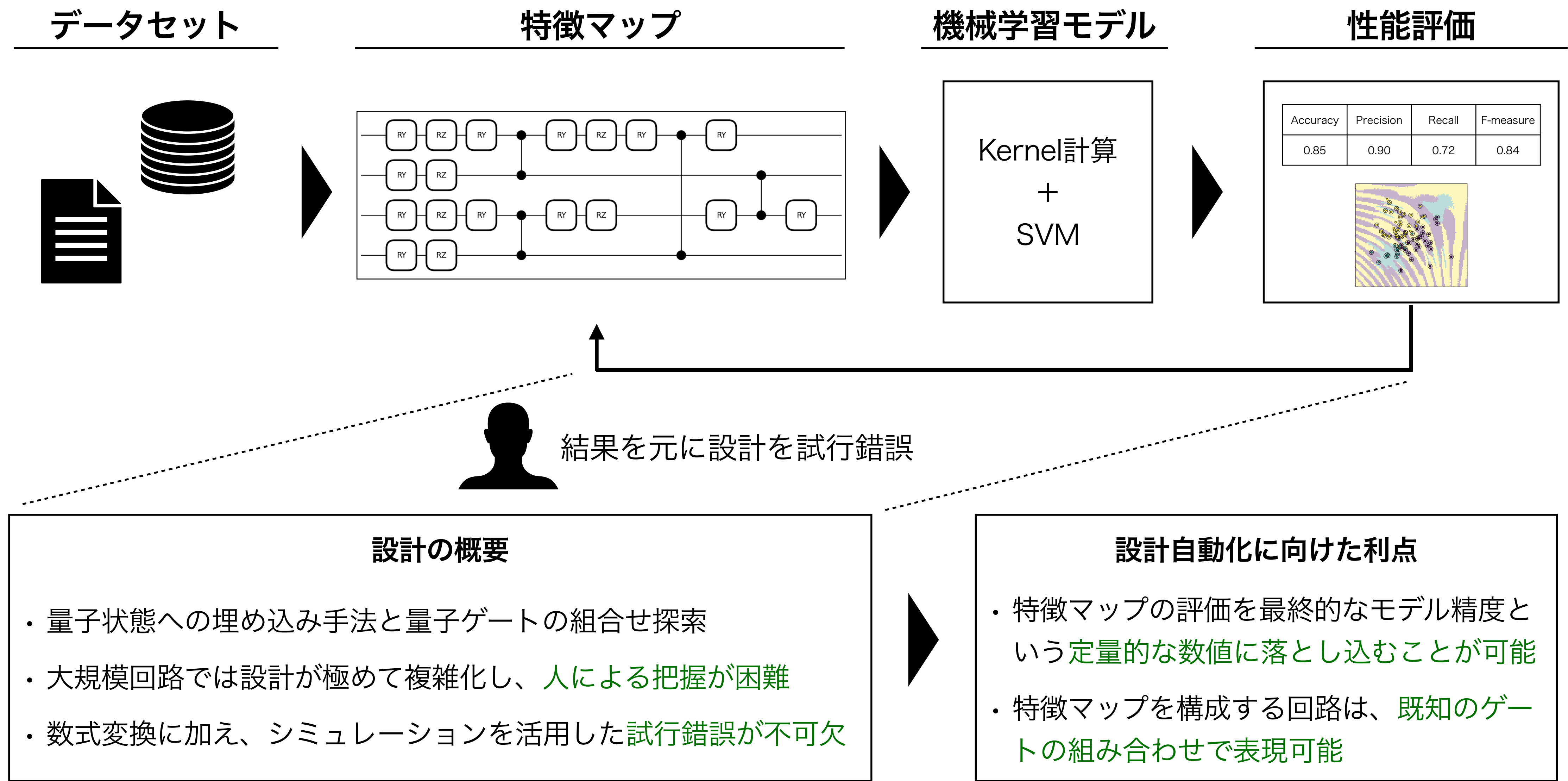
量子カーネルベースの機械学習

- ・量子特徴マップで変換された値の内積をカーネルとして利用可能
 - ・カーネル計算以外の機械学習ロジックは古典コンピュータで計算可能
- 回帰・分類ともにQSVM等の量子機械学習モデルが提案・検証されてきている

目次

- 量子特徴マップと機械学習
- 量子特徴マップの自動生成に向けて
- 「Automating quantum feature map design via large language models」の論文紹介
- まとめと今後の展望

人手による量子特徴マップの設計



© 2011 Pearson Education, Inc. All rights reserved. Printed in the United States of America. This publication is protected by copyright. Any unauthorized reproduction or distribution, in any form or by any means, without written permission from Pearson Education, Inc., is prohibited.



- 論文等を整形した最新情報の活用
- arXiv APIから条件指定で論文を抽出し、検索可能な形式でDBに保存

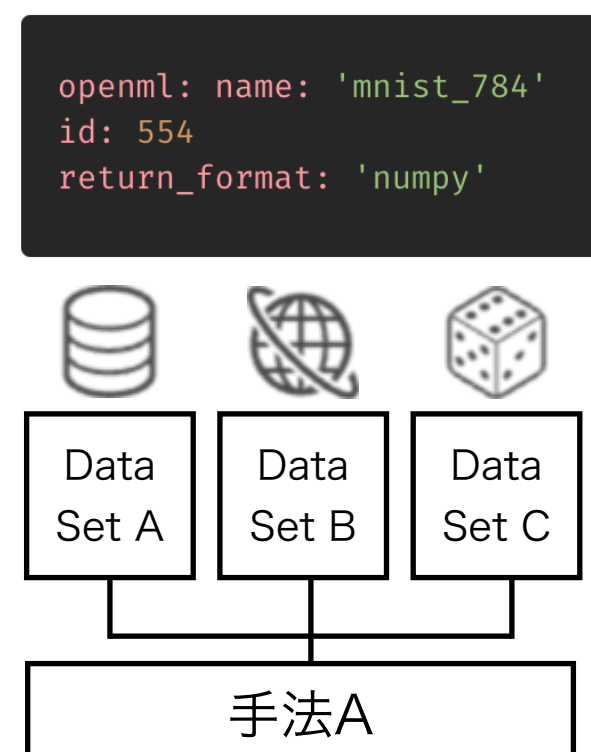
- ・ 専門家の思考過程をプロンプトとして落とし込む
- ・ 量子の特性を活かしながら、実行可能な方向へ導く

- 量子特徴マップの生成以外の部分をツールで保証
- 試行錯誤がFeedback可能なログとして蓄積

Appendix : 量子機械学習のための実験管理ツール QXMTの全体像

データセット

- 自動でダウンロードやダミーデータの生成を実施
- データの前処理をpipelineとして保存し、configで再呼び出し可能



デバイス

- 多種多様なシミュレータ、実機を容易に切り替え可能な形で提供

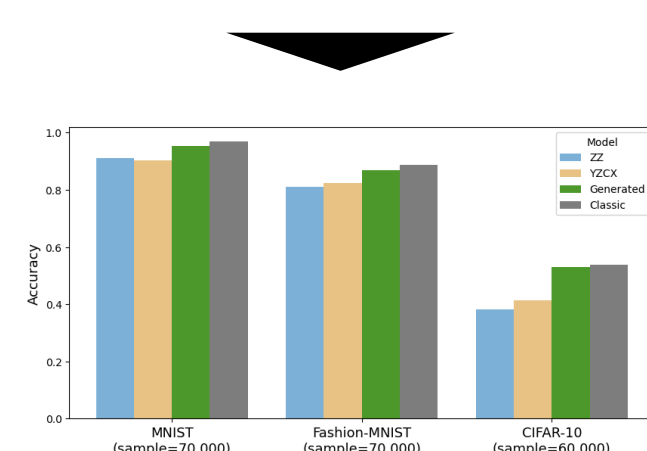


実験サイクルをIDで
管理し再現性を担保

評価

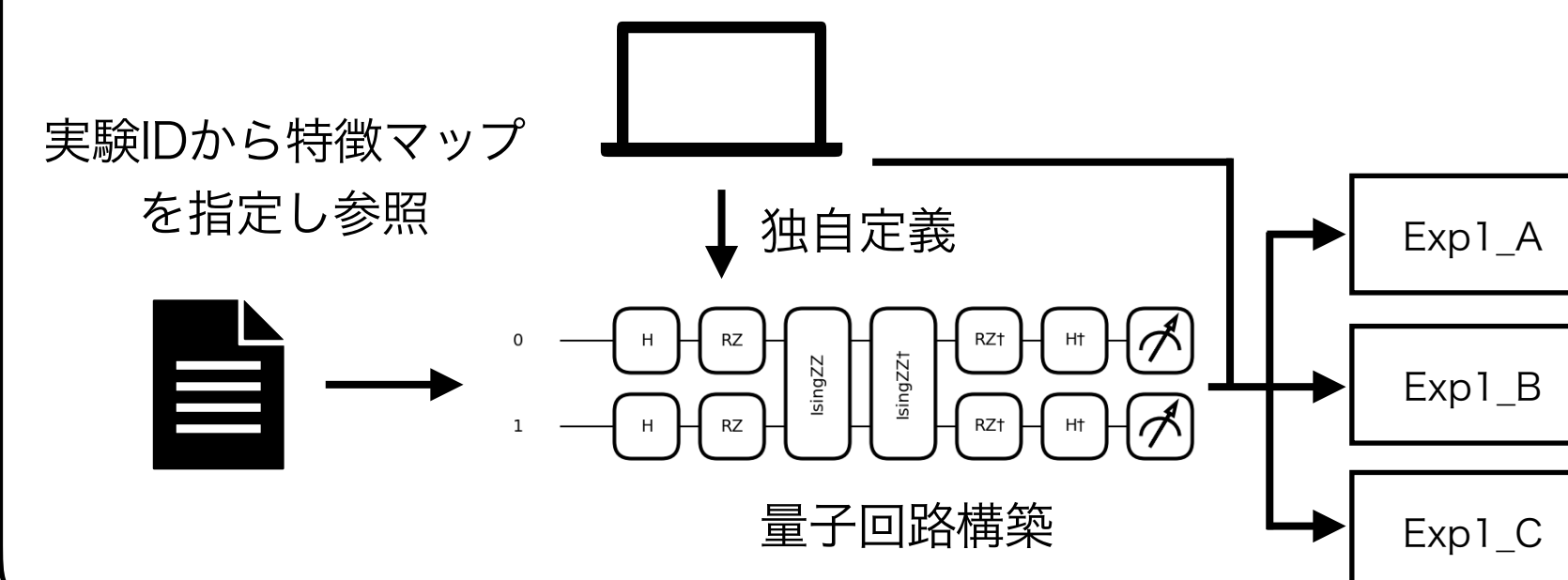
- 多様な評価指標の結果を実験IDに紐付けて保存
- 主要な可視化方法を提供

run_id	accuracy	f1_score
1	0.45	0.59
2	0.40	0.45



量子特徴マップの設計

- 主要な量子回路をconfig経由で提供
- 独自回路を容易に実装するテンプレートを提供
- パラメータ等の実験条件の変更を容易に対応

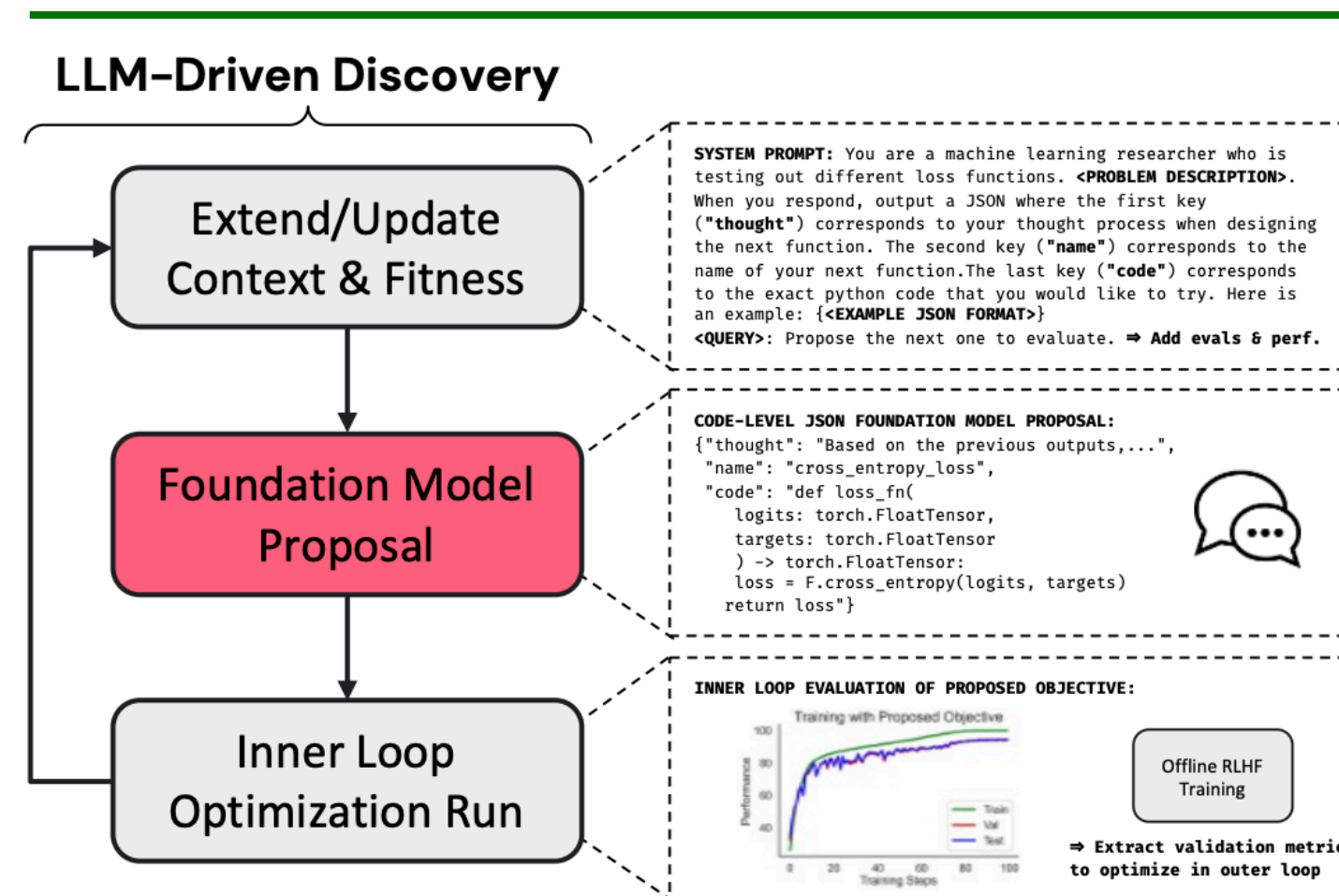


先行研究：古典機械学習におけるモデルの構造探索自動化

古典機械学習では、ニューラルネットワーク等のモデルの構造探索においてAutoMLと呼ばれる技術や強化学習を用いた自動化が行われている

LLMの登場により、外部知識の活用や自然言語を用いた指示の明確化などが可能となり、より汎用的かつ高精度な構造探索を行うことが可能な手法が提案されてきている

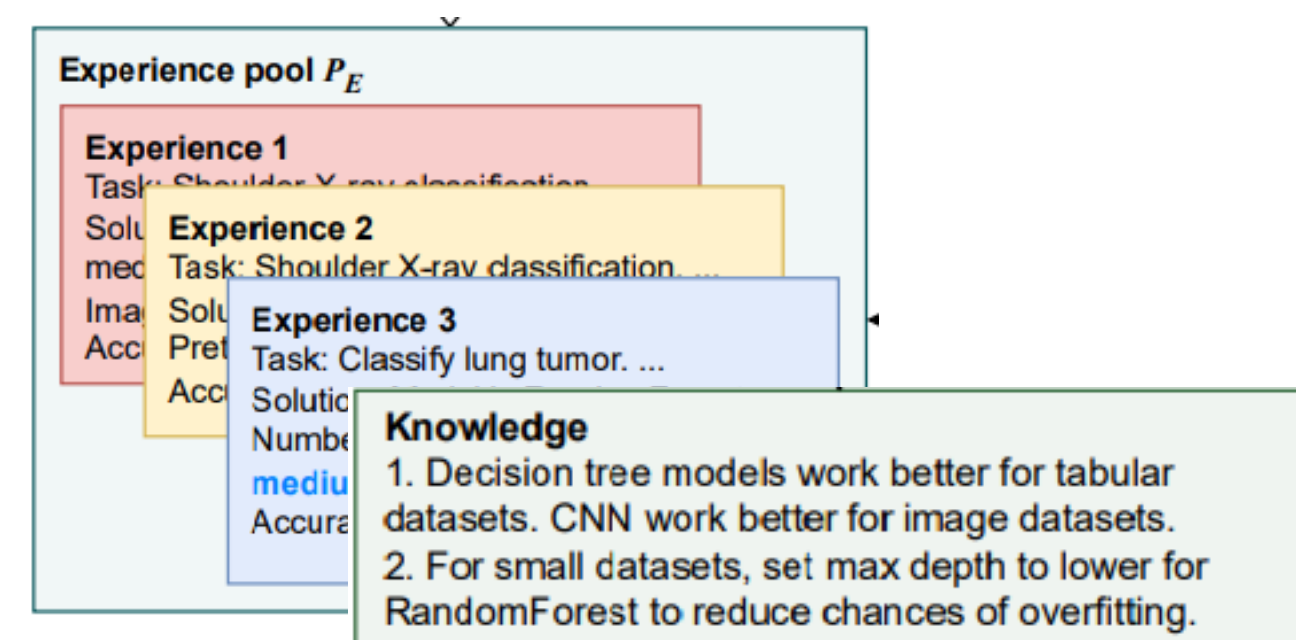
Loss関数の設計自動化



<https://arxiv.org/pdf/2406.08414>

- 特定のデータセットに対するLoss関数の設計をprompt指示
- 出力はLoss関数のコードを元に評価を実施し、結果をpromptにフィードバックすることで次の候補を生成

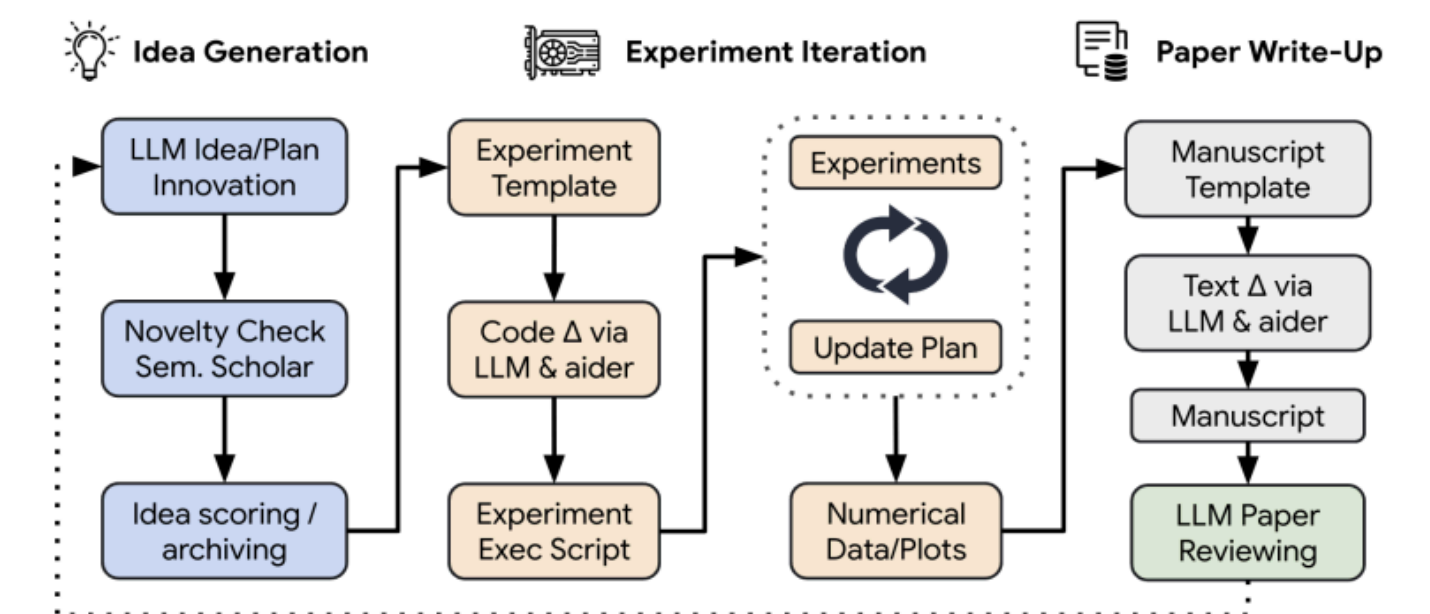
MLCopilot



<https://arxiv.org/pdf/2304.14979>

- 機械学習の論文をクロールし、手法の概要や精度評価の結果をLLMで検索可能な形で保存 (Experiment Pool)
- 上記をLLMでKnowledgeとして抽出
- MLタスクをinputとして、類似度を計算しExperiment PoolとKnowledgeを抽出。LLMに入力することで解法を出力

論文執筆全体を自動化



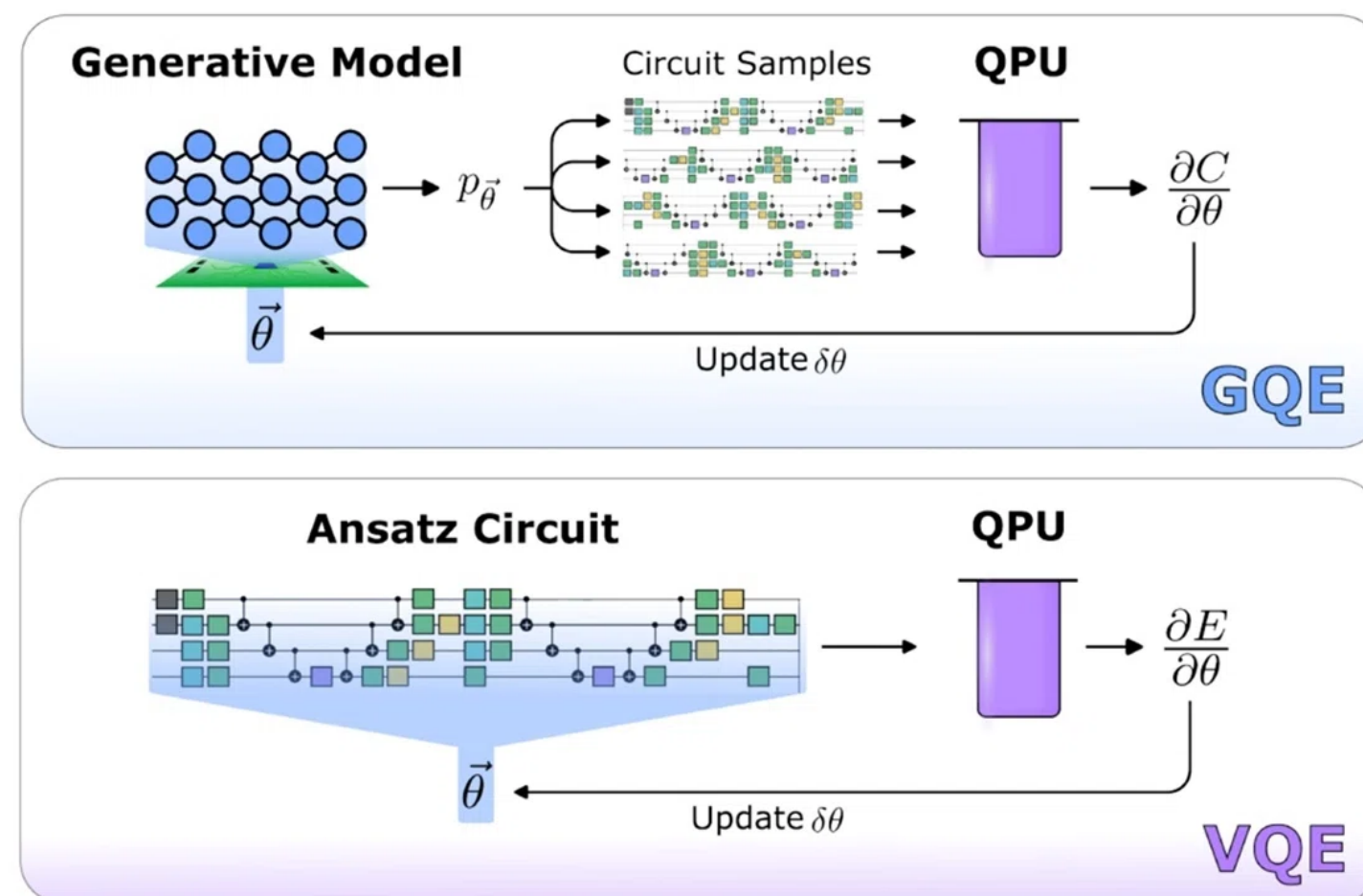
<https://arxiv.org/pdf/2408.06292>

- アイディア出しから論文執筆までの一連のフローをLLMで自動化
- Semantic Scholar APIを用いて生成されたアイディアの新規性を評価
- 人間と同じように実験の計画と実行を複数回行いアイディアをブラッシュアップ
- 実験に必要なコード生成・実行も自動化

先行研究：量子回路生成におけるLLMの活用

量子回路生成に合わせたTransformerモデルの学習方法の提案やLLMを用いた高精度な量子機械学習モデル構築のためのアプローチが提案されてきている

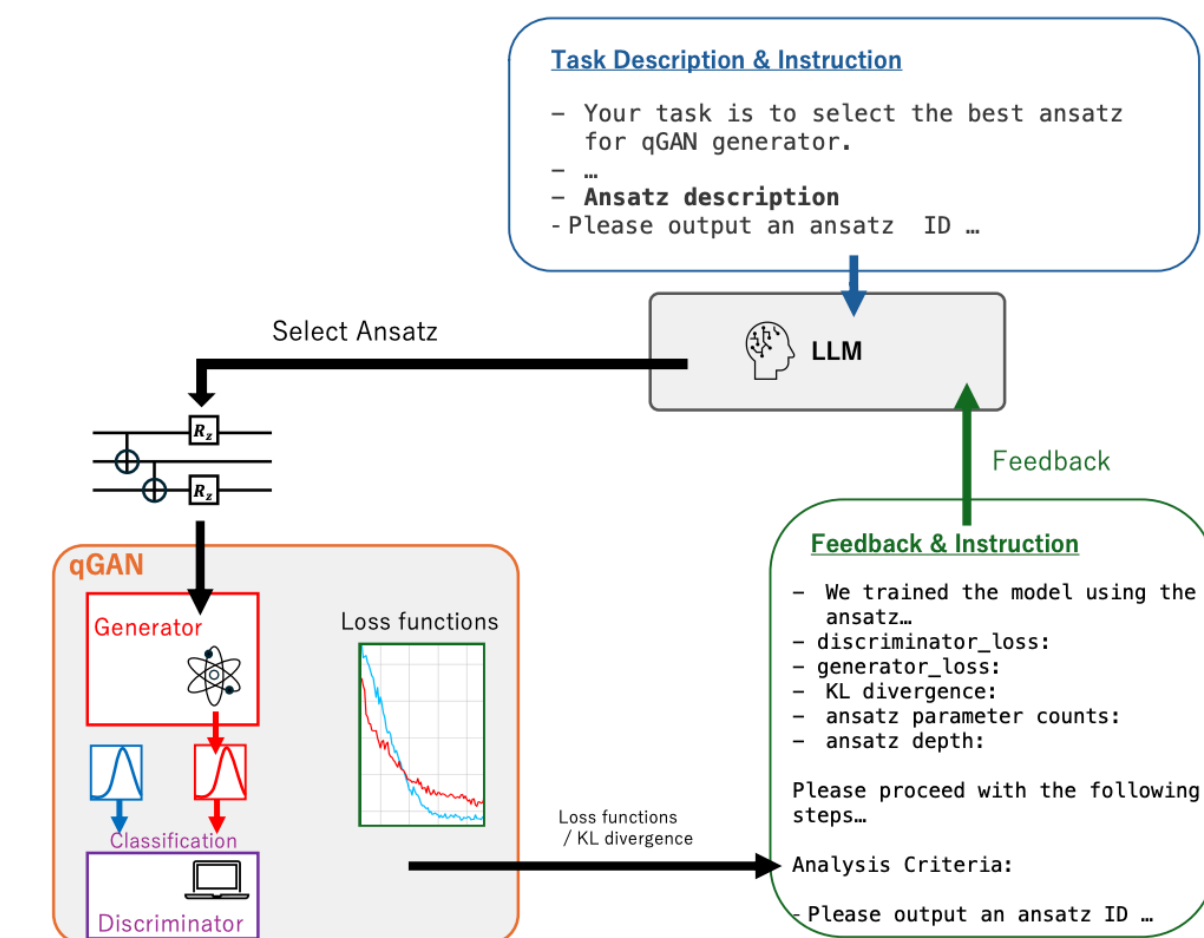
量子回路を出力するTransformerモデル



<https://arxiv.org/abs/2401.09253>

- AnsatzをTokenと見做し、タスクに合わせて量子回路を出力するTransformerモデルを提案
- 化学分子の基底状態探索問題において、有効な量子回路が生成できることを検証

qGANのための量子回路生成を行うコンセプト



<https://arxiv.org/abs/2503.12884>

- 事前に複数のAnsatzをLLMが選択可能な形式で保存
- qGANの学習を行い、その結果をLLMにフィードバックすることでAnsatzの選択を改善

目次

- 量子特徴マップと機械学習
- 量子特徴マップの自動生成に向けて
- 「Automating quantum feature map design via large language models」の論文紹介
- まとめと今後の展望

LLMを用いた量子特徴マップのための自動回路生成

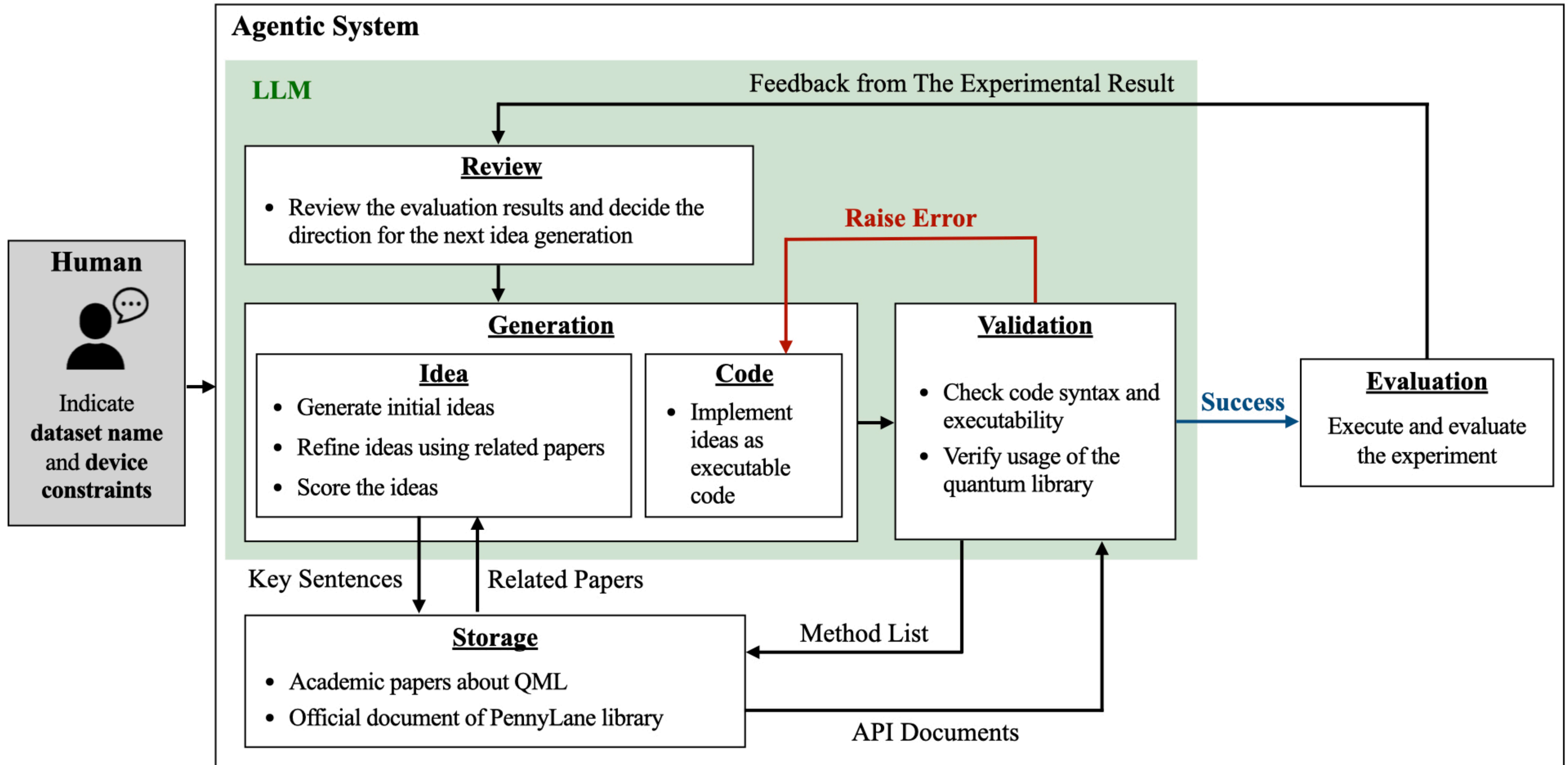
Automating quantum feature map design via large language models

Kenya Sakka, Kosuke Mitarai, Keisuke Fujii

Quantum feature maps are a key component of quantum machine learning, encoding classical data into quantum states to exploit the expressive power of high-dimensional Hilbert spaces. Despite their theoretical promise, designing quantum feature maps that offer practical advantages over classical methods remains an open challenge. In this work, we propose an agentic system that autonomously generates, evaluates, and refines quantum feature maps using large language models. The system consists of five components: Generation, Storage, Validation, Evaluation, and Review. Using these components, it iteratively improves quantum feature maps. Experiments on the MNIST dataset show that it can successfully discover and refine feature maps without human intervention. The best feature map generated outperforms existing quantum baselines and achieves competitive accuracy compared to classical kernels across MNIST, Fashion-MNIST, and CIFAR-10. Our approach provides a framework for exploring dataset-adaptive quantum features and highlights the potential of LLM-driven automation in quantum algorithm design.

<https://arxiv.org/abs/2504.07396>

量子特徴マップ自動生成のアーキテクチャ



Generationモジュール (アイデア)

① 量子特徴マップの実装案を自然言語および数式で出力

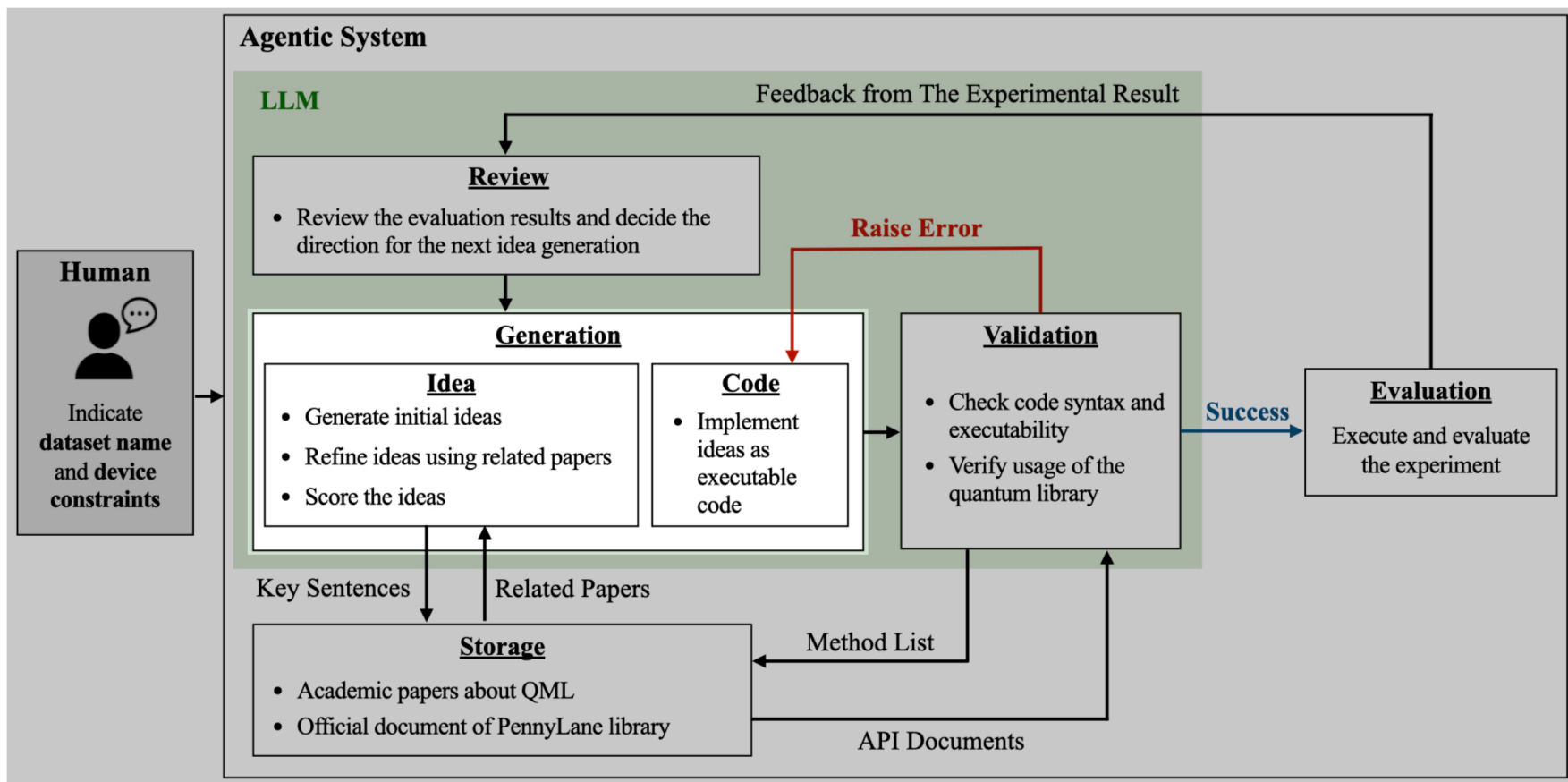
- ・ 提案する量子特徴マップの名前
- ・ 量子特徴マップの数式
- ・ アイデアの概要と詳細
- ・ 検索用のクエリ

② アイデアの改善

- ・ 学術論文を参照し、ブラッシュアップ (詳細はStorageモジュール)

③ 生成したアイデアの定量評価

- ・ Originality
- ・ Feasibility
- ・ Versatility



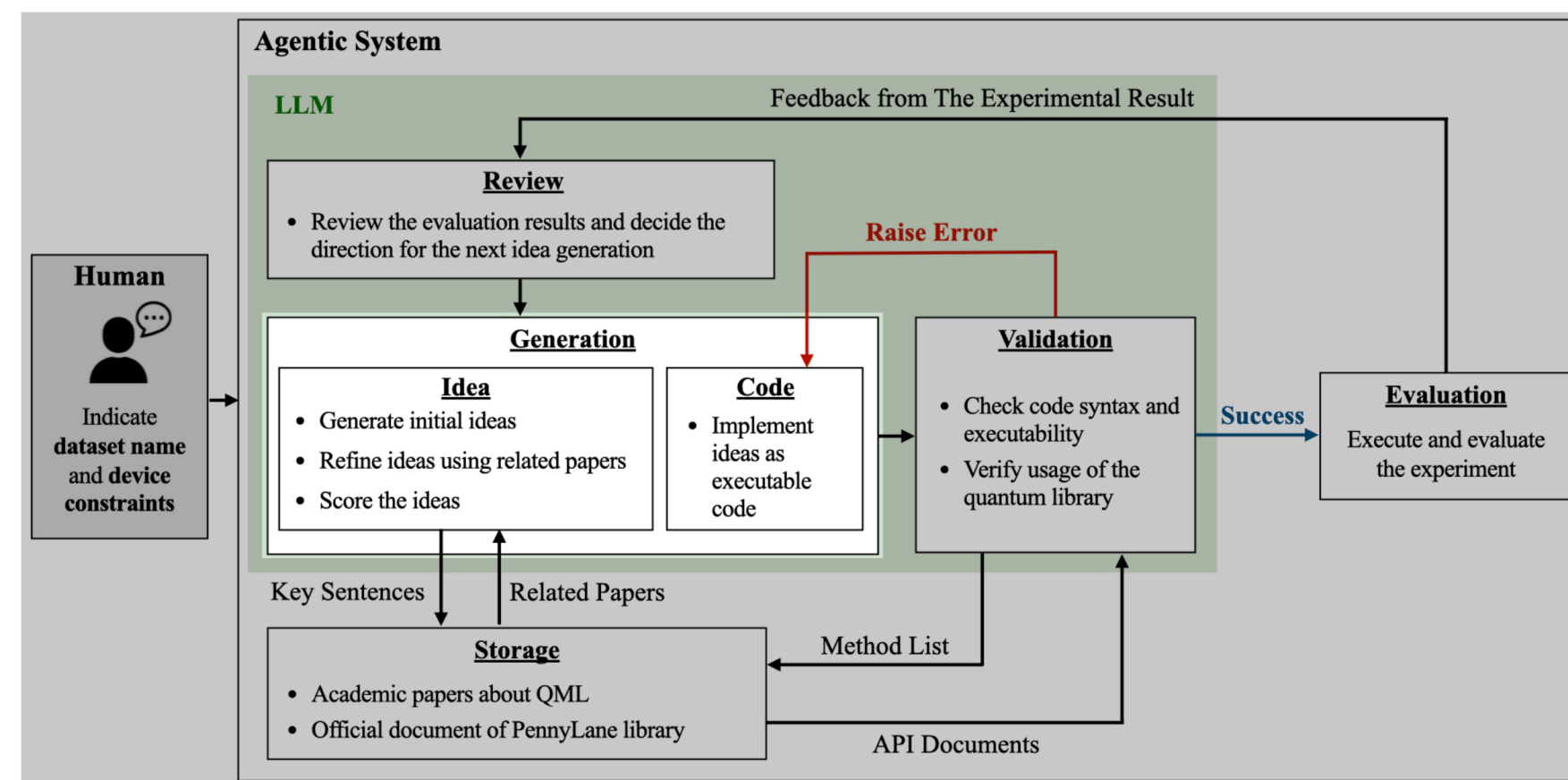
生成結果の例

```
1 {
2   'feature_map_name': 'Hybrid CPS Mid-Range with Dynamic
  Global Fusion Feature Map',
3   'summary': 'This feature map retains fixed local encoding
  via  $R_Y(\pi \cdot x)$  rotations on 5 layers of 16 features and
  implements a three-stage intra-layer entanglement protocol.
  Immediate neighbor coupling is performed with CRX gates
  using ... ',
4   'explanation': 'In this design the 80-dimensional
  PCA-reduced input is partitioned into 5 layers ( $l = 1, \dots, 5$ )
  of 16 features each. For each layer, the first 10 features
  are encoded on a 10-qubit register using fixed  $R_Y(\pi \cdot x)$ 
  rotations to preserve essential feature amplitudes.
  Intra-layer entanglement is organized in three distinct
  stages. ... ',
5   'formula': '\\(\\ket{\\Phi(x)} =
  \\text{MultiRZ}\\Bigl(\\pi\\,\\sum_{l=1}^5
  \\delta_l\\,x_{16(l-1)+10}\\Biggr) \\prod_{l=1}^5 ... ',
6   'key_sentences': ['The 80-dimensional input is partitioned
  into 5 layers of 16 features, with each layer's first 10
  features robustly encoded via fixed  $R_Y(\pi \cdot x)$  rotations on a
  10-qubit register.', ... ]
7 }
```


Generationモジュール (コード)

実行可能な量子特徴マップのコードを出力

- 前のステップで生成されたアイデア + PythonクラスのテンプレートをLLMに入力
- 量子機械学習のための実験管理ツールQXMTと互換性あり
 - 実験の設定をYAMLファイルで管理可能
 - 特徴マップ以外のロジックをライブラリに一任
 - 実験ログの蓄積が容易



生成結果の例

```
1 import numpy as np
2 import pennylane as qml
3 from qxmt.constants import PENNYLANE_PLATFORM
4 from qxmt.feature_maps import BaseFeatureMap
5
6 # new imports can be added below this line if needed.
7
8
9 class
10     AlternatingCPSIntraLayerWithCrossLayerCRotGlobalFeatureMap(BaseFeatureMap):
11 super().__init__(PENNYLANE_PLATFORM, n_qubits)
12     self.n_qubits = n_qubits
13     if delta_weights is None:
14         self.delta_weights = [0.15, 0.25, 0.35, 0.15, 0.10]
15     else:
16         self.delta_weights = delta_weights
```

≈

```
53         idx3 = base + 10 + ((j + 4) % 6)
54         psi2 = 0.25 * norm(x[idx1]) + 0.5 * norm(x[idx2]) + 0.25 *
55         norm(x[idx3])
56         angle2 = np.pi * psi2
57         qml.ControlledPhaseShift(phi=angle2, wires=[j, (j + 2) %
58         self.n_qubits])
59
60         # Intermediate cross-layer entanglement: fixed qubit pairs
61         chi = 0.0
62         for l in range(5):
63             chi += norm(x[16 * l + 12])
64             chi /= 5.0
65         fixed_pairs = [(0, 5), (1, 6), (2, 7), (3, 8), (4, 9)]
66         for a, b in fixed_pairs:
67             qml.CRot(phi=np.pi * chi, theta=0.0, omega=0.0, wires=[a, b])
68
69         # Global entanglement: apply a MultiRZ gate across all qubits
70         global_sum = 0.0
71         for l in range(5):
72             global_sum += self.delta_weights[l] * norm(x[16 * l + 10])
73         global_angle = np.pi * global_sum
74         qml.MultiRZ(theta=global_angle, wires=list(range(self.n_qubits)))
```

Storageモジュール

VectorDB Pinecone

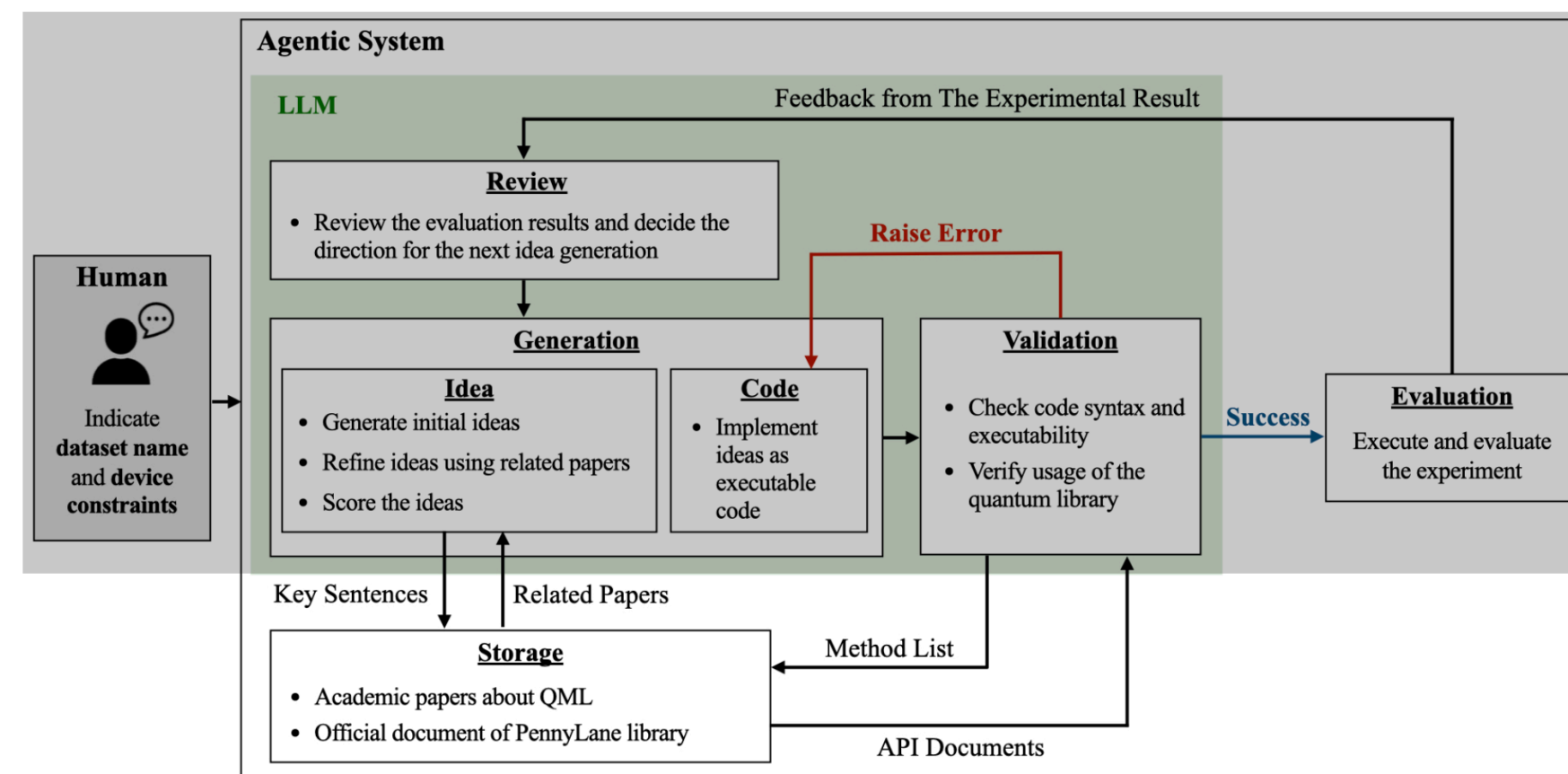
- ドキュメントを1024token毎に1536次元のベクトル空間に埋め込み格納
 - 論文はベクトル検索で取得
 - 検索クエリはLLM自身が生成
 - APIリファレンスはキーワード検索で取得
 - “qml.”を目印にメソッド名で完全一致

学術論文

- ArxivのAPIを用いて論文のPDFデータを取得
 - 期間：2020-01-01 - 2024-12-31
 - キーワード：Quantum Machine Learning
 - 件数：998
- アイデアの改善、スコアリング時に参照

PennyLaneのAPIリファレンス

- Web上のリファレンスは最新状態になっていないこともあり、コード修正での利用は困難
 - 利用バージョンのソースコードに対して、引数やdocstringを解析しドキュメントを作成
- コードのValidation時に参照



Validationモジュール

4段階のコード検証

- ① “PyCompile”によるコンパイルチェック
- ② “ast”によるシンタックスチェック
- ③ PennyLaneのメソッドチェック (Storageモジュールからドキュメントを取得)
- ④ Dry runによる実行チェック

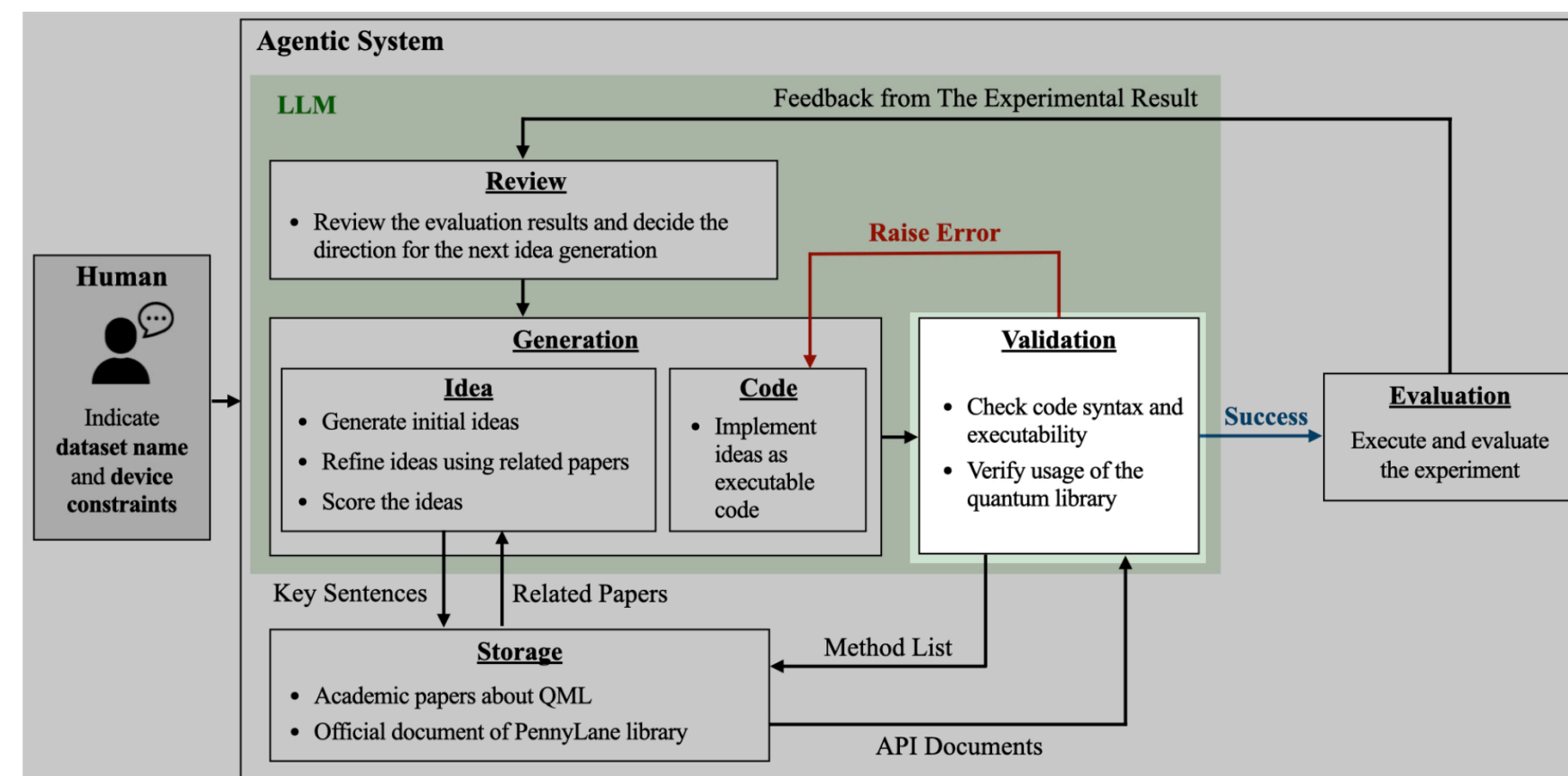
エラーが発生した場合

- ① ソースコード+エラーメッセージをLLMに入力し、修正を依頼
- ② 修正済みのコードに対して4段階の検証を実施
- ③ エラーが出なくなる、または最大回数までステップ①と②を繰り返す

エラーの例

Error occurred during dry run:
OrbitalRotation: wrong number of wires.
3 wires given, 4 expected.

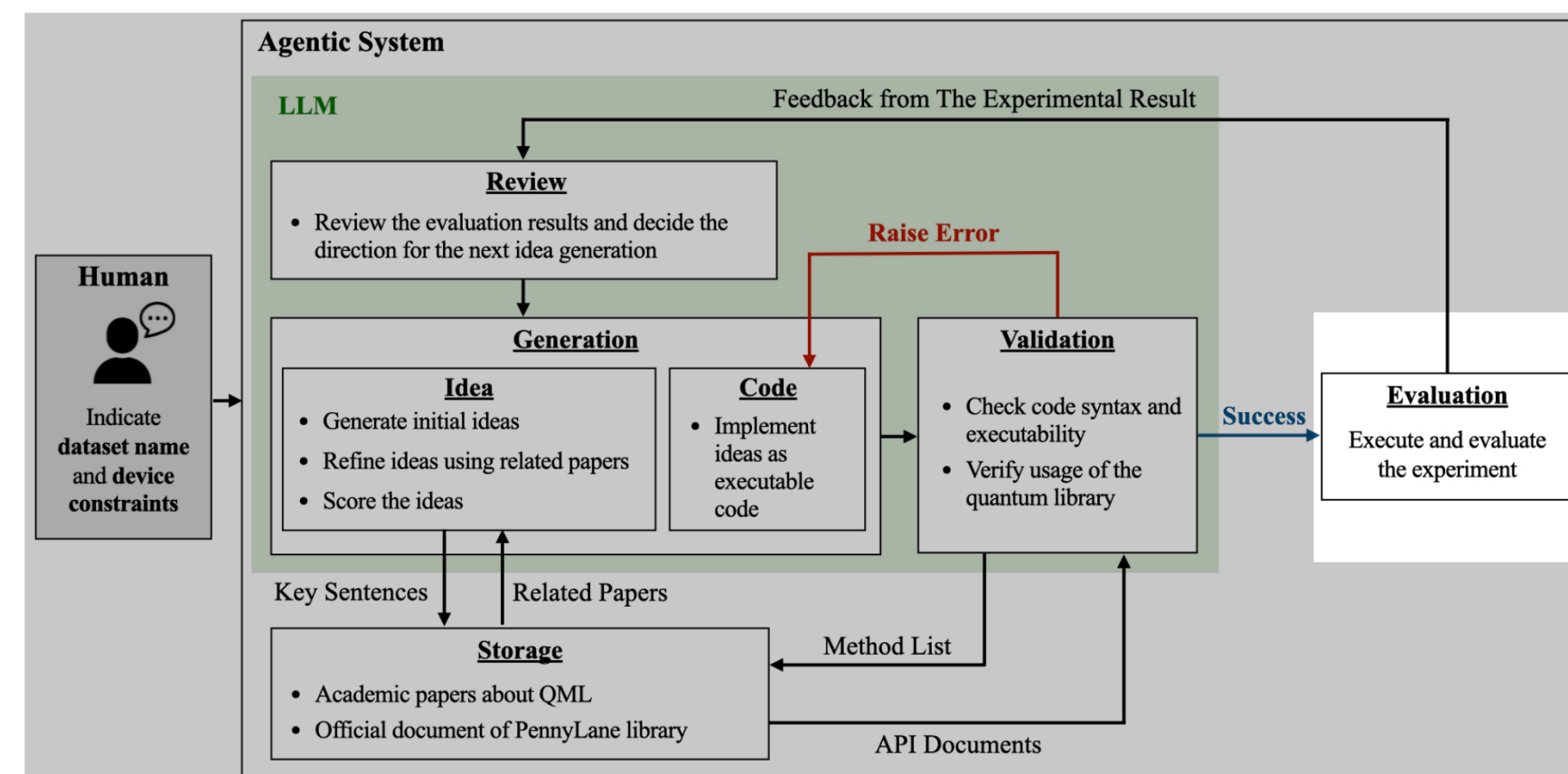
Error occurred during dry run:
RX.__init__() got an unexpected
keyword argument 'theta'



Evaluationモジュール

生成・実装されたアイデアの定量評価

- Validation済みコードで実験を実行し、精度を評価
 - QXMTを利用することで、設定ファイルの更新のみで評価結果を再現可能な形式で保存可能
- Reviewモジュールに、アイデアの評価結果や実行時間をフィードバック



設定ファイルと実験ログの例

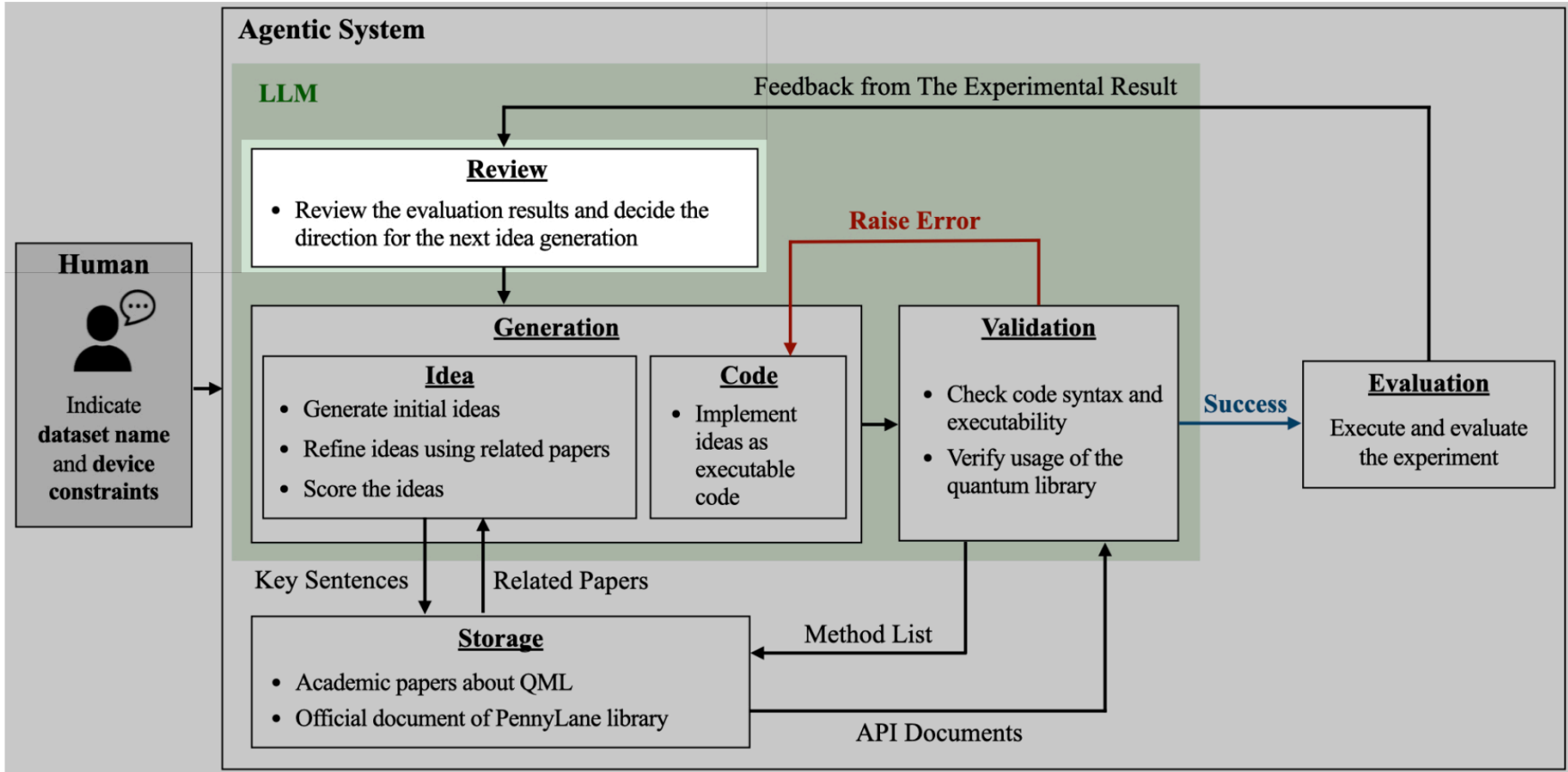
```
1 feature_map:
2   module_name: "qxmt.feature_maps.pennylane"
3   implement_name: "ZZFeatureMap"
4   params:
5     reps: 2      この項目のみ生成結果に更新
6
7 kernel:
8   module_name: "qxmt.kernels.pennylane"
9   implement_name: "FidelityKernel"
10  params: {}
11
12 model:
13   name: "qsvc"
14   params:
15     C: 1.0
16     gamma: "s
17
18 evaluation:
19   default_metric:
20     "accuracy",
```

```
1 "runs": [
2 {
3   "run_id": 1,
4   "desc": "This feature map partitions the ...",
5   "remote_machine": null,
6   "commit_id": "e3xwfed97a9a502xxxxx012e32dd2",
7   "config_file_name": "config.yaml",
8   "execution_time": "2025-02-18 20:15:57.195522 JST+0900",
9   "runtime": {
10    "train_seconds": 75.222216,
11    "validation_seconds": 32.417682,
12    "test_seconds": 32.737439
13  },
14  "evaluations": {
15    "validation": {
16      "accuracy": 0.6755,
17      "precision": 0.6725665417346935,
18      "recall": 0.6690851849431843,
19      "f1_score": 0.6681353851224371
20    },
21    "test": {
22      "accuracy": 0.663,
23      "precision": 0.6582870612817644,
24      "recall": 0.6582230526228374,
25      "f1_score": 0.655553411216151
26    }
27  }
28 }
```

Reviewモジュール

結果をレビューし、次のアイデア生成に活用

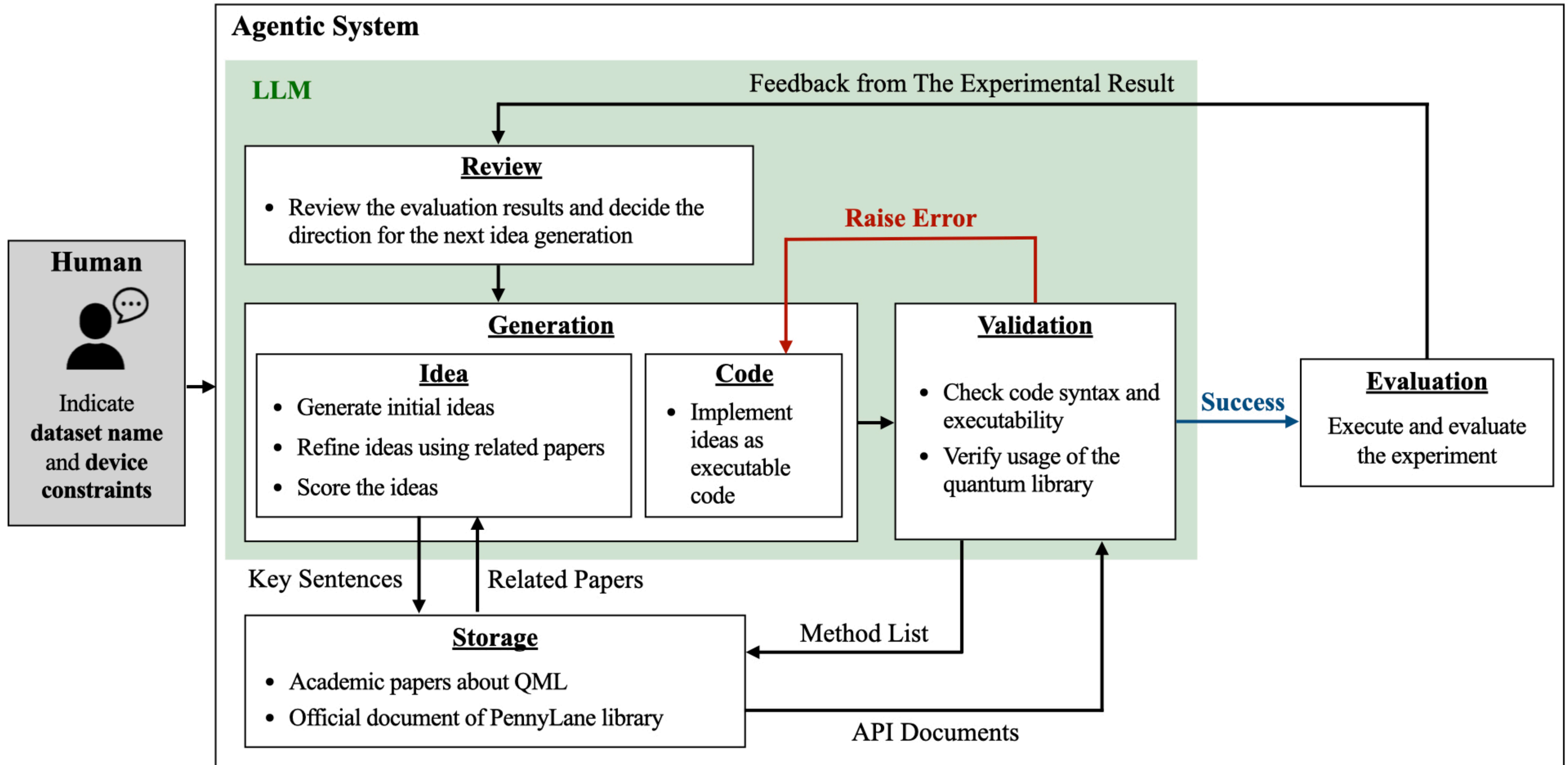
- 改善幅 (= Accuracy(t) - Accuracy (t-1))とし、レビューの方向性を決定
- 以下の情報をLLMに入力し、次のアイデア生成に向けたKeep PointsとSuggestionsを出力
 - レビューの方向性
 - アイデアの概要と評価結果
 - アイデアの優劣 (idea 2 > idea 5 … > idea 1)



スコアの変化量によるレビューの方向性

Significantly Improved	Please review the changes or factors that likely led to this improvement by referring to all past trials, analyze their impact, and propose how we can enhance these aspects further to sustain or amplify the positive trend.
Improved	Please examine the elements that contributed to this progress by referencing all past trials, assess their effectiveness, and suggest additional refinements or strategies to achieve more significant advancements.
Unchanged	Please investigate the potential reasons for this stagnation by comparing all past trials, identify any bottlenecks or limitations, and propose actionable strategies to introduce meaningful progress.
Slightly Dropped	Please review the factors or changes that may have negatively impacted the results by analyzing all past trials, evaluate their significance, and propose targeted solutions to recover or improve performance in subsequent trials.
Significantly Dropped	Please thoroughly analyze the root causes of this drop by referencing all past trials, including any critical changes or issues in the process, and recommend urgent actions to address these challenges effectively and recover performance.

量子特徴マップ自動生成のアーキテクチャ



実験の設定

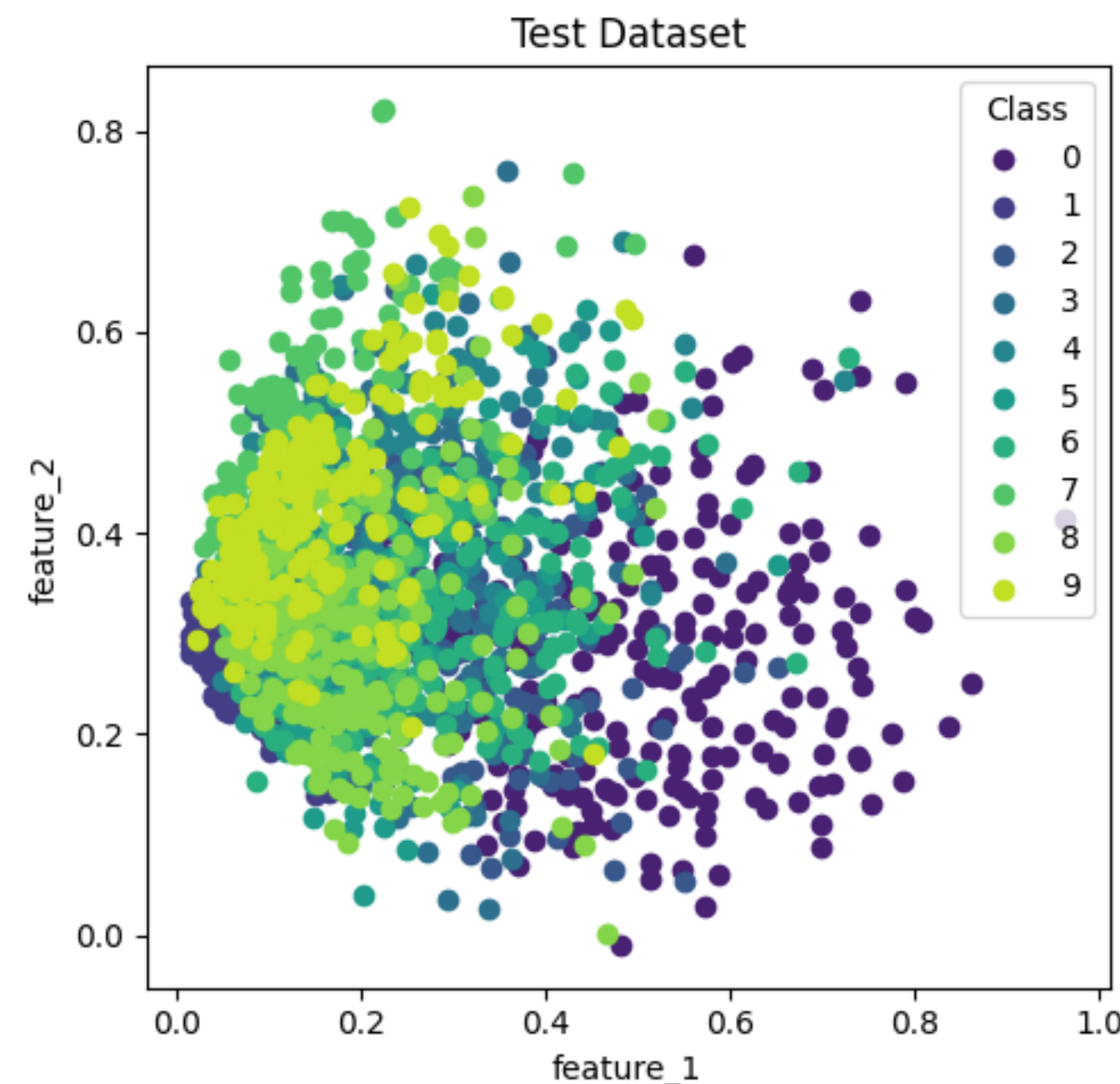
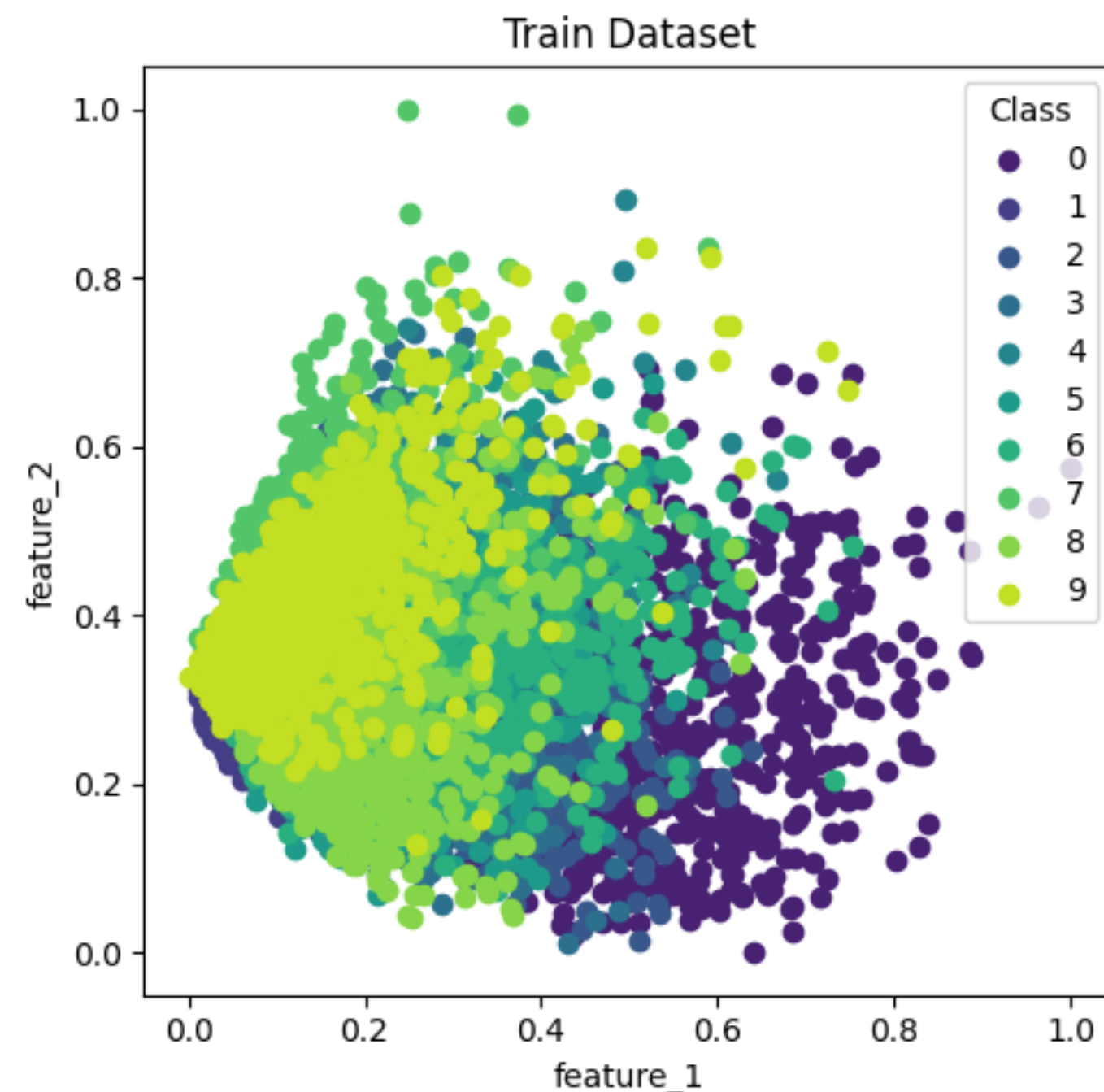
データセット

- データセット

- MNIST784
- Label : 10クラス
- n_sample : 10,000 (Train : Val : Test = 6 : 2 : 2)

- デバイス (シミュレータ)

- “lightning.qubit” (pennylane)
- 10qubit



※ PCAで10次元に圧縮したのち、第1、第2主成分を軸に取り可視化

機械学習モデル (Seed)

・ 前処理 :

- ・ 32x32の画像をPCAで80次元に圧縮
- ・ 値を[0, 1]に正規化

・ 特徴マップ : Empty

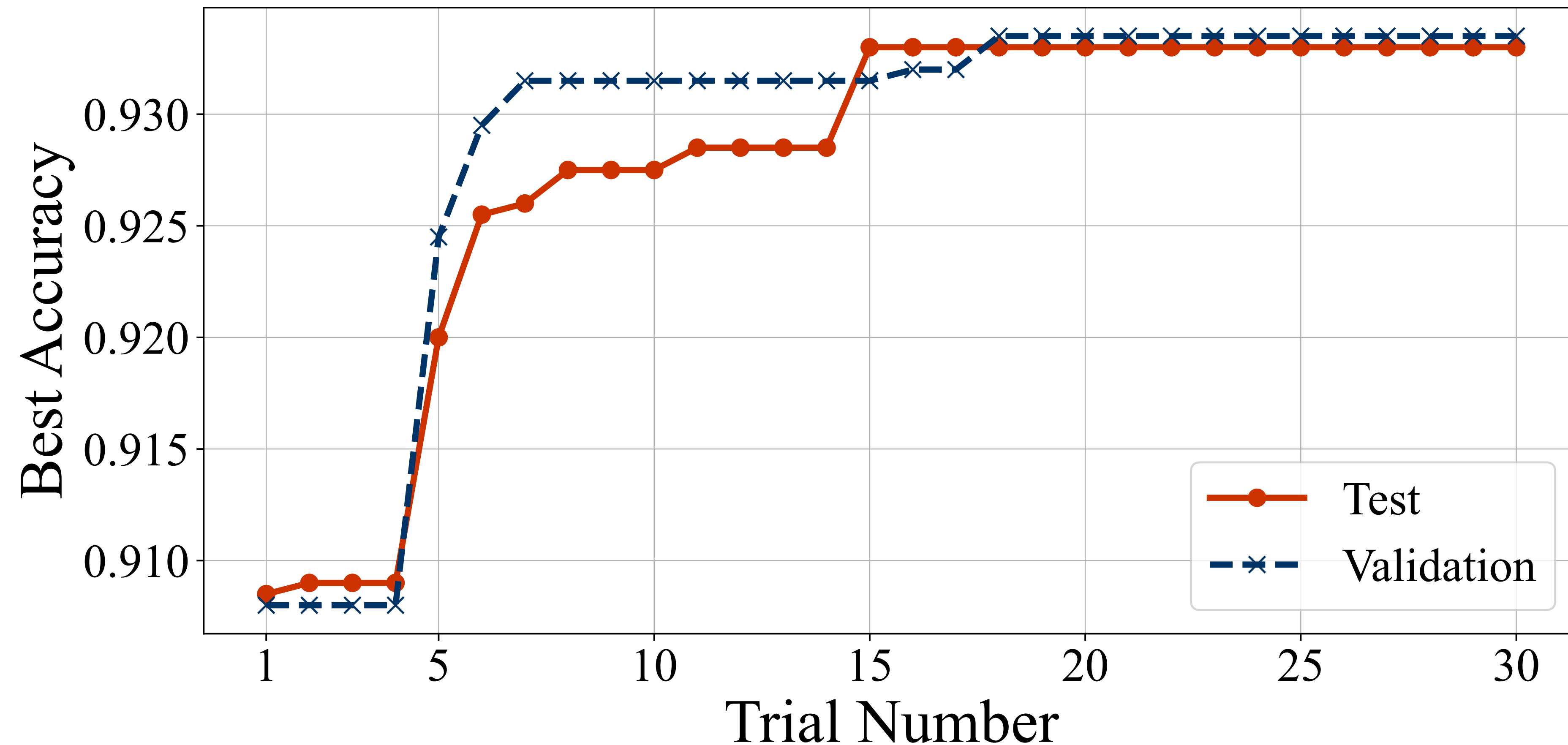
・ カーネル : Fidelity Kernel

・ モデル :

Support Vector Classifier

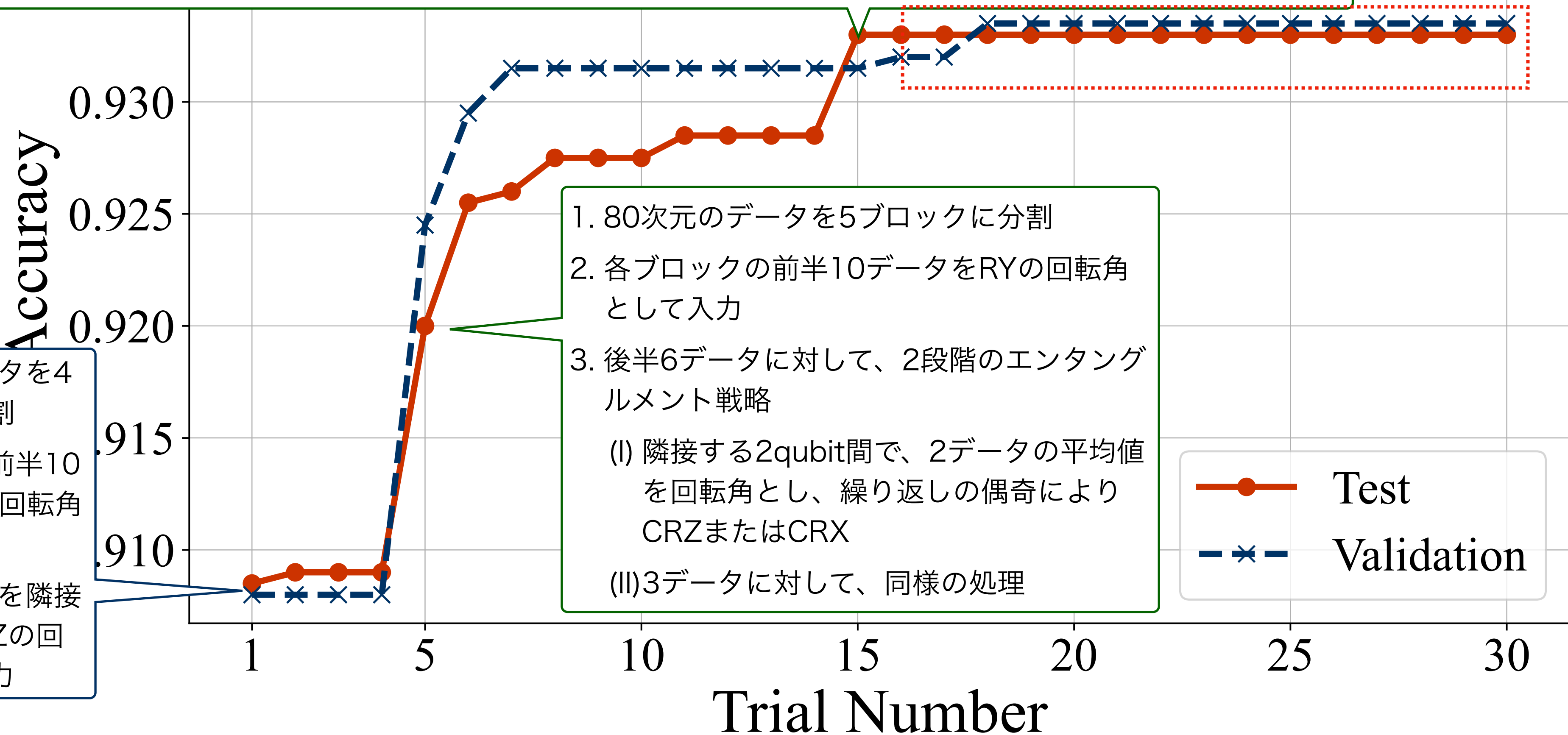
量子特徴マップの生成結果

- 精度評価の結果をフィードバックし、LLMをOptimizerとして活用することで、データセットを高精度で分類可能な量子回路を自動生成できていることを確認



量子特徴マップの生成結果

1. 80次元のデータを5ブロックに分割
2. 前半10データに対して、異なるスケーリング項でRYとRXに入力 (Local Encoding)
3. データの統計量を入力とし、CRZ、CRX、IsingXXを適用 (Immediate Neighbor Entanglement)
4. 異なるデータの統計量を用いて、CRYを適用 (Next-Nearest Neighbor Entanglement)
5. より離れたqubit間でentanglementを導入 (Mid-Range Entanglement)
6. 全入力データから計算される統計量を元にRZを全qubitに適用 (Global Entanglement)

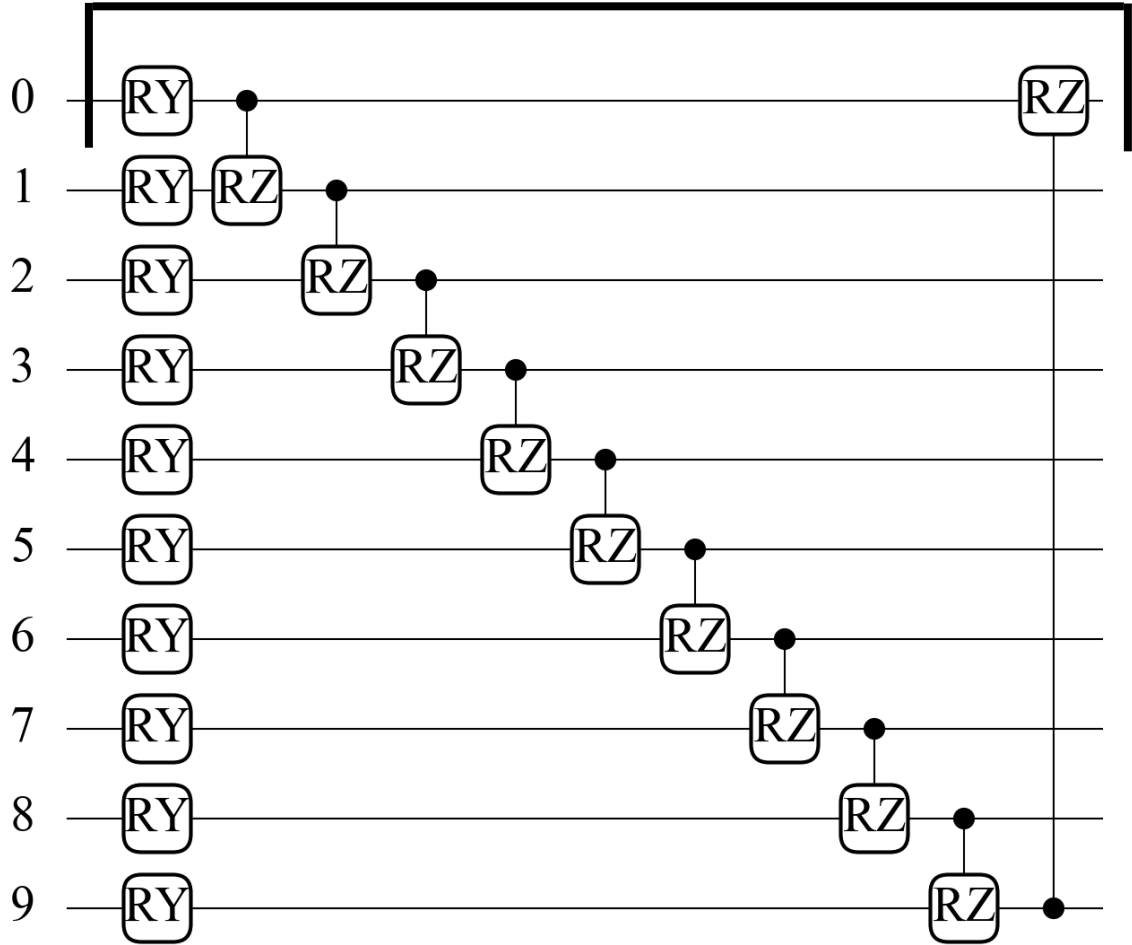


- ・ゲートの入れ替え
- ・ハイパーパラメータの変更
- ・複雑な統計量の導入による精度低下
- ・探索的な回路提案

自動生成された量子回路

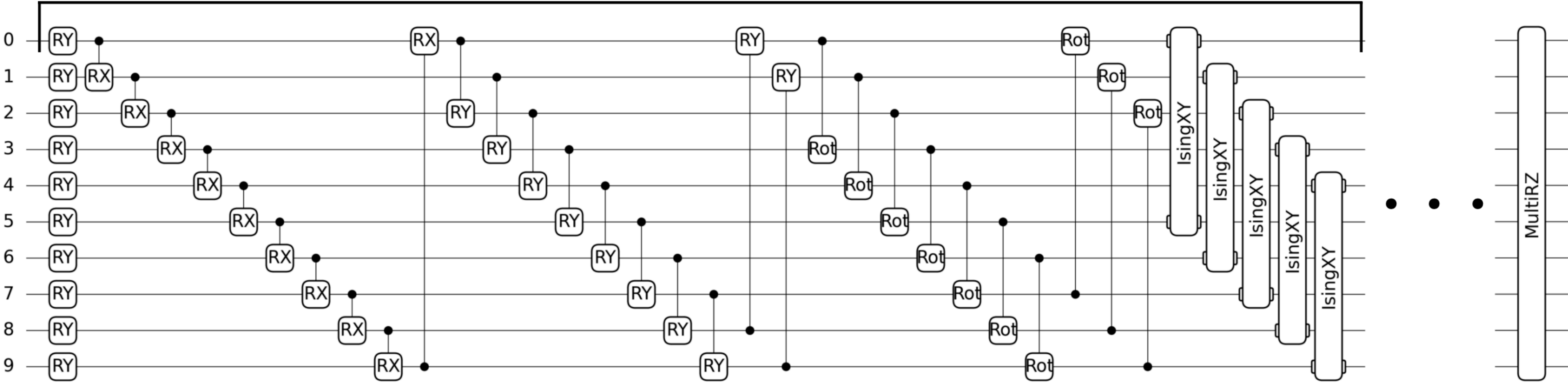
First
Trial

繰り返し数 : 4



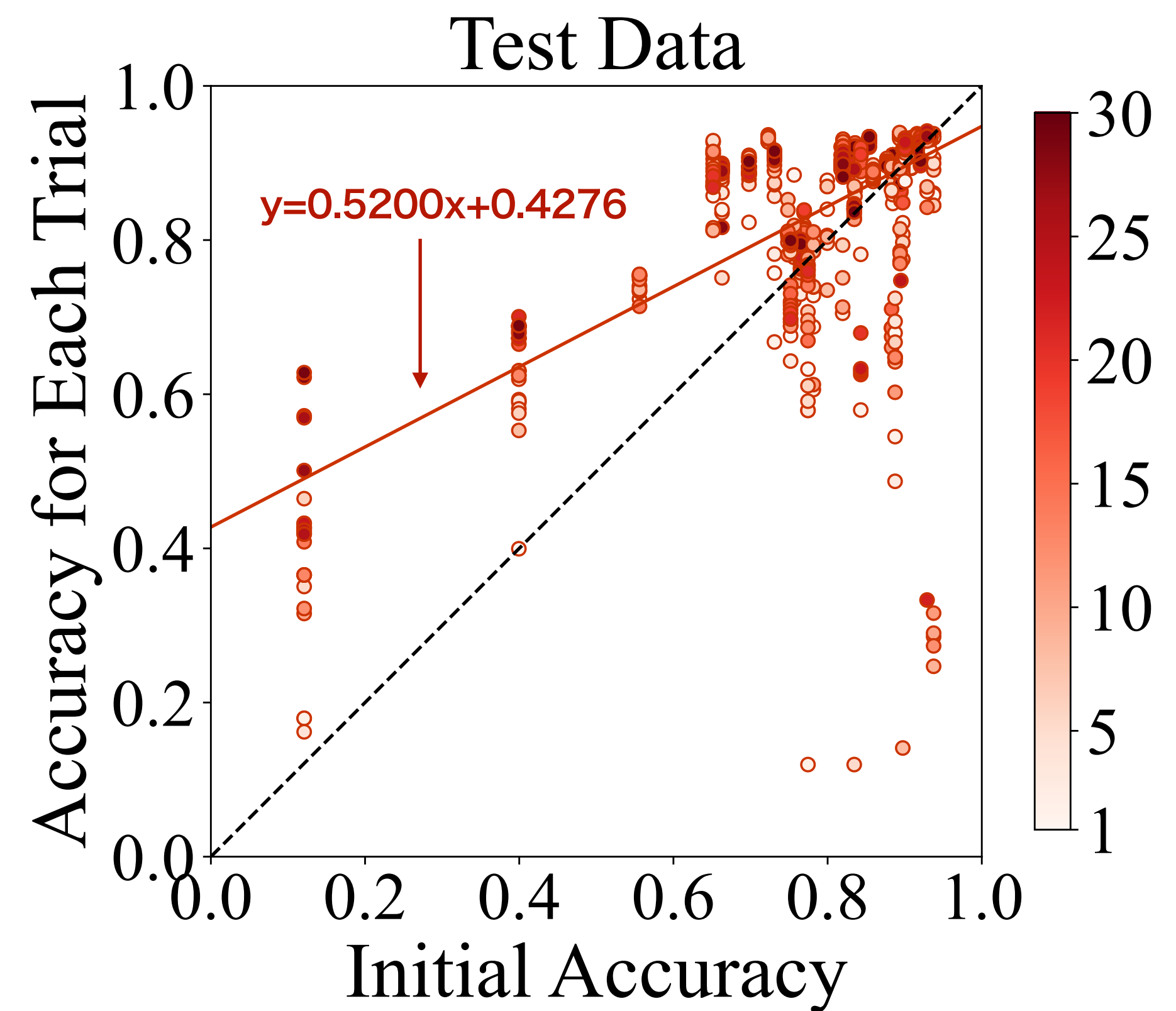
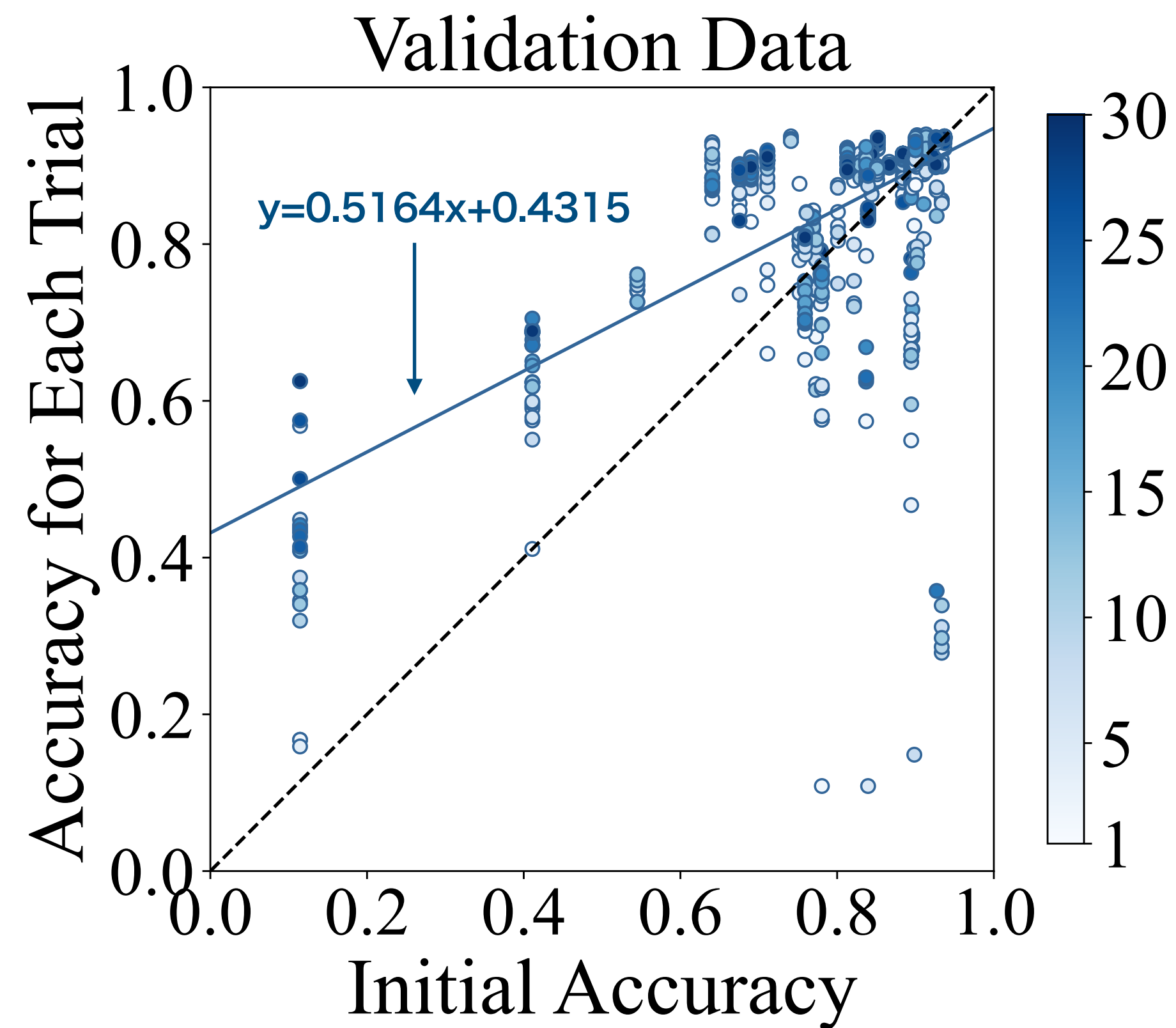
Best
Trial

繰り返し数 : 5



全45実験の評価結果の軌跡と傾向

- 全45実験について、初期生成アイデアの評価結果と全トライアルの評価結果の関係性を可視化。
最終的に生成される量子特徴マップの精度は、初期アイデアに依存する傾向



量子特徴マップの汎用性を評価

- ・ 生成された特徴マップは、全てのデータセットで広く使われている既存の量子特徴マップを上回る
- ・ 古典に対しては、線形・非線形問わず多くの場合で上回ったが、最も一般的なRBFには僅かに及ばない

Table 1: Generated quantum feature map performance on different datasets

Type	Method	Accuracy		
		MNIST (10,000 sample)	Fashion-MNIST (10,000 sample)	CIFAR-10 (10,000 sample)
Classical	Linear Kernel	0.9385 ± 0.0002	0.8437 ± 0.0009	0.4087 ± 0.0011
	Polynomial Kernel	0.9667 ± 0.0058	0.8702 ± 0.0030	0.5375 ± 0.0014
	Sigmoid Kernel	0.9343 ± 0.0002	0.8189 ± 0.0120	0.4079 ± 0.0006
	RBF Kernel	0.9765 ± 0.0005	0.8864 ± 0.0014	0.5669 ± 0.0085
Quantum	ZZ FeatureMap	0.9255 ± 0.0009	0.8252 ± 0.0023	0.3907 ± 0.0016
	NPQC FeatureMap	0.9644 ± 0.0028	0.8749 ± 0.0026	0.4903 ± 0.0188
	YZCX FeatureMap	0.9727 ± 0.0006	0.8778 ± 0.0049	0.4753 ± 0.0341
	Generated Feature Map (Ours)	0.9731 ± 0.0008	0.8835 ± 0.0021	0.5290 ± 0.0030

目次

- 量子特徴マップと機械学習
- 量子特徴マップの自動生成に向けて
- 「Automating quantum feature map design via large language models」の論文紹介
- **まとめと今後の展望**

まとめと今後の展望

- LLMを活用することで、**量子特徴マップのアイデア生成から量子回路生成、評価までの一連のプロセスを自動化し、自律的に改善可能**であることを確認した。加えて、代表的なベンチマークデータセットで生成結果の精度検証を行った
- VQEやQCLなどの異なる量子機械学習アルゴリズムに対するAnsatzの自動生成を可能にすることで、**適用範囲の拡大および理論と応用のギャップを埋めることを目指す**