理解度推定に基づく適応的プログラミング学習支援の提案

佐伯 介[*1]・坂田 瑛祐[*1]・黒坂 修喜[*1]
指導教員：渡邉 寿昭[*2]
Email: Toshiaki_Watanabe@education.metro.tokyo.jp

*1：東京都立多摩科学技術高等学校科学技術科 3 年
*2：東京都立多摩科学技術高等学校

◎Key Words　大規模言語モデル，プログラミング学習支援，理解度推定，個別最適化学習，対話型学習環境

1. はじめに

近年の大規模言語モデル（Large Language Model：LLM）の発展に伴い，LLM を活用したプログラミング学習が広がっている。LLM を用いたプログラミング学習は自然言語による質問応答や即自的なフィードバックによって学習成果の向上や心理的不安の軽減をもたらすことが報告されている一方で，過度な依存や知識定着の難しさが指摘されている。

Ramazan Yilmaz らの研究(1)では，ChatGPT を用いた学習者が，用いていない学習者と比べて，計算論的思考スキル，プログラミングの自己効力感，学習意欲の 3 点で高いスコアを示したと報告している。一方で，Gregor Jošt らの研究(2)では，LLM の使用頻度と最終成績の間に負の相関が確認されている。

この 2 つの結果は，プログラミング学習における LLM の影響は条件や領域によって変化することを示唆している。

認知負荷理論(3)では，初心者に対して有効な指導方法が，より経験豊富な学習者に対しては効果を失うか，かえって負の効果をもたらすことが知られている。この現象は「熟達化反転効果」と呼ばれる。したがって，AI による支援も学習者の習熟度に合わせて変化させる必要があると考える。

Steve Woollaston ら(4)は，学習支援 AI の設計原理として，「意図的な摩擦」と「動的なフェーディング」を提案している。前者は AI が即座に回答を提示せずに学習者に考えさせる「摩擦」を設計に組み込むことで，批判的思考と深い学習を促すものであり，後者は学習者の習熟度に応じて AI の支援を自動的に減らし，最終的な学習者の自律性の確保を目指すものである。

これらの先行研究から，学習支援のあり方は学習者の状態に応じて変化することが望ましいと考えられる。そこで本研究では，学習者の習熟度に応じて LLM による支援の方向性，介入の程度を段階的に調整するプログラミング学習環境を設計し，その有効性について検討する。

2. 目的

本研究では，学習者のプログラミングスキルを複数の観点から評価し，その状態に応じた学習支援を実現するため，支援内容や程度を動的に調整する AI プログラミング学習環境の設計・構築を行う。

3. 評価観点に基づく支援レベルの設計

プログラミングスキルの評価観点として、本研究では、次の 4 観点を設定した。

問題理解（Q）

問題の内容や制約条件を理解し，求められていることを考えて行動の方針を決めるスキル。問題文中の条件の整理や，制約条件からの計算量の推定などが当てはまる。

コード読解（R）

自身以外が記述したコードを読み、実装意図や使われているアルゴリズムを理解し，整理する能力。変数や関数の役割の把握や，バグの原因箇所の特定などが当てはまる。

設計（D）

問題の要求を満たす実装方法やアルゴリズムを検討し，実装に先立って計画を建てるスキル。モジュールや関数の設計，適切なデータ構造やアルゴリズムの選択などが当てはまる。

実装（I）

設計に基づいて実際にプログラムを記述するスキル。また、エラーや要件に応じてプログラムを修正するスキル。デバッグ作業もこれに含まれる。

本研究ではこれら 4 つの観点に基づいた適応支援を行うこととした。観点ごとに習熟度の推定を行い，観点ごとに支援のレベルを決定する。これにより得意，不得意に応じて支援内容を変更することができ，不得意なスキルについては重点的に支援を行うことができる。

また，すべての問題が 4 つのスキルのすべてを要求するわけではない。例えば，コードを読解し，説明することを目的とした問題ではコード読解能力が主に求められる一方で，実装能力はほとんど要求されない。このような問題ごとの差異を考慮するため，その問題において求められるスキルにおうじて，4 つの観点の一部に注目して適応を行う。

さらに，学習者の習熟度のみを参照した適応では，問題

の難易度の上昇と支援の減少が重なり，こう難易度問題における過少支援が発生する可能性がある。そこで，学習者の習熟度だけでなく問題の難易度も観点別に管理し，両者の差に基づいて支援内容を決定することで，学習者にとっての相対的な問題難易度に応じた適応支援を実現する。

4. 方法

4.1 基本設計

本研究では，学習者のプログラミングスキルを問題理解（Q），コード読解（R），設計（D），実装（I）の4つの軸でモデル化し，4観点の習熟度に応じてAIによる支援内容を動的に調整する。

システムはeラーニング形式のWebアプリケーションとして実装する。学習時には問題画面とAIとのチャット画面を並列に表示し，学習者はAIとの対話を行いながら学習を進めることができる。

各観点の習熟度は学習者ごとにレーティング値として管理する。また，各問題について，関連する観点に対して難易度レーティングを設定する。問題によっては一部の観点を使用せず，習熟度更新および支援内容決定の対象外とする。 例えば，コードの読解説明を行う問題では，実装を伴わないためI軸の設定を行わない。問題の難易度レーティングは問題作成者が各観点に対して事前に設定し，動的な更新は行わない。

4.2 習熟度レーティングの更新

学習者の習熟度レーティングは，Eloレーティングの考え方を参考にしつつ，問題難易度と学習者習熟度の差を反映する独自の更新方式によって算出する。

学習者の能力軸レーティングを R_u，問題の難易度レーティングを R_p とする。また，更新量の調整係数を K とする。

まず，学習者の習熟度と問題難易度の関係を表す係数 (C) を次式によって求める。

$$C = \frac{1}{1 + \exp\left(-\frac{10(R_p - R_u)}{100}\right)}$$

問題難易度が学習者の習熟度を上回る場合には C は大きくなり，逆に学習者の習熟度が問題難易度を上回る場合には小さくなる。

次に，問題難易度に基づく基本変動量 B を次式で定義する。

$$B = \frac{R_p}{K}$$

学習者が問題に正答した場合のレーティング変化量 ΔR は

$$\Delta R = BC$$

とする。

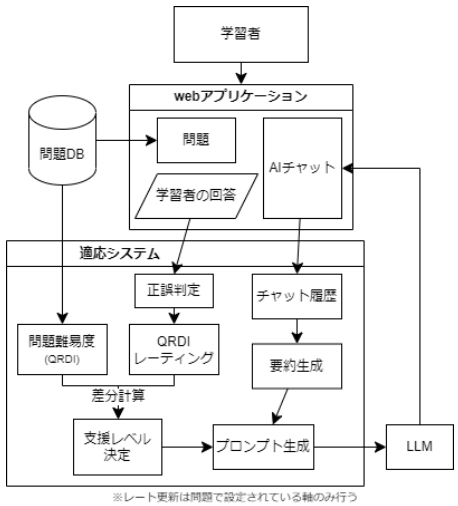

図1 システム構成図

表1 支援レベル

レベル	支援内容
Level 5	具体的手順や詳細な助言を提示
Level 4	設計例や部分的な実装例
Level 3	方針の示唆
Level 2	抽象的ヒント
Level 1	能動的支援なし

一方，不正答の場合は

$$\Delta R = -B(1 - C)$$

とする。

この更新方式により，学習者にとって難易度の高い問題に正答した場合には大きなレーティング上昇が得られる。一方で，難易度の高い問題への不正答によるレーティング低下は小さく抑制される。逆に，学習者にとって容易な問題への不正答は大きなレーティング低下となる。これにより，各能力軸の習熟度を継続的に推定することが可能となる。

4.3 AIによる適応的支援

AIによる支援内容は，各能力軸における学習者の習熟度レーティングと問題の難易度レーティングとの差に基づいて決定する。

支援レベルは5段階とし，レベルが高いほど具体的かつ積極的な支援を行う。例えば，高い支援レベルでは解法の方向性や設計方針に関する具体的な助言を与える。一方，低い支援レベルでは直接的な回答を避け，学習者自身の思考を促すためのヒントや質問を中心とした支援を行う。

問題の難易度が学習者の習熟度を大きく上回る能力軸については，その能力軸に対応した支援レベルを高く設定する。一方，学習者の習熟度が問題難易度を十分に上回る能力軸については，支援レベルを低く設定する。

これにより，学習者にとっての相対的な問題難易度に応じた支援が可能となる。さらに，同程度の難易度の問題に対しては，学習者の習熟度向上に伴って支援量を段階的に減少させることができる。この仕組みは，意図的な摩

擦および動的なフェーディングの考え方を実現するものである。

4.4 対話履歴を用いた支援内容の調整

能力軸ごとのレーティングに加えて，学習者と AI との対話履歴も支援内容の決定に利用する。

学習者の質問内容や誤解の傾向，思考過程に関する情報を対話履歴から抽出し，それらを要約した調整用プロンプトを生成する。この調整用プロンプトをシステムプロンプトに付与することで，レーティングでは表現しきれない学習者固有の特徴を支援内容へ反映する。

これにより，同一のレーティングを持つ学習者であっても，個々の理解状況や学習上の課題に応じた支援を提供できる。

5. 実験予定

実験参加者は高校生から大学生のプログラミング学習者を対象とする。

提案システムの有効性を検証するため，以下の 3 条件で比較実験を行う。

[A] 理解度に応じて支援を動的に調整する AI 支援付学習（提案手法）

[B] 支援レベルを固定した AI 支援付学習（静的 AI 支援）

[C] 支援なし（非 AI 学習）

被験者には最初に AI 補助なしでの能力テストを実施し，正答率，回答速度を記録する。その後，被験者には開発したアプリケーションを用いて学習を実施させる。

各条件における課題達成度・回答速度を比較する。また，AI 支援群の被験者の AI との会話内容を分析する。加えて，アンケート調査を通じて AI 支援の有用性および心理的傾向を測定する。最後に，能力テストを再度実施し，正答率，回答速度を記録する。

使用言語は Python とし，問題は自己制作したものに加え、MBPP(Mostly Basic Python Problems)をはじめとした外部問題データセットを改変したものを利用する。

評価においては，以下の項目を検証する。

- 学習前後の能力テストの点数の変化を分析し，学習の効果を比較検証する。
- AI との対話内容を分析し，学習者が教員 AI をどのように利用したかを定性的に考察する。
- アンケートの結果をもとに，学習者にとっての支援の有用性，自己効力感，AI への依存自覚度などを検証する。

本研究で実装した AI が学習補助者としての役割を適切に遂行できている場合，能力テストの点数や回答速度は，A 群が B 群，C 群と比べてより向上しているはずである。逆に A 群が C 群と比較して悪い結果を記録した場合，AI 利用が学習者の自律的思考に十分寄与せず，学習者が AI に依存している可能性がある。

6. 今後の展望

6.1 モデルの検討

本研究では，学習支援に用いる大規模言語モデルとして GPT-4o-mini を使用したが，他のモデルを使用した場合の挙動差については検討されていない。

具体的には GPT5.2 などの上位モデルや，Google Gemini，Claude Sonnet などの他社モデル，GPT-5-Codex といったコーディング特化モデルとの比較を実施したい。

また，現在は学習者との対話，問題回答の採点，会話ログの分析に AI が用いられているが，タスクの性質が異なるため，それぞれに適切なモデルを割り当てることが理想である。

6.2 AI による問題の自動生成

本研究では，学習者が解く問題は事前に用意されたものとなっている。

AI により，学習者の苦手に合わせた教材，問題を動的に生成することができれば，より高い学習効果を目指せると考える。

ただし，考察で述べたような問題に対する相対的な難易度による適応が AI 生成の問題でも同様に機能するようにするためには，生成された問題の難易度の数値を適切に設定しなければならない。さらに，CDM を導入した場合，属性ごとに値を設定する必要がある。AI によって問題を生成する場合，こういった数値設定で適切な値を設定することについて，大きな課題があると予想される。

7. おわりに

本研究では，学習者の能力を問題理解（Q），コード読解（R），設計（D），実装（I）の 4 軸としてモデル化し，能力に応じた AI 支援を行うシステムを提案した。

今後，実際に本システムによる実験を実施し，効果の検証を行いたい。

謝辞

論文執筆にあたり，多大なるご指導とご助言を賜りました東京都立多摩科学技術高等学校 樫村陽子先生に深く感謝申し上げます。

また，本研究で利用した各種オープンソースソフトウェアおよびデータセットの開発・提供者の皆様に感謝いたします。

参考文献

(1) Ramazan Yilmaz, Fatma Gizem Karaoglan Yilmaz：“The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation”，(2023).

(2) Gregor Jošt, Viktor Taneski, Sašo Karakatič：“The Impact of Large Language Models on Programming Education and Student Learning Outcomes”，(2024)

(3) Slava Kalyuga, Paul Ayres, Paul Chandler, John Sweller：“The Expertise Reversal Effect”，(2003).

(4) Steve Woollaston, Brendan Flanagan, Isanka Wijerathne, Hiroaki Ogata：“Agentic AI and Pedagogical Best Practice: The Tension Between Automation and Learning”，(2026).