P2-09

プログラミング学習者の行き詰まり検出におけるデータ拡張の検討

Data Augmentation for Impasse Detector in Programming Exercises

内山魁[*1]，野口靖浩[*2]，小暮悟[*2]，山本頼弥[*3]，山下浩一[*3]，小西達裕[*2]
Kai Uchiyama[*1], Yasuhiro Noguchi[*2], Satoru Kogure[*2],
Raiya Yamamoto[*3], Koichi Yamashita[*3], Tatsuhiro Konishi[*2]
[*1]静岡大学大学院総合科学技術研究科情報学専攻
[*1]Department of Informatics, Graduate School of Integrated Science and Technology,
[*2]静岡大学情報学部
[*2]Faculty of Informatics, Shizuoka University
[*3]常葉大学経営学部
[*3]Faculty of Business Administration, Tokoha University
Email: uchiyama.kai.20@shizuoka.ac.jp

あらまし：演習形式のプログラミング授業では学習者数に比べて教員数が少ないことから，各学習者の状況をリアルタイムに把握してサポートすることが難しい．その状況に対して学習者の行き詰まりを自動的に検出して教員に伝える方法が開発されている．しかしながら，行き詰まるタイミングはコーディング活動時間全体と比べて頻度が低いため，データ数の不均衡の問題が生じる．そこで本稿では，収集したコーディング活動データに対してデータ拡張手法を適用し，データ数の不均衡を改善することで検出モデルの性能向上を試みた結果を報告する．
キーワード：プログラミング言語教育，データ拡張，機械学習，行き詰まり検出，GAN

1. はじめに

学習者にコーディング課題を提示した後，教員が学習者間の巡回しながら各自の演習状況を把握して，適宜サポートを行う演習形態がある．しかし，教員の人数が少ない場合や，オンライン・オンデマンドなどの学習形態の変化によって，教員が学習者の状況をリアルタイムに把握することが難しい場合があり，学習者の行き詰まりへの支援に問題がある．

この問題に対して演習中の学習者のコーディング活動を周期的に収集し，これらの活動データから行き詰まりを自動検出して教員に通知する仕組みが提案されている(1)(2)．しかしながら，行き詰まりのタイミングは，学習者のコーディング活動全体に比べて頻度が低く，カテゴリ間の量の不均衡が生じやすい．カテゴリ間のデータ量の不均衡は機械学習モデルの性能に影響を及ぼすことで知られており，教育分野においても不均衡の改善によってモデルの性能向上を目指す研究が進められている(3)(4)．

本稿では行き詰まりの発生頻度からくるデータの不均衡に対しデータ拡張手法による改善を行い，拡張前後で機械学習の分類性能を比較検討する．

2. 使用したデータ

本稿では小暮ら(2)の研究で収集されたデータセットを対象にデータ拡張による分類性能の向上を図る．行き詰まり時のタグ付けの困難から，小暮ら(2)のデータセットは，ある時点のコーディング活動データから 10 分後に学習者がコードを完成することができるかを予測する問題に置き換えている．これは教師が想定する解答時間を経た後に一定の試行錯誤の時間を経てもコードを完成しない場合には行き詰まりとみなして支援を行うことを意図している．このデータセットでは，予測対象はコードが完成した時点のデータであるが，演習時間の大半ではコードが完成した時点以外のデータであるため，不均衡が生じている．従って，コードが完成した時点のデータを拡張して不均衡の改善を行う．

情報学系の大学のプログラミング演習からデータを収集した．演習課題は数値型の違いや繰り返し処理の動作などである．各学習者のコーディング活動を 1 分毎に収集すると共に，収集した各時点のソースコードに対してビルド・テストを実施し，プログラムの状態を付与した．

- 直前の時点からの編集の有無
- プログラムの文字数
- プログラムの行数
- ビルドをしたかどうか
- ビルド時のエラー数
- 実行をしたかどうか
- プログラムの状態(1.ビルドエラー，2.実行エラー，3.ロジックエラー，4.完成)

本研究では，プログラムの状態 1.2.3 を未完成，状態 4 を完成とし，任意の時点から 10 分後のプログラムの状態（完成・未完成）を予測することとした．収集したデータでは「未完成」が 1426 個，「完成」が 486 個となっている．

3. データ拡張

3.1 データ拡張手法

データ拡張手法は，Barros ら(3)と Volarić ら(4)の退

学者の予測モデルの研究で不均衡の改善に用いられた手法を参考にした．Barros ら[3]は SMOTE，ADASYN を，Volarić ら[4]は GAN によるデータ拡張を用いた．本稿では乱数ノイズの付与，SMOTE，ADASYN，GAN，GAN を拡張した WGAN の 5 つによる拡張を試みた．

3.2 データ拡張手法の実装

SMOTE，ADASYN は Python の imbalanced-learn ライブラリ[i]を用いた．GAN，WGAN は pytorch ライブラリ[ii]を使用して実装した．

GAN : 生成器・識別器ともに 5 層の全結合層，パラメータ更新頻度は 3:1, 最適化アルゴリズムに Adam, 損失関数にバイナリクロスエントロピーを使用した．バッチサイズ 8，最大エポック 20000 とした．

WGAN : 生成器・識別器ともに 5 層の全結合層，パラメータ更新頻度は 1:5，最適化アルゴリズムに RMSProup，損失関数には Wasserstein 距離を使用した．バッチサイズ 8，最大エポック 20000 とした．

4. データ拡張によるモデルの構築と結果

4.1 分類手法

先行研究[2]の分類手法に対して，拡張データを適用したことによる性能の変化を評価する．分類手法は決定木，SVM，ランダムフォレスト，ロジスティクス回帰，多層パーセプトロン（MLP）である．

モデル評価は収集データの各時点の 10 分後の状態が「未完成」であることを正とした場合と，「完成」を正とした場合の f1-socre を評価した．データセットは 4:1 の割合で学習データと test データに分割し，さらに学習データを train データと validation データに 4:1 の割合で分割して学習，評価を行った．

4.2 分類結果

表 1 に未完成分類性能，表 2 に完成分類性能を示す．分類手法と拡張手法，validation データと test データの違いによる 50 の組み合わせの内, 未完成分類性能では 11 の組み合わせで性能が改善され, 完成分類性能では 28 の組み合わせで性能が改善された. 完成分類性能の test データにて最もスコアの良い組み合わせは決定木と SMOTE の 0.62 であり，拡張前の 0.49 から 0.13 増加した．また同一の組み合わせ（決定木と SMOTE）の未完成分類においては validation と test が拡張前のスコアからそれぞれ 0.03 と 0.02 の減少となり同様の性能向上は見られなかった. また，その他，どの拡張手法・分類手法の組み合わせにおいてもあまり大きな向上は見られず，今回のデータセットとモデルにおいては，今回の拡張手法では十分な性能向上は得られなかった．

表 1　未完成分類性能

	評価データ	決定木	SVM	RandomForest	ロジスティクス回帰	MLP
拡張前	validation	0.92	0.89	0.96	0.88	0.89
	test	0.72	0.73	0.75	0.74	0.76
乱数での増幅	validation	0.91(-0.01)	0.80(-0.09)	0.96(-)	0.81(-0.07)	0.87(-0.02)
	test	**0.81(+0.09)**	0.72(-0.01)	**0.76(+0.01)**	0.72(-0.02)	0.73(-0.03)
SMOTE	validation	0.89(-0.03)	0.88(-0.01)	0.95(-0.01)	0.79(-0.09)	0.85(-0.04)
	test	0.70(-0.02)	**0.74(+0.01)**	**0.78(+0.03)**	0.69(-0.05)	0.70(-0.06)
ADASYN	validation	0.91(-0.01)	0.87(-0.02)	0.95(-0.01)	0.78(-0.10)	0.84(-0.05)
	test	0.65(-0.07)	**0.74(+0.01)**	**0.80(+0.05)**	0.70(-0.04)	0.72(-0.04)
GAN	validation	0.91(-0.01)	0.86(-0.03)	0.96(-)	0.86(-0.02)	0.88(-0.01)
	test	**0.76(+0.04)**	0.73(-)	**0.76(+0.01)**	0.73(-0.01)	0.72(-0.04)
WGAN	validation	0.91(-0.01)	0.86(-0.03)	0.95(-0.01)	0.86(-0.02)	0.89(-)
	test	**0.8(+0.08)**	**0.74(+0.01)**	**0.76(+0.01)**	0.74(-)	0.71(-0.05)

表 2　完成分類性能

	評価データ	決定木	SVM	RandomForest	ロジスティクス回帰	MLP
拡張前	validation	0.65	0.55	0.84	0.48	0.45
	test	0.49	0.39	0.24	0.34	0.42
乱数での増幅	validation	0.56(-0.09)	0.51(-0.04)	0.82(-0.02)	**0.54(+0.06)**	**0.55(+0.10)**
	test	**0.58(+0.09)**	**0.57(+0.18)**	**0.34(+0.10)**	**0.59(+0.25)**	0.42(-)
SMOTE	validation	0.64(-0.03)	0.17(-0.38)	0.83(-0.01)	**0.49(+0.01)**	**0.53(+0.08)**
	test	**0.62(+0.13)**	0.04(-0.35)	**0.52(+0.28)**	**0.43(+0.09)**	0.39(-0.03)
ADASYN	validation	**0.68(+0.03)**	0.17(-0.38)	0.83(-0.01)	**0.49(+0.01)**	**0.46(+0.01)**
	test	**0.5(+0.01)**	0.05(-0.34)	**0.53(+0.29)**	**0.49(+0.15)**	**0.55(+0.13)**
GAN	validation	0.56(-0.09)	0.47(-0.08)	0.84(-)	**0.49(+0.01)**	**0.62(+0.17)**
	test	0.32(-0.17)	**0.41(+0.02)**	**0.37(+0.13)**	**0.38(+0.04)**	0.28(-0.14)
WGAN	validation	0.58(-0.07)	0.49(-0.06)	0.82(-0.02)	**0.55(+0.07)**	**0.58(+0.13)**
	test	**0.54(+0.05)**	**0.40(+0.01)**	**0.35(+0.11)**	**0.40(+0.06)**	0.40(-0.02)

5. まとめ

本稿では，コーディング活動データにプログラムの状態（完成・未完成）を付与し，各時点から 10 分後の状態を予測するモデルを対象とし，データ不均衡のデータ拡張手法（乱数ノイズの付与, SMOTE, ADASYN，GAN，WGAN）による改善を試みた．データ拡張を適用した結果，今回のデータセットに対しては，決定木と SMOTE の組み合わせにおいて改善が見られたが，全体としては大きな改善には至らなかった．

参考文献

(1) Yamashita, K., Sugiyama, T., Kogure, S., Noguchi, Y., Konishi, T., Itoh, Y. “An Educational Support System Based on Automatic Impasse Detection in Programming Exercises”, Proceedings of ICCE2017, pp.288-295 (2017)

(2) 小暮悟, 池亀智紀, 野口靖浩, 山下浩一, 山本頼弥, & 小西達裕：“プログラミング演習中の学習者の演習行動履歴を用いた演習支援”. 人工知能学会全国大会論文集 第 37 回 , p. 1R4OS10a03 (2023)

(3) Barros, T. M., Souza Neto, P. A., Silva, I., & Guedes, L. A. “Predictive models for imbalanced data:A school dropout perspective”. Education Sciences, 9(4), p.275(2019)

(4) Volarić, T., Ljubić, H., Dominković, M., Martinović, G., & Rozić, R. “Data Augmentation with GAN to Improve the Prediction of At-Risk Students in a Virtual Learning Environment”. International Conference on Artificial Intelligence in Education ,pp. 260-265(2023)

[i] imbalanced-learn : ver.0.11.0 を使用

[ii] pytorch : ver.2.1.0 を使用