

プログラミング・ロジックを考慮した ソースコード推薦システムの検討

重田 智希 鷹野 孝典

神奈川県立大学 情報学部 情報工学科

1. はじめに

ソフトウェア開発において、プログラミング効率やソースコード再利用性を高めるためにAPIやソースコード推薦機能に関する提案がなされており[1][2]、プログラミング統合開発環境などに組み込まれている[3]。プログラミングでは実現する機能に応じてソースコードの実装方法に多様性が現れる。このため、ソースコード推薦においては、プログラミング・ロジックを考慮することで推薦するソースコードの候補数を削減することが望ましい。

本研究では、プログラミング・ロジックの文脈を考慮したソースコード推薦システムを提案する。提案方式は、ソースコードを対象とした機械学習において、プログラミング・ロジックの学習と予測を容易にするために、プログラム実行ステップを機能ラベル列として抽象化表現し、プログラミング・ロジックの分岐に対応したソースコード片の推薦を行う(図1)。

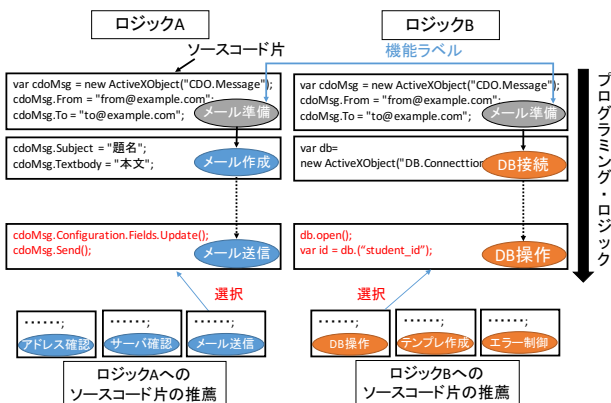


図1 プログラミング・ロジックに応じたソースコード片の推薦

2. 提案システム

提案システムでは、LSTM (Long short-term memory) を適用することにより、プログラミング・ロジックの学習・予測機能を実現する。ここで、本研究では、ある機能を表現するソースコード片を”機能コード片”，ソースコードを機能コード片の並びとして考えたときに現れる機能の並び順をプログラミング・ロジックと呼ぶ。

[プログラミング・ロジックの学習]

プログラミング・ロジックには、ある一定の規則性があり、同じ目的のために書かれるソースコードであれば、類似したプログラミング・ロジックを持つと仮定する。この仮定のもと、機能コード片に対応する機能ラベルを導入し、プログラミング・ロジック x を機能ラベルのステップ列 $P_x = \{f_1, f_2, f_3, \dots, f_n\}$ としてデータ表現することにより、LSTMによる学習を行う(図2)。ここで類似機能を持つ機能コード片には同一の機能ラベルを付与することで、ソースコード間の差異を吸収して扱うことができる。LSTMを適用した学習により生成した予測モデルを、プログラミング・ロジック予測モデルと呼ぶ。

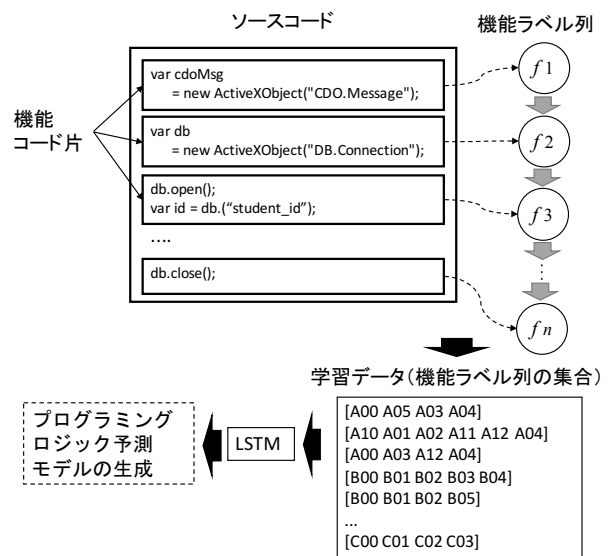


図2 プログラミング・ロジックの学習

A Recommendation System of Source Code based on Programming Logic

Tomoki Shigeta and Kosuke Takano, Department of Information and Computer Sciences, Faculty of Information and Technology, Kanagawa Institute of Technology

[プログラミング・ロジックの予測とソースコード推薦]

プログラミング・ロジック予測モデルは機能ラベルのステップ列 $P_x = \{f_1, f_2, f_3, \dots, f_{n-1}\}$ の入力に対して、次機能ラベル f_n を予測する (図 3)。

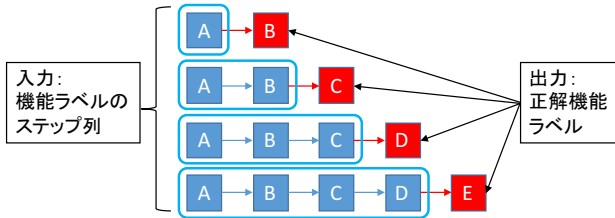


図 3 機能ラベルのステップ列に対する予測例

ソースコード推薦は下記手順で行う。

Step-1: 次機能ラベルの候補となる機能ラベル $C = \{c_1, c_2, c_3, \dots, c_k\}$ をスコア s_i ($i=1,2,\dots, k$) とともに抽出する。

Step-2: C の各機能ラベルをスコアに基づいてランキングする。

Step-3: 上位にランキングされた各機能ラベルに対応付けられた機能コード片を推薦する。

3. 実験

メール送信, ネットワーク処理, 深層学習に関して収集したソースコードを機能コード片に分割し, 23 種類の機能ラベルを手作業で付与した. 表 1 に機能ラベルの例を示す. 機能ラベルのステップ列集合 (表 2) を対象として LSTM を適用して学習することにより, プログラミング・ロジック予測モデルを生成した.

表 1 機能ラベルの例

機能ラベル	機能説明
A00	import 文 (メール関係)
A01	メール情報 (送信先/送信元) 設定
A02	メール内容設定
A03	メール情報/メール内容設定
A04	メールサーバ接続・送信
A05	エンコード処理
A12	メール作成
B01	IP アドレス設定
B02	バッファサイズ設定
C01	モデル作成
C02	学習層の追加
C03	学習処理の実行

表 3 にプログラミング・ロジック予測モデルに入力した機能ラベルのステップ列に対して, 出力された機能ラベルの予測結果を示す. 予測結果では, 確率分布が 1 位であった機能ラベルを選択している. 表 3 の結果より, [A00 (import 文)→A01 (メール情報設定)]に対して[A02 (メール

内容設定)], [A00 (import 文)→A03 (メール情報/メール内容設定)]に対して[A12 (メール作成)]のように, プログラミング・ロジックの文脈に応じた機能ラベルを予測できることが確認できる. 結果として, A03 に対応付けられる機能コード片 (表 4) を推薦可能となる.

表 2 学習した機能ラベルのステップ列の例

[A00 A05 A03 A04]
[A10 A01 A02 A11 A12 A04]
[A00 A01 A02 A12 A07 A08]
[A00 A01 A02 A07 A08 A09]
[A00 A03 A04]
[A00 A03 A12 A04]
[B00 B01 B02 B03 B04]
[B00 B01 B02 B05]
[C00 C01 C02 C03]

表 3 機能ラベルの予測結果の例

(入力)ステップ列	(出力)機能ラベル
A00 A05	A03
A00 A01	A02
A00 A03	A12
A10 A01	A02

表 4 機能コード片 (A03) の例

```
from_addr = 'spam@example.com'
to_addr = 'your_email_address@docomo.ne.jp'
msg = create_deco(from_addr, to_addr)
```

4. まとめ

実験結果から, 提案システムにより機能ラベル列の学習と予測が実現可能であることが確認できた. 今後の課題として, 大量のプログラミング・ロジックから学習・生成した予測モデルを用いてソースコード推薦性能を評価し, 提案システムの有用性を検証していく予定である.

参考文献

[1] 山本 哲男, ”回帰結合ニューラルネットワークを利用した API 推薦手法”, ソフトウェアエンジニアリングシンポジウム 2016 論文集, 2016, pp.25-33, 2016.

[2] Kim Mens, Angela Marie Lozano, ”Source Code-Based Recommendation Systems”, Recommendation Systems in Software Engineering, pp.93-130, 2014.

[3] Microsoft, ”Introducing Visual Studio IntelliCode”, available at <https://blogs.msdn.microsoft.com/visualstudio/2018/05/07/introducing-visual-studio-intellicode/>, 2018.