重要語抽出を用いたプログラム手続き記述の生成

Procedure description generation for Programing Learning using key word extraction

高橋 明義[*1]，椎名 広光[*2]
Akiyoshi TAKAHASHI[*1], Hiromitsu SHIINA [*2]
[*1] 岡山理科大学大学院総合情報研究科
[*1]Graduate School of Informatics, Okayama University of Science
[*2] 岡山理科大学
[*2] Okayama University of Science
Email: i18im03ta@ous.jp

あらまし：プログラムの理解や手順の理解を助けるために，ソースコードから手続きを自動生成を目的としている．手法としてはプログラムリストの条件や繰り返しのブロックごとについて，コメントのペアを LSTM を用いた Encoder-Decoder モデルで学習することで，学習対象にあったプログラムのコメント生成を行っている．しかし，手続き生成のための学習データ量を増やすことが難しいため，プログラムの問題文を取り入れ，手続きの生成に生かすことで，より対象に即したプログラムの手続き生成を行っている．
キーワード：アルゴリズム学習，手順生成，ニューラルネットワーク，LSTM

1. はじめに

大学のプログラミング教育において，プログラムの文法の学習だけでなくプログラムの手順やアルゴリズムの理解が問題となっている．これまで手続き学習の研究については，新開ら[(1)]は手作業によるアルゴリズムの学習についての研究があげられるが，我々は携帯端末での手続き学習システムを行ってきている．手続き学習システムについては，システム化により学習者の正答率や間違い方の傾向や課題の難易度を測ることが可能である．しかし，手続き学習システムについては，課題の作成の方が問題となる．特に，課題の手続きの作成では，多くの課題を人手で作成するのは困難であり，また，開発者による差が大きいという問題がある．

そこで，これまでプログラムのソースコードから手続きを自動生成の開発を行ってきた．プログラムの構造と自然言語の要約技術を利用した方法と，ニューラルネット翻訳技術を 1 行ごとのソースコードとコメントのペアデータに適用する手法を提案してきている[(2)]．

しかし，行単位でのコメント生成では，手続きの記述とみなすことには問題がある．また，学習者支援と考える場合，講義資料のような関連する情報があるので，単純にソースコードをコメントに変換するよりも関連情報に近づけたコメント生成も重要である．

本研究では，ソースコードの条件や繰り返しのブロック単位に対するコメントを，講義資料から抽出した重要語を加味した自動生成について行っている．また，通常のニューラルネット翻訳の場合，大量の学習データを必要とするのに対して，大学の情報系学科 1 年生での講義に使われた例や課題のプログラム 53 件程度でもある程度の生成ができているところがこの研究課題での利点である．また，評価としては，機械翻訳で使われて評価基準の BLEU[(3)]のスコアについても示す．

2. 手順生成システム

プログラミングには，変数の宣言，値の入力，計算，結果の出力などのようにいくつかの手順がある．このプログラミング手順テストは，プログラミングを手順化し，その手順をいくつかに分ける．そして分けられた手順をプログラミングの流れに合うように上から順に並び替えるテストである．このテストでは，講義に対する学習状況の把握に利用している(図 1).

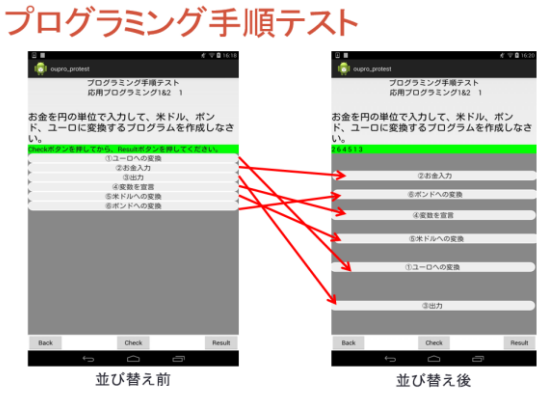

図 1 手順学習

例えば，為替の変換の問題「お金を円の単位で入力して，米ドル，ポンド，ユーロに変換するプログラムに変換するプログラムを作成しなさい.」の場合，本研究では，問題を解く手続きを図 1 のように分解し，その手順を並べ替える問題をテストするシステムを開発している．

3.　手続きの自動生成

本研究は，簡単に言うとソースコードからコメントや手続きへの翻訳を行っているということができ，主要な技術としてはニューラルネット翻訳の Encoder-Decoder モデルを利用している．処理としては，ソースコードとコメントの対を対訳データとするが，翻訳データにあたるコメントは，ソースコードに付されているコメントを抽出すること以外に，コメント部分を利用者の言語レベルに合わせた適用用コメントに入れ替えておくことも可能である．またそれによって生成されるコメントも学習したコメントのレベルに合わせるように生成可能である．以下に手順を述べる．

(1) コメントの付されたプログラムからコメントを抽出しソースプログラムとコメントに分割し，それぞれを単語に分割する．

(2) ソースコードとコメント(または適用コメント)をそれぞれ単語単位に分割し，if や for の処理のブロックごとのソースコードに単語のつながりの関係性を学習する．学習は，トークンに分割したソースコードを 1 単語ずつ LSTM に入力し，文脈情報を蓄積し，その後コメントを 1 単語ずつ LSTM に入力し，文脈情報を蓄積しながら単語を生成して学習を行う．

(3)重要語の重みを大きくしたコメント生成では，重要語について Encoder-Decoder モデルのコメント生成層からの出力の単語選択確率を大きくする処理を行う(図 2).

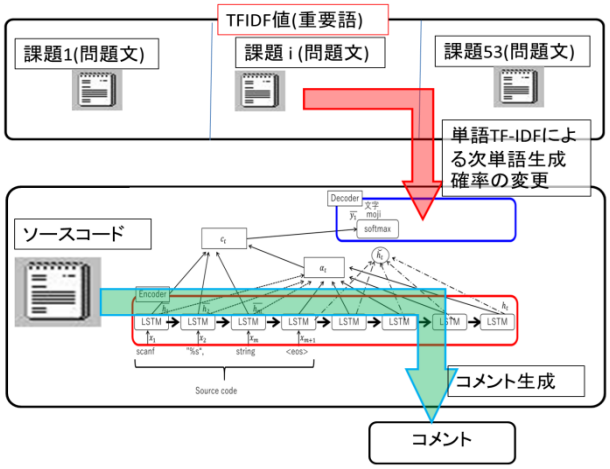

図２　重要語を優先するコメント自動生成

4.　生成されたコメントの評価

ソースコードとコメントの対訳を学習し，外部情報を利用した場合のコメントの生成例と BLEU を表 1 に示す．プログラムに対するコメントが正しくないものは BLEU の値が低くなっている．

外部情報の重要語を利用することによって，一部の生成コメントを変化させている．5 行目の生成を見るとプログラムと関係のない「配列の加算」というコメントから，文字列の操作を説明するコメントに変わっており，BLEU の値も高くなっていることがわかる．

表 1 コメントの生成例(代表的な例を選択)

行	ソースコード	生成コメント	BLEU
1	# include < stdio . h >	出力無し	
2	int main (void) {	出力無し	
3	char string [MAX]={'￥ 0 '}; int i ;	繰り返し 処理 繰り返す 処理	0
4	printf (" 文字 列 を 入力 し て く ださい "); scanf ("% s ", string); printf ("% s が 入力 さ れ まし た . ￥ n ", string);	入力 処理	1.0
5	for (i = 0 ; string [i] != '￥ 0 ' ; i ++) { if (string [i] >= ' a ' && string [i]<=' z '){ string [i] = string [i] - ' a ' +' A '; } else if (string [i] >= ' A ' && string [i]<=' Z '){ string [i] = string [i] - ' A ' +' a '; ￥ } }	(外部情報不使用) 配列 の 加算 A + B + C (外部情報使用) 文字 列 の 1 文字 ごと に 文字 を 判定 し て 数える	0 1.0
6	printf (" 変換 後 % s ￥ n ", string); return 0 ; }	結果 の 出力	1.0

外部情報を使わないコメント生成は，アンダーライン無し．外部情報を使うコメント生成は，アンダーラインあり．

5.　今後の課題

重要語については，外部情報の影響を如何に取り組むかという課題である．文脈を考慮する場合 attention を導入した Encoder Decoder モデルが考えられ，外部情報と文脈との関連を取り入れられるように改良が必要である．また，人手での利用数を増やしていきたい．

参考文献

(1) 新開他: “手作業の学習教材を活用したアルゴリズム教育の試み”,日本教育工学会第 31 回全国大会, pp.407-408 (2015)

(2) 高橋他: “問題文に合わせたプログラミングコメントの自動生成”，教育システム情報学会，特集論文研究会, Jsise Research Report Vol. 33, no.7 (2019-3), pp.51-55, (2019).

(3) 隅田他: “機械翻訳システム評価法の最前線”, 情報処理, 46 巻 5 号, pp.552-557(2005).