統計的機械翻訳を用いた自動コメント生成

小 田 悠 介[†1] グラム ニュービッグ[†1]

自然言語処理の一分野である統計的機械翻訳は，日本語と英語のような異なる言語間の対応関係を大量のデータから発見し，翻訳に利用する技術である．我々は，近年大量に得られるようになったソースコードと文書の間の関係性を解析するためのツールとして，統計的機械翻訳を利用する可能性に着目している．本稿ではその一例として，ソースコードに文単位でコメントを付与するシステムを統計的機械翻訳を用いて構築し，実際に生成されたコメントを評価した例を紹介する．

Automatic Comment Generation Using Statistical Machine Translation

Yusuke Oda [†1] and Graham Neubig[†1]

Statistical machine translation (SMT), an area of natural language processing, is specialized to find relationships between two languages and use them for translation process. We are focusing on SMT techniques to analyze relationships between numerous source codes and documents involved in. In this paper, we show an example of SMT-based source code analysis and the result of its evaluation by applying sentense-wise automatic comment generation.

1. はじめに

Web などを通してソフトウェアリポジトリやそれに伴う自然言語情報を大量に入手可能となったことで，これらに含まれる規則や対応関係を効率的に抽出する手法の需要が増している．特に自然言語の関係する研究では，コメントとソースコードの関係のマイニング[1)] や，ソースコードからの自動コメント生成[2)] などが挙げられる．これらの手法では統計処理または人手で注意深く作成されたルールを用いてプログラムと自然言語の関係性を捉えようとしている．しかし，統計処理のみではプログラムの細部が考慮されず，プログラム本来の複雑な情報は捨てられる場合が多い．一方ルールに基づく手法はプログラムの細部を意識した処理が可能な反面，システムの更新は基本的に人手であり，大規模化するほど管理の負担が無視できなくなる．

ルールに基づく手法は，システムの動作に統計処理を導入することによって人手による細部の管理を避けることができるようになる．このような手法は特に統計的機械翻訳 (Statistical Machine Translation: SMT) の分野で盛んに研究されている．SMT では日本語と英語など異なる言語間の対応関係を確率モデルによって定式化し，翻訳ルールの抽出や翻訳結果の生成を最適化問題や探索問題として扱う．このため対象のデータさえあれば機械学習によるシステムの自動構築が可能であり，人手による管理は原理的に必要ない．本稿では自動コメント生成システムを SMT の枠組みで構築し，生成されたコメントの評価を行うことで，ソースコードへ SMT を適用する有効性を考察する．

†1 奈良先端科学技術大学院大学
Nara Institute of Science and Technology

2. SMT による自動コメント生成

ソースコードからコメントへの翻訳モデルとして，本稿では Tree-to-String 翻訳[3)] と呼ばれる手法を使用した．Tree-to-String 翻訳では，原文を構文解析した結果である構文木が与えられた下で，翻訳結果として最も相応しい目的言語の単語列を推定する．形式的には，原文 $\boldsymbol{f}$，原文の構文木 $T_{\boldsymbol{f}}$，翻訳文 $\boldsymbol{e}$ として，式 1 の事後確率最大化問題として考える．

$$\hat{\boldsymbol{e}} \simeq \arg\max_{\boldsymbol{e}} \Pr(\boldsymbol{e}|T^*_{\boldsymbol{f}}) \tag{1}$$

図 1 に自動コメント生成システムの概略を示す．本稿では入力として Python，出力として日本語を対象とした．Tree-to-String 翻訳器を学習するためには，入力言語の構文木と出力言語の単語列が 1 文ごとに対応した対訳コーパスが必要である．本稿では Python ソースコードに 1 行ずつ人手でコメントを付与したデータ 722 行を用意し，10 分割の交差検証を行った．ソースコードの構文解析には Python 標準ライブラリ

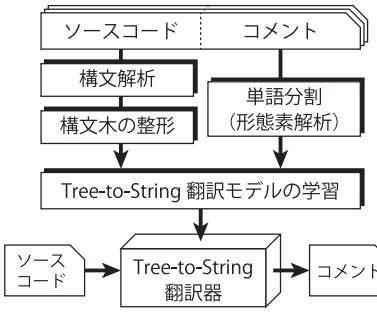

図 1 SMT に基づく自動コメント生成システム
Fig. 1 SMT-based comment generation system.

である ast を使用し，生成された構文木をいくつかの規則により Tree-to-String 翻訳に適した形へ変換した．日本語の単語分割には MeCab[4]，翻訳モデルの学習には Travatar[5] を使用した．アルゴリズムの詳細については省略するため，詳しくは自然言語処理研究会での報告[6] を参照されたい．本稿では当報告書の手法のうち，reduced 手法の結果のみについて掲載する．

図 2 に，生成されたコメントに対する Acceptability[7] と呼ばれる主観評価結果を示す．ここから，7 割以上の文に対して 5（文法的に正しく流暢）の評価を得ていることが分かる．一方，ほとんどの評価が 5 か 1（理解不能）に偏っており，翻訳精度が敏感に主観評価へ反映されることが分かる．

表 1 に実際の入出力を示す．最初の 2 例はいずれも剰余に関する式だが，0 との比較では「割り切れるなら」が特別に生成されていることが分かる．このような差異を人手で実現する場合，例外を発見する度にルールの更新を行う必要がある．一方 SMT に基づく手法では学習データに事例を含んでいればよく，システム作成に係る負担の大きな軽減となる．残りの 2 例は翻訳ミスにより誤ったコメントを生成しており，このような文をいかに抑制するかが今後の課題である．

3. 終わりに

本稿ではソフトウェア工学への自然言語処理の適用例としてソースコードに対する統計的機械翻訳 (SMT) の適用を挙げ，実際の例として文単位の自動コメント生成の実験について述べた．SMT の枠組みにより，多くの文に対して適切なコメントが生成可能であることを示した．今後の展望としては，本手法を使用してソースコード読解支援システムを構成し，実際の教育現場で試験を行う予定である．また，文よりも広範囲のデータ，例えば制御構造単位，ドキュメント単位で SMT の手法を適用する方法についても検討してゆく．

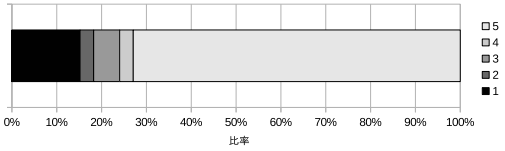

図 2 生成されたコメントの Acceptability 分布
Fig. 2 Acceptability distribution of generated comments.

表 1 生成されたコメントの例
Table 1 Examples of generated comments.

Python	`if x % 5 == 0:`
Comment	もし x が 5 で割り切れるなら
Accept.	5
Python	`if x % 5 == 1:`
Comment	もし x を 5 で割った余りが 1 であれば
Accept.	5
Python	`return a + b`
Comment	a と b に True，そうでなければ False を返す
Accept.	1
Python	`if m < N:`
Comment	mN 未満であれば
Accept.	1

謝　辞

本研究の一部は，頭脳循環を加速する戦略的国際研究ネットワーク推進プログラムの助成を受け実施したものである．

参 考 文 献

1) Tan, L., Zhou, Y. and Padioleau, Y.: aComment: Mining Annotations from Comments and Code to Detect Interrupt Related Concurrency Bugs, *Proc. ICSE* (2011).
2) Sridhara, G., Hill, E., Muppaneni, D., Pollock, L. and Vijay-Shanker, K.: Towards automatically generating summary comments for Java methods, *Proc. ASE* (2010).
3) Huang, L., Knight, K. and Joshi, A.: Statistical syntax-directed translation with extended domain of locality, *Proc. AMTA* (2006).
4) Kudo, T., Yamamoto, K. and Matsumoto, Y.: Applying Conditional Random Fields to Japanese Morphological Analysis., *Proc. EMNLP* (2004).
5) Neubig, G.: Travatar: A Forest-to-String Machine Translation Engine based on Tree Transducers, *Proc. ACL* (2013).
6) 小田悠介，札場寛之，ニュービッグ・グラム，サクティ・サクリアニ，戸田智基，　中村哲：ソースコード構文木からの統計的自動コメント生成，*IPSJ-SIGNL219* (2014).
7) Goto, I., Chow, K.P., Lu, B., Sumita, E. and Tsou, B.K.: Overview of the patent machine translation task at the NTCIR-10 workshop, *NTCIR-10* (2013).