Vol.2018-SE-198 No.4
2018/3/9

機械学習によるCoq上の命題論理の自動証明に関する研究

金原 雅典[a)]　佐藤 亮介[b)]　鵜林 尚靖[c)]　亀井 靖高[d)]

概要：プログラムの正しさを形式的に保証することは，そのプログラムに誤りがないということを確認する点で開発者にとって有用である．形式的に証明することは大変であり，自動化することでその手間を減らすことができる．そこで本研究では，機械学習を用いた命題の自動証明を試みる．その第一段階として，対象の論理を命題論理に限定して既存の機械学習によって学習・自動証明が可能かどうかを評価する．命題論理式は変数の数と，命題論理式を木構造にした時の深さを限定してプログラム上で生成を行なった．本研究での “証明” は，自然言語により書かれたものではなく，定理証明支援システム Coq 上での証明である．Coq とは，プログラムの正しさの証明や安全なプログラムを書くことに使用される証明支援システムである．学習は機械学習モデルの一つであるシーケンス変換モデルを用いて行う．Coq 上の証明をシーケンス変換モデルに学習させることにより，学習に用いるデータセットが十分な量あれば一般の機械学習の方法でも学習可能かどうかの評価を行なった．学習結果としては，データセットと同じ範囲の入力に対しては約 90% の割合で正しい出力を得られた．また，データセットの範囲を超える深さの命題論理式を入力した場合は正しい出力は約 53% の割合となった．

1. はじめに

プログラムの正しさを証明することは，プログラムに誤りがないということを保証し，膨大なテストケースを検査したり余計なデバッグ作業を無くす，あるいは減らすことに貢献する．プログラムの正しさを証明することで，開発の効率化及びプログラムの信頼性の向上が見込める．人力で形式的に証明する作業は手間がかかるため大変である．証明作業を自動化することができれば，プログラムの正しさを証明する手間を大幅に減らすことができる．

人間の知能や作業を代替する手法として，近年特に注目を集めているものの一つに機械学習手法がある．機械学習は，機械翻訳を用いた自然言語間の翻訳や，強化学習による囲碁プログラムなど，様々な分野で研究・応用され，多様な手法が考案されている．機械学習手法の一つである深層学習（Deep Learning）は画像分類 [7,11] や画像生成 [4,6] などの問題における特定のパターンを発見することが必要な分野で特に成功を収めている．証明の分野においても，シーケンスの入出力を対応させて学習する機械翻訳のように，命題とその証明の関係も命題から証明言語への変換であると見ることができ，命題とその証明をトークン列で表現すれば機械学習を適用することで自動証明が可能になると考えられる [10]．しかし，機械学習による証明の自動生成の研究はまだほとんど進んでいない．

本研究では，機械学習によって自動証明が可能かどうかを調査するために，まず第一段階として，対象の論理が単純であり，機械学習が学習を行うのに十分な量のデータセットが存在すれば，機械学習で学習可能かどうかの評価を行う．単純な論理であるため生成や確認がしやすく，充分な量のデータセットを用意することができるので，対象の論理は命題論理に限定する．命題論理の証明のデータセットを網羅的にプログラムで作成し，シーケンス変換モデル [1,2,12] を使用して学習させる．命題論理式は無限に存在するため，変数の個数を 3 個以下に，命題論理式の「深さ」を命題論理式を木構造にした時の深さとし，その深さを 3 以下に限定して網羅的に生成を行なった．

本研究で対象とする “証明” は自然言語により書かれたものではなく，定理証明支援システム Coq [8] 上で書かれたものである．Coq とは関数型言語の文法で定理や関数を記述することができる．Coq での証明構築は，Tactic と呼ばれる Coq に対する命令を対話的に適用していくことで，Coq 上の文法で記述された証明の副目標である「ゴール」やそのゴールが成り立つための「仮定」を変化させていくことで行う．例えば，命題論理の証明でゴールが “$P \vee Q$”，仮定が “H: $P \wedge R$”（H はその仮定に付けられた名前）という状態の時，ゴールの形が $\vee$ の時にその左側を新しいゴールにする Tactic “left” を適用するとゴールが P に変化す

a) kanahara@posl.ait.kyushu-u.ac.jp
b) sato@ait.kyushu-u.ac.jp
c) ubayashi@ait.kyushu-u.ac.jp
d) kamei@ait.kyushu-u.ac.jp

るといったように適切な Tactic の適用を行なっていく．もし，Tactic が適用できない場合は Coq はエラーを返し何も変化させない．本研究では，Coq の証明構築を機械学習モデルの一つであるシーケンス変換モデルを用いて学習させることを試みる．次の組み合わせをシーケンス変換モデルに学習させた．

入力　ゴールと仮定の列
出力　その時適用できる Tactic 全ての集合

Coq での自動証明は以下の手順で行う．証明したい命題論理の「ゴール」と「仮定」を学習させたシーケンス変換モデルに入力として渡し，その出力された Tactic を Coq 上で適用する．変化した結果，まだ証明すべきゴールが残っているならばそのゴールと仮定を再び入力として渡し Tactic を決定し，それを適用する．証明すべきゴールが残っていない場合は証明完了となる．シーケンス変換を用いた Coq 上の命題論理の自動証明の流れを**図 1** に図示する．

今回の実験で行なっていることは，ゴールと仮定からその時適用できる Tactic 全ての集合を学習させ，入力であるゴールと仮定を学習したモデルに与えることでその時適用できる Tactic 全ての集合を推論することであり，その証明構築において証明完了に至るための最も適切な Tactic を推論しているわけではない．本研究の本来の目的は定理を与えられた時に自動でその定理の証明を完了させることである．本来の証明では適用できる Tactic の中からもっとも適切な Tactic を一つ選択して適用しなければならないため，今回の実験で行なっている内容だけでは正しい証明構築を行うことはできない．図 1 のように自動証明を行うためには，出力されたその時適用できる Tactic の集合の中から，最も証明構築に適切な Tactic を選択して適用させる必要がある．適用できる Tactic の中から最も適切な Tactic を選択することは今後の課題である．今回の実験で確認する内容は，学習させたモデルが出力する Tactic がどのような結果を返すか，どのように学習データセットを与えれば良いかの今後の指針立てのための実験である．

学習したモデルが推論した出力が正しい結果を返しているかどうか，正しい出力を返していない場合どのような出力の傾向や問題点がありどのように改善していけばいいかの調査項目として，以下の二つの RQ を設定した．

RQ1　深さが 3 以下の命題論理式を入力として与えた時，正しく推論できるか
RQ2　深さが 4〜5 の命題論理式を入力として与えた時，正しく推論できるか

RQ1 では，深さ 3 以下，つまりデータセットに含まれる命題論理式と同じ深さの命題論理式を入力として与えた時正しく推論できるかを確認し，機械学習モデルがうまく学習されて動作しているかどうかを評価する．RQ2 では，深さ 4〜5，つまりデータセットには含まれない命題論理式を入力として与えた時正しく推論できるかを確認し，自動証明できる範囲を命題論理全体や命題論理以外の定理にも拡張できるかどうかを評価する．

学習に使用するデータセットを 56,075 個生成した．結果として，RQ1 ではランダムに生成した 10000 個の命題論理に対して 90% の割合で正しく推論できた．RQ2 ではランダムに生成した 7479 個の命題論理に対して正しい出力を返したのは 53% であり，半数以上を正しく推論できた．

以降，第 2 章では対象とする証明について説明する．第 3 章では利用したシーケンス変換モデルについて述べる．第 4 章では評価手法とその結果について述べる．第 5 章では妥当性への脅威ついて述べる．第 6 章では関連研究について述べ，第 7 章では本研究のまとめと今後の発展について述べる．

2. 学習対象とする証明

本章では，学習対象とする証明について述べる．まず，本研究で用いた命題論理について定義する．次に，その “証明” を構築する Coq について述べる．

2.1 命題論理式の定義

本研究では学習のデータセットとして，直観主義論理において恒真である命題論理の証明を用いる．直観主義論理を扱う理由は，Coq で扱う論理は直観主義論理にであるためである．古典論理が「命題が真であるか偽であるかのどちらかである」を前提とするのに対し，「真であるとは証明されることである」としたのが直観主義論理である．古典論理では恒真であっても直観主義論理では恒真ではない場合がある．例えば，古典論理では排中律（$P \vee \neg P$）や二重否定除去（$\neg\neg P \rightarrow P$）は成り立つが，直観主義論理では成り立たない．命題論理を用いた理由は，その論理の構造が単純であり，機械的に多くのデータセットが作成可能であるためである．対象の命題論理式は以下の BNF 記法で表せる．

$$f ::= x \mid \mathit{True} \mid \mathit{False} \mid f \ \wedge \ f \mid f \ \vee \ f \mid f \ \rightarrow \ f$$

x は変数，f は命題論理式である．結合子は論理積 $\wedge$，論理和 $\vee$，含意 $\rightarrow$ の 3 つを用いる．否定は対象の論理を単純化するためにここでは対象としないこととする．[*1]

定義 2.1 (命題論理式の深さ)　命題論理式 f の深さを $\mathit{depth}(f)$ と書き，以下のように定義する．

[*1] 否定 $\neg P$ は $P \rightarrow \mathit{False}$ と表すことができる．

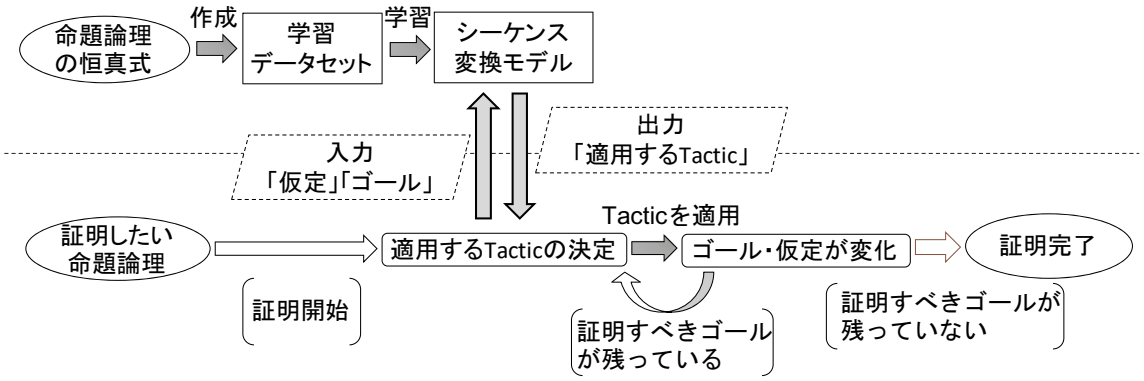

図 1　シーケンス変換を用いた Coq 上での自動証明

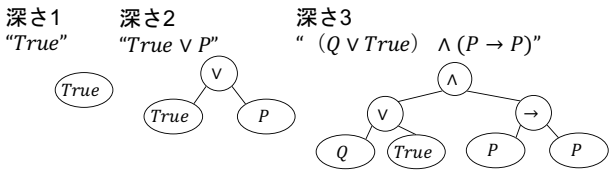

図 2　命題論理式の木構造とその深さ

$$
\begin{aligned}
depth(x) &= 1 \\
depth(True) &= 1 \\
depth(False) &= 1 \\
depth(f_1 \wedge f_2) &= max(depth(f_1),\ depth(f_2)) + 1 \\
depth(f_1 \vee f_2) &= max(depth(f_1),\ depth(f_2)) + 1 \\
depth(f_1 \rightarrow f_2) &= max(depth(f_1),\ depth(f_2)) + 1
\end{aligned}
$$

深さをわかりやすく図にしたものを**図 2** にあげる．変数の個数や深さを大きくするにつれて命題論理式の数は爆発的に大きくなるため，本研究の実験では変数の個数を 3 個以下，深さを 3 以下として生成を行った．

2.2　Coq

本研究では定理証明支援システム Coq [8] 上での証明を対象として学習を行う．Coq とは定理証明支援システムであり，証明作業を支援し正しさを保証するツールである．Coq は高階述語論理を扱うことができるため，多くの論理を表現できる．Coq は自動で証明を行ってくれるといったものではなく，あくまで人の証明作業を手助けするツールであり，計算における電卓のようなものである．Coq によって証明できたものには四色定理の完全に機械化された証明 [5] などがある．Coq では OCaml のような関数型言語の文法で定義や関数を記述することができる．例えば，自然数 n と m を加算する関数 $plus$ は，関数の定義を行うキーワード *Definition* を用いて以下のように記述できる．

$$Definition\ plus\ (n\ m\ :\ nat)\ :\ nat\ :=\ n\ +\ m.$$

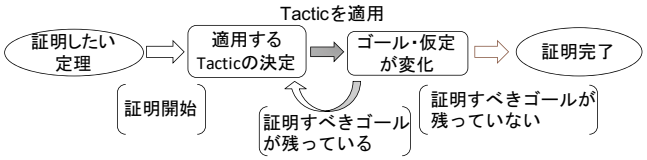

図 3　Coq の証明構築

証明構築は Tactic と呼ばれる Coq に対する計算命令を対話的に適応していくことで，Coq 上の記法で記述された証明の副目標である「ゴール」や「仮定」を変化させていき，命題の正しさを保証することができる．証明を開始した時点でのゴールは証明すべき命題そのものになる．ゴールが変化した結果，複数になった場合は，それぞれの新たなゴールに対して Tactic の適用を行なっていく．例えば，"$(P \wedge Q) \rightarrow (P \vee R)$" という命題を証明するときは**表 1** のように適切な Tactic を順に適用させて証明を行う．Tactic を適用させていった結果，「証明すべきゴールがない」という状態になれば証明終了である．Tactic が適用できない場合，Coq はエラーを返し，ゴールと仮定には何も変化を起こさない．Coq 全体の証明構築の流れは**図 3** に図示する．

証明を構築する上ではどのように証明するか，つまりどの Tactic を適用するかは証明者が選択する必要がある．適用する Tactic の選択は命題が複雑なものになるに連れて難しく手間のかかるものになる．適用すべき適切な Tactic を自動で推論することができれば，ゴールや仮定を変化させるなどの作業は Coq 自体が行なってくれるため，証明構築全体を自動化することができるようになる．

ゴールと仮定から適用する Tactic を選択することは，機械翻訳のような入力から対応した出力への変換であると考えられる．本研究では，入力である「ゴールと仮定」から出力である「適用する適切な Tactic」への変換だとして，機械学習により適用する Tactic を自動で推論できないかという実験を行う．

3.　シーケンス変換モデル

近年，再帰型ニューラルネットワーク（RNN）を含む

表 1　Tactic 適用例

ステップ	適用前		適用		適用後	
	ゴール	仮定	Tactic	操作	ゴール	仮定
1	$(P \wedge Q) \to (P \vee R)$	–	intro	“→” の前半を仮定に追加し，後半を新たなゴールにする	$P \vee R$	H: $P \wedge Q$
2	$P \vee R$	H: $P \wedge Q$	left	“∨” の左側を新たなゴールにする	P	H:$P \wedge Q$
3	P	H: $P \wedge Q$	destruct H	仮定 H の “∧” の右側と左側をそれぞれ仮定に追加する	P	H1: P, H2: Q
4	P	H1: P, H2: Q	apply H1	仮定 H1 をゴールに適用する	証明すべきゴールがない（証明完了）	

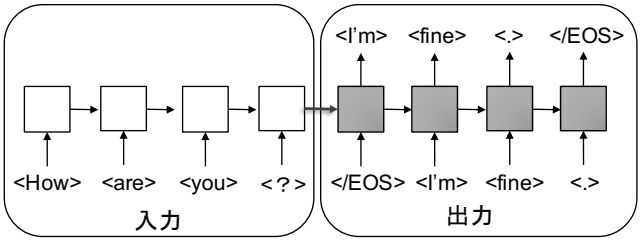

図 4　シーケンス変換モデルによる会話機械学習のイメージ

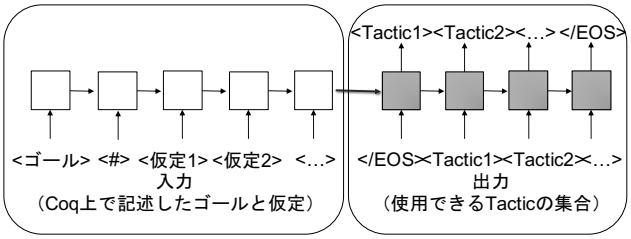

図 5　シーケンス変換モデルによる命題論理の学習のイメージ

ディープニューラルネットワーク（DNN）は多くの分野で成功を収めている機械学習の最も一般的な技術となっている．DNN は，画像分類 [7,11] や画像生成 [4,6]，音声認識 [3] など，その問題におけるあるパターンを発見する問題において特に有効な手段である．

本研究の機械学習の変換モデルとしてシーケンス変換モデル [1,2,12] を用いて学習を行った．シーケンス変換モデルとは，教師あり学習の一種である．入力を処理するエンコーダと出力を生成するデコーダの二つの RNN から成るモデルであり，自然言語間の機械翻訳や会話の機械学習などに利用されている．本研究では，TensorFlow[*2]の Seq2Seq を使用した．学習の際は，まず文章を単語毎にトークン化したものをデータセットとして用意する．学習モデルに入力と出力の組であるデータセットを渡し，**図 4** のように，エンコーダでは入力のシーケンスをベクトルに変換，そのベクトルをデコーダに入力し，出力シーケンスを生成するように RNN の学習を行う．学習させた後のモデルに入力のシーケンスを与えると，それに対応するような出力のシーケンスを推論して出力する．

本研究では，入出力の組を以下のように設定し学習させた．イメージとしては**図 5** のようになる．

入力　ゴールと仮定の列
出力　その時適用できる Tactic 全ての集合

出力である Tactic の集合は適用できる Tactic を “,” で区切ったものを用意する．また，機械学習手法は各種パラメータを調整することでより良い学習結果が得られることが報告されているが，本研究ではパラメータの細かい調整は行なっていない．使用したパラメータを**表 2** にあげる．

[*2] https://www.tensorflow.org

表 2　シーケンス変換モデルのパラメータ

パラメータ	設定した数値
学習率	0.5
学習率減衰	0.99
勾配クリップ基準	5.0
ミニバッチのサイズ	32
中間層の数	3
中間層各層のユニット数	512
入力の語彙数	4000
出力の語彙数	4000
bucketing に使う各 buket の大きさ	[(5, 10), (10, 15), (20, 25), (40, 50)]

4. 評価手法と結果

4.1 データセット作成

本研究のデータセットは以下のような手順で作成した．まず，第 2 章で定めた構文に従い Coq 上で扱える形式で恒真な命題論理式を生成した．範囲は変数の個数を 3 個以下，深さを 3 以下に限定した．網羅的に命題論理式の生成を行ない，恒真でないものを取り除いた．生成例を**表 3** にあげる．生成した恒真式の命題論理式数は 6,736 個となった．

次に，上記で生成した命題論理式からデータセットを作成した．上記で生成した命題論理式をゴールとし，仮定がない状態を初期状態として**表 4** にあげる Tactic をそれぞれ適用し，適用できた場合は変化したゴール及び仮定を新たな状態として再帰的に Tactic を適用した．ある Tactic が適用できた場合，その状態（ゴールと仮定）を入力，適

表 3 生成した命題論理式例

深さ	変数数	命題論理式
1	0	$True$
2	0	$True \land True$
	1	$True \lor P$
	1	$P \to True$
3	0	$(Q \lor True) \to (False \lor True)$
	1	$(P \land True) \lor True$
	1	$True \land (False \to R)$
	1	$(R \land True) \to True$
	2	$(False \land R) \to Q$
	2	$(R \to R) \lor (P \lor P)$

表 4 命題論理に使用する Tactic 一覧

Tactic	説明
intro	ゴールの式が含意の場合，$\to$ の前提を仮定に追加し，結論を新たなゴールにする
split	ゴールが論理積の場合，$\land$ の左右を別々に新たな 2 つのゴールにする
left	ゴールが論理和の場合，$\lor$ の左側を新たなゴールにする
right	ゴールが論理和の場合，$\lor$ の右側を新たなゴールにする
exfalso	ゴールを “$False$” にする
apply I	ゴールが “$True$” の場合，そのゴールを証明完了とする
以下の Tactic は仮定が複数ある場合それぞれに対して適用できる	
apply H	仮定 H が，ゴールと等しい時はそのゴールを証明完了とし，仮定 H が含意であり結論がゴールと等しい場合は仮定 H の前提を新たなゴールにする
destruct H	仮定 H の場合分けを行い，仮定 H の前提を新たな仮定として追加する

用できた Tactic を出力として保存した．生成例を**表 5** にあげる．生成したデータセット組数は 56,075 個となった．

以上の手法により生成したデータセットを用いシーケンス変換モデルで学習を行った．

4.2 RQ1．深さが 3 以下の命題論理式を入力として与えた時，正しく推論できるか

概要．データセットを学習させた後には，まずモデルの学習がうまくいっているかを確認する必要がある．この RQ ではモデルの学習が行われているかどうかを，深さ 3 以下，つまり学習データセットに含まれる命題論理式を与えることで推論がうまくいっているかどうかを確認する．

評価手法．推論できたかの確認は以下の方法にて行う．

(1) 深さを 3 以下に限定して，ランダムに命題論理式を 10000 個生成する．
(2) 生成した命題論理式からその時適用可能な Tactic の集合を作成し，これを正しい Tactic の集合とする．
(3) 生成した命題論理を学習させたデータセットの入力の形式に変換し，学習させたモデルに入力する．
(4) 出力された文字列を Tactic の区切り文字に使っていた “;” で分割し，その集合が正しい Tactic の集合と等しいかを確認し，正答率を出す．

結果．結果は以下のようになった．

命題論理式数 10000 個
正答数 9061 個
正答率 90.61%

9 割近くデータセットと同じ出力を返し，学習させた条件内ではおおよそ正しくゴールと仮定に対応する適切な Tactic の集合を返すことが確認できた．うまく推論できなかった出力結果を**表 6** にあげる．うまく推論できなかった出力結果を確認すると，「区切り文字が消滅していてうまく分割できない」，「適用できるはずの Tactic が存在しない」，「適用できないはずの Tactic を出力している（仮定 H が存在しないのに仮定 H を利用する Tactic が存在する）」，「無意味な区切り文字の連続」などが見受けられた．

> 深さが 3 以下の命題論理式を入力として与えた時，約 90% の割合で正しく推論できた．

4.3 RQ2．深さが 4，5 の命題論理式を入力として与えた時，正しく推論できるか

概要．今後，命題論理以外の定理の証明をするにあたって，データセットに含まれているものだけではなくデータセットに含まれてないものも，入力と出力の組み合わせのパターンから推論できる必要がある．この RQ ではデータセットに含まれていない命題論理式でも，深さ 3 以下の学習データセットで推論できるかを確認する．これにより，データセットに含まれていないものでも入力と出力の組み合わせのパターンから推論できるかを評価する．

評価手法．推論できたかどうかの確認は以下の方法にて行う．

(1) 深さを 5 以下に限定して，ランダムに命題論理式を 10000 個生成する．
(2) 生成した命題論理式の中から深さが 3 以下のものを取り除く．
(3) RQ1 の (2)〜(4) と同様にして正答率をだす．

結果．結果は以下のようになった．

命題論理式数 7479 個
正答数 3978 個
正答率 53.18%

表 5　生成したデータセット例

入力		出力
ゴール	仮定	
$True$	–	exfalso, apply I
$True \to True$	–	intro, exfalso
$True \wedge True$	–	split, exfalso
$True \vee False$	–	left, right, exfalso
P	H: P	exfalso, apply H
P	H: $True \to P$	exfalso, apply H
$Q \wedge R$	H: $False$	split, exfalso, destruct H
$(P \wedge False) \to (P \vee False)$	–	intro, exfalso
$False \to P$	H: $(Q \wedge P)$	intro, exfalso, destruct H
P	H0: $(Q \wedge P)$, H1: $False$	exfalso, destruct H0, destruct H1

適用できる Tactic を正しい形で返すという望ましい出力は全体の約 53% であり，約半数が正しく推論できた．結果の例は**表 7** にあげる．正しく出力できたものとできなかったものを比較すると，RQ1 でも見受けられたものの他に，仮定があればとりあえず “destruct” などを適用しようとしていることや，構文木の構成によらず，一番左に出現する演算子を優先的に見て判断しようとしているようであることが確認できた．例えば，命題論理式 $(True \wedge (R \wedge Q)) \to (((P \vee P) \wedge False) \to R)$ は一番外側の演算子は含意 $\to$ であり，Tactic の “intro” を適用すべきであるが，一番左の演算子 $\wedge$ に対応する Tactic の “split” を適用しようとしている．

深さが 4，5 の命題論理式を入力として与えた時は，約 53% の割合で正しく推論できた．

5. 妥当性への脅威

本章では，調査結果についての妥当性への脅威について述べる．

内的妥当性への脅威． RQ1 の正答率が 100% にならないことから，内的妥当性が低い可能性がある．これは，DNN が常に求める正しい形で応答を返すとは限らないという性質に起因すると考えられる．これは DNN がある程度の範囲で表現の揺れが許容される画像認識や自然言語間の分野では成功しているが，表現の揺れが許されない自動証明やプログラム生成の分野においてはあまり発展していない大きな理由の一つであると考えられる．

対応としては，Sekiyama ら [10] が学習モデルの出力を最も近い文法的に正しい形式に置き換えて処理をしているように，出力結果を正しい形式に変更し直す後処理を加えることが考えられる．また，Seq2Seq での学習をより適切に行うために，現在入力の命題論理式のシーケンスは中置記法で演算子を記述しているが，DNN や Seq2Seq の学習の性質を考慮しデータセットとして与えるシーケンスの形式を前置記法や後置記法などの別の形式に変更することが考えられる．それから，シーケンス変換モデルのパラメータを調整することも学習結果に大きな影響を与える一因であると考えられるため，パラメータの細かな調整も必要であると考えられる．

外的妥当性への脅威． RQ1 の正答率より RQ2 の正答率が低くなっていることから，外的妥当性が低いことが考えられる．この原因として，過学習が発生していることが挙げられる．Seq2Seq のチュートリアルの英仏間翻訳[*3] のデータセット量と比較すると，今回作成したデータセット数が 56,075 組に対し，サンプルのデータセット数が 646,605 組と，データセットの量がそもそも極端に少ないことがわかる．また，今回の実験では命題論理の範囲を限定して網羅的に生成したため，狭い範囲に偏って学習されてしまったと考えられる．

対応としては，データセット量を増やして学習するために，命題論理の変数の個数や深さを増やすことで範囲を拡張し，網羅的にではなくランダムに選択して作成することが考えられる．

構成概念妥当性への脅威． Coq の証明構築には Tactic は最も適切なものを 1 つ選択し適用する必要があるが，機械学習の出力の正答率を 100% にすることは難しいため，構成概念妥当性が低いことが考えられる．

自動証明を実現するためには，学習させたモデルだけではなく，機械学習と Coq の間で入力や出力を編集して受け渡しする外部のプログラムが必要であると考える．

6. 関連研究

Sekiyama らの研究 [10] では，直観主義命題論理に限定して，機械翻訳等に用いられるシーケンス変換モデル Seq2Seq [13] を用いて自動証明構築を行なっている．ここでは，対象の言語は，変数と含意 $\to$ からなる命題論理を

[*3] http://www.statmt.org/wmt15/translation-task.html, Europarl v7

表 6 RQ1：正しく推論できなかった出力例

入力		出力	
ゴール	仮定	正しい出力	推論した出力
$(True \wedge True) \vee Q$	–	left, right, exfalso	left , right exfalso
$P \rightarrow (P \wedge True)$	–	intro, split, exfalso	intro, split exfalso
P	H: $False \wedge P$	exfalso, apply H, destruct H	exfalso, destruct H
$False \wedge P$	H: $False$	split, exfalso, destruct H	split , exfalso
$False \rightarrow True$	–	intro, exfalso	intro , exfalso , destruct H
$False$	H0: Q, H1: $False$	apply H1, destruct H1	apply H0 , destruct H0

表 7 RQ2：出力例

入力		出力	
ゴール	仮定	正しい出力	推論した出力
正しく推論できた例			
$True \vee ((False \rightarrow (R \rightarrow P)) \vee False)$	–	left, right, exfalso	left , right , exfalso
$P \wedge (False \vee False) \rightarrow True)$	H: $False$	split, exfalso, destruct H	split , exfalso , destruct H
$True$	H: R, H0:$(P \rightarrow False) \rightarrow (False \rightarrow False)$	apply I, exfalso, destruct H0	exfalso , apply I , destruct H0
$(R \vee True) \vee R$	H: $P \wedge R$	left, right, exfalso, destruct H	left , right , exfalso , destruct H
正しく推論できなかった例			
$(True \rightarrow ((True \vee P) \vee Q)) \vee Q$	–	left, right, exfalso	, , , right exfalso
$(((P \rightarrow R) \wedge Q) \rightarrow P) \vee True$	–	left, right, exfalso	, , right exfalso
$(True \wedge (R \wedge Q)) \rightarrow (((P \vee P) \wedge False) \rightarrow R)$	–	intro, exfalso	split , exfalso
$False \wedge (True \rightarrow False)$	H: $(False \wedge (True \rightarrow False)) \rightarrow False$	split, exfalso, destruct H	intro , exfalso , destruct H
$True$	H: $False \rightarrow ((Q \wedge False) \rightarrow R)$	apply I, exfalso	apply I , destruct H
R	H: $False \wedge R$, H0: $(False \vee R) \rightarrow Q$	exfalso, destruct H	exfalso , destruct H , destruct H0

対 $\times$ と和 $+$ で拡張した単純な命題論理と，その命題論理に対応する単純型付きラムダ計算である．データセットには以下の入力・出力を与えている．

入力 抽象構文木として表される命題論理の命題 T
出力 対（$\times$，Pair）と和（$+$，Sum）で拡張された単純型付 λ 計算の抽象構文木として表される T の証明項 M

学習を行なったモデルが出力する証明項 M' は正しい証明項 M と似てはいるが，必ずしも文法的に正しい証明項であるとは限らないため，Myers [9] の編集距離に関して最も近い文法的に正しいトークン系列を生成するアルゴリズムを出力された証明項 M' に適用している．その後，Myers のアルゴリズムによって編集された証明項 M'' が正しい証明項であるかを Haskell プログラムに変換し，証明項 M'' の型を確認している．型確認の結果，M'' の型が入力の命題 T の型と一致する場合，カリー＝ハワード同型対応によって M'' は命題 T の正しい証明項であると結論づけることができる．学習結果としては，利用したベンチマークの問題のデータセットのうち，約半数が正しい証明を生成できたと述べている．また，出力された証明項のうち，誤っていた証明項 M' は構文木の編集距離に関して正しい証明項 M に近いことが確認されたと述べている．Sekiyama らの研究では証明全体を学習対象にしているのに対し，本研究では Coq による証明構築の 1 ステップごとを学習対象にしている．すなわち，本研究では一つの複雑な問題をより簡単な複数の部分問題に分割しているため，本研究の学習データの方が学習対象とする問題が証明全体を対象とするよりも簡単になっていると考えられる．本研究での学習結果では，“right, exfalso” と出力すべき所に対して “right exfalso” のように区切り文字 “,” がない状態で出力する場合があり，空白で区切られた場合は仮定 H を引数に取る Tactic “apply H” などの空白の区切りとの区別がつかないことがある．そのため，Sekiyama らの研究で行なったように，より正しい出力結果になるように出力に編集を加える操作を行うべきだと考えている．

Zhong ら [14] は，機械学習を用いて自然言語から SQL のクエリ文を生成する Seq2SQL の研究を行なっている．機械学習の対象が SQL のクエリ文であり，形式的に定まったシーケンスであるという点で参考にできると考えている．自然言語によって記述された質問文を入力として，SQL のクエリ式を出力するよう，拡張した Pointer Network を用いて学習させている．Pointer Network とは，Seq2Seq が学習したデータセットの語彙の中から出力を生成するのに対し，それに加えて入力の語彙も利用して出力を生成するモデルである．また，WikiSQL という Wikipedia の 24241 個のテーブルに分散された，手書きで注釈をつけた 80654 件の質問と SQL クエリのデータセットをリリースしている．強化学習を WikiSQL に適用することにより，Seq2SQL はベースラインであるセマンティックパーサーを上回り，実行速度を 35.9% から 59.4% に、論理形式の精度を 23.4% から 48.3% に向上させたと述べている．また，本研究では学習させる命題に適した学習モデルの調整を行なっていないため学習モデルを学習対象に合わせて拡張するという手法や，本研究では命題論理以外の一般的な命題に拡張する際のデータセットの量の問題を強化学習で解決できないかと考えているため，その点についても参考になると考えている．

7. まとめと今後の展開

本研究では，命題論理に対象を限定した命題の自動証明が可能かの検証を行うために，Coq 上の命題論理の恒真式の証明に対して機械学習モデルのシーケンス変換を用いて学習を行い，その評価を行った．結果としては，学習させたデータセットと同じ命題論理は 90% の割合で正しく推論でき，また，データセットよりも深さが大きい命題論理では 53% の確率で正しく推論できた．

今後の展開としては，データセット量を増やしたりシーケンスの形式を変更することで，変数や深さに制限のない命題論理の正答率を向上させることである．

命題論理の自動証明ができると確認できた場合，命題論理だけではなく関数を用いた一階述語論理へと学習範囲を拡張する．学習に用いるデータセットは Coq の標準ライブラリや，Coq 開発陣以外が証明し一般に公開されている Coq のライブラリ[*4]を用いて作成する．

教師あり学習のシーケンス変換による命題論理の自動証明ができなかった場合，強化学習などを用いて学習させることを考えている．

謝辞

本研究は JSPS 科研費 JP26240007 の助成を受けたものである．

[*4] https://coq.inria.fr/opam/www/

参考文献

[1] Bahdanau, D., Cho, K. and Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate, *CoRR*, Vol. abs/1409.0473 (online), available from ⟨http://arxiv.org/abs/1409.0473⟩ (2014).

[2] Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H. and Bengio, Y.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, *CoRR*, Vol. abs/1406.1078 (online), available from ⟨http://arxiv.org/abs/1406.1078⟩ (2014).

[3] Dahl, G. E., Yu, D., Deng, L. and Acero, A.: Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition, *IEEE Trans. Audio, Speech & Language Processing*, Vol. 20, No. 1, pp. 30–42 (online), DOI: 10.1109/TASL.2011.2134090 (2012).

[4] Denton, E. L., Chintala, S., Szlam, A. and Fergus, R.: Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks, *CoRR*, Vol. abs/1506.05751 (online), available from ⟨http://arxiv.org/abs/1506.05751⟩ (2015).

[5] Gonthier, G.: The Four Colour Theorem: Engineering of a Formal Proof, *Computer Mathematics, 8th Asian Symposium, ASCM 2007, Singapore, December 15-17, 2007. Revised and Invited Papers*, p. 333 (2007).

[6] Gregor, K., Danihelka, I., Graves, A. and Wierstra, D.: DRAW: A Recurrent Neural Network For Image Generation, *CoRR*, Vol. abs/1502.04623 (online), available from ⟨http://arxiv.org/abs/1502.04623⟩ (2015).

[7] He, K., Zhang, X., Ren, S. and Sun, J.: Deep Residual Learning for Image Recognition, *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778 (online), DOI: 10.1109/CVPR.2016.90 (2016).

[8] The Coq development team: *The Coq proof assistant reference manual*, 8.7.1 edition (2017).

[9] Myers, G.: Approximately Matching Context-Free Languages, *Inf. Process. Lett.*, Vol. 54, No. 2, pp. 85–92 (online), DOI: 10.1016/0020-0190(95)00007-Y (1995).

[10] Sekiyama, T., Imanishi, A. and Suenaga, K.: Towards Proof Synthesis Guided by Neural Machine Translation for Intuitionistic Propositional Logic, *CoRR*, Vol. abs/1706.06462 (online), available from ⟨http://arxiv.org/abs/1706.06462⟩ (2017).

[11] Simonyan, K. and Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition, *CoRR*, Vol. abs/1409.1556 (online), available from ⟨http://arxiv.org/abs/1409.1556⟩ (2014).

[12] Sutskever, I., Vinyals, O. and Le, Q. V.: Sequence to Sequence Learning with Neural Networks, *CoRR*, Vol. abs/1409.3215 (online), available from ⟨http://arxiv.org/abs/1409.3215⟩ (2014).

[13] TensorFlow: Sequence-to-Sequence Models, https://www.tensorflow.org/versions/master/tutorials/seq2seq#sequence-to-sequence-models.

[14] Zhong, V., Xiong, C. and Socher, R.: Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning, *CoRR*, Vol. abs/1709.00103 (2017).