グラフ構造を用いた機械学習による HOL の自動証明に関する研究

Towards automated proof of HOL by machine learning using graph structure

電気電子情報工学専攻　　AES1807　佐藤　貴一　（指導教員　西村　俊二）

Key Words: Theorem Proving, HOL ,Machine Learning ,GCN

１．緒　　言

定理証明支援系は数学的な定理証明や言語の意味論，型システムの健全性などの形でプログラムの正当性の検証に利用されている．この検証は，プログラムの挙動を関数や述語の形で記述し，正しく動作することを証明することにより行われる．検証が行われたプログラムでは，記述した性質が成り立っていることから，考慮している範囲に関してバグが存在しないと考えられる．定理証明の他にプログラムの正当性を確かめる手法としてテストが挙げられる．テストはテストケースを作成し，任意の値をプログラムに与えることでエラーを減らしていくため要求仕様がすべて網羅されているか判定するのが困難である．定理証明を用いた検証では，テストとは異なり目標となる仕様を数学の定理として証明することでプログラムの正当性を確かめるため，より網羅性の高い検証が可能となる．

しかし，定理証明を用いてプログラムを形式的に証明することは多くの労力を必要とする．定義や公理を選択し適用していくことにより定理を証明するが，難解な定理を証明する場合は，定義や公理の適用回数が膨大になる可能性がある．

定義や公理を適用した後の中間的な論理ステップをあらかじめ予測することで定理証明の自動化が可能になると考えられている[1]．もし証明の自動化が実現できれば，プログラムの正当性を確かめる手間を大幅に減らすことができ，かつ，網羅的な検証が可能となる．

定理証明を自動化する手段として，本研究では機械学習手法を適用する．証明したいステートメントに対してある中間的な論理ステップが有用であるか否か機械学習を用いて推定し，機械学習が定理証明の自動化に有効であるか検証を行う．証明で用いられる論理式は構文規則に従う性質を持っている．このような性質はデータ構造の一つであるグラフ構造で表すことが可能であるため，入力形式がグラフ構造の機械学習モデルである Graph Convolutional Networks(GCN)[3]を用いて有用性判別を行う．中間的な論理ステップの有用性判別における GCN の有効性を示すため，高階述語論理で記述された証明のデータセットを用いて実験を行う．

２．関連研究

2.1 定理証明の自動化に関する研究について

定理証明器 HOL の証明を機械学習を用いて自動化する研究として，Kaliszyk ら[1]は高階述語論理で記述された証明のデータセットを既存の機械学習モデルで学習し，証明の中間的な論理ステップが証明したいステートメントに対して有用であるかを判別している．有用性判別に使用されている機械学習モデルはロジスティック回帰と Convolutional Neural Network(CNN)，Recurrent Neural Network(RNN)である．Kaliszyk らはこれらのモデルの正解率を比較し，中間的な論理ステップの有用性判別にはどのモデル構造が適しているか考察を行っている．

2.2 関連研究の問題点　　Kaliszyk らが用いたモデルでは，証明の複雑な構造を捉えることは難しいと述べている．ロジスティック回帰は文脈を考慮せずに有用な論理ステップと非有用な論理ステップを分類することが可能であるが，71%と精度が十分でない．対して CNN は，文脈を考慮せずに論理ステップのパターンマッチを行うことで 82%の精度を出した．また，RNN は文脈を考慮することができ，精度は 83%となった．これは，単純なパターンマッチを行った CNN より 1 ポイント高い結果とはなったが，論理ステップの順序が有意に活用されていないことを示した．

３．提案手法

3.1 方法　　定理証明で用いられる論理式は構文規則に従う性質を持っている．このような性質はデータ構造の一つであるグラフ構造で表すことが可能であり，かつ，変数の名前を変更しても意味は変わらないという性質があることが Kusumoto らのプレプリント[2]で述べられている．Fig. 1 に論理式をグラフ変換した様子を示す．これらの性質を有効活用するために，機械学習モデルの一つである GCN を用いて証明の中間的な論理ステップの学習を行う．GCN の手法は[3][4]を参考にしている．

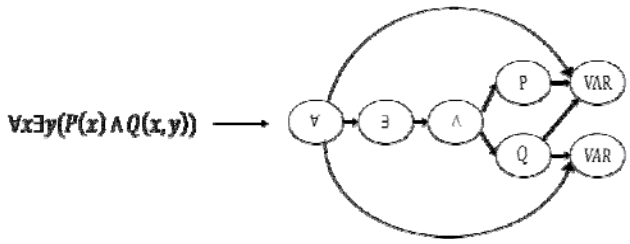

Fig. 1　Graph conversion

3.2　Graph Convolutional Networks　GCN はグラフ構造における畳み込み演算を定義し，CNN と類似した振る舞いをするモデルである．CNN の畳み込み層では，任意のサイズのカーネルを移動させながら畳み込みを行うことで特徴量を抽出することができる．この畳み込み演算は，画像などの規則性のあるデータに対して適用できる．一方，グラフ構造はノードの接続関係が不定形であるため CNN の畳み込み演算を適用することができない．そこで，近年，グラフ構造に対して畳み込みを適用する方法が研究されている[5]．ここでは，入力層，畳み込み層，プーリング層，集約層から構成される GCN モデルを説明する．

入力層

本研究で用いるデータはグラフ構造である．グラフ構造は，ノードとノード間を結ぶエッジから構成されるデータ構造である．グラフ構造のデータは GCN を容易に適用することが可能である．GCN ではグラフ構造データの各ノードが特徴ベクトルを保持している．Fig. 2 に GCN に入力として与えるグラフ構造の概念図を示す．ここで，v_iは各ノードの特徴ベクトル，M_iは入力ベクトルの長さを表現している．各ノードの特徴ベクトル$v_1, v_2, v_3, v_4, v_5, v_6$の隣接関係は，隣接行列によって表現される．

畳み込み層

画像データなどに用いられる畳み込みは，入力するデータの一部分をニューラルネットワークで計算し出力を得る．入力するデータの小領域を移動させながら畳み込み演算を行うことで，入力データの局所的な情報を変換処理し，特徴を抽出するフィルタとして扱うことが可能となる．入力したデータをそれらのフィルタと活性化関数をそれぞれ適用した特徴ベクトルに変換する．

GCN における畳み込み層では，各ノードに対して重み W を用いた特徴量の線形変換を行い，その後に各ノードの特徴ベクトルの更新を行う．任意のノードに対する特徴ベクトルの更新は任意のノードとそれに隣接しているノードの特徴ベクトルを加算した後に活性化関数を用いて，得られた出力を新たな特徴ベクトルとして更新する．

ノード数Nで構成されているグラフデータに対して畳み込み層を適用する際の概念図を Fig.3 に示す．ここでは 1 番目のノードに対して畳み込みを適用した場合を例に挙げて説明する．1 番目のノードに対して畳み込みを行う際，1 番目のノードおよび隣接しているノードである 2 番目，4 番目，5 番目の特徴ベクトル$\boldsymbol{v_1, v_2, v_4, v_5}$を用いて 1 番目の特徴ベクトルの更新を行う．特徴ベクトル$\boldsymbol{v_1, v_2, v_4, v_5}$に重みを掛け，そのベクトルを要素ごとに加算することで 1 つのベクトルにまとめ，**ReLU** 関数を適用することで得られた**conv** $\boldsymbol{v_1}$を新たな特徴ベクトルとして更新する．畳み込み層においてはこれらの処理をN個のノードすべてに対して行う．(1)に活性化関数として用いられる **ReLU** 関数を示す．ここで x は任意のノードにそれに隣接しているノードの特徴ベクトルを加算したものである．

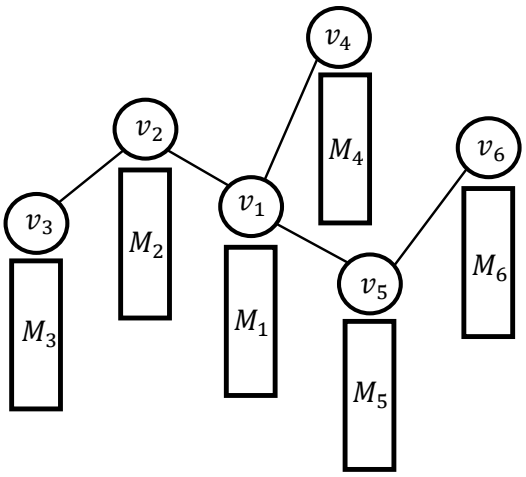

Fig. 2　Graph structure

$$\mathbf{ReLU(x)} = \begin{cases} \boldsymbol{x(x \geq 0)} \\ \boldsymbol{0(x < 0)} \end{cases} \tag{1}$$

ノード$\boldsymbol{i}$における更新式を(2)に示す．ここで $\boldsymbol{\sigma}$ は ReLU 関数などの活性化関数，$\boldsymbol{c_{i,j}}$は正規化のための定数，$\boldsymbol{N_i^k}$は$\boldsymbol{x_i}$から **k** 以下の距離でつながっている近傍ノードの集合，$\boldsymbol{W^{(l)}}$は$\boldsymbol{l}$層における学習重み行列を表すものとする．

$$h_i^{(l+1)} = \sigma\left(h_i^{(l)} W_0^{(l)} + \sum_{j \in N_i} \frac{1}{c_{ij}} h_j^{(l)} W_1^{(l)}\right) \tag{2}$$

プーリング層

プーリング層は，畳み込み層から出力された特徴ベクトルのサイズを圧縮して，代表値を残す処理を行う．代表値の決定方法として，区間内で平均をとる平均プーリング，区間内で最大となる値をとる最大プーリングなどがよく用いられる．プーリングは畳み込み層で抽出された特徴の位置感度を低下させる．即ち，画像データで考えたとき，画像内の物体が変形したり，位置が変わって

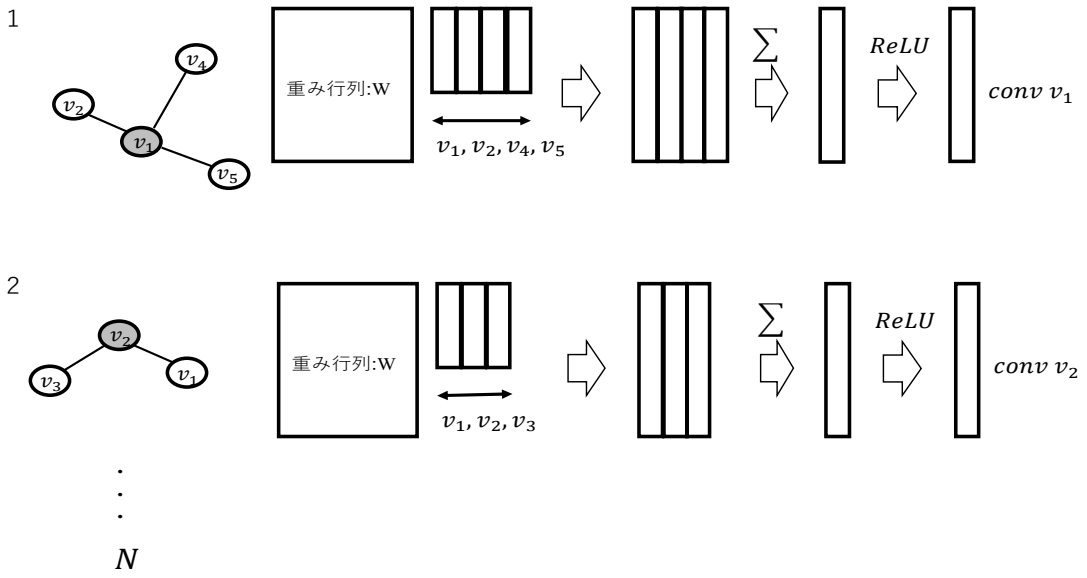

Fig. 3　GCN における畳み込みの概念図

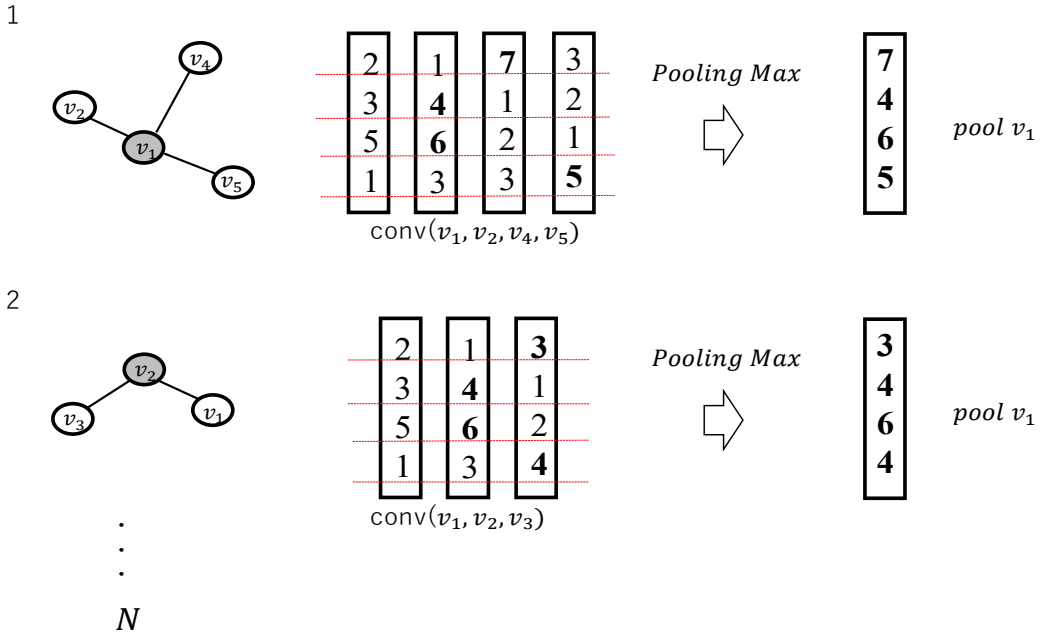

Fig.4　GCN におけるプーリングの概念図

いたとしても，その変化がプーリングによって無くなり，出力が不変のものとなる．これにより特徴を学習させることができる．GCN におけるプーリングでは，任意のノードに対する特徴ベクトルの更新は任意のノードとそれに隣接しているノードの特徴ベクトルを比較し，これらの代表値を要素としたベクトルを新たな特徴ベクトルとする．Fig. 4 にノード数 N のグラフデータに対してプーリング処理を適用する際の概念図を示す．ここでは，各要素における最大値を代表値とする最大プーリングを用いて 1 番目のノードに対してプーリング処理を行う場合を例に挙げて説明する．1 番目のノードに対してプーリング処理を行う場合，1 番目のノードおよび隣接しているノードである 2 番目，4 番目，5 番目の特徴ベクトルを用いて，1 番目のノードの特徴ベクトルを更新する．特徴ベクトル $\boldsymbol{conv\ (v_1, v_2, v_4, v_5)}$ を比較して各要素の代表値を選出し，これらを要素とした $\boldsymbol{pool\ v_1}$ を新たな特徴ベクトルとして更新する．これらの処理を N 個のノード全てに適用する

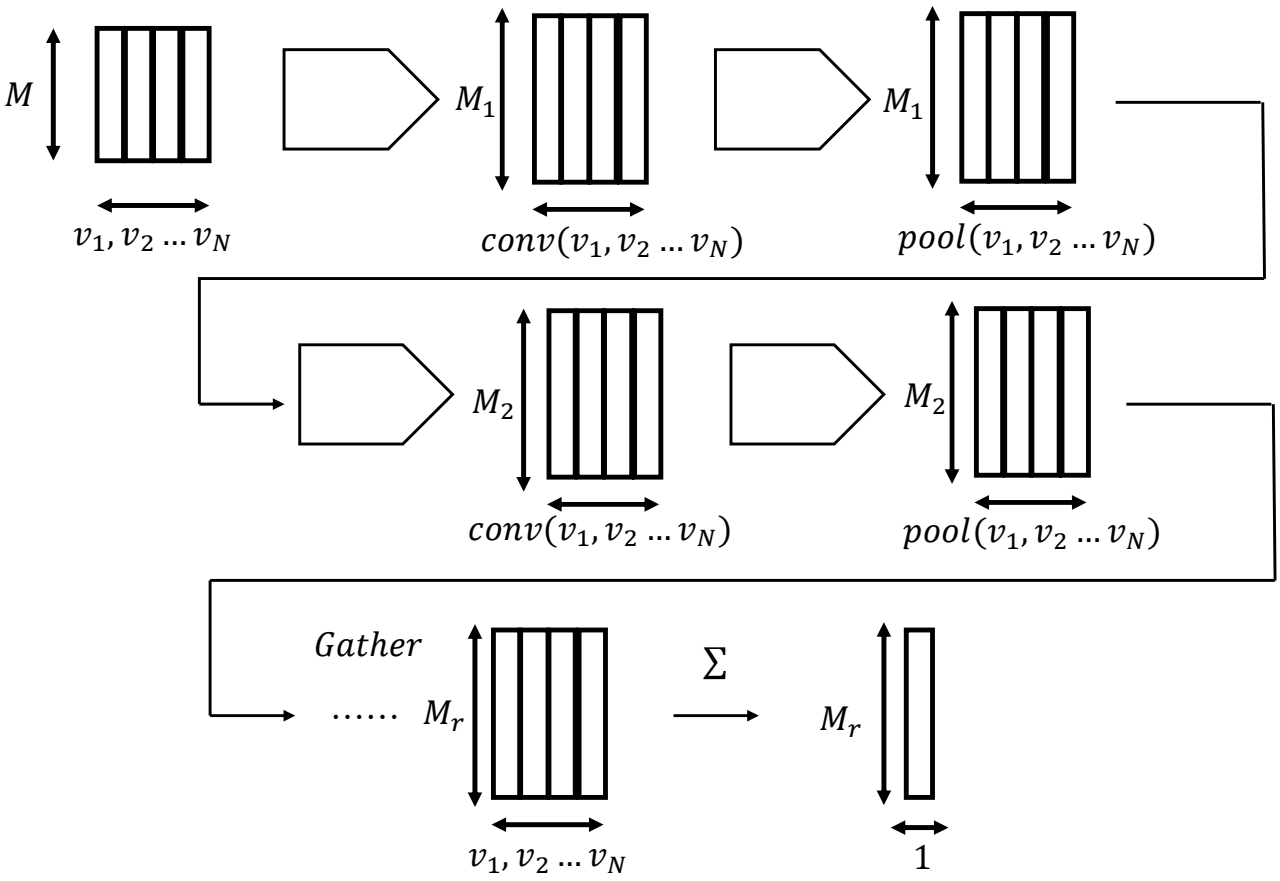

Fig5　GCN における集約図の概念図

集約層

集約層は畳み込み，プーリングの処理を数回繰り返した後に適用される．集約層では畳み込み層，プーリング層により更新された特徴ベクトルを各要素ごとに加算することで 1 つのベクトルにまとめる．これにより出力は単一ベクトルとなる．つまり，初期の入力ではノードそれぞれの特徴ベクトルであったものが最終的にはグラフ全体の特徴ベクトルとして出力する．Fig.5 にノード数 N のグラフデータに対して畳み込みとプーリングを数回繰り返した後に集約層を適用する際の概念図を示す．ここでは入力された特徴ベクトル数を N，各特徴ベクトルの長さは M とする．入力された特徴ベクトルは畳み込みとプーリングを数回適用され，新たな特徴ベクトルに更新される．これらの処理によって更新された長さ M_r の特徴ベクトルは，集約層で各要素ごとに加算されることで 1 つのベクトルにまとめられる．これによって，出力は M_r の長さをもつ単一ベクトルとなる．

重み学習

GCN では集約層において出力された特徴ベクトルに対して，重み W の全結合ニューラルネットワークを適用することで予測値を出力する．ここで訓練データと予測値の誤差を表す損失関数を定義する．GCN における学習の目的は，畳み込み層での重みを更新していくことによって損失関数を最適化することである．

その他の GCN

GCN には，大きく 3 種類が存在する．3 種類の GCN を以下に示す．

(i)Graph Convolutional Networks(GCNs)

GCNs の特徴は，グラフの周辺ノードからノードの潜在表現を得るとき，周辺ノードに適用する重みは同じである．加えて，ノードの潜在表現を得ることは可能だが，エッジの潜在表現を得ることはできない[3].

(ii)GNNs with edge embeddings

GNNs with edge embeddings は，ノードの潜在表現を得ることに加えてエッジの潜在表現を得ることが可能となった手法である．ノードの潜在表現からエッジの潜在表現を生成し，周辺ノードの潜在表現とエッジの潜在表現を合わせることで対象ノードの推定を行う．この場合，エッジの表現にノードの表現が依存することになり，計算が煩雑となり大規模なグラフには適用することが困難となる[6].

(iii)Graph Attention Networks

(ii)と比較して，エッジの表現力は弱まるが，高速な演算が可能となる手法である[7].

4．実験

4.1 比較手法　本研究では，Kaliszyk ら[1]が導入した高階述語論理のデータセットを用いて証明の中間的な論理ステップの有用性判別を行い GCN を用いた場合と Kaliszyk らが用いた機械学習のベースラインモデルの場合との分類精度(Accuracy)の評価を行う．判別性能を測る指標として，証明したいステートメントに対して GCN が有用な中間論理ステップを判別できた割合を用いた．判別したい 2 つの事柄を Positive と Negative の二つのクラスに分けることを考える. Positive を GCN に与え，正しく Positive であると判別できた数を TP(True Positive)，Positive を与え，Negative と間違えて判別した数を FN(False Negative)，Negative を与え，Positive と間違えて判別した数を FP(False Positive), Negative を与え, Negative と正しく判別できた数を TN(True Negative)とした. (3)に判別精度を求める式を示す.

$$\text{Accuracy}=(TP+TN)/(TP+FP+FN+TN) \quad (3)$$

また，Positive と判別されたものが実際に Positive であった程度を測定する指標である適合率(Precision)，実際に Positive であるものが Positive と予測された程度を測定する指標である再現率(Recall)，適合率と再現率の調和平均である f1 値を用いる．(4)に適合率を求める式を示す.

$$\text{Precision} = TP/(TP+FP) \quad (4)$$

(5)に再現率を求める式を示す.

$$\text{Recall} = TP/(TP+FN) \quad (5)$$

(6)に f1 値を求める式を示す.

$$f1 = 2\times\text{Precision}\times\text{Recall}/(\text{Precision}+\text{Recall}) \quad (6)$$

また, GCN はバッチサイズの個数ごとにノードを畳み込むためバッチサイズが精度に影響を与えると考えられる．バッチサイズと正解率に関するデータ取得を行う.

4.2 データセット　本研究で用いたデータセットは，多変量解析ライブラリ[8]と定理証明器 HOL Light[9]で記述された証明，ケプラー予想の形式的証明から生成されている．データセットに関する詳細を Table1 に示す．学習データ(Train Conjectures)が 9,999 個，テストデータ(Test Conjectures)が 1,411 個含まれている．証明済みのステートメントには中間的な論理ステップが関連付けられており，合計 2,209,076 の推測文の組がある．中間的な論理ステップには証明済みのステートメントに対して，有用(Positive)であるか否か(Negative)のラベルが付いている. 有用であるステップは合計 1,104,538 個あり有用でないステップも同様の数が含まれている．データセットの構成を Fig. 6 に示す．1 つのデータセットに証明したいステートメントと前提の集合，そして実際に証明で使用された中間論理ステップと使用されなかった中間論理ステップが含まれている.

Table1　HolStep dataset statistics

	Train	Test	Positive	Negative
Examples	2013046	196030	1104538	1104538
Avg.length	503.18	440.20	535.52	459.66
Cojectures	9999	1411	--	--
Avg. dependencies	29.58	22.82	--	--

また，証明したいステートメントと中間論理ステップを Bag of words(BoW)を用いて抽出し, 頻度順に並べ替えたものを特徴ベクトルとした．BoW は，ある文章における単語の出現回数を数えることにより，文章の特徴を表現して 1 つのベクトルにする手法である. その後, グラフ構造化し, GCN への入力データとしている.

4.3 モデル構成とパラメータ　本研究で用いるモデルは 2 層のグラフ畳み込みネットワークを使用し，隠れ層の活性化関数としては ReLU 関数，出力層の活性化関数としてソフトマックス関数を用いた. また，過学習を避けるためにドロップアウトを導入した．重み行列は，最適化のための勾配降下法には Adam[10]を用いた．これらは Python3.6 で実装し，ニューラルネットワークの構築は Tensorflow を用いた．モデルの構成を Fig. 7 に示す.

本実験では，平等に比較を行うためハイパパラメータの設定を Kaliszyk らのベースラインモデルと条

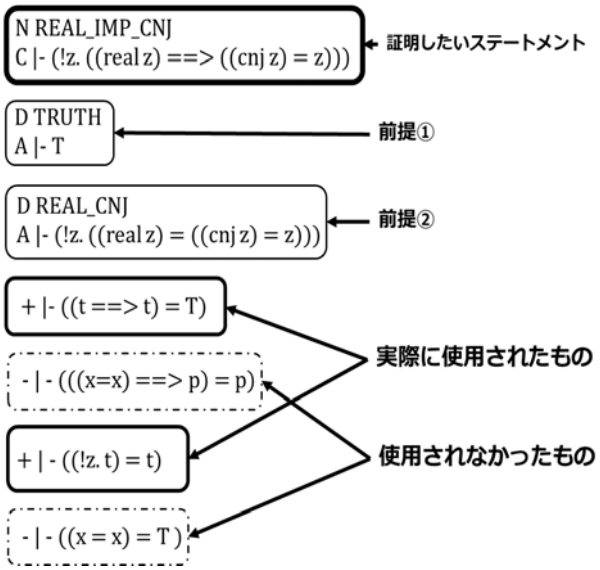

Fig. 6　Data set organization

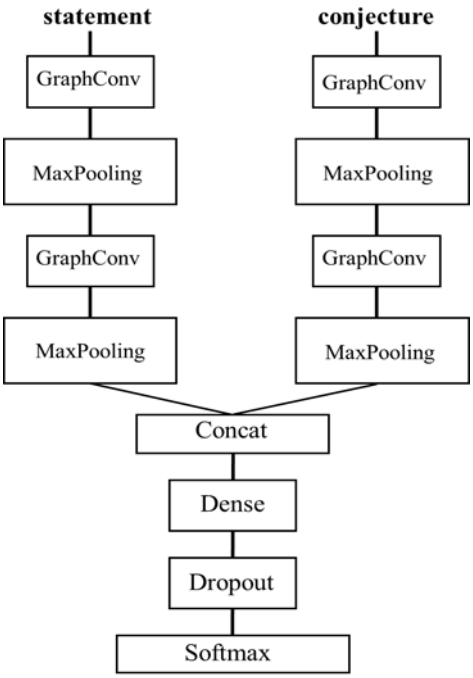

Fig. 7　Model Architecture

件を合わせて実験を行った．最大エポック数は 2000，学習率は 0.001，ドロップアウト率は 0.5 とした．また，バッチサイズは 128 とした．

5．結果

Table 2 に Kaliszyk らが用いたモデルと GCN の分類精度を評価したものを示す．結果から，ロジスティック回帰を 9 ポイント上回る性能を示している．この結果は，ロジスティック回帰のような比較的単純なアルゴリズムの機械学習モデルより分類性能に優れていることを示した．また，GCN の正解率は既存モデルの CNN を 2 ポイント下回り, RNN を 3 ポイント下回る性能を示している．Table3 には GCN の混同行列を示す．予測した 196,030 個の中間論理ステップの内，実際に正解した数は TP と TN の和である 157,629 個であった．混同行列の各要素から求められる各性能指標は Table4 に示す. 分類精度は 80%, 再現率は 85%, 適合率は 78%，f1 値は 81%となった．

Table2　Classification accuracy

model	CNN	RNN	Logistic regression	**GCN**
Acuraccy(%)	82	83	71	**80**

Table 3　Confusion Matrix

Actual＼Predict	**Positive**	**Negative**
Positive	82934[TP]	15081[FN]
Negative	23320[FP]	74695[TN]

Table 4　GCN performance details

Each performance index	Value(%)
Accuracy	80
Recall	85
Precision	78
f1	81

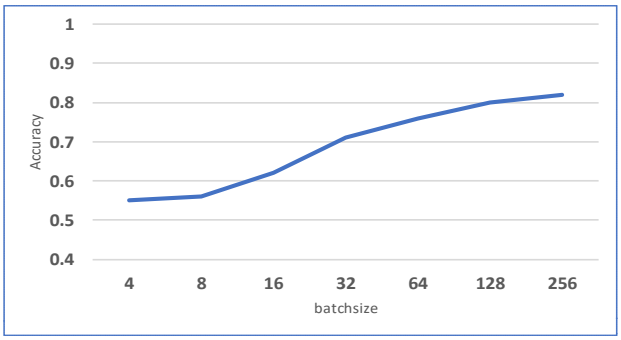

Fig. 8 バッチサイズと分類精度の推移

6．考察

6.1 各性能指標に関する考察　前項 5．の結果から，本研究で用いた GCN モデルではグラフ構造を有意に活用できていないことが示唆された．これは，本研究で使用した GCN がノードの潜在表現のみを得るモデルであり，エッジの潜在表現を得ていないことが原因として考えられる．エッジの潜在表現を得ることで，グラフ構造化した論理式を文字列としてではなく，不変的な構造として捉えることで, GCN のよるグラフ構造の認識精度が向上すると考えられる. グラフ変換する前のデータを BoW ではなく，新たな仮説として，データセットの事前学習を行い入力データをベクトルの埋め込み表現にし，グラフ構造に変換することで，論理式の不変的な構造を捉えられる可能性がある．

6.2 バッチサイズと正解率に関する考察　バッチサイズと正解率の推移を Fig.8 に示す．バッチサイズを大きくすることで，正解率の向上を確認することができた．特にバッチサイズを 256 に設定した場合，82%の正解率を得ることができた．GCN はバッチサイズの個数ごとにノードを畳み込みを行い特徴量を学習していく．したがってバッチサイズによってグラフの特徴量を学習する領域が決定し，その領域を大きくするほど大局的な特徴量を学習することができると考えられる．

7．結　　言

本稿では多くの労力を必要とする定理証明の自動化に関する研究の第一段階として，機械学習を用いて証明を構成する中間的な論理ステップを予測し，予測された中間的な論理ステップの有

用性判別を行った．有用性判別に使用した機械学習モデルは，入力形式がグラフ構造の GCN である． GCN の評価実験は既存のモデルである CNN と RNN との有用性判別の精度を比較した．GCN では 80%の精度を得ることができたが既存のモデルである CNN を 2 ポイント下回り，RNN を 3 ポイント下回る性能を示した．また，バッチサイズを従来モデルより大きくすることで正解率の向上を確認することができた．

今後の課題として，入力データをグラフ構造に変換する際，ノードの潜在表現に加えてエッジの潜在表現も獲得し，論理式を文字列として認識するのではなく，論理式の不変的な構造を認識する方法の検討が挙げられる．

参考文献

[1] Cezary Kaliszyk, Francois Chollet, Christian Szegedy, “"HOLSTEP": A Machine Learning Dataset For Higher-Order Logic Theorem Proving,” ICLR2017.

[2] Mitsuru Kusumoto, Keisuke Yahata, Masahiro Sakai, “Automated Theorem Proving in Intuitionistic Propositional Logic by Deep Reinforcement Learning,” 2018.

[3] Thomas N.Kipf,
Max Welling, “Semi-Supervised Classification With Graph Convolutional Networks,” 2016.

[4] 江口遼平, “グラフ構造に基づいたアルカロイド化合物の分類および代謝経路予測,” 2018.

[5] Gilmer,J.,Schoenholz,S.S.,Riley,P.F.,Vinyals,O.,andDahl, G.E., “Neural message passing for quantum chemistry,” 2017.

[6] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, Richard Zemel, “Neural Relational Inference for Interacting Systems,” 2018.

[7] PetarVelickovic*,GuillemCucurull*,ArantxaCasanova*, Adriana Romero,Pietro Lio,Yoshua Bengio, “Graph Attention Networks,” Published as a conference paper at ICLR 2018, 2017.

[8] J. Hrrison, “The HOL Light Theory of Euclidean Space,” 2013.

[9] J. Harison, “HOL Light,” 2011.

[10] Diederik P. Kingma, Jimmy Lei Ba, “A Method For Stochastic Optimization,” 2015.