人工知能学会研究会資料
SIG-FPAI-B502-19

全部分グラフ指示子に基づく決定木学習

Decision tree learning over all subgraph indicators

横山侑政[1*]　瀧川一学[2]
Yusei Yokoyama[1]　Ichigaku Takigawa[2]

[1] 北海道大学 工学部
[1] School of Engineering, Hokkaido University
[2] 北海道大学大学院情報科学研究科
[2] Graduate School of Information Science and Technology, Hokkaido University

Abstract: We propose an algorithm for learning decision trees over all possible subgraph indicators for a given set of graphs. Recent advances in machine learning provide several existing methods that can perform simultaneous feature learning from all possible indicators of subgraphs occurring in a given set of graphs. However these methods basically target linear models over all subgraph indicators. Thus decision tree learning for graphs would give foundations to learn non-linear models of subgraph indicators, potentially followed by ensemble methods such as gradient boosting and random forest. We first derive an upper bound for the decrease of two impurity measures by a supergraph of the given subgraph feature. Then we present an learning algorithm with subgraph search and pruning based on the upper bound and an enumeration tree of all possible subgraphs. Several experimental validations on two benchmark datasets are also reported.

1　はじめに

グラフは，自然言語処理 [2]，RNA 二次構造 [3]，低分子化合物の構造式 [1] などの知識処理に幅広く用いられている重要なデータ構造である．近年こうした科学分野のグラフデータが公共データベースに蓄積・整備されるようになり，有効な利活用が喫緊の課題となっている．特に，グラフデータからの教師付き学習は，生命科学や物質科学における構造活性相関や構造物性相関の定量的モデルとして機械学習分野で研究されており，より高精度で高効率な手法が求められている．

グラフデータからの教師付き学習では，特徴量として部分グラフの有無 (部分グラフ指示子) を用いることが多い．この場合，検査する部分グラフ特徴の選択が学習モデルの予測精度を左右する．しかし，適切な特徴集合はデータによって異なる．一方で，与えられたグラフデータに生起している全ての部分グラフを列挙することは，現実的には不可能である．例えば，300 個程度の典型的な分子グラフデータに生起する部分グラフの総数は，少なく見積もっても 100 億個に上る．従って，グラフカーネル法 [6] や ECFP 法 [5] など，対象となる部分グラフのクラスを予め発見的に制限する手法や，gBoost 法 [7] や Adaboost に基づく手法 [2] など，必要な部分グラフのみを効率的に探索しながら学習を行う方法が提案されている．後者のアプローチでは全ての部分グラフ指示子を探索候補にできるため，有効な特徴量を見逃すリスクがない．そのため，様々なデータに対して適応的に良い予測モデルを構築できる．しかし，従来手法では線形モデルの学習に限られている．

そこで，本論文では，全部分グラフ指示子を対象として非線形モデルである決定木を学習する手法を提案する．決定木学習は，2 値のブール変数である部分グラフ指示子との適合性が高く，効率良く簡潔な予測ルールを与えることができる．また勾配ブースティングやランダムフォレストを用いたアンサンブル学習の弱学習器として使うこともでき，将来的に発展させることが期待できる．

2　準備

本研究では，グラフデータからの教師あり学習問題を考える．すなわち，トレーニングデータは N 個のグラフで，個々のグラフ G_i にはラベル y_i が付与されており，任意のグラフ G に対してラベル y を予測する $y = f(G)$ なる関数 f をこれらのデータから学習することを目的とする．ここで対象とするグラフは，無向グラフで，各頂点と各辺にラベルが付与されているものとする．ま

*連絡先： 北海道大学工学部 知識メディア研究室
北海道札幌市 北区北 13 条西 8 丁目
E-mail: yuuseitori@art.ist.hokudai.ac.jp

た，部分グラフは連結なもののみを考える．

特徴量としては全ての部分グラフ指示子を用いる．グラフデータに生起する全部分グラフを $g_1, g_2, \ldots, g_K$ とすると，グラフ G の特徴ベクトル X は $X = (\mathrm{I}(\mathrm{G} \supseteq \mathrm{g_1}), \ldots, \mathrm{I}(\mathrm{G} \supseteq \mathrm{g_K}))$ となる．ここで $G \supseteq g_j$ は，グラフ G がグラフ g_j と同型な部分グラフを持つことを表し，各部分グラフ指示子 $\mathrm{I}(\mathrm{G} \supseteq \mathrm{g_j})$ は $G \supseteq g_j$ が真のとき 1 を返し，偽のとき 0 を返す．ただし，現実的なデータで生起する全ての部分グラフを陽に列挙するのは不可能であるため，提案手法では陽には構築しない．

決定木学習は多クラスの分類や回帰でも確立されているが，簡単のため本論文では 2 値分類 $y \in \{+1, -1\}$ のみを扱う．ラベル $y = +1$ の正例データ n_+ 個，ラベル $y = -1$ の負例データ n_- 個のデータ集合を $D(n_+, n_-)$ と表記し，集合 $D(n_+, n_-)$ の大きさ $n_+ + n_-$ を $|D|$ で表す．$D(n_+, n_-)$ を特徴「部分グラフ g_j を含むかどうか」で分割したとき，$\mathrm{I}(\mathrm{G_i} \supseteq \mathrm{g_j}) = 1$ となるデータの集合を $D_1(n'_+, n'_-)$，$\mathrm{I}(\mathrm{G_i} \supseteq \mathrm{g_j}) = 0$ となるデータの集合を $D_0(n''_+, n''_-)$ と表記する． 本稿では「$D(n_+, n_-)$ を部分グラフ g_j を含むかどうかで分割する」ことを「データ集合 D を部分グラフ g_j で分割する」と書く．

3 決定木

3.1 決定木学習

決定木は予測モデルの 1 つである．入力データの特徴ベクトル X に対して質問を積み重ね，予測したラベル y を出力する．図 1 に簡単な決定木の例を示す．

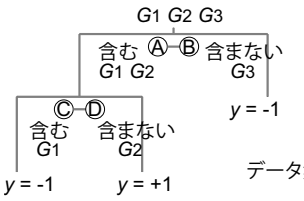

図 1: 決定木の例

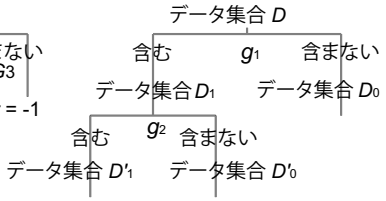

図 2: 決定木学習の例

決定木学習は，与えられたトレーニングデータから，関係関数 f を表現する決定木を構築する手法である．決定木の構築は，各ノードにおいてデータ集合 D を特徴ベクトルのひとつの要素を使って分割することで進められる．分割されたデータ集合 D_1, D_0 は，それぞれ子ノードへ送られる．子ノードは送られてきたデータ集合を分割する．このように，分割が再帰的に繰り返されて決定木が成長していく．データが十分に分割された時点で分割を止め，分割をしないノードは葉ノードとし，ノードに送られたデータの中で最も多いラベルを付与する．決定木学習の様子を図 2 に示す．

3.2 分割の基準

決定木学習の再帰的データ分割の目標は，各分割内のデータの不純度を下げることである．不純度の指標はいくつかあるが，本稿では誤識別率 $\mathrm{miss}(n_+, n_-)$

$$\mathrm{miss}(n_+, n_-) = 1 - \frac{\max(n_+, n_-)}{n_+ + n_-} = \frac{\min(n_+, n_-)}{n_+ + n_-}$$

および gini 指数 $\mathrm{gini}(n_+, n_-)$

$$\mathrm{gini}(n_+, n_-) = \frac{2n_+n_-}{(n_+ + n_-)^2}$$

を用いる．ある不純度指標 imp に対して,「分割前の不純度」$\mathrm{imp}(n_+, n_-)$ と「$D_1(a, b)$ と $D_0(c, d)$ への分割後の不純度の重み付き平均 (分割後の平均不純度)」

$$\Phi_{\mathrm{imp}}(a, b, c, d) = \frac{|D_1|}{|D|}\,\mathrm{imp}(a, b) + \frac{|D_0|}{|D|}\,\mathrm{imp}(c, d)$$

の差 $\Delta_{\mathrm{imp}} = \mathrm{imp}(n_+, n_-) - \Phi_{\mathrm{imp}}(a, b, c, d)$ が大きいものを良い分割とする．すなわち，データ集合 $D(n_+, n_-)$ を細分する分割の中では分割後の平均不純度 $\Phi_{\mathrm{imp}}(a, b, c, d)$ が最小になる分割が最も良い分割となる．本稿では Δ_{miss} および Δ_{gini} を考える．

3.3 分割の停止・葉の統合

分割を停止する基準はいくつかある．その 1 つが，ノードに送られてきたデータのラベルが 1 種類になった時点で終了させる，というものである．しかし，この基準で分割を停止すると，過学習が起こる．

過学習を解決する方法として，葉の統合がある．子ノードが全て葉ノードであるノードに注目し，不純度の減少が基準値 γ 以下の分割を取り止めるものである．つまり，

$$|D|/N \times \mathrm{imp}(n_+, n_-) < \gamma \qquad (1)$$

となるノードの葉ノードを統合する．

3.4 分割後の個数の制限

$D(n_+, n_-)$ を分割するとき，例えば $D_1(1, 0)$ と $D_0(n_+ - 1, n_-)$ に分割すると，過学習の原因になる．これを解決する方法として，分割後の個数 $\min(|D_1|, |D_0|)$ に制限をかける方法が 2 つある．

1 つは，トレーニングデータの個数 N に対する割合 τ で分割を制限するものである．つまり，

$$\min(|D_1|, |D_0|) < N \times \tau \qquad (2)$$

となる分割は採用しない．決定木学習で構築される決定木の全てのノードで制限の個数は同じになる．これを固定分割制限と呼ぶことにする．

もう 1 つの方法は，決定木の各ノードの分割前のデータの個数 $|D|$ に対する割合 τ で分割を制限するものである．つまり，

$$\min(|D_1|, |D_0|) < |D| \times \tau \tag{3}$$

となる分割は採用しない．これを割合分割制限と呼ぶことにする．

4 部分グラフの列挙

部分グラフ指示子を特徴とする場合，トレーニングデータに生起する部分グラフの有無を個々のグラフについて調べ，特徴ベクトルを構築する必要がある．本研究では，生起する部分グラフを調べるために，頻出部分グラフを列挙する gSpan アルゴリズム [4] を用いる．gSpan は N 個のグラフのうち m 個以上のグラフに生起する部分グラフ g_j を全列挙する．パラメータ m は minimum support と呼ばれる．図 3 のような表を考えると，縦 1 列の 1 の数は m 以上となる．本稿で対象とする全部分グラフ指示子の決定木学習とは $m = 1$ の場合のデータ表に対する決定木を得ることである．

特徴（部分グラフ） / グラフ	g_1 Ⓐ-Ⓑ	g_2 Ⓐ-Ⓑ-Ⓒ	g_3 Ⓐ-Ⓑ-Ⓒ-Ⓓ	g_4 Ⓔ-Ⓕ	…
G_1	1	1	1	0	
G_2	1	1	0	1	
G_3	1	0	0	0	
G_4	0	0	0	1	
⋮					

図 3: 部分グラフ指示子の陽な特徴ベクトル表

gSpan では，部分グラフを列挙するため図 4 のような探索木 (列挙木) を利用する．列挙木のノードは 1 つの部分グラフ g_j を表す．各ノードからは，そのノードの部分グラフの 1 つに対してエッジが張られる．

gSpan は，列挙木を深さ優先で探索する．各ノードの探索では，入力グラフ集合 $\{G_i \mid 0 < i < N\}$ に対して，そのノードの部分グラフ g_j を含むかどうかを調べる．この操作をノードが保持するグラフ集合の大きさが minimum support を下回るまで再帰的に繰り返す．

5 提案手法

提案手法では，列挙木を深さ優先探索しながら決定木学習を行う．列挙木の各ノード g で部分グラフ指示子を計算したあと，データ集合 D を部分グラフ g で分割したときの不純度の減少度を求める．同時に，子孫ノード g' における不純度の減少の上限値も計算し，可能なら列挙木を枝刈りする．5.1 節では上限値の計算を，5.2 節では枝刈りのアルゴリズムについて説明する．

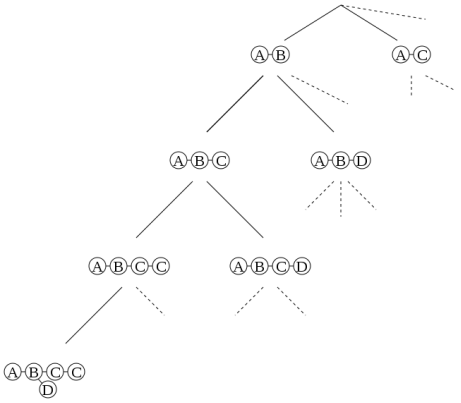

図 4: 部分グラフを列挙するための探索木

5.1 不純度の減少の上限値

グラフ g をグラフ g' の部分グラフとする ($g' \supseteq g$)．トレーニングデータ G_i について，$G_i \not\supseteq g \Rightarrow G_i \not\supseteq g'$ となることから，以下のことが言える．

今，データ集合 $D(n_+, n_-)$ を部分グラフ g で分割すると $D_1(a, b)$ と $D_0(c, d)$ に分かれたとする．このとき，g の任意の拡大グラフ $g'(\supseteq g)$ による分割は，変数 $0 \le x \le a$, $0 \le y \le b$ を用いて，「$D_1(a-x, b-y)$ と $D_0(c+x, d+y)$ に分かれ，不純度の減少は $\Delta(a-x, b-y, c+x, d+y)$」と書ける．これにより，不純度の減少について次の上限値の存在を示すことができる．

定理 1 (誤識別率を基準としたときの上限値)
データ集合 D を g で分割し $D_1(a, b)$ と $D_0(c, d)$ が得られたとする．このとき g のどんな拡大グラフ g' を考えても，g' による分割の不純度の減少量は

$$\max(\Delta_{\text{miss}}(0, b, c+a, d), \Delta_{\text{miss}}(a, 0, c, d+b))$$

を超えない．

証明. $\Delta_{\text{miss}}(a-x, b-y, c+x, d+y)$ が $(x, y) = (a, 0)$ または $(0, b)$ のどちらかで最大値をとることを示せば良い．Δ_{miss} の定義より，x, y の関数 $\min(a-x, b-y) + \min(c+x, d+y)$ が $(x, y) = (a, 0)$ または $(0, b)$ のどちらかで最小値をとることを示せば良いと分かる．

$\min(a-x, b-y) + \min(c+x, d+y) = \min(a+c, a-x+d+y, b-y+c+x, b+d)$ より，$0 \le x \le a$, $0 \le y \le b$ の範囲では $(x, y) = (a, 0)$ または $(0, b)$ で最小値 $\min(c, d)$ をとる． □

定理 2 (gini 指数を基準としたときの上限値)
データ集合 D を g で分割し $D_1(a,b)$ と $D_0(c,d)$ が得られたとする．このとき g のどんな拡大グラフ g' を考えても，g' による分割の不純度の減少量は

$$\max(\Delta_{\text{gini}}(0,b,c+a,d), \Delta_{\text{gini}}(a,0,c,d+b))$$

を超えない．

証明. 分割後の平均不純度 $\Phi_{\text{gini}}(a-x,b-y,c+x,d+y)$ が $(x,y)=(a,0)$ または $(0,b)$ のどちらかで最小値をとることを示せば良い．ここで

$$F(x,y) = \left(\frac{(a-x)(b-y)}{a-x+b-y} + \frac{(c+x)(d+y)}{c+x+d+y}\right)$$

と置くと，$\Phi_{\text{gini}}(a,b,c,d) = (2/|D|)\cdot F(x,y)$ となるので，$F(x,y)$ が $(x,y)=(a,0)$ または $(0,b)$ のどちらかで最小値をとることを示せば良いことが分かる．

$0 \le x \le a$, $0 \le y \le b$ に対し，$\partial F/\partial x \ge 0 \Leftrightarrow x < ((a+c)y + ad - bc)/(b+d)$ および $\partial F/\partial y \le 0 \Leftrightarrow y > ((b+d)x - ad + bc)/(a+c)$ が成り立つ．さらに，$x < ((a+c)y + ad - bc)/(b+d) \Leftrightarrow y > ((b+d)x - ad + bc)/(a+c)$ であるので，$\partial F/\partial x \ge 0 \Leftrightarrow \partial F/\partial y \le 0$ を得る．従って，$F(x,y)$ は x を減らせば減少するときは，y を増やせば減少する．同様にして $\partial F/\partial x \le 0 \Leftrightarrow \partial F/\partial y \ge 0$ も得られる．従って，$F(x,y)$ は x を増やせば減少するときは，y を減らせば減少する．$F(x,y)$ は直線上 $x = ((a+c)y + ad - bc)/(b+d)$ で最大値を持ち，直線の上下でこのような二領域を持つ．ゆえに，$(x,y)=(a,0)$ または $(0,b)$ のどちらかで最小値をとる． □

5.2 列挙木の枝刈り

決定木学習においてデータ集合 D を分割をするとき，表 3 の特徴 (列)g_j の中から，不純度の減少が最大になる特徴 (列)g_j を 1 つだけ選ぶ必要がある．データ集合 D の分割に用いるのはその 1 列だけであり，他の列は本質的に不要である．提案手法では，不純度の減少が最大にならないと分かる特徴のグラフ指示子を計算しない．その方法を以下で説明する．

今，列挙木のノード N_j を探索しているとする．ノード N_j の部分グラフ g_j で分割したときの不純度の減少 Δ_j は Δ_{miss} および Δ_{gini} の定義より計算できる．定理 1 または定理 2 を使えば，ノード N_j の子孫ノードで分割したときの不純度の減少の上限値 U を計算できる．ノード N_j の子孫ノードのうち，どのノードで分割をしても，不純度の減少値は上限値 U を超えない．

もし既に計算した不純度の減少 Δ_j の最大値 $\Delta_{\max}$ が上限値 U よりも大きい場合には，ノード g_j の子孫ノードを探索する必要はない．Algorithm 1 は，列挙木の枝刈りをしながら，最良の分割を与えるノード N_{best} を選ぶ．ただし，calcImpurityReduction(N_j) はノード N_j の部分グラフ g_j で分割したときの不順度の減少 Δ_j を返し，calcUpperBound(N_j) はノード N_j の子孫ノードでデータ集合 D 分割をしたときの，不純度の減少の上限値 U を返す．

列挙木の枝刈りを行うと，グラフ特徴表の特徴 (列)g_j のうち，決定木構築に不要な特徴 (列)g_j は計算せずに済む．枝刈りをせずに構築した決定木と，枝刈りを行いながら構築した決定木は同じものになる．つまり，全ての部分グラフ指示子 $\mathrm{I}(\mathrm{G_i} \supseteq \mathrm{g_j})$ $(0 < i \le N, 0 < g_j \le K)$ を使ったことになり，提案手法は全部分グラフ指示子に基づく決定木学習手法であると言える．

Algorithm 1 最良の分割を与えるノードの選択

```
N_best ← NULL
Δ_max ← 0
while 列挙木の未探索ノードがある do
  N_j ← 列挙木における次のノード
  Δ_j ← calcImpurityReduction(N_j)
  if Δ_max < Δ_j then
    Δ_max ← Δ_j
    N_best ← N_j
  end if
  U ← calcUpperBound(N_j)
  if U < Δ_max then
    N_j の子孫ノードを枝刈りする
  end if
end while
return N_best, Δ_max
```

6 実験

6.1 実験方法

提案する全部分グラフ指示子に基づく決定木学習の評価のため，3 つの実験を行う．評価は 10 fold cross validation による学習精度の推定値に基づいて行う．学習精度の尺度として，正答率 (accuracy, ACC) と ROC 曲線下の面積 (Area under the curve, AUC) の 2 つを用いる．また, 実行時間として，トレーニングデータを読み込んでから，決定木を学習し，テストデータのラベルの予測が終わるまでの時間を測り，平均値をとる．比較のため，データは既存研究 [7] で用いられた 2 種のベンチマークデータセット Mutag と CPDB を用いた．決定木学習時，データ集合 $D(n_+, n_-)$ の分割はノードに送られてきたデータのラベルの種類が 1 つになる $(\min(n_+, n_-) = 0)$ まで行い，そのあと葉の統合をする．

実験 1 では，Mutag のデータを対象として，パラメータを変化させたときの学習精度の変動を調べる．変化させるパラメータは，不純度の基準，分割後データ数の制限，葉の統合の基準の 3 つとする．不純度の基準は，誤識別率と gini 指数の 2 つを調べる．分割制限として，式 (3) の割合分割制限を採用し，τ は 0 から 0.50 の間で変化させる．葉の制限として，式 (1) を採用し，γ は 0 から 0.40 の間で変化させる．

実験 2 では，頻出部分グラフを全列挙し，これらの部分グラフ指示子に基づく特徴ベクトルを陽に得た後，通常の決定木学習を行う場合を検証する．これにより，全部分グラフ指示子に基づく提案手法の効果を確認する．minimum support m は 10 から 110 の間で，10 ずつ変化させる．不純度の基準として，gini 指数を採用する．分割制限と葉の統合では，式 (3) と式 (1) を採用し，$\tau = 0, \gamma = 0$ とする．

実験 3 では，グラフに対する教師あり学習の既存手法と学習精度を比較する．他手法の学習精度は参考文献 [7] から引用する．提案手法の学習精度は，各データで以下のものを用いる．Mutag は不純度の基準として gini 指数を用い，$\tau = 0.06$ で式 (3) の割合分割制限をかけ，葉の統合は行わない (式 (1) における $\gamma = 0$)．CPDB は不純度の基準として 誤識別率を使い，$\tau = 0.01$ で式 (2) の固定分割制限をかけ，葉の統合は式 (1) を採用し $\gamma = 0.06$ で行う．

6.2 データの性質

用いたデータの詳細を表 1 に示す．Mutag はデータは少なく，個々のグラフが大きめで，部分グラフの列挙が難しい．CPDB はデータが多く，個々のグラフが小さめで，部分グラフの列挙は易しい．実行時間 (秒) の項目は，Mutag，CPDB それぞれ minimum support $m = 10$ で gSpan を走らせたときの実行時間 (秒) を示している．

表 1: データの性質

	Mutag	CPDB
データ数	188	684
平均ノード数	17.9	14.1
平均エッジ数	19.8	14.6
実行時間 (秒)	67s	1s

7 結果

7.1 パラメータによる変化 (実験 1)

パラメータを変化させたときの学習精度の変化を図 5 に示す．不純度として誤識別率を用いたときは $\tau = 0.10$ $\gamma = 0.02$ で，**ACC 最高値 0.818, 実行時間 4.99 秒** であった．不純度として gini 指数を用いたときは $\tau = 0.06$ $\gamma = 0$ で，**ACC 最高値 0.871, 実行時間 1.92 秒** であった．

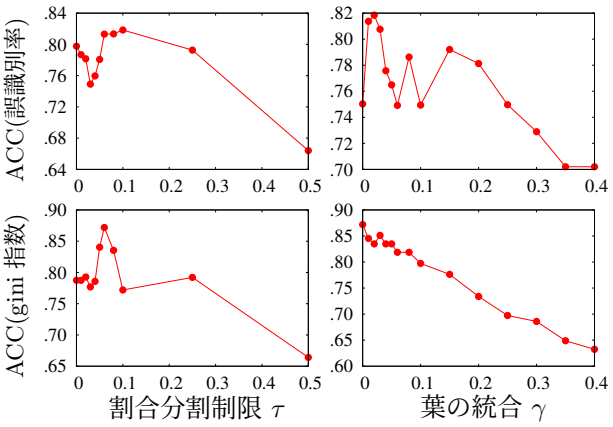

図 5: パラメータ変化による学習精度の変動

7.2 頻出部分グラフ特徴との比較 (実験 2)

枝刈りをせず，minimum support m を用いて部分グラフを制限したときの学習精度 (ACC) と実行時間の変動の様子を図 6 に示す．実行時間のグラフの y 軸は対数スケールである．minimum support $m = 10$ で **ACC 0.793, 実行時間 93 秒**，minimum support $m = 110$ で **ACC 0.632, 実行時間 0.2 秒**であった．m を下げると精度は良くなるが，実行時間が指数的に増大する．

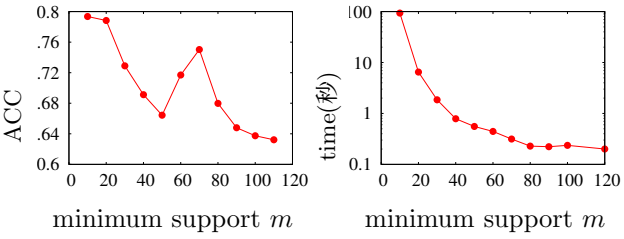

図 6: 枝刈りをしない場合の minimum support の変化による学習精度と実行時間の変動

7.3 既存手法との比較 (実験 3)

他手法と ACC，AUC を表 2 で比較する．既存手法は ACC，AUC は [7] から引用した．各列で最高値のものを太字にした．

表 2: 他手法との精度の比較

	Mutag		CPDB		出
	ACC	AUC	ACC	AUC	典
MGK	0.808	0.901	0.765	0.756	[7]
freqSVM	0.808	0.906	0.778	0.845	
gBoost	0.852	**0.926**	**0.788**	**0.854**	
提案手法	**0.871**	0.901	0.726	0.778	—

8 考察

8.1 実験結果の考察

実験 1 の結果からパラメータ τ も γ も大きくし過ぎると学習精度が落ちることが分かる．また，単純に凸なグラフとはならず，スパイクができるのは決定木らしいと言える．

実験 2 の結果から，minimum support m を下げると ACC は向上する傾向があることが分かる．しかし，枝刈りをしない場合は実行時間が指数的に増えるので，minimum support m を下げて学習するのは現実的には不可能である．提案手法を用いれば，minimum support $m = 1$ にしたときと同じ学習を短時間 (Mutag の場合 5 秒ほど) で行える．この点で提案手法は優れている．

実験 3 の結果から，提案手法は既存手法と比べ，部分グラフの列挙が難しいデータを対象にした学習に向いていると推測できる．その根拠として，提案手法の学習精度が既存手法と比べ，CPDB では悪いが Mutag では良いことが挙げられる．CPDB の部分グラフの列挙は易しいので，既存手法でも minimum support m を十分に下げて学習できる．Mutag の部分グラフの列挙は難しいので，既存手法では minimum support m を十分に下げることができず，学習精度があまり良くならない．しかし，提案手法を使えば minimum support $m = 1$ にしたときと同じ学習ができ，学習精度を向上させることができる．

8.2 割合分割制限の問題点

CPDB を対象としたときに割合分割制限をすると，実行時間が非常に長くなり実行不可能になる．詳細に調べてみると，葉ノードの近くの分割で，列挙木の枝刈りが行われていないことが分かった．原因として，以下の 3 点が考えられる．1 つ目に，葉に近いノードの分割では，ノードのデータが比較的似ているグラフになる．2 つ目に，葉に近いノードのデータ数は少ないため，割合分割制限では実質制限にならない．3 つ目に，割合分割制限は固定分割制限よりも決定木の葉ノードが増えやすい．以上 3 点より，列挙木の枝刈りが起こりにくくなり，列挙木を非常に深くまで探索することが頻発し，実行時間が長くなると考えられる．固定分割制限を使えば実行時間は短くなり，実行可能になる．

9 結論

本研究では，全部分グラフ指示子に基づく決定木の学習を可能にした．結果として，列挙木の枝刈りをしない既存の決定木学習手法よりも，学習精度は向上し実行時間の短縮も短くなった．決定木は勾配ブースティングやランダムフォレストを用いたアンサンブル学習の弱学習器 (base learner) として使えるので，今後はこれらの手法を用い，学習精度の向上を試みる．

謝辞

本研究は科研費 26330242, 26120503 の助成を受けたものです．

参考文献

[1] Takigawa, I., Mamitsuka, H.: Graph Mining: Procedure, Application to Drug Discovery and Recent Advances. *Drug Discovery Today*, Vol. 18, No. 1–2, pp. 50–57 (2013)

[2] Kudo, T., Maeda, E., Matsumoto, Y.: An Application of Boosting to Graph Classification. *NIPS 2004*, pp. 729–736 (2004)

[3] Karklin, Y., Meraz, R.F., Holbrook, S.R.: Classification of Non-Coding RNA Using Graph Representations of Secondary Structure. *Pacific Symposium on Biocomputing*, pp. 5–16 (2005)

[4] Yan, X., Han, J.: gSpan: Graph-Based Substructure Pattern Mining. *ICDM 2002*, pp. 721–724 (2002)

[5] Rogers, D., Hahn, Dm: Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, Vol. 50, No. 5, pp. 742–754 (2010)

[6] Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman Graph Kernels. *Journal of Machine Learning Research*, Vol. 12, pp. 2539–2561 (2011)

[7] Saigo, H., Nowozin, S., Kadowaki, T., Kudo, T., Tsuda, K.: gBoost: A Mathematical Programming Approach to Graph Classification and Regression. *Machine Learning*, Vol. 75, No. 1, pp. 69–89 (2009)