部分グラフ共起に基づくグラフ分類

Graph classification based on co-occurrence of subgraphs

岡崎　文哉 [1*]　瀧川　一学 [1,2]
Fumiya Okazaki[1]　Ichigaku Takigawa[1,2]

[1] 北海道大学 大学院 情報科学研究科
[1] Graduate School of Information Science and Technology, Hokkaido University
[2] 科学技術振興機構，さきがけ
[2] JST,PRESTO

Abstract: The graph classification is an important challenging task with many applications such as predicting the activity of chemical compounds. The presence or absence of subgraph patterns is often used as features for constructing a classifier. In this paper, we propose a graph classification method considering co-occurrences of subgraphs. This corresponds to an extension of existing linear methods to models including interaction terms of subgraph features. Subgraph indicators are 0-1 variables that have strong correlations due to the subgraph isomorphism between occurring subgraphs, and existing sparse linear models can be improved by taking these nonlinear interactions into account. We also present several variations to reduce candidate features increased by considering co-occurrence with some experimental verifications.

1　はじめに

グラフは，自然言語処理 [1]，RNA 二次構造 [2]，低分子化合物の構造式 [3] 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蓄積・共有・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 [4]．例えば，医薬品の候補化合物の構造は分子グラフとして表現でき，標的分子に対して生物活性があるか否かの予測は，各々がグラフで表現されたデータの 2 クラス分類問題として定式化できる．本研究では，グラフ G_i と正解ラベル y_i の対が N 個の訓練データ $\{G_i, y_i\}_{i=1}^N$, $y_i \in \{+1, -1\}$ として与えられる時，未知のグラフ G のラベル y を予測する分類器を構築する 2 クラスの教師付き分類問題を扱う．ただし，各々のグラフ G_i および G は，頂点や辺に離散ラベルが付与されている連結無向グラフとする．

グラフ分類の汎用の特徴量としては部分グラフの有無（部分グラフ指示子）を用いることが多い．この時，各々のグラフは各部分グラフを持つか持たないかの 0-1 特徴ベクトルで表現される (図 1)．特徴量としてどのような部分グラフを調べれば良いかは実際の学習結果に大きく影響を与えるが，訓練データに生起しているすべての部分グラフを列挙することも現実的に不可能である．そこで，グラフカーネル法 [5] や ECFP 法 [6] などの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が提案されている．一方，gBoost 法 [7] や Adaboost に基づく手法 [1] では，生起している部分グラフの中から分類に必要な部分グラフを適用的に探索・発見し，スパース正則化を用いて特徴学習と同時に分類器を構築していく方法が提案されている．このアプローチでは有効な特徴量自体もデータから学習でき，様々な種類のグラフデータに対する高い汎用性が期待できるため，本研究ではこのアプローチに着目する．

従来の特徴学習アプローチは，各々の部分グラフの有無を単一の特徴量とする弱学習器を Boosting で加法的に追加する枠組に基づいている．したがって，結果として構築される予測器は，部分グラフの有無の 0-1 変数に対して線形のモデルとなる．しかし，部分グラフの有無を示す 0-1 変数は，部分グラフ同士の包含関係による強い特殊な相関を持ち，また 0-1 変数であるため有限値 (多変量ではブール超立方体の端点のみ) しか取り得ないなど，スパース正則化が暗に前提とする条件としては極めて悪条件となり，線形モデルには改

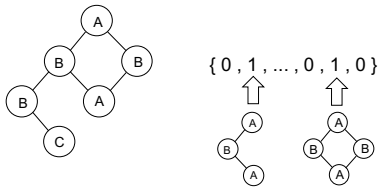

図 1: グラフに対する 0-1 ベクトル表現の例

*連絡先：北海道大学 大学院 情報科学研究科
〒 060-0814 札幌市北区北 14 条西 9 丁目
E-mail: fokazaki@ist.hokudai.ac.jp

善の余地があると考えられる．

本稿では，gBoost 法 [7] を拡張し，部分グラフとその共起を考慮した学習モデルを提案する．2 つの部分グラフの共起の有無は，各々の部分グラフの有無の交互作用項と等価となり，線形モデルは 2 次の交互作用を加えた多項式モデルへと拡張される．こうした交互作用の有効性については，Factorization Machines[8]，交互作用モデルの検定 [9]，複数の遺伝子変異の共起の薬剤効果への影響 [10]，化合物における部分構造共起 [11] など様々な有効事例が報告されており，グラフ分類においても有用な拡張が期待できる．

2 準備

本節では，用いる記法の定義および既存手法である gBoost アルゴリズム [7] の概要を説明する．入力として N 個のグラフとそれぞれのクラスラベル $\{G_i, y_i\}_{i=1}^N, y_i \in \{+1, -1\}$ が訓練データとして与えられた時，訓練データのグラフ集合 $\{G_i\}_{i=1}^N$ に少なくとも 1 回は出現する全ての部分グラフの集合を $\mathcal{T}$ とする．この集合 $\mathcal{T}$ は調べるべき候補部分グラフの全ての集合であるが，各々の G_i の全部分グラフの集合の共通集合と等しく，現実的なグラフデータでは組合せ的に巨大な集合となり，明示的な全列挙や数え上げ ($|\mathcal{T}|$ を求めること) は困難である．ただし，多くの実問題では大きな部分グラフは通常分類問題に寄与しないため，$|\mathcal{T}|$ としてサイズ制約 (例えば辺数 σ 以下の部分グラフ) を課す場合も多く，以降はそのような場合も含めた候補集合とする．

調べる部分グラフの候補集合 $\mathcal{T}$ に対して，各グラフ G は $|\mathcal{T}|$ 次元の 0-1 ベクトル $\boldsymbol{x} = (x_1, x_2, \ldots, x_{|\mathcal{T}|})', x_i = I(t_i \subseteq G), \forall t_i \in \mathcal{T})$ で表現できる．ここで，$I(\cdot)$ は括弧内の真偽が真なら 1，偽なら 0 になる指示関数である．以降は簡単のため，有無を調べる部分グラフ t を添字に用いて，誘導される 0-1 変数を $x_t = I(t \subseteq G), t \in \mathcal{T}$ と表す．この時，各 t に対して Boosting の基底となる弱学習器を以下で定義する．

$$h(\boldsymbol{x}; t, \omega) = \omega(2x_t - 1)$$

ただし，$\omega \in \Omega = \{+1, -1\}$ である．gBoost では，以下の関数型を持つ予測モデルを学習する．h の定義より，実関数 f は 0-1 変数 x_t に関して線形の関数となる．

$$f(\boldsymbol{x}) = \sum_{(t,\omega)\in\mathcal{T}\times\Omega} a_{t,\omega} h(\boldsymbol{x}; t, \omega) \tag{1}$$

ただし，$a_{t,\omega} \geq 0$ である．2 値分類（+1 or -1）の場合，$f(\boldsymbol{x}) \geq 0$ の時に+1，$f(\boldsymbol{x}) \leq 0$ の時に-1 と予測する．

gBoost では，式 (1) のモデルの係数 $\alpha_{t,\omega}$ を LPBoost[12] という線形計画問題で定式化される Boosting で学習する．グラフ G_i の 0-1 ベクトル表現を $\boldsymbol{x}_i$ とすると，LPBoost の線形計画問題の双対問題は次で与えられ，主問題の変数である式 (1) の係数はラグランジュ乗数 $\boldsymbol{\lambda} = \{\lambda_i\}_{i=1}^N$ から得られる．

$$\min_{\boldsymbol{\lambda},\gamma} \gamma$$
$$\text{s.t.} \begin{cases} \sum_{i=1}^N \lambda_i y_i h(\boldsymbol{x}_i; t, \omega) \leq \gamma, \forall (t, \omega) \in \mathcal{T} \times \Omega \\ \sum_{i=1}^N \lambda_i = 1,\ 0 \leq \lambda_i \leq D,\ i = 1, \ldots, N. \end{cases}$$

この線形計画を列生成法と呼ばれる逐次的な解法で解く手続きは Boosting と見なせ，LPBoost と呼ばれる各反復で全ての係数を更新する totally corrective な Boosting である．列生成法で追加する列の探索は，グラフ分類の場合の双対問題では，次の gain $g(t)$ が最大となる部分グラフ $t \in \mathcal{T}$ の探索に対応する．

$$g(t) = \sum_{i=1}^N \lambda_i y_i h(\boldsymbol{x}_i; t, \omega) \tag{2}$$

gBoost ではこの特徴探索に gSpan[13] と呼ばれる頻出部分グラフ列挙アルゴリズムの木型の探索空間 (探索木) を用いる．この探索木では，親ノードのグラフは子ノードのグラフと部分グラフ同型の関係になるため，子ノードをたどる際に，部分グラフを持つグラフが減ることはあっても，増えることはない．この性質を用いることで，現在探索中の部分グラフ t に対して，その子孫ノードの部分グラフの gain の最大値は次の bound $b(t)$ で抑えられる．

$$b(t) = \max\{\, 2 \sum_{\{i|y_i=+1, t\subseteq G_i\}} \lambda_i - \sum_{i=1}^N y_i \lambda_i,\ 2 \sum_{\{i|y_i=-1, t\subseteq G_i\}} \lambda_i + \sum_{i=1}^N y_i \lambda_i \} \tag{3}$$

従って今まで調べた中で gain 最大の暫定最適解に対して，探索中の部分グラフ t の上界 $b(t)$ がその暫定最適解の gain を超えない場合は，t の子孫ノードの探索は必要はなく，枝刈りできる．この分枝限定法の仕組みにより，gBoost は，巨大な候補集合 $\mathcal{T}$ から効率よく必要となる特徴を学習することができる．

3 部分グラフ共起を用いた学習

ここでは，従来の gBoost のモデルの式 (1) を，部分グラフの共起を考慮した次のモデルで置き換えた場合の学習を考える．

$$f(\boldsymbol{x}) = \sum_{(t,\omega)\in\mathcal{T}\times\Omega} \alpha_{t,\omega} h(\boldsymbol{x}; t, \omega) + \sum_{t,t'\in\mathcal{T}, t\neq t', \omega\in\Omega} \alpha_{t,t',\omega} h(\boldsymbol{x}; t, t', \omega) \tag{4}$$

ただし，$h(\boldsymbol{x}; t, t', \omega) = \omega(2x_t x_{t'} - 1)$ とする．

1 つ目の項は gBoost のモデル式であり各々の部分グラフの有無が y に与える独立効果を表す 1 次項，2 つ目の項が 2 つの部分グラフ t, t の共起の相乗効果を表す 2 次交互作用項である．ここでは，3 次以上の交互作用は考えず，2 つの部分グラフの交互作用のみ考慮する．

交互作用項は 1 つの部分グラフ t の有無 x_t が 2 つの部分グラフ t, t' の共起の有無 $x_t x_{t'}$ になっただけであり，式 (2) の gain $g(t)$ および式 (3) の bound $b(t)$ も同様に拡張でき，これを $g(t, t')$ および $b(t, t')$ とする．

候補部分グラフ $t \in \mathcal{T}$ の全ての共起を調べるには単純に $O(|\mathcal{T}|^2)$ の追加計算コストが必要となり，事前に部分グラフとその共起の有無を全列挙した後に学習する方法では実データを処理できない．よって，本稿では，既存手法である gBoost 法の探索・最適化の枠組みに基づき，陽に全列挙せず，必要な部分グラフとその共起のみを効率的に探索しながら式 (4) のモデルを学習するグラフ分類手法として，厳密法 (3.1 節) と近似法 (3.1 節) の 2 つの手法を提案する．

3.1 提案手法 1: 厳密法

まず，gBoost をモデル (4) を学習するよう拡張した厳密法を示す．gBoost はモデル (1) の学習自体は LPBoost の反復法に基づいているが，各反復において gain $g(t)$ が最大となる部分グラフの探索を行う．部分グラフの共起を考慮するための拡張ではこの部分にのみ変更が必要となるため，以降ではこの部分のみを扱う．この部分の手続き，ある反復で得られている $\{\lambda_i\}_{i=1}^N$ に基づき $g(t)$ 最大の部分グラフ t^*，その gain $g^* = g(t^*)$，および，対応する弱学習機パラメタ ω^*, を返す手続きを getMaxGain() とする．この探索の大枠は gSpan アルゴリズム [13] とその中で用いられる DFS コードと呼ばれるグラフ表現に基づいている．

対象の式 (4) のモデルで gBoost と同様逐次的に項を追加していく上で，まず最初に getMaxGain() を実行し，1 次項で得られる最大 gain g^* を得た後，$g^* < g(t, s)$ となる共起 (t, s) を探索すれば十分である．従って，gBoost の getMaxGain() をアルゴリズム 1 に示す getMaxGainPair() で置き換えることで，モデル (4) の学習が厳密に可能である．(t, s) と (s, t) という同一の共起の冗長な探索を防ぐために，gSpan[13] の DFS コード表現の辞書順 (以下，DFS 辞書順) を用いる．

3.2 提案手法 2: Top-k 法

提案手法 1 では厳密に全部分グラフとその共起すべてを考慮したモデルが学習できる．しかし，実データでは全ての共起のうち，分類に寄与するものはほんのわずかであるため，厳密に全てを見ずに寄与する可能性がありそうなもののみを効率的に探索したい．そこで，共起を探索する際，gain $g(t)$ および bound $b(t)$ で定義される次の指標 $j_\alpha(t)$

$$j_\alpha(t) = \alpha|g(t)| + (1 - \alpha)b(t)$$

に関して上位 k 個の部分グラフ集合 $\mathcal{T}_k$ を求め，共起 $(t, s) \in \mathcal{T}_k \times \mathcal{T}$ のみを探索する近似法を提案する．このようにすることで，探索する特徴数は $O(k|\mathcal{T}|)$ となり，パラメータ k で探索にかかるコストと厳密探索の近似精度とのトレードオフを制御できる．gain は直接的にその部分グラフの良さを表し，bound はその部分グラフの子ノード，あるいはその部分グラフとの共起により良いパターンが見つかる可能性を表す．そのため，gain と bound のトレードオフとなるこの指標を用いる．$g(t)$ および $b(t)$ は gBoost で各 t に対して元々計算するため，追加で $j_\alpha(t)$ を保持し，最後にその上位 k の集合 $\mathcal{T}_k$ を返すように getMaxGain() を修正した手続きを getMaxGainTopK() とすると，アルゴリズム 2 に示す擬似コードとなる．この際，CoProject(t, s) では DFS 辞書順での枝刈りが厳密法同様に行われる．上位集合 $t \in \mathcal{T}_k$ への制限の場合，この枝刈りは本来不要な探索以上に探索空間を刈ってしまうが，同一の共起を複数調べないようにする方が効果が大きいため許容する．

Algorithm 1 部分グラフとその共起の探索 (厳密法)

```
1: procedure getMaxGainPair
2:     グローバル変数：g*, ω*, t*, p*
3:     (t*, g*, ω*) ← getMaxGain()
4:     for all t ∈ 1 辺のみの部分グラフ do          ▷ 共起の探索
5:         Project(t)
6:     if g* が for 文前後で更新あり then
7:         return (p*, g*, ω*)                    ▷ 交互作用項を返す
8:     else
9:         return (t*, g*, ω*)                    ▷ 通常の 1 次項を返す
10: function Project(t)
11:     if t が最小 DFS コードでない then
12:         return
13:     t に対する gain g(t), bound b(t) を計算
14:     if b(t) < g* then
15:         return
16:     for all s ∈ 1 辺のみの部分グラフ do
17:         CoProject(t, s)
18:     for all t' ∈ t の最右拡張グラフ do
19:         Project(t')
20: function CoProject(t, s)
21:     if s が最小 DFS コードでない then
22:         return
23:     if t < s then                              ▷ DFS 辞書順
24:         return
25:     s に対する gain g(s),bound b(s) を計算
26:     if b(s) < g* then
27:         return
28:     t, s に対する gain g(t, s),bound b(t, s) を計算
29:     if b(t, s) < g* then
30:         return
31:     if ω ∈ Ω のいずれかに対し g(t, s) > g* then
32:         g* ← g(t, s), p* ← (t, s), ω* ← ω
33:     for all s' ∈ s の最右拡張グラフ do
34:         CoProject(t, s')
```

3.3 共起探索の効率化

2 つの部分グラフ t, s の共起 (t, s) の有無の探索について，部分グラフの有無の 0-1 変数の性質を利用して，

Algorithm 2 部分グラフとその共起の探索 (Top-k)

```
1: procedure getMaxGainPairTopK
2:     グローバル変数 : g*, ω*, t*, p*
3:     (t*, g*, ω*, T_k) ← getMaxGainTopK()
4:     for all t ∈ T_k do
5:         for all s ∈ 1 辺のみの部分グラフ do
6:             CoProject(t, s)
7:     if g* が for 文前後で更新あり then
8:         return (p*, g*, ω*)                ▷ 交互作用項を返す
9:     else
10:        return (t*, g*, ω*)                ▷ 通常の 1 次項を返す
```

さらに不要な探索を枝刈りできる．ここでは 2 つの方法を示す．

まず 1 つ目として，共起 (t, s) において t と s が部分グラフ同型の場合，$g(t, s)$ や $b(t, s)$ は片方のグラフの gain および bound と等しくなる．従って，$i = 1, \ldots, N$ の sum の形で表される $g(t, s)$ や $b(t, s)$ の新たな計算は不要であり，省略することができる．ただし，部分グラフ同型の判定は NP 完全であるため陽に実行せず，gSpan アルゴリズムの探索木では親ノードと子ノードは部分グラフ同型になっている性質を利用して，簡単に判定できる場合にのみ適用する．

2 つ目として，gBoost では gain 最大の部分グラフを探索する際に同一の gain 値を持つものがあれば先に探索されたもの (DFS 辞書順で小さいもの) を採用するタイブレークを行う．また，gSpan アルゴリズムの探索木における親ノードと子ノードは部分グラフ同型であるため，支持度 ($\{G_i\}_{i=1}^N$ の中でその部分グラフを持つものの数) が同じであれば，gain 値は同じになる．よって，部分グラフを探索している時，親ノードと同じ支持度を持つグラフに対して共起を考える必要がない．

3.4 モデル構築の効率化:二段階法

本節では，提案手法 1 および 2 にともに適用可能な，モデル構築の効率化について述べる．提案手法 1・2 は gBoost の拡張であり，前述したように LPBoost の反復法に基づいている．各反復では，モデル式 (4) の項を一つ追加し各項の係数 $\alpha_{t,\omega}, \alpha_{t,t',\omega}$ を更新する．LPBoost の枠組みに従えば，1 次の項と 2 次の項のどちらを追加するかは各反復で自動的に決まる．これに対して，予め 1 次モデルを収束するまで学習してから，それを改善できる交互作用項があれば共起を探索して追加する二段階学習が考えられる．グラフの場合，まず 1 次モデルを収束するまで学習しきってから，これを warm start として用いて，共起探索を伴う交互作用項の学習を行う方が，計算コストの高い共起探索時の枝刈りの効果がより大きいと期待できる．

ここで，gBoost 構成後に共起を探索・追加する二段階法を"手法 1"と呼び，最初から共起を探索する方法を"手法 2"と呼ぶ．これらは，厳密法にも Top-k 法にも適用できるため，4 つの提案手法のバリエーション（"厳密法 1"，"厳密法 2"，"Top-k 法 1"，"Top-k 法 2"）を 4 節で実データを用いた実験によって解析する．

データ名	ALL	ATOM	BOND	正例	負例
CPDB	684	25.2	25.6	341	343
Mutag	188	26.3	28.1	125	63
CAS	4337	30.3	31.3	2401	1936
AIDS(CAvsCM)	1503	59.0	61.6	422	1081

表 1: 使用したデータセット

ν	0.01,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
σ	6,8,10
ϵ	0.01
k	10,30,100,500
α	0.2,0.4,0.6,0.8

表 2: 使用したパラメータ

4 実験

本節では，グラフ分類に対して共起を考慮することによる有用性を検証するために，実データを用いた実験を行う．共起を考慮することで，モデル構築のための時間が大幅に増加する．そのため，各ハイパーパラメータに対するモデル構築時間に制限 (30 分) を設け，制限時間内に構築されたモデルに対してのみ評価を行い，正答率等により有用性が見られるかどうか検証を行う．また，共起を探索することによる探索数の増加，それに対して Top-k 手法による探索数の変化や Top-k のパラメータ α を変化させたときの影響を検証する実験を行う．

4.1 使用したデータセット

本稿では，gBoost の元論文 [7] で用いられた化合物データセットのうち 4 つを使用した．ただし，元論文 [7] と異なり，化合物のグラフ表現は，水素を明示的に付与し，Sybyl Mol2 形式で頂点ラベル、辺ラベルを付与した分子グラフである．表 1 に各データセットのグラフ数 (ALL)，平均ノード (ATOM)・エッジ数 (BOND)，正例と負例の数を示す．

4.2 精度に関する実験

実験に用いたパラメータは表 2 に示す通りである．ν, ϵ は LPBoost[12] のパラメータである．σ は特徴として使用する部分グラフのサイズ (エッジ数) を制限するパラメータである．既存手法 gBoost 法と提案手法 1:厳密法については，k と α は存在しない．実験手法は，表に示したパラメータの組合せをすべて使用し，最良の結果を評価する．評価の方法として，最良となる正答率 (ACC) とその時の AUC を 10-分割交差検証により評価する．

既存手法に対して，提案手法が特徴として用いる探索空間は既存手法を完全に包含しているため，使用できる時間に制限を設けなければ，原理上精度が下がることは無いと考えられる．本実験では，公平のために

1 つのモデルを構築するために要した時間が 30 分を超えたハイパーパラメータに関しては評価無しとした．

各データセットに対する実験結果を表 3 に示す．既存手法に比べ，厳密法では，Mutag では精度が向上したものの，他のデータでは精度改善が見られなかった．これは，より複雑な構造を特徴として用いることによる過学習 (主に CPDB に対して) やモデル構築時間制限 (CAS，AIDS) が原因だと考えられる．一方，Top-k 法の結果では，すべてのデータで精度の向上が見られた．全部分グラフの共起を入れないことで過学習を抑制できている可能性が考えられる．また，CAS や AIDS といったデータセットでは，k の数が小さい時，制限時間内で計算可能なパラメータで既存手法より良い結果が得られている．

厳密法，Top-k 法ともに二段階学習である手法 1 に比べ，無制約な手法 2 は精度があまりよくない．これは節 4.3 でも言及する探索数の増加によって評価なしとなるものが多い事も考えられるが，手法 1 では，gBoost 構築後に共起を探索するため，通常の部分グラフの表現で十分である (共起で表現する必要がない) 部分に関して構築した後に共起を探索するため，過学習を抑制する効果があるのではないかと考えられる．

4.3 探索特徴とその計算時間に関する実験

4.2 節では，各データセットに対して各手法を用いて分類した際の精度について比較実験を行った．本節では，部分グラフの共起を考慮にいれることで探索がどれほど増加するのかを比較するための実験を行う．この実験で用いるデータセットは 4.2 節の実験で用いた Mutag，CPDB を用いる．この比較において，$\nu = 0.4$, $\sigma = 10$ を固定のパラメータとして用いた．

実際にかかる計算時間のほとんどは，特徴を探索する時間と線形計画問題を解く時間である．線形計画問題を解く時間は追加した特徴数が増加すればするほど増加する．本研究では，特徴となる構造を探索した個数に着目して比較実験を行う．図 2 に実験結果を示す．

それぞれの結果において，"1"は gBoost 構成後に共起を探索する方法であり，"2"は最初から共起を探索する方法である．"strict"は厳密法，"top"は Top-k 法であり，それぞれ k の数は 50，400，1600，6400，all を用いた．ここで all はすべての特徴と共起を見ることを指すため，厳密法と特徴の探索空間は同じである．

Top-k の k を変えることで，どの程度共起の構造を探索するかを選ぶことができる．gBoost 構成後に共起を探索する手法 1 では，手法 2 に比べ厳密法，Top-k 法ともに総探索数は抑えられている．しかし，収束までのイテレーション数が増加することで線形計画問題を解く回数が増えるため，トレードオフが必要である．また，Mutag に比べ，CPDB では共起探索数がとても多いことが見て取れる．共起を探索する候補となる通常の部分グラフが多いほど，共起の探索が増加するためであり，データセットによって k の値をうまく設定することで探索数を調節する必要があるだろう．

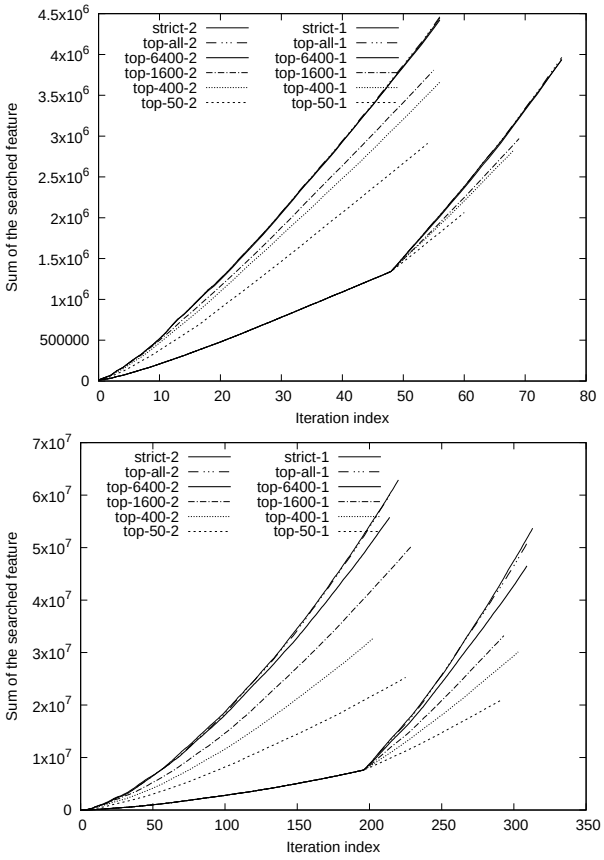

図 2: Mutag(上)CPDB(下) に対する特徴量探索数に対する比較実験結果

4.4 Top-k 法のパラメータ α に関する実験

Top-k を選ぶ指標である α が，Top-k に対してどのように寄与するのかを実験する．用いたデータは Mutag, CPDB で，パラメータは $\nu = 0.4$，$\sigma = 10$ を用いた．α を 0 から 1 まで 0.1 ずつ変化させたときの gBoost の目的関数値と毎イテレーションの共起探索数の平均を k が 50，100，all を用いて比較する．実験結果を表 4 に示す．

まず k が all の時，すべての特徴を探索するため目的関数値は同じである．良い特徴が早く発見された時，探索の枝刈りが良く効き探索数が少なくなることがあり，$\alpha = 1.0$ の時どちらのデータも探索数が最小である．ここで，k が 50，100 の時を見てみると，α が大きくなるに連れて目的関数の値が大きくなり，平均共起探索数が減少する傾向がある．しかし，k が 50 と小さい時，$\alpha = 1.0$ は目的関数の値が下がるときがある．これは，gain の値の上位 bound が低いものが存在したからだと考えられる．この実験の結果，共起を探索する Top-k の選択指標は α が 1.0 に近づく，つまり，部分グラフ特徴の gain をより重視したほうがよい探索が可能であるという傾向が見られた．

Data		gBoost	厳密法 1	厳密法 2	Top-k 法 1	Top-k 法 2
CPDB	ACC	77.5 (± 4.9)	77.3 (± 5.2)	76.5 (± 6.7)	**79.6 (± 8.4)**	78.5 (± 4.5)
	AUC	77.8 (± 6.3)	78.4 (± 4.5)	75.7 (± 8.0)	79.3 (± 8.1)	79.2 (± 5.7)
		ν 0.2, σ 8	ν 0.2, σ 10	ν 0.4, σ 10	ν 0.1, σ 6,α 0.6,k 30	ν 0.4, σ 8,α 0.4,k 30
Mutag	ACC	84.2 (± 9.7)	86.2 (± 10.5)	84.6 (± 10.6)	**86.3 (± 10.3)**	85.7 (± 9.1)
	AUC	85.9 (± 9.1)	88.7 (± 9.1)	86.3 (± 8.9)	85.5 (± 10.3)	85.5 (± 9.5)
		ν 0.2,σ 10	ν 0.2,σ 6	ν 0.2,σ 6	ν 0.3,σ 10,α 0.4,k 500	ν 0.3,σ 10,α 0.8,k 10
CAS	ACC	81.7 (± 2.7)	70.8 (± 3.9)	70.8 (± 4.5)	**82.5 (± 2.5)**	80.8 (± 2.6)
	AUC	83.0 (± 2.4)	72.9 (± 4.4)	73.1 (± 4.8)	83.4 (± 2.3)	82.1 (± 2.8)
		ν 0.4,σ 6	ν 0.8,σ 6	ν 0.8,σ 8	ν 0.4,σ 6,α 0.6,k 10	ν 0.5,σ 6,α 0.2,k 10
AIDS	ACC	82.7 (± 2.3)	76.6 (± 3.1)	76.8 (± 3.0)	83.5 (± 2.1)	**83.7 (± 2.5)**
	AUC	67.7 (± 5.7)	95.8 (± 2.2)	95.8 (± 2.2)	68.5 (± 4.6)	68.0 (± 5.2)
		ν 0.4,σ 6	ν 0.9,σ 10	ν 0.8,σ 6	ν 0.4,σ 6,α 0.6,k 30	ν 0.4,σ 6,α 0.6,k 30

表 3: 精度実験 (制限時間 (30 分) 内に構築されたモデルによる比較)

Data	k		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Mutag	50	obj	0.1580	0.1580	0.1584	0.1589	0.1590	0.1590	0.1598	0.1606	0.1602	**0.1627**	0.1610
	50	ave	24930	25044	25467	26303	26338	26341	22409	20006	4222	3315	514
	100	obj	0.1592	0.1592	0.1592	0.1592	0.1598	0.1598	0.1601	0.1614	0.1618	0.1627	**0.1635**
	100	ave	30547	30585	30648	30650	30277	30550	28411	26492	11023	3708	1510
	all	obj	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648	0.1648
	all	ave	52923	52923	52918	52898	52833	52824	52803	52686	52577	52448	52448
CPDB	50	obj	0.0266	0.0266	0.0267	0.0265	0.0265	0.0261	0.0263	0.0261	0.0269	**0.0274**	0.0265
	50	ave	70422	70381	70668	70455	66804	62254	62992	59698	57596	54033	3968
	100	obj	0.0268	0.0272	0.0269	0.0271	0.0270	0.0268	0.0269	0.0273	0.0273	**0.0278**	0.0263
	100	ave	79524	83987	71307	79888	80336	78979	79673	76317	76230	67827	5125
	all	obj	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285	0.0285
	all	ave	243310	243276	243237	243190	243128	243036	242907	242711	242399	250276	239396

表 4: α を変化させたときの目的関数値と平均共起探索数

5 結論と今後の展望

本稿では，グラフ分類問題に対して入力グラフに含まれる部分グラフとその共起を用いてモデル構築をする手法を提案した．厳密に共起を探索する手法では，特徴探索数が膨大に増加し，精度向上しないデータセットもあった．そこで提案した Top-k 法では，特徴探索数を抑え，精度向上の結果となった．今後の展望として，交互作用項に対する正則化を変えて過学習を抑えるモデルを構築したい．

謝辞

本研究は JSPS 科研費 17H01783，17K19953 および JST さきがけの助成を受けたものです．

参考文献

[1] T. Kudo, E. Maeda, and Y. Matsumoto. An application of boosting to graph classification. In *NIPS*, pp. 729–736, 2004.

[2] R. B. Altman, T. A. Jung, T. E. Klein, A. K. Dunker, and L. Hunter, editors. *Biocomputing 2005, Proceedings of the Pacific Symposium, Hawaii, USA, 4-8 January 2005*. World Scientific, 2005.

[3] I. Takigawa and H. Mamitsuka. Graph mining: procedure, application to drug discovery and recent advances. *Drug Discovery Today*, Vol. 18, No. 1–2, pp. 50–57, 2013.

[4] 瀧川一学. 多数のグラフからの統計的機械学習. 深化する機械学習技術の進展とその応用特集号. システム/制御/情報, Vol. 60, No. 3, pp. 107–112, 2016.

[5] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pp. 321–328, 2003.

[6] D. Rogers and M. Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, Vol. 50, No. 5, pp. 742–754, 2010.

[7] H. Saigo, S. Nowozin, T. Kadowaki, T. Kudo, and K. Tsuda. gboost: a mathematical programming approach to graph classification and regression. *Machine Learning*, Vol. 75, No. 1, pp. 69–89, 2009.

[8] S. Rendle. Factorization machines. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pp. 995–1000, 2010.

[9] S. Suzumura, K. Nakagawa, Y. Umezu, K. Tsuda, and I. Takeuchi. Selective inference for sparse high-order interaction models. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 3338–3347, 2017.

[10] H. J. Cordell. Detecting gene–gene interactions that underlie human diseases. *Nature Reviews Genetics*, Vol. 10, No. 6, pp. 392–404, 2009.

[11] E.-W. Lameijer, J. N. Kok, T. Bäck, and A. P. IJzerman. Mining a chemical database for fragment co-occurrence: discovery of “chemical cliches”. *Journal of chemical information and modeling*, Vol. 46, No. 2, pp. 553–562, 2006.

[12] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, Vol. 46, No. 1-3, pp. 225–254, 2002.

[13] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*, pp. 721–724, 2002.