4F1-OS-5-1

ブースティングに基づく制約付きK-meansアルゴリズム

Boosted Constrained K-means Algorithm

岡部正幸*1　山田誠二*2
Masayuki OKABE　Seiji YAMADA

*1豊橋技術科学大学
Toyohashi University of Technology

*2国立情報学研究所／総合研究大学院大学／東京工業大学
National Institute of Informatics

COP-KMEANS is a light weight constrained clustering algorithm in terms of computational cost, which is desirable for interactive data mining systems that potentially require responce performance. However, COP-KMEANS has several problems caused by the order of the data assignment in the clustering procedure. This paper proposes a method to avoid such problems and improve the performance of COP-KMEANS by using boosting technique. Based on the boosting framework, we control the data assignment order using weights that are given to must/cannot-link constraints in the process of boosting. Experimental results show that our proposed method effectively improves the performance of COP-KMEANS.

1. はじめに

クラスタリングはデータマイニングにおける基礎的なデータ分析手法として広く利用されているが，近年，データペアに対し制約情報を与えることでクラスタリング性能の向上を目指す制約付きクラスタリングに関する研究が進んでいる．

制約付きクラスタリングは，制約情報を予め用意しておくことで従来のクラスタリングのようにバッチ処理的に計算を行うこともできるが，タスクによっては制約情報がシステム利用者から逐次的に与えられるといった状況も想定される．このようなタスクでは，クラスタリング処理およびその結果の分析に基づく制約情報の付与というサイクルが生じ，システム利用者のフィードバックを反映させたデータ分析が可能となることが期待される．例えば，ある特定の対象の抽出を目的とした画像のセグメンテーション作業では，抽出に失敗したデータをシステム利用者が確認して初めて制約情報を与えるといった作業スタイルも考えられ，この場合，システムは利用者の制約情報の付与に応じてクラスタリングを逐次的に実行する必要がある．

このように，制約付きクラスタリングは対話的にデータ分析を行うシステムの要素技術として役立つ可能性を持っているが，実際の利用を考えた場合，その応答性能が重要となる．制約付きクラスタリングアルゴリズムは，これまでにいくつか提案されているが，その多くは距離学習 [Davis 07] やカーネル行列学習 [Hoi 07] をベースにしたものが多く，対話的なシステムへの適用を考えた場合，その多大な計算量に起因する応答性能の低さが問題となることが予想される．

そこで本研究では，制約付きクラスタリングアルゴリズムの中で計算量が比較的少ない COP-KMEANS[Wagstaff 01] に着目する．COP-KMEANS の計算量は，k-means と同じくクラスタリング対象となるデータ数の線形オーダーであるため，応答性能を考えた場合に有利である反面，距離学習などをベースとしたものに比べクラスタリング性能の面では不利であり，また，クラスタ中心へのデータの割当順序によっては，クラスタ集合が生成されず停止してしまうというアルゴリズム的な欠点が存在する．

連絡先: 岡部正幸，豊橋技術科学大学情報メディア基盤センター
〒 441-8580 豊橋市天伯町雲雀ヶ丘 1-1
okabe@imc.tut.ac.jp

本研究では，COP-KMEANS が持つこれらの問題点を補う方法としてブースティングの適用を試みる．ブースティングでは，訓練データの重みを適応的に変化させながら生成した複数の弱学習器を統合することで，単独の弱学習器よりも精度の高い判別を行うことができるとされている．我々は，COP-KMEANS をブースティングにおける弱学習器に見立て，弱学習器が生成した複数のクラスタリング結果を統合することで最終的に精度の高いクラスタ集合を得ることを目指す．また，同時にブースティングによって与えられた制約に対する重みを，COP-KMEANS におけるクラスタ中心へのデータ割当順序の決定に用いるなどの修正を COP-KMEANS に対して行う．

2. COP-KMEANS

COP-KMEANS は分割型クラスタリングの代表的アルゴリズムである k-means をベースとしており，その基本アルゴリズムは，1) 初期クラスタ中心の選択，2) 各データのクラスタ中心への割当，3) クラスタ中心の更新という主に 3 つの処理からなる．この内の 2) の処理について，通常の k-means では，データを割り当てるべきクラスタ中心は，データとの距離が最も近いものが選択されるが，COP-KMEANS の場合は，その他に，“与えられたすべての制約に違反しないもの” という条件も合わせて考慮される．つまり，データとの距離が最も近いクラスタ中心であっても，そのクラスタ中心に既に割り当てられているデータとの間に制約違反が確認された場合は，別のクラスタ中心に割り当てられる．また，候補となるクラスタ中心すべてに制約違反が確認された場合は，割り当て可能なクラスタ中心がないことからアルゴリズムが停止する．

COP-KMEANS では，このような状況が発生した場合への対処は特に規定されておらず，クラスタリングが成功に終わるか失敗に終わるかは，クラスタ中心へのデータの割り当て順序に大きく依存するという特徴をもつ．本研究では，COP-KMEANS が持つこのような問題に対処するため，以下のアプローチを取る．

1. 制約をすべて充足させることを諦める．データ数に対する制約数の割合が増加するにつれ，制約を完全に充足させることが困難になるため，部分的な充足しか行えない場合でもクラスタリング処理を完了させる．

2. ブースティングを利用することにより，制約集合が部分的にしか充足されないクラスタリングアルゴリズムを用いた場合でも，精度の高いクラスタリングが行えることを目指す．

次章では，このアプローチに基づくブースティングベースの制約付きクラスタリングアルゴリズムとブースティングにおける弱学習器として利用するために修正した COP-KMEANS アルゴリズムの詳細について説明する．

3. ブースティングに基づく制約付きクラスタリング

前章における議論に基づき，本章ではブースティングベースの制約付きクラスタリングアルゴリズムと修正版 COP-KMEANS を提案する．提案方法の特徴は主に以下の 2 つである．

1. ブースティングを導入することにより，制約集合が部分的にしか充足されていない複数のクラスタリング結果を統合して，精度の高いクラスタリング結果を生成する．
2. ブースティングにおける弱学習器として修正版 COP-KMEANS を利用している．修正版 COP-KMEANS では，ブースティング過程において制約に与えられる重みを利用して，クラスタ中心へのデータ割当順序を決定する．また，制約の充足に失敗してもアルゴリズムは停止しない．

ブースティングは，ランダム推定に対しある程度の優位度をもつ弱学習器が生成する弱仮説を統合することにより，精度の高い判定を行うアンサンブル学習方法の一つである．本研究では，ブースティングの適用対象となる学習問題を制約付きクラスタリングとし，制約集合を訓練データ，修正版 COP-KMEANS を弱学習器と対応づける．これにより，各制約には充足回数に基づく重みが計算されるが，修正版 COP-KMEANS ではこの重みを利用してクラスタ中心へのデータ割当順序を決定する．

以下では，提案方法の本体となるブースティングベースの制約付きカーネル K-means（Boosted Constrained Kernel K-means）とブースティングにおける弱学習器として利用するために修正した COP-KMEANS アルゴリズム（Modified COP-KMEANS）について説明する．

3.1 Boosted Constrained Kernel K-means

Algorithm 1 に Boosted Constrained Kernel K-means アルゴリズムの概要を示す．このアルゴリズムは，代表的ブースティングアルゴリズムである AdaBoost[Freund 97] をベースとしている．弱学習器として，後述する修正版 COP-KMEANS を利用することを想定しており，訓練データである各制約（must-link: 同じクラスタに属さなければならないデータペア，cannot-link: 異なるクラスタに属さなければならないデータペア）の重み計算と，修正版 COP-KMEANS が生成したクラスタリング結果を統合してカーネル行列を生成する．このカーネル行列を用いたカーネル k-means を実行することにより，最終的なクラスタリング結果が出力される．

このアルゴリズムにおいて．修正版 COP-KMEANS が行うのは制約付きクラスタリングであるが，ブースティング的な解釈では，任意のデータペアが must/cannot-link のどちらであるかを推定していると捉えることもできる．ブースティング自体はクラスタリングを直接的に行うわけではないが，質の良いカーネル行列を生成することで，クラスタリング精度を向上させることに間接的に寄与しているといえる．

Algorithm 1 Boosted Constrained Kernel K-means

1: 入力：データ集合 $X = \{x_1, ..., x_{|X|}\}$，制約集合 $S = \{(i_1, j_1, y_1), ..., (i_{|S|}, j_{|S|}, y_{|S|})\}$
2: 出力：クラスタ集合 C
3: $w_i^0 \leftarrow \frac{1}{|S|} \quad (i = 1 \sim |S|)$
4: **for** $t = 1$ to T **do**
5: Modified COP-KMEANS アルゴリズムを実行し，その結果に基づき，カーネル行列 K^t を生成する．

$$k^t(i,j) = \begin{cases} 1 & (x_i, x_j) \text{ が同じクラスタに属する場合} \\ -1 & (x_i, x_j) \text{ が異なるクラスタに属する場合} \end{cases}$$

6: 充足に失敗した制約の重みの割合 ϵ_t を計算する．

$$\epsilon_t = \frac{\frac{1}{2}\sum_{n=1}^{|S|} w_n^t I(k^t(i_n, j_n), y_n)}{\sum_{n=1}^{N} w_n^t} \tag{1}$$

$I(a,b)$ は，$a \neq b$ のとき 1，$a = b$ のとき 0 を返す関数．(i_n, j_n) は，n 番目の制約におけるデータペアのインデックス，y_n はその制約の種類を示す値（must-link の場合は 1，cannot-link の場合は-1 の値を取る）

7: ϵ_t を用いて，K^t の重み α_t を計算する．

$$\alpha_t = \ln\left\{\frac{1-\epsilon_t}{\epsilon_t}\right\} \tag{2}$$

8: 各制約の重み係数 w_n^{t+1} を更新する．

$$w_n^{t+1} = w_n^t \exp\left\{\alpha_t I(k^t(i_n, j_n), y_n)\right\} \tag{3}$$

9: **end for**
10: 以下の式を用いて最終的なカーネル行列 K を求める．

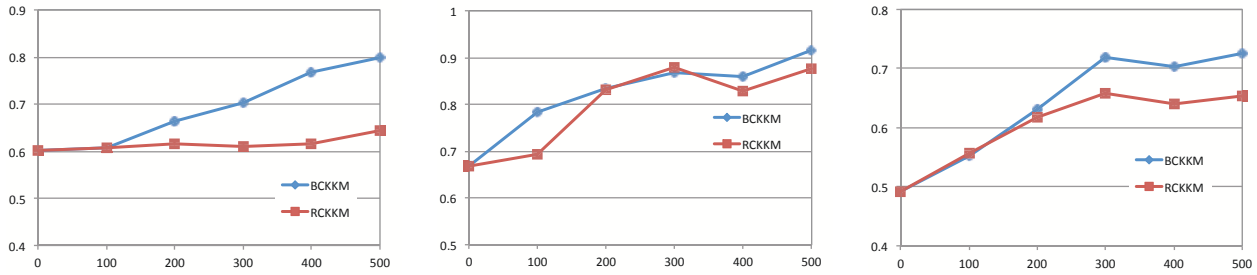
$$K = \sum_{t=1}^{T} \alpha_t K^t \tag{4}$$

11: K を用いてカーネル K-means を実行し，最終的なクラスタリング集合 C を得る．

3.2 Modified COP-KMEANS

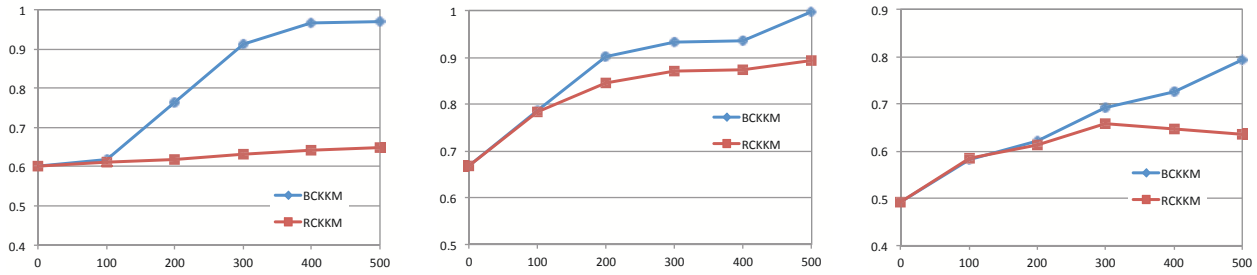
Algorithm 2（本稿末尾に掲載）に，ブースティングにおける弱学習器として利用する修正版 COP-KMEANS アルゴリズムの概要を示す．このアルゴリズムは，ブースティングによって与えられた各制約の重みを利用してクラスタ中心へのデータ割当順序を決定している．一般的に割当順序が早ければ制約が充足される確率も上がるが，重みの大きな制約は充足に失敗している回数の多いものであり，そのような制約の割当を優先的に行うことで充足の成功確率を高めている．

また，割当順序が与えられるのは個々のデータではなく制約であるため，クラスタ中心へのデータ割当は制約対象となっている 2 つのデータについて同時に行う．このため，データペアがそれぞれ既に割当済みかどうか，また制約が must/cannot-link のどちらであるかによって適宜対応する処理を行う必要がある．このアルゴリズムにおいて制約の充足に失敗するのは，データペアがともに既に割当済みの場合であり，must-link であるにもかかわらず異なるクラスタ中心へ割り当てられてい



(a) Iris (b) Wine (c) Glass

図 1: ブースティングステップ数 $T = 10$ の場合の結果（グラフの縦軸は NMI，横軸は制約数を表す）



(a) Iris (b) Wine (c) Glass

図 2: ブースティングステップ数 $T = 100$ の場合の結果（グラフの縦軸は NMI，横軸は制約数を表す）

る，または cannot-link にもかかわらず同じクラスタ中心へ割り当てられている場合である．このように，各データのクラスタ中心への割当は必ず行われ，オリジナルの COP-KMEANS のようにアルゴリズムが停止することはない．

クラスタ割当の優先順序を変えただけでは，制約充足数の増加はそれほど期待できないが，充足される制約の部分集合が変化するため，異なるクラスタリング結果を生成することができ，アンサンブル学習としての効果が期待できる．

4. 実験

本章では，提案手法の効果を調べるために行った実験結果について述べる．

4.1 実験設定

実験データには UCI レポジトリ*1から，3 つのデータセット Iris (150,3)，Wine (178,3)，Glass(214,6) を使用した．カッコ内の数値は順にデータ数，クラス数を表す．各データセットは，前処理として特徴ベクトルのスケーリング（属性毎にその属性の最大値で割る操作）と正規化 (単位ベクトル化) を行った．

提案手法（以下，BCKKM と呼ぶ）の実験結果のほか，ブースティングによるデータ割当順序決定の効果を調べるため，比較手法としてデータの割当順序をランダムに決定する方法（以下，RCKKM と呼ぶ）との比較を行った．RCKKM は，制約の重み（w_n^t）をランダムに決定し，学習結果の重み（α）をすべて同じ値とすること以外は提案手法と同じ処理を行う．両手法とも，データとクラスタ中心との距離尺度としてユークリッド距離を用いた．

制約の生成については，ランダムにデータペアを選択し，各データセットのクラスラベルに基づいて must/cannot-link のラベルを与えた．クラスタリング結果の評価には，正規化相互情報量 (Normalized Mutual Information, NMI) [Tang 07] を用いた．

4.2 実験結果

図 1 は，ブースティングステップ数を 10 とした場合の 2 つの手法の結果を示したものである．縦軸は NMI の値，横軸は制約数を表している．NMI の値は各制約数につき用意した異なる 10 セットの制約集合に対する平均値である．

提案手法（BCKKM）は，3 つのデータセット全てにおいて比較手法（RCKKM）の性能を上回る結果となった．Iris データセットについて，制約数が 100 のときに，2 つの手法に差はほとんど見られなかったが，制約数が 200 以降になると，2 つの手法の差は制約数が増えるにつれて大きくなった．これは BCKKM の性能が向上する一方，RCKKM は制約の効果があまり見られなかったためである．Wine データセットについては，制約数の増加とともに BCKKM の性能が向上することが確認されたが，RCKKM との差はそれほど拡がらないという結果になった．Glass データセットについては，Iris と傾向が似ており，制約数 300 以降で 2 手法の差が顕著になるという結果になった．

次に，ブースティングステップ数を 100 とした場合の 2 つの手法の結果を図 2 に示す．いずれのデータセットにおいても，ステップ数 10 の場合と同様の傾向が見られるが，ステップ数を増やすことによって，制約数 200 以降における BCKKM と RCKKM との差がステップ数 10 の場合よりも拡がったことが分かる．特に Wine については，2 手法の差が顕著になった．また，Iris では制約数 200 以降において性能が急激に上昇する結果となり，Glass についても制約数 400 以降における性能の向上が確認された．

以上より，提案手法の特徴であるブースティングに基づく

*1 http://archive.ics.uci.edu/ml/

データ割当順序の決定の効果が確認されたが，いくつか注意すべき点がある．まず，充足に失敗する制約が少ない場合（今回の実験においては各データセットでの制約数 100 の場合が該当する），各制約の重みが変化しないため，データの割当順序はランダムに生成された順序と等しくなり，ブースティングの効果が発揮されない．また，ブースティングステップ数の増加とともに性能が向上する傾向が見られたが，その分計算量も増加するため，性能と計算コストのトレードオフを探る必要がある．

5. まとめ

本研究では，制約付きクラスタリングを対話的なデータマイニングシステムにおいて利用することを想定し，計算量が少なくかつ精度の高いアルゴリズムの開発を目的として，ブースティングをベースとし，弱学習器に修正版 COP-KMEANS を用いる方法を提案した．提案手法は，AdaBoost アルゴリズムがベースとなっており，各制約の重みを適応的に変化させることにより，弱学習器としての修正版 COP-KMEANS から得られる複数のクラスタリング結果をカーネル行列として統合し，カーネル K-means を実行することで最終的なクラスタリング結果を出力する．修正版 COP-KMEANS はブースティングによって各制約に与えられた重みを利用してデータ割当順序を変化させることにより，充足された制約の部分集合が異なる複数のクラスタリング結果を出力することができる．

実験結果では，提案手法の特徴であるブースティングによるデータ割当順序の決定の効果が確認された．また，その効果は制約数およびブースティングステップ数の増加と比例することが分かった．

今後，より多くのデータセットにおける実験を行い，提案手法の効果についてより詳細な分析を行っていく予定である．

参考文献

[Davis 07] Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S.: Information-Theoretic Metric Learning, in *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pp. 209–216 (2007)

[Freund 97] Freund, Y. and Schapire, R. E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *J. Comput. Syst. Sci.*, Vol. 55, No. 1, pp. 119–139 (1997)

[Hoi 07] Hoi, S. C. H., Jin, R., and Lyu, M. R.: Learning Nonparametric Kernel Matrices from Pairwise Constraints, in *Proceedings of the 24th international conference on Machine learning*, pp. 361–368, ACM (2007)

[Tang 07] Tang, W., Xiong, H., Zhong, S., and Wu, J.: Enhancing Semi-Supervised Clustering: A Feature Projection Perspective, in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 707–716 (2007)

[Wagstaff 01] Wagstaff, K. and Roger, S.: Constrained K-means Clustering with Background Knowledge, in *Proceedings of the 18th International Conference on Machine Learning*, pp. 577–584 (2001)

Algorithm 2 Modified COP-KMEANS

入力：データ集合 X，制約集合 S，ブースティングステップ t における各制約の重み $w_n^t (n = 1 \sim |S|)$
出力：クラスタ集合 C

初期クラスタ中心を選択する．
初期クラスタ集合 $C_0 \leftarrow \phi$
for $r = 1$ to R **do**
  制約を重み w_n^t の値に基づいて，降順にソートする
  ソートした順に制約データペア (x_i, x_j) をクラスタ中心に割り当てる．(x_i, x_j) がそれぞれ既に割り当てられているか，そうでないかによって場合分けする

  if (x_i, x_j) がともにまだ割り当てられていない場合 **then**
    (x_i, x_j) に最も近いクラスタ中心をそれぞれ c_i，c_j とし，そのクラスタ中心との距離をそれぞれ $d(x_i, c_i)$，$d(x_j, c_j)$ とする
    if (x_i, x_j) が must-link の場合 **then**
      $d(x_i, c_i) < d(x_j, c_j)$ の場合は c_i へ，それ以外の場合は c_j へ (x_i, x_j) を割り当てる．
    else if (x_i, x_j) が cannot-link の場合 **then**
      if $c_i \neq c_j$ の場合 **then**
        x_i を c_i へ，x_j を c_j へそれぞれ割り当てる
      else
        $d(x_i, c_i) < d(x_j, c_j)$ の場合は x_i を c_i へ，x_j を c_j の次に近いクラスタ中心へ割り当てる．
      end if
    end if

  else if x_i が既に割り当てられており，かつ x_j がまだ割り当てられていない場合 **then**
    x_i が割り当てられているクラスタ中心を c_i とする
    if (x_i, x_j) が must-link の場合 **then**
      x_j を c_i へ割り当てる．
    else if (x_i, x_j) が cannot-link の場合 **then**
      c_i と異なり，かつ x_j に最も近いクラスタ中心に x_j を割り当てる．
    end if

  else if x_i がまだ割り当てられておらず，かつ x_j が既に割り当てられている場合 **then**
    x_j が割り当てられているクラスタ中心を c_j とする
    if (x_i, x_j) が must-link の場合 **then**
      x_i を c_j へ割り当てる．
    else if (x_i, x_j) が cannot-link の場合 **then**
      c_j と異なり，かつ x_i に最も近いクラスタ中心に x_i を割り当てる．
    end if
  end if

  制約に関連しない残りのデータをそれぞれ最も距離の近いクラスタ中心へ割り当てる．

  if 前回生成したクラスタ集合から変動がない場合 **then**
    現在のクラスタ集合を出力して，終了．
  else
    クラスタ中心を更新して，次のステップへ．
  end if
end for