行列分解による多クラス分類とその応用

岡野原 大輔† 辻井 潤一†‡§
†東京大学情報理工学系研究科コンピュータ科学専攻
‡School of Computer Science, University of Manchester
§NaCTeM (National Center for Text Mining)
{ hillbig, tsujii }@is.s.u-tokyo.ac.jp

概 要

本稿では，行列分解によって異なるクラス間でパラメータを共有する多クラス分類器を提案する．従来の多クラス分類器は各クラス毎に異なる重みベクトルを利用する線形識別器を利用するため，異なるクラス間でパラメータを共有できない他，訓練例が少ないクラスの学習が困難である問題が存在した．提案手法では各クラスに対応する重みベクトルから構成される重み行列 W を $W = U^T V$ と分解された形で保持し，学習・推論を行う．これは入力および出力を低次元の潜在空間に写像した上での内積を測っていることに対応し，異なるクラス間での情報共有が可能となる．さらに U と V が疎であるような L_1 正則化を適用し，これら行列をハッシュ関数を利用してコンパクトに保持する．言語モデルにおいて提案手法を適用し，提案手法の有効性を調べた．

1 はじめに

与えられた入力に対し，複数の候補解から正解を一つ選ぶタスク，いわゆる多クラス分類は文書分類，構文解析，系列ラベリングなどの多くの自然言語処理タスクの基礎であり重要な問題である．従来のタスクではクラス種類数が小さく，また各クラスの訓練例が十分にあるような問題が扱われていたが，近年では，クラス種類数が数百〜一万といった候補解が非常に大きな問題や，各クラスの訓練例の偏りが大きい問題などが扱われるようになってきている．

例えば，HPSG や CCG のような語彙化文法による構文解析においては，各単語に対し語彙項目を選択する，いわゆる Super-Tagging と呼ばれる処理が行われる．この場合，候補の語彙項目の数は大きく，またその出現偏りも大きい [1]．また，言語モデルのように，次の単語を予測するタスクにおいても N-gram モデルのような単純な統計情報に基づくモデルではなく，各単語が候補解の一つであるような多クラス問題として定式化し，より強力な特徴情報と学習モデルを利用する方法が提案されている [2, 3]．文章中に現れた単語から，Wikipedia のエントリへのマッピングを学習する問題 [4] も，このような問題の一例である．

こうした問題における課題は主に二つある．一つ目は，各クラスの出現回数の偏りが大きく，一部のクラスは学習が十分に行えない場合があることである．例えば，言語モデルにおいては，訓練例中に数回しか出現しない単語が大部分を占めており，これらの推定は大幅に難しくなる．二つ目はパラメータ保持のコストが大きいことである．従来の線形識別器の多くは，入力に依存した特徴ベクトルと出力に依存した特徴ベクトルの直積を利用して，特徴ベクトルを定義しており，特徴種類数，候補回数が大きい場合，パラメータ数は非常に大きくなる．

我々はこれら二つの問題を解決するために，次のような多クラス分類器を提案する．まず，異なるクラス間の情報を共有するために，各クラスに対応する重みベクトルからなる重み行列 W が二つの行列に分解できることを想定する $W = U^T V$．これは協調フィルタリングなどで使われている手法と同じである．次にこれらの重みベクトルが疎であるように L_1 正則化を適用し，さらにハッシュ関数を利用してこれらの行列を明示的に持たずにコンパクトな表現を実現する．

2 関連研究

今回提案した手法のように，行列を分解することにより，特徴情報を共有する手法はマルチタスク学習や多クラス行列などの分野で多く提案されてきた．これらの手法では元の重み行列を陽に保持し，行列分解による特徴共有の効果を元の重み行列に対する様々な正則化の形で適用する．

Amit ら [5] は，重み行列 W が $W = U^T V$ と分解された上で，それらの行列にフロベニウスノルムによる正則化が適用された形での学習を提案している

$$\arg\min_{U,\,V} \frac{1}{2}\|U\|_F^2 + \frac{1}{2}\|V\|_F^2 + C\sum_{i=1}^{n} l(\mathbf{x},\, y; U^T V) \quad (1)$$

ただし $\|A\|_F^2 = \sum_{i,\,j} a_{ij}^2$ は A のフロベニウスノルムであり，$l(\mathbf{x}_i,\, y_i; U^T V)$ は各訓練例に関する損失関数である．この時，フロベニウスノルムの和は，以下のように $W = U^T V$ のトレースノルムに対応することが知られている

$$\begin{aligned} \|W\|_\Sigma &= \min_{U^T V = W} \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2) && (2)\\ &= \sum_i |\gamma_i|, && (3) \end{aligned}$$

但し，γ_i は W の特異値である．この直接の最適化は難しいが，[5] らは γ_i を微分可能な関数で近似し，W を勾配法で直接最適化する手法を提案している．

Argyriou らは [6] らは同様の行列分解を V が直交行列の場合で考えている．この場合も陽に重み行列を保持する必要があるため，必要な計算リソースは非常に大きい．

我々の研究にもっとも近い研究として Bai ら [7] が Polynomial Semantic Indexing を提案している．この手法は与えられたクエリに対して，各文書のスコアを決定するランキング学習の問題を扱っている．具体的には，クエリと文書がともに Bag-of-words 表現によって $\mathbf{q} \in R^m$, $\mathbf{d} \in R^m$ と表されているときに，クエリ q に対する文書 d に対するスコア $f(q,\, d)$ を次のように求める．

$$f(q,\, d) = \sum_{i,\,j} W_{i,\,j} q_i d_j = \mathbf{q}^T W \mathbf{d}, \quad (4)$$

但し，$W \in R^{m \times m}$ は重み行列である．ランキングは $f(q,\, d)$ が大きい順に文書をソートすることにより実現できる．この時，異なる単語間で情報を共有できるように，重み行列を $W = U^T V + I$ と表す，但し $I \in R^{m \times m}$ は単位行列である．これは行列分解によって異なる単語間の情報をとらえてスコアが決定されるモデルになっており，また I を加えることによって，もとの単語による共起情報 ($q^T d$) が考慮されている．学習の時は，各クエリに対して関連した文書 d^+ と，関連しなかった文書 d^- が与えられ $f(q,\, d^+) > f(q,\, d^-)$ となるように U と V の更新を行う．

我々の提案手法は，これら既存研究と同じように異なるクラス間の情報共有を重み行列の行列分解によって実現する．さらに非常に巨大な行列を扱えるよう，行列に対して疎になるよう L_1 正則化を適用し，さらにハッシュ表現によってこれらの行列のコンパクトな表現を実現する．

3 行列分解を用いた多クラス分類

本稿では，入力 $\mathbf{x} \in R^m$ が与えられた時，r 個の候補解の中から正解 $y \in \{1, 2, \ldots, r\}$ を一つ推論する他クラス分類問題を考える．我々のモデルでは従来手法と同様に，次のような線形識別器を利用する．入力 $\mathbf{x}$ に対し出力が y であった時のスコアを次のように定義する．

$$s(\mathbf{x},\, y) = \mathbf{w}_y^T \mathbf{x} \quad (5)$$

但し，$\mathbf{w}_y \in R^m$ は候補解 y に対応する m 次元の重みベクトルである．

そして，このスコアが大きい候補解を $\boldsymbol{x}$ に対する予測とする，

$$y^* = \arg\max_y \; s(\mathbf{x},\, y). \quad (6)$$

多クラス SVM，多クラスロジスティック回帰モデルなど多くの分類器がこの形で表される．

次に $\mathbf{e}_i \in R^r$ を i 番目の要素のみが 1 であり，それ以外は 0 であるようなベクトルとする．また，i 列目が $\mathbf{w}_i$ からなる重み行列 $W \in R^{m \times r}$ を定義する．すると先程のスコアは次のように再定式化される，

$$s(\mathbf{x},\, y) = \mathbf{x}^T W \mathbf{e}_y. \quad (7)$$

次に，この重みベクトルに関する情報を複数のクラス間で共有するために，重み行列 W が $W = U^T V$ と二つの行列 $U \in R^{k \times m}$，$V \in R^{k \times m}$ の積として表す．このモデルでは異なるクラスが重みを共有できる．例えば，性質が似た候補解が存在した場合，それらに対する重みベクトルは似ているはずでありそれらの情報は共有される．スコアは U，V を用いて次のように表される．

$$\begin{aligned} s(\mathbf{x},\, y) &= \mathbf{x}^T W \mathbf{e}_y && (8)\\ &= \mathbf{x}^T U^T V \mathbf{e}_y && (9)\\ &= (U\mathbf{x})^T V \mathbf{e}_y && (10) \end{aligned}$$

このことから入力と出力はそれぞれ $U\mathbf{x} \in R^k$, $V\mathbf{e}_y \in R^k$ によって，それぞれ，k 次元の潜在空間にマッピングされ，そこでの内積を計算していると考えることができる．

本手法では最終的なモデルは従来の線形分類器と変わらないが，隠れ層で異なるクラスに関する学習例が利用されるため，より効率的に学習できることが期待される．

このモデルは隠れ層が一層であるようなニューラルネットワークと似ているが，提案手法はあくまで最終的な関数は線形であるのに対して，多層パーセプトロンでは，隠れ層で非線形の変換を行う場合，最終的な関数は非線形という違いがある．提案手法はこのことから、効率的に学習、推論が行える一方，多層パーセプトロンの方が強力なモデルになっている．

4 オンライン学習

次にモデルパラメータ U，V のオンライン学習を考える．行列が分解されている場合の学習問題の多くは凸最適化ではなく，今回の学習で解くのも凸最適化ではない．n 個の訓練例 $\{(\mathbf{x}^{(i)}, y^{(i)}\}$ $(i = 1, \ldots, n)$ が与えられたとする．学習では各訓練例に対し，正解のラベルが不正解のラベルよりも高いスコアであるようにパラメータを更新する．

具体的には次の最適化問題を解くことでパラメータを学習する．

$$W^* = \arg\min_W L(W) \tag{11}$$

$$L(W) = \sum_{i=1}^{n} l(\mathbf{x}^{(i)}, y^{(i)}) \tag{12}$$

$$l(\mathbf{x}, y) = [1 - s(\mathbf{x}, y) + s(\mathbf{x}, y^*)]_+ \tag{13}$$

但し，$[a]_+ = \max(0, a)$ であり $y^* = \arg\max_{y' \neq y} s(\mathbf{x}, y')$ である．これをオンライン学習で行う場合，ランダムに訓練例 $(\mathbf{x}, y)$ を選び，$1 - s(\mathbf{x}, y) + s(\mathbf{x}, y^*) > 0$ の時，次のようにパラメータを更新する

$$U := U + \lambda V(\mathbf{e}_y - \mathbf{e}_{y*})\mathbf{x}^T \tag{14}$$

$$V := V + \lambda U\mathbf{x}(\mathbf{e}_y - \mathbf{e}_{y*})^T \tag{15}$$

但し，$0 < \lambda < 1$ はユーザーが設定する更新幅である．実装で，毎回訓練例毎にランダムに (14)，(15) のどちらかのみの更新を行った．

次に，U と V が疎になるようにするために，L_1 正則化を適用し，最終的な目的関数は次の通りとなる．

$$L(W) = \sum_{i=1}^{n} l(\mathbf{x}^{(i)}, y^{(i)}) + C(\|U\|_1 + \|V\|_1) \tag{16}$$

但し $C > 0$ はハイパーパラメータであり，重み行列がどの程度疎になるかどうかを決定する．

この更新は各パラメータの更新時に定数を引く操作によって実現される [8]．実際には更新時には少数のパラメータのみが関係するので，各パラメータにおいて累積されたペナルティを保持し，パラメータ更新時にこの累積ペナルティをまとめて適用する．

U と V の初期値は Gaussian 分布からランダムに決定した．

5 ハッシュを利用したコンパクト表現

L_1 正則化を与えることにより U と V はともに疎になるように学習される．これは，各特徴が少数の潜在特徴のみに影響を与えることを意味している．

しかしながら，U と V はともに非常に大きいため，そのまま保存するには多くの依然として多くのリソースを必要とする．このため，我々では Hash Kernel [9] を利用し行列をコンパクトに表現する．

Hash Kernel においては，ベクトル $\boldsymbol{x}$ は次のように $\phi(x) \in R^t$ で表される

$$\phi_i(x) = \sum_{j:h(j)=i} \xi(j)x_j \tag{17}$$

但し，h，ξ はそれぞれ入力を $\{-1, +1\}, [1 \ldots, t]$ に写像するようなハッシュ関数である．これを利用し特徴ベクトル $\boldsymbol{x}$，$\boldsymbol{y}$ 間の内積を $\phi(x)^T\phi(y)$ として計算した場合，これが元の内積とどの程度近くなるかについては次の結果が知られている [9].

$$E_{h,\xi}[\phi(x)^T\phi(y)] = \boldsymbol{x}^T\boldsymbol{y} \tag{18}$$

$$V_{h,\xi}[\phi(x)^T\phi(y)] = \mathrm{O}(1/t) \tag{19}$$

但し，$E_{h,\xi}$，$V_{h,\xi}$ はそれぞれ確率変数 h,ξ に対する期待値と分散である．

本稿ではこれらのハッシュによる表現を行列 U と V に適用する．具体的には，U と V を保持する代わりに長さ t のベクトルを保持し、各行列の i 行 j 列の成分にアクセスする場合は i と j を入力とするハッシュ関数を利用してベクトル中の添字番号を計算し，そこを行列の成分値とみなして操作を行う．

6 推論

ここでは訓練 $\boldsymbol{x}$ に対する推論，つまり次の操作

$$\arg\max_y \quad s(\mathbf{x}, y) = \arg\max_y \quad \mathbf{x}^T W\mathbf{e}_y \tag{20}$$

を求める問題を考える．まず，候補解の数が非常に多いため，全ての候補解 y について $s(\mathbf{x}, y)$ を調べ最大値を求めるのは計算量が大きい．

表 1: 言語モデルにおける

Method	CSJ (Perp.)	BNC (Perp.)
Trigram + 修正 KN	133.5	230.3
提案手法 (k=10000)	125.7	220.0

まず $\mathbf{z} = U\mathbf{x} \in R^k$ とおく．これは入力 $\mathbf{x}$ に対する各潜在値のスコアであり，$\mathbf{x}$ が疎な場合高速に求めることができる．つぎに実際に $\mathbf{z}$ が疎であることを利用して $\mathbf{z}V$ を高速に計算し，各ラベルのスコアを求める．

7 実験

提案手法の有効性を示すために，次の単語を予測する言語モデルにおける評価を行った．コーパスとして日本語話し言葉コーパス (CSJ) と BNC Corpus(BNC) を利用し，これらをそれぞれ訓練データとテストデータに分けた．訓練データにおける単語数はそれぞれ 5379 万と 885 万であり単語種類数は約 116 万と約 3 万であった．

提案手法はもともと確率モデルではないが次のようにして確率を推定するようにした．

$$p(y|\boldsymbol{x}) = \frac{\exp(s(\boldsymbol{x}, y))}{\sum_{y'} \exp(s(\boldsymbol{x}, y'))} \tag{21}$$

これは多クラス Logistic 回帰と同じ式である．但し，学習時の更新式はそのままのものを利用した．

ベースラインとして Tri-gram + 修正 Knerser-Ney Smoothing による手法と比較を行った．提案手法の特徴としては Trigram と同様に 3 単語列までの N-gram を利用した．表 1 に結果を示す．提案手法の方が精度が優れているが，差はそれほど大きくなかった．この原因として提案手法が確率モデルに対する最適化ではなく，マージンベースの最適化であることが挙げられる．

8 まとめと今後の課題

本稿では，行列分解を利用した多クラス分類器を提案した．提案手法は異なるクラス間の情報を共有するために重み行列を分解された形で保持し，それらを更新するように学習を行う．今回は言語モデルタスクに対し本手法の有効性を検証したが，他の多クラス問題に適用していきたいと考えている．

提案手法は近年新たに注目を受けている Deep Neural Network [10] のように隠れ層を含むモデルを利用した学習と考えることができる．今後隠れ層やそれらの依存関係をうまく設定するなどしてより強力な予測器が得られると考えられ，それらの効率的な学習やコンパクト表現が重要であると考えられる．

参考文献

[1] T. Matsuzaki, Y. Miyao, and J. Tsujii. Efficient hpsg parsing with supertagging and cfg-filtering. In *Proc. of IJCAI*, pages 1671–1676, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[2] S. F. Chen. Shrinking exponential language models. In *Proc. of NAACL*, pages 468–476, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[3] S. F. Chen. Performance prediction for exponential language models. In *Proc. of NAACL*, pages 450–458, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[4] D. Milne and Ian H I. H. Witten. Learning to link with wikipedia. In *Proc. of CIKM*, pages 509–518, New York, NY, USA, 2008.

[5] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proc. of ICML*, pages 17–24, 2007.

[6] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[7] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, C. Cortes, and M. Mohri. Polynomial semantic indexing. In *Proc. of NIPS*, 2009.

[8] Y. Tsuruoka, J. Tsujii, and S. Ananiadou. Stochastic gradient descent training for L1-regularized log-linear models with cumulative penalty. In *Proc. of ACL IJCNLP*, pages 477–485, 2009.

[9] Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, and S. V. N. Vishwanathan. Hash kernels for structured data. *JMLR*, 10:2615–2637, 2009.

[10] G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.