F-002

SVM 型多クラス分類器のためのフランクウルフ学習法
Frank-Wolfe Learning Algorithm For SVM-type Multi-Category Classifiers

田島 賢哉 * 加藤 毅 *
Kenya Tajima Tsuyoshi Kato

1. はじめに

多クラス分類とは，あらかじめ定めておいたクラスラベルの一つを入力データに割り当てるタスクであり，多くの教師付き機械学習の問題は多クラス分類問題に帰着される．伝統的には，パターン認識の理論や研究は，より単純な 2 クラス分類を通して発達した歴史を持ち，サポートベクトルマシン (SVM) はその過程で生まれた方法論のうち最も成功したアルゴリズムである．９０年代においては，従来の SVM を多クラス分類問題に適用するには，One-vs-rest というアプローチが用いられていた．このアプローチは，クラスごとに独立な問題に分解して SVM を適用するものであった．そのため，クラス間の相関を学習できないという弱点があった．Crammer&Singer は，これに代わる方法として，多クラス分類器を一つの最適化問題で表現する方法を提案した [2]．この方法は，**多クラス SVM**(**MC-SVM**) と呼ばれる．多クラス SVM の出現以降，文法解析 [5]，変形部品モデル [3] などに応用される**構造 SVM** [12]，多変量性能基準を最適化する $\mathbf{SVM_{multi}}$ [6]，大規模多クラス分類タスクに対応するための**トップ-k SVM** [9]，ランキング予測 [13] など，Crammer&Singer のモデルを基盤とした数多くの変種が出現した．

上述の変種は，安定で効率的な学習アルゴリズムを伴わなければ，実用的とは言えない．これらの多くは，予測器ごとに特化した形で学習アルゴリズムが開発されてきた．たとえば，構造 SVM および $\mathrm{SVM}_{\mathrm{multi}}$ に対しては，切断平面法が用いられた [7]．トップ-k SVM には，座標上昇法に基づくアルゴリズムが 2 つの研究グループによって開発された [9, 1]．しかし，両方のアルゴリズムは誤った理論から導かれたものであったため，最適解に到達できないという欠陥を抱えていた [8]．Kato & Hirohashi は，トップ-k SVM の双対問題に**フランクウルフ法** [4] を適用すると，フランクウルフ法における 2 ステップ，すなわち，**方向決定ステップ**および**線探索ステップ**はともに閉形式で表すことが出来，その計算量は $O(mn^2)$ で抑えられることを示した．フランクウルフ法は，凸多面体上で凸関数を最小化するための汎用最適化アルゴリズムであり，方向決定ステップおよび線探索ステップを正確に実行すれば**劣線形収束**することが保証されている．

本研究では，トップ-kSVM 以外の多クラス SVM の変種に対しても，双対問題にフランクウルフ法を適用した時，方向決定ステップおよび線探索ステップがともに閉じた形で表されることを発見した．本論文では，各ステップが閉形式になるための条件を明らかにする．機械学習にしばしば用いられる勾配法との最も大きな違いは，フランクウルフ法は，ステップサイズのような，学習問題ごとに調整しなくてはならないパラメータがないこと，および解の精度が保証できることである．本研究の発見によって，多クラス SVM から派生した広範な変種の学習において，ステップサイズ不要で，解の精度が保証される最適化アルゴリズムが利用できるようになる．

さらに，本研究の成果を損失関数の**モーロー包**をとった場合にも拡張した．モーロー包をとると関数は滑らかになり，一般に最適解への収束が高速化されるため，モーロー包は機械学習において広く用いられている．本研究では，損失関数のモーロー包をとっても，フランクウルフ法の各ステップは閉じた形で表されることを見出した．証明や導出は文献 [11] に記載する．

記法: $\pi(j; \boldsymbol{s}) \in [m] := \{1, \ldots, m\}$ はベクトル $\boldsymbol{s} \in \mathbb{R}^m$ における j 番目に大きい要素の添え字を表す．曖昧性の危険がないときは，$\boldsymbol{s}$ を省略することもある．ベクトル $\boldsymbol{s}$ の要素の大きさによってベクトル $\boldsymbol{x}$ の要素を並び替えたベクトルを $\boldsymbol{x}_{\boldsymbol{\pi}(\boldsymbol{s})} := [x_{\pi(1;\boldsymbol{s})}, \ldots, x_{\pi(m;\boldsymbol{s})}]^\top$ と表す．$\boldsymbol{e}_i$ は第 i 要素のみ 1 の単位ベクトル．$\mathbf{1}$ は全要素が 1 のベクトル．

2. 多クラス SVM とその変種

一般性を失わずに，MC-SVM [2] におけるクラス数を m とし，そのクラスラベルを $\{1, \ldots, m\}$ で表すとする．入力 $\boldsymbol{x} \in \mathcal{X}$ に対して，クラス j に対する特徴ベクトルを $\boldsymbol{\psi}(\boldsymbol{x}, j) \in \mathbb{R}^d$ と表す．MC-SVM では，分類器のモデルパラメータ $\boldsymbol{w} \in \mathbb{R}^d$ を使って，$\operatorname*{argmax}_{j \in [m]} \langle \boldsymbol{w}, \boldsymbol{\psi}(\boldsymbol{x}, j) \rangle$ を予測結果とする．訓練用例題集合 $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n) \in \mathcal{X} \times [m]$ を使って，モデルパラメータ $\boldsymbol{w}$ の値を決定するために，次の目的関数を最小化する：

$$P(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \Phi(\boldsymbol{\Psi}(\boldsymbol{x}_i)^\top \boldsymbol{w}\,;\, y_i). \quad (1)$$

ただし，$\boldsymbol{\Psi}(\boldsymbol{x}_i)$ は m 個の特徴ベクトルを連結した行列である (i.e. $\boldsymbol{\Psi}(\boldsymbol{x}) := [\boldsymbol{\psi}(\boldsymbol{x}, 1), \ldots, \boldsymbol{\psi}(\boldsymbol{x}, m)] \in \mathbb{R}^{d \times m}$)；$\lambda$ は正則化定数で正の値が設定される；$\Phi(\cdot\,;\, y) : \mathbb{R}^m \to \mathbb{R}$ は損失関数であり，MC-SVM では $\Phi(\cdot\,;\, y)$ に**マックスヒンジ損失** $\Phi_{\mathrm{mh}}(\cdot\,;\, y)$ [2] を採用している．マックスヒンジ損失はクロネッカデルタ $\delta_{\cdot,\cdot}$ を使って，

$$\Phi_{\mathrm{mh}}(\boldsymbol{s}\,;\, y) := \max_{j \in [m]} (s_j - s_y + 1 - \delta_{j,y}) \quad (2)$$

と定義されている．

フェンシェル双対： 目的関数 $P(\cdot)$ のフェンシェル双対は

$$D(\boldsymbol{A}) := -\frac{\lambda}{2}\|\boldsymbol{w}(\boldsymbol{A})\|^2 - \frac{1}{n}\sum_{i=1}^{n} \Phi^*(-\boldsymbol{\alpha}_i\,;\, y_i). \quad (3)$$

*群馬大学理工学府

で表される．ただし，$\boldsymbol{A} := [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n] \in \mathbb{R}^{m \times n}$；$\boldsymbol{w}(\boldsymbol{A}) := \frac{1}{\lambda n} \sum_{i=1}^{n} \boldsymbol{\Psi}(\boldsymbol{x}_i)\boldsymbol{\alpha}_i$；$\Phi^*(\cdot\,; y)$ は $\Phi(\cdot\,; y)$ の**凸共役**である．目的関数 $P(\boldsymbol{w})$ を直接最小化する代わりに，フェンシェル双対 $D(\boldsymbol{\alpha})$ を最大化して得られる $\boldsymbol{A}_\star$ を使って，主変数の最小解 $\boldsymbol{w}_\star := \boldsymbol{w}(\boldsymbol{A}_\star)$ を得ることができる．最適解では双対ギャップ $P(\boldsymbol{w}(\boldsymbol{A}_\star)) - D(\boldsymbol{A}_\star)$ が 0 になる．このことから，十分小さい正の定数 ϵ を使って，$P(\boldsymbol{w}(\boldsymbol{A})) - D(\boldsymbol{A}) \leq \epsilon$ を満たしたとき，反復を停止すれば，主変数 $\boldsymbol{w} = \boldsymbol{w}(\boldsymbol{A})$ の目的誤差 $P(\boldsymbol{w}) - P(\boldsymbol{w}_\star)$ が ϵ 以下になることが保証される．

構造 SVM: Crammer & Singer の MC-SVM [2] では正解以外の予測結果に対して，一律 1.0 の損失を与える 0/1 損失を想定し，その凸代替損失 (convex surrogate loss) を採用していた．構造 SVM [12] では，第 i 訓練用例題に対して，予測結果がクラス $\hat{y}$ となるとき (i.e. $\operatorname{argmax}_{j \in [m]} \langle \boldsymbol{w}_j, \boldsymbol{\psi}(\boldsymbol{x}_i, j) \rangle = \hat{y}$ となるとき)，任意の非負の損失 $\Delta_j^{(i)}$ となるような損失関数を想定し，その凸代替損失

$$\Phi_{i,\mathrm{sh}}(\boldsymbol{s}\,; y) := \max_{j \in [m]} \left(s_j - s_y + \Delta_j^{(i)} \right) \tag{4}$$

を採用した．

重みなしトップ-k SVM :重みなしトップ-k SVM [9] も MC-SVM の変種の一つである．MC-SVM は入力 $\boldsymbol{x} \in \mathcal{X}$ に対して，一つのクラスラベルを割り当てる予測器を学習する方法であった．トップ-k SVM は，m 個の予測スコア $\boldsymbol{s} = [s_1, \ldots, s_m]^\top := \boldsymbol{\Psi}(\boldsymbol{x})^\top \boldsymbol{w}$ のうち最も k 個のスコアに対応するクラス $\{\pi(1; \boldsymbol{s}), \ldots, \pi(k; \boldsymbol{s})\}$ を予測結果とする．トップ-k SVM では，このような予測器の学習を行うために，その損失関数は，

$$\Phi_{utk}(\boldsymbol{s}\,; y) := \max \left\{ 0, \frac{1}{k} \sum_{j=1}^{k} (\boldsymbol{s} - s_y \mathbf{1} + \mathbf{1} - \boldsymbol{e}_y)_{\pi(j)} \right\} \tag{5}$$

のように設計されている．

重みなし Usunier SVM: 重みなし Usunier SVM は，Usunier ら [13] がランキングのために導入した損失関数をトップ-k の予測結果を出力する多クラス分類器を学習できるように再設計した損失関数

$$\Phi_{uu}(\boldsymbol{s}\,; y) := \frac{1}{k} \sum_{j=1}^{k} \max \left\{ 0, (\boldsymbol{s} - s_y \mathbf{1} + \mathbf{1} - \boldsymbol{e}_y)_{\pi(j; \boldsymbol{s} - \boldsymbol{e}_y)} \right\} \tag{6}$$

を使って学習される．

重みつきトップ-k SVM : Kato & Hirohashi [8] は重みなしトップ-k 損失関数 (5) を $\rho_1 \geq \ldots \geq \rho_{m-1} \geq \rho_m = 0$ なる重み $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_m]^\top \in \mathbb{R}^m$ を使って，次のように拡張した：

$$\Phi_{\mathrm{wtk}}(\boldsymbol{s}\,; y) := \max \left\{ 0, \sum_{j=1}^{m} (\mathbf{1}_m - \boldsymbol{e}_y + \boldsymbol{s} - s_y \mathbf{1}_m)_{\pi(j)} \, \rho_j \right\}. \tag{7}$$

重みつき Usunier SVM: 重みなし Usunier 損失も，$\rho_1 \geq \ldots \geq \rho_{m-1} \geq \rho_m = 0$ なる重み $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_m]^\top \in \mathbb{R}^m$ を使って，次のような拡張を考えることができる：

$$\Phi_{\mathrm{wu}}(\boldsymbol{s}\,; y) := \sum_{j=1}^{m} \max \left\{ 0, (\mathbf{1}_m - \boldsymbol{e}_y + \boldsymbol{s} - s_y \mathbf{1}_m)_{\pi(j)} \right\} \rho_j. \tag{8}$$

3.Max Dot Over Simplex 型損失関数

本節では，本研究で開発した学習アルゴリズムが適用可能な学習機械を定式化する．ターゲットとなる学習機械は (1) で表される正則化経験リスクの最小化によって $\boldsymbol{W}$ の値を決定する．その損失関数 $\Phi(\cdot\,; y)$ は，次に与える **Max Dot Over Simplex 型** (**mdos 型**) とする．

Definition 3.1. 関数 $\Phi : \mathbb{R}^m \to \mathbb{R}$ に対して，$\forall y \in [m]$, $\boldsymbol{s} \in \mathbb{R}^m$,

$$\Phi(\boldsymbol{s}\,; y) = \max_{\boldsymbol{\beta} \in \mathcal{B}} \langle \boldsymbol{\beta}, \mathbf{1} - \boldsymbol{e}_y + \boldsymbol{s} - s_y \mathbf{1} \rangle \,. \tag{9}$$

なるコンパクトな単体 $\mathcal{B}$ が存在するとき，Φ は *mdos* 型であるという．

前節で紹介したマックスヒンジ損失 (2)，**構造ヒンジ損失** (4)，**重みなしトップ-k ヒンジ損失** (5)，**重みなし Usunier 損失** (6)，**重みつきトップ-k ヒンジ損失** (7)，**重みつき Usunier 損失** (8) のいずれの損失関数も mdos 型であることを示すことができる．それぞれ，対応する単体は次のように与えられる：

- マックスヒンジ損失 (2) および構造ヒンジ損失 (4) に対する単体：$\mathcal{B}_{\mathrm{sh}} := \Delta(1)$．ただし，
$$\Delta(r) := \left\{ \boldsymbol{\beta} \in \mathbb{R}_+^m \,\middle|\, \|\boldsymbol{\beta}\|_1 \leq r \right\} \tag{10}$$
- 重みなしトップ-k ヒンジ損失 (5) に対する単体：$\mathcal{B}_{\mathrm{utk}} := \Delta_{\mathrm{tk}}(k, 1)$．ただし，
$$\Delta_{\mathrm{tk}}(k, r) := \left\{ \boldsymbol{\beta} \in \Delta(r) \,\middle|\, \boldsymbol{\beta} \leq \frac{\|\boldsymbol{\beta}\|_1}{k} \mathbf{1} \right\} \tag{11}$$
- 重みなし Usunier 損失 (6) に対する単体：$\mathcal{B}_{\mathrm{uu}} := \Delta_{\mathrm{u}}(k, 1)$．ただし，
$$\Delta_{\mathrm{u}}(k, r) := \left\{ \boldsymbol{\beta} \in \Delta(r) \,\middle|\, \boldsymbol{\beta} \leq \frac{1}{k} \mathbf{1} \right\} \tag{12}$$
- 重みつきトップ-k ヒンジ損失 (7) に対する単体 [8]:
$$\mathcal{B}_{\mathrm{wtk}} := \left\{ \boldsymbol{\beta} \in \mathbb{R}^m \,\middle|\, \exists \zeta \in \mathbb{R},\ \forall \ell \in [L], \right.$$
$$\left. \exists \boldsymbol{\lambda}_\ell \in \Delta_{\mathrm{tk}}(k_\ell, \rho'_\ell k_\ell),\ \zeta = \frac{\langle \mathbf{1}, \boldsymbol{\lambda}_\ell \rangle}{k_\ell \rho'_\ell},\ \boldsymbol{\beta} = \sum_{\ell=1}^{L} \boldsymbol{\lambda}_\ell \right\}, \tag{13}$$

- 重みつき Usunier 損失 (8) に対する単体：

$$\mathcal{B}_{\mathrm{wu}} := \Big\{ \boldsymbol{\beta} \in \mathbb{R}_+^m \,\Big|\, \forall \ell \in [L], \\ \exists \boldsymbol{\lambda}_\ell \in \Delta_{\mathrm{u}}(1/\rho'_\ell, k_\ell \rho'_\ell),\, \boldsymbol{\beta} \le \sum_{\ell=1}^{L} \boldsymbol{\lambda}_\ell \Big\}. \tag{14}$$

式 (13), (14) における L および $\rho'_1, \ldots, \rho'_L$ は次のように定義されるものである：自然数 L は次の集合の要素数を表す：

$$\mathcal{K} := \{ k \in [m] \,|\, \rho_k > \rho_{k+1} \}. \tag{15}$$

集合 $\mathcal{K}$ に含まれる L 要素を $k_1, \ldots, k_L$ とする．ただし，$1 \le k_1 < \cdots < k_L < m$ のように整列しておく．$\rho'_1, \ldots, \rho'_L$ は $\rho'_\ell := \rho_{k_\ell}$ のように定義した．

以上の結果は，次の定理でまとめられる．

Theorem 3.1. 5 つの損失関数 $\Phi_{sh}(\cdot; y)$, $\Phi_{utk}(\cdot; y)$, $\Phi_{uu}(\cdot; y)$, $\Phi_{wtk}(\cdot; y)$, および $\Phi_{wu}(\cdot; y)$ はいずれも *mdos* 型である．

Theorem 3.1 の証明は [11] 参照．

4. フランクウルフ法

本節では，mdos 型損失関数を持つ MC-SVM の変種に対する学習アルゴリズムを示す．本研究で開発した最適化アルゴリズムは，フランクウルフ法を使って双対目的関数 $D(\boldsymbol{A})$ を最大化するものである．フランクウルフ法は，反復的に双対目的関数 $D(\boldsymbol{A})$ を最大化する．フランクウルフ法の各反復は，**方向決定ステップ**と**線探索ステップ**からなる．反復 t における解を $\boldsymbol{A}^{(t)} = \left[\boldsymbol{\alpha}_1^{(t)}, \ldots, \boldsymbol{\alpha}_n^{(t)}\right]$ と書くとすると，方向決定ステップでは，次の線形計画問題を解く：

$$\boldsymbol{U}^{(t-1)} \in \operatorname*{argmax}_{\boldsymbol{U} \in \mathrm{dom}(-D)} \left\langle \nabla D(\boldsymbol{A}^{(t-1)}), \boldsymbol{U} \right\rangle. \tag{16}$$

線探索ステップでは，$\boldsymbol{A}^{(t-1)}$ と $\boldsymbol{U}^{(t-1)}$ をつなぐ線分上で，最適な点を探索する：

$$\gamma^{(t-1)} := \operatorname*{argmax}_{\gamma \in [0,1]} D\left((1-\gamma)\boldsymbol{A}^{(t-1)} + \gamma \boldsymbol{U}^{(t-1)} \right). \tag{17}$$

これら $\boldsymbol{U}^{(t-1)}$ および $\gamma^{(t-1)}$ を使って，解を

$$\boldsymbol{A}^{(t)} := (1-\gamma^{(t-1)})\boldsymbol{A}^{(t-1)} + \gamma^{(t-1)} \boldsymbol{U}^{(t-1)} \tag{18}$$

のように更新する．効率的に最大化問題を解くうえでキーとなるは方向決定ステップ (16) である．目的関数 $-D$ の有効ドメイン $\mathrm{dom}(-D)$ が単体ならば，方向決定ステップ (16) は汎用線形計画ソルバーを使えば $\boldsymbol{U}^{(t-1)}$ を求められる．しかし，各反復で汎用線形計画ソルバーを使ってしまうと多大な計算時間を要してしまうことになる．これに対して，本研究では次の定理を見つけた：

Theorem 4.1. $D(\boldsymbol{A})$ を $\boldsymbol{A}$ に関して最大化する問題にフランクウルフ法を適用することを考える．損失関数 $\Phi(\cdot; y)$ は *mdos* 型と仮定する．すると，方向決定ステップも線探索ステップも閉形式で表される．

Theorem 4.1 の証明は [11] 参照．具体的には，方向決定ステップ (16) の解は次のように与えられることを発見した：$\boldsymbol{U}^{(t-1)}$ の第 i 列 $\boldsymbol{u}_i^{(t-1)}$ は

$$\boldsymbol{u}_i^{(t-1)} \in -\partial\Phi\left(\boldsymbol{\Psi}(\boldsymbol{x}_i)^\top \boldsymbol{w}(\boldsymbol{A}^{(t-1)}); y_i \right) \tag{19}$$

を満たすように設定する．ただし，$\partial\Phi(\cdot; y)$ は $\Phi(\cdot; y)$ の劣微分である．

線探索ステップも $\gamma^{(t-1)} = \max(0, \min(1, \hat{\gamma}^{(t-1)}))$ のように閉形式解で得られる．ただし，

$$\hat{\gamma}^{(t-1)} := \frac{\sum_{i=1}^n \left\langle \Delta\boldsymbol{\alpha}_i^{(t-1)}, \boldsymbol{e}_{y_i} - \boldsymbol{\Psi}(\boldsymbol{x}_i)^\top \boldsymbol{w}(\boldsymbol{A}^{(t-1)}) \right\rangle}{\lambda n \|\boldsymbol{w}(\Delta\boldsymbol{A}^{(t-1)})\|^2}, \tag{20}$$

$$\Delta\boldsymbol{A}^{(t-1)} := \boldsymbol{U}^{(t-1)} - \boldsymbol{A}^{(t-1)},$$

$\Delta\boldsymbol{\alpha}_i^{(t-1)}$ は $m \times n$ 行列 $\Delta\boldsymbol{A}^{(t-1)}$ の第 i 列である．

5. モーロー包への拡張

モーロー包は微分不可能な凸関数を平滑な関数に変換する技法である．例えば，今日広く使われている**平滑化ヒンジ損失**や**フーバー損失**は，それぞれ，ヒンジ損失および絶対値誤差損失のモーロー包である．多くの最適化アルゴリズムは平滑な目的関数のほうが短時間で最適解に収束させることができるため，モーロー包は学習を高速化するための有用な手段になる．凸な損失関数 $\Phi(\cdot; y) : \mathbb{R}^m \to \mathbb{R}$ のモーロー包は以下で与えられる：

$$\Phi_{\mathrm{m}}(\boldsymbol{s}; y) := \left(\Phi^*(\cdot; y) + \frac{\gamma}{2}\|\cdot\|^2 \right)^* (\boldsymbol{s}; y). \tag{21}$$

ただし，γ は平滑化パラメータと呼ばれる正の定数である．$\Phi(\cdot; y)$ が凸関数である限り，そのモーロー包 $\Phi_{\mathrm{m}}(\cdot; y)$ は $(1/\gamma)$-**平滑**が保証される．このモーロー包に基づく正則化経験リスク

$$P_{\mathrm{m}}(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^n \Phi_{\mathrm{m}}(\boldsymbol{\Psi}(\boldsymbol{x}_i)^\top \boldsymbol{w}; y_i) \tag{22}$$

を最小化するために，そのフェンシェル双対

$$D_{\mathrm{m}}(\boldsymbol{A}) := -\frac{\lambda}{2}\|\boldsymbol{w}(\boldsymbol{A})\|^2 - \frac{1}{n}\sum_{i=1}^n \Phi_{\mathrm{m}}^*(-\boldsymbol{\alpha}_i; y_i) \tag{23}$$

をフランクウルフ法で最大化することを考える．ここで留意すべきことは，損失関数 $\Phi(\cdot; y)$ が mdos 型で

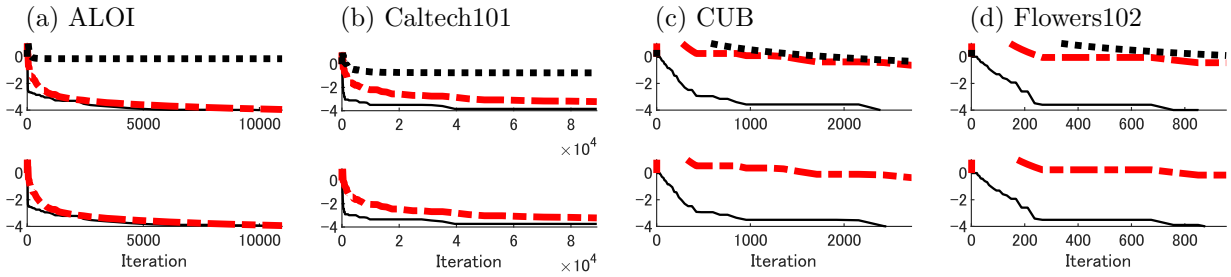

図 1: 収束のふるまい．すべてのグラフの横軸は反復数，上段のグラフの縦軸は目的誤差の常用対数，下段の縦軸は双対ギャップの常用対数を示す．実線は LSFW 法，破線は Std FW 法，点線は GD 法．

あっても，そのモーロー包はもはや mdos 型ではないことである．これにより，前節で示した最適化アルゴリズムは，直接，利用できなくなる．これに対して，本研究では次の発見をした：

Theorem 5.1. $D_m(\boldsymbol{A})$ を $\boldsymbol{A}$ に関して最大化するためにフランクウルフ法を適用したとき，各反復における方向決定ステップおよび線探索ステップは閉形式で求められる．

方向決定ステップおよび線探索ステップの具体的な更新則と Theorem 5.1 の証明は [11] 参照．Theorem 5.1 により，mdos 型損失関数のみならず，mdos 型損失関数のモーロー包に対しても，フランクウルフ法の各ステップは閉じた形で表されることが判明した．

6. 実験

提案した学習アルゴリズムの数値実験の結果を報告する．4 個のデータセット，ALOI ($n = 10,339$, $d = 128$), Caltech101 ($n = 6,339$, $d = 256$), CUB ($n = 6,033$, $d = 4,096$), Flowers102 ($n = 2,040$, $d = 4,096$) を用いた．提案したフランクウルフ法が**勾配降下法 (GD)** と比べてどれほど高速に学習できるか比較した．GD 法には **Pegasos 法** [10] を用いた．Pegasos 法も劣線形収束が保証されている．提案したフランクウルフ法の各反復において線探索を行うものになっていた．線探索の代わりに $\gamma_t = 2/(t+1)$ としても劣線形収束の理論は保持される [4]．線探索をしないフランクウルフ法と区別するために，線探索を行う提案法を**線探索フランクウルフ法 (LSFW)**，線探索をしないフランクウルフ法を**標準フランクウルフ法 (Std FW)** と呼ぶことにし，Std FW 法とも比較した．

図 1 に収束のふるまいを示す．上段の縦軸は目的誤差 $P(\boldsymbol{w}^{(t)}) - P(\boldsymbol{w}^\star)$，下段は双対ギャップ $P(\boldsymbol{w}^{(t)}) - D(\boldsymbol{A}^{(t)})$ を表す．$\boldsymbol{w}^{(t)}$ は反復 t における主変数の値で，LSFW 法および Std FW 法では，$\boldsymbol{w}^{(t)} = \boldsymbol{w}(\boldsymbol{A}^{(t)})$ によって得た．いずれのデータセットでも，GD 法と比べて，LSFW 法ははるかに高速に最適解に収束した．Std FW 法は，Caltech101，CUB，Flowers102 においては，LSFW 法と大きな開きがあった．これは，理論的には線探索しなくても劣線形収束するが，実応用においては線探索が重要なことを示唆するものである．

謝辞 本研究は JSPS 科研費 19K04661 の助成を受けたものである．

参考文献

[1] Dejun Chu, Rui Lu, Jin Li, Xintong Yu, Changshui Zhang, and Qing Tao. Optimizing top-k multiclass SVM via semismooth newton algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 29(12):6264–6275, December 2018.

[2] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.*, 2:265–292, March 2002.

[3] P F Felzenszwalb, R B Girshick, D McAllester, and D Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, September 2010.

[4] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, March 1956. doi:10.1002/nav.3800030109.

[5] Sam Hare, Stuart Golodetz, Amir Saffari, Vibhav Vineet, Ming-Ming Cheng, Stephen L. Hicks, and Philip H.S. Torr. Struck: Structured output tracking with kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2096–2109, October 2016.

[6] Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning - ICML 05*. ACM Press, 2005.

[7] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, May 2009.

[8] Tsuyoshi Kato and Yoshihiro Hirohashi. Learning weighted top-k support vector machine. In Wee Sun Lee and Taiji Suzuki, editors, *Proceedings of The Eleventh Asian Conference on Machine Learning*, volume 101 of *Proceedings of Machine Learning Research*, pages 774–789, Nagoya, Japan, 17–19 Nov 2019. PMLR.

[9] Maksim Lapin, Matthias Hein, and Bernt Schiele. Top-k multiclass svm. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 325–333, Cambridge, MA, USA, 2015. MIT Press.

[10] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: primal estimated sub-gradient solver for SVM. *Math. Program.*, 127(1):3–30, 2011.

[11] Kenya Tajima, Yoshihiro Hirohashi, and Tsuyoshi Kato. Frank-Wolfe algorithm for learning SVM-type multi-category classifiers, Aug 2020. to appear on arXiv.

[12] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[13] Nicolas Usunier, David Buffoni, and Patrick Gallinari. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML09*. ACM Press, 2009.