LSHTC4のためのTTI文書分類システム

TTI Text Classification System for LSHTC4

佐々木 裕　マハマド・ゴラム・ソフラブ
Yutaka Sasaki　Mohammad Golam Sohrab
豊田工業大学
Toyota Technological Institute (TTI)

{yutaka.sasaki,sohrab}@toyota-ti.ac.jp

1　はじめに

2012 年に開催された Third PASCAL Large-Scale Hierarchical Text Classification (LSHTC3) Challenge [1] に続いて 2014 年 1 月に LSHTC4 チャレンジが開催された．LSHTC3 のトラック 1 は，Wikipedia カテゴリーへの文書分類を対象にしており，データサイズにより Medium サブタスクと Large サブタスクの 2 つのタスクが設定された．LSHTC4 のトラック 1 は，Large データを対象とするタスクのみが設定された．LSHTC3 のトラック 1 の Large サブタスクと LSHTC4 のトラック 1 は同一タスクである．

Wikipedia Medium データと Large データを表 1 にまとめた．Medium サブタスクは，階層化された 50,312 の Wikipedia カテゴリー (末端ノードの数は 36,504) に関連付けられた 456,866 の訓練用 Wikipedia 文書に基づいて，81,262 のテスト用 Wikipedia 文書を分類するという非常にチャレンジングなタスクである．文書には末端のカテゴリーのみが付与されており，カテゴリー体系は，循環のない合流を許す階層構造（すなわち DAG）になっている．我々は，LSTHC3 の Medium サブタスクに参加し，その結果は 17 システム中の上位に位置付けられるものであった [2]．

Large サブタスクは，階層化された 478,020 の Wikipedia カテゴリー (末端ノードの数は 325,055) に関連付けられた 2,365,436 の訓練用 Wikipedia 文書に基づいて，452,167 のテスト用 Wikipedia 文書を分類するというさらにチャレンジングなタスクである．また，カテゴリー体系も一般のグラフ構造であり，カテゴリの上位下位関係の循環も許されている．

Wikipedia Large データ（LSHTC4 トラック 1 の対象データ）は，そのデータおよび階層的カテゴリのサイズが巨大なため，LSHTC3 の Medium タスク向けに構築した我々の分類システムでは扱えなかった．具体的に問題となった点は下記の通りである．

- データと階層が巨大なため，末端カテゴリに割当られたデータを階層のトップまで伝播する処理に必要なメモリが使用したサーバのメモリ上限（96GB）を越え，処理できない．
- また，上記の処理はメモリが不足しなくても，推定によると実用的な時間内には終了しない．
- 学習アルゴリズムとして高速な SVM-perf [3] を用いていたが，約 200 万の訓練データに基づき，90 万個近いモデルを作成するには，十分な学習速度ではなかった．単純な推計では，学習に 16 日以上かかる計算になる．

LSHTC 4 向けに開発したシステムでは，上記の問題点を解決し，実用的な時間内に学習とテストが実行できることを確認した．

表 1: Wikipedia Medium データと Large データの比較

	Medium	Large
#training data	456,866	2,365,436
#distinct features	346,299	1,617,899
largest feature ID	2,085,164	2,085,166
max #features per document	1,349	5,054
min #features per document	2	2
average #features per document	48.05	43.53
max #documents per leaf category	11,400	378,768
min #documents per leaf category	1	1
average #documents per leaf category	23.09	23.73
max #categories per document	50	198
min #categories per document	1	1
average #categories per document	1.84	3.26
#test data	81,262	452,167
#distinct features	132,296	627,935
largest feature ID	2,085,164	2,085,164
max #features per document	903	2,361
min #features per document	2	2
average #features per document	47.62	43.80
#categories in the hierarchy	50,312	478,020
#leaf categories	36,504	325,055
#edges	65,333	863,261

2 TTI LSHTC 4 システム：学習フェーズ

2.1 ボトムアップ伝播

学習フェーズにおいてカテゴリー階層構造を利用するためには，訓練データ ID が階層構造に対応づけられていなければならない．訓練用データには末端のカテゴリーにしか付与されていないため，学習の前処理として，訓練データ ID を階層構造に従って末端ノードからルートに向ってボトムアップに伝播する必要がある．階層構造は複数の親ノードを許すため，複数の親ノードがある場合は，分岐しながらボトムアップに階層に訓練データ ID を割当る．この過程は単純であるが，本タスクのように階層構造が非常に大規模な場合は，実装レベルでメモリの消費量と処理時間のバランスに注意する必要がある．実際，Large データでは，カテゴリーサイズが Medium の十倍程度であり，訓練データサイズも数倍あるため，簡単にはデータを階層構造に実用時間内に伝播することができない．

そこで，LSHTC4 では，データ ID を末端からルートに向って伝播するのではなく，末端ノードに割当られているデータを記憶しておき，末端ノードの ID をルートに向って伝播する．この効率化によって 30GB 程度のメモリ使用量で全訓練データの階層への伝播が実現できた．

2.2 トップダウン学習

LSHTC 4 において，より高い分類性能を達成するために SVM を機械学習アルゴリズムとして採用した．具体的には，階層構造の各エッジに対して SVM 分類器を割り当て，総計 874,219 の SVM モデルを学習した．[1]

LSHTC3 用のシステムと同様に，学習はトップダウンに行う．対象のノードに伝播された末端ノード集合を対象に，各エッジについて，その下位（子）カテゴリーに伝播されている末端ノード集合に割当られているデータを正例とし，その他を負例として SVM モデルを学習し，エッジに関連付ける．

ここで，対象ノード配下の末端ノードに割当られているデータは全て一度展開しなければならな

[1] 複数のトップノードが存在するため，仮想的なルートからトップノードへのエッジの数だけモデル数が大きくなっている．

い．なぜなら，正例側の末端ノードに割当られているデータの中には，負例側の末端ノードに割当られているデータが含まれていることがあるからである．

学習の高速化のために，Pegasos (Primal Estimated sub-GrAdient SOlver for SVM) [4] を，Pegasos の共同考案者である豊田工大シカゴ校（TTIC）の協力をえて，C++により実装した．SVM-perf はデータをファイルで受け渡していたが，システム全体を C++で記述し，Pegasos の学習ルーチンを直接呼び出すことで，Pegasos の高速で高精度な学習能力とあいまって，システム全体の高性能化が実現できた．

Pegasos は SGD-SVM と同様に確率的降下法により SVM の目的関数の最適化

$$\min_{\boldsymbol{w}} \frac{\lambda}{2} \parallel \boldsymbol{w} \parallel + \frac{1}{N} \sum_{(\boldsymbol{x},y)\in \boldsymbol{D}} \max(0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle).$$

に対して，逐次的に最適な重みベクトルを求める．加えて，重みベクトルを半径 $1/\sqrt{\lambda}$ の L_2-球に写像する．

$\eta_t = \frac{1}{\lambda t}$ とする．ランダムに選ばれた $(\boldsymbol{x}_t, y_t)$ に対して, もし $y_t \boldsymbol{w}_t \cdot \boldsymbol{x}_t < 1$ であれば，下記のように重みベクトルを更新する．

$$\begin{aligned} \boldsymbol{w}_{t+\frac{1}{2}} &= (1 - \eta_t \lambda)\boldsymbol{w}_t + \eta_t y_t \boldsymbol{x}_t. \\ \boldsymbol{w}_{t+1} &= \min\left(1, \frac{1/\lambda}{\|\boldsymbol{w}_{t+\frac{1}{2}}\|}\right) \boldsymbol{w}_{t+\frac{1}{2}}. \end{aligned}$$

Pegasos のパラメータ λ は $\frac{1}{CN}$ である．また，繰り返し数は $\max(10000, 5N)$ とした．[2]サンプリングは正例と負例を交互にサンプリングする balanced stochastic サンプリングである．

3 TTI LSHTC 4 システム：分類フェーズ

テストデータは，LSHTC3 システムと同様に，階層構造に従って，ルートカテゴリーから末端のカテゴリに向かって，先に学習した SVM 分類器により振り分けられる．

[2]繰り返し回数は多いほど精度が良くなり，100N 回程度繰り返すことで，高い予測性能が得られことが経験的に分かっているが，実用的にはこの程度の繰り返し回数が限界である．

3.1 トップダウン分類と確信度

また，後段の枝刈りに備えて，階層的分類に合わせて分類の確信度も逐次的に計算していく．LSHTC 3 システムと同様に，正例と負例のバランスを修正するためのバイアス β を導入する．つまり，親ノード p から子ノード c への分類を決定する場合，$\mathrm{SVM}_{pc}(x) > \beta$ であれば，x は子ノードに分類される．$\mathrm{SVM}_{AB}(x) > \beta$ と $\mathrm{SVM}_{AC}(x) > \beta$ の両方が満たされた場合は，x は A と B の両方に分類される．経験に基づき $\beta = -0.5$ とした．

SVM(x) の値を，下記のシグモイド関数により [0,1] の実数値に正規化する．

$$\sigma_\alpha(x) = \frac{1}{1 + \exp(-\alpha x)}.$$

x が末端ノード n に至ると，確信度 $c_\alpha(x, n)$ は以下の式で計算される．

$$c_\alpha(x, n) = \prod_{(n_1,n_2)\in E} \sigma_\alpha(SVM_{n_1 n_2}(x)),$$

ここで，E はルートから n に至る経路で辿ってきたエッジの集合である．実装上は，分岐の時点でそのノードに至る確信度を計算しておき，下位の分岐の際に利用する．Medium データによる経験から $\alpha = 2.0$ とする．

3.2 大域的枝刈り

分類誤りのため各分岐のローカルな分類の時点で判断された分岐が，階層の下位において良くない分岐であったとわかることがある．これは，確信度に閾値を設け，大域的に枝刈りすることで実現できる．確信度に対する閾値 θ は，$\theta = 0.27$ とした．この確信度による枝刈りは性能向上に非常に効果的があり，LSHTC3 において TTI システムが上位にランクされた要因であった．

4 実験結果

4.1 実験の設定

Pegasos を用いて，874,219 のエッジ SVM 分類器を学習した．C パラメータは 0.5 とした． 96GB メモリ搭載の Xeon 3.0GHz サーバにより実験を行った．Large データは数値ベクトルに変換され

表 2: LSHTC4 Wikipedia Large タスクの結果

Name	Acc	EBF	LBMaF	LBMiF	HF
TTI	0.3185	0.3866	0.1920	0.3644	0.4295
anttip	0.3152	0.3681	0.1919	0.3038	0.4546
k-NN Baseline	0.2724	0.3471	0.1486	0.3015	0.4616

たデータセットが配布されているので，そのデータを利用した．データは，標準的なストップワードを除いた単語ユニグラムに基づく特徴ベクトルである．

LSHTC3 においてシステム比較に用いられた評価尺度は下記の 5 種類である，

- Accuracy (Acc):
  $1/\mid D \mid \sum_{i \in D} \mid Y_i \cap Z_i \mid /(\mid Y_i \cup Z_i \mid)$
  ここで，D はデータの集合，Y_i は正解ラベル集合，Z_1 は出力ラベル集合である．
- Example-based F1 measure (EBF):
  $1/\mid D \mid \sum_{i \in D} 2 \mid Y_i \cap Z_i \mid /(\mid Y_i \mid + \mid Z_i \mid)$
- Label-based Macro-average F1 measure (LBMaF): カテゴリ毎の F1 スコアを平均したマクロ F1 スコア
- Label-based Micro-average F1 measure (LBMiF): カテゴリを区別しない F1 スコア
- Hierarchical F1-measure (HF) [5]: 末端カテゴリの上位のカテゴリも考慮した EBF スコア．

4.2 評価結果

表 2 に LSHTC4 トラック 1 の公式評価の結果を示す．特に，高い Micro-F1 スコアを得ることができた．その理由については，現在検証中である．

2,365,436 訓練データによる階層構造中の 874,219 エッジに対する学習時間は 3,029 分（約 50 時間）であった．テストフェーズにおける 452,167 データの 325,055 カテゴリへの分類時間は 266 分（約 4.4 時間）であった．これらの時間は，ファイル入出力やオーバヘッドを含めたすべての処理時間を含む．このように，SVM を用いた大規模な階層的分類を現実的な処理時間内で実現することができた．

5 まとめと今後の課題

LSHTC 4 Wikipedia Large データを用いて大規模階層的文書分類実験を行い，参加 3 システム中で最上位にランクされる TTI システムを構築した．大規模階層に含まれる約９０万のエッジすべてに対応する SVM 分類器を構築した．学習を 50 時間，分類を約 5 時間で実現できた．

今後の課題として，学習アルゴリズムの改良による分類精度の向上と GPGPU 等による超並列処理による学習／分類処理の高速化があげられる．

謝辞

本研究は，科研費 25330271 の助成を受けた．

参考文献

[1] PASCAL LSHTC3 Challenge, http://lshtc.iit.demokritos.gr/.

[2] 佐々木裕, デイヴィー・ヴィッセンバッシャー: LSHTC3 データを対象にした大規模階層的文書分類, 2013 言語処理学会年次大会, 2013.

[3] T. Joachims: A Support Vector Method for Multivariate Performance Measures, *Proc. of the International Conference on Machine Learning (ICML-05)*, pp. 377–384, 2005.

[4] S. Shalev-Shwartz, Y. Singer, and N. Srebro: Pegasos: Primal estimated sub-gradient solver for SVM, *in Proc. of the 24th international conference on Machine learning*, 2007.

[5] S. Kiritchenko: Hierarchical Text Categorization, Its Application to Bio-informatics, *Ph.D. thesis*, University of Ottawa Ottawa, Ont., Canada, 2006.