機械学習テンプレートライブラリ

清水伸幸† 宮尾祐介‡
†東京大学情報基盤センター図書館電子化部門
shimizu@r.dl.itc.u-tokyo.ac.jp
‡東京大学大学院情報理工学系研究科コンピュータ科学専攻
yusuke@is.s.u-tokyo.ac.jp

1 はじめに

本稿では，様々な構造化機械学習手法を容易に利用することができる**機械学習テンプレートライブラリ**を紹介する．近年の自然言語処理では，多くの場面で機械学習を用いることが必要となってきている．一方で，言語の構造をより適切にモデル化した機械学習手法がさかんに研究されている [2]．これらの手法は様々な自然言語処理タスクにおいてその有用性が示されているが，より複雑なモデルになるにつれて実装は煩雑になり，また学習手法間の理論的・経験的関係も把握しづらくなってきている．そこで，自然言語処理へ機械学習を応用する研究や，より自然言語処理に適した機械学習手法の開発を容易にするために，機械学習ライブラリの設計・開発を行った．

自然言語処理における機械学習の研究では，ほぼ同じデータ構造で，損失関数や最適化アルゴリズムが異なる手法が多く提案されている．例えば，品詞タグ付けなどの系列ラベル付け問題に対しては，以下のような機械学習手法が提案されている．

- Conditional random fields (CRF) [8]
- Max-margin Markov network [14]
- Structured perceptron [4]

これらの手法は，共通のデータ構造 (マルコフ連鎖) を対象としている．一方，損失関数はそれぞれ異なり，そのため最適化アルゴリズムも異なっている．このように，共通のデータ構造に対して様々な学習手法が提案されているという状況は，より複雑なデータ構造が必要な構文解析においても同様である．

以上のような状況を鑑み，機械学習テンプレートライブラリは以下のような目的を持って設計されている．

- 同じデータ構造について，様々な損失関数や最適化アルゴリズムを容易に比較できる．
- 木構造や連鎖構造など，言語の構造を表現するのに適したデータ構造を様々な機械学習アルゴリズムで利用できる．
- データ構造に依存する計算 (デコード，期待値計算など) と，学習アルゴリズム (損失関数，最適化アルゴリズム) を切り離し，どちらを改良しても残りの部分を利用可能にする．
- 新たなデータ構造 (または学習器) を実装すれば，様々な学習器 (またはデータ構造) との組み合わせを容易に試すことができる．
- 既存の機械学習ソフトウェアと遜色ない時間・メモリ効率性を目指す．

2 背景

本研究では，以下のような式で識別関数を定義する機械学習モデルを対象とする．

$$\eta(y|x) = f(\mathbf{w} \cdot \Phi(x, y))$$

ここで，x, y は履歴とターゲット，Φ は素性関数，$\mathbf{w}$ はパラメタ（重み）ベクトル，f は任意の関数である．例えば，文書分類では x は文書で y はラベル，品詞タグ付けでは x は文で y は品詞列，係り受け解析では x は文で y は係り受け木となる．

現在自然言語処理で利用されている多くの機械学習手法は，上記のように定式化することができる．代表的なものでは以下のものがある．

パーセプトロン　自然言語処理では構造化されたパーセプトロンが良く用いられる．パーセプトロンは単純，高速でありながら競争力のある性能を発揮することで知られている．以下、アルゴリズムを簡潔に解説する．まず，パラメタベクトル $\mathbf{w}$ をゼロに初期化しておく．次に教師データ集合から，i 番目の履歴とターゲットを選び、これを x_i, y_i とする． y_i は x_i に対して与えられた正解となるターゲット構造である．パーセプトロンでは，この x_i に対して逐次，ターゲット構造を予測する．予測された構造を y' と呼び，予測するステップをデコードと呼ぶ．

$$y' := argmax_y(\mathbf{w} \cdot \Phi(x_i, y))$$

例えば，ターゲット構造が品詞列であるとすると，ありうる全ての品詞列を列挙し，与えられた重みベクトル $\mathbf{w}$ において，スコアが最大となる品詞列 y' を取り出す．こ

の y' が，現時点でモデルが最も最適であると予測する構造である．次に，パーセプトロンによる重みベクトルの更新について述べる．単純に予測された構造 y' と、正しいと知られる構造 y_i の素性の差分を取り，対応する重みについて，正しい構造から欠けている素性については+1, 間違った構造にのみ存在する素性であれば-1 する．

$$\mathbf{w} := \mathbf{w} + \Phi(x_i, y_i) - \Phi(x_i, y')$$

この二つのステップを繰り返すことによって学習が進む．省略したが，よく使われるバージョンでは性能向上のため，更新した重みベクトルの平均を取り，過学習を防ぐといった仕組みも組み込まれている [4].

MIRA　パーセプトロンによる重みベクトルの更新について，「予測された構造 y' と、正しいと知られる構造 y_i の素性の差分を取り，対応する重みについて，正しい構造から欠けている素性については+1, 間違った構造にのみ存在する素性であれば-1 する」と述べたが，MIRA は，この重みの更新を単に+1, -1 ではなく，更新された重みベクトルでちょうど正しい構造が予測できる数値にするよう定式化された学習法である [9].

ログ線形モデル (最大エントロピーモデル, ME)　ログ線形モデルは，教師データにおいて素性が現れる確率と，モデルが予測する素性の出現確率が同じになるよう，重みベクトルを最適化する学習手法である [8]．最適化の方法によって，擬似ニュートン法 [13], orthant-wise 擬似ニュートン法 [1] などが用いられる．

Max-margin モデル (サポートベクタマシン, SVM)　正しい構造 y_i のスコア $\mathbf{w} \cdot \Phi(x_i, y_i)$ と、間違った構造 y' のスコア $\mathbf{w} \cdot \Phi(x_i, y')$ の差が最大となるよう重みベクトルを最適化する学習手法である [14, 15, 2, 16].

一方，自然言語処理においては，言語の構造を表現するために古くから連鎖構造や木構造などのデータ構造，およびその上に定義される確率モデルや動的計画法などが研究されてきた．近年，これらのデータ構造に対して機械学習を適用した研究やソフトウェアが多数提案されている．代表的なものでは以下のものがある．

- Conditional random fields [13]
- Structured perceptron [4, 5]
- Feature forest model [11]
- Max-margin parsing [15]
- 最大スパニングツリーアルゴリズムを応用した依存構造解析 [10]
- 行列木定理を応用した依存構造解析 [7]

これらの機械学習手法で用いているデータ構造は，古くから用いられている連鎖構造や木構造と本質的に同じである．また，学習アルゴリズムも基本的にはビタビアルゴリズムや前向き・後向きアルゴリズムなどの既存のアルゴリズムを利用したものになっている．したがって，データ構造およびそれに付随するアルゴリズムと，機械学習の損失関数および最適化アルゴリズムを切り離すことで，様々な構造化機械学習手法を見通しよく実装することができる．

3　設計

機械学習テンプレートライブラリは，データ構造と学習器を組み合わせることで様々な機械学習手法を容易に利用できるようにすることを目指している．具体的には，C++のテンプレートによる抽象化を利用し，データ構造のクラスと学習器のテンプレートクラスを組み合わせることで，様々な機械学習手法を実現する．テンプレートによる抽象化により，データ構造と学習器の関係を見通しよく実装することができ，新たなデータ構造や学習器を実装するのも容易になる．また，コンパイラによる最適化が行われるため，効率性も期待できる．

例えば，マルコフ連鎖 CRF を利用する場合は，マルコフ連鎖を表すクラス Markov と，ログ線形モデルの学習器を表すテンプレートクラス LogLinear を用いて，以下のようなプログラムを記述する．

```
Markov markov;
LogLinear<Markov> crf(&markov);
while (!crf.isConverged()) {
  crf.iterate();
}
```

同様に，同じデータ構造でパーセプトロン (構造化パーセプトロン) を利用する場合は，以下のようなプログラムとなる．プログラムはほとんど同じで，LogLinear クラスが Perceptron クラスに置き換わっただけであることに注意されたい．

```
Markov markov;
Perceptron<Markov> sp(&markov);
while (!sp.isConverged()) {
  sp.iterate();
}
```

非常に抽象化されているにもかかわらず，C＋＋ではテンプレートがコンパイル時に展開されるため，インライン化など，プログラムの実行速度に対する最適化が容易であるという利点がある．図 1 に，ライブラリの全体像を示す．

学習器は，データ構造クラスに定義されているいくつかのメソッドを利用して必要な統計量を計算し，パラメタを更新する．表 1 に，主な学習器と必要なメソッドを示す．学習器が利用する主なメソッドは以下のとおりである．

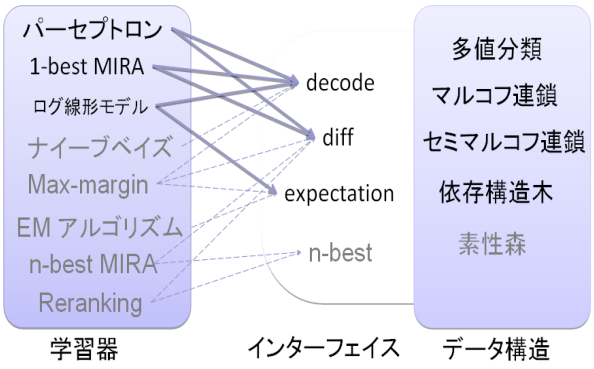

図 1: 機械学習テンプレートライブラリのデザイン

	デコード	差分	期待値	k-best
パーセプトロン	√	√	–	–
1-best MIRA	√	√	–	–
k-best MIRA	√	√	–	√
ログ線形モデル	√	–	√	–
Max-Margin	√	–	√	–

表 1: 学習器と必要なメソッド

デコード $\eta(y|x)$ を最大にする y を求める．パーセプトロンや MIRA の学習などで用いられる．ビタビアルゴリズムなどを用いる．

差分 二つのターゲット y_1, y_2 の差分を計算する．パーセプトロンや MIRA の学習で用いられる．

期待値 $\Phi(x, y)$ の $\eta(y|x)$ による重みつき平均 (期待値) を計算する．ログ線形モデルや max-margin モデルの学習で用いられる．前向き・後向きアルゴリズムなどを用いる．

例えば，テンプレートクラス Perceptron は以下のように実装されている．

```
template <class DataStruct>
class Perceptron : public Learner {
public:
  typedef typename DataStruct::Target Target;
  typedef typename DataStruct::History History;
  typedef typename DataStruct::Feature Feature;
  typedef typename DataStruct::FeatVec FeatVec;
  typedef typename DataStruct::Corpus Corpus;

  DataStruct* data_struct;
  Corpus* corpus;

public:
  Perceptron(DataStruct* ds, Corpus* cp)
    : data_struct(ds), corpus(cp) {}

  ...

  void iterate() {
    for (typename Corpus::const_iterator
         iter = corpus->begin();
         iter != corpus->end(); ++iter) {
      Target predict;
      data_struct->decode(iter->history,
                          weight, predict));
      Target d;
      data_struct->diff(iter->target, predict, d);
      FeatVec f =
        data_struct->extractFeature(d, iter->history);
      FeatVec f1 =
        data_struct->extractFeature(d1, iter->history);
      update(f, f1, weight);
    }
  }
};
```

前半部分はデータ構造に関する型定義である．メソッド iterate は，パーセプトロンのパラメタ更新アルゴリズムを実装している部分である．データ構造クラスに定義されているメソッド decode (デコード), diff (差分), および extractFeature (素性抽出) を利用し，それらの結果を利用してパラメタを更新している．この実装は特定のデータ構造に依存していないことに注意されたい．したがって，decode, diff, extractFeature が定義できる任意のデータ構造で，このパーセプトロン学習器を利用することができる．

データ構造クラスでは，構造から情報を取り出すためのメソッドを定義する．例えば，もっとも単純なデータ構造である多クラス分類のクラス Classifier は，以下のように定義されている．

```
class Classifier {
public:
  typedef int Target;
  typedef vector<FeatureID> History;

  ...

  bool decode(History history,
              WeightMap weight,
              Target& target) {
    double scores[num_classes];
    for(int t = 0; t < num_classes; ++t) {
      FeatVec features;
      extractFeature(t, history, features);
      scores[t] = features.product(weight);
    }
    double* max_it
      = max_element(scores, scores + num_classes);
    target = max_it - scores;
    return true;
  }
};
```

メソッド decode では，各ターゲットについて素性を抽出 (extractFeature) し，それと重みベクトルとの内積をとることでスコアを計算し，スコアが最大になるターゲットを返す．マルコフ連鎖などのより複雑なデータ構造の場合は，ビタビアルゴリズムなどを実装することになる．

4 現状と今後の予定

機械学習テンプレートライブラリは，ソースコードと，典型的なデータ構造・学習器についてバイナリパッケージを公開する予定である．いくつかの構造については実装が進んでおり，依存構造木では CoNLL2007 Shared Task の英語コーパスにおいて，パーセプトロン学習（10 iteration）が 4 時間ほどで終了することを実証し，ライブラリの有用性を確認した．現在，今年度中の公開を目指して以下のデータ構造および学習器の実装を行っている．

データ構造

- 多クラス分類 [3]
- マルコフ連鎖，セミマルコフ連鎖 [12]
- 依存構造木 [7]
- 素性森 [11]

学習器

- パーセプトロン
- 1-best MIRA
- 2 ノルムのログ線形モデル (擬似ニュートン法)
- 1(および 2) ノルムのログ線形モデル (orthant-wise 擬似ニュートン法)

今後は，max-margin アルゴリズムや k-best MIRA を実装することを予定している．さらに，デコードや期待値計算を用いた他のアルゴリズム，例えば EM アルゴリズムなどの実装を検討している．また，現在は素性ベクトルと重みベクトルの内積で定義されるモデルのみを対象としているが，将来的には素性ベクトル同士の内積を用いるモデルも対象とすることで，クラスタリング，近傍分類法，距離関数学習などの手法や，exponentiated gradient [2, 6] などの最適化手法，およびカーネル法なども実装できると考えている．

参考文献

[1] G. Andrew and J. Gao. Scalable training of l1-regularized log-linear models. In *Proc. of the 24th International Conference on Machine Learning*, 2007.

[2] P. Bartlett, M. Collins, B. Taskar, and D. McAllester. Exponentiated gradient algorithms for large-margin structured classification,. In *Advances in Neural Information Processing Systems 16*, 2004.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. of Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[5] M. Collins and B. Roark. Incremental parsing with the perceptron algorithm. In *Proc. of the 42rd Annual Meeting of the ACL*, 2004.

[6] A. Globerson, T. Koo, X. Carreras, and M. Collins. Exponentiated gradient algorithms for log-linear structured prediction. In *Proc. of the 24th International Conference on Machine Learning*, 2007.

[7] T. Koo, A. Globerson, X. Carreras, and Michael Collins. Structured predictcion models via the matrix-tree theorem. In *Proc. of Empirical Methods in Natural Language Processing (EMNLP)*, 2007.

[8] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the 18th International Conference on Machine Learning*, 2001.

[9] R. McDonald, K. Crammer, and F. Pereira. Online large-margin training of dependency parsers. In *Proc. of the 43rd Annual Meeting of the ACL*, 2005.

[10] R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of the Joint Conf. on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005.

[11] Y. Miyao and J. Tsujii. Maximum entropy estimation for feature forests. In *Proc. of the Human Language Technology Conf. (HLT)*, 2002.

[12] S. Sarawagi and W. Cohen. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 16*, 2004.

[13] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proc. of the Human Language Technology Conf. (HLT)*, 2003.

[14] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems 15*, 2003.

[15] B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. Max-margin parsing. In *Proc. of Empirical Methods in Natural Language Processing (EMNLP)*, 2004.

[16] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. In *Jornal of Machine Learning Research 6*, 2005.