Efficient graphical models for sequence segmentation

Sunita Sarawagi
IIT Bombay
sunita@iitb.ac.in

Abstract

Segmentation of sequences is an important modeling primitive with several applications. Training and inference of segmentation models involves dynamic programming computations that in the worst case can be cubic in the length of a sequence. In contrast, typical sequence labeling models require linear time.

We propose an alternative graphical model for efficient sharing of potentials across overlapping segments. We then design message passing algorithms that are significantly faster than the original cubic algorithms. When segmentation models are posed as large margin structured classification tasks, our algorithm directly impact the computation of marginals for exponentiated gradient training algorithms [1] and modes for cutting plane algorithms [7].

1 Introduction

Segmentation of sequences is an important primitive in several tasks including, Named entity recognition, information extraction, part of speech tagging, shallow parsing, pitch accent prediction, and, protein/gene finding. Given an input sequence $x_1 \dots x_n$, the goal in segmentation is to find all segments that belong to a set of entities Y. In the basic form, the set of segments are non-overlapping and cover a continuous subset of tokens. In the general case, the segments are allowed to be both overlapping and dis-continuous [4].

Traditionally, applications such as Named Entity Recognition (NER) or Information Extraction (IE) that are naturally segmentation problems were approximated as sequential labeling problems where each token x_i in the token is assigned a label y_i from a set of labels Y, or a label from an expanded set B-C-E-U. However, when extraction is not a stand-alone problem but has to interoperate with other tasks like match of extracted entities with existing database of structured entities [3], it is more accurate and convenient to directly model the extraction of segments. Also, in extraction tasks that involve segments that are discontinuous as in [2, 4], the discontinuity gives rise to segments that cannot be handled via traditional chain models.

2 Graphical models for segmentation

Given an input sequence $\boldsymbol{x} = x_1 \dots x_n$, a segmentation $\boldsymbol{s}$ of $\boldsymbol{x}$ consists of a sequence of variable length segments $\boldsymbol{s} = \langle s_1, \dots, s_p \rangle$ where each segment $s_j = \langle t_j, u_j, y_j \rangle$ consists

of a *start position* t_j, an *end position* u_j, and a *label* $y_j \in Y$. Conceptually, a segment means that the tag y_j is given to all x_i's between $i = t_j$ and $i = u_j$, inclusive. We consider the case of contiguous, disjoint segments which implies that $t_{j+1} = u_j + 1$. The label of a segment depends on the x properties around the segment position and the label of the segment prior to it. Often, for limiting computation requirements, the maximum length of a segment is restricted to a pre-defined constant L.

We express segmentations as graphical models by expanding the label set so that for each label y in Y we create four labels y_B, y_E, y_C, y_U indicating that the label is assigned to a position marking respectively the beginning of a segment, end of a segment, continuation of a segment, and, a single word segment. The graphical model for segmentation consists of two kinds of cliques:

- segment cliques c_{ij} formed by fully connecting all contiguous nodes from y_i to y_j — these are meant to denote a segment from i to j and thus all nodes are required to have the base label y appropriately specialized to y_B, y_E, y_C, y_U.
- transition cliques between adjacent nodes $i, i+1$ to capture the dependence of the label of a segment starting at $i+1$ on the label of the segment ending at position i.

This yields a graph with cliques of size L where computing exact marginals and modes could be intractable. For example, if we do not restrict the maximum segment length, we get a complete graph over all n nodes. However, the restricted choice of labels that can be assigned to a clique and the special form of the cliques, enables us to design a message passing algorithm that runs in $O(nL^2)$ time as follows:

Let $\boldsymbol{\theta}_{i'i}$ denote potentials over segment cliques spanning from node i' to i and $\boldsymbol{\theta}_i$ denote the potentials over transition edges from $i-1$ to i. The forward message from a node i is computed as

$$\alpha_i(y) = \begin{cases} \sum_{\max(i-L,1) \le i' \le i} \sum_{y' \in \mathcal{Y}} \alpha_{i'-1}(y') \theta'_i(y', y) \theta_{i'i}(y) & \text{if } i > 0 \\ 1 & \text{if } i = 0. \end{cases} \quad (1)$$

where L = maximum segment length and we use the shortcut $\theta_{i'i}(y)$ to denote $\theta_{i'i}(y_B, y_C, \ldots y_C, y_E)$. The $O(nL^2)$ complexity comes about due to potentials over length L cliques.

Similarly, the backward messages $\boldsymbol{\beta}_i$ from node i to $i-1$ can be computed and these can be used to find the marginal for a segment clique $i'i$ as:

$$\mu_{i'i}(y) = \frac{\alpha_i(y) \beta_{i+1}(y)}{Z(\boldsymbol{x})}$$

where $Z(\boldsymbol{x}) = \sum_y \alpha_n(y)$.

The forward messages need to be passed only along transition cliques and this enables the algorithm to run in time that is polynomial in the size of the graph. Other examples of graphical models where restrictions on labels assignments has been exploited for tractable message passing are Associative Markov Networks (AMN) [6]. The segmentation model is different from an AMN in two ways: the clique structure is more regular, and, the potential structure is richer because of dissociative potentials in transition cliques. The message passing algorithm is optimal for arbitrary size of Y whereas AMNs yield optimal answers only for binary label sets.

3 Efficient graphical models

When $L \approx n$, the segmentation algorithm becomes cubic in n and this is a problem for practical settings. Also, the above model is restricted to full segment potentials. Ideally,

we would like to allow potentials to be defined over subset of the full segment cliques, so that several segments can share the same potential. Thus, we allow potentials of the form $\phi_{i,i+2}(y_B, y_C, y_C)$ so that these can be shared over all segments that start at position i and end after position $i+2$. A special case of this is when cliques are of size no greater than 2 and this gives rise to the popular NER Markov model based on B-C-E-U labels. We generalize these models to where potentials can be defined over arbitrary subsequences of $\boldsymbol{x}$ provided they are associated with segment consistent labels. This gives rise to three additional kinds of cliques:

- $\boldsymbol{\theta'}_{i':>i}$ which denotes potentials shared over all segments starting at i' but ending anywhere after i.
- $\boldsymbol{\theta'}_{<i':i}$ which denotes potentials shared over all segments ending at i but starting anywhere before i'.
- $\boldsymbol{\theta'}_{<i':>i}$ which denotes potentials shared over all segments ending after i and starting before i'.

We designed a message passing algorithm that can run in time that is proportional to $O(nm^2)$ where m is the largest subsequence over which potentials are defined and not necessarily the largest length of a sequence. Thus, for sequence labeling tasks where $m = 2$, this will reduce to the standard $O(n)$ forward-backward algorithm, even through it can potentially output segments much larger than 2. We generalize this to the case of an arbitrary set of cliques. We show how to directly compute marginals over the smaller potentials instead of summing over segment-level potentials which can be very large.

The main challenge in designing an efficient algorithm is that potentials could overlap in arbitrary ways and we cannot afford to pass messages only along transition edges. A key insight we exploit is that for segments longer than m, a set of inter-segment messages can be combined over a decomposable set of potentials. We show how this is done for forward messages. Equation 1 in matrix notation and with no L restriction becomes:

$$\boldsymbol{\alpha}_i = \sum_{i' \leq i} (\boldsymbol{\alpha}_{i'-1} \boldsymbol{\theta}_{i'}) * \boldsymbol{\theta}_{i':i}$$

where the symbol "*" denotes element-wise multiplication of vectors of the same size. In the rest of the paper, we will drop the use "*" to reduce clutter and assume it to be implicitly present when two vectors of the same length abut. Let $\boldsymbol{\theta}_{i':i}$ denote the product of all potentials applicable to segment $i'i$, this will include all potentials of the form $\boldsymbol{\theta'}_{uv}$ where either $u = i'$ or $u = <j$ for all $j > i'$ and $v = i$ or $v = >j$ for all $j < i$. Similarly, let $\boldsymbol{\theta}_{i':>i}$ denote product of all potentials applicable to segments where the start boundary is *equal to* i' and end boundary anywhere after i. $\boldsymbol{\theta}_{i':i} = \boldsymbol{\theta}_{i':>i-1} \boldsymbol{\theta}_{i':(>i-1 \rightarrow i)}$ where $\boldsymbol{\theta}_{i':(>i-1 \rightarrow i)}$ includes all potentials of the form $\boldsymbol{\theta'}_{r:i}$ where $r = i'$ or $r = <j$ for $j > i'$ $\boldsymbol{\theta}_{i':(>i-1 \rightarrow i)}$ is the same for all $i' \leq i - m$ since any feature with end boundary tied to position i cannot have its start boundary at any value less than $i - m$.

Let $\boldsymbol{\alpha\theta}_i = \boldsymbol{\alpha}_{i-1} \boldsymbol{\theta}_i$

$$\begin{aligned} \boldsymbol{\alpha}_i &= \sum_{i' \leq i-m} \boldsymbol{\alpha\theta}_{i'} \boldsymbol{\theta}_{i':>i-1} \boldsymbol{\theta}_{i-m:(>i-1 \rightarrow i)} + \sum_{i-m<i' \leq i} \boldsymbol{\alpha\theta}_{i'} \boldsymbol{\theta}_{i':i} \\ &= \boldsymbol{\alpha}_{\leq i-m:>i-1} \boldsymbol{\theta}_{i-m:(>i-1 \rightarrow i)} + \sum_{i-m<i' \leq i} \boldsymbol{\alpha\theta}_{i'} \boldsymbol{\theta}_{i':i} \end{aligned} \quad (2)$$

where $\boldsymbol{\alpha}_{\leq i-m:>i-1}$ denote the sum over all possible segmentations where the last segment's start boundary is $\leq i - m$ and the end boundary is open-ended at $i - 1$. We can compute this term recursively as follows:

$$\boldsymbol{\alpha}_{\leq i-m+1:>i} = \begin{cases} \boldsymbol{\alpha}_{\leq i-m:>i-1} \boldsymbol{\theta}_{i-m:(>i-1 \rightarrow >i)} + \boldsymbol{\alpha\theta}_{i-m+1} \boldsymbol{\theta}_{i-m+1:>i} & \textit{if } i \geq m \\ \boldsymbol{\theta}_{0:>m-1} & \textit{if } i = m-1 \\ 0 & \textit{otherwise} \end{cases}$$

$\boldsymbol{\theta}_{i':i}$ is computed incrementally from $\boldsymbol{\theta}_{i'+1:i}$ through data structures that can efficiently find all potentials that become invalid and valid with the change of the start offset by one.

Thus, by maintaining an additional set of n forward terms denoting $\boldsymbol{\alpha}_{\leq i-m+1:>i}$ we are able to compute α_i by summing over only m instead of $i-1$ terms.

The computation of the most likely segmentation involves two dynamic programming equations similar to the two equations for $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_{\leq i-m+1:>i}$ above. The computation of traditional beta terms can be done using a similar backward run.

The two forward and two backward messages can be combined to compute the marginals for various potentials. We show an example for computing marginals $\boldsymbol{\mu}_{<s:e}$ for potentials of the form $\boldsymbol{\theta'}_{<s:e}$ without explicitly summing over all segments with start position less than s.

$$\boldsymbol{\mu}_{<s:e} = \frac{\sum_{i' \leq s} \boldsymbol{\alpha}\boldsymbol{\theta}_{i'}\boldsymbol{\theta}_{i':e}\boldsymbol{\beta}_e}{Z(\boldsymbol{x})}$$

We can simplify this computation so as to not require summing over s terms as follows:

$$\boldsymbol{\mu}_{<s:e} = \frac{1}{Z(\boldsymbol{x})} \begin{cases} \boldsymbol{\mu}_{<(s-1):e} + \boldsymbol{\alpha}\boldsymbol{\theta}_{s-1}\boldsymbol{\theta}_{s-1:e}\boldsymbol{\beta}_e & \text{if } e-s < m, s > 0 \\ \boldsymbol{\beta}_e\boldsymbol{\alpha}_{<s:>e-1}\boldsymbol{\theta}_{s:(>e-1 \rightarrow e)} & \text{if } e-s = m, s > 0 \ (see\ Eq:\ 2) \\ \boldsymbol{\theta}_{0:e}\boldsymbol{\beta}_e & \text{if } s = 0 \\ \text{not needed} & \text{if } e-s > m \end{cases}$$

In the above equation, the $\boldsymbol{\mu}_{s:e}$ values are only computed for the case where $e-s \leq m$. This explains why for the second equation where $e-s=m$, we could not recurse on $\boldsymbol{\mu}_{<(s-1):e}$. Overall, this enables us to compute all marginals in $O(nm^2)$ time without imposing any hard limit on the length of the segment. Also, for shared potentials, the computation is done only once instead of repeatedly for each overlapping segment it is associated with.

Concluding remarks Our experiments on the impact of the improved message passing algorithm for various training algorithms, including the exponentiated gradient algorithm for max-margin classification and LBFGS algorithm for Semi-CRFs [5] show that we obtain a factor of three to ten reduction in running time. This makes the running time for segmentation comparable to the running time for sequence labeling while allowing the flexibility to exploit a more powerful and flexible set of feature potentials. Further, our message passing algorithm could lend insights into simplifying other graphical models.

References

[1] P. L. Bartlett, M. Collins, B. Taskar, and D. McAllester. Exponentiated gradient algorithms for large-margin structured classification. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 113–120. MIT Press, Cambridge, MA, 2005.

[2] J. Bockhorst and M. Craven. Markov networks for detecting overlapping elements in sequence data. In *Advances in Neural Information Processing Systems (NIPS-17), MIT Press.*, 2005.

[3] I. Mansuri and S. Sarawagi. A system for integrating unstructured data into relational databases. In *Proc. of the 22nd IEEE Int'l Conference on Data Engineering (ICDE)*, 2006.

[4] R. McDonald, K. Crammer, and F. Pereira. Flexible text segmentation with structured multilabel classification. In *Human Language Technology Conference Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005.

[5] S. Sarawagi and W. W. Cohen. Semi-markov conditional random fields for information extraction. In *NIPs*, 2004.

[6] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks. In *Twenty First International Conference on Machine Learning (ICML04), Banff, Canada.*, 2004.

[7] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6(Sep):1453–1484, 2005.