Training Log Linear Models using Smoothed Hamming Loss

Olga Russakovsky, in collaboration with Samuel Gross, Chuong Do, Serafim Batzoglou

December 15, 2006

1 Introduction

In a paper that is to appear in NIPS this year [1], we proposed a new objective function for training Conditional Random Fields (CRFs). When using the traditional log-likelihood training, the training objective function is fundamentally different from the testing objective (the accuracy of the resulting parse). We wanted to develop a method of training CRFs where the training and testing objectives would be more similar. We proposed using a differentiable approximation of the Hamming loss as the training objective. In order to make the approach feasible, we derived a dynamic programming algorithm to efficiently calculate the gradient of this new function. We then used BFGS to train the CRF with respect to this objective, and test its performance on sequence labeling problems. Using the trained parameters, we parsed new, unlabeled sequences by choosing, for each position independently, the label with the highest posterior probability. The results were very promising, showing that especially on difficult problems, where at each position the posterior probabilities of the different labels were very similar, the maximum accuracy function significantly outperformed the log-likelihood objective. We also tested our idea on a large-scale problem of gene prediction, and showed that this new objective function is much more resistant to noisy labels in the data.

For this project, we wanted to see if this idea can generalize to other applications as well, such as RNA folding or NLP part-of-speech parsing. However, both of those problems can't be solved with a linear sequence labeling model, and so we are using a more general tree-structured model instead.

2 Algorithmic idea

2.1 Notation

Let $\mathcal{X}^L$ denote an input space of all possible input sequences, and let $\mathcal{Y}^{LxL}$ denote the space of all possible output labels (a label can be assigned to a span (i, j) of the sequence).

2.2 Conditional Log-Linear Models

We define the conditional probability of a labeling $\mathbf{y}$ given an input sequence $\mathbf{x}$ as

$$P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \frac{\exp\left(\sum_{n \in T_{1,L,\mathbf{y}}} \mathbf{w}^T \mathbf{f}(,\mathbf{x})\right)}{\sum_{\mathbf{y}' \in \mathcal{Y}^{LxL}} \exp\left(\sum_{n \in T_{1,L,\mathbf{y}'}} \mathbf{w}^T \mathbf{f}(n,\mathbf{x})\right)} = \frac{\exp\left(\mathbf{w}^T \mathbf{F}_{1,L}(\mathbf{x},\mathbf{y})\right)}{\mathcal{Z}(\mathbf{x})}, \tag{1}$$

where we define $n \in T_{a,b,\mathbf{y}}$ as the nodes in the parse tree corresponding to the labeling $\mathbf{y}$ covering the span a, b of the sequence $\mathbf{x}$, the *summed feature mapping*, $\mathbf{F}_{a,b}(\mathbf{x},\mathbf{y}) = \sum_{n \in T_{a,b,\mathbf{y}}} \mathbf{f}(n,\mathbf{x})$ and where the $\mathcal{Z}(\mathbf{x})$ is the *partition function* used to normalize the distribution.

2.3 Relationship to Stochastic Context Free Grammars

CLLMs can be used in the framework of Stochastic Context Free Grammars (SCFGs) SCFGs include production rules r_1, r_2, ... r_n and probabilities p_1, p_2, ..., p_n corresponding to each rule. Consider sequences generated according to this grammar. Then for a particular parse $\mathbf{y}$ corresponding to some sequence $\mathbf{x}$,

$$P(\mathbf{y}) = \prod_i p_i^{n_i(y)} \tag{2}$$

where $n_i(\mathbf{y})$ is the number of times r_i occurs in the parse $\mathbf{y}$.

Now consider a CLLM model incorporating this SCFG. The features of each node in the parse tree thus depend on the production rule used. Furthermore, consider setting

$$\mathbf{F}_{1,L}(\mathbf{x}, \mathbf{y}) = \mathbf{n} \tag{3}$$

$$\mathbf{w} = log(\mathbf{p}) \tag{4}$$

In other words, the feature count of the i^{th} feature in the CLLM is n_i, and the weight of that feature is the $log(p_i)$. Then it can be shown that the two expressions for the probability of the parse (corresponding to the CLLM and the SCFG) are equivalent.

Finally, from now on assume that the grammar we're working with is unambiguous, so there is a unique translation from the given labeling $\mathbf{y}$ to a parse tree.

2.4 Maximum accuracy

Now, instead of training this model to maximize the log likelihood of the data, as is conventional, we instead want to train it to maximize the total accuracy of the labeling (which is the objective we care about in the first place). In particular, let $\mathcal{D} = \{\mathbf{x}^{(t)}, \mathbf{y}^{(t)}\}_{t=1}^m$ be the set of training examples. We wish to maximize

$$\mathcal{A}(\mathbf{w} : \mathcal{D}) = \sum_{t=1}^{m} \sum_{n \in T_{1,L,\mathbf{y^{(t)}}}} 1[P(y_n = y_n^{(t)} | \mathbf{x}^{(t)}) > P(y_n = a | \mathbf{x}^{(t)}) \text{ for all } a \neq y_n^{(t)}] \tag{5}$$

In other words, for each node in the parse tree corresponding to an input labeling $y^{(t)}$, we add 1 to the objective if our CLLM correctly predicts the label for the span of that node. However, we can't do the maximization directly on a step function. Instead we let Q be a differentiable function arbitrarily similar to the step function $I(x) = 1[x > 0]$, and instead maximize

$$\tilde{\mathcal{A}}(\mathbf{w} : \mathcal{D}) = \sum_{t=1}^{m} \sum_{n \in T_{1,L,\mathbf{y^{(t)}}}} \mathcal{Q}[P(y_n = y_n^{(t)} | \mathbf{x}^{(t)}) > P(y_n = a | \mathbf{x}^{(t)}) \text{ for all } a \neq y_n^{(t)}] \tag{6}$$

$$= \sum_{t=1}^{m} \sum_{n \in T_{1,L,\mathbf{y^{(t)}}}} \mathcal{Q}[\sum_{\mathbf{y} \in \mathcal{Y}^{LxL} : y_n = y_n^{(t)}} \frac{\exp(\mathbf{w}^T \mathbf{F}(\mathbf{x}^{(t)}, \mathbf{y}))}{\mathcal{Z}(\mathbf{x})} - max_{a \neq y_n^{(t)}} \sum_{\mathbf{y} \in \mathcal{Y}^{LxL} : y_n = a} \frac{\exp(\mathbf{w}^T \mathbf{F}(\mathbf{x}^{(t)}, \mathbf{y}))}{\mathcal{Z}(\mathbf{x})}] \tag{7}$$

Now we use the chain rule to compute the gradient of $\mathcal{A}$ with respect to $\mathbf{w}$. For the sake of space, we omit the discussion of the dynamic programming algorithms used to efficiently compute the gradient of this function; however, the equations are fairly straight-forward generalizations of the method presented in [1], except that now they are similar to the inside/outside algorithms of Context Free Grammars instead of the forward/backward algorithms. For the function Q, we use

$$Q(x; \lambda) = \frac{1}{1 + \exp(-\lambda x)} \tag{8}$$

As $\lambda \to \infty$, $Q(x; \lambda) \to 1\{x > 0\}$, so $\tilde{\mathcal{A}}(\mathbf{w} : \mathcal{D}) \to \mathcal{A}(\mathbf{w} : \mathcal{D})$. However, the approximation $\tilde{\mathcal{A}}(\mathbf{w} : \mathcal{D})$ is smooth for any $\lambda > 0$.

2.5 Training

Within this framework, we're currently using the L-BFGS algorithm to optimize the parameters of the model. In the future, we're considering experimenting with different algorithms; however, this algorithm worked well for the linear sequence-labeling case, so we're continuing to use it for now. Note that one drawback of the proposed objective function is its non-convexity. In an attempt to avoid falling into a local minimum while training, we first optimize the parameters using the convex likelihood function, and then attempt to improve on them using the maximum accuracy algorithm. This is the same approach that was used for the gene prediction experiments.

3 RNA folding

We are planning to apply this model to various different applications. The one we're focusing on currently is RNA secondary structure prediction. In particular, we would like to extend the CONTRAfold[2] project to encorporate this new training algorithm.

3.1 Biology background

An RNA is a single-stranded linear molecule of nucleic acids that conveys genetic information. The nucleic acid alphabet consists of just 4 characters: A, C, G and U. The RNA molecule can fold on itself as bonds between these nucleotides form. The resulting secondary structure of RNA is very important for its function, as it determines the types of interactions it can have with other molecules around it. Therefore, being able to accurately predict RNA secondary structure is a necessary step to being able to understand the function of the molecule. For the purposes of this project, we're interested in correctly predicting

1. the pairings within the molecule (nucleotide i pairs with and bonds to nucleotide j)
2. the unpaired nucleotides

The tradeoff between the value of these two occurrences will be controlled by a parameter α, because predicting a nucleotide pairing correctly is much more important to determining the overall structure of the molecule than predicting that a certain nucleotide is unpaired; however, it should be strictly worse to predict an incorrect pairing than to not make a prediction.

3.2 Context free grammars

The problem of RNA secondary structure prediction fits very nicely into the framework of context free grammars. In particular, consider the following grammar proposed by [3] and referred to as the *G6 grammar*:

$$S \rightarrow LS|L \tag{9}$$

$$L \rightarrow aF\hat{a}|a \tag{10}$$

$$F \rightarrow aF\hat{a}|LS \tag{11}$$

where a, $\hat{a}$ can correspond to any of the 4 nucleotides. This grammar is the first step in testing out the algorithm, since it is much simpler than a real grammar which would be used by an RNA folder, yet it is unambiguous and has been shown to work quite well nevertheless [3].

3.3 Limitations

One of the limitations of this approach is that it can't predict pseudoknots or any other structure that is not completely nested. However, no method is currently able to predict these structures with a reasonable time complexity, and, furthermore, these structures are fairly rare so high accuracy can be achieved even by programs that don't take them into account.

4 Results

The results are still pending, however, there are a couple of experiments that we're in the process of running. For these tests, we're using RNA sequences from the Rfam database, using the same selection criteria as in [2]. Thus only the structures that have been generated by biological experiments and not those predicted by a computational program are going to be used.

In the immediate future, we plan on accomplishing the following tasks:

1. Comparing the performance of maximum accuracy and likelihood parsing using the simple G6 grammar
2. Encoding the more complicated grammar used in CONTRAfold, and comparing the performance directly to the published results
3. Comparing the speed of our more generalized model with CONTRAfold

Preliminary results should be available by the end of next week.

5 Future Directions

Furthermore, some other options that we're going to explore are

1. Evaluating the tradeoff of λ – the potential gain in accuracy from using a more exact approximation versus the longer training time and even the potential loss of accuracy as the function becomes harder to optimize. One option to consider is increasing λ slowly as the training progresses.
2. Evaluating the performace of other objective functions besides approximate accuracy and likelihood – e.g. max-margin methods
3. Generalizing from RNA secondary structure prediction to a more complicated model of NLP part of speech tagging.

Overall, this is really just the beginning of this work. The goal is to conclude the project and produce publishable results within the next two quarters.

6 References

1. Gross, S.S, Russakovsky, O, Do, C.B, Batzoglou, S. Training Conditional Random Fields for Maximum Labelwise Accuracy. To appear in *Neural Information Processing Systems (NIPS) 2006*.
2. Do, C.B, Woods, D.A, and Batzoglou, S. (2006). CONTRAfold: RNA Secondary Structure Prediction without Energy-Based Models. *Bioinformatics*, 22(14):e90-e98.
3. Dowell, R.D. and Eddy, S. R. Evaluation of Several Lightweight Stochastic Context-Free Grammars for RNA Secondary Structure Prediction. *BMC Bioinformatics*, 5:71, 2004.