Optimization and Evaluation of Probabilistic-Logic Sequence Models

Henning Christiansen and Ole Torp Lassen

Research group PLIS: Programming, Logic and Intelligent Systems
Department of Communication, Business and Information Technologies
Roskilde University, P.O.Box 260, DK-4000 Roskilde, Denmark
E-mail: {henning, otl}@ruc.dk

Abstract. Analysis of biological sequence data demands more and more sophisticated and fine-grained models, but these in turn introduce hard computational problems. A class of probabilistic-logic models is considered, which increases the expressive power from HMM's and SCFG's regular and context-free languages to, in principle, Turing complete languages. In general, such models are computationally far too complex for direct use, so optimization by pruning and approximation is needed. The first steps are made towards a methodology for optimizing the models by approximations using auxiliary models for preprocessing or splitting them into submodels. An evaluation method for approximating models is suggested based on automatic generation of samples. These models and evaluation processes are illustrated in the PRISM system developed by other authors.

1 Introduction

Our main goal with the present work is to promote new strong models defined in declarative languages (such as extensions to Prolog) for biological sequence analysis. This provides the familiar advantages of declarative programming; it is known that standard models such as HMM and SCFG are embedded in natural ways, they can be extended and combined in very flexible ways, and long-distance context-sensitive dependencies can be modelled using logical variables and arbitrary auxiliary data-structures and predicates.

The resulting expressive power up to in principle Turing complete languages, evidently has serious consequences for computational complexity. The present work suggests the first steps towards a methodology for optimizing such models by approximations and assessing the quality of the approximating models. We focus currently on models that can be expressed in the PRISM system [11, 12], but our general framework is independent of the particular formalism used.

2 Probabilistic-Logic Models and Basic Assumptions

We consider probabilistic models that describe relationships between *sequence data* and *annotations* that capture the information or the "semantics" embedded

in the data, and we intend that such models can be used for computing the best annotation for a given sequence, where "best" is defined as the one with highest probability.

The models defined below may attach *auxiliary annotations* to a sequence, which are projections of the full annotation; they are redundant, but are utilized later when we introduce preprocessors which may gradually produce larger and larger parts of the full annotation.

Definition 1. *An annotation domain of degree* $n, n \geq 0$ *is a sequence of sets* $[\mathcal{A}_0, \mathcal{A}_2, \ldots, \mathcal{A}_n]$ *equipped with projection functions* $\pi : \mathcal{A}_{i-1} \rightarrow \mathcal{A}_i$, $i = 1 \ldots, n$.

Example 1. For given sets $Street, City, Country$, we may have $\mathcal{A}_0 = Street \times City \times Country$, $\mathcal{A}_1 = City \times Country$, $\mathcal{A}_2 = Country$, and π clips of the first element of the tuple to which it is applied.

Models range over sets of ground first-order atoms built over a finite signature which includes list construction for representation of sequences and operators that form annotations. The underline symbol is used to denote an "anonymous variable" as in Prolog, i.e., each occurrence is a unique variable.

Definition 2. *A* probabilistic-logic model $m = \langle L_m, P_m \rangle$ *consists of a* logical part L_m *which is a set of ground atoms of a predicate* $m(A_o, A_1, \ldots, A_n, S)$ *and a* probabilistic part *which is a probability distribution over* L_m. *The arguments of predicate* m *are characterized as follows.*

- $[A_0, \ldots, A_n]$ *belongs to an annotation domain specific for* m $[\mathcal{A}_0, \mathcal{A}_2, \ldots, \mathcal{A}_n]$, *and* $\pi(A_{i-1}) = A_i$, $i = 1 \ldots, n$; *this property is referred to as a* functional dependency.
- A_0 *is called the* (full) annotation, $A_1, \ldots, A_n$ *the* auxiliary annotations,
- S *represents the sequence.*

The distribution is extended to any nonground m *atom,* M, *as follows.*

$$P_m(M) =_{\text{def}} \sum_{M' \text{ is a ground instance in } L_m \text{ of } M} P_m(M')$$

For given model m, we use the following shorthands; S is a sequence and $A_0, A_1, \ldots, A_n$ are (auxiliary) annotations.

$$P_m(S) = P_m(m(\_, \_, \ldots, \_, S))$$
$$P_m(A_0|S) = P_m(m(A_0, \_, \ldots, \_, S)|m(\_, \_, \ldots, \_, S))$$

We refer to the task of computing $\text{argmax}_\alpha(P_m^{annot(S)}(\alpha))$ as *prediction* and as *predicting the best annotation* for the given S; recall that $\text{argmax}_\alpha(\cdots)$ is a value of α that provides a maximal value of the inner expression.

The following basic assumptions are made about the interpretation of the probabilities provided by such models.

Assumption 1 *A model may be distinguished as* canonical*, and its quality is not questioned; it is assumed to represent the best knowledge about the domain (e.g., a projection of nature) available among domain specialists (e.g., biologists).*

Assumption 2 *The probabilities given by a* canonical *model are inherently correlated to the quality of a given sample: an annotation with a relatively high probability for a given sequence scores high with accepted measurements, e.g., precision and recall for genes found by a model when compared with laboratory experiments; and vice versa. Furthermore, a sequence with high probability shows high similarity with those sequences that can be observed in nature.*

We notice some immediate consequences of these assumptions, when P is a canonical model and S a sequence with a relatively high probability.

- Two annotations A and A' with relatively high probabilities $P(A|S)$ and $P(A'|S)$ are similar, i.e., when comparing their detailed structure, there will be a considerable overlap in the occurrences of phenomena that are counted in accepted measurements (e.g., particular genes counted for recall and precision).
- An annotation A' produced by an implemented procedure is acceptable (i.e., scores high in accepted measurements), whenever the magnitude of $P(A'|S)$ is comparable to $P(A|S)$ where A is the best annotation of S given by P, P being the probability distribution of a canonical model.

These observations allow us to consider approximating models that prune not only low probability candidate annotations, but also some of the high probability ones.

3 Optimization by Preprocessing and Combined Submodels

An interesting canonical model is most likely computationally too complex for predictions over sequences of realistic size and we consider in this section approaches to produce approximating models in terms of preprocessing to address the issues of complexity. We also show how their qualities may be evaluated. By preprocessing, we mean producing auxiliary annotations by specialized analyses, expected to run significantly faster that the canonical model itself.

3.1 Preprocessing

Consider a canonical model m^c with annotation domain $[\mathcal{A}_0, \ldots, \mathcal{A}_n]$ and projections π. By a *preprocessor system* for m^c, we refer to a set of $n+1$ models $m_i^a = \langle L_i^a, P_i^a \rangle$, $i = 0, \ldots, n$ such that $m_i^a(A_i, \ldots, A_n, S) \in L_i$ implies that $A_k \in \mathcal{A}_k$ and $\pi(A_{k+1}) = A_k$, $k = i-1, \ldots, n$. A preprocessor system gives rise

to the composition of an approximating distribution defined as follows.

$$
\begin{aligned}
P^a(A_0,\ldots,A_n,S) =_{\text{def}} & \; P_0^a(A_0,\ldots,A_n,S) \\
& \text{where} \\
& A_n = \operatorname*{argmax}_{A_n}(P_n^a(A_n,S)) \\
& A_{n-1} = \operatorname*{argmax}_{A_{n-1}}(P_n^a(A_{n-1},A_n,S)) \\
& \quad \vdots \\
& A_1 = \operatorname*{argmax}_{A_1}(P_{i-1}^a(A_1,\ldots,A_n,S))
\end{aligned}
$$

Notice that we use argmax to fix a particular value gradually for each annotation $A_n, A_{n-1}, \ldots$ except for the topmost A_0 where a degree of freedom is kept open, as to define a distribution. However, we may expect that P^a is used for the prediction of a best A_0 for given sequence S.

There is one big difficulty in comparing the distributions P^c and P^a in that they do not in general describe the same annotations for the same S. So our Assumption 2 above is essential in order to consider approximating models by preprocessing of any use. This means that the annotations predicted by a good P^a must be among those that score high in P^c, but we cannot insist on the reverse property.

An especially interesting instance of this general framework case is a preprocessor that chops the given sequence into smaller pieces, and then applies specialized versions of a canonical interpreter to each sequence; this is an effective application of divide-and-conquer which may have significant influence on time complexity. Consider a canonical model $m^c(A,S)$ where the annotation A is of the form $[t_1 : A_1, \ldots, t_k : A_k]$ where each t_i indicates a specific subsequence *type* and A_i an annotation of subsequence S_i, where $S = S_1 \bullet \cdots \bullet S_k$; "$\bullet$" denotes concatenation. We assume that m^c is implemented in terms of detailed models m_{ti}^c for the different types.

For the optimization of m^c, we may introduce an auxiliary model m^{chop} producing a unique chopping into subsequences of claimed type and define an approximation model as follows.

$$
\begin{aligned}
P^a([t_1 : A_1,\ldots,t_k : A_k],S) =_{\text{def}} & \; p_1 \times \ldots \times p_k \\
& \text{where} \\
& [t_1 : S_1,\ldots,t_k : S_k] = \\
& \quad \operatorname*{argmax}_{[t_1:S_1,\ldots,t_k:S_k]} P^{chop}([t_1 : S_1,\ldots,t_k : S_k],S) \\
& p_i = P_{ti}^c(A_i,S_i), i = 1,\ldots,k
\end{aligned}
$$

Notice that argmax for P^a can be obtained by evaluating an argmax value for each $P_{t_i}^c$ separately.

3.2 Estimating the Quality of an Approximating Model

In order to evaluate the quality of the approximating model respective to the canonical model, we would ideally compare their respective predicted annotations; but in particular the complexity of the canonical model prevents us from doing so.

However, sampling provides us with samples $\langle A, S\rangle$ that we can analyze with the approximating model and compare the result. Ignoring for the moment auxiliary annotations, we consider canonical and approximating models $m^c(A, S)$ and $m^a(A, S)$, and suggest the following process to collect a set $\mathbf{R}$ of observed ratios between probabilities for the annotation guessed by sampling and the one found by approximation for the same sequence.

1. $\mathbf{R} = \emptyset$
2. generate a sample $m^c(A^c, S)$,
3. let $A^a = \text{argmax}_{A^a}(P^a(A^a, S))$
4. insert $P^c(A^a, S)/P^c(A^c, S)$ into $\mathbf{R}$
5. goto 2, unless the designated time is exhausted

Notice at this point that probabilistic models often tend to assign unnaturally high probabilities to short sequences, and if this is known to be the case for $m^c(A, S)$, we may in this process discard any sequence of length below a certain minimum.

We cannot present evaluation criteria that reflect a deep statistical analysis, and we doubt that this is possible without detailed assumptions about a particular class of models. At present, we suggest a subjective analysis of the set of ratios involving the following rules of thumb.

- A substantial segment of ratios greater that 1 may indicate that there is a class of sequences, each of which has several good annotations with about the same probability, and that the approximation is likely able to find one of them in such cases.
  (Obviously some observed ratios greater than 1 may be caused by m^c suggesting a very bad annotation, so such a *single* observation does not say much about m^a.)
- A substantial segment of ratios below but close to 1 may indicate that there is a class of sequences, each of which has only few good annotations and that the approximation is able to find them.
- A substantial segment of ratios far below 1 indicates that there is a class of sequences for which the approximation is very inaccurate.

4 Case Study: A Simple Genefinder in PRISM

We sketch here a setup with a canonical model based on a SCFG, and an approximating model involving an auxiliary model that provides a unique splitting into subsequences according to the principles outlined in section 3.1.

4.1 A Canonical Model

The model is based on a SCFG that describes a structure for genomic sequences consisting of subsequences of genes and noncoding regions. Noncoding regions are seen as unstructured sequences of random letters.

A gene is indicated by a start and a stop codon (one of $\{\langle a,t,g\rangle, \langle g,t,g\rangle, \langle t,t,g\rangle\}$ and of $\{\langle t,a,a\rangle, \langle t,g,a\rangle, \langle t,a,g\rangle\}$, respectively), and their lengths must be a multiple of 3 as to fit with a codon structure. Within a gene, the grammar indicates possible two-dimensional structures of so-called hairpins. So for example, the subsequence $agata \ldots tatct$ may form a hairpin, where the outermost 5+5 letters form a stem and those indicated by the dots form a loop at the top; for simplicity we do not allow recursive hairpin structures.

The grammar rules are straightforward and omitted; the hairpin structures do not necessarily follow the codon structure so the grammar rules need to keep track of substring lengths modulo 3. A given hairpin is as long as possible, i.e., attractor pairs are not allowed at the positions next to the two ends of the stem.

For a given sequence, this grammar may assign many different parse trees due to the ambiguity caused by accidental triplets that are identical to start- and stop codons, and there are likely alternative foldings into hairpins. Probabilities in the model were set manually so as to keep sequence lengths sufficiently small for our experiment.

4.2 An Approximative Model

As described in section 3.1, we provide an approximating model by an auxiliary model that chops a sequence into subsequences classified as genes and noncoding; it ignores any structure inside a gene, except that genes must be multiples of 3 and wrapped in proper start/stop codons. Subsequently these substrings are analyzed with the well defined subgrammars of the canonical one for the two subsequence types.

As described above, when the combined model is executed for prediction, first the best chopping is found and fixed, and then the complex grammar rules find best subtrees for each subsequence in isolation.

4.3 Evaluation by Sampling

We executed the procedure described in section 3.2 for generating samples and finding the ratio of the canonical probabilities of the two annotations (here: parse trees) given by sampling and by the approximating model. Due to performance problems, we added a timeout for the parsing of the subsequences. We ended up considering 18 samples of lengths between 20 and 150; shorter sample strings were discarded.

It appears for one third of the samples, that the separation into subsequences defined by the sampling procedure and by the chopper model are identical (or very close to identical), represented by ratios above 0.99. In the remaining cases, we observed ratios above 0.9. However, due to the small number of samples and

their short lengths, we cannot at present give any clear conclusions about the quality of the chopper or the entire approximative model. With more substantial test data we would expect fewer ratios of 1 and a few > 1.

5 Related Work

Bayesian networks, HMMs, and SCFGs are traditional methods for sequence analysis that can be seen as instances of probabilistic-logic models; while their flexibility for modeling and formal expressive power is far below the models we are aiming at (PRISM and similar), there exists a plethora of efficient algorithms and implemented systems; see [5] for background and overview. These provide a catalogue of possible preprocessors to be used within our approach

More general and powerful formalisms have been suggested as extensions to logic programs or equally expressive formalisms within the last 15 years, we may mention PRISM [11, 12] that we have exemplified, Stochastic Logic Programs [10], Stochastic functional Programs [7], and Relational Bayesian Networks [6]. While such models have been used for systems biology, e.g., [3, 1], the application to analysis of sequence data seems to be sparse, which we may tentatively attribute to prejudices concerning efficiency in both time and space.

We may refer to [4] as a precursor of the present work, where similar ideas are applied for a comparative test of three different genefinder programs [2, 8, 9].

6 Conclusion and Future Work

We advocate the use of probabilistic-logic models based on logic programs (or similarly expressive languages) as the basis for analysis of biological sequence data; due to flexibility and generality, such models are candidates for providing better and more detailed finds than the currently most used methods, however, this is still to be proved.

We intend that such models should be developed without consideration about performance in order to document in a formal way and as faithfully as possible, the available knowledge about the phenomena being modeled in what we called a canonical model.

Using preprocessors, e.g., based on existing and efficiently implemented technologies, as a way to reach realistic execution times, we intend to get the best of both worlds, flexibility and sophistication of the probabilistic-logic models combined with feasible execution times.

Finally, we recommended heuristics for the evaluation of implemented and approximating versions based on sampling and probability measurements, which are basically the only computational usages that can be made of such canonical models. Although we described this method for implementations based on probabilistic-logic technology, the evaluation method can be applied for any implementation. Consider the example of section 4; here we can rebuild the approximating model using software specialized for SCFGs for both the chopping

and detailed analysis of substrings. We intend to extend the methodology with a more firm statistically basis for evaluation of the measurements produced by the sampling principle.

Acknowledgement: This work is supported by the project "Logic-statistic modelling and analysis of biological sequence data" funded by the NABIIT program under the Danish Strategic Research Council, and the CONTROL project, funded by Danish Natural Science Research Council.

References

1. Marenglen Biba, Stefano Ferilli, Nicola Di Mauro, and Teresa Maria Altomare Basile. A hybrid symbolic-statistical approach to modeling metabolic networks. In Bruno Apolloni, Robert J. Howlett, and Lakhmi C. Jain, editors, *KES (1)*, volume 4692 of *Lecture Notes in Computer Science*, pages 132–139. Springer, 2007.
2. Chris Burge and Samuel Karlin. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268:78–94, 1997.
3. Jianzhong Chen, Stephen Muggleton, and Jose Santos. Abductive stochastic logic programs for metabolic network inhibition learning. In Paolo Frasconi, Kristian Kersting, and Koji Tsuda, editors, *MLG*, 2007.
4. Henning Christiansen and Christina Mackeprang Dahmcke. A machine learning approach to test data generation: A case study in evaluation of gene finders. In Petra Perner, editor, *MLDM*, volume 4571 of *Lecture Notes in Computer Science*, pages 742–755. Springer, 2007.
5. Richard Durbin, Sean Eddy, Anders Krogh, and Graeme Mitchison. *Biological Sequence Analysis*. Cambridge University Press, 1998.
6. Manfred Jaeger. Relational bayesian networks. In Dan Geiger and Prakash P. Shenoy, editors, *UAI*, pages 266–273. Morgan Kaufmann, 1997.
7. Daphne Koller, David A. McAllester, and Avi Pfeffer. Effective bayesian inference for stochastic programs. In *AAAI/IAAI*, pages 740–747, 1997.
8. Anders Krogh. Using database matches with for HMMGene for automated gene detection in Drosophila. *Genome Research*, 10(4):523–528, 2000.
9. A. Lukashin and M. Borodovsky. Genemark.hmm: new solutions for gene finding. *Nucleic Acids Research*, 26(4):1107–1115, 1998.
10. Stephen Muggleton. Learning from positive data. In Stephen Muggleton, editor, *Inductive Logic Programming Workshop*, volume 1314 of *Lecture Notes in Computer Science*, pages 358–376. Springer, 1996.
11. Taisuke Sato and Yoshitaka Kameya. Parameter learning of logic programs for symbolic-statistical modeling. *J. Artif. Intell. Res. (JAIR)*, 15:391–454, 2001.
12. Taisuke Sato and Yoshitaka Kameya. Statistical abduction with tabulation. In Antonis C. Kakas and Fariba Sadri, editors, *Computational Logic: Logic Programming and Beyond*, volume 2408 of *Lecture Notes in Computer Science*, pages 567–587. Springer, 2002.