Power Law Unbounded Markov Prediction

Jan Gasthaus, Yee Whye Teh
Gatsby Computational Neuroscience Unit
University College London
London, WC1N 3AR, UK
{j.gasthaus, ywteh}@gatsby.ucl.ac.uk

Frank Wood
Department of Statistics
Columbia University
New York, NY 10027, USA
fwood@stat.columbia.edu

1 Introduction

We consider compression of sequences using a predictive model that incrementally estimates a distribution over what symbol comes next from the preceding sequence of symbols. As our predictive model we use the sequence memoizer (SM) [1], a nonparametric Bayesian model for sequences of unbounded complexity. This model is combined with an entropy coder, for instance the arithmetic coder of [2], to yield a method for compressing sequence data.

At the algorithmic level, the proposed method is somewhat similar to the unbounded context length variants of the well known PPM and CTW algorithms [3, 4]. At a conceptual level, however, our approach is quite different: We take a Bayesian approach, treating the distributions over next symbols as latent variables on which we place a hierarchical nonparametric prior, and predicting the next symbol by averaging over the posterior distribution (conditioned on the sequence observed thus far). The prediction problem thus essentially becomes an incremental inference problem, where the posterior distribution is incrementally updated after observing each symbol.

2 Model

Our compressor is based on an underlying probabilistic model referred to as the *sequence memoizer* (SM) [1]. This is a hierarchical Bayesian nonparametric model composed of Pitman-Yor processes originally conceived of as a model for languages (sequences of words). The model describes the conditional probability of each symbol $s \in \Sigma$ following each context $\mathbf{u} \in \Sigma^*$ using a latent variable $G_{\mathbf{u}}(s)$. Collecting the variables into a vector, $G_{\mathbf{u}} = [G_{\mathbf{u}}(s)]_{s\in\Sigma}$ is a probability vector (non-negative entries summing to one), and the full (infinite) set of latent variables in the model is $\mathcal{G} = \{G_{\mathbf{u}}\}_{\mathbf{u}\in\Sigma^*}$. The joint probability of $\mathbf{x}$ and $\mathcal{G}$ is simply:

$$P(\mathbf{x}, \mathcal{G}) = P(\mathcal{G}) \prod_{i=0}^{|\mathbf{x}|-1} G_{\mathbf{x}_{1:i}}(\mathbf{x}_{i+1}) \tag{1}$$

where the rightmost term is the probability of each symbol conditioned on the sequence thus far, and $P(\mathcal{G})$ is the prior over the variables. Taking a Bayesian approach and marginalizing out $\mathcal{G}$, we get a distribution $P(\mathbf{x})$ with which we can compress $\mathbf{x}$.

Succinctly, we can describe the prior $P(\mathcal{G})$ as follows:

$$G_\varepsilon \mid d_0, H \quad \sim \quad \mathcal{PY}(d_0, H) \tag{2a}$$

$$G_{\mathbf{u}} \mid d_{|\mathbf{u}|}, G_{\sigma(\mathbf{u})} \quad \sim \quad \mathcal{PY}(d_{|\mathbf{u}|}, G_{\sigma(\mathbf{u})}) \qquad \forall \mathbf{u} \in \Sigma^+ \tag{2b}$$

where H is a base vector (in the following assumed to be uniform) and $d_{|\mathbf{u}|}$ are the discount parameters. The structure of the model is an unbounded-depth tree, with each node indexed by a context $\mathbf{u} \in \Sigma^*$, labelled by the probability vector $G_{\mathbf{u}}$, and with parent given by $\sigma(\mathbf{u})$, where $\sigma(\mathbf{u})$ denotes the longest proper suffix of $\mathbf{u}$.

The structure encodes the prior knowledge that the predictive distributions over the subsequent symbols in different contexts are similar to each other, with contexts sharing longer suffixes being more similar to each other. In other words, the later symbols in a context are more important in predicting the subsequent symbol. Also, by virtue of using PYPs, the SM encodes an assumption that sequences of symbols have power-law properties.

To make computations in the SM tractable, it is essential to reduce the size of the model. Firstly, we can marginalize out all $G_{\mathbf{u}}$ not associated with the data in $\mathbf{x}$. This reduces the size of the tree from infinite to quadratic in $|\mathbf{x}|$. Secondly, we can marginalize out all non-branching nodes in the resulting tree, which further reduces the size to linear in $|\mathbf{x}|$. The second reduction can be performed analytically due to a theorem of [5], which in our situation simply states: if $G_1|G_0 \sim \mathcal{PY}(d_1, G_0)$ and $G_2|G_1 \sim \mathcal{PY}(d_2, G_1)$ then marginally $G_2|G_0 \sim \mathcal{PY}(d_1 d_2, G_0)$ is a Pitman-Yor process as well with modified discount parameters. Further details of this reduction can be found in [1]. Note that the algorithm in [1] requires the entire observation sequence to be known to construct the context tree efficiently and one of the contributions of this work is the forward incremental algorithm for constructing the context tree, which is essential for text compression since the decoder does not have access to the entire sequence until it has finished decoding.

3 Prediction, Inference and Estimation

Exact Bayesian computations in the SM model, such as finding the predictive distribution $\mathrm{P}(s|\mathbf{x}_{1:i})$, are intractable. We described an incremental algorithm for estimating $\mathrm{P}(s|\mathbf{x}_{1:i})$ based on viewing the Pitman-Yor distributed random vectors $G_{\mathbf{u}}$ in terms of the Chinese restaurant process (CRP), which can be described as follows: Initialize two sets of counts $\{c_s, t_s\}_{s\in\Sigma}$ to 0, and consider the following generative process: Draw $y_1 \sim H$ and set $c_{y_1} = t_{y_1} = 1$; for $n = 2, 3, \ldots$ and each $s \in \Sigma$, with probability $\frac{c_s - d t_s}{c_\cdot}$ set $y_n = s$ and increment c_s, and with probability $\frac{d t_\cdot}{c_\cdot} H(s)$ set $y_n = s$ and increment *both* c_s and t_s. The process generates a random sequence of symbols $y_1, y_2, \ldots \in \Sigma$. These symbols are dependent through the updates on the counts (e.g. if $y_1 = s$ we will more likely see $y_2 = s$ as well). Coming back to the PYP, since G is a probability vector, we can treat it as a multinomial distribution over Σ. Consider a sequence of iid draws $y_1, y_2, \ldots \sim G$. The randomness in G induces dependencies among the y_n's once G is marginalized out, and the resulting distribution over the sequence $y_1, y_2, \ldots$ is precisely captured by the Chinese restaurant process.

In this view, the random vectors $G_{\mathbf{u}}$ are not explicitly represented; instead, the marginal distribution of draws from $G_{\mathbf{u}}$ is captured by the counts $\{c_{\mathbf{u}s}, t_{\mathbf{u}s}\}_{s\in\Sigma}$ of the CRP. Because the $G_{\mathbf{u}}$'s are hierarchically coupled in the SM (2), the CRPs are coupled as well, leading to a joint generative process for the counts $\mathcal{S}_{\mathbf{x}_{1:i}} = \{c_{\mathbf{u}s}, t_{\mathbf{u}s}\}_{\mathbf{u}\in\Pi_{\mathbf{x}_{1:i}}, s\in\Sigma}$ in all contexts on the tree. Given these sets of counts for all contexts $\mathbf{u} \in \Pi_{\mathbf{x}_{1:i}}$ (which we will in the following jointly refer to as the *state* of the model $\mathcal{S}_{\mathbf{x}_{1:i}}$), the predictive probability for a symbol s following context $\mathbf{u}$ is given by

$$\mathrm{P}(s|\mathbf{u}, \mathcal{S}_{\mathbf{x}_{1:i}}) = \begin{cases} \frac{c_{\mathbf{u}s\cdot} - d_{\mathbf{u}} t_{\mathbf{u}s}}{c_{\mathbf{u}\cdot\cdot}} + \frac{d_{\mathbf{u}} t_{\mathbf{u}\cdot}}{c_{\mathbf{u}\cdot\cdot}} \mathrm{P}(s|\pi(\mathbf{u}), \mathcal{S}_{\mathbf{x}_{1:i}}) & \text{if } c_{\mathbf{u}\cdot\cdot} \neq 0; \\ \mathrm{P}(s|\pi(\mathbf{u}), \mathcal{S}_{\mathbf{x}_{1:i}}) & \text{otherwise,} \end{cases} \tag{3}$$

where $c_{\mathbf{u}\cdot\cdot} = \sum_{s\in\Sigma} c_{\mathbf{u}s\cdot}$, $t_{\mathbf{u}\cdot} = \sum_{s\in\Sigma} t_{\mathbf{u}s}$, and $\pi(\mathbf{u})$ is the parent of $\mathbf{u}$ in the context tree. This is simply a recursive application of the predictive probabilities in the CRPs associated with the contexts on the path from $\mathbf{x}_{1:i}$ to the root. The quantity we are actually interested in, $\mathrm{P}(s|\mathbf{x}_{1:i})$, can then be approximated by averaging (3) over posterior samples of the CRP counts $\mathcal{S}_{\mathbf{x}_{1:i}}$ conditioned on $\mathbf{x}_{1:i}$.

The task thus becomes obtaining samples from the posterior over the CRP counts conditioned on the observations $\mathbf{x}_{1:i}$. In our previous work [6, 1] these samples were obtained using Markov chain Monte Carlo methods, which is too computationally intensive for the sequential setting considered here. Here we take a different approach: at each iteration i we simply update the counts associated with the new observation of $\mathbf{x}_{i+1}$ in the CRP associated with context $\mathbf{x}_{1:i}$, and do not resample all the other counts. In the sequential Monte Carlo literature, this is simply a particle filter with only one particle; we call this variant 1PF.

To obtain a better approximation to the posterior, the particle filter can be run with multiple particles. Surprisingly, this results in a rather negligible improvement. There are two reasons for this. Firstly, as [6] noted, the posterior is unimodal, so it is easy for the particle filter to obtain a sample close to

	1PF	UKN	PPM*	PPMZ	CTW
Average bits/character	**1.89**	1.91	2.09	1.93	1.99

Table 1: Compression performance in terms of average log-loss (average bits per character under optimal entropy encoding) for the Calgary corpus.

the mode and this single sample is likely to perform well. Secondly, much of the predictive ability of the SM comes from the hierarchical sharing of counts which is already present in a single sample.

We also consider an alternative, non-probabilistic count update approximation: Instead of maintaining the counts $t_{\mathbf{us}}$, we constrain them to be 1 whenever $c_{\mathbf{us}} > 0$ and 0 otherwise. As noted by [6], this approximation yields predictive probabilities equal to the ones that would be obtained using interpolated Kneser-Ney, a state-of-the-art smoothing algorithm for language models; we call this variant UKN (for unbounded-depth Kneser-Ney).

4 Results

We evaluated the compression performance on various types of input sequences by making next symbol predictions on the Calgary corpus – a well known compression benchmark corpus consisting of 14 files of different types and varying lengths. The measure used for comparing the different algorithms is the *average log-loss* $\ell(\mathbf{x}_{1:N}) = -\frac{1}{N}\sum_{i=1}^{N}\log_2 p(x_i|\mathbf{x}_{1:i-1})$ which corresponds to the average number of bits per symbol required to encode the sequence using an optimal code. For all our experiments we treat the input files as sequences of bytes, i.e. with a 256 alphabet size.

A summary ot the results is shown in Table 1. For comparison, we also show the results of two PPM variants and one CTW variant in the final three columns. PPM* was the first PPM variant to use unbounded-length context, and the results for PPM-Z are (to our knowledge) among the best published results for a PPM variant on the Calgary corpus.

In addition to the experiments on the Calgary corpus, compression performance was also evaluated on the 100MB excerpt of an XML text dump of the English version of Wikipedia used in the Large Text Compression Benchmark and the Hutter Prize compression challange. On the Wikipedia excerpt, the UKN algorithm (without context mixing) achieved a log-loss of 1.66 bits/symbol amounting to a compressed file size of 20.80 MB. While this is significantly worse than 16.23 MB achieved by the currently best PAQ-based compressors (but on par with heavily hand-tuned PPM variants), it demonstrates that the described approach can scale to sequences of this length.

Finally, we explored the performance of the algorithm when using a larger alphabet. In particular, we used an alphabet of 16-bit characters to compress Big-5 encoded Chinese text. On a representative text file, the Chinese Union version the the bible, we achieved a log-loss of 4.35 bits per Chinese character, which is significantly better than the results reported by [7] (5.44 bits).

References

[1] F. Wood, C. Archambeau, J. Gasthaus, L. James, and Y. W. Teh. A stochastic memoizer for sequence data. In *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

[2] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.

[3] J. G. Cleary and W. J. Teahan. Unbounded length contexts for PPM. *The Computer Journal*, 40:67–75, 1997.

[4] F. M. J. Willems. The context-tree weighting method: Extensions. *IEEE Transactions on Information Theory*, 44(2):792–798, 1998.

[5] J. Pitman. Coalescents with multiple collisions. *Annals of Probability*, 27:1870–1902, 1999.

[6] Y. W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the Association for Computational Linguistics*, pages 985–992, 2006.

[7] Peilang Wu and W. J. Teahan. A new PPM variant for Chinese text compression. *Natural Language Engineering*, 14(3):417–430, 2007.