Sparse Adaptive Dirichlet-Multinomial-like Processes

Marcus Hutter
(presented by Tor Lattimore)

Canberra, ACT, 0200, Australia
http://www.hutter1.net/

2013

Problem Setup

- Data: Short sequence over large alphabet from unknown source.
- Regime: Base alphabet $\mathcal{X}$ larger than sequences length n.
- Problem: Estimation, Modelling, Prediction, *Compression*.
- Online alg: Predict next symbol x_{t+1} given only past symbols $x_{1:t}$.
- I.i.d: Assume unknown i.i.d. sampling distribution. Data often not i.i.d. but subsequence with given context is (closer to) i.i.d.
- Applications: machine learning, information theory, data compression, language modelling, document analysis.
- Example: Typical documents comprise a small fraction of the available 100 000+ English words,
  and words have different length/complexity/frequency.

The Dirichlet-Multinomial Distribution

= generalized Laplace rule = Carnap's induction inference scheme
= Polya urn scheme = Chinese restaurant process

$$\mathrm{DirM}(x_{n+1}=i|x_{1:n}) = \frac{n_i+\alpha_i}{n+\alpha_+}$$

- n_i = number of times $i \in \mathcal{X}$ appeared in $x_{1:n} \equiv (x_1,...,x_n)$.
- α_i = parameter = fictitious prior counts of i.
- $\alpha_+ := \sum_{i\in\mathcal{X}} \alpha_i$ = total mass = precision = concentration.
  Theoretically motivated choices for α_i (all equal by symmetry):

Dirichlet	Laplace	KT&others	Perks	Haldane	Hutter
$\alpha_i = \frac{\alpha_+}{\lvert\mathcal{X}\rvert}$	1	$\frac{1}{2}$	$\frac{1}{\lvert\mathcal{X}\rvert}$	0	$\frac{m}{2\lvert\mathcal{X}\rvert \ln\frac{n+1}{m}}$

- They are all problematic for large base alphabet $\mathcal{X}$.
- Existing solutions: empirically optimize or sample or average $\boldsymbol{\alpha}$.
- New solution (last column): Analytically optimize exact redundancy.
  m is the number of different symbols that appear in $x_{1:n}$.

Main Contribution

- Introduce an estimator S closely related to DirM
  but easier to analyze and slightly superior.
- Reserve escape probability to symbols not seen so far.
- Derive optimal adaptive escape parameter $\beta \widehat{=} \alpha_+$
  based on data-dependent redundancy,
  rather than expected or worst-case bounds.

The resulting estimator:
(i) is simple, (ii) online, (iii) fast,
(iv) performs well for all m, small, middle and large,
(v) is independent of the base alphabet size,
(vi) non-occurring symbols induce no redundancy,
(vii) the constant sequence has constant redundancy,
($viii$) symbols that appear only finitely often have
bounded/constant contribution to the redundancy,
(ix) is competitive with (slow) Bayesian mixing over all sub-alphabets.

Main Model S

$$S(x_{t+1}=i|x_{1:t}) \;:=\; \begin{cases} \dfrac{n_i^t}{t+\beta_t} & \text{for} \quad n_i^t>0 \\ \dfrac{\beta_t w_i^t}{t+\beta_t} & \text{for} \quad n_i^t=0 \end{cases}$$

- β_t = concentration parameter.
- w_i^t = weight of new symbol i at time t.
- n_i^t = number of times i appears in $x_{1:t}$.

Difference to $\text{DirM}(x_{t+1}=i|x_{1:t}) = \frac{n_i^t+\beta w_i}{t+\beta}$:

- Cases instead of sum.
- Time-dependent parameters.
- Easier to analyze.

CodeLength and Redundancy

Performance measure(s):

Code Length = –log-likelihood =

$$\mathrm{CL}_S(x_{1:n}) \;:=\; \ln 1/S(x_{1:n})$$

$\stackrel{+}{=}$ Redundancy = log-loss regret w.r.t. ML i.i.d. source:

$$R_S(a_{1:n}) \;:=\; \mathrm{CL}_S(x_{1:n}) - n\,H(\hat{\boldsymbol{\theta}}), \quad \text{where} \quad \hat{\theta}_i := n_i/n$$

Optimal Constant β

Approximate solution: $\beta^{min} \approx \beta^* := \frac{m}{2 \ln \frac{n}{m}}$

Discussion: $m \gg \ln n \Rightarrow$ "frequently" new symbols $\Rightarrow$ reserve more probability mass for new symbols $\Rightarrow$ make β large. $\surd$.

Discussion: $m \ll \ln n \Rightarrow$ new symbol rare $\Rightarrow$ reserve most probability mass for old symbols $\Rightarrow$ make β small. $\surd$.

Redundancy of S for "optimal" constant β^*

$$R_S^{\beta^*}(x_{1:n}) \leq \underbrace{\mathrm{CL}_w(\mathcal{A}) - m\ln m}_{\text{CL of unsorted } \mathcal{A}} + \sum_{j\in\mathcal{A}} \underbrace{\tfrac{1}{2}\ln n_j}_{R \text{ of } j} + \underbrace{m\ln\ln\tfrac{en}{m} + 0.6m}_{\text{small}} \lesssim \underbrace{\frac{D}{2}\ln n}_{\text{KT}}$$

where $\mathcal{A}$ = set of symbols observed

- Not an expectation. Holds on individual sequences.
- Bound is independent of base alphabet size D.
  $\Rightarrow$ Holds even for infinite and continuous alphabet $\mathcal{X}$.
  The weights w_i^t become (sub)probability densities.

Adaptive Variable β_t^*

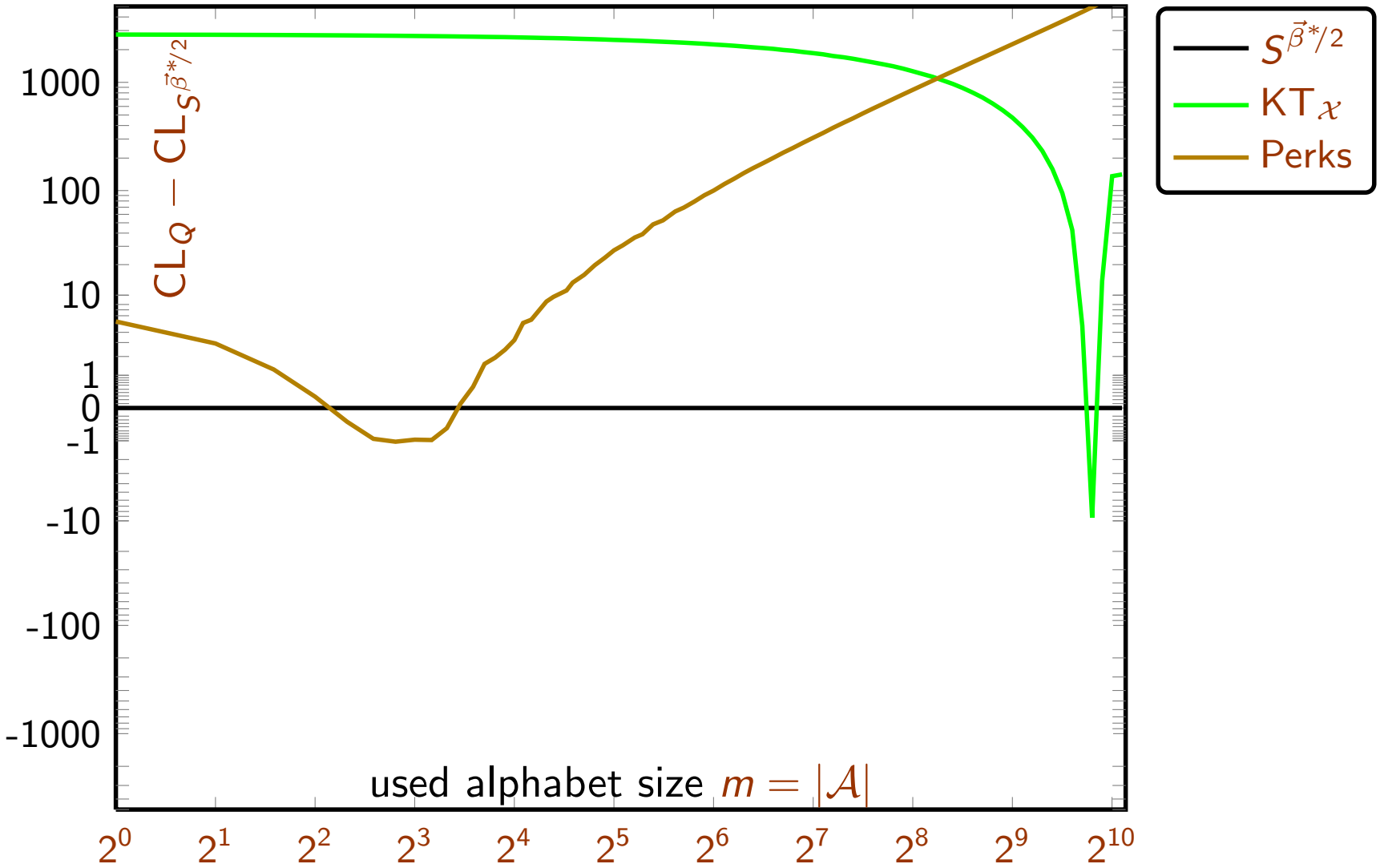
Problem: $\beta^* = m/2 \ln \frac{n}{m}$ depends on m and n $\Rightarrow$ S^{β^*} not online.

Solution: Replace $n \rightsquigarrow t$ and $m \rightsquigarrow m_t$, both known at time t and converging to n and m respectively, and regularize $t \rightsquigarrow t+1$:

Adaptive Variable $\beta_t^* := \frac{m_t}{2 \ln \frac{t+1}{m_t}}$

- Still same redundancy bound but somewhat worse constants.

Artificial Uniform Data



$\theta_{1:m} \sim$Uniform, $\theta_{m+1:D} = 0$, $n = 1024$, $D = 10\,000$, varying m.
The online/offline/oracle estimators have solid/dashed/dotted lines.

Summary of Experiments

Results are similar for other (n, D, m) combinations. Code length differences can be more or less pronounced, but are seldom reversed.

In short,

- $\mathrm{KT}_{\mathcal{X}}$ performs very poorly unless $m \approx D$;
- Perks performs poorly unless $m \lesssim \ln n$;
- $S^{\vec{\beta}^*/2}$ works in both situations