10-704: Information Processing and Learning **Spring 2012**

Lecture 12: Universality for Hierarchical Classes

Lecturer: Aarti Singh *Scribes: Daniel Munoz*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

12.1 Universal Coding for Stationary Processes

Last time we looked at redundancy bounds for IID and Markov processes, here we consider the more general class of stationary processes $\mathcal{P}$ (with a finite alphabet), *i.e.*, $\mathcal{P}_{\texttt{Markov-m}} \subset \mathcal{P}_{\texttt{stationary}}$, where $p(x^n) = \prod_{i=1}^n p(x_i|x_{i-m}^{i-1})$ for $p \sim \mathcal{P}_{\texttt{Markov-m}}$. We can approximate any stationary processes by increasing the order of the Markov process. Last class we ended with the theorem which we'll now prove.

Theorem 12.1 *Let $p \in \mathcal{P}_{\texttt{stationary}}$ with entropy rate $H_p(\mathcal{X}) = \lim_{m\to\infty} H_m$ where $H_m = H(X_{m+1}|X_1,\ldots,X_m)$. If C^m is a universal code for $\mathcal{P}_{\texttt{Markov-m}}$, then*

$$\mathbb{E}_p[R_{p,C^m}] \le H_m - H_p(\mathcal{X}) + \frac{1}{n}\frac{|\mathcal{X}|^m(|\mathcal{X}|-1)\log n}{2} + \frac{1}{n}K_m \tag{12.1}$$

On the rhs, the first two terms are the approximation error due to using a Markov-m process to approximate a stationary process, and the last two terms are the cost/expected redundancy for using a universal code for the Markov-m process.

Proof: Let q denote a universal predictor for a Markov-m process using which C^m is built. For each $p \in \mathcal{P}_{\texttt{stationary}}$, let $p^m(x^n) = p(x^m)\prod_{t=m+1}^n p(x_t|x_{t-m}^{t-1})$ denote the best Markov-m approximation to p. Then

$$\begin{aligned}
\mathbb{E}_p[R_{p,q}] &= \frac{1}{n}D_n(p||q) = \frac{1}{n}\mathbb{E}_p\left[\log\frac{p(x^n)}{q(x^n)}\right] = \frac{1}{n}\mathbb{E}_p\left[\log\frac{p(x^n)}{p^m(x^n)}\right] + \frac{1}{n}\mathbb{E}_p\left[\log\frac{p^m(x^n)}{q(x^n)}\right] && (12.2)\\
&= \frac{1}{n}D_n(p||p^m) + \frac{1}{n}\mathbb{E}_p\left[\log\frac{p^m(x^n)}{q(x^n)}\right], && (12.3)
\end{aligned}$$

The first term is the approximation error and the second is the estimation error. In the last class, we upper-bounded the expectation by $|\mathcal{X}|^m(|\mathcal{X}|-1)\log n/2 + K_m$. Now we upper-bound bound the first term.

$$\begin{aligned}
D_n(p||p^m) &= \mathbb{E}_p[\log p(x^n)] - \mathbb{E}_p[\log p^m(x^n)]\\
&= \mathbb{E}_p[\log p(x^n)] - \left(\mathbb{E}_p[\log p(x^m)] + \mathbb{E}_p[\log p(x_{m+1}|x_1^m)] + \ldots + \mathbb{E}_p[\log p(x_n|x_{n-m}^{n-1})]\right)\\
&= \mathbb{E}_p[\log p(x^n)] - (\mathbb{E}_p[\log p(x^m)] + (n-m)\mathbb{E}_p[\log p(x_{m+1}|x_1^m)])\\
&= -H(X^n) + H(X^m) + (n-m)H(X_{m+1}|X_1^m)\\
&= -H(X^n) + H(X^m) + (n-m)H_m
\end{aligned}$$

where the 3rd line holds due to p being stationary. Now we will upper-bound the first two terms by lower-bounding its negative

$$
\begin{aligned}
H(X^n) - H(X^m) &= \mathbb{E}_p\left[\log \frac{p(x^n)}{p(x^m)}\right] = \mathbb{E}_p[\log p(x_{m+1}^n | x_1^m)] \\
&= \mathbb{E}_p\left[\log \prod_{i=m+1}^{n} p(x_i|x_1^{i-1})\right] \quad \text{(chain rule)} \\
&= \sum_{i=m+1}^{n} H(X_i|X_1^{i-1}) \quad \text{(definition)} \\
&\geq (n-m)H_p(\mathcal{X})
\end{aligned}
$$

To see the last step, consider any term

$$
\begin{aligned}
H(X_i|X_1^{i-1}) &= H(X_j|X_{j-i+1}^{j-1}) \quad \text{(for any } j \geq i \text{ by stationarity)} \\
&\geq H(X_j|X_1^{j-1}) \quad \text{(conditioning does not increase entropy)}
\end{aligned}
$$

Since this is true for all $j \geq i$, it is also true if we let $j \to \infty$. Thus, $H(X_i|X_1^{i-1}) \geq H_p(\mathcal{X})$.

Therefore,

$$
\begin{aligned}
D_n(p||p^m) &\leq -(n-m)H_p(\mathcal{X}) + (n-m)H_m & (12.4) \\
&\leq n(H_m - H_p(\mathcal{X})), & (12.5)
\end{aligned}
$$

where the last line holds since $H_m \geq H_p(\mathcal{X})$ due to a Markov process being an approximation. ■

This theorem says we can handle stationary processes by designing a code that is universal for Markov-m processes by letting m scale with n (to drive approximation error to zero). However, we can only allow $m = o(\log n)$ in order to make sure the estimation error also goes down with n.

Also note that, in this case, the expected redundancy does not go to zero uniformly for all $p \in \mathcal{P}_{stationary}$, unlike universal coding of Markov processes. Thus, we achieve *weak* universality for stationary processes, whereas universal codes for Markov processes are *strongly* universal.

12.2 Hierarchical Universality

So far we have seen either small classes of iid or Markov processes on finite alphabets for which uniform redundancy rates via universal coding are possible, or a very large class of all stationary processes for which uniform redundancy bounds are not possible. A drawback of previous results is that if we use codes that are universal for a large class (say Lempel-Ziv codes which are universal for stationary processes), and if the source happened to be well-behaved (e.g. iid or Markov), then such codes may yield worse redundancy rates than an encoding which is tailored to the simpler class. Thus, there is a tradeoff between how large a class of source processes our code can be universal for and the redundancy rate it provides. Can we design a code for a large class that also works as well as the best code if the class happens to be well-behaved? The answer is yes.

Consider the countable union of a sequence of index sets $\Theta = \cup\{\Theta_k\}_{k\geq 1}$, where k indicates the complexity of the class. Often these indexed sets are nested ($\Theta_1 \subset \Theta_2 \subset \Theta_3 \ldots$).

Examples:

- Θ_k : k'th order Markov sources

- Θ_k : Histogram with k bins (non-parametric)
- Θ_k : Trees with k leaves

We would like to design a code that is universal for Θ but if the source happens to be a member of Θ_k, then it acts as if k was known.

In [RY84], the authors proposed such a code that adapts to the unknown order of a Markov process. The idea was to design two-stage codes, that first encode the class index k and then encode data using the best predictor for class Θ_k. Finally, the correct class is chosen adaptively as:

$$\hat{k} = \arg\min_k [L(k) + L_k(x^n)]. \tag{12.6}$$

where $L(k)$ is the length of a prefix code for integer k and $L_k(x^n)$ is the length of a universal code for class Θ_k. From a machine learning viewpoint, the first term can be thought of as a regularizer and the second term measures how well it fits the data. If $p \in \Theta_m$, the first term is bounded by $\log m + \log\log m$ (see homework problem on designing prefix codes for integers) and the second term by the bound on expected redundancy of universal code for $Theta_m$: $|\mathcal{X}|^m(|\mathcal{X}| - 1)\log n/2 + K_m$. In the next class, we will discuss how this relates to Minimum Descriptor Length (MDL).

References

[RY84] B. Ryabko and B. Ya, "Twice-universal coding," *Problems of Information Transmission.*, 1984, n3, pp.173-177.