Universal Loseless Compression: Context Tree Weighting(CTW)

Youngsuk Park

Dept. Electrical Engineering, Stanford University

Dec 9, 2014

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model. e.g. Huffman coding and Arithmetic coding

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model.
  e.g. Huffman coding and Arithmetic coding
- In the case where the actual model is unknown, we can still compress the sequence with universal compression.
  e.g. Lempel Ziv, and CTW

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model.
  e.g. Huffman coding and Arithmetic coding
- In the case where the actual model is unknown, we can still compress the sequence with universal compression.
  e.g. Lempel Ziv, and CTW
- **Aren't the LZ good enough?**

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model.
  e.g. Huffman coding and Arithmetic coding
- In the case where the actual model is unknown, we can still compress the sequence with universal compression.
  e.g. Lempel Ziv, and CTW
- **Aren't the LZ good enough?**
  - Yes. Universal w.r.t class of finite-state coders, and converging to entropy rate for stationary ergodic distribution

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model. e.g. Huffman coding and Arithmetic coding
- In the case where the actual model is unknown, we can still compress the sequence with universal compression.
  e.g. Lempel Ziv, and CTW
- **Aren't the LZ good enough?**
  - Yes. Universal w.r.t class of finite-state coders, and converging to entropy rate for stationary ergodic distribution
  - No. convergence is slow

Universal Coding with Model Classes

- Traditional Shannon theory assume a (probabilistic) model of data is known, and aims at compressing the data optimally w.r.t. the model.
  e.g. Huffman coding and Arithmetic coding
- In the case where the actual model is unknown, we can still compress the sequence with universal compression.
  e.g. Lempel Ziv, and CTW
- **Aren't the LZ good enough?**
  - Yes. Universal w.r.t class of finite-state coders, and converging to entropy rate for stationary ergodic distribution
  - No. convergence is slow
- We explore CTW.

Universal Compression and Universal Estimation

- A good compressor implies a good estimate of the source distribution.
  - For a given sequence x^n, $\mathbb{P}(x^n)$ be true distribution and $\mathbb{P}_L(x^n)$ be estimated distribution.

Universal Compression and Universal Estimation

- A good compressor implies a good estimate of the source distribution.
  - For a given sequence x^n, $\mathbb{P}(x^n)$ be true distribution and $\mathbb{P}_L(x^n)$ be estimated distribution.
  - We know that $\mathbb{P}_L(x^n) = \frac{2^{-L(x^n)}}{k_n}$ achieves the minimum expected code length where $k_n = \sum x^n 2^{-L(x^n)} \leq 1$ by Kraft Inequality

Universal Compression and Universal Estimation

- A good compressor implies a good estimate of the source distribution.
  - For a given sequence x^n, $\mathbb{P}(x^n)$ be true distribution and $\mathbb{P}_L(x^n)$ be estimated distribution.
  - We know that $\mathbb{P}_L(x^n) = \frac{2^{-L(x^n)}}{k_n}$ achieves the minimum expected code length where $k_n = \sum x^n 2^{-L(x^n)} \leq 1$ by Kraft Inequality
  - Then the expected redundancy is

$$\begin{aligned}\mathsf{Red}_n(\mathbb{P}_L, \mathbb{P}) &= \mathbb{E}[L(x^n) - \log \frac{1}{\mathbb{P}(x^n)}] && (1)\\ &= -\log k_n + D(\mathbb{P}||\mathbb{P}_L) && (2)\end{aligned}$$

Universal Compression and Universal Estimation

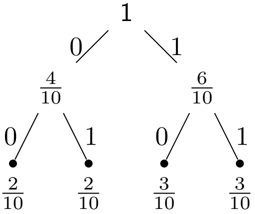
- A good compressor implies a good estimate of the source distribution.
  - For a given sequence x^n, $\mathbb{P}(x^n)$ be true distribution and $\mathbb{P}_L(x^n)$ be estimated distribution.
  - We know that $\mathbb{P}_L(x^n) = \frac{2^{-L(x^n)}}{k_n}$ achieves the minimum expected code length where $k_n = \sum x^n 2^{-L(x^n)} \leq 1$ by Kraft Inequality
  - Then the expected redundancy is

$$\text{Red}_n(\mathbb{P}_L, \mathbb{P}) = \mathbb{E}[L(x^n) - \log \frac{1}{\mathbb{P}(x^n)}] \tag{1}$$

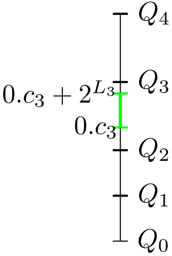
$$= -\log k_n + D(\mathbb{P}||\mathbb{P}_L) \tag{2}$$

  - A good compressor has a small redundacny. Therefore, a good compressor implies a good estimate of the true distribution.

- **Huffman code** an optimal prefix code

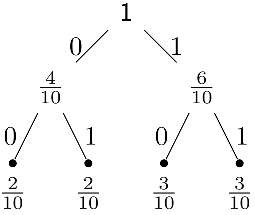


- **Arithmetic Code**
  For $L(x^n) = \left\lceil \log(\frac{1}{\mathbb{P}(x^n)}) \right\rceil + 1$,
  $.c = \left\lceil Q(x^n) \cdot 2^{L(x^n)} \right\rceil \cdot 2^{-L(x^n)}$

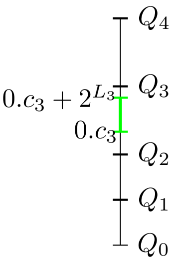


Two Entropy Coding for known parameters

- **Huffman code** an optimal prefix code



- **Arithmetic Code**
  For $L(x^n) = \left\lceil \log(\frac{1}{\mathbb{P}(x^n)}) \right\rceil + 1$,
  $.c = \left\lceil Q(x^n) \cdot 2^{L(x^n)} \right\rceil \cdot 2^{-L(x^n)}$



- What if parameters are unknown?

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?
Yes! with acceptable pointwise redundacny for all x^n

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?
Yes! with acceptable pointwise redundacny for all x^n
Note that $\min_{C \in \mathcal{C}} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{\mathbb{P}_\theta} = n\hat{H}(x^n) = nh_2(\hat{\theta})$

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?
Yes! with acceptable pointwise redundacny for all x^n
Note that $\min_{C \in \mathcal{C}} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{\mathbb{P}_\theta} = n\hat{H}(x^n) = nh_2(\hat{\theta})$

- **Two part code:** Use $\lceil \log(n+1) \rceil$ bits to encode n_1 and then tune code to $\theta = \frac{n_1}{n}$.

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?
Yes! with acceptable pointwise redundacny for all x^n
Note that $\min_{C \in \mathcal{C}} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{\mathbb{P}_\theta} = n\hat{H}(x^n) = nh_2(\hat{\theta})$

- **Two part code:** Use $\lceil \log(n+1) \rceil$ bits to encode n_1 and then tune code to $\theta = \frac{n_1}{n}$.
  - Then

$$\text{Red}_n(L, x^n) = \frac{\log(n+1)+2}{n} \to 0$$

Bernoulli Model: Two part code

Let $\mathcal{C} = \{\mathbb{P}_\theta, \theta \in [0,1]\}$, $X^n \overset{\text{iid}}{\sim} Bern(\theta)$. But we don't know θ.
Can we we still design the compressor?
Yes! with acceptable pointwise redundacny for all x^n
Note that $\min_{C \in \mathcal{C}} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{\mathbb{P}_\theta} = n\hat{H}(x^n) = nh_2(\hat{\theta})$

- **Two part code:** Use $\lceil \log(n+1) \rceil$ bits to encode n_1 and then tune code to $\theta = \frac{n_1}{n}$.
  - Then
$$\mathsf{Red}_n(L, x^n) = \frac{\log(n+1) + 2}{n} \to 0$$
  - Natural but dependent on n(not sequential)

Bernoulli Model: Mixture and Plug in code

- **Better code:**

Bernoulli Model: Mixture and Plug in code

- **Better code:**

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with n_1(n_1 is away from $0, n$), $L(x^n) = \log \binom{n}{n_1}$

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1 (n_1$ is away from $0, n)$, $L(x^n) = \log \binom{n}{n_1}$

$$\mathrm{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \qquad (3)$$

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1 (n_1$ is away from $0, n)$, $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0} \bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t | x^t)$ where

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with n_1(n_1 is away from $0, n$), $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0}\bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1}\frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with n_1(n_1 is away from $0, n$), $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0}\bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t | x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession
  - Use bias estimator of the conditional probability(plug-in approach)

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1 (n_1$ is away from $0, n)$, $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0} \bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t | x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession
  - Use bias estimator of the conditional probability(plug-in approach)
- **Mixture code** Mix over Dirichlet's density : $\frac{d\theta}{\Gamma(\frac{1}{2})^2 \sqrt{\theta(1-\theta)}} = dw(\theta)$

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1 (n_1$ is away from $0, n)$, $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0} \bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t | x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession
  - Use bias estimator of the conditional probability(plug-in approach)
- **Mixture code** Mix over Dirichlet's density : $\frac{d\theta}{\Gamma(\frac{1}{2})^2 \sqrt{\theta(1-\theta)}} = dw(\theta)$

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with n_1(n_1 is away from $0, n$), $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \rightarrow 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0}\bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession
  - Use bias estimator of the conditional probability(plug-in approach)
- **Mixture code** Mix over Dirichlet's density : $\frac{d\theta}{\Gamma(\frac{1}{2})^2\sqrt{\theta(1-\theta)}} = dw(\theta)$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)}{2n} + O(\frac{1}{n}) \rightarrow 0 \tag{4}$$

  - Note that $\mathbb{P}_L(x^n) = \int_0^1 \mathbb{P}_\theta(x^n) dw(\theta) = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t)$

Bernoulli Model: Mixture and Plug in code

- **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1(n_1$ is away from $0, n)$, $L(x^n) = \log \binom{n}{n_1}$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)+2}{2n} + O(\frac{1}{n}) \to 0 \tag{3}$$

  - Uniform assignment over types is equivalent to uniform mixture, $\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0}\bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t)$ where
  - where $p_L(0|x^t) = \frac{n_0(x^t)+1}{t+2}$ is Rule of succession
  - Use bias estimator of the conditional probability(plug-in approach)
- **Mixture code** Mix over Dirichlet's density : $\frac{d\theta}{\Gamma(\frac{1}{2})^2\sqrt{\theta(1-\theta)}} = dw(\theta)$

$$\text{Red}_n(L, x^n) \leq \frac{\log(n+1)}{2n} + O(\frac{1}{n}) \to 0 \tag{4}$$

  - Note that $\mathbb{P}_L(x^n) = \int_0^1 \mathbb{P}_\theta(x^n)dw(\theta) = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t)$
  - where $p_L(0|x^t) = \frac{n_0(x^t)+\frac{1}{2}}{t+1}$ is KT(Krichevski-Trofimov) estimator only depending on n_0, n_1

Tree Source with Known Model

If the next symbol depends on past symbols, then Tree model is a good choice.
Let a binary tree have leaves S. Then $\theta \in \mathbb{R}^{|S|}$.
For a fixed x^n, define the number of 0's after s as $n_0(s)$. Similarly define $n_1(s)$

Tree Source with Known Model

If the next symbol depends on past symbols, then Tree model is a good choice.
Let a binary tree have leaves S. Then $\theta \in \mathbb{R}^{|S|}$.
For a fixed x^n, define the number of 0's after s as $n_0(s)$. Similarly define $n_1(s)$

- For a $x^n = 00 \mid 0\mathbf{10}0\mathbf{10}1\mathbf{10}0$, sequence
  $n = 10, n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$

Tree Source with Known Model

If the next symbol depends on past symbols, then Tree model is a good choice.
Let a binary tree have leaves S. Then $\theta \in \mathbb{R}^{|S|}$.
For a fixed x^n, define the number of 0's after s as $n_0(s)$. Similarly define $n_1(s)$

- For a $x^n = 00 \mid 0\mathbf{10}0\mathbf{10}1\mathbf{10}0$, sequence
  $n = 10, n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$
- For each s, calculate KT estimator for $\mathbb{P}_{L,s}(n_0(s), n_1(s))$ and use $\mathbb{P}_L(x^n) = \prod_{s \in S} \mathbb{P}_{L,s}(n_0(s), n_1(s))$ for Arithmetic coding

Tree Source with Known Model

If the next symbol depends on past symbols, then Tree model is a good choice.

Let a binary tree have leaves S. Then $\theta \in \mathbb{R}^{|S|}$.

For a fixed x^n, define the number of 0's after s as $n_0(s)$. Similarly define $n_1(s)$

- For a $x^n = 00 \mid 0\mathbf{10}0\mathbf{10}1\mathbf{10}0$, sequence $n = 10, n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$
- For each s, calculate KT estimator for $\mathbb{P}_{L,s}(n_0(s), n_1(s))$ and use $\mathbb{P}_L(x^n) = \prod_{s \in S} \mathbb{P}_{L,s}(n_0(s), n_1(s))$ for Arithmetic coding
- Then redundancy is

$$\text{Red}_n(L, x^n) = L(x^n) - \log \frac{1}{\mathbb{P}(x^n)} \tag{5}$$

$$< \log(\frac{1}{\mathbb{P}_L(x^n)}) + 2 - \log \frac{1}{\mathbb{P}(x^n)} \tag{6}$$

$$= \log \frac{1}{\prod_{s \in S} \mathbb{P}_{L,s}(n_0(s), n_1(s))} - \log \frac{1}{\mathbb{P}(x^n)} + 2 \tag{7}$$

Tree Source with Known Model

- Continued.

$$\text{Red}_n(L, x^n) = \log\left(\frac{\prod_{s\in S} {\theta_s}^{n_0(s)} {\bar{\theta}_s}^{n_1(s)}}{\prod_{s\in S} P_{L,s}(n_0(s), n_1(s))}\right) + 2 \tag{8}$$

$$\leq \sum_{s\in S} \left(\frac{1}{2}\log(n_0(s) + n_1(s)) + 1\right) + 2 \tag{9}$$

$$\leq |S| \left(\frac{1}{2}\log\frac{\sum_{s\in S}(n_0(s) + n_1(s))}{|S|} + 1\right) + 2 \tag{10}$$

$$= |S| \left(\frac{1}{2}\log\frac{n}{|S|} + 1\right) + 2 \tag{11}$$

Tree Source with Known Model

- Continued.

$$\mathrm{Red}_n(L, x^n) = \log\left(\frac{\prod_{s\in S} \theta_s^{\,n_0(s)} \bar{\theta}_s^{\,n_1(s)}}{\prod_{s\in S} P_{L,s}(n_0(s), n_1(s))}\right) + 2 \tag{8}$$

$$\leq \sum_{s\in S} \left(\frac{1}{2}\log(n_0(s) + n_1(s)) + 1\right) + 2 \tag{9}$$

$$\leq |S| \left(\frac{1}{2}\log\frac{\sum_{s\in S}(n_0(s) + n_1(s))}{|S|} + 1\right) + 2 \tag{10}$$

$$= |S| \left(\frac{1}{2}\log\frac{n}{|S|} + 1\right) + 2 \tag{11}$$

- $O(\log(n))$ term is the cost for unknown paramters $\theta_s, s \in S$

Tree Source with Known Model

- Continued.

$$\text{Red}_n(L, x^n) = \log\left(\frac{\prod_{s\in S} \theta_s^{\,n_0(s)} \bar{\theta}_s^{\,n_1(s)}}{\prod_{s\in S} P_{L,s}(n_0(s), n_1(s))}\right) + 2 \tag{8}$$

$$\leq \sum_{s\in S} \left(\frac{1}{2}\log(n_0(s) + n_1(s)) + 1\right) + 2 \tag{9}$$

$$\leq |S| \left(\frac{1}{2}\log \frac{\sum_{s\in S}(n_0(s) + n_1(s))}{|S|} + 1\right) + 2 \tag{10}$$

$$= |S| \left(\frac{1}{2}\log \frac{n}{|S|} + 1\right) + 2 \tag{11}$$

- $O(\log(n))$ term is the cost for unknown paramters $\theta_s, s \in S$
- This upperbound through KT estimator meets the lowerbound of *Minimax* redundancy, *i.e.,* $\frac{|S|}{2}\log n + o(1)$

Tree Source with Unknown Model

What if we do not know the tree model either?

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$
  But it should be optimized over all trees and is not sequentially implementable

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$
  But it should be optimized over all trees and is not sequentially implementable
- **CTW**(Context Tree Weighting) defines a weighted coding distribution which takes into account all the possible tree sources that could lead to the sequence that we observe

Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$
  But it should be optimized over all trees and is not sequentially implementable
- **CTW**(Context Tree Weighting) defines a weighted coding distribution which takes into account all the possible tree sources that could lead to the sequence that we observe

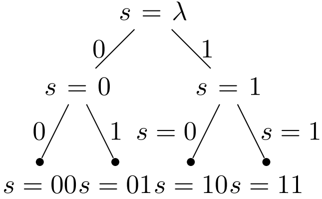
Tree Source with Unknown Model

What if we do not know the tree model either?
With assumption that tree has at most D depth,

- Two part code: Use n_T bits for representing tree and then use KT estimator for that tree, $L_T(x^n) + n_T$
  But it should be optimized over all trees and is not sequentially implementable
- **CTW**(Context Tree Weighting) defines a weighted coding distribution which takes into account all the possible tree sources that could lead to the sequence that we observe
- **Question:** The leaves S from this context tree might be different from actual tree model. But still good?

Tree Source with Unknown Model

- Calculate $n_0(s), n_1(s)$ for all $s \in \{s \mid s \in \{0,1\}^*, |s| \le D\}$



Ex) $x^n = 00 \mid 0100101100$

$n_0(\mathbf{0}) = 3, n_1(\mathbf{0}) = 3$

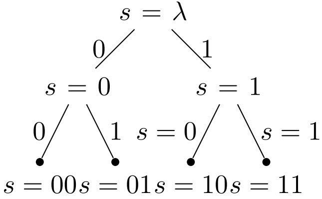
...

$n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$

$(x^n = 0\mathbf{10}0\mathbf{10}1\mathbf{10}0)$

Tree Source with Unknown Model

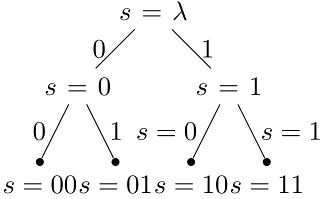
- Calculate $n_0(s), n_1(s)$ for all $s \in \{s \mid s \in \{0,1\}^*, |s| \leq D\}$



Ex) $x^n = 00 \mid 0100101100$
$n_0(\mathbf{0}) = 3, n_1(\mathbf{0}) = 3$
...
$n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$
$(x^n = 0\mathbf{10}0\mathbf{10}1\mathbf{10}0)$

- Calculate corresponding KT estimator $P_e(n_0(s), n_1(s))$ for each s.

- Calculate $n_0(s), n_1(s)$ for all $s \in \{s \mid s \in \{0,1\}^*, |s| \leq D\}$



Ex) $x^n = 00 \mid 0100101100$
$n_0(\mathbf{0}) = 3, n_1(\mathbf{0}) = 3$
...
$n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$
$(x^n = 0\mathbf{10}0\mathbf{10}1\mathbf{10}0)$

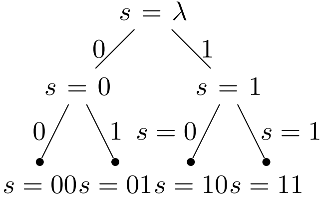
- Calculate corresponding KT estimator $P_e(n_0(s), n_1(s))$ for each s.
- Assign weighting probability

$$\mathbb{P}_w^s = \begin{cases} \mathbb{P}_e(n_0(s), n_1(s)) & \text{if } l(s) = D \\ \frac{\mathbb{P}_e(n_0(s), n_1(s)) + \mathbb{P}_w^{0s} P_w^{1s}}{2} & \text{if } l(s) \neq D \end{cases} \tag{12}$$

And take $\mathbb{P}_L(x^n) = \mathbb{P}_w^{\lambda}(x^n)$ and use it for Arithmetic coding.

Tree Source with Unknown Model

- Calculate $n_0(s), n_1(s)$ for all $s \in \{s \mid s \in \{0,1\}^*, |s| \leq D\}$



Ex) $x^n = 00 \mid 0100101100$
$n_0(\mathbf{0}) = 3, n_1(\mathbf{0}) = 3$
...
$n_0(\mathbf{10}) = 2, n_1(\mathbf{10}) = 1$
$(x^n = 0\mathbf{10}0\mathbf{10}1\mathbf{10}0)$

- Calculate corresponding KT estimator$P_e(n_0(s), n_1(s))$ for each s.
- Assign weighting probability

$$\mathbb{P}_w^s = \begin{cases} \mathbb{P}_e(n_0(s), n_1(s)) & \text{if } l(s) = D \\ \frac{\mathbb{P}_e(n_0(s), n_1(s)) + \mathbb{P}_w^{0s} P_w^{1s}}{2} & \text{if } l(s) \neq D \end{cases} \tag{12}$$

And take $\mathbb{P}_L(x^n) = \mathbb{P}_w^\lambda(x^n)$ and use it for Arithmetic coding.

- Corollary: Suppose x^n is drawn from $\mathbb{P}_1$ or $\mathbb{P}_2$. The one can achieve a redundacny of 1 bit by using the weighted distribution $\mathbb{P}^w = \frac{\mathbb{P}_1 + \mathbb{P}_2}{2}$

Tree Source with Unknown Model

Tree Source with Unknown Model

- Then

$$P_w^{\lambda}(x^n) = \sum_{S' \in \text{all } T_D} 2^{\Gamma_D(S')} \cdot \prod_{s \in S'} \mathbb{P}_e(n_0(s), n_1(s)) \tag{13}$$

$$\geq 2^{1-2|S|} \cdot \prod_{s \in S} \mathbb{P}_e(n_0(s), n_1(s)) \tag{14}$$

$$\geq 2^{1-2|S|} \mathbb{P}_L^{\text{known}}(x^n) \tag{15}$$

Tree Source with Unknown Model

- Then

$$P_w^{\lambda}(x^n) = \sum_{S' \in \text{all } T_D} 2^{\Gamma_D(S')} \cdot \prod_{s \in S'} \mathbb{P}_e(n_0(s), n_1(s)) \tag{13}$$

$$\geq 2^{1-2|S|} \cdot \prod_{s \in S} \mathbb{P}_e(n_0(s), n_1(s)) \tag{14}$$

$$\geq 2^{1-2|S|} \mathbb{P}_L^{\text{known}}(x^n) \tag{15}$$

- The redundancy is

$$\text{Red}_n(x^n) = \text{Red}_n^{\text{known}}(x^n) + 2|S| - 1 \tag{16}$$

$$= \frac{|S|}{2} \log \frac{n}{|S|} + |S| + 2 + 2|S| - 1 \tag{17}$$

Summary

- Two part code vs. Mixture code(Plug-in code)

Summary

- Two part code vs. Mixture code(Plug-in code)
- Look at KT estimator which is is optimal in minimax sense. And the mixture is sequentially implementable by KT estimator.

Summary

- Two part code vs. Mixture code(Plug-in code)
- Look at KT estimator which is is optimal in minimax sense. And the mixture is sequentially implementable by KT estimator.
- Even though we do not know the tree model(only depth is given), can design good compressor using CTW.

Reference

- Willems; Shtarkov; Tjalkens (1995), The Context-Tree Weighting Method: Basic Properties 41, IEEE Transactions on Information Theory
- Begleiter; El-Yaniv; Yona (2004), On Prediction Using Variable Order Markov Models 22, Journal of Artificial Intelligence Research: Journal of Artificial Intelligence Research, pp. 385421
- Some tutorial and lecture note