Universal Loseless Compression: Context Tree Weighting (CTW)

Youngsuk Park

Dept. Electrical Engineering, Stanford University

Dec 9, 2014
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.
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.
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?
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.
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.
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.
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.
A good compressor implies a good estimate of the source distribution.

For a given sequence $x^n$, $P(x^n)$ be true distribution and $P_L(x^n)$ be estimated distribution.

We know that $P_L(x^n) = \frac{2^{-L(x^n)}}{k_n}$ achieves the minimum expected code length where $k_n = \sum x^n 2^{-L(x^n)} \leq 1$ by Kraft Inequality.

A good compressor has a small redundancy. Therefore, a good compressor implies a good estimate of the true distribution.
A good compressor implies a good estimate of the source distribution.

- For a given sequence $x^n$, $P(x^n)$ be true distribution and $P_L(x^n)$ be estimated distribution.
- We know that $P_L(x^n) = \frac{2^{-L(x^n)}}{k_n}$ 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

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

(1)

$$= - \log k_n + D(P \| P_L)$$

(2)
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 $\mathbb{P}_L(x^n) = 2^{-L(x^n)}/k_n$ 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

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

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

A good compressor has a small redundancy. Therefore, a good compressor implies a good estimate of the true distribution.
Two Entropy Coding for known parameters

- **Huffman code** an optimal prefix code

```
  1
  /\  \
0   1
  /\  \
0 0 0
 2 2 3
```

- **Arithmetic Code**

For $L(x^n) = \left\lceil \log\left(\frac{1}{\mathbb{P}(x^n)}\right) \right\rceil + 1$,
\[
  c = \left\lceil Q(x^n) \cdot 2^L(x^n) \right\rceil \cdot 2^{-L(x^n)}
\]
Two Entropy Coding for known parameters

- **Huffman code** an optimal prefix code

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

- What if parameters are unknown?
Let \( C = \{ \mathbb{P}_\theta, \theta \in [0, 1] \}, \ X^n \overset{iid}{\sim} Bern(\theta). \) But we don’t know \( \theta. \)
Bernoulli Model: Two part code

Let $C = \{P_\theta, \theta \in [0, 1]\}$, $X^n \overset{iid}{\sim} Bern(\theta)$. But we don’t know $\theta$. Can we still design the compressor?

Two part code: Use $\lceil \log(n + 1) \rceil$ bits to encode $n$ and then tune code to $\theta = n^2$. Then $R_{n}(L,x^n) = \log(n + 1) + 2n \to 0$ natural but dependent on $n$ (not sequential).
Let $C = \{P_{\theta}, \theta \in [0, 1]\}$, $X^n \overset{iid}{\sim} \text{Bern}(\theta)$. But we don’t know $\theta$.

Can we still design the compressor?

Yes! with acceptable pointwise redundancy for all $x^n$
Let $C = \{ P_\theta, \theta \in [0, 1] \}$, $X^n \overset{iid}{\sim} Bern(\theta)$. But we don’t know $\theta$. Can we still design the compressor?
Yes! with acceptable pointwise redundancy for all $x^n$

Note that $\min_{C \in C} L_C(x^n) = \min_{\theta \in [0, 1]} \log \frac{1}{P_\theta} = n \hat{H}(x^n) = nh_2(\hat{\theta})$
Let $C = \{P_\theta, \theta \in [0, 1]\}$, $X^n \overset{iid}{\sim} Bern(\theta)$. But we don’t know $\theta$. Can we still design the compressor?

Yes! with acceptable pointwise redundancy for all $x^n$

Note that $\min_{C \in C} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{P_\theta} = n \hat{H}(x^n) = nh_2(\hat{\theta})$

- **Two part code:** Use $\lceil \log(n + 1) \rceil$ bits to encode $n_1$ and then tune code to $\theta = \frac{n_1}{n}$. 
Let $\mathcal{C} = \{P_\theta, \theta \in [0, 1]\}$, $X^n \overset{iid}{\sim} \text{Bern}(\theta)$. But we don’t know $\theta$. 
Can we still design the compressor?
Yes! with acceptable pointwise redundancy for all $x^n$
Note that $\min_{C \in \mathcal{C}} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{P_\theta} = n\hat{H}(x^n) = nh_2(\hat{\theta})$

- **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

$$\text{Red}_n(L, x^n) = \frac{\log(n + 1) + 2}{n} \to 0$$
Let $C = \{P_\theta, \theta \in [0, 1]\}$, $X^n \text{iid} \sim Bern(\theta)$. But we don’t know $\theta$.
Can we still design the compressor?
Yes! with acceptable pointwise redundancy for all $x^n$
Note that $\min_{C \in C} L_C(x^n) = \min_{\theta \in [0,1]} \log \frac{1}{P_\theta} = n \hat{H}(x^n) = nh_2(\hat{\theta})$

- **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
  $$\text{Red}_n(L, x^n) = \frac{\log(n + 1) + 2}{n} \to 0$$
- Natural but dependent on $n$(not sequential)
Better code:
Better code:
**Better code:** Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 \) (\( n_1 \) is away from 0, \( n \)), \( L(x^n) = \log \left( \binom{n}{n_1} \right) \)
**Better code:** Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 \) (\( n_1 \) is away from 0, \( n \)), \( L(x^n) = \log \left( \binom{n}{n_1} \right) \)

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1) + 2}{2n} + O\left(\frac{1}{n}\right) \to 0 \tag{3}
\]
**Better code:** Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 (n_1 \text{ is away from } 0, n) \), \( L(x^n) = \log \binom{n}{n_1} \)

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1) + 2}{2n} + O\left(\frac{1}{n}\right) \to 0 \quad (3)
\]

- Uniform assignment over types is equivalent to uniform mixture,
  \[
P_L(x^n) = \int_0^1 \theta^{n_0} \bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t) \quad \text{where}
\]
Better code: Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 \) (\( n_1 \) is away from 0, \( n \)), \( L(x^n) = \log \left( \binom{n}{n_1} \right) \)

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

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

where \( p_{L}(0|x^t) = \frac{n_0(x^t)+1}{t+2} \) is Rule of succession
Better code: Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 (n_1 \text{ is away from 0, } n) \), \( L(x^n) = \log \left( \binom{n}{n_1} \right) \)

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

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

Use bias estimator of the conditional probability (plug-in approach)
Better code: Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1 \) (\( n_1 \) is away from 0, \( n \)), \( L(x^n) = \log \binom{n}{n_1} \)

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1) + 2}{2n} + O\left(\frac{1}{n}\right) \to 0 \quad (3)
\]

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

\[
\frac{d\theta}{\Gamma(\frac{1}{2})^2 \sqrt{\theta(1-\theta)}} = dw(\theta)
\]
• **Better code:** Enumerate all $\binom{n}{n_1}$ sequences with $n_1$ ($n_1$ is away from 0, $n$), $L(x^n) = \log \binom{n}{n_1}$

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1) + 2}{2n} + O\left(\frac{1}{n}\right) \to 0
\]  

(3)

• Uniform assignment over types is equivalent to uniform mixture,

\[
\mathbb{P}_L(x^n) = \int_0^1 \theta^{n_0} \bar{\theta}^{n_1} 1 d\theta = \binom{n}{n_1}^{-1} \frac{1}{n+1} = \prod_{t=0}^{n-1} p_L(x^t_{t+1}|x^t)
\]

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:

\[
\frac{d\theta}{\Gamma(\frac{1}{2})^2 \sqrt{\theta(1-\theta)}} = dw(\theta)
\]
**Better code:** Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1(n_1 \text{ is away from } 0, n) \), \( L(x^n) = \log \binom{n}{n_1} \)

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1) + 2}{2n} + O\left(\frac{1}{n}\right) \to 0 \quad (3)
\]

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

**Mixture code** Mix over Dirichlet’s density:

\[
\frac{d\theta}{\Gamma\left(\frac{1}{2}\right)^2 \sqrt{\theta(1-\theta)}} = dw(\theta)
\]

\[
\text{Red}_n(L, x^n) \leq \frac{\log(n + 1)}{2n} + O\left(\frac{1}{n}\right) \to 0 \quad (4)
\]

- Note that \( \mathbb{P}_L(x^n) = \int_0^1 \mathbb{P}_\theta(x^n) \, dw(\theta) = \prod_{t=0}^{n-1} p_L(x_{t+1}^t|x^t) \)
Better code: Enumerate all \( \binom{n}{n_1} \) sequences with \( n_1(n_1 \text{ is away from } 0, n) \), \( L(x^n) = \log \binom{n}{n_1} \)

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

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

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 : \( \frac{d\theta}{\Gamma(\frac{1}{2})^2 \sqrt{\theta(1-\theta)}} \) = \( dw(\theta) \)

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

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 \( n_0, n_1 \)
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 \( x^n = 00 | 0100101100 \), sequence
  - \( n = 10, n_0(10) = 2, n_1(10) = 1 \)
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 $x^n = 00 \mid 0100101100$, sequence $n = 10$, $n_0(10) = 2$, $n_1(10) = 1$
- For each $s$, calculate KT estimator for $P_{L,s}(n_0(s), n_1(s))$ and use $P_L(x^n) = \prod_{s \in S} P_{L,s}(n_0(s), n_1(s))$ 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 $x^n = 00 \mid 0\textbf{100101100}$, sequence $n = 10, n_0(\textbf{10}) = 2, n_1(\textbf{10}) = 1$
- 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

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

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

$$= \log \frac{1}{\prod_{s \in S} \mathbb{P}_{L,s}(n_0(s), n_1(s))} - \log \frac{1}{\mathbb{P}(x^n)} + 2$$  \hspace{1cm} (7)
Tree Source with Known Model

Continued.

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

(8)

\[
\leq \sum_{s \in S} \left( \frac{1}{2} \log(n_0(s) + n_1(s)) + 1 \right) + 2
\]

(9)

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

(10)

\[
= |S| \left( \frac{1}{2} \log \frac{n}{|S|} + 1 \right) + 2
\]

(11)
Tree Source with Known Model

Continued.

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

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

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

\[
= |S| \left( \frac{1}{2} \log \frac{n}{|S|} + 1 \right) + 2 \quad (11)
\]

\(O(\log(n))\) term is the cost for unknown parameters \(\theta_s, s \in S\)
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 parameters \(\theta_s, s \in S\)
- 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?
What if we do not know the tree model either?
With assumption that tree has at most $D$ depth,
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, $L_T(x^n) + n_T$
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, $L_T(x^n) + n_T$
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
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
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.
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?
Calculate \( n_0(s), n_1(s) \) for all \( s \in \{ s \mid s \in \{0, 1\}^*, |s| \leq D \} \)

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

\( n_0(0) = 3, n_1(0) = 3 \)

... \( n_0(10) = 2, n_1(10) = 1 \)

\( x^n = 0100101100 \)
Calculate \( n_0(s), n_1(s) \) for all \( s \in \{ s \mid s \in \{0, 1\}^* , |s| \leq D \} \)

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

\( n_0(0) = 3, n_1(0) = 3 \)

... \( n_0(10) = 2, n_1(10) = 1 \)

(\( x^n = 0100101100 \))

- Calculate corresponding KT estimator \( P_e(n_0(s), n_1(s)) \) for each \( s \).
  - \( s = \lambda \)
  - \( s = 0 \)
  - \( s = 1 \)
  - \( s = 00 \)
  - \( s = 01 \)
  - \( s = 10 \)
  - \( s = 11 \)
Calculate \( n_0(s), n_1(s) \) for all \( s \in \{ s \mid s \in \{0, 1\}^*, |s| \leq D \} \)

Ex) \( x^n = 00 | 0100101100 \)

\( n_0(0) = 3, n_1(0) = 3 \)

\[ n_0(10) = 2, n_1(10) = 1 \]

\[ (x^n = 0100101100) \]

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

Assign weighting probability

\[
P_w^s = \begin{cases} 
P_e(n_0(s), n_1(s)) & \text{if } l(s) = D \\
\frac{P_e(n_0(s), n_1(s)) + P_0^s P_{1s}^w}{2} & \text{if } l(s) \neq D
\end{cases}
\] (12)

And take \( P_L(x^n) = P_w^\lambda(x^n) \) and use it for Arithmetic coding.
Calculate $n_0(s), n_1(s)$ for all $s \in \{ s \mid s \in \{0, 1\}^*, |s| \leq D \}$

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

$n_0(0) = 3, n_1(0) = 3$

... $n_0(10) = 2, n_1(10) = 1$

($x^n = 0100101100$)

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

Assign weighting probability

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

And take $P_L(x^n) = P_w^\lambda(x^n)$ and use it for Arithmetic coding.

Corollary: Suppose $x^n$ is drawn from $P_1$ or $P_2$. The one can achieve a redundancy of 1 bit by using the weighted distribution $P_w = \frac{P_1 + P_2}{2}$
\[ P(\lambda w(x^n)) = \sum_{S' \in \text{all } T} \Gamma_{D, 2}(S') \cdot \prod_{s \in S'} P_e(n_0(s), n_1(s)) \] (13)

\[ \geq 2^{1 - 2|S|} \cdot \prod_{s \in S} P_{\text{known}}(x^n) \] (14)

The redundancy is

\[ \text{Red}_{n}(x^n) = \text{Red}_{\text{known}}(x^n) + 2|S| - 1 \] (16)

\[ = |S|^2 \log n |S| + |S| + 2 + 2|S| - 1 \] (17)
Tree Source with Unknown Model

Then

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

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

\[ \geq 2^{1 - 2|S|} \mathbb{P}_{L}^{\text{known}}(x^n) \]
Then

\[ P_w^\lambda(x^n) = \sum_{S' \in \text{all } T_D} 2^{\Gamma_D(S')} \cdot \prod_{s \in S'} P_e(n_0(s), n_1(s)) \]  

\[ \geq 2^{1 - 2|S|} \cdot \prod_{s \in S} P_e(n_0(s), n_1(s)) \]  

\[ \geq 2^{1 - 2|S|} P_{L}^{\text{known}}(x^n) \]  

The redundancy is

\[ \text{Red}_n(x^n) = \text{Red}_{n}^{\text{known}}(x^n) + 2|S| - 1 \]  

\[ = \frac{|S|}{2} \log \frac{n}{|S|} + |S| + 2 + 2|S| - 1 \]
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 optimal in minimax sense. And the mixture is sequentially implementable by KT estimator.
Two part code vs. Mixture code (Plug-in code)

Look at KT estimator which 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), we can design good compressor using CTW.
- Some tutorial and lecture note