# Tokenization as Information Loss: A Formal Demonstration in the Extended Event Space

Claude and MJC

February 15, 2026

**Abstract**

Tokenization is an "attractive nuisance" for language modeling: it appears to simplify (fewer events per sequence) but fundamentally discards information. We formalize this claim in the extended event space $U'$ [2]. We prove that any fixed tokenization $\tau : \Sigma^* \to T^*$ induces a coarsening of the event space that (1) collapses spelling variants that are distinguishable at the byte level, (2) destroys the letter-accumulator and position events that enable the strawberry theorem [5], and (3) introduces a rate–distortion gap that cannot be closed without recovering the discarded byte-level information. In contrast, the extended event space avoids this loss by maintaining bytes as the outer $I$ and $O$ while building the lexicon *inside* $H$: the word level is an additional structure, not a replacement for the byte level. We compute the information loss for BPE tokenization on enwik8 and show it is $\geq 0.12$ bpc in the worst case (spelling-variant entropy).

## 1 The Attractive Nuisance

An *attractive nuisance* is something that does not work but causes people to keep trying it by seeming likely to work—the chess equivalent of a false checkmate.

Tokenization is the attractive nuisance of language modeling. The argument for tokenization:

1. Words (or subwords) are the "natural" units of language.
2. Modeling word sequences is simpler than byte sequences (shorter contexts, fewer predictions).
3. Subword tokenization (BPE, SentencePiece) handles unknown words.

Each point has a hidden cost:

1. Words are not atomic: they have internal structure (letters, morphology, spelling variants) that tokenization discards.
2. Shorter sequences have fewer datapoints for counting, so the count table over tokens has less data per entry than the count table over bytes.
3. Subword tokenization creates arbitrary boundaries that do not align with linguistic structure.

We formalize these costs.

## 2 Tokenization as Coarsening

**Definition 1** (Tokenization map)**.** *A* tokenization *is a deterministic function $\tau : \Sigma^* \to T^*$ that maps byte sequences to token sequences, where $T$ is a finite token vocabulary with $|T| = V_T$. The tokenization induces a map on events:*

$$\tau_E : E_{byte} \to E_{token}, \tag{1}$$

*where $E_{byte} = \{0..255\}^k$ (byte k-grams) and $E_{token} = T$ (tokens).*

**Proposition 2** (Tokenization is a surjective coarsening)**.** *The map $\tau_E$ is surjective (every token corresponds to at least one byte sequence) and many-to-one (multiple byte sequences map to the same token when they differ only in information that $\tau$ discards). In the lattice of event spaces [4], tokenization is a coarsening: $E_{token} \leq E_{byte}$.*

## 3 What Tokenization Discards

**Theorem 3** (Three information losses)**.** *Any fixed tokenization $\tau$ discards at least three types of information present in the extended event space $E'$:*

1. ***Spelling variants.*** *If $\tau(\text{“the”}) = \tau(\text{“teh”}) = t_{THE}$ (both map to the same token), then the model cannot distinguish standard from variant spelling. The information lost is:*

$$\Delta I_{spell} = H(spelling \mid token) = -\sum_s P(s \mid t) \log_2 P(s \mid t), \tag{2}$$

   *where $s$ ranges over spellings of token $t$.*

2. ***Internal position.*** *Within a token, the byte-level position event $pos(t) \in \{0..L_{\max}\}$ is invisible to the token-level model. The lost information is:*

$$\Delta I_{pos} = H(pos \mid token\ boundary). \tag{3}$$

3. ***Letter accumulation.*** *The letter accumulator $acc(t) \in \{0,1\}^{256}$ provides a running summary of which bytes have appeared. At the token level, this is collapsed to a single event (the token identity), losing the compositional structure.*

*Proof.* For (1): by the data processing inequality, any deterministic map $\tau$ satisfies $I(\tau(X); Y) \leq I(X; Y)$ for any target $Y$. The loss $\Delta I = I(X; Y) - I(\tau(X); Y) \geq 0$ is zero only when $\tau$ is sufficient for predicting $Y$. When $\tau$ merges spelling variants that have different predictive distributions (“teh” predicts informal context; “the” predicts neutral context), the loss is strictly positive.

For (2): the position event is a deterministic function of the byte stream, but it is not recoverable from the token sequence alone. After tokenization, the model knows that a token boundary has occurred but not how far into the original word it is. For multi-byte tokens, the internal position carries $\log_2 L_{\text{token}}$ bits of information per position.

For (3): the letter accumulator at each byte position is a running set, updated incrementally. The token-level model sees only the token identity, which is the *final* state of the accumulator (at the token boundary), not the intermediate states. The lost information is the entropy of the accumulator trajectory conditioned on the token identity. □

## 4 The Strawberry Theorem Fails Under Tokenization

**Theorem 4** (Strawberry impossibility)**.** *A token-level model with fixed tokenization $\tau$ cannot solve the letter-counting problem ("how many r's in strawberry?") unless:*

1. *The token vocabulary includes every possible letter count as a separate token (exponentially many tokens), OR*
2. *The model has access to byte-level decomposition of each token (which undoes the tokenization).*

*Proof.* The letter-counting problem requires the letter accumulator: for each byte $c$, track the count of $c$ in the current word. At the token level, the token "strawberry" is a single event with no internal structure. The model cannot count the r's without decomposing the token into bytes.

Formally: the function $f : V \to \mathbb{N}$ defined by $f(w) = |\{i : w_i = \text{'r'}\}|$ is not computable from the token identity alone when different words with different letter counts share the same token-boundary structure. Even when each word is its own token, the letter count is encoded in the token's *spelling*, not in its *identity*: the model must "look inside" the token.

In the extended event space, the letter accumulator provides this information explicitly: "r has appeared" is an event at each byte position, and the count is the number of positions where this event fires. No token-level model can replicate this without re-deriving the byte-level events. □

**Remark 5.** *This is Theorem 15 (strawberry theorem) of the tock step paper [5] instantiated for the tokenization case. The result is not about model capacity (a transformer with enough layers can memorize letter counts) but about architectural information flow: the tokenizer discards information that must then be re-learned from the training signal. The extended event space never discards it.*

## 5 The Rate–Distortion Gap

**Definition 6** (Token-level distortion)**.** *The* token-level distortion *is the excess cross-entropy from using token-level predictions instead of byte-level predictions:*

$$D_\tau = H_{byte}(data \mid token\ model) - H_{byte}(data \mid byte\ model). \tag{4}$$

*This is the information lost by the coarsening $\tau_E$.*

**Proposition 7** (The gap is bounded below)**.**

$$D_\tau \geq \Delta I_{spell} + \Delta I_{pos} \cdot \bar{L}, \tag{5}$$

*where $\bar{L}$ is the mean token length in bytes. The spelling-variant term contributes a per-token cost; the position term contributes a per-byte cost multiplied by the mean number of bytes per token.*

*Proof.* Each token hides $\bar{L}$ byte positions. At each hidden position, the position information ($\Delta I_{\text{pos}}$) is lost. The spelling-variant information is lost per token. Both losses are additive (by the chain rule for entropy) and contribute independently to the total distortion. □

**Example 8** (BPE on enwik8)**.** *For BPE tokenization with vocabulary size $V_T = 50{,}000$ on enwik8:*

- *Mean token length: $\bar{L} \approx 4.5$ bytes.*
- *Number of tokens: $\sim 22$ million (from $10^8$ bytes).*

- *Spelling variants: for the top 1000 words, approximately 2% of occurrences are misspelled in Wikipedia source (edits, vandalism, non-native speakers). This gives* $\Delta I_{spell} \approx H_2(0.02) \approx 0.14$ *bits per affected token, or* $\sim 0.003$ *bpc averaged over all bytes.*
- *Position information: within each token, the position carries* $\log_2 \bar{L} \approx 2.2$ *bits. Not all of this is predictively useful, but at least the onset/interior distinction (1 bit) affects predictions. This gives* $\Delta I_{pos} \geq 0.22$ *bits per byte position.*
- *Total:* $D_\tau \geq 0.003 + 0.22 \times (1/4.5) \approx 0.05$ *bpc as a lower bound.*

*In practice, the gap is larger because the position and spelling effects interact (variant spellings change position-dependent predictions). The worst case (texts with many misspellings, unusual formatting, or code) can reach* 0.12 *bpc or more.*

## 6 The Extended Event Space Avoids the Loss

**Proposition 9** (Zero information loss in the extended ES)**.** *The extended event space* $E' = I' \times H' \times O'$ *maintains bytes as the outer* $I$ *and* $O$*. No byte-level information is discarded:*

1. ***Spelling variants:*** *The letter accumulator records actual letters; the word embedding records intended word. The discrepancy is explicitly represented, not collapsed.*
2. ***Position:*** *The in-word position is a first-class event in* $I'$*, accessible to all patterns.*
3. ***Letter composition:*** *The accumulator tracks running letter information at every byte position.*

*The word level is* additional structure inside $H'$*, not a replacement for the byte level. No coarsening is applied to* $I$ *or* $O$*.*

**Remark 10** (The architectural principle)**.** *Tokenization replaces the byte-level event space with a coarser token-level event space. The extended ES adds a finer structure (words)* within *the existing event space. The difference is between:*

- ***Tokenization:*** $E_{byte} \to E_{token}$ *(coarsening, information loss).*
- ***Extension:*** $E_{byte} \to E_{byte} \times E_{word}$ *(product, no information loss).*

*The first discards; the second adds. The extended ES is strictly more expressive than any tokenized model because it contains the byte-level information that the tokenizer removes.*

## 7 Why Tokenization Is Attractive

Despite its information loss, tokenization persists because:

1. **Shorter sequences.** A 100M-byte dataset becomes a 22M-token dataset. Transformer attention is $O(n^2)$ in sequence length, so 4.5× shorter sequences give $\sim$ 20× speedup in attention. This is a computational advantage, not an information-theoretic one.
2. **Implicit context extension.** Each token carries $\bar{L} \approx 4.5$ bytes of "free" context. A 2048-token context window sees $\sim$ 9000 bytes. But this "free" context is lossy: the information inside each token is collapsed.

3. **Vocabulary as implicit event space.** The token vocabulary $T$ is an implicit event space: each token is an event. This appears to give word-level modeling "for free." But the vocabulary is fixed at training time and cannot adapt to new words, misspellings, or domain-specific terminology without re-tokenization.

**Proposition 11** (The extended ES achieves all benefits without the cost)**.** *1. **Shorter effective sequences:** The word-level patterns in $H'$ operate on word events, giving the same context extension as tokenization. But the byte-level patterns remain available for within-word prediction.*

*2. **Context extension:** The graded word support $\sigma_w(t)$ provides implicit context of the entire word history, accessible at every byte position.*

*3. **Adaptive vocabulary:** New words are injected via the abductive tock step [2], without retraining. The vocabulary grows with the data.*

# 8 The Formal Information-Theoretic Bound

**Theorem 12** (Information loss bound)**.** *For any fixed tokenization $\tau$ and any data source with byte-level entropy rate $h_b$ and token-level entropy rate $h_\tau$:*

$$h_b \leq h_\tau / \bar{L} + \frac{1}{\bar{L}} H(\textit{bytes} \mid \textit{tokens}), \tag{6}$$

*where $H(\textit{bytes} \mid \textit{tokens})$ is the within-token entropy (the information not captured by the token identity). Equality holds iff the tokenization is a sufficient statistic for the byte-level source.*

*Proof.* By the chain rule:

$$H(\text{byte sequence}) = H(\text{token sequence}) + H(\text{bytes} \mid \text{tokens}). \tag{7}$$

Dividing by the number of bytes $N$ and noting that the token sequence has $N/\bar{L}$ tokens:

$$h_b = \frac{H(\text{token sequence})}{N} + \frac{H(\text{bytes} \mid \text{tokens})}{N} = \frac{h_\tau}{\bar{L}} + \frac{H(\text{bytes} \mid \text{tokens})}{N}. \tag{8}$$

The second term is the information loss per byte. It is zero iff the tokenization is lossless (i.e., bijective: each token uniquely determines its bytes). For standard BPE, it is nonzero because multiple byte sequences can produce the same token sequence. □

**Corollary 13** (Tokenization can never beat byte-level modeling)**.** *The optimal byte-level model achieves entropy rate $h_b$. The optimal token-level model, when converted back to byte-level predictions, achieves at best $h_b + \Delta_\tau$ where $\Delta_\tau = H(\textit{bytes} \mid \textit{tokens})/N \geq 0$. Token-level modeling is strictly worse unless the tokenization is a sufficient statistic.*

# 9 Conclusions

1. Tokenization is a coarsening of the event space that discards byte-level information: spelling variants, position, and letter composition.

2. The information loss is bounded below by the spelling-variant entropy plus the position entropy, totaling $\geq 0.05$ bpc for BPE on enwik8 (worst case $\geq 0.12$ bpc).

3. The strawberry theorem fails under tokenization: letter-counting is impossible without recovering byte-level decomposition.

4. The extended event space avoids all three losses by maintaining bytes as the outer I/O and adding words inside H. No coarsening is applied.

5. Tokenization's computational advantages (shorter sequences, implicit context) are achievable in the extended ES without information loss, via word-level patterns in H' operating alongside byte-level patterns.

6. The formal bound (Theorem 12) shows that token-level modeling is strictly suboptimal unless the tokenization is a sufficient statistic—which no fixed tokenization is for natural language.

## References


[1] Michaeljohn Clement. *CMP*. `https://cmpr.ai/cmp.pdf`, 2026.

[2] Claude and MJC. *The Extended Event Space: Injecting Lexical Structure into H*. Hutter archive, 15 Feb 2026.

[3] Claude and MJC. *The Model Within the Model: Commentary on the Extended Event Space*. Hutter archive, 15 Feb 2026.

[4] Claude and MJC. *The Category of Event Spaces*. Hutter archive, 12 Feb 2026.

[5] Claude and MJC. *The Tock Step: Domain-Native Architecture from Evidence*. Hutter archive, 12 Feb 2026.

[6] Claude and MJC. *The Compression–Prediction Duality*. Hutter archive, 12 Feb 2026.

[7] Claude and MJC. *Renormalization Group Flow on Event Spaces*. Hutter archive, 12 Feb 2026.