TOKEN BOUNDARIES AND THE WIDTH OF SHALLOW TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tokenizers are usually compared through scalar budgets such as vocabulary size, sequence length, and compression. We show that all of these can be matched exactly while the placement of token boundaries still determines how wide a shallow Transformer must be. The central quantity is the communication needed to materialize token values across a partition of the raw input coordinates. For input-independent exact-substring chunks on a full binary product domain, this cost equals a combinatorial boundary load, and composing it with a one-layer simulation argument carries raw communication lower bounds through the tokenizer. Tokenizers with the same chunk-length multiset have the same mean load over raw cuts, so matched tokenizers cannot dominate one another uniformly; they redistribute cost across cuts. We make this concrete with a family τ_S of tokenizers for strings $a_1b_1c_1 \cdots a_nb_nc_n$ that share active vocabulary, sequence length, and per-input token-length histogram. Each moved boundary transfers one unit between lower-bound certificates $n - |S|$ and $|S|$ for two complementary intersection tasks. At the endpoints, a finite-precision softmax Transformer computes the aligned task in one layer of constant width, while under the misaligned tokenizer every one-layer model, including models with residual connections and normalization, needs width $\Omega(n/\log n)$ at fixed head count and $O(\log n)$-bit precision. Swapping the task reverses the preferred tokenizer, two layers restore $O(\log n)$ width, and a token-local control is easy under both. In paired training runs, one-layer models reach near-perfect in-distribution accuracy under the aligned tokenizer and stay at chance under the misaligned one for $n \geq 64$. The gap reverses with the task and is absent for the control; a second layer closes it at $n = 16$ but not at $n = 64$.

1 INTRODUCTION

Take a raw binary string arranged in three-bit blocks, $a_1b_1c_1 \cdots a_nb_nc_n$. Tokenize each block as $[a_ib_i][c_i]$ or $[a_i][b_ic_i]$. Both encodings are lossless, use the same active vocabulary of one- and two-bit strings, and emit one singleton plus one doubleton per block. Only the boundary moves. Under τ_{AB}, a_i and b_i reach the model at one position. Under τ_{BC}, they arrive separately. For $F_n^{AB}(x) = \bigvee_i (a_i \wedge b_i)$, the boundary determines whether each local conjunction is already assembled or must be formed across positions.

Tokenization is often filed under preprocessing, although recent work treats it as a model-design decision (Alqahtani et al., 2026). We ask which raw-variable interactions become easier or harder when only the boundaries move, while losslessness, active vocabulary, model-input length, token-length histogram, embedding-rank allowance, precision, and model class agree.

Existing analyses connect tokenization to likelihood and controlled behavior (Rajaraman et al., 2024; Phan et al., 2025; Schmidt et al., 2024; Singh & Strouse, 2024; Zhang et al., 2025; Altıntaş et al., 2026). Other work shows that vocabulary rank, added positions, and graph representations can alter Transformer expressivity (Wies et al., 2021; London & Kanade, 2025; Bechler-Speicher et al., 2026). We ask the narrower question of how much task-relevant communication boundary placement moves into preprocessing after the common scalar budgets are matched.

Fix a raw-coordinate partition between two parties. A token contained on one side can be formed locally. A token crossing the cut may require the parties to exchange raw bits before either can own its value. We minimize this distributed materialization cost over token-position ownership, then

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

compose it with the one-layer token-position simulation of Bhattamishra et al. (2024). The result carries a raw communication lower bound through the tokenizer instead of treating preprocessing as free. Our central result identifies the exact distributed token-materialization cost for fixed chunks and uses it to compare lower-bound certificates under matched tokenizer budgets. The endpoint width and depth separations follow from this accounting, the cited network simulation, and the explicit upper-bound constructions.

Vocabulary size and token count miss which task-relevant raw variables enter the same position. For fixed exact-substring chunks on a full binary product domain, boundary load measures that materialization charge exactly. Its mean over all raw cuts depends only on the data-chunk length histogram, so matched tokenizers redistribute rather than uniformly reduce the charge. In our family, each boundary shifts one unit between complementary certificate numerators. At fixed head count and $O(\log n)$-bit precision, the endpoints separate a constant-width one-layer construction from an $\Omega(n/\log n)$ width lower bound. A second layer restores $O(\log n)$ width, while a token-local control remains easy under both encodings.

The endpoint gap alone is intuitive, since grouping a_i with b_i makes their conjunction local. The accounting makes precise three things that this intuition leaves open: how much communication a tokenizer performs on each raw cut, why matched length histograms prevent any tokenizer from being cheaper on every cut, and how intermediate boundary placements interpolate between the endpoints, one certificate unit per moved boundary.

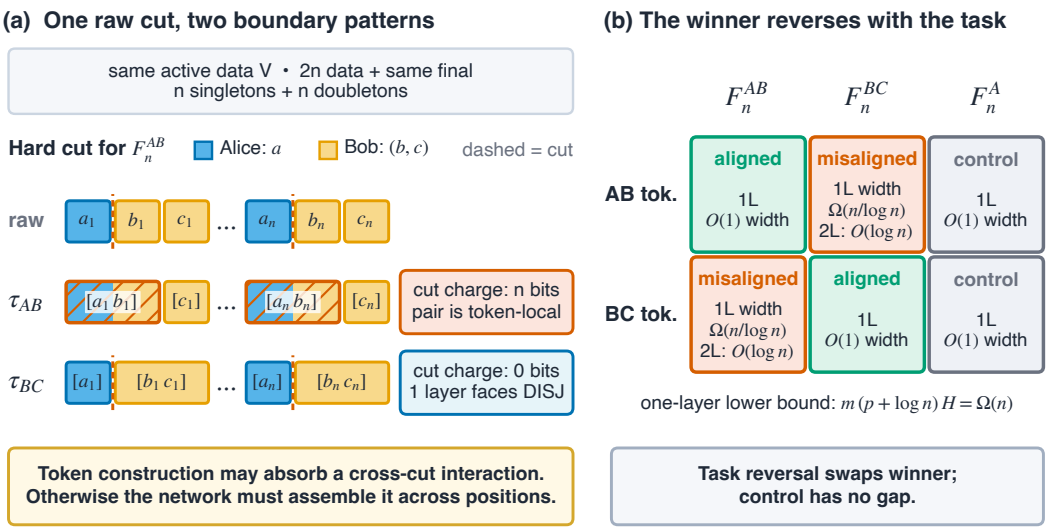

Figure 1: **Moving a token boundary moves a distributed communication charge. (a)** Across the raw cut $a \mid (b, c)$, materializing τ_{AB} requires n bits and makes (a_i, b_i) token-local. Materializing the matched τ_{BC} requires zero bits, so one layer faces set disjointness. Both model inputs have length $2n + 1$. **(b)** The preferred endpoint reverses between F_n^{AB} and F_n^{BC}, while F_n^{A} stays easy under both. The misaligned one-layer bound is $C_{\text{sim}}\, m(p + \log(2n+1))H \geq n$, where C_{sim} is an absolute constant. At fixed H and $p = O(\log n)$, it yields $m = \Omega(n/\log n)$ in one layer, compared with $O(\log n)$ width in two.

1.1 Contributions

1. **Raw-to-token lower-bound transfer.** We minimize the distributed communication needed to materialize assigned token values, charge it to token construction, and compose the residual communication with an existing one-layer simulation. Transcript-public adaptive ownership extends the transfer to content-dependent boundaries. A subsequential tokenizer is charged for one state and one output counter at each raw ownership-run boundary.
2. **An exact fixed-chunk formula and conservation law.** On a full binary product domain, the distributed materialization cost of input-independent exact-substring chunks equals

combinatorial boundary load. Equal data-chunk length histograms conserve its aggregate over raw cuts, so unequal matched profiles must cross.

3. **Matched certificate allocation.** Every τ_S uses the same six active data-token types, exactly $2n$ data positions followed by the same public final token, and the same per-input length histogram. The paired lower-bound numerators are $n - |S|$ and $|S|$, whose sum is n. Endpoint reversal, two-layer repair, and an invariant control complete the separation.
4. **Learned-model evidence under a prespecified paired protocol.** With data, parameter counts, initialization, and optimization shared, one-layer models on the forward task reach near-perfect in-distribution accuracy under the aligned tokenizer and stay at chance under the misaligned one for $n \geq 64$. At width 32, the preferred tokenizer flips with the task and the token-local control shows no gap; a second layer closes the gap at $n = 16$ but not at $n = 64$.

2 Related work

Formal tokenizer theory studies encoder-decoder exactness, consistency, finiteness, and sequentiality (Gastaldi et al., 2025). Learned dictionaries can approach Markov-source entropy, yet the encoder still matters for a fixed dictionary (Rajaraman et al., 2024), and exact marginalization can recover the induced byte process (Phan et al., 2025). BPE has known optimization limits (Zouhar et al., 2023b; Whittington et al., 2025). Rényi efficiency is not a universal proxy for downstream behavior (Zouhar et al., 2023a; Cognetta et al., 2024). Practical BPE has a constant-memory left-to-right implementation (Berglund & van der Merwe, 2023). The extended 2025 version of Berglund et al. (2025) constructs an input-deterministic subsequential transducer for proper BPE dictionaries, and canonical BPE and MaxMatch admit finite-state representations (Cognetta & Okazaki, 2025). Extended byte-BPE also preserves regular and context-free language classes (Geng et al., 2024).

Lossless encodings can differ under bounded predictors. Fragmentation can raise finite-context log-loss (Jafari Fesharaki et al., 2026), and reversible representations can differ in predictive codelength (Wang, 2026). Noncanonical, adversarial, and merge-list-free encodings hold the text fixed while changing its representation (Geh et al., 2024; 2025; Sawada & Goyal, 2025; Jain et al., 2026). Controlled studies also find effects from segmentation, grouping direction, atomic alignment, and tokenizer choice (Schmidt et al., 2024; Singh & Strouse, 2024; Zhang et al., 2025; Altıntaş et al., 2026). We ask for a shallow-model separation under stricter per-instance matching and an explicit raw-to-token charge.

Expressivity work varies embedding rank, positions, or generated computation (Wies et al., 2021; London & Kanade, 2025; Merrill & Sabharwal, 2023; 2024; Chen et al., 2024). We hold all three fixed. Our network-side engine is the arbitrary-token-partition simulation of Bhattamishra et al. (2024). Related results use an adjacent-pair hint for two-layer triple detection (Sanford et al., 2023) and communication reductions for one-layer function composition (Peng et al., 2024). For large outputs, Fontes et al. (2023) permit a split across the parties. Our adaptive cost is a transcript-public, token-atomic specialization, not a new output model or state-handoff principle.

Most closely, Bechler-Speicher et al. (2026) derive task-dependent depth regimes for graph tokenizations and reconstruct tokens that depend on both parties through communication. We do not claim the first communication analysis of tokenization. Our narrower contribution is a minimized raw-to-token charge, its exact fixed-chunk boundary-load value, and complementary certificates under matching of the raw string, active vocabulary, positions, token-length histogram, embedding-rank allowance, and model class.

Boundary load has the form of a cardinality-based small-side hypergraph splitting function, an established combinatorial object (Veldt et al., 2022). We do not claim that cut function or its elementary averaging identity in isolation. The tokenizer-specific contribution is the exact identification with materialization communication and the resulting matched-encoder comparison.

3 FORMAL SETUP

3.1 RAW-STRING FUNCTIONS AND TOKENIZERS

Let $X \subseteq \Sigma^L$ be finite, let $F : X \to \{0, 1\}$ be defined before tokenization, and let the deterministic tokenizer be $\tau : X \to V^N$. We assume losslessness, so concatenating the represented token strings reconstructs the raw input. A common public final token may represent the empty string. It counts toward N, but adds neither materialization communication nor an unmatched position.

The active vocabulary contains the token types attained anywhere on X. Our comparisons match that set, output length and token-length histogram on every input, a public token-embedding rank allowance, and the downstream model class.

3.2 TRANSFORMER MODEL

We use a causal restriction of the finite-precision softmax Transformer of Bhattamishra et al. (2024): H-head attention, a position-wise ReLU network, width m, public positional embeddings, p-bit arithmetic, and output at a common final token. The constructions use no residuals or normalization; the one-layer lower bounds also hold with residual connections and position-wise normalization (Appendix A.1). Width consequences assume $H = O(1)$, while the general lower bound applies to mH.

Non-uniform parameters may depend on N, not on input values. We set the source-grid constant to $K_c = 12$, preserving $O(\log N)$ precision. The source simulation allows arbitrary token-position partitions. A public constant-one coordinate supplies the affine shifts used by the constructions. Our final token attends to every earlier position, so its one-layer protocol bound applies. Throughout, C_{sim} denotes the absolute constant in that protocol bound; Theorem 4 of the source states it as $C_{\mathrm{sim}} = 3$, and only its constancy matters here.

3.3 DISTRIBUTED MATERIALIZATION COST

Fix a bipartition $P = (A, B)$ of the raw coordinates. Alice receives x_A, Bob receives x_B, and both know the tokenizer and model, while $Q = (Q_A, Q_B)$ fixes a partition of the N token positions. Define $\mathrm{ECC}_\tau(P; Q)$ as the minimum worst-case number of bits communicated by a deterministic protocol until Alice knows the token values in Q_A and Bob knows those in Q_B. Minimizing over token ownership gives

$$\mathrm{ECC}_\tau(P) = \min_Q \mathrm{ECC}_\tau(P; Q).$$

For content-dependent boundaries, a fixed Q can be unnecessarily restrictive. Define $\mathrm{AECC}_\tau(P)$ by allowing the transcript to determine a public ownership mask $Q(t)$. At termination, Alice must know every token in $Q_A(t)$, and Bob must know every token in $Q_B(t)$. Fixed ownership is a special case, so $\mathrm{AECC}_\tau(P) \le \mathrm{ECC}_\tau(P)$. Adaptive ownership is not cosmetic: Appendix A.5.1 gives a three-state, two-run, fixed-length tokenizer for which every fixed mask costs $\Omega(N)$, while the transcript-public state-and-count protocol costs $O(\log N)$.

The fixed-ownership quantity is the minimum communication needed to materialize assigned token values in a distributed thought experiment. It is a proof quantity, not wall-clock tokenization cost. A partition is *τ-separable* when $\mathrm{ECC}_\tau(P) = 0$. Such a cut permits fully local token ownership because each party can construct its assigned tokens without communication.

Let $\mathrm{CC}_P(F)$ be the deterministic communication complexity of the raw function under P. We use the one-output convention with a designated party producing F. Requiring both parties to output the value changes the cost by at most one bit. After charging token materialization, define the residual one-layer lower-bound certificate

$$\mathrm{RCC}_\tau(F) = \max_P \left[\mathrm{CC}_P(F) - \mathrm{ECC}_\tau(P)\right]_+ .$$

4 RAW-TO-TOKEN LOWER-BOUND TRANSFER

Theorem 1 (Raw-to-token lower-bound transfer)**.** *If a one-layer finite-precision Transformer computes $F(x)$ from $\tau(x)$ for every $x \in X$, then, for each raw partition P,*

$$[\mathrm{CC}_P(F) - \mathrm{ECC}_\tau(P)]_+ \le [\mathrm{CC}_P(F) - \mathrm{AECC}_\tau(P)]_+ \le C_{\mathrm{sim}}\, m(p + \log N)H.$$

Consequently,

$$\mathrm{RCC}_\tau(F) \le C_{\mathrm{sim}}\, m(p + \log N)H.$$

Run an $\mathrm{AECC}_\tau(P)$-optimal protocol, then select the arbitrary-token-partition simulation of Bhattamishra et al. (2024) from its public ownership mask. Appendix A.3.1 gives the full composition. The bound charges distributed token materialization, not centralized encoder latency, and does not characterize exact minimum model width. When P is τ-separable, the raw communication lower bound transfers unchanged.

4.1 EXACT BOUNDARY LOAD FOR FIXED CHUNK TOKENIZERS

We next specialize the encoding cost to fixed chunk tokenizers. Public, input-independent boundaries divide a binary string into M consecutive data chunks $C_1, \ldots, C_M$, and each data token is the exact substring in its chunk. A shared public final token adds one zero-cost position, so the model length may be $N = M + 1$.

For a raw partition $P = (A, B)$, define

$$\mathrm{BL}_\tau(P) = \sum_{j=1}^{M} \min\{|C_j \cap A|, |C_j \cap B|\}.$$

Theorem 2 (Chunk-cut formula)**.** *On the full domain $\{0,1\}^L$,*

$$\mathrm{AECC}_\tau(P) = \mathrm{ECC}_\tau(P) = \mathrm{BL}_\tau(P).$$

Consequently, every one-layer Transformer computing F from $\tau(x)$ satisfies

$$[\mathrm{CC}_P(F) - \mathrm{BL}_\tau(P)]_+ \le C_{\mathrm{sim}}\, m(p + \log N)H$$

for every raw partition P.

Assigning each data chunk to the party missing fewer coordinates attains $\mathrm{BL}_\tau(P)$. For adaptive ownership, each transcript leaf fixes a public mask. On the full product domain, its exact-substring outputs fix at least $\mathrm{BL}_\tau(P)$ missing bits, and the deterministic rectangle property gives the matching lower bound. Appendix A.4.1 gives the complete proof. Correlations or compressive token maps can break this equality, although the raw-to-token transfer remains valid.

Appendix A.3.2 records two further consequences as Corollaries 1 and 2: separable cuts transfer raw lower bounds unchanged, and boundary load gives a computable lower bound for chunk tokenizers. Averaging over cuts gives a third.

Corollary 3 (Length-histogram conservation)**.** *Let $\ell_j = |C_j|$ be the lengths of the nonpublic data chunks and define*

$$\mu(q) = 2^{-q} \sum_{k=0}^{q} \binom{q}{k} \min\{k, q-k\}.$$

On the full domain $\{0,1\}^L$, for labelled raw-coordinate assignments $A \subseteq [L]$,

$$2^{-L} \sum_{A \subseteq [L]} \mathrm{AECC}_\tau(A, A^c) = 2^{-L} \sum_{A \subseteq [L]} \mathrm{ECC}_\tau(A, A^c) = \sum_{j=1}^{M} \mu(\ell_j).$$

Hence two fixed chunk tokenizers with the same data-chunk length multiset have equal mean materialization cost. If their cut profiles differ, each has strictly lower cost on at least one cut.

Appendix A.4.3 strengthens the identity to every fixed cut size $|A| = r$ and gives a closed form for μ. For contiguous chunks it also sums the profile over every prefix cut. These are materialization-communication statements, not claims about runtime, accuracy, or minimum width.

4.2 A FINITE-STATE INTERFACE FOR CONTENT-DEPENDENT BOUNDARIES

Proposition 1 (Subsequential tokenizer interface). *Suppose $L, N, q \geq 1$ and $\tau : X \to V^N$ is computed by a deterministic subsequential transducer with q states and atomic token-ID outputs. If $r(P)$ is the number of maximal raw-coordinate ownership runs, then*

$$\mathrm{AECC}_\tau(P) \leq (r(P) - 1)\left(\lceil \log_2 q \rceil + \lceil \log_2(N+1) \rceil\right).$$

Consequently, every one-layer Transformer computing F from $\tau(x)$ satisfies

$$\left[\mathrm{CC}_P(F) - (r(P) - 1)\left(\lceil \log_2 q \rceil + \lceil \log_2(N+1) \rceil\right)\right]_+ \leq C_{\mathrm{sim}}\, m(p + \log N)H.$$

At each ownership-run boundary, the current party sends the transducer state and cumulative output count. Those public messages determine token intervals owned by the parties that emitted them. Appendix A.5.1 handles empty and multi-token transitions and terminal output. This proposition specializes split-output communication (Fontes et al., 2023). It applies only to a fixed-N slice with atomic outputs. The state count can also be large. Prior work supplies the finite-state tokenizer representation (Berglund et al., 2025; Cognetta & Okazaki, 2025). The contribution here is its materialization accounting and Transformer composition.

5 EXACT ALLOCATION OF LOWER-BOUND CERTIFICATES AND ENDPOINT TRADEOFF

5.1 CONSTRUCTION

Let

$$\Sigma = \{0,1\}, \qquad x = a_1 b_1 c_1 \cdots a_n b_n c_n \in \{0,1\}^{3n}, \qquad V_{\mathrm{data}} = \{0, 1, 00, 01, 10, 11\}.$$

The two endpoint tokenizers place their boundary at different locations within each three-bit block:

$$\tau_{AB}(x) = (a_1 b_1, c_1, \ldots, a_n b_n, c_n),$$

$$\tau_{BC}(x) = (a_1, b_1 c_1, \ldots, a_n, b_n c_n).$$

For any public $S \subseteq [n]$, let τ_S use $[a_i b_i][c_i]$ on blocks $i \in S$ and $[a_i][b_i c_i]$ on blocks $i \notin S$. Thus $\tau_{[n]} = \tau_{AB}$ and $\tau_\varnothing = \tau_{BC}$.

Across the domain, every τ_S uses every member of V_{data}, produces exactly $2n$ data tokens, and emits one singleton and one doubleton per block. Appending the same public final token gives every member model-input length $N = 2n + 1$, the same number of positions, and the same lookup-table allowance.

Table 3 in Appendix A records the full matched resource contract. Only the adjacent raw coordinates represented by one token change.

Consider the functions

$$F_n^{AB}(x) = \bigvee_i (a_i \wedge b_i), \qquad F_n^{BC}(x) = \bigvee_i (b_i \wedge c_i), \qquad F_n^{A}(x) = \bigvee_i a_i.$$

The task F_n^{AB} aligns with τ_{AB}, while F_n^{BC} aligns with τ_{BC}. The control task F_n^A is token-local under both.

Each encoder supports an entire label class. After fixing τ_{AB}, the same local construction handles $\bigvee_i g(a_i, b_i)$ for any Boolean g chosen by the model. The tokenizer τ_{BC} supports the symmetric (b_i, c_i) class, and conjunction supplies the communication lower bound used here.

Theorem 3 (Allocation of lower-bound certificates and endpoint tradeoff). *In the causal model of Section 3.2 with $p = O(\log n)$:*

1. *for every $S \subseteq [n]$, a one-layer model computing F_n^{AB} under τ_S satisfies $C_{\mathrm{sim}}\, m(p+\log(2n+1))H \geq n - |S|$, while one computing F_n^{BC} satisfies $C_{\mathrm{sim}}\, m(p + \log(2n+1))H \geq |S|$;*
2. *one layer and constant width compute F_n^{AB} under τ_{AB}, whereas every one-layer model under τ_{BC} satisfies $C_{\mathrm{sim}}\, m(p + \log(2n+1))H \geq n$;*
3. *two layers, two heads, and width $O(\log n)$ compute F_n^{AB} under τ_{BC};*
4. *swapping AB and BC reverses the endpoint separation; and*
5. *one layer and constant width compute F_n^{A} under both endpoint tokenizers.*

5.2 Certificate allocation

Let P_{AB} give Alice a and Bob (b, c), and let P_{BC} give Alice (a, b) and Bob c. The chunk-cut formula gives

$$\mathrm{ECC}_{\tau_S}(P_{AB}) = |S|, \qquad \mathrm{ECC}_{\tau_S}(P_{BC}) = n - |S|.$$

Restricting the nuisance string reduces the corresponding task to n-bit set non-disjointness. Theorem 1 therefore gives the two certificate numerators $n - |S|$ and $|S|$. Their sum is n. Each moved boundary transfers one unit between the designated certificates while all scalar tokenizer budgets remain fixed. This identity does not claim that exact minimum widths follow the same curve. Figure 2 in Appendix A plots the complementary allocation.

5.3 Endpoint bounds and depth repair

At either misaligned endpoint, fixing $c = 0^n$ or $a = 0^n$ reduces the task to set non-disjointness on a tokenizer-separable cut, yielding $C_{\mathrm{sim}}\, m(p + \log(2n+1))H \geq n$. At an aligned endpoint, marking token `11` and uniformly aggregating the markers computes the task with one head and constant width on the fixed-point grid. For a misaligned input, a two-head first layer uses public $O(\log n)$-dimensional sign codes to retrieve the predecessor and current bits, writes their conjunction at the appropriate positions, and a second layer aggregates the markers. Appendix A.8–A.9 gives the exact finite-precision constructions, including the reverse task and invariant control.

6 Learned one-layer models

The theorems concern exact computation, so whether gradient-trained finite-width models move in the same direction is a separate question. We prespecified a paired protocol (Appendix B). Within a pair, raw examples, tokenizer budgets, parameter counts, initialization, minibatch order, optimizer, and learning rate are shared, and only the grouping changes. Each model receives 2,000 AdamW updates and is evaluated at its final step. Negative training examples have disjoint a and b supports, and positive examples share exactly one index; every support has size $n/4$, so support size does not reveal the label. The shifted OOD split changes the support weights and admits multiple intersecting indices. Learning rates were selected per width and depth on excluded pilot data before any reported run.

Table 1 reports all 18 completed pairs of the 162-pair coarse registry (Appendix B): the forward task F_n^{AB} at depth one, $n \in \{16, 64, 128\}$, widths $m \in \{8, 32, 128\}$, and two model seeds, for 18 pairs and 36 trained models. Under the aligned τ_{AB}, 16 of 18 models classify every ID test example correctly, and the other two reach 97.3% and 99.9% balanced accuracy. Under the misaligned τ_{BC}, every model with $n \geq 64$ stays between 49.3% and 50.9% on the ID test split and between 48.7% and 51.7% on the OOD split, including models of width 128. At $n = 16$, only the wider τ_{BC} models rise above chance, to at most 67.4% ID and 76.9% OOD balanced accuracy. On the shifted OOD split, two aligned models also degrade, to 78.2% and 72.9%; each misclassifies every example in at least one OOD cell.

Table 1 covers one task at one depth. At these sizes, with the source's constant and 32-bit arithmetic, the exact lower bound excludes none of the widths used, so the τ_{BC} failures concern trainability under a fixed budget, not measured minimum widths. Monotone finite-width curves are a stronger hypothesis than Theorem 3; if they failed, the exact-computation bounds would be unaffected.

Reversal, depth, and control. The comparisons that separate a task-dependent effect of boundary placement from a tokenizer that is uniformly harder to train are the reverse task, the invariant

Table 1: **Depth-one models trained on F_n^{AB}: all 18 completed coarse pairs.** Balanced accuracy (%) for model seeds 1 / 2, truncated to one decimal, so 100.0 denotes a perfect score. Each pair shares data, initialization, minibatch order, parameter count, and learning rate. The OOD split shifts support weights and allows multiple intersecting indices. Chance is 50%.

		ID test			Shifted OOD		
n	tokenizer	$m = 8$	$m = 32$	$m = 128$	$m = 8$	$m = 32$	$m = 128$
16	τ_{AB}	100.0 / 100.0	100.0 / 100.0	100.0 / 100.0	99.2 / 100.0	78.2 / 100.0	100.0 / 100.0
	τ_{BC}	49.9 / 50.4	56.0 / 60.0	67.1 / 67.3	48.3 / 52.7	61.7 / 67.4	75.1 / 76.8
64	τ_{AB}	100.0 / 100.0	100.0 / 100.0	100.0 / 100.0	99.9 / 100.0	100.0 / 99.2	100.0 / 100.0
	τ_{BC}	50.2 / 49.8	50.0 / 50.0	50.0 / 50.8	50.1 / 49.9	50.3 / 50.5	50.2 / 51.6
128	τ_{AB}	97.3 / 99.9	100.0 / 100.0	100.0 / 100.0	72.9 / 99.9	100.0 / 99.9	100.0 / 100.0
	τ_{BC}	49.8 / 50.0	49.9 / 49.8	49.3 / 49.3	50.4 / 49.8	50.3 / 49.6	49.6 / 48.7

Table 2: **Reverse task, control, and depth two (width 32).** Balanced accuracy (%) for seeds 1 / 2, truncated to one decimal. Registry settings executed on CPU rather than the registered T4. F_n^{BC} is aligned with τ_{BC}; F_n^A is token-local under both tokenizers and has no OOD split.

		ID test		Shifted OOD	
task	n	τ_{AB}	τ_{BC}	τ_{AB}	τ_{BC}
F_n^{BC}, depth 1	16	56.8 / 55.4	100.0 / 100.0	61.7 / 61.7	81.8 / 86.5
	64	49.9 / 49.9	100.0 / 100.0	49.4 / 49.7	100.0 / 84.6
F_n^A, depth 1	16	100.0 / 100.0	100.0 / 100.0	– / –	– / –
	64	100.0 / 100.0	100.0 / 100.0	– / –	– / –
F_n^{AB}, depth 2	16	100.0 / 100.0	100.0 / 100.0	77.4 / 77.0	100.0 / 99.8
	64	100.0 / 100.0	50.0 / 49.9	98.0 / 66.6	50.5 / 51.2

control, and depth two. We ran twelve further pairs from the same registry at width 32, seeds 1–2, and $n \in \{16, 64\}$: F_n^{BC} and F_n^A at depth one, and F_n^{AB} at depth two (Table 2). Data, seeds, and hyperparameters are the registry's. These runs used local CPUs instead of the registered T4 accelerator, so they are supplementary to the frozen protocol. The reversal and the control behave as predicted at both sizes. On the reverse task, τ_{BC} models reach 100% ID balanced accuracy, while τ_{AB} models reach 55.4–56.9% at $n = 16$ and 49.9–50.0% at $n = 64$. The control F_n^A reaches 100% under both tokenizers. Depth repair holds only at the smaller size. At $n = 16$, depth-two τ_{BC} models reach 100% on F_n^{AB}, compared with 56.0–60.1% at depth one (Table 1). At $n = 64$ they stay at chance (49.9–50.1%), although Theorem 3 gives an exact two-layer solution of width $O(\log n)$; by the protocol's own criterion, the learned depth-repair prediction is not supported at $n = 64$ in these runs. Out of distribution, aligned reverse-task models reach 81.8–100% against 49.4–61.8% for misaligned ones, and aligned depth-two models reach 66.6–98.1%. The remaining registry pairs, including the third seed and $n = 128$, have not been run; we report every completed run.

A deployed tokenizer. GPT-4's cl100k_base tokenizer chunks digit runs left to right into groups of at most three (L2R). Comma separators force right-to-left grouping (R2L), which Singh & Strouse (2024) found substantially improves GPT-3.5 and GPT-4 addition accuracy. Both chunkings share a length multiset, so by Corollary 3 their mean boundary loads are equal; exact enumeration ($L \leq 12$) confirms this. They differ in where the free cuts fall. Under R2L, the place-value cut at 10^k is free exactly when $3 \mid k$, for every length. Under L2R, two operands share free cuts only when their lengths agree modulo 3 (Appendix A.12). We report this correspondence, not a cause.

7 Scope and Limitations

Theorem 1 assumes deterministic lossless tokenization and fixed output length. Proposition 1 covers subsequential tokenizers with atomic outputs. Theorem 2 and Corollary 3 are exact only for input-independent chunks on a full product domain, and Theorem 3 uses the matched τ_S family. The

networks use exact finite-precision attention plus ReLU; the explicit constructions omit residuals and normalization. A separate typed synthetic family admits canonical-BPE endpoint realizations with prescribed four-rule dictionaries on a publicly typed structured domain (Appendix A.11). The construction matches vocabulary cardinalities rather than represented-string vocabularies and does not establish a realization of the interior-S family or characterize learned or natural-language BPE behavior.

The proposition gives no practical-tokenizer separation. A BPE or MaxMatch application must verify its complete pipeline, deterministic subsequentiality, atomic outputs, and fixed length. Correlation or compression can lower exact-chunk cost, and a large state space can weaken the finite-state bound. Materialization measures information, not runtime. The network simulation and disjointness bound are prior work, and the constant C_{sim} is inherited from it. All lower bounds concern exact computation; our claims do not cover learning, optimization, language quality, or tokenizer rankings.

8 Conclusion

Matching vocabulary size, sequence length, and token-length histograms does not make two tokenizers computationally equivalent for a shallow Transformer. For fixed exact-substring chunks on a full product domain, the tokenizer's contribution to a one-layer lower bound is exactly its boundary load across the raw cut, and matched length histograms conserve the average load, so matched tokenizers trade cost across cuts rather than dominate one another. Within τ_S, each moved boundary transfers one unit between certificates $n-|S|$ and $|S|$. At fixed head count and $O(\log n)$-bit precision, alignment separates constant one-layer width under the aligned tokenizer from an $\Omega(n/\log n)$ one-layer lower bound under the misaligned one, and a second layer restores $O(\log n)$ width. The typed-BPE corollary (Appendix A.11) shows that canonical BPE can realize both endpoints on a structured synthetic domain.

These results suggest reporting which raw interactions a tokenizer makes token-local, alongside its compression. Open questions include the exact minimum widths along the interior of τ_S, whether learned finite-width models track the certificate allocation, and whether boundary-load profiles of learned tokenizers on natural text predict which interactions shallow models find hard.

Reproducibility statement

Appendix A states every assumption and gives complete proofs and explicit finite-precision constructions. Appendix B specifies the data generator, run grid, learning-rate selection, metrics, and failure criteria. Code, data generators, and a Colab notebook are available at `https://github.com/lifebecrazy/token-boundaries`. The supplement contains the locked environment, audited datasets, checksum-frozen pilot and coarse registries, selected learning rates, runner, and tests, including exact-arithmetic checks of the boundary-load formula, the conservation identity, and the finite-precision steps of the constructions on small instances. Smoke tests and CPU or MPS benchmarks are excluded from scientific selection.

AI use statement

Large language model assistants (OpenAI Codex and Anthropic Claude) were used for literature search and summarization; for developing the research question, theoretical claims, proofs, hypotheses, and methodology; for drafting and editing proofs, code, figures, and text; for generating synthetic data; and for building deterministic executable checks of the constructions. They were not used for human-subject data, translation, or qualitative analysis. Citations were checked against primary sources. The authors take full responsibility for the final text, claims, proofs, code, data, figures, and artifacts.

References

Sawsan Alqahtani, Mir Tafseer Nayeem, Md Tahmid Rahman Laskar, Tasnim Mohiuddin, and M Saiful Bari. Stop taking tokenizers for granted: They are core design decisions in large

language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8410–8432. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.eacl-long.394. URL https://aclanthology.org/2026.eacl-long.394/.

Gül Sena Altıntaş, Malikeh Ehghaghi, Brian Lester, Fengyuan Liu, Wanru Zhao, Marco Ciccone, and Colin Raffel. TokSuite: Measuring the impact of tokenizer choice on language model behavior. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2512.20757.

Maya Bechler-Speicher, Gilad Yehudai, Gil Harari, Clayton Sanford, Amir Globerson, and Joan Bruna. Lost in tokenization: Fundamental trade-offs in graph tokenization for transformers, 2026. URL https://arxiv.org/abs/2605.22471.

Martin Berglund and Brink van der Merwe. Formalizing BPE tokenization. *Electronic Proceedings in Theoretical Computer Science*, 388:16–27, 2023. doi: 10.4204/EPTCS.388.4. URL https://doi.org/10.4204/EPTCS.388.4.

Martin Berglund, Willeke Martens, and Brink van der Merwe. Constructing a BPE tokenization DFA, 2025. URL https://arxiv.org/abs/2405.07671v2. Revised and extended version of the CIAA 2024 paper.

Satwik Bhattamishra, Michael Hahn, Phil Blunsom, and Varun Kanade. Separations in the representational capabilities of transformers and recurrent architectures. In *Advances in Neural Information Processing Systems 37*, pp. 36002–36045, 2024. doi: 10.52202/079017-1135. URL https://doi.org/10.52202/079017-1135.

Lijie Chen, Binghui Peng, and Hongxun Wu. Theoretical limitations of multi-layer transformer, 2024. URL https://arxiv.org/abs/2412.02975.

Marco Cognetta and Naoaki Okazaki. Tokenization as finite-state transduction. *Computational Linguistics*, 51(4):1119–1149, 2025. doi: 10.1162/coli.a.23. URL https://aclanthology.org/2025.cl-4.2/.

Marco Cognetta, Vilém Zouhar, Sangwhan Moon, and Naoaki Okazaki. Two counterexamples to tokenization and the noiseless channel. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 16897–16906. ELRA and ICCL, 2024. doi: 10.63317/44odcnu7ofdt. URL https://aclanthology.org/2024.lrec-main.1469/.

Lila Fontes, Sophie Laplante, Mathieu Laurière, and Alexandre Nolin. The communication complexity of functions with large outputs. In *Structural Information and Communication Complexity*, pp. 427–458. Springer Nature Switzerland, 2023. doi: 10.1007/978-3-031-32733-9_19. URL https://doi.org/10.1007/978-3-031-32733-9_19.

Juan Luis Gastaldi, John Terilla, Luca Malagutti, Brian DuSell, Tim Vieira, and Ryan Cotterell. The foundations of tokenization: Statistical and computational concerns. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=B5iOSxM2I0.

Renato Geh, Honghua Zhang, Kareem Ahmed, Benjie Wang, and Guy Van Den Broeck. Where is the signal in tokenization space? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3966–3979. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.230. URL https://aclanthology.org/2024.emnlp-main.230/.

Renato Geh, Zilei Shao, and Guy Van Den Broeck. Adversarial tokenization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 20738–20765. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1012. URL https://aclanthology.org/2025.acl-long.1012/.

Saibo Geng, Sankalp Gambhir, Chris Wendler, and Robert West. Byte BPE tokenization as an inverse string homomorphism, 2024. URL https://arxiv.org/abs/2412.03160.

Amirmehdi Jafari Fesharaki, Mohammadamin Rami, and Aslan Tchamkerten. Effective context in transformers: An analysis of fragmentation and tokenization, 2026. URL https://arxiv.org/abs/2605.13485.

Kanishk Jain, Matthew Day, and Tankut Can. Emergent retokenization symmetry in large language models: Phenomenology and applications, 2026. URL https://arxiv.org/abs/2606.15521.

Charles London and Varun Kanade. Pause tokens strictly increase the expressivity of constant-depth transformers. In *Advances in Neural Information Processing Systems 38*, pp. 99681–99711, 2025. doi: 10.52202/085713-3001. URL https://doi.org/10.52202/085713-3001.

William Merrill and Ashish Sabharwal. The parallelism tradeoff: Limitations of log-precision transformers. *Transactions of the Association for Computational Linguistics*, 11:531–545, 2023. doi: 10.1162/tacl_a_00562. URL https://aclanthology.org/2023.tacl-1.31/.

William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NjNGlPh8Wh.

Binghui Peng, Srini Narayanan, and Christos Papadimitriou. On limitations of the transformer architecture, 2024. URL https://arxiv.org/abs/2402.08164.

Buu Phan, Brandon Amos, Itai Gat, Marton Havasi, Matthew Muckley, and Karen Ullrich. Exact byte-level probabilities from tokenized language models for FIM-tasks and model ensembles. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=zGej22CBnS.

Nived Rajaraman, Jiantao Jiao, and Kannan Ramchandran. An analysis of tokenization: Transformers under markov data. In *Advances in Neural Information Processing Systems 37*, pp. 62503–62556, 2024. doi: 10.52202/079017-1999. URL https://doi.org/10.52202/079017-1999.

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. In *Advances in Neural Information Processing Systems 36*, pp. 36677–36707, 2023. doi: 10.52202/075280-1596. URL https://doi.org/10.52202/075280-1596.

Tomohiro Sawada and Kartik Goyal. Train it and forget it: Merge lists are unnecessary for BPE inference in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35045–35058. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1775. URL https://aclanthology.org/2025.emnlp-main.1775/.

Craig W. Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. Tokenization is more than compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 678–702. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.40. URL https://aclanthology.org/2024.emnlp-main.40/.

Aaditya K. Singh and DJ Strouse. Tokenization counts: The impact of tokenization on arithmetic in frontier LLMs, 2024. URL https://arxiv.org/abs/2402.14903.

Nate Veldt, Austin R. Benson, and Jon Kleinberg. Hypergraph cuts with general splitting functions. *SIAM Review*, 64(3):650–685, 2022. doi: 10.1137/20M1321048. URL https://doi.org/10.1137/20M1321048.

Yi Wang. How far should tokenization go? predictive effectiveness and relational losslessness, 2026. URL https://arxiv.org/abs/2608.18025.

Philip Whittington, Gregor Bachmann, and Tiago Pimentel. Tokenisation is NP-complete. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28133–28153. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1365. URL https://aclanthology.org/2025.acl-long.1365/.

Noam Wies, Yoav Levine, Daniel Jannai, and Amnon Shashua. Which transformer architecture fits my data? a vocabulary bottleneck in self-attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11170–11181. PMLR, 2021. URL https://proceedings.mlr.press/v139/wies21a.html.

Xiang Zhang, Juntai Cao, Jiaqi Wei, Yiwei Xu, and Chenyu You. Tokenization constraints in LLMs: A study of symbolic and arithmetic reasoning limits, 2025. URL https://arxiv.org/abs/2505.14178.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. Tokenization and the noiseless channel. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5184–5207. Association for Computational Linguistics, 2023a. doi: 10.18653/v1/2023.acl-long.284. URL https://aclanthology.org/2023.acl-long.284/.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Tim Vieira, Mrinmaya Sachan, and Ryan Cotterell. A formal perspective on byte-pair encoding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 598–614. Association for Computational Linguistics, 2023b. doi: 10.18653/v1/2023.findings-acl.38. URL https://aclanthology.org/2023.findings-acl.38/.

A Complete proofs and constructions

A.1 Model interface

Let $X \subseteq \Sigma^L$ be a finite raw-input domain and $F : X \to \{0,1\}$ a raw-string function. A tokenizer is a deterministic lossless map $\tau : X \to V^N$ of fixed output length N. A common fixed final token may be included in N; because its value is public, either party may own its position.

Let $T \in \mathrm{TF}^1_{m,p,H}$ be the one-layer finite-precision Transformer class analyzed by Bhattamishra et al.: width m, p-bit parameters and intermediate values, H attention heads, arbitrary token/position embeddings, an attention block, and a position-wise feed-forward block. The output is read from the final position. Their Theorem 4 applies to an arbitrary partition of token positions and gives a deterministic two-party protocol of cost at most

$$B_T(N) = C_{\mathrm{sim}}\, m(p + \log N)H$$

bits for computing the Transformer's output. Here C_{sim} is an absolute constant; the source theorem states it as $C_{\mathrm{sim}} = 3$. We keep it symbolic because none of our conclusions depends on its value: every lower bound below is used only up to this constant factor.

We use the causal specialization of that attention-plus-feed-forward model, without residual connections or normalization. The final position can attend to every earlier position, so the same simulation bound applies. The lower bound therefore needs no stronger model restriction than the source theorem. The upper bounds below are constructed directly in this causal specialization. As usual for circuit-style expressivity results, we use a non-uniform family: parameters and public positional embeddings may depend on N, but never on the input values. Their magnitudes and the dimension of every embedding remain within the stated width and precision bounds. We reserve one public positional coordinate equal to 1 and route a constant-one value through attention. Linear maps can therefore realize the key shift $-r$ and the feed-forward thresholds -1 and -2 without an unstated bias term; the extra coordinate changes width only by a constant. We fix the source grid constant once at $K_c = 12$, so the representable range is $[-N^{12}, N^{12}]$, the step is $\Delta = N^{-12}$, and the grid has $2N^{24} + 1$ values, so $p = \lceil \log_2(2N^{24} + 1) \rceil = 24 \log_2 N + O(1) = O(\log N)$ bits index it. The particular value 12 is inessential; fixing it explicitly makes the small-N range check in the causal lookup construction auditable.

Remark (residual connections and normalization). The one-layer simulation uses only two properties of the network: each position's key and value vectors are functions of that position's token and public position, and the output is a function of the final position's attention output together with the final token and position. Residual connections around the attention and feed-forward blocks, and position-wise normalization before or after them, preserve both properties, because the final token and position are known to the party that owns the final position (to both parties when the final token is public). Let that party combine the exchanged attention sums; the other party's partial sums have the same format as its own, so the cost is unchanged. It can therefore apply these maps locally, and every lower bound below holds unchanged for one-layer models with residual connections and position-wise normalization. For the upper bounds, residual connections cost only a constant factor in width: each sublayer writes its output to fresh coordinates that are zero in its input, so the residual sum stores the input and the output without interference, and later maps read the coordinates they need. We state the explicit constructions without residuals and do not treat normalization in the upper bounds.

A.2 Distributed materialization cost

Fix a bipartition $P = (A, B)$ of the raw coordinates. Alice receives x_A, Bob receives x_B, and both know P, τ, T, and all public parameters.

For a fixed bipartition $Q = (Q_A, Q_B)$ of the N token positions, define $\mathrm{ECC}_\tau(P; Q)$ as the minimum worst-case number of bits communicated by a deterministic protocol after which Alice knows $\tau(x)_j$ for every $j \in Q_A$ and Bob knows $\tau(x)_j$ for every $j \in Q_B$. Define

$$\mathrm{ECC}_\tau(P) = \min_Q \mathrm{ECC}_\tau(P; Q).$$

For transcript-public adaptive ownership, a deterministic materialization protocol may determine a token-position partition $Q(t) = (Q_A(t), Q_B(t))$ from its transcript t. At termination, Alice must know $\tau(x)_j$ for every $j \in Q_A(t)$, and Bob must know $\tau(x)_j$ for every $j \in Q_B(t)$. Let $\mathrm{AECC}_\tau(P)$ be the minimum worst-case communication of such a protocol. Every fixed-Q protocol is admissible, so

$$\mathrm{AECC}_\tau(P) \le \mathrm{ECC}_\tau(P).$$

A raw partition P is τ**-separable** when $\mathrm{ECC}_\tau(P) = 0$. Equivalently, for some fixed token-position partition, every token owned by Alice is a function only of x_A, and every token owned by Bob is a function only of x_B.

Let $\mathrm{CC}_P(F)$ denote deterministic two-party communication complexity of F under the raw partition P. We use the one-output convention with a designated party producing F; requiring both parties to output the value changes the cost by at most one bit. Define

$$\mathrm{RCC}_\tau(F) = \max_P \left[\mathrm{CC}_P(F) - \mathrm{ECC}_\tau(P)\right]_+ .$$

This is a proof-side lower-bound certificate after charging token materialization, not a measure of centralized tokenizer runtime.

A.3 Raw-to-token lower-bound transfer

A.3.1 Theorem 1 (raw-to-token lower-bound transfer)

If $T(\tau(x)) = F(x)$ for all $x \in X$, then for every raw partition P,

$$\mathrm{CC}_P(F) \le \mathrm{AECC}_\tau(P) + C_{\mathrm{sim}}\, m(p + \log N)H \le \mathrm{ECC}_\tau(P) + C_{\mathrm{sim}}\, m(p + \log N)H.$$

Consequently,

$$\mathrm{RCC}_\tau(F) \le C_{\mathrm{sim}}\, m(p + \log N)H.$$

Proof Fix P, and execute an optimal transcript-public adaptive materialization protocol. At its end, the transcript determines $Q(t) = (Q_A(t), Q_B(t))$. Alice knows the token values at positions $Q_A(t)$, and Bob knows the values at positions $Q_B(t)$. Token embeddings, positional embeddings, and all projection weights are public, so each party can construct the exact embedded rows for the positions it owns.

For every possible Q, publicly fix the Bhattamishra et al. Theorem 4 simulation for that partition of token positions. Both parties select the same simulation from $Q(t)$. It communicates at most $C_{\mathrm{sim}}\, m(p + \log N)H$ bits and exactly reconstructs the output of the one-layer Transformer under the same finite-precision arithmetic. Because $T(\tau(x)) = F(x)$, the composed protocol computes F. Its total cost is at most

$$\mathrm{AECC}_\tau(P) + C_{\mathrm{sim}}\, m(p + \log N)H,$$

which upper-bounds $\mathrm{CC}_P(F)$. The fixed-ownership inequality follows from $\mathrm{AECC}_\tau(P) \le \mathrm{ECC}_\tau(P)$. Rearranging, taking the positive part, and maximizing over P proves the residual statement. □

A.3.2 COROLLARY 1 (ZERO-COST SEPARABLE CUTS)

Define

$$\mathrm{TCC}_\tau(F) = \max_{P:\,\mathrm{ECC}_\tau(P)=0} \mathrm{CC}_P(F).$$

Every one-layer Transformer computing F from $\tau(x)$ satisfies

$$\mathrm{TCC}_\tau(F) \le C_{\mathrm{sim}}\, m(p+\log N)H.$$

Proof Apply Theorem 1 to every τ-separable partition, where the materialization term is zero, then take the maximum. □

A.4 EXACT CUT FORMULA FOR FIXED CHUNK TOKENIZERS

A **fixed chunk tokenizer** on $\{0,1\}^L$ has M raw-dependent data-token positions, specified by a public ordered partition $C_1,\dots,C_M$ into consecutive intervals whose concatenation is $[L]$. Its j-th data token is the exact substring x_{C_j}. The chunks and their output positions do not depend on the input values. A common public final token may be appended as a data-independent position; then the model-input length is $N = M+1$ (otherwise $N = M$). The final position contributes zero to every materialization cost below.

For a raw partition $P = (A, B)$, define its tokenizer boundary load

$$\mathrm{BL}_\tau(P) = \sum_{j=1}^{M} \min\{|C_j \cap A|, |C_j \cap B|\}.$$

A.4.1 THEOREM 2 (CHUNK-CUT FORMULA)

For every fixed chunk tokenizer on the full binary domain and every raw partition P,

$$\mathrm{AECC}_\tau(P) = \mathrm{ECC}_\tau(P) = \mathrm{BL}_\tau(P).$$

More precisely, for a fixed token-position ownership partition $Q = (Q_A, Q_B)$,

$$\mathrm{ECC}_\tau(P;Q) = \sum_{j\in Q_A\cap[M]} |C_j\cap B| + \sum_{j\in Q_B\cap[M]} |C_j\cap A|.$$

Proof For the upper bound, send each raw bit that is missing from the owner of its data chunk. Alice sends $x_{C_j\cap A}$ to Bob for every $j \in Q_B\cap[M]$, and Bob sends $x_{C_j\cap B}$ to Alice for every $j\in Q_A\cap[M]$. Each owner can then form its exact substring token, and the communication equals the displayed sum.

For the matching lower bound, let u concatenate the Alice-owned bits appearing in Bob-owned data chunks and let v concatenate the Bob-owned bits appearing in Alice-owned data chunks. Fix every other raw bit and vary (u,v) over its full binary product domain. Given Bob's local input, the tuple of exact data tokens he must output is injective in u: those tokens contain every bit of u at a public coordinate. Thus Bob recovers u from his local input and transcript. Symmetrically, Alice recovers v.

Every transcript class of a deterministic protocol is a combinatorial rectangle $U_t \times V_t$. If U_t contained two values $u \neq u'$, then for any fixed $v \in V_t$, Bob would have the same local input and transcript but would need to output two different missing strings. Thus U_t is a singleton. Symmetrically, V_t is a singleton. The protocol therefore needs at least $2^{|u|+|v|}$ transcript leaves and at least $|u|+|v|$ worst-case communicated bits: a deterministic bit protocol with worst-case cost c is a binary tree of depth at most c and hence has at most 2^c leaves. This equals the upper bound.

Minimizing over fixed Q assigns each data chunk independently to the party missing fewer of its bits. The public final token can be assigned arbitrarily at zero cost. This gives $\mathrm{ECC}_\tau(P) = \mathrm{BL}_\tau(P)$.

It remains to lower-bound adaptive ownership. Consider a transcript leaf t of any deterministic adaptive protocol. Its inputs form a rectangle $U_t \times V_t$, and its transcript fixes a public ownership mask $Q(t)$. Let

$$s_t = \sum_{j \in Q_A(t) \cap [M]} |C_j \cap B| + \sum_{j \in Q_B(t) \cap [M]} |C_j \cap A|.$$

Correctness forces the missing coordinates of every Alice-owned exact-substring token to be fixed over V_t. It symmetrically forces the missing coordinates of every Bob-owned token to be fixed over U_t. Hence the leaf contains at most 2^{L-s_t} inputs. Every assignment has $s_t \geq \mathrm{BL}_\tau(P)$, so every leaf contains at most $2^{L-\mathrm{BL}_\tau(P)}$ inputs. A cost-c protocol has at most 2^c leaves and must cover 2^L inputs. Therefore $c \geq \mathrm{BL}_\tau(P)$. Together with $\mathrm{AECC}_\tau(P) \leq \mathrm{ECC}_\tau(P)$, this proves the three-way equality. □

A.4.2 Corollary 2 (computable lower bound for chunk tokenizers)

If a one-layer Transformer computes F from a fixed chunk tokenizer τ, then every raw partition P satisfies

$$[\mathrm{CC}_P(F) - \mathrm{BL}_\tau(P)]_+ \leq C_{\mathrm{sim}}\, m(p + \log N)H.$$

Thus the abstract materialization term in Theorem 1 becomes a combinatorial cut quantity for this tokenizer class. The equality requires the full product domain and exact substring tokens; it is an upper bound, but need not remain a lower bound, when the raw domain has correlations or the token map compresses chunk contents.

A.4.3 Corollary 3 (length-histogram conservation)

Let $\ell_j = |C_j| \geq 1$ be the lengths of the M raw-dependent data chunks. The public final token is excluded because it depends on no raw coordinate. Define

$$\mu(q) = 2^{-q} \sum_{k=0}^{q} \binom{q}{k} \min\{k, q-k\} = \frac{q}{2} - \frac{q}{2^q} \binom{q-1}{\lfloor (q-1)/2 \rfloor}.$$

For labelled raw-coordinate assignments $A \subseteq [L]$, so that A and A^c are counted separately,

$$2^{-L} \sum_{A \subseteq [L]} \mathrm{AECC}_\tau(A, A^c) = 2^{-L} \sum_{A \subseteq [L]} \mathrm{ECC}_\tau(A, A^c) = \sum_{j=1}^{M} \mu(\ell_j).$$

More strongly, for every $r \in \{0, \ldots, L\}$,

$$\sum_{\substack{A \subseteq [L] \\ |A| = r}} \mathrm{ECC}_\tau(A, A^c) = \sum_{j=1}^{M} \sum_{k=0}^{\ell_j} \binom{\ell_j}{k} \binom{L - \ell_j}{r-k} \min\{k, \ell_j - k\},$$

and the same identity holds with AECC_τ. We take an out-of-range binomial coefficient to be zero. For the contiguous chunks in Theorem 2, the prefix cuts satisfy

$$\sum_{t=1}^{L-1} \mathrm{ECC}_\tau([t], [L] \setminus [t]) = \sum_{t=1}^{L-1} \mathrm{AECC}_\tau([t], [L] \setminus [t]) = \sum_{j=1}^{M} \left\lfloor \frac{\ell_j^2}{4} \right\rfloor.$$

Consequently, two fixed chunk tokenizers with the same data-chunk length multiset have the same aggregate materialization cost over all labelled cuts, within every fixed cut size, and over the prefix family. If their profiles differ on one of these families, each tokenizer has strictly lower cost on at least one cut in that family.

Proof Theorem 2 lets us replace both materialization costs by boundary load. Fix a chunk C_j. Among the subsets A with $|A| = r$, exactly $\binom{\ell_j}{k}\binom{L-\ell_j}{r-k}$ have $|A \cap C_j| = k$. Multiplying this count by $\min\{k, \ell_j - k\}$, summing over k, and then summing over chunks proves the fixed-size identity. Without conditioning on $|A|$, exactly $\binom{\ell_j}{k}2^{L-\ell_j}$ labelled subsets have $|A \cap C_j| = k$. Division by 2^L gives $\sum_j \mu(\ell_j)$. The closed form follows from

$$\min\{k, q-k\} = \frac{q}{2} - \frac{|q-2k|}{2}$$

and the telescoping binomial identity

$$\sum_{k=0}^{q} \binom{q}{k} |q-2k| = 2q \binom{q-1}{\lfloor (q-1)/2 \rfloor}.$$

For a contiguous chunk of length ℓ, only the $\ell - 1$ prefix cuts inside that chunk contribute. Their loads are $\min\{k, \ell - k\}$ for $k = 1, \ldots, \ell - 1$, whose sum is s^2 when $\ell = 2s$ and $s(s+1)$ when $\ell = 2s+1$. Both cases equal $\lfloor \ell^2/4 \rfloor$. Finally, equal aggregates rule out weak pointwise dominance with a strict inequality: any strict saving must be offset by a strict increase. □

This corollary concerns deterministic materialization communication. It does not order tokenizer runtime, downstream accuracy, or minimum model width. The small-side summand is an established cardinality-based hypergraph splitting function; the tokenizer-specific content comes from its exact identification with AECC and ECC.

A.5 A FINITE-STATE INTERFACE FOR CONTENT-DEPENDENT BOUNDARIES

A.5.1 PROPOSITION 1 (SUBSEQUENTIAL TOKENIZER INTERFACE)

Suppose $L, N, q \geq 1$ and a deterministic subsequential transducer with q states computes

$$\tau(x) = \lambda(z_0, x_1) \cdots \lambda(z_{L-1}, x_L)\rho(z_L) \in V^N,$$

where each transition output $\lambda(z, a)$ lies in V^*, and the terminal output $\rho(z)$ lies in V^*. Thus each output symbol is one atomic token ID. Let $r(P)$ be the number of maximal monochromatic runs in the raw-coordinate ownership sequence. Then

$$\mathrm{AECC}_\tau(P) \leq (r(P) - 1)\left(\lceil \log_2 q \rceil + \lceil \log_2(N+1) \rceil\right).$$

Consequently, if a one-layer Transformer computes F from $\tau(x)$, then

$$[\mathrm{CC}_P(F) - (r(P) - 1)\left(\lceil \log_2 q \rceil + \lceil \log_2(N+1) \rceil\right)]_+ \leq C_{\mathrm{sim}}\, m(p + \log N)H.$$

Proof Simulate the transducer from left to right. The owner of the first raw run starts in the public initial state. At each of the $r(P) - 1$ ownership changes, the current simulator sends the next state using $\lceil \log_2 q \rceil$ bits and the cumulative number of emitted tokens using $\lceil \log_2(N+1) \rceil$ bits. The recipient continues from that state and count.

Assign every token emitted while processing a raw run to that run's owner. Include the terminal output $\rho(z_L)$ in the final run. The transmitted cumulative counts determine every internal token-interval boundary from the transcript, and the public fixed length N determines the final endpoint. Each party knows all atomic token IDs emitted during its own runs. The transcript therefore defines a valid public adaptive ownership mask and the displayed communication bound. This argument allows transitions that emit zero, one, or several tokens.

The Transformer inequality follows by substituting this materialization protocol into Theorem 1. □

The public output counter is essential for this proof. State handoff alone does not turn input-dependent emission positions into one fixed ownership set. The proposition also assumes a fixed output length and a deterministic subsequential transducer with atomic token-ID outputs. A generic finite-state relation or an unmodeled production preprocessing pipeline does not satisfy these premises automatically.

A fixed-mask counterexample. Let $M \geq 1$, let $\Gamma = \{\#, 0, 1\}$, and let X_M contain strings of $2M$ aligned two-symbol blocks. Each block belongs to

$$\{\#0, \#1, 00, 01, 10, 11\},$$

and exactly M blocks begin with $\#$. Define

$$\tau(\#b) = (\#b), \qquad \tau(bc) = (b, c), \qquad b, c \in \{0, 1\}.$$

Thus $V = \{\#0, \#1, 0, 1\}$, and every input produces $N = 3M$ atomic tokens.

Three states suffice. State E expects the first symbol of a block, S follows $\#$, and D follows a leading bit. From E, reading $\#$ emits nothing and enters S; reading $b \in \{0, 1\}$ emits b and enters D. From S, reading b emits $\#b$ and returns to E. From D, reading c emits c and returns to E. The terminal output is empty, and transitions outside the promise may be defined arbitrarily.

Partition the raw coordinates after the first M blocks. If Alice's prefix contains k hash- leading blocks, it emits $\ell = 2M - k$ tokens, exactly at positions $[\ell]$. Fix any token-position mask $Q = (Q_A, Q_B)$. After fixing a valid block-type layout and revealing it publicly, all $3M$ payload bits remain independent. Bob must learn the Alice bits at $[\ell] \setminus Q_A$, while Alice must learn the Bob bits at $Q_A \setminus [\ell]$. The same transcript-rectangle argument as in Theorem 2 gives

$$\mathrm{ECC}_\tau(P; Q) \geq |Q_A \triangle [\ell]|.$$

The domain contains a layout with $k = M$, hence $\ell = M$, and one with $k = 0$, hence $\ell = 2M$. The triangle inequality for symmetric difference therefore gives

$$\max\{|Q_A \triangle [M]|, |Q_A \triangle [2M]|\} \geq \left\lceil \frac{|[M] \triangle [2M]|}{2} \right\rceil = \lceil M/2 \rceil.$$

This holds for every fixed Q, so

$$\mathrm{ECC}_\tau(P) \geq \lceil M/2 \rceil = \Omega(N), \qquad q = 3, \quad r(P) = 2, \quad N = 3M.$$

By Proposition 1, the adaptive cost is at most $\lceil \log_2 3 \rceil + \lceil \log_2(N + 1) \rceil = O(\log N)$. Hence the fixed ownership set, not the transducer state, is the obstruction.

A.6 Proof of Theorem 3: a matched tokenizer continuum

Let $\Sigma = \{0, 1\}$, let $x = a_1 b_1 c_1 \cdots a_n b_n c_n \in \{0, 1\}^{3n}$, and let

$$V = \{0, 1, 00, 01, 10, 11\}.$$

Define

$$\tau_{AB}(x) = (a_1 b_1, c_1, \ldots, a_n b_n, c_n)$$

and

$$\tau_{BC}(x) = (a_1, b_1 c_1, \ldots, a_n, b_n c_n).$$

More generally, for every public subset $S \subseteq [n]$, define

$$\tau_S(x)_i = \begin{cases} (a_i b_i, c_i), & i \in S, \\ (a_i, b_i c_i), & i \notin S, \end{cases}$$

with block outputs concatenated in order. Hence $\tau_{[n]} = \tau_{AB}$ and $\tau_{\varnothing} = \tau_{BC}$. Every member of the family is lossless, uses the same active vocabulary V, and emits exactly one singleton and one doubleton per block.

Thus every τ_S outputs exactly $2n$ data tokens. Appending the same final token gives model-input length $N = 2n + 1$.

Table 3: **Matched resources in the tokenizer witness.** Every data-vocabulary item is active, both conditions append the same public final token, and neither condition gains dummy token types. They share the position count, length histogram, embedding-rank allowance, and decoder budget. The changed resource is the pair of adjacent raw coordinates represented by one data token.

Property	τ_{AB}	τ_{BC}	Match
Active data vocabulary	all $\Sigma \cup \Sigma^2$	all $\Sigma \cup \Sigma^2$	exact
Number of data-token positions	$2n$	$2n$	exact
Public final / model length	same / $2n + 1$	same / $2n + 1$	exact
Singletons / doubletons	n/n	n/n	exact per input
Maximum raw span	2	2	exact
Encoder	fixed period-3 grouping	fixed period-3 grouping	symmetric
Embedding-rank allowance	same bound	same bound	exact
Decoder class / budget	shared	shared	exact

Define

$$F_n^{AB}(x) = \bigvee_{i=1}^{n} (a_i \wedge b_i), \qquad F_n^{BC}(x) = \bigvee_{i=1}^{n} (b_i \wedge c_i),$$

and the invariant control

$$F_n^{A}(x) = \bigvee_{i=1}^{n} a_i.$$

A.6.1 Exact materialization costs and certificate allocation

Let P_{AB} give Alice a and Bob (b, c), and let P_{BC} give Alice (a, b) and Bob c. For every $S \subseteq [n]$,

$$\mathrm{ECC}_{\tau_S}(P_{AB}) = |S|, \qquad \mathrm{ECC}_{\tau_S}(P_{BC}) = n - |S|.$$

Consequently, if a one-layer Transformer computes the indicated task from τ_S, then

$$\begin{aligned} F_n^{AB}: &\quad C_{\mathrm{sim}}\, m\big(p + \log(2n+1)\big) H \geq n - |S|, \\ F_n^{BC}: &\quad C_{\mathrm{sim}}\, m\big(p + \log(2n+1)\big) H \geq |S|. \end{aligned}$$

The two residual lower-bound numerators sum to n. Moving one block boundary from BC to AB transfers one bit of token-materialization communication, and hence one unit between the two lower-bound certificates.

Proof Under P_{AB}, exactly the chunks $a_i b_i$ with $i \in S$ cross the raw cut, each with one coordinate on either side. All other chunks are local. Theorem 2 therefore gives $\mathrm{ECC}_{\tau_S}(P_{AB}) = |S|$. Under P_{BC}, exactly the chunks $b_i c_i$ with $i \notin S$ cross, so the cost is $n - |S|$.

Fixing the nuisance string reduces each target under its corresponding partition to n-bit set non-disjointness, whose deterministic communication complexity is at least n. The two inequalities follow from Theorem 1. □

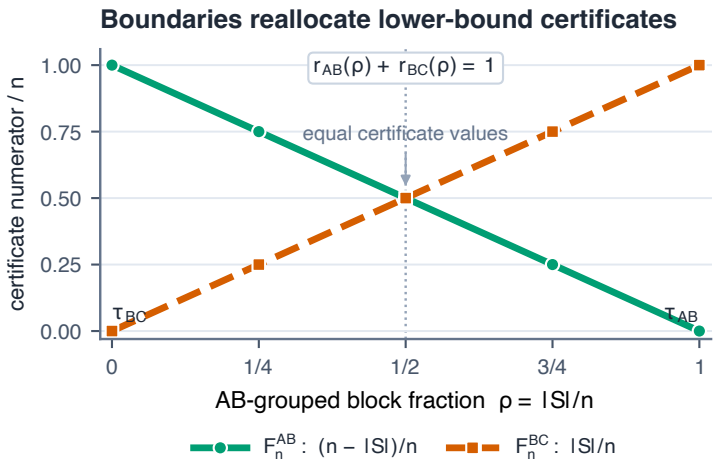

Figure 2: **A matched tokenizer family reallocates paired one-layer lower-bound certificates.** At AB-grouped fraction $\rho = |S|/n$, the normalized lower-bound numerators are $1 - \rho$ for F_n^{AB} and ρ for F_n^{BC}. The two curves sum to one while every scalar resource remains matched. The curves plot theoretical certificate numerators, not measured widths.

At the endpoints, $S = [n]$ and $S = \varnothing$, the materialization costs are exactly n and zero in opposite directions. Thus the endpoint witness is not an isolated pair: it is the pair of extreme points of a fully resource-matched allocation curve.

A.7 Misaligned one-layer lower bounds

A.7.1 One-layer lower bound for F^{AB} under τ_{BC}

If a one-layer $T \in \mathrm{TF}^1_{m,p,H}$ computes F_n^{AB} from $\tau_{BC}(x)$, then

$$C_{\mathrm{sim}}\, m\big(p + \log(2n+1)\big)H \geq n.$$

Proof Give Alice $a = (a_1, \ldots, a_n)$ and Bob (b, c). Under τ_{BC}, Alice locally constructs every odd token a_i, and Bob locally constructs every even token $b_i c_i$. The common final token is public. Hence this raw partition is τ_{BC}-separable.

Fix $c = 0^n$. On the resulting subdomain,

$$F_n^{AB}(x) = \bigvee_i (a_i \wedge b_i)$$

is set non-disjointness. Its deterministic communication complexity is at least n bits. Corollary 1 therefore gives $n \leq C_{\mathrm{sim}}\, m(p + \log(2n+1))H$. □

An analogous randomized claim would require a corresponding randomized simulation of the Transformer. We do not claim that extension here; every lower bound in this paper concerns deterministic exact computation.

A.7.2 Reverse lower bound

If a one-layer $T \in \mathrm{TF}^1_{m,p,H}$ computes F_n^{BC} from $\tau_{AB}(x)$, then

$$C_{\mathrm{sim}}\, m\big(p + \log(2n+1)\big)H \geq n.$$

Proof Give Alice (a, b) and Bob c. Under τ_{AB}, Alice constructs every odd token $a_i b_i$, and Bob constructs every even token c_i. Fix $a = 0^n$. The task is then set non-disjointness on b, c, so the same argument as Section A.7.1 applies. □

A.8 Aligned one-layer upper bounds

A.8.1 Aligned blockwise intersection

In the finite-precision model above, F_n^{AB} under τ_{AB} and F_n^{BC} under τ_{BC} are each computable by a one-layer Transformer with one head, constant width, and $p = O(\log n)$ bits.

Construction Consider F_n^{AB} under τ_{AB}; the other case is identical. Give token `11` a value marker 1 and every other vocabulary item and the final token marker 0. Because doubleton tokens occur exactly at the aligned positions, a marker 1 is equivalent to $a_i = b_i = 1$ for some block.

Set the final token's query and all keys to zero. Its attention weights are uniform over the $N = 2n + 1$ positions, so its attention output is k/N, where k is the number of blocks with $a_i = b_i = 1$. A constant-width ReLU network can separate $k = 0$ from $k \geq 1$; in exact arithmetic, for example,

$$h(z) = \mathrm{ReLU}(Nz) - \mathrm{ReLU}(Nz - 1)$$

equals 0 at $z = 0$ and 1 at every $z = k/N$ with integer $k \geq 1$.

This construction is exact under the source paper's grid. For the fixed $K_c = 12$ above, the grid step is $\Delta = N^{-K_c}$, so

$$\frac{1}{N} = N^{K_c - 1}\Delta$$

is representable. With all logits equal to zero, every exponentiated logit is exactly 1, their sum is exactly N, and every softmax weight is exactly $1/N$. Hence $z = k/N$, multiplication by N yields the integer k, and both ReLU evaluations return exact grid values. The largest intermediate is at most N, inside the range $[-N^{K_c}, N^{K_c}]$. Thus $p = \lceil \log_2(2N^{2K_c} + 1) \rceil = O(\log n)$ suffices. □

A.8.2 Invariant control

F_n^A is computable with one head, constant width, and $O(\log n)$ precision under both τ_{AB} and τ_{BC}.

Proof Under τ_{AB}, mark doubleton tokens `10` and `11`; under τ_{BC}, mark singleton token `1` at the odd positions. Uniform final-token attention and the threshold construction in Section A.8.1 compute OR over the markers. The vocabulary distinguishes singleton `1` from doubleton tokens, so no positional test is needed. The finite-precision operations are unchanged. □

A.9 Misaligned two-layer repair

A.9.1 Exact causal lookup

Fix $N \geq 2$ and a public target map $\pi : [N] \to [N]$ satisfying $\pi(i) \leq i$. A causal attention head of width $O(\log N)$ and precision $O(\log N)$ can retrieve the value at position $\pi(i)$ exactly at every position i.

Proof Set $r = \lceil 32 \ln N \rceil$, and choose sign vectors $u_1, \ldots, u_N \in \{-1, 1\}^r$ such that

$$\langle u_i, u_j \rangle \leq r/2 \quad \text{for all } i \neq j.$$

Such a family exists by sampling independent Rademacher vectors: Hoeffding bounds the failure probability for one pair by $e^{-r/8}$, and a union bound over fewer than N^2 pairs is strictly below one. Put both u_i and $u_{\pi(i)}$ in the public positional embedding at position i. A shared block-linear query map selects the target-code block, and a shared key map selects the current-position block. With

$$Q_i = [\eta u_{\pi(i)}, \eta], \qquad K_j = [u_j, -r],$$

the target logit is exactly zero, whereas every non-target logit is at most $-\eta r/2$. Taking $\eta = \lceil 2K_c \ln N \rceil + 1 = O(\log N)$ gives $\exp(-\eta r/2) < \Delta = N^{-K_c}$, the grid step of the finite-precision model. The source implementation stabilizes softmax by subtracting the row maximum before exponentiation. Here the unique maximum is already the target logit zero, so stabilization changes no logit. Appendices B.2 and D of Bhattamishra et al. (2024) round every intermediate down to the grid and apply that rule to the exponentials. Hence the target exponential is exactly 1, every non-target exponential is exactly 0, and the causal softmax row is exactly one-hot at $\pi(i)$. The target remains available because $\pi(i) \le i$. All code entries and the shift $-r$ are integers, the embedding dimension is $O(\log N)$, and the largest query/key coordinate has $O(\log N)$ magnitude. The largest logit magnitude is at most $2\eta r$. With $K_c = 12$, $2\eta r \le 2(24 \ln N + 2)(32 \ln N + 1) \le N^{12}$ for every $N \ge 2$: the inequality holds at $N = 2$, and the logarithm of its right-to-left ratio is increasing thereafter. Thus no logit clips, while $p = O(\log N)$ still suffices. □

A.9.2 One additional layer repairs the boundary

There are causal two-layer Transformers with $O(1)$ heads, width $O(\log n)$, and $p = O(\log n)$ that compute F_n^{AB} under τ_{BC} and F_n^{BC} under τ_{AB}.

Proof For F_n^{AB} under τ_{BC}, position $2i$ contains $b_i c_i$ and its predecessor contains a_i. In the first layer, use two instances of the exact lookup construction in Section A.9.1. The predecessor head uses

$$\pi_{\mathrm{prev}}(j) = \begin{cases} j-1, & j \le 2n \text{ is even}, \\ j, & \text{otherwise}, \end{cases}$$

and the self head uses $\pi_{\mathrm{self}}(j) = j$. Their value maps expose, respectively, the bit from the predecessor singleton and the first bit of the current doubleton; each map returns zero on token types outside that role. The self-head value also includes the public positional coordinate $e_j = \mathbf{1}[j \le 2n \text{ is even}]$. The multi-head output projection routes the predecessor bit, self bit, and e_j as three separate coordinates to the same position-wise ReLU network. Both lookups are exact, so the retrieved scalars $r_j, s_j \in \{0, 1\}$ satisfy $r_{2i} = a_i$ and $s_{2i} = b_i$. Map the final token to zero in both value maps. At every odd data position, $e_j = 0$, the self-bit map is zero, and $r_j \le 1$; at the final position all three coordinates are zero. The position-wise ReLU network therefore writes

$$z_j = \mathrm{ReLU}(r_j + s_j + e_j - 2) = \begin{cases} a_i \wedge b_i, & j = 2i, \\ 0, & \text{otherwise}. \end{cases}$$

The second layer gives the final token uniform attention over the $N = 2n + 1$ first-layer outputs and applies the exact threshold construction from Section A.8.1 to $\sum_j z_j / N$.

For F_n^{BC} under τ_{AB}, position $2i$ contains c_i; the predecessor token $a_i b_i$ exposes b_i. The same two-head construction applies after changing the two value maps: the predecessor map reads the second bit of a doubleton and is zero on singletons, while the self map reads a singleton and is zero on doubletons. Thus the off-target positions again feed at most one active bit and $e_j = 0$ to the gate. The total width is dominated by a constant number of $r = O(\log N)$-dimensional positional code blocks, the number of heads is two, and $p = O(\log N)$. This is a self-contained causal specialization of the sparse-feature construction behind Bhattamishra et al., Theorem 13. □

A.10 Combined separation

The balanced construction yields in the model specified in Appendix A.1:

- one layer and constant width for the aligned task;
- one layer only with $m(p + \log n)H = \Omega(n)$ for the misaligned task;
- two layers and $O(\log n)$ width for the misaligned task;
- the reverse separation when the target changes from AB to BC; and

- one-layer constant width under both tokenizers for the invariant control.

Within this matched family and the two intersection tasks, neither endpoint tokenizer dominates. The gap is determined by whether task-coupled raw variables share a token, while active vocabulary, encoded length, and token-length histogram remain identical.

A.11 A CANONICAL-BPE REALIZATION ON A TYPED DOMAIN

Corollary (typed canonical-BPE endpoints). The two binary endpoint constructions admit canonical-BPE realizations after a public, coordinatewise change of alphabet. This is a separate synthetic construction, not a BPE realization of the original binary tokenizers. Its communication and network consequences are inherited from the endpoint results of Theorem 3 under the same model assumptions.

For $n \geq 1$, take six atomic symbols $\widetilde{\Sigma} = \{A_0, A_1, B_0, B_1, C_0, C_1\}$ and the full structured product

$$\widetilde{X}_n = (\{A_0, A_1\} \times \{B_0, B_1\} \times \{C_0, C_1\})^n.$$

Each displayed label denotes one symbol. The domain is not all of $\widetilde{\Sigma}^{3n}$. Define the public bijection

$$\iota_n(a_1 b_1 c_1 \cdots a_n b_n c_n) = A_{a_1} B_{b_1} C_{c_1} \cdots A_{a_n} B_{b_n} C_{c_n}.$$

For each endpoint, rank its following four rules in any fixed total order:

$$D_{AB} : (A_a, B_b) \mapsto A_a B_b, \quad a, b \in \{0, 1\},$$
$$D_{BC} : (B_b, C_c) \mapsto B_b C_c, \quad b, c \in \{0, 1\}.$$

Each dictionary contains only its four distinct pairs. Use atomic initialization, identity normalization, no pretokenization boundaries, no dropout, and no additional merges or automatic markers. Append the common public EOS token, representing the empty string, only after BPE.

Proof. Both dictionaries are proper ranked dictionaries because every operand is atomic; no earlier rule is needed to construct an operand (Berglund et al., 2025, Definition 3 and Algorithm 1). Every eligible pair lies inside one typed block. Eligible pairs are disjoint, and merging one neither destroys another eligible pair nor creates a new one, since no merged symbol is a rule operand. Canonical BPE therefore performs exactly n merges, independently of the ranking, and produces

$$\widetilde{\tau}_{AB}(\iota_n(x)) = (A_{a_1} B_{b_1}, C_{c_1}, \ldots, A_{a_n} B_{b_n}, C_{c_n}),$$
$$\widetilde{\tau}_{BC}(\iota_n(x)) = (A_{a_1}, B_{b_1} C_{c_1}, \ldots, A_{a_n}, B_{b_n} C_{c_n}).$$

These displays omit EOS. All four rules are available, but they need not all fire on a particular input. Concatenating the represented strings recovers the input exactly.

The recoding and its inverse act independently at public coordinates. Thus either party can translate its raw input without communication, and for every raw partition P and $\widetilde{F} = F \circ \iota_n^{-1}$,

$$\mathrm{CC}_P(\widetilde{F}) = \mathrm{CC}_P(F).$$

There are also endpoint-specific active-token bijections

$$\phi_{AB}(ab) = A_a B_b, \quad \phi_{AB}(c) = C_c,$$
$$\phi_{BC}(a) = A_a, \quad \phi_{BC}(bc) = B_b C_c,$$

with EOS fixed. Applying these maps or their inverses to locally materialized tokens preserves every protocol. Consequently, for $s \in \{AB, BC\}$, every P, and every token partition Q,

$$\mathrm{ECC}_{\widetilde{\tau}_s}(P; Q) = \mathrm{ECC}_{\tau_s}(P; Q).$$

The equality also holds after minimizing over fixed ownership or allowing transcript-public adaptive ownership.

Copying each binary embedding row to its corresponding active typed row preserves every downstream activation. Conversely, restricting a typed model to active rows gives the binary model. Keep all other weights, positional embeddings, and finite-precision operations unchanged. The existing

endpoint results therefore transfer: the aligned intersection tasks have constant-width one-layer constructions, while any one-layer Transformer computing the corresponding typed task exactly from the opposite endpoint must satisfy

$$C_{\mathrm{sim}}\, m\big(p + \log(2n+1)\big)H \geq n.$$

The two-layer $O(\log n)$-width repairs and constant-width invariant control transfer as well, under their original assumptions.

Each endpoint stores six atomic and four merged data types, but only six data types are active over the full domain: four doubletons and the two remaining singletons. Thus both have 10 stored and six active data types, or 11 and seven including EOS. Each input yields $2n$ data tokens with n singletons and n doubletons, and model length $N = 2n + 1$. The cardinalities and length histograms match; the represented-string vocabularies and ordered length profiles do not. Setting the four unused rows to zero in the transferred construction preserves the original embedding rank, width, heads, depth, and precision. The equal 11-row tables nevertheless contain $4m$ more nominal embedding parameters than the original seven-row binary tables. □

This corollary concerns prescribed endpoint dictionaries only. It does not establish a realization of the interior-S family, learned merge discovery, or behavior of natural-language or pretrained BPE tokenizers. It neither changes the frozen binary experiments nor inherits their stronger identical-active-vocabulary comparison.

A.12 A DEPLOYED TOKENIZER: LEFT-TO-RIGHT VERSUS RIGHT-TO-LEFT DIGIT GROUPING

GPT-3.5 and GPT-4 use the cl100k_base tokenizer. Its pre-tokenizer splits each digit run into pieces of at most three digits, starting from the left (pattern `\p{N}{1,3}`). We checked this with `tiktoken` for $L = 4, \ldots, 12$. An L-digit number is chunked as $\mathrm{L2R} = (3, \ldots, 3, r)$, where $r \equiv L \pmod 3$. Thousands separators change the digit chunks to $\mathrm{R2L} = (r, 3, \ldots, 3)$. Singh & Strouse (2024) report that enforcing R2L in this way substantially improves the addition accuracy of GPT-3.5 and GPT-4.

We treat the digits as raw coordinates and the commas as delimiters. Both chunkings are public and input-independent, so Theorem 2 applies, and they share a chunk-length multiset. Corollary 3 therefore makes their mean boundary loads equal, both over all labelled cuts and for each cut size. Exact enumeration over all 2^L cuts ($L \leq 12$) confirms this. The common mean is $\frac{3}{4}n_3 + \frac{1}{2}n_2$, where n_c is the number of chunks of length c.

The schemes differ in where their zero-load cuts fall. Let P_k separate the digits of place value below 10^k from the rest. Under either scheme $\mathrm{BL}(P_k) \in \{0, 1\}$, and each scheme frees $\lceil L/3 \rceil - 1$ of the $L - 1$ cuts. R2L frees exactly the cuts with $k \equiv 0 \pmod 3$, for every L. L2R frees the cuts with $k \equiv L \pmod 3$ (Table 4). In $a + b$, the joint cut at 10^k is free under R2L for every $k \in \{3, 6, \ldots\}$, whatever the operand lengths. Under L2R this happens only when $L_a \equiv L_b \pmod 3$. For $(L_a, L_b) = (7, 5), (8, 6), (9, 4), (6, 6)$, R2L has $2, 2, 2, 1$ jointly free cuts and L2R has $0, 0, 1, 1$. The total load summed over k is the same under both schemes.

L	L2R chunks	free k, L2R	free k, R2L
7	$(3, 3, 1)$	$\{1, 4\}$	$\{3, 6\}$
8	$(3, 3, 2)$	$\{2, 5\}$	$\{3, 6\}$
9	$(3, 3, 3)$	$\{3, 6\}$	$\{3, 6\}$
10	$(3, 3, 3, 1)$	$\{1, 4, 7\}$	$\{3, 6, 9\}$
11	$(3, 3, 3, 2)$	$\{2, 5, 8\}$	$\{3, 6, 9\}$

Table 4: Zero-load place-value cuts P_k, $1 \leq k < L$. R2L chunks are the L2R chunks reversed.

This analysis records which place-value interactions each tokenizer assembles inside a single token. It is consistent with the empirical R2L advantage, but it does not explain that advantage. Our bounds concern exact one-layer computation. They do not model carries, depth, training, or separator tokens.

B PRESPECIFIED EXPERIMENTAL PROTOCOL

This appendix records the outcome-independent protocol for the empirical arm. It does not report the CPU/MPS mechanics checks or the excluded learning-rate pilot as scientific evidence.

B.1 COMPARISON CONTRACT

For every paired run, the two tokenizer conditions share raw examples and ordering, the seven-token inventory $V = \{0, 1, 00, 01, 10, 11, \texttt{<eos>}\}$, encoded length, attention mask, number of singleton and doubleton tokens, embedding dimension, head count, depth, feed-forward dimension, classifier, parameter count, initialization seed, minibatch schedule, optimizer, learning-rate schedule, maximum updates, stopping rule, data splits, and evaluation code. The sole intervention is

$$\tau_{AB}(abc) = [ab][c] \quad \text{versus} \quad \tau_{BC}(abc) = [a][bc].$$

No condition receives unused vocabulary padding, an extra position, a tokenizer-specific optimizer, or a separately tuned learning rate. The same contract applies to every partial tokenizer $\tau_{S_{n,\rho}}$, where only the public boundary schedule varies.

B.2 SHORTCUT-RESISTANT DATA

Independent Bernoulli bits are unsuitable because non-disjointness becomes almost always positive as n grows. The prespecified distribution instead fixes $k = \max(1, \lfloor n/4 \rfloor)$. For F^{AB}, sample a support $A \subset [n]$ uniformly with $|A| = k$. A negative example uses a uniformly sampled support $B \subset [n] \setminus A$ with $|B| = k$. A positive example selects one witness $w \in A$ and sets $B = \{w\} \cup C$, where $C \subset [n] \setminus A$ and $|C| = k - 1$. The nuisance support for c also contains exactly k coordinates. Both labels therefore have identical marginal Hamming weights and differ only in the relevant intersection. The F^{BC} generator permutes coordinate roles through the same implementation.

For the control F^{A}, negative examples have $a = 0^n$, positive examples contain one uniformly selected active a-coordinate, and b, c follow the same fixed-weight nuisance distribution in both classes. The control is fixed before results and cannot be replaced.

Each final task, size, and data seed has 20,000 training, 4,000 validation, and 20,000 in-distribution test examples. The two intersection tasks additionally have 20,000 OOD examples; the control has no registered OOD split. The OOD intersection cells vary individual support weights over $\{\lfloor n/8 \rfloor, \lfloor n/4 \rfloor, \lfloor n/3 \rfloor\}$ and intersection size over $\{0, 1, 2, 4\}$ where feasible. For $n \leq 6$, exhaustive evaluation over all relevant raw pairs checks the hand-constructed analytical models. This exhaustive split is not a learned-run endpoint because the learned grid begins at $n = 16$.

Every serialized record contains the raw string, task, label, supports, witness indices, data seed, split, and token IDs for all prespecified boundary schedules. Before training, a deterministic audit checks decoding, labels, raw split disjointness, active vocabulary, encoded lengths, token-length histograms, specified OOD-cell membership, and marginal weights.

B.3 MODELS AND ANALYTICAL GATE

Track A is the theorem-faithful causal micro-Transformer: learned shared token embeddings, learned shared positional embeddings, two softmax-attention heads, a width-$4m$ ReLU feed-forward block, no residual connection, no normalization, a common final token, and a binary classifier. Track B adds pre-normalization and residual connections as a separate robustness test; it does not expand the formal theorem.

Before any learned result is eligible, hand-constructed aligned one-layer and misaligned two-layer models must achieve exact exhaustive small-n accuracy. Failure of this gate indicates a mismatch between the implementation and theorem model rather than a negative empirical result.

B.4 PRESPECIFIED SCHEDULES AND HYPERPARAMETER SELECTION

The Track A coarse grid contains

- $n \in \{16, 64, 128\}$,
- $m \in \{8, 32, 128\}$,
- depth $D \in \{1, 2\}$,
- all three tasks and both endpoint tokenizers, and
- three paired model seeds.

This is 162 paired jobs, or 324 learned conditions. Final data seeds are $910000 + 100n + j$, where $j \in \{1, 2, 3\}$ is the fixed task order. Model seeds are 950001–950003 and minibatch seeds are 960001–960003. All learned runs use one CUDA accelerator, 2,000 AdamW updates, batch size 256, weight decay 0.01, gradient clipping at norm 1, 100 linear warm-up updates, cosine decay, and validation every 100 updates.

Learning rates are selected from $\{10^{-4}, 3\times10^{-4}, 10^{-3}\}$ using pilot data excluded from final reporting. For each width/depth cell, the selector averages final validation balanced accuracy over all three tasks and both tokenizers, breaks ties by lower mean validation BCE and then lower learning rate, and applies the selected rate to every task and tokenizer.

The checksum-frozen pilot registry contains 54 paired jobs (108 learned conditions); its SHA-256 is

`1797e3df108c61adf86d106c7c258e52d06cedcf6e833105b0a11833331194ed`.

All 54 pilot pairs have been validated, and the selected learning rates and 162-pair coarse registry were frozen before coarse execution. The coarse registry SHA-256 is

`cffeeacb1bbc23def9f035cc4f6a3a4578112b259a0d0703d86fa2b93f333429`.

Refinement and allocation registries are generated and frozen before their respective runs.

The confirmatory policy fixes $n \in \{64, 128\}$ and both depths before coarse execution. It first requires all 162 coarse results and checkpoints to validate. For each n, the selector pools upward 99% crossings of three-seed mean final-validation balanced accuracy across both intersection tasks, both tokenizer conditions, and both depths. If only $[8, 32]$ is occupied, it inserts widths 12 and 20; if only $[32, 128]$ is occupied, it inserts 50 and 80; if both are occupied, it inserts 16 and 64. With no occupied interval it records `no_bracket` and adds no fallback. Downward and non-monotone curves are recorded rather than used to choose another interval.

Every selected width is run at both depths and on all three tasks. Each inserted-width cell contains ten new paired jobs. The first three replay model seeds 950001–950003 and minibatch seeds 960001–960003 at the new width; the remaining seven use fresh IDs 950004–950010 and 960004–960010. Coarse endpoint results stay separate three-seed descriptive cells. An inserted width inherits the selected learning rate from the nearest coarse width in log-width space, with exact ties resolved toward the smaller width. The maximum is 240 paired jobs, or 480 learned conditions. Any compute reduction must be checksum-frozen before coarse outcomes are inspected and cannot remove paired seeds, the reverse task, or the invariant control afterward.

The allocation extension uses

$$S_{n,\rho} = \{i \in [n] : \lfloor i\rho \rfloor > \lfloor (i-1)\rho \rfloor\}$$

for $\rho \in \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$. The three interior fractions are run at $n \in \{64, 128\}$, $m \in \{32, 128\}$, depth one, both intersection tasks, and the same three coarse-grid seeds, adding 72 learned conditions. Endpoint values are reused rather than retrained.

B.5 ESTIMANDS, REPORTING, AND REJECTION CRITERIA

Reported 99%, 99.9%, and 100% threshold crossings use recorded ID-test accuracy without rounding. This differs from the validation balanced accuracy used for refinement selection and the balanced-accuracy differences used for paired intervals.

The primary estimands are the paired tokenizer accuracy difference, the proportion of seeds reaching 99%, 99.9%, and 100% accuracy, the minimum width crossing each threshold, the relative improvement from one to two layers in aligned and misaligned conditions, and the difference-in-differences between the forward and reverse tasks. Learned runs report balanced accuracy, BCE, training accuracy, ID and OOD accuracy, parameter count, updates, and wall-clock compute. The analytical gate separately reports exhaustive accuracy through $n = 6$. Within each ten-seed refinement cell and evaluation split, let d_s be the balanced-accuracy difference between the two tokenizer conditions at matched model seed s. We report a two-sided 95% percentile interval for $\bar{d}$ from 10,000 resamples of the ten paired differences, sampling ten indices with replacement using NumPy `Generator(PCG64(20260909))`; the generator is reset to that seed for each cell and split. Threshold-crossing widths and three-seed coarse or allocation cells are not bootstrapped. Those cells are descriptive, and every seed point is shown.

For the allocation family, empirical transition widths are plotted against ρ alongside the normalized theoretical residual numerators on a separate axis. We test only the prespecified opposing directions; we do not fit equality between finite-size transition width and the asymptotic bound.

The communication-alignment interpretation is rejected if the same tokenizer wins both intersection tasks, the control gap is comparable to either directional gap, two layers do not preferentially repair misalignment, partial-alignment curves move together or show no reproducible opposing structure, paired gaps vanish under OOD evaluation, token-ID permutation changes results beyond seed noise, or a claimed gap depends on tokenizer-specific tuning. Negative results remain in the report and cannot be replaced with a post hoc task or distribution without an exploratory label.