Supplemental Material: Streaming Semi-CRF Inference

This supplement provides formal algorithmic details for the streaming Semi-CRF implementation.

1 Notation

Symbol	Description	Shape
T	Sequence length	scalar
K	Maximum segment duration	scalar
C	Number of labels (states)	scalar
B	Batch size	scalar
$\mathcal{S}_{t,c}$	Cumulative projected scores	$(B, T+1, C)$
$\mathcal{T}_{c',c}$	Transition scores (source c' to dest c)	(C, C)
$\mathcal{B}_{k,c}$	Duration bias for duration k, label c	(K, C)
$\tilde{\alpha}_t(c)$	Log-forward message at position t, label c	(B, C)
$\tilde{\beta}_t(c)$	Log-backward message	(B, C)
$\boldsymbol{\alpha}$	Ring buffer for forward messages	(B, K, C)
Ω	Checkpointed ring buffer states	(B, N, K, C)
$\mathcal{N}$	Cumulative log-normalization factors	(B, N)
Δ	Checkpoint interval ($\approx \sqrt{TK}$)	scalar
$\log Z$	Log partition function	$(B,)$

Table 1: Notation. Tilde ($\tilde{\cdot}$) denotes log-domain quantities. $N = \lceil T/\Delta \rceil$ is the number of checkpoints.

2 Edge Potential Decomposition

The key innovation enabling $O(KC)$ memory is computing edge potentials on-the-fly from cumulative scores:

$$\tilde{\psi}(t, k, c, c') = \underbrace{(\mathcal{S}_{t,c} - \mathcal{S}_{t-k,c})}_{\text{segment content}} + \underbrace{\mathcal{B}_{k,c}}_{\text{duration bias}} + \underbrace{\mathcal{T}_{c',c}}_{\text{transition}} \tag{1}$$

where t is the segment end position (1-indexed), $k \in \{1, \ldots, K\}$ is the duration, c is the destination state, and c' is the source state. The prefix-sum decomposition allows $O(1)$ edge computation instead of $O(k)$.

3 Streaming Forward Algorithm

Algorithm 1 maintains a ring buffer $\boldsymbol{\alpha} \in \mathbb{R}^{K \times C}$ storing the K most recent forward messages. Following the Flash Attention [1] and Mamba [2] pattern, we normalize at checkpoint boundaries to prevent overflow at extreme T.

Memory: $O(KC)$ ring buffer + $O(\sqrt{T/K} \cdot KC)$ checkpoints. **Time**: $O(TKC^2)$.

4 Backward Pass with Checkpointing

Algorithm 2 processes segments in reverse, recomputing α from checkpoints following gradient checkpointing [3]. The saved $\mathcal{N}_i$ restores the true scale when computing marginals.

Algorithm 1: Streaming Semi-CRF Forward Scan

Input: $\mathcal{S}$, $\mathcal{T}$, $\mathcal{B}$, checkpoint interval Δ
Output: $\log Z$, checkpoints $(\Omega, \mathcal{N})$

$\boldsymbol{\alpha} \leftarrow -\infty$; $\quad \boldsymbol{\alpha}[0,:] \leftarrow 0$; $\quad \mathcal{N}_{\text{accum}} \leftarrow 0$;
$\Omega[0] \leftarrow \boldsymbol{\alpha}$; $\quad \mathcal{N}[0] \leftarrow 0$;
for $t \leftarrow 1$ **to** T **do**
  $\mathbf{v}_t \leftarrow -\infty \in \mathbb{R}^C$;
  for $k \leftarrow 1$ **to** $\min(K, t)$ **do**
    $\tilde{\boldsymbol{\alpha}}_{\text{prev}} \leftarrow \boldsymbol{\alpha}[(t-k) \bmod K, :]$;
    $\mathbf{h} \leftarrow \big(\mathcal{S}_{t,:} - \mathcal{S}_{t-k,:}\big) + \mathcal{B}_{k,:}$;
    $\mathbf{E} \leftarrow \mathbf{h}[:, \text{None}] + \mathcal{T}^\top$;
    $\mathbf{s}_k \leftarrow \text{LSE}(\tilde{\boldsymbol{\alpha}}_{\text{prev}}[\text{None}, :] + \mathbf{E}, \text{axis} = 1)$;
    $\mathbf{v}_t \leftarrow \text{LSE}(\mathbf{v}_t, \mathbf{s}_k)$;
  $\boldsymbol{\alpha}[t \bmod K, :] \leftarrow \mathbf{v}_t$;
  if $t \bmod \Delta = 0$ **then** `// Normalize at checkpoint [1]`
    $n \leftarrow t/\Delta$; $\quad s \leftarrow \max_c \mathbf{v}_t(c)$;
    $\mathcal{N}_{\text{accum}} \leftarrow \mathcal{N}_{\text{accum}} + s$; $\quad \boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} - s$;
    $\Omega[n] \leftarrow \boldsymbol{\alpha}$; $\quad \mathcal{N}[n] \leftarrow \mathcal{N}_{\text{accum}}$;
$\log Z \leftarrow \text{LSE}(\boldsymbol{\alpha}[T \bmod K, :]) + \mathcal{N}_{\text{accum}}$;
return $\log Z$, $(\Omega, \mathcal{N})$;

5 Gradient Computation

Gradients are computed via marginal probabilities:

$$\mu(t, k, c, c') = \exp\big(\tilde{\alpha}_{t-k}(c') + \tilde{\psi}(t, k, c, c') + \tilde{\beta}_t(c) + \mathcal{N}_i - \log Z\big) \tag{2}$$

For per-sequence parameters:

$$\nabla \mathcal{S}_{t,c} = \frac{\partial \mathcal{L}}{\partial Z_b} \cdot \left(\sum_{k,c'} \mu_b(t, k, c, c') - \sum_{k,c'} \mu_b(t+k, k, c, c') \right) \tag{3}$$

For shared parameters, we accumulate per-batch then reduce via einsum:

$$\nabla \mathcal{T}_{c',c} = \sum_b \frac{\partial \mathcal{L}}{\partial Z_b} \cdot \sum_{t,k} \mu_b(t, k, c, c') \tag{4}$$

$$\nabla \mathcal{B}_{k,c} = \sum_b \frac{\partial \mathcal{L}}{\partial Z_b} \cdot \sum_{t,c'} \mu_b(t, k, c, c') \tag{5}$$

6 Numerical Stability

Zero-centering. Cumulative scores are zero-centered before cumsum to bound magnitude at large T: $\bar{s}_{t,c} = s_{t,c} - \frac{1}{T} \sum_\tau s_{\tau,c}$.

Log-domain. All computations use logsumexp: $\text{LSE}(\mathbf{x}) = \max(\mathbf{x}) + \log \sum_i \exp(x_i - \max(\mathbf{x}))$.

Masking. Invalid positions use -10^9 (not $-\infty$) to avoid NaN in gradients.

Variable-length batches. For sequences ending at $L < T$, the normalization shift is masked to zero for $t > L$, freezing $\mathcal{N}_{\text{accum}}$ at the correct value.

Algorithm 2: Streaming Semi-CRF Backward Pass

Input: $\mathcal{S}$, $\mathcal{T}$, $\mathcal{B}$, checkpoints $(\Omega, \mathcal{N})$, $\log Z$, upstream $\partial\mathcal{L}/\partial Z$
Output: $\nabla\mathcal{S}$, $\nabla\mathcal{T}$, $\nabla\mathcal{B}$

$\boldsymbol{\beta} \leftarrow -\infty$; $\quad \boldsymbol{\beta}[T \bmod K, :] \leftarrow 0$;
Initialize gradient accumulators;
for $i \leftarrow N_{ckpts} - 1$ **to** 0 **do**
  $t_{\text{start}} \leftarrow i \cdot \Delta$; $\quad t_{\text{end}} \leftarrow \min((i+1) \cdot \Delta, T)$;
  $\mathcal{N}_i \leftarrow \mathcal{N}[i]$;
  $\boldsymbol{\alpha}_{\text{local}} \leftarrow$ RecomputeAlpha($\Omega[i]$, t_{start}, t_{end});
  for $t \leftarrow t_{end} - 1$ **to** t_{start} **do**
    $\tilde{\boldsymbol{\alpha}}_t \leftarrow \boldsymbol{\alpha}_{\text{local}}[t - t_{\text{start}}, :]$;
    $\tilde{\boldsymbol{\beta}}_t \leftarrow -\infty$;
    for $k \leftarrow 1$ **to** $\min(K, T - t)$ **do**
      $\tilde{\boldsymbol{\beta}}_{\text{next}} \leftarrow \boldsymbol{\beta}[(t+k) \bmod K, :]$;
      $\tilde{\psi} \leftarrow$ Eq. (1);
      `// Log-norm correction restores true` α `scale`
      $\log \mu \leftarrow \tilde{\boldsymbol{\alpha}}_t[\text{None}, :] + \tilde{\psi} + \tilde{\boldsymbol{\beta}}_{\text{next}}[:, \text{None}] + \mathcal{N}_i - \log Z$;
      $\mu \leftarrow \exp(\log \mu)$;
      Accumulate $\nabla\mathcal{S}$, $\nabla\mathcal{T}$, $\nabla\mathcal{B}$ from μ;
      $\tilde{\boldsymbol{\beta}}_t \leftarrow \text{LSE}(\tilde{\boldsymbol{\beta}}_t, \text{LSE}(\tilde{\psi} + \tilde{\boldsymbol{\beta}}_{\text{next}}[:, \text{None}], \text{axis} = 0))$;
    $\boldsymbol{\beta}[t \bmod K, :] \leftarrow \tilde{\boldsymbol{\beta}}_t$;
return $\nabla\mathcal{S}$, $\nabla\mathcal{T}$, $\nabla\mathcal{B}$;

7 Implementation Correspondence

Math	Code Variable	Notes
$\mathcal{S}_{t,c}$	`cum_scores[:, t, c]`	Boundary at index 0
$\mathcal{T}_{c',c}$	`transition[c_src, c_dst]`	Source-first storage
$\mathcal{B}_{k,c}$	`duration_bias[k, c]`	$k \in \{1, \ldots, K\}$
$\boldsymbol{\alpha}$	`alpha_ring`	Ring buffer
Ω	`ring_checkpoints`	Saved at intervals
$\mathcal{N}$	`log_norm_checkpoints`	Cumulative log-norm
Δ	`checkpoint_interval`	$\approx \sqrt{TK}$

Table 2: Mapping between mathematical notation and implementation.

References

[1] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *NeurIPS*, 2022.

[2] Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[3] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training Deep Nets with Sublinear Memory Cost. *arXiv preprint arXiv:1604.06174*, 2016.