SUPPLEMENTARY MATERIAL FOR
Compact, Efficient and Unlimited Capacity: Language Modelling with Compressed Suffix Trees

Algorithm 4 Compute one-sided occurrence counts, $N^{1+}(\bullet\,\alpha)$ or $N^{1+}(\alpha\,\bullet)$ for pattern α

```
Precondition: node $n$ in CST $t$ matches $\alpha$
1: function N1PLUS($t, n, \alpha$)
2:     $o \leftarrow 1$
3:     if string-depth($n$) $= |\alpha|$ then
4:         $o \leftarrow$ degree($n$)
5:     return $o$
```

Algorithm 5 Compute backward occurrence counts, $N^{1+}(\bullet\,\alpha)$, using only forward CST

```
Precondition: $v_F$ is the node in the forward CST $t_F$ matching pattern $\alpha$
Precondition: the CSA component, $a_F$ of $t_F$ is a wavelet tree
1: function N1PLUSBACK1($t_F, v_F, \alpha$)
2:     $S \leftarrow$ int-syms($a_F$, [lb($v_F$), rb($v_F$)])
3:     return $|S|$
```

Function/Constant	Description	Complexity
SAS	sample rate of the suffix array. determines the number of jumps in $\mathcal{T}^{bwt}$ required before a suffix array value can be accessed	8 (in our exp.)
SA$[i]$	access the i-th element of the suffix array	$O(SAS \log\sigma)$
leaf(n)	tests if node n is a leaf of the t	$O(1)$
string-depth(n)	pattern length for the path from root to n (inclusive). Requires *SA*$[i]$ access if leaf	$O(1)$ non-leaf; $O(SAS \log\sigma)$ leaf
edge(n, k)	k^{th} symbol in the edge label from root for node n. Requires *SA*$[i]$ access	$O(SAS \log\sigma)$
degree(n)	number of child nodes under parent n	$O(\sigma/64)$
children(n)	list of all d child nodes under n	$O(\sigma/64 + d)$
back-search$([l, r], s)$	finds the node $v = [l', r']$ from parent node $\alpha = v' = [l, r]$ matching the pattern $s\alpha$. Requires 2 RANK operations on the wavelet tree	$O(\log\sigma)$
fw-search$([l, r], s)$	finds the node $v = [l', r']$ from parent node $\alpha = v' = [l, r]$ matching the pattern αs. Requires $\log\sigma$ accesses to *SA* and one LCP access	$O(SAS \log^2\sigma + LCP_C)$
int-syms$(a, [l, r])$	finds the set of symbols $P(\alpha)$ preceeding pattern α matched by $[l, r]$; returns a list of tuples describing the bounds and the preceding symbol $\langle l, r, s\rangle$	$O(\|P(\alpha)\| \log\sigma)$

Table 1: Summary of CSA and CST functions used and their time complexity of inference. The above assumes that n or (equivalently) $[l, r]$ matches α in the CSA a and/or CST t.

Algorithm 6 Compute Kneser-Ney probability, $P\big(w_k | w_{k-(n-1)}^{k-1}\big)$, using a single CST

```
1:  function ProbKneserNey1($t_F, \mathbf{w}, n$)
2:      $v_F \leftarrow \text{root}(t_F)$                          ▷ match for context $w_{k-i}^{k-1}$
3:      $v_F^{\text{all}} \leftarrow \text{root}(t_F)$                     ▷ match for $w_{k-i}^{k}$
4:      $p \leftarrow 1$
5:      for $i \leftarrow 1$ to $n$ do
6:          $v_F^{\text{all}} \leftarrow \text{back-search}([\text{lb}(v_F^{\text{all}}), \text{rb}(v_F^{\text{all}})], w_{k-i+1})$      ▷ update matches in CST
7:          if $i > 1$ then
8:              $v_F \leftarrow \text{back-search}([\text{lb}(v_F), \text{rb}(v_F)], w_{k-i+1})$
9:          $D \leftarrow$ discount parameter for $n$-gram
10:         if $i = n$ then          ▷ compute the ‘count’ and ‘denominator’ for the full match
11:             $c \leftarrow \text{size}(v_F^{\text{all}})$
12:             $d \leftarrow \text{size}(v_F)$
13:         else
14:             $c \leftarrow \text{N1pBack1}(t_F, v_F^{\text{all}}, \bullet\, w_{k-i+1}^{k-1})$
15:             $d \leftarrow \text{N1pFrontBack1}(t_F, v_F, \bullet\, w_{k-i+1}^{k-1}\,\bullet)$     ▷ N.b., precompute $N^{1+}(\bullet\,\bullet)$
16:         if $i > 1$ then
17:             if $v_F$ is valid then       ▷ compute backoff probability, or backoff for unseen contexts
18:                 $d \leftarrow \text{size}(v_F)$
19:                 $q \leftarrow \text{N1p}(t_F, v_F, w_{k-i+1}^{k-1}\,\bullet)$
20:                 $p \leftarrow \frac{1}{d}\left(\max(c - D, 0) + Dqp\right)$
21:         else
22:             $p \leftarrow \frac{c}{d}$
23:     return $p$
```

Algorithm 7 Precompute Kneser-Ney discounts

```
1:  function PrecomputeDiscounts($t_R, n$)
2:      $c_{k,j} \leftarrow 0 \quad \forall k \in [1,n], j \in [1,4]$          ▷ number of $k$-grams with count $j$
3:      $N_{k,j}^{1} \leftarrow 0 \quad \forall k \in [1,n], j \in [1,4]$       ▷ number of $k$-grams with $N^1 \bullet \alpha = j$
4:      $N^{1+}(\bullet\,\bullet) \leftarrow 0$                              ▷ number of unique bigrams
5:      for $v_R \leftarrow \text{descendents}(\text{root}(t_R))$ do              ▷ depth-first search over nodes in CST
6:          $d_P \leftarrow \text{string-depth}(\text{parent}(v_R))$
7:          $d \leftarrow \text{string-depth}(v_R)$                        ▷ find the length of the edge
8:          for $k \leftarrow d_P + 1$ to $\min(d, d_P + n)$ do
9:              $s \leftarrow \text{edge}(v_R, k)$
10:             if $s$ is the end of sentence sentinel then
11:                 skip all children of $v_R$
12:             else
13:                 $f \leftarrow \text{size}(v_R)$                          ▷ retrieve pattern frequency
14:                 if $1 \le f \le 4$ then
15:                     $c_{k,f} \leftarrow c_{k,f} + 1$
16:                 if $f = 2$ then
17:                     $N^{1+}(\bullet, \bullet) \leftarrow N^{1+}(\bullet, \bullet) + 1$
18:                 $g \leftarrow N^{1+}(t_R) v_R \dot{\alpha}$                   ▷ retrieve occurrence count
19:                 if $1 \le g \le 4$ then
20:                     $N_{g,f}^{1} \leftarrow c_{k,f} + 1$
21:     return $c, N^1, N^{1+}(\bullet, \bullet)$
```

Language		Size (MB)	Tokens (M)	Token Types	Sentences (K)
Bulgarian	BG	36.11	8.53	114930	329
Czech	CS	53.48	12.25	174592	535
German	DE	171.80	44.07	399354	1785
English	EN	179.15	49.32	124233	1815
Finnish	FI	145.32	32.85	721389	1737
French	FR	197.68	53.82	147058	1792
Hungarian	HU	52.53	12.02	318882	527
Italian	IT	186.67	48.08	178259	1703
Portuguese	PT	187.20	49.03	183633	1737
Wikipedia		8637	9057	196	87835

Table 2: Tokens and sentence counts refer to the training partition.