Supplementary Material for Fast Algorithms for Learning with Long $N$-grams via Suffix Tree Based Matrix Multiplication

Hristo S. Paskov

1 Modified Sparse Matrix Format

The standard compressed sparse column (CSC) format for a sparse $M \times N$ matrix $X$ consisting of $n_z$ non-zero entries stores three arrays:

1. The jc array, an array of size $N + 1$ such that $jc[i + 1] - jc[i]$ gives the number of non-zero entries in column $i$.

2. The ir array, an array of size $n_z$ in which indices $jc[i], \ldots, jc[i + 1] - 1$ contain the row ids of the non-zero entries in column $i$.

3. The $x$ array, a double array of size $n_z$ containing the non-zero entries of $X$ in the same order that they are listed in the ir array.

This matrix format is inefficient when storing frequency data since we know all entries in $x$ are non-negative integers. Moreover, the number of bits needed to store each index in the jc array is $\lceil \log_2 n_z \rceil$ which can be significantly larger than $\lceil \log_2 U^X \rceil$ where $U^X$ is the largest number of non-zero elements in any column. Our modified CSC format simply replaces the jc array with an integer array of frequency containing the row ids of the non-zero entries in column $i$. Moreover, the number of bits needed to store each integer. Direct multiplication with $X$ requires $\Theta(n^2)$ operations whereas multiplication with $X'$ requires $\Theta(n)$ operations.

Next, to show that the node matrix can be inefficient, consider a document corpus comprised of $K$ documents and an alphabet of $K$ distinct characters $c_1, \ldots, c_K$. The $i$th document $D_i = c_1c_2 \ldots c_i$ is comprised of the first $i$ characters of the alphabet and the total corpus length is $n = \frac{K^2 + K}{2}$.

By inspecting the structure of the suffix tree $T_C$ for this corpus, it is possible to show that both the all $N$-grams matrix $X$ and all $N$-grams node matrix $X'$ have $\Theta(K^2)$ non-zero entries and thus require $\Theta(n\sqrt{n})$ memory to store and $\Theta(n\sqrt{n})$ operations to multiply.

In particular, consider the branch $\beta_1$ corresponding to suffix $D_K[1]$, i.e. the suffix consisting of $K$ characters and equal to the entire document $D_K$. Note that there is a document $D_i$ equalling every prefix $[i]D_K = c_1c_2 \ldots c_i$ of $D_K$. By construction, for $i = 1, \ldots, K - 1$, every occurrence of the substring $[i]D_K$ in $C$ is either followed by $c_{i+1}$ (for example in document $D_{i+1}$) or is the end of a document (i.e. $D_i$). This structure implies that $\beta_1$ contains $K - 1$ internal nodes pertaining to the first $K - 1$ characters in $D_K[1]$ and that the edge labels connecting these nodes contain a single character. For $i < K$ the internal node pertaining to character $c_i$ has two children: a leaf indicating the end of document $D_i$ and another internal node corresponding to character $c_{i+1}$. The final node in $\beta_1$ has character label $c_K$ and is a leaf signalling the end of $D_K$. If we count this node (for simplicity), the node pertaining to character $i$ appears in exactly $K - i + 1$ documents, so the column for substring $[i]D_K$ in the (all) node matrix $X'$ contains $K - i + 1$ non-zero entries. The $K$ prefixes of $D_K$ each pertain to a node in $\beta_1$ and have a column in $X'$ with a total of

$$\sum_{i=1}^{K} (K - i + 1) = \frac{K^2 + K}{2}$$

non-zero entries.

The other strings in the corpus are formed in a similar manner by looking at the prefixes of $c_i \ldots c_K$, i.e. all pre-
Our reasoning shows that any \( z \in Q \subset \zeta \) that \( \mu \in X \) has \( n \) columns. By iterating our earlier reasoning we see that branch \( \beta_k \) corresponds to (all prefixes of) suffix \( D_K[k] \) and it accounts for \( k \) of these nodes. In total these \( k \) nodes contribute

\[
\sum_{i=1}^{k} (k - i + 1) = \frac{k^2 + k}{2}
\]

(1)

non-zero entries to \( X \).

By summing equation (1) from \( k = 1, \ldots, K \) we find that \( X \) has \( \Theta(K^3) \), i.e. \( \Theta(n \sqrt{\pi}) \), non-zero entries and therefore is as inefficient as the naïve all \( N \)-grams matrix!

\section{Proof of Theorem 4}

Suppose that \( f \) is \( J \)-PI where \( J = \{ \zeta_1, \ldots, \zeta_m \} \) and let \( X^* \) be the set of minimizers of \( \min_{x \in \mathbb{R}^d} f(x) \). If \( X^* \) is empty then our proof is trivial, so we assume that \( X^* \) is not empty. The central idea behind our proof is that \( X^* \) must contain a Cartesian product of permutahedrons (Ziegler, 1995). In particular, given a finite vector \( a \in \mathbb{R}^n \), the permutahedron \( \mathbb{P}(a) \subset \mathbb{R}^n \) on \( a \) is the polyhedron formed by taking the convex hull of all \( n! \) \( n \)-vectors whose entries are some permutation of the entries of \( a \).

In order to see how this relates to \( f \), let \( x \in X^* \) be optimal and let \( x_{\zeta_k} \) denote the \( n_k = |\zeta_k| \) entries in \( x \) with indices in \( \zeta_k \). Since \( f \) is \( J \)-PI, it follows that \( f \)'s value remains unchanged if we permute the \( x_{\zeta_k} \) arbitrarily. In fact, by definition, if \( \bar{x} \) is the vector formed by arbitrarily permuting the entries within each \( \zeta_k \in J \), then \( f(x) = f(\bar{x}) \) so \( \bar{x} \in X^* \) is optimal as well. Assume, without loss of generality, that \( \zeta_1 = \{1, \ldots, n_1\} \), \( \zeta_2 = \{n_1 + 1, \ldots, n_1 + n_2\} \) and so on and define

\[
Q = \mathbb{P}(x_{\zeta_1}) \times \mathbb{P}(x_{\zeta_2}) \times \cdots \times \mathbb{P}(x_{\zeta_m}).
\]

Our reasoning shows that any \( z \in Q \) is optimal and hence \( Q \subset X^* \).

Now consider the centroid of \( Q \), \( \mu \in \mathbb{R}^d \). The centroid of \( \mathbb{P}(a) \) for \( a \in \mathbb{R}^n \) is simply the \( n \)-vector with \( \frac{1}{n} \sum_{i=1}^{n} a_i \) in every entry (Ziegler, 1995). Moreover, since \( Q \) is a Cartesian product of polyhedra, its centroid is given by stacking the centroids of its constituent polyhedra. Let \( \eta \in \mathbb{R}^m \) have its entries be \( \eta_k = \frac{x_{\zeta_k}}{n_k} \), i.e. the mean of the elements in \( x_{\zeta_k} \) and define \( V \in \{0, 1\}^{d \times m} \) to be the binary matrix in which column \( k \) has ones in indices \( \zeta_k \) and is all 0 otherwise. It follows that \( \mu = V \eta \), and since \( \mu \in Q \subset X^* \), there must be a minimizer of \( f \)'s whose entries are identical in each of the \( \zeta_k \).

This reasoning then shows that constrained problem

\[
\min_{x \in \mathbb{R}^d} f(x) \quad \text{subject to} \quad x \in \text{col } V. \quad (2)
\]

is a constrained convex problem (with a linear constraint) and therefore has a minimum that is lower bounded by the minimum of our original (unconstrained) problem. By construction of \( \mu \), we see that it satisfies the linear constraint and is an optimal point for both problems. It follows, then, that the minimizers of the problem in equation (2) are a subset of \( X^* \). Moreover, solving equation (2) will always provide a minimizer of the original optimization problem.

We can then replace the subspace constraint by noting that \( x \in \text{col } V \) if and only if \( x = V z \) for some \( z \in \mathbb{R}^d \). This leads to a problem which is equivalent to the problem in (2), namely

\[
\min_{z \in \mathbb{R}^m} f(V z) \quad (3)
\]

It follows that we obtain a minimizer of our original problem simply by setting \( x = V z \), i.e. \( x_i = z_i \) where \( i \in \zeta_k \). Importantly, equation (3) is a smaller minimization problem over \( m \) variables instead of \( d \) variables. We note that this proof is entirely geometric and the details of how problem (3) might further be reduced algebraically are problem dependent. QED.

\section*{References}