# 1 Tensor Networks

Our goal is to define a class of models over strings. We begin by constructing a model for strings of a fixed length $n$, i.e., over $\Sigma^n$, using the formalism of *tensor networks.*

A tensor $T$ is a multi-dimensional array of real numbers. We denote a tensor $T_v$ with $k$ axes and a corresponding tuple of dimensions $D_v = (d_1, \dots, d_k)$ as a map:

$$T_v : [d_1] \times \cdots \times [d_k] \mapsto \mathbb{R} \quad .$$

As a shorthand, we write $[D_v] \coloneqq [d_1] \times \cdots \times [d_k]$.

The fundamental operation on tensors is *contraction*, which generalizes matrix multiplication. Given two tensors, say $T_u$ with indices $(j_1, \dots, j_m, i_e)$ and $T_v$ with indices $(i_e, l_1, \dots, l_p)$, their contraction over the shared index $i_e$ of size $d_e$ produces a new tensor $T_C$. The new tensor's indices are the union of the uncontracted indices from $T_u$ and $T_v$:

$$T_C(j_1, \dots, j_m, l_1, \dots, l_p) = \sum_{i_e=1}^{d_e} T_u(j_1, \dots, j_m, i_e) \cdot T_v(i_e, l_1, \dots, l_p) \quad .$$

We now define a specific architecture relevant to our work.

**Definition 1.1** (Layered Tensor Network over $\Sigma^n$)**.** A *layered tensor network* $\mathcal{T}$ over $\Sigma^n$ is defined by a graph $G = (V, E)$ with the following structure:

- $V$ is a set of vertices, partitioned into layers $V_0, V_1, \dots, V_L$. Each vertex $v \in V$ corresponds to a tensor $T_v$.
- The input layer $V_0 = \{v_{0,1}, \dots, v_{0,n}\}$ consists of $n$ vertices. The corresponding tensors $\{T_{v_{0,1}}, \dots, T_{v_{0,n}}\}$ are vectors (tensors with one axis) of size $|\Sigma|$. These are initialized with a one-hot encoding of a string $w \in \Sigma^n$.
- $E$ is the set of edges, representing contractions. Edges only connect vertices in adjacent layers, i.e., $E \subseteq \bigcup_{l=0}^{L-1} \{\{u, v\} \mid u \in V_l, v \in V_{l+1}\}$.
- For any tensor $T_v$, its number of axes must equal the degree of its corresponding vertex $v$, i.e., $|D_v| = \deg(v)$. For any edge $e = \{u, v\} \in E$, the corresponding axes in $T_u$ and $T_v$ must have the same dimension.

The full contraction of the network, after initializing the input tensors with a string $w$, yields a scalar output $\mathcal{T}(w)$. This scalar induces a probability distribution over $\Sigma^n$ via:

$$S_{n,\mathcal{T}}(w) \coloneqq \frac{f(\mathcal{T}(w))}{\sum_{w' \in \Sigma^n} f(\mathcal{T}(w'))} \quad ,$$

where $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ is a suitable non-negative function, such as the exponential function $f(x) = \exp(x)$.

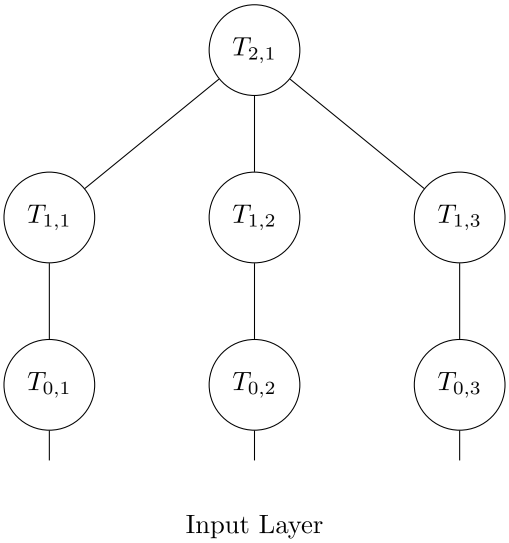

Figure 1: A basic tensor network over $\Sigma^3$.

**Definition 1.2** (Normalized and Non-Negative Tensor Networks)**.** Let $\mathcal{T}$ be a tensor network over $\Sigma^n$ with scalar output $\mathcal{T}(w)$ for each $w \in \Sigma^n$. Define the total mass of the network as

$$|\mathcal{T}| \coloneqq \sum_{w \in \Sigma^n} \mathcal{T}(w) \quad .$$

We say $\mathcal{T}$ is *normalized* iff $|\mathcal{T}| = 1$.

Furthermore, a tensor network is said to be *non-negative* iff for all $w \in \Sigma^n$ we have $\mathcal{T}(w) \geq 0$.

**Remark 1.1.** We can enforce all tensor networks of our model space to be non-negative by only allowing for non-negative tensors in the networks.

**Definition 1.3** (Normalization of Tensor Networks)**.** Let $\mathcal{T}$ be a tensor network over $\Sigma^n$, and let $H \coloneqq V \setminus I$ be the set of non-input tensors, and define $|H|$ as its cardinality. The *induced normalized tensor network* $\dfrac{\mathcal{T}}{|\mathcal{T}|}$ is the same network as $\mathcal{T}$, but each entry of each tensor in $H$ is scaled by the factor $\dfrac{1}{\sqrt[|H|]{|\mathcal{T}|}}$.

**Proposition 1.1.** *Let $J \subseteq [n]$ and let $\mathcal{T}$ be a tensor network over $\Sigma^n$. Define a modified network $\mathcal{T}_J$ where for all $j \in J$, the input tensor $T_{0,j}$ is initialized to the all-ones vector (i.e., $\mathbf{1} \in \mathbb{R}^{|\Sigma|}$). Then for any $w \in \Sigma^{[n]\setminus J}$:*

$$\sum_{w_J \in \Sigma^{|J|}} \mathcal{T}(w_J, w) = \mathcal{T}_J(w) \quad .$$

*Proof.* The output $\mathcal{T}(w)$ is a multilinear function of its input tensors. Let's focus on the input tensor $T_{0,j}$ for some $j \in [n]$. We can express the network's output as a linear function of this input vector:

$$\mathcal{T}(w) = \langle A_j(w_{\setminus j}), T_{0,j} \rangle \quad ,$$

where $T_{0,j}$ is the one-hot vector for the symbol $w_j$, and $A_j(w_{\setminus j})$ is a tensor representing the contraction of the entire network *except* for $T_{0,j}$. The value of $A_j$ depends on all other inputs, denoted $w_{\setminus j}$.

Now, let's compute the marginal sum over all possible symbols for position $j$:

$$\begin{aligned}
\sum_{c\in\Sigma} \mathcal{T}(w_{\setminus j}, w_j = c) &= \sum_{c\in\Sigma} \langle A_j(w_{\setminus j}), \mathbf{e}_c \rangle \\
&= \langle A_j(w_{\setminus j}), \sum_{c\in\Sigma} \mathbf{e}_c \rangle \quad \text{(by linearity of the inner product)} \\
&= \langle A_j(w_{\setminus j}), \mathbf{1} \rangle \quad ,
\end{aligned}$$

where $\mathbf{e}_c$ is the one-hot vector for symbol $c$ and $\mathbf{1}$ is the all-ones vector. This last expression is precisely the definition of $\mathcal{T}_{\{j\}}(w_{\setminus j})$, where the input at position $j$ has been replaced by the all-ones vector.

The proposition follows by applying this argument repeatedly for each $j \in J$. □

**Corollary 1.1.** *Let $\mathcal{T}$ be a tensor network over $\Sigma^n$, and let $\mathcal{T}_{[n]}$ be the network where all input tensors are initialized to the all-ones vector. Then:*

$$\mathcal{T} \text{ is normalized} \iff \mathcal{T}_{[n]} = 1 \quad ,$$

*i.e., the total contraction of the network with all-one input tensors equals 1.*

**Lemma 1.1.** *Let $\mathcal{T}$ be a tensor network over $\Sigma^n$. The induced normalized tensor network $\frac{\mathcal{T}}{|\mathcal{T}|}$ is indeed normalized, and if additionally $\mathcal{T}$ is non-negative and $f \equiv id$, we have for all $w \in \Sigma^n$:*

$$S_{n,\mathcal{T}}(w) = S_{n,\frac{\mathcal{T}}{|\mathcal{T}|}}(w) \quad .$$

*Proof.* Let $H$ be the set of non-input tensors in $\mathcal{T}$, and let $|H| = m$. In the induced normalized network, every tensor in $H$ is scaled by a factor $\alpha = \dfrac{1}{\sqrt[m]{|\mathcal{T}|}}$. Since the final output $\mathcal{T}(w)$ is a multilinear contraction over the tensors, this means the scalar output for any $w \in \Sigma^n$ becomes:

$$\left(\prod_{v \in H} \alpha\right) \cdot \mathcal{T}(w) = \alpha^m \cdot \mathcal{T}(w) = \frac{1}{|\mathcal{T}|} \cdot \mathcal{T}(w) \quad .$$

Hence,

$$\left(\frac{\mathcal{T}}{|\mathcal{T}|}\right)(w) = \frac{\mathcal{T}(w)}{|\mathcal{T}|} \quad .$$

Summing over all $w \in \Sigma^n$,

$$\left|\frac{\mathcal{T}}{|\mathcal{T}|}\right| = \sum_{w \in \Sigma^n} \frac{\mathcal{T}(w)}{|\mathcal{T}|} = \frac{1}{|\mathcal{T}|} \sum_{w \in \Sigma^n} \mathcal{T}(w) = \frac{|\mathcal{T}|}{|\mathcal{T}|} = 1 \quad .$$

Moreover, since the normalization rescales all outputs by the same constant, the ratio of the terms to the total sum remains unchanged:

$$S_{n,\frac{\mathcal{T}}{|\mathcal{T}|}}(w) = \frac{\left(\frac{\mathcal{T}(w)}{|\mathcal{T}|}\right)}{\sum_{w' \in \Sigma^n}\left(\frac{\mathcal{T}(w')}{|\mathcal{T}|}\right)} = \frac{\mathcal{T}(w)}{|\mathcal{T}|} \cdot \frac{1}{1} = S_{n,\mathcal{T}}(w) \quad .$$

This completes the proof. □

One might ask whether our definition for tensor networks is bit restrictive, as it only allows for contraction over *pairs* of tensors. But what if we wanted to contract, say, three tensors at once over a common index?

**Proposition 1.2.** *Let $V' \subseteq V$ be a set of tensors in a tensor network, each containing an axis of dimension $d$ labeled by a shared index $i$. Contracting all tensors in $V'$ over the shared index $i$ is equivalent to contracting each tensor individually with a single tensor*

$$\delta_{|V'|} : [d]^{|V'|} \mapsto \mathbb{R}_{\geq 0}$$

*defined by*

$$\delta_{|V'|}(i_1, \ldots, i_{|V'|}) = \begin{cases} 1 & \text{if } i_1 = \cdots = i_{|V'|} \ , \\ 0 & \text{otherwise.} \end{cases} \quad .$$

*That is, a full contraction over a shared index can be implemented by introducing a single copy tensor connected to each tensor in $V'$.*

*Proof.* Each tensor $T_v$ for $v \in V'$ has an index $i \in [d]$ corresponding to the shared axis. The contraction over this index is defined by summing over the common value of $i$ across all tensors:

$$\sum_{i=1}^{d} \prod_{v \in V'} T_v(\ldots, i, \ldots) \quad .$$

Now consider a new tensor $\delta_{|V'|}$ of order $|V'|$, defined as 1 if all indices are equal and 0 otherwise. Let each tensor $T_v$ maintain its original indices, but connect to $\delta_{|V'|}$ via the position corresponding to $v$.

The contraction over this shared structure gives:

$$\sum_{i_1, \ldots, i_{|V'|}} \left( \prod_{v \in V'} T_v(\ldots, i_v, \ldots) \right) \delta_{|V'|}(i_1, \ldots, i_{|V'|}) \quad .$$

By definition of $\delta_{|V'|}$, this enforces $i_1 = \cdots = i_{|V'|}$, reducing the above to:

$$\sum_{i=1}^{d} \prod_{v \in V'} T_v(\ldots, i, \ldots) \quad ,$$

which is exactly the original contraction. Hence, the two constructions are equivalent. □

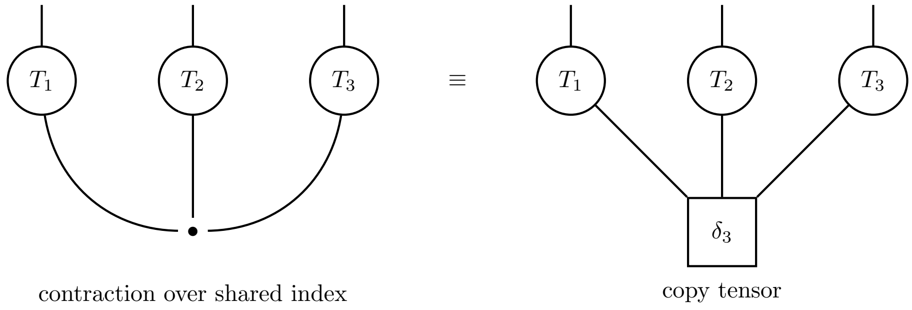

Figure 2: Contracting multiple tensors over one shared index is equivalent to contracting them individually with a single copy tensor.

## 1.1 Reshaping a Tensor (Matricization)

An order-$n$ tensor can be rearranged into a matrix by partitioning its indices into two disjoint sets. This operation, commonly known as *matricization*, *unfolding*, or *flattening*, is fundamental for applying linear algebra tools to higher-order tensors. We denote the matricization of a tensor $T$ with respect to an index partition $(\mathcal{I}_A, \mathcal{I}_B)$ as $[[T]]_{\mathcal{I}_A,\mathcal{I}_B}$.

**Definition 1.4** (Tensor Matricization)**.** Let $T \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_n}$ be an order-$n$ tensor with index set $\mathcal{I} = \{1, 2, \ldots, n\}$. Let $(\mathcal{I}_A, \mathcal{I}_B)$ be a bipartition of $\mathcal{I}$, such that $\mathcal{I}_A \cup \mathcal{I}_B = \mathcal{I}$ and $\mathcal{I}_A \cap \mathcal{I}_B = \emptyset$.

The matricization of $T$ with respect to this partition is the matrix $\boldsymbol{M} = [[T]]_{\mathcal{I}_A,\mathcal{I}_B}$, of size $D_A \times D_B$, where:

- The number of rows is $D_A = \prod_{i \in \mathcal{I}_A} d_i$.
- The number of columns is $D_B = \prod_{j \in \mathcal{I}_B} d_j$.

The elements of $\boldsymbol{M}$ are obtained by mapping the elements of $T$. Specifically, an element $T_{i_1,i_2,\ldots,i_n}$ is mapped to an element $\boldsymbol{M}_{\alpha,\beta}$, where the row index $\alpha$ is determined by the indices $\{i_k\}_{k \in \mathcal{I}_A}$ and the column index $\beta$ is determined by $\{i_k\}_{k \in \mathcal{I}_B}$.

To define this mapping explicitly, let us fix an ordering for the indices within the partitions, such that $\mathcal{I}_A = \{a_1, \ldots, a_{|\mathcal{I}_A|}\}$ and $\mathcal{I}_B = \{b_1, \ldots, b_{|\mathcal{I}_B|}\}$. The multi-indices are then mapped to scalar indices, typically in lexicographical order, as follows:

$$\alpha = 1 + \sum_{j=1}^{|\mathcal{I}_A|} (i_{a_j} - 1) \left( \prod_{l=j+1}^{|\mathcal{I}_A|} d_{a_l} \right)$$

$$\beta = 1 + \sum_{j=1}^{|\mathcal{I}_B|} (i_{b_j} - 1) \left( \prod_{l=j+1}^{|\mathcal{I}_B|} d_{b_l} \right)$$

### 1.1.1 An Upper Bound for the Rank of Matrices in Tensor Networks

We can analyze these matrices $[[T]]_{\mathcal{I}_A,\mathcal{I}_B}$ which depend on the partition $\{\mathcal{I}_A, \mathcal{I}_B\}$ of the indices of our tensor network. When picturing the network as a graph, we can separate the indices in $\mathcal{I}_A$ from the ones in $\mathcal{I}_B$ with a cut. Metaphorically, we see that $\mathcal{I}_A$ and $\mathcal{I}_B$ *communicate* over this cut. Hence, we might assume that if the bond dimensions of the cut are small, so is the mutual information between the two associated sets of random variables. In fact, this intuition is backed up by mathematical evidence:

**Theorem 1.1.** *Let $T$ be a tensor represented by a tensor network over a set of physical indices $\mathcal{I} = \{i_1, \ldots, i_N\}$. For any bipartition of these indices into sets $\mathcal{I}_A$ and $\mathcal{I}_B$, which induces a cut through a set of internal bonds $E_{cut} = \{e_1, \ldots, e_c\}$ with dimensions $\{d_1, \ldots, d_c\}$, the rank of the reshaped matrix $\boldsymbol{M} \coloneqq [[T]]_{A,B}$ corresponding to this partition is bounded by:*

$$\operatorname{rank} \boldsymbol{M} \leq \prod_{k=1}^{c} d_k \quad .$$

*Proof.* A tensor network represents a tensor $T$ by contracting a set of smaller tensors. Let's visualize a generic tensor network. A bipartition of the physical indices $(\mathcal{I}_A, \mathcal{I}_B)$ divides the graph of the network into two subgraphs.

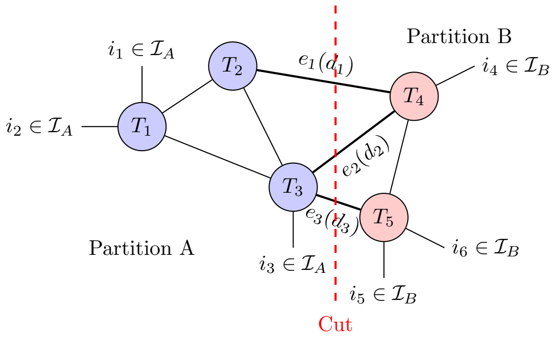

Figure 3: A generic tensor network with a cut partitioning the physical indices into $\mathcal{I}_A$ (blue) and $\mathcal{I}_B$ (red). The cut crosses bonds $e_1, e_2, e_3$.

We can group all tensors on the left of the cut (Partition A) and contract their internal bonds among themselves. This results in a single, large tensor, which we call $U$. The indices of $U$ are the physical indices in $\mathcal{I}_A$ and the cut bond indices $\{e_1, \ldots, e_c\}$. Likewise, we contract all tensors in Partition B to form a tensor $V$, with indices from $\mathcal{I}_B$ and $\{e_1, \ldots, e_c\}$.

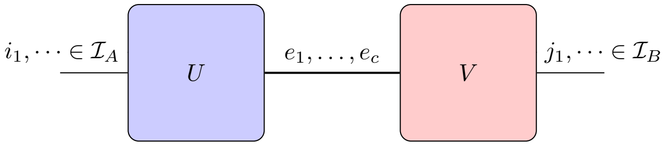
$i_1, \cdots \in \mathcal{I}_A$
$U$
$e_1, \ldots, e_c$
$V$
$j_1, \cdots \in \mathcal{I}_B$

Figure 4: Contracting the subgraphs results in two tensors, $U$ and $V$, connected by the cut bonds.

The original tensor $\mathcal{T}$ is recovered by contracting $U$ and $V$ over the cut indices. Now, we reshape $U$ and $V$ into matrices. Let $\alpha$ be a multi-index that combines all cut indices $\{e_1, \ldots, e_c\}$. The dimension of this multi-index is $D_{cut} = \prod_{k=1}^{c} d_k$. Let $I_A$ be the multi-index for all physical indices in $\mathcal{I}_A$. Let $I_B$ be the multi-index for all physical indices in $\mathcal{I}_B$.

We reshape the tensor $U$ into a matrix $\mathbf{U}$ of size $(\dim(I_A) \times D_{cut})$ where the rows are indexed by $I_A$ and the columns by $\alpha$. We reshape the tensor $V$ into a matrix $\mathbf{V}$ of size $(D_{cut} \times \dim(I_B))$ where the rows are indexed by $\alpha$ and the columns by $I_B$.

The reshaped tensor matrix $\boldsymbol{M}$ (with rows $I_A$ and columns $I_B$) is exactly the matrix product of $\mathbf{U}$ and $\mathbf{V}$:

$$\boldsymbol{M} = \mathbf{U} \cdot \mathbf{V} \quad .$$

Note that the rank of a product of matrices is at most the minimum of their individual ranks:

$$\operatorname{rank} \boldsymbol{M} = \operatorname{rank}(\mathbf{U} \cdot \mathbf{V}) \leq \min\{\operatorname{rank} \mathbf{U}, \operatorname{rank} \mathbf{V}\}$$

Furthermore, since $\operatorname{rank} \mathbf{U} \leq D_{\text{cut}}$ and $\operatorname{rank} \mathbf{V} \leq D_{\text{cut}}$, it directly follows that:

$$\operatorname{rank} \boldsymbol{M} \leq D_{\text{cut}} = \prod_{k=1}^{c} d_k \quad .$$

□

**Remark 1.2.** Such a cut that minimizes the product of the bond dimensions can be found by a standard min-cut algorithm using the logarithms of the axes sizes.

**Remark 1.3.** Combining this theorem with theorem **??**, we see that the mutual information between two blocks of random variables is upper bounded by the weight of the min-cut, where the values of the edges are set to the logarithm of the associated axes sizes.