Attention via $\log \sum \exp$ energy

Alexander Tschantz

January 20, 2025

1 General Framework

We consider a directed graph of A nodes, where each node is a vector $\{\boldsymbol{v}_a\}_{a=1}^{A}$ with $\boldsymbol{v}_a \in \mathbb{R}^d$. Each node $\boldsymbol{v}_a$ has a set of parents $\mathcal{P}(a) \subseteq \{1, 2, \ldots, A\}$.

We define a generic *similarity* function

$$\mathrm{sim}\big(\boldsymbol{v}_a, \boldsymbol{v}_p\big),$$

which measures how well $\boldsymbol{v}_a$ is "explained by" or "aligned with" its parent $\boldsymbol{v}_p$. This function may have parameters, as in a dot-product model $\boldsymbol{v}_a^\top \boldsymbol{v}_p$, or a conditional-probability-based model such as a Gaussian log-likelihood term $-\frac{1}{2}(\boldsymbol{v}_a - \boldsymbol{v}_p)^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{v}_a - \boldsymbol{v}_p)$.

1.1 Energy Function

Each node $\boldsymbol{v}_a$ has a $\log \sum \exp$-type term over its parents. We sum these across all nodes to define the total energy:

$$E\big(\{\boldsymbol{v}_a\}\big) \;=\; -\sum_{a=1}^{A} \ln\Big(\sum_{p \,\in\, \mathcal{P}(a)} \exp\big(\mathrm{sim}(\boldsymbol{v}_a, \boldsymbol{v}_p)\big)\Big).$$

Informally, each $\boldsymbol{v}_a$ seeks to place high mass on those parents $\boldsymbol{v}_p$ yielding large similarity scores.

1.2 Gradient Updates

For a single node $\boldsymbol{v}_a$, the gradient of the energy decomposes into two parts, reflecting two ways in which $\boldsymbol{v}_a$ can appear in the summations:

$$-\frac{\partial E}{\partial \boldsymbol{v}_a} \;=\; \underbrace{\sum_{p \in \mathcal{P}(a)} \mathrm{softmax}_p\Big(\mathrm{sim}(\boldsymbol{v}_a, \boldsymbol{v}_p)\Big) \frac{\partial}{\partial \boldsymbol{v}_a} \mathrm{sim}\big(\boldsymbol{v}_a, \boldsymbol{v}_p\big)}_{\text{(``being explained by its parents'')}} + \underbrace{\sum_{c \,:\; a \in \mathcal{P}(c)} \mathrm{softmax}_a\Big(\mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_a)\Big) \frac{\partial}{\partial \boldsymbol{v}_a} \mathrm{sim}\big(\boldsymbol{v}_c, \boldsymbol{v}_a\big)}_{\text{(``explaining its children'')}}.$$

Above, the summation $\sum_{p \in \mathcal{P}(a)}$ iterates over the parents of a, while $\sum_{c:a \in \mathcal{P}(c)}$ iterates over all children c such that a is in their parent set. The softmax is taken over the appropriate parent indices in each case.

1.3 Proof of Gradient (Appendix)

A full derivation, with explicit sums over $\dashv \in \mathcal{P}(c)$, is given in Appendix A. There we show how collecting terms in the derivative leads precisely to the two-term decomposition above.

2 Gaussian Mixture Models (GMMs)

Setup. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$, where each $\boldsymbol{x}_i \in \mathbb{R}^d$. We consider K mixture components, each with mean $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and covariance $\boldsymbol{\Sigma}_k$. Defining π_k as the mixing proportion, a standard GMM log-likelihood term can be written in a form that matches our framework.

In particular, define:

$$\boldsymbol{A}_{ik} \;=\; \ln \pi_k \;-\; \tfrac{1}{2}\,(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \quad \in \; \mathbb{R}^{N \times K}.$$

Energy.

$$E^{\text{GMM}}(\boldsymbol{X}, \{\boldsymbol{\mu}_k\}) \;=\; -\sum_{i=1}^{N} \ln\Bigl(\sum_{k=1}^{K} \exp\bigl(\boldsymbol{A}_{ik}\bigr)\Bigr).$$

Gradient. If we differentiate w.r.t. $\boldsymbol{\mu}_k$, then

$$-\frac{\partial E^{\text{GMM}}}{\partial \boldsymbol{\mu}_k} \;=\; \sum_{i=1}^{N} \text{softmax}_k\bigl(\boldsymbol{A}_{ik}\bigr)\; \boldsymbol{\Sigma}_k^{-1}\bigl(\boldsymbol{x}_i - \boldsymbol{\mu}_k\bigr).$$

Setting this gradient to zero yields the usual GMM M-step:

$$\boldsymbol{\mu}_k \;=\; \frac{\sum_{i=1}^{N} \text{softmax}_k\bigl(\boldsymbol{A}_{ik}\bigr)\; \boldsymbol{x}_i}{\sum_{i=1}^{N} \text{softmax}_k\bigl(\boldsymbol{A}_{ik}\bigr)}.$$

3 Self-Attention

Setup. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$, where each $\boldsymbol{x}_i \in \mathbb{R}^d$. We define two learnable weight matrices, $\boldsymbol{W}^Q, \boldsymbol{W}^K \in \mathbb{R}^{d\times d}$, and construct

$$\boldsymbol{Q} \;=\; \boldsymbol{W}^Q\, \boldsymbol{X}, \quad \boldsymbol{K} \;=\; \boldsymbol{W}^K\, \boldsymbol{X}.$$

Let $\boldsymbol{q}_c$ be the c-th column of $\boldsymbol{Q}$ and $\boldsymbol{k}_b$ the b-th column of $\boldsymbol{K}$. Then we define

$$\text{sim}\bigl(\boldsymbol{x}_b, \boldsymbol{x}_c\bigr) \;\widehat{=}\; \boldsymbol{k}_b^\top \boldsymbol{q}_c.$$

On a single line, we gather these into the matrix:

$$\boldsymbol{A}_{bc} = \boldsymbol{k}_b^\top \boldsymbol{q}_c \quad \in\; \mathbb{R}^{N\times N}.$$

Energy.

$$E^{\text{SA}}(\boldsymbol{X}) \;=\; -\sum_{c=1}^{N} \ln\Bigl(\sum_{b=1}^{N} \exp\bigl(\boldsymbol{A}_{bc}\bigr)\Bigr).$$

Gradient. Differentiating w.r.t. $\boldsymbol{x}_i$ gives two terms, as each token $\boldsymbol{x}_i$ acts both as a "query" for some other tokens and a "key" for yet others:

$$-\frac{\partial E^{\text{SA}}}{\partial \boldsymbol{x}_i} \;=\; \underbrace{\sum_{b=1}^{N} \text{softmax}_b\bigl(\boldsymbol{A}_{b,i}\bigr)\, \boldsymbol{W}_Q^\top \boldsymbol{W}_K\, \boldsymbol{x}_b}_{\text{(query side)}} \;+\; \underbrace{\sum_{c=1}^{N} \text{softmax}_i\bigl(\boldsymbol{A}_{i,c}\bigr)\, \boldsymbol{W}_K^\top \boldsymbol{W}_Q\, \boldsymbol{x}_c}_{\text{(key side)}}.$$

4 Cross Attention

Setup. In cross attention, we have one set of *query* vectors and a separate set of *key* vectors. Let $\boldsymbol{Q} = \boldsymbol{W}^Q\, \boldsymbol{X}^Q \in \mathbb{R}^{d\times N_Q}$ and $\boldsymbol{K} = \boldsymbol{W}^K\, \boldsymbol{X}^K \in \mathbb{R}^{d\times N_K}$, where $\boldsymbol{X}^Q$ has N_Q query tokens and $\boldsymbol{X}^K$ has N_K key tokens. Denote $\boldsymbol{q}_c$ as the c-th query column of $\boldsymbol{Q}$ and $\boldsymbol{k}_b$ as the b-th key column of $\boldsymbol{K}$.

We define

$$\boldsymbol{A}_{b,c} \;=\; \boldsymbol{k}_b^\top\, \boldsymbol{q}_c \quad \in\; \mathbb{R}^{N_K\times N_Q}.$$

This is a matrix of pairwise similarities between keys and queries.

Energy.

$$E^{\text{Cross}}(\boldsymbol{X}^Q, \boldsymbol{X}^K) \;=\; -\sum_{c=1}^{N_Q} \ln\Bigl(\sum_{b=1}^{N_K} \exp\bigl(\boldsymbol{A}_{b,c}\bigr)\Bigr).$$

Minimizing this encourages each query $\boldsymbol{q}_c$ to place large mass on keys $\boldsymbol{k}_b$ that yield higher dot-products.

Gradient. As in the self-attention derivation, taking the derivative w.r.t. a single query or key token yields sums weighted by the appropriate softmax terms. For example, w.r.t. the query-side vector $\boldsymbol{x}_i^Q$,

$$-\frac{\partial E^{\text{Cross}}}{\partial \boldsymbol{x}_i^Q} \;=\; \sum_{b=1}^{N_K} \text{softmax}_b\big(\boldsymbol{A}_{b,i}\big)\, \boldsymbol{W}_Q^\top\, \boldsymbol{W}_K\, \boldsymbol{x}_b^K \;+\; \dots$$

and similarly a key $\boldsymbol{x}_j^K$ appears in the "explaining children" part of the gradient.

5 Hopfield Networks (Softmax Version)

Setup. We have data vectors $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N]$, each $\boldsymbol{x}_i \in \mathbb{R}^d$, and memory vectors $\boldsymbol{m}_\mu \in \mathbb{R}^d$ for $\mu = 1, \dots, K$. Define

$$\boldsymbol{A}_{i\mu} \;=\; \boldsymbol{x}_i^\top\, \boldsymbol{m}_\mu \quad \in \mathbb{R}^{N \times K}.$$

Energy.

$$E^{\text{Hopfield}}(\boldsymbol{X}) \;=\; -\sum_{i=1}^{N} \ln\Big(\sum_{\mu=1}^{K} \exp\big(\boldsymbol{A}_{i\mu}\big)\Big).$$

Gradient.

$$-\frac{\partial E^{\text{Hopfield}}}{\partial \boldsymbol{x}_i} \;=\; \sum_{\mu=1}^{K} \text{softmax}_\mu\big(\boldsymbol{A}_{i\mu}\big)\; \boldsymbol{m}_\mu.$$

Hence each $\boldsymbol{x}_i$ is updated toward a softmax-weighted combination of the memory vectors.

6 Slot Attention

Slot Attention can be seen as cross attention in which we normalize across slots (queries) for each token (key), rather than the usual normalization across the token dimension.

Setup. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N]$ be the set of tokens (e.g. image patches), where each $\boldsymbol{x}_j \in \mathbb{R}^d$. We also have a set of S latent "slots": $\boldsymbol{\mu}_i \in \mathbb{R}^d$ for $i = 1, \dots, S$. As in cross attention, we define learnable transforms:

$$\boldsymbol{W}_K,\; \boldsymbol{W}_Q \;\in \mathbb{R}^{d \times d}.$$

The tokens serve as keys (and potentially values), while the slots serve as queries. However, the key difference is that *each token decides its distribution over slots* (hence the normalization is over i for each fixed token j).

Energy. We write the negative log-likelihood as a sum over tokens $j = 1, \dots, N$, and in each term we do a $\log \sum \exp$ over the slots $i = 1, \dots, S$. Concretely,

$$E^{\text{Slot}}\big(\{\boldsymbol{\mu}_i\}\big) \;=\; -\sum_{j=1}^{N} \ln\Big(\sum_{i=1}^{S} \exp\big(\text{sim}(\boldsymbol{x}_j, \boldsymbol{\mu}_i)\big)\Big),$$

where $\text{sim}(\boldsymbol{x}_j, \boldsymbol{\mu}_i) = \big(\boldsymbol{W}_K\, \boldsymbol{x}_j\big)^\top \big(\boldsymbol{W}_Q\, \boldsymbol{\mu}_i\big)$.

$$\boldsymbol{A}_{j,i} = \big(\boldsymbol{W}_K\, \boldsymbol{x}_j\big)^\top \big(\boldsymbol{W}_Q\, \boldsymbol{\mu}_i\big) \quad \in \mathbb{R}^{N \times S}.$$

Gradient. Taking the derivative w.r.t. a single slot $\boldsymbol{\mu}_i$ yields

$$-\frac{\partial E^{\text{Slot}}}{\partial \boldsymbol{\mu}_i} \;=\; \sum_{j=1}^{N} \text{softmax}_i\big(\boldsymbol{A}_{j,i}\big)\; \boldsymbol{W}_Q^\top\, \boldsymbol{W}_K\, \boldsymbol{x}_j.$$

Hence each slot $\boldsymbol{\mu}_i$ aggregates information from all tokens j, but the weight is proportional to $\exp(\boldsymbol{A}_{j,i})$ normalized across *the slots* i. This produces the usual iterative update rule:

$$\boldsymbol{\mu}_i^* \;=\; \sum_{j=1}^{N} \mathrm{softmax}_i\Big(\boldsymbol{\mu}_i^\top \boldsymbol{W}_Q^\top \boldsymbol{W}_K \,\boldsymbol{x}_j\Big)\; \boldsymbol{W}_Q^\top \,\boldsymbol{W}_K \,\boldsymbol{x}_j,$$

which is sometimes referred to as *inverted cross attention.*

A Proof of the Gradient Decomposition

For completeness, we provide a short derivation of the gradient expression. Recall that our energy is

$$E\big(\{\boldsymbol{v}_a\}\big) \;=\; -\sum_{c=1}^{A} \ln\Big(\sum_{\dashv\,\in\,\mathcal{P}(c)} \exp\big(\mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_\dashv)\big)\Big).$$

Differentiating w.r.t. $\boldsymbol{v}_a$:

$$\frac{\partial E}{\partial \boldsymbol{v}_a} \;=\; -\sum_{c=1}^{A} \frac{\partial}{\partial \boldsymbol{v}_a} \ln\Big(\sum_{\dashv\,\in\,\mathcal{P}(c)} \exp\big(\mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_\dashv)\big)\Big).$$

Inside the sum, only terms $\mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_\dashv)$ with $\dashv = a$ or $c = a$ will contribute. Carefully extracting these leads to the "being explained by parents" plus "explaining children" split in the main text:

$$-\frac{\partial E}{\partial \boldsymbol{v}_a} \;=\; \sum_{\dashv\,\in\,\mathcal{P}(a)} \mathrm{softmax}_\dashv\big(\mathrm{sim}(\boldsymbol{v}_a, \boldsymbol{v}_\dashv)\big)\, \frac{\partial\, \mathrm{sim}(\boldsymbol{v}_a, \boldsymbol{v}_\dashv)}{\partial \boldsymbol{v}_a} \;+\; \sum_{\substack{c=1\\ a\in\mathcal{P}(c)}}^{A} \mathrm{softmax}_a\big(\mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_a)\big)\, \frac{\partial\, \mathrm{sim}(\boldsymbol{v}_c, \boldsymbol{v}_a)}{\partial \boldsymbol{v}_a}.$$