pRSL: Interpretable Multi–label Stacking by Learning Probabilistic Rules Supplementary Material

Michael Kirchhof[1] **Lena Schmid**[1] **Christopher Reining**[2] **Michael ten Hompel**[2] **Markus Pauly**[1]

[1]Department of Statistics, TU Dortmund University, Dortmund, Germany
[2]Chair of Material Handling and Warehousing, TU Dortmund University, Dortmund, Germany

1 GRADIENTS

To perform gradient descent, we differentiate the joint log likelihood of the true categories $\boldsymbol{\ell}^* = (\ell_1^*, \dots, \ell_J^*)$ by the noisy–or's inhibition probabilities q_{jm}^k for all rules $k = 1, \dots, K$, and all $m = 1, \dots, M(j)$ categories of labels $j = 1, \dots, J$. To simplify notation, we define

$$
\begin{aligned}
\boldsymbol{R}_{-k} :=& (R_1, \dots, R_{k-1}, R_{k+1}, \dots, R_K),\\
\boldsymbol{L}_{-\boldsymbol{j}} :=& (L_1, \dots, L_{j-1}, L_{j+1} \dots, L_J),\\
\boldsymbol{\ell}_{-j} :=& (\ell_1, \dots, \ell_{j-1}, \ell_{j+1} \dots, \ell_J),\\
\boldsymbol{\ell}_{-j}^* :=& (\ell_1^*, \dots, \ell_{j-1}^*, \ell_{j+1}^* \dots, \ell_J^*),\\
\mathcal{L}_{j=m} :=& \bigotimes_{v=1}^{j-1} \{1, \dots, M(v)\} \otimes\\
& \{m\} \otimes \bigotimes_{v=j+1}^{J} \{1, \dots, M(v)\},\\
\mathcal{L}_{j\neq m} :=& \bigotimes_{v=1}^{j-1} \{1, \dots, M(v)\} \otimes\\
& \{1, \dots, m-1, m+1, \dots, M(j)\} \otimes\\
& \bigotimes_{v=j+1}^{J} \{1, \dots, M(v)\} \text{ and}\\
q_{-j\ell}^k :=& \prod_{v=1}^{j-1} q_{v\ell_v}^k \cdot \prod_{v=j+1}^{J} q_{v\ell_v}^k \text{ and}\\
q_{-j\ell^*}^k :=& \prod_{v=1}^{j-1} q_{v\ell_v^*}^k \cdot \prod_{v=j+1}^{J} q_{v\ell_v^*}^k \ .
\end{aligned}
$$

1.1 ALL LABELS KNOWN

As discussed in the paper, we first differentiate in the case where all true categories $\ell_1^*, \dots, \ell_J^*$ are known.

Case 1: Category is incorrect. We start for those q_{jm}^k where m is not the true category, that is $\ell_j^* \neq m$:

$$
\begin{aligned}
D_{jm}^k :=& \frac{\partial}{\partial q_{jm}^k} P(\boldsymbol{L} = \boldsymbol{\ell}^* | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x})\\
=& \frac{\partial}{\partial q_{jm}^k} \frac{P(R_k = 1, \boldsymbol{L} = \boldsymbol{\ell}^* | \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x})}{P(R_k = 1 | \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x})} \ .
\end{aligned}
$$

From here on, all probabilities stay conditioned on $\boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x}$, so we will write this condition behind the expression in the following formulas.

$$
\begin{aligned}
D_{jm}^k =& \frac{\partial}{\partial q_{jm}^k} \frac{P(R_k = 1, \boldsymbol{L} = \boldsymbol{\ell}^*)}{\sum_{\boldsymbol{\ell} \in \mathcal{L}} (1 - q_{j\ell_j}^k q_{-j\ell}^k) P(\boldsymbol{L} = \boldsymbol{\ell})} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x}\\
=& \frac{\partial}{\partial q_{jm}^k} (P(R_k = 1, \boldsymbol{L} = \boldsymbol{\ell}^*)) \cdot\\
& \left(\sum_{\boldsymbol{\ell} \in \mathcal{L}} P(\boldsymbol{L} = \boldsymbol{\ell}) - \sum_{\boldsymbol{\ell} \in \mathcal{L}_{j\neq m}} q_{j\ell_j}^k q_{-j\ell}^k P(\boldsymbol{L} = \boldsymbol{\ell}) - \right.\\
& \left. q_{jm}^k \sum_{\boldsymbol{\ell} \in \mathcal{L}_{j=m}} q_{-j\ell}^k P(\boldsymbol{L} = \boldsymbol{\ell}) \right)^{-1} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \ .
\end{aligned}
$$

Simplifying the notation, the above expression can be written as:

$$
\begin{aligned}
D_{jm}^k &= \frac{\partial}{\partial q_{jm}^k} \frac{a_1}{a_2 - q_{jm}^k a_3} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x}\\
&= \frac{a_1 a_3}{(a_2 - q_{jm}^k a_3)^2} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x}
\end{aligned}
$$

Supplement for the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021).

for suitable choices of a_1, a_2, and a_3. Substituting back and multiplying with $q^k_{jm}(q^k_{jm})^{-1}$ we get:

$$\begin{aligned}D^k_{jm} =&P(R_k=1,\boldsymbol{L}=\boldsymbol{\ell}^*)\cdot\\ &\frac{\sum\limits_{\boldsymbol{\ell}\in\mathcal{L}_{j=m}} q^k_{jm}q^k_{-j\ell}P(\boldsymbol{L}=\boldsymbol{\ell})}{q^k_{jm}P(R_k=1)^2}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ =&P(R_k=1,\boldsymbol{L}=\boldsymbol{\ell}^*)\cdot\\ &\frac{P(L_j=m,R_k=0)}{q^k_{jm}P(R_k=1)^2}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

The first term in the nominator is computationally complex and might get numerically instable with an increasing number of labels J, but since we optimize for the log–likelihood, it vanishes via the chain rule:

$$\begin{aligned}&\frac{\partial}{\partial q^k_{jm}}\log(P(\boldsymbol{L}=\boldsymbol{\ell}^*|\boldsymbol{R}=\mathbf{1},\boldsymbol{x}))\\ =&\frac{1}{P(\boldsymbol{L}=\boldsymbol{\ell}^*|\boldsymbol{R}=\mathbf{1},\boldsymbol{x})}D^k_{jm}\\ =&\frac{P(R_k=1)}{P(\boldsymbol{L}=\boldsymbol{\ell}^*,R_k=1)}\cdot\\ &P(R_k=1,\boldsymbol{L}=\boldsymbol{\ell}^*)\frac{P(L_j=m,R_k=0)}{q^k_{jm}P(R_k=1)^2}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ =&\frac{P(L_j=m,R_k=0)}{q^k_{jm}P(R_k=1)}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ =&\frac{P(L_j=m|R_k=0)(1-P(R_k=1))}{q^k_{jm}P(R_k=1)}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

Note that $P(L_j=m|R_k=0)$ is returned for all categories $m=1,\ldots,M(j)$ of all labels $L_j, j=1,\ldots,J$, by a single marginal query. Hence, two marginal queries to the network are sufficient to compute the gradient of all inhibition probabilities related to a rule.

Case 2: Category is correct. Let us now differentiate for q^k_{jm} where m is the true category, that is $\ell^*_j=m$:

$$\begin{aligned}D^k_{jm} =&\frac{\partial}{\partial q^k_{jm}}P(\boldsymbol{L}=\boldsymbol{\ell}^*|\boldsymbol{R}=\mathbf{1},\boldsymbol{x})\\ =&\frac{\partial}{\partial q^k_{jm}}(P(\boldsymbol{L}=\boldsymbol{\ell}^*)-q^k_{j\ell^*_j}q^k_{-j\ell^*}P(\boldsymbol{L}=\boldsymbol{\ell}^*))\cdot\\ &\left(\sum_{\boldsymbol{\ell}\in\mathcal{L}}P(\boldsymbol{L}=\boldsymbol{\ell})-\sum_{\boldsymbol{\ell}\in\mathcal{L}_{j\neq m}}q^k_{j\ell_j}q^k_{-j\ell}P(\boldsymbol{L}=\boldsymbol{\ell})-\right.\\ &\left.q^k_{jm}\sum_{\boldsymbol{\ell}\in\mathcal{L}_{j=m}}q^k_{-j\ell}P(\boldsymbol{L}=\boldsymbol{\ell})\right)^{-1}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

Again, we simplify the notation to make differentiation easier to see:

$$\begin{aligned}&=\frac{\partial}{\partial q^k_{jm}}\frac{b_1-q^k_{jm}b_2}{b_3-q^k_{jm}b_4}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=\frac{b_1b_4-b_2b_3}{(b_3-q^k_{jm}b_4)^2}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

Substituting back and multiplying by $q^k_{jm}(q^k_{jm})^{-1}$, we get:

$$\begin{aligned}&=\frac{P(\boldsymbol{L}=\boldsymbol{\ell}^*)(q^k_{jm}b_4-q^k_{jm}q^k_{-j\ell}b_3)}{q^k_{jm}P(R_k=1)^2}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{l}^*)(P(R_k=0,L_j=m)-\\ &\ q^k_{jm}q^k_{-j\ell}(P(L_j=m)+P(L_j\neq m)-\\ &\ P(R_k=0,L_j\neq m))))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{\ell}^*)(P(R_k=0,L_j=m)-\\ &\ q^k_{jm}q^k_{-j\ell}(P(L_j=m)+\\ &\ P(R_k=1,L_j\neq m))))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{\ell}^*)(P(R_k=0,L_j=m)-\\ &\ q^k_{jm}q^k_{-j\ell}(P(R_k=0,L_j=m)+P(R_k=1,L_j=m)+\\ &\ P(R_k=1,L_j\neq m))))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{\ell}^*)(P(R_k=0,L_j=m)-\\ &\ q^k_{jm}q^k_{-j\ell}(P(R_k=0,L_j=m)\\ &\ +P(R_k=1))))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{l}^*)((1-q^k_{jm}q^k_{-j\ell})P(R_k=0,L_j=m)-\\ &\ q^k_{jm}q^k_{-j\ell}P(R_k=1)))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\\ &=(q^k_{jm}P(R_k=1)^2)^{-1}\cdot\\ &\ (P(\boldsymbol{L}=\boldsymbol{\ell}^*,R_k=1)P(R_k=0,L_j=m)-\\ &\ P(\boldsymbol{L}=\boldsymbol{\ell}^*)q^k_{jm}q^k_{-j\ell}P(R_k=1)))\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

Just as before, the computational heavy terms cancel out when we optimize for log–likelihood:

$$\begin{aligned}&\frac{\partial}{\partial q^k_{jm}}\log(P(\boldsymbol{L}=\boldsymbol{l}^*|\boldsymbol{R}=\mathbf{1},\boldsymbol{x}))\\ =&\frac{P(L_j=m|R_k=0)(1-P(R_k=1))}{q^k_{jm}P(R_k=1)}-\\ &\frac{q^k_{-j\ell^*}}{1-q^k_{jm}q^k_{-j\ell^*}}\mid \boldsymbol{R}_{-k}=\mathbf{1},\boldsymbol{x}\ .\end{aligned}$$

This term is similar to the previous gradient and requires the same marginal queries.

1.2 MISSING LABELS

In this case only a subset of labels $\boldsymbol{L}' \subset \boldsymbol{L}$ is known and takes the categories $\boldsymbol{L}' = \boldsymbol{\ell}'$, while the ground truth for all other labels $\boldsymbol{L}^0 = \boldsymbol{L} \backslash \boldsymbol{L}'$ is unknown. In consequence, the optimization goal changes slightly. We define $\mathcal{L}^0, \mathcal{L}^0_{j=m}, \mathcal{L}^0_{j\neq m}$ and $\boldsymbol{\ell}^0$ in analogy to before.

Case 1: Category is known and incorrect. As in the previous section, we will first consider the case where $L_j \in \boldsymbol{L}'$ is known and m is not the true category, that is $\ell_j \neq m$:

$$\begin{aligned}
&\frac{\partial}{\partial q^k_{jm}} P(\boldsymbol{L}' = \boldsymbol{l}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}) \\
=&\frac{\partial}{\partial q^k_{jm}} \frac{\sum\limits_{\boldsymbol{\ell}^0 \in \mathcal{L}^0} P(R_k = 1, \boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0 | \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x})}{P(R_k = 1 | \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x})}
\end{aligned}$$

As the nominator is a factor independent of q^k_{jm}, the sum can be placed before the expression and the computations follow those in Section 1.1. When taking the logarithm, the gradient is divided by $P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x})$, so that the resulting gradient remains the same as in Section 1.1:

$$\begin{aligned}
&\frac{\partial}{\partial q^k_{jm}} \log(P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x})) \\
=&\frac{P(L_j = m | R_k = 0)(1 - P(R_k = 1))}{q^k_{jm} P(R_k = 1)} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \ .
\end{aligned}$$

Case 2: Category is known and correct. In the case that m is known and the true category, that is $L_j \in \boldsymbol{L}'$ and $\ell_j = m$, the sum can again be factored out. Thus, we can again make use of the gradients from Section 1.1:

$$\begin{aligned}
&\frac{\partial}{\partial q^k_{jm}} P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}) \\
=&\sum_{\boldsymbol{\ell}^0 \in \mathcal{L}^0} \frac{\partial}{\partial q^k_{jm}} P(\boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0 | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}) \\
=&(q^k_{jm} P(R_k = 1)^2)^{-1} \cdot \\
&(P(\boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1) P(R_k = 0, L_j = m) - \\
&P(\boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) P(R_k = 1))) \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \ .
\end{aligned}$$

When taking the logarithm, the expression gets slightly more complicated than in Section 1.1:

$$\begin{aligned}
&\frac{\partial}{\partial q^k_{jm}} \log(P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x})) \\
=&(P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}))^{-1} \frac{\partial}{\partial q^k_{jm}} P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}) \\
=&\frac{P(L_j = m | R_k = 0)(1 - P(R_k = 1))}{q^k_{jm} P(R_k = 1)} - \\
&\frac{P(\boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0)}{P(\boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1)} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \\
=&\frac{P(L_j = m | R_k = 0)(1 - P(R_k = 1))}{q^k_{jm} P(R_k = 1)} - \\
&\frac{P(R_k = 0 | \boldsymbol{L}' = \boldsymbol{\ell}')}{P(R_k = 1 | \boldsymbol{L}' = \boldsymbol{\ell}')} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \ .
\end{aligned}$$

So, one additional marginal query has to be calculated per rule. Note that this expression simplifies to that of Section 1.1 if all labels are known, that is if $\boldsymbol{L}' = \boldsymbol{L}$, because the conditional probabilities in the second fraction are then the rules conditional probability tables, given by $\boldsymbol{q}$ alone.

Case 3: Label is unknown. Interestingly, the gradients can also be computed for labels that have no ground truth, that is $L_j \in \boldsymbol{L}^0$. Obviously, we do not need to distinguish whether m is correct or not. Let $\boldsymbol{L}^0_{-j} := \boldsymbol{L}^0 \backslash L_j$ and let $q^k_{-j\ell} := \prod_{v:L_v \in \boldsymbol{L}'} q^k_{v\ell'_v} \cdot \prod_{v:L_v \in \boldsymbol{L}^0_{-j}} q^k_{v\ell_v}$ for simpler notation.

$$\begin{aligned}
&\frac{\partial}{\partial q^k_{jm}} P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \boldsymbol{1}, \boldsymbol{x}) \\
=&\frac{\partial}{\partial q^k_{jm}} \sum_{\boldsymbol{\ell}^0 \in \mathcal{L}^0} \Big((1 - q^k_{j\ell^*_j} q^k_{-j\ell^*}) \cdot \\
&\frac{P(\boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0)}{P(R_k = 1)} \Big) \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \\
=&\frac{\partial}{\partial q^k_{jm}} \Bigg(\sum_{\boldsymbol{\ell}^0 \in \mathcal{L}^0} P(\boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0) - \\
&\sum_{\boldsymbol{\ell}^0 \in \mathcal{L}^0_{-j}} q^k_{j\ell^*_j} q^k_{-j\ell^*} P(\boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0) - \\
&q^k_{jm} \sum_{\boldsymbol{\ell}^0 \in \mathcal{L}^0_{-j}} q^k_{-j\ell^*} P(\boldsymbol{L}' = \boldsymbol{\ell}', \boldsymbol{L}^0 = \boldsymbol{\ell}^0) \Bigg) \cdot \\
&\Bigg(\sum_{\boldsymbol{\ell} \in \mathcal{L}} P(\boldsymbol{L} = \boldsymbol{\ell}) - \sum_{\boldsymbol{\ell} \in \mathcal{L}_{j\neq m}} q^k_{j\ell_j} q^k_{-j\ell} P(\boldsymbol{L} = \boldsymbol{\ell}) - \\
&q^k_{jm} \sum_{\boldsymbol{\ell} \in \mathcal{L}_{j=m}} q^k_{-j\ell} P(\boldsymbol{L} = \boldsymbol{\ell}) \Bigg)^{-1} \mid \boldsymbol{R}_{-k} = \boldsymbol{1}, \boldsymbol{x} \ .
\end{aligned}$$

Again, we substitute to make differentiation easier to see:

$$
\begin{aligned}
&= \frac{\partial}{\partial q_{jm}^k} \frac{c_1 - q_{jm}^k c_2}{c_3 - q_{jm}^k c_4} \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \\
&= \frac{c_1 c_4 - c_2 c_3}{(c_3 - q_{jm}^k c_4)^2} \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \ .
\end{aligned}
$$

By substituting back and replacing the sum expressions with corresponding probability terms, we receive:

$$
\begin{aligned}
=&(P(R_k = 1)^2)^{-1} \cdot \\
&(P(\boldsymbol{L}' = \boldsymbol{\ell}') - P(L_j \neq m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0)) \cdot \\
&(q_{jm}^k)^{-1} P(L_j = m, R_k = 0) - (q_{jm}^k)^{-1} \cdot \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) \cdot (P(L_j = m) + \\
&P(L_j \neq m) - P(L_j \neq m, R_k = 0)) \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \\
=&(q_{jm}^k P(R_k = 1)^2)^{-1} \cdot \\
&(P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}') + \\
&P(L_j \neq m, \boldsymbol{L}' = \boldsymbol{\ell}') - P(L_j \neq m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0)) \cdot \\
&P(L_j = m, R_k = 0) - P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) \cdot \\
&(P(L_j = m) + P(L_j \neq m, R_k = 1))) \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \\
=&(q_{jm}^k P(R_k = 1)^2)^{-1} \cdot \\
&(P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) + \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1) + \\
&P(L_j \neq m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1)) \cdot \\
&P(L_j = m, R_k = 0) - P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) \cdot \\
&(P(L_j = m, R_k = 0) + P(L_j = m, R_k = 1) + \\
&P(L_j \neq m, R_k = 1))) \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \\
=&(q_{jm}^k P(R_k = 1)^2)^{-1} \cdot \\
&(P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1) P(L_j = m, R_k = 0) + \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) P(L_j = m, R_k = 0) + \\
&P(L_j \neq m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1) P(L_j = m, R_k = 0) - \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) P(L_j = m, R_k = 0) - \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) P(L_j = m, R_k = 1) - \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) \cdot \\
&P(L_j \neq m, R_k = 1)) \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \\
=&(q_{jm}^k P(R_k = 1)^2)^{-1} \cdot \\
&(P(\boldsymbol{L}' = \boldsymbol{\ell}', R_k = 1) P(L_j = m, R_k = 0) - \\
&P(L_j = m, \boldsymbol{L}' = \boldsymbol{\ell}', R_k = 0) \cdot \\
&P(R_k = 1)) \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \ .
\end{aligned}
$$

As before, the log likelihood removes the computationally complicated terms:

$$
\begin{aligned}
&\frac{\partial}{\partial q_{jm}^k} \log(P(\boldsymbol{L}' = \boldsymbol{\ell}' | \boldsymbol{R} = \mathbf{1}, \boldsymbol{x})) \\
=&\frac{P(L_j = m, R_k = 0)}{q_{jm}^k P(R_k = 1)} - \\
&\frac{P(L_j = m, R_k = 0 | \boldsymbol{L}' = \boldsymbol{\ell}')}{q_{jm}^k P(R_k = 1 | \boldsymbol{L}' = \boldsymbol{\ell}')} \mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \ .
\end{aligned}
$$

At this point it can be seen that the gradient reduces to 0 if no label has a ground truth, that is $L' = \emptyset$, which makes intuitive sense. Applying one last transformation gives a form that is easier to compute:

$$
\begin{aligned}
=&\frac{P(L_j = m | R_k = 0)(1 - P(R_k = 1))}{q_{jm}^k P(R_k = 1)} - \\
&\frac{P(L_j = m | R_k = 0, \boldsymbol{L}' = \boldsymbol{\ell}')(1 - P(R_k = 1 | \boldsymbol{L}' = \boldsymbol{\ell}'))}{q_{jm}^k P(R_k = 1 | \boldsymbol{L}' = \boldsymbol{\ell}')} \\
&\mid \boldsymbol{R}_{-k} = \mathbf{1}, \boldsymbol{x} \ .
\end{aligned}
$$

So, overall four marginal queries are required to compute this gradient, or two more than in the case of known labels.

2 EXTENSION OF NOISY–OR

The ordinary noisy–or gate as defined in Pearl [1988] is connected to a set of binary input variables $L_1, \ldots, L_J$. Each variable can only set the gate R_k to $R_k = 1$ with a probability $1 - q_{j1}^k$ if the variable itself is $L_j = 1$. Thus the conditional probability distribution that defines R_k is:

$$
P(R_k = 0 | L_1 = \ell_1, \ldots, L_J = \ell_J) = \prod_{j=1}^{J} (q_{j1}^k)^{\ell_j} \ .
$$

In our case, the input variables may have multiple categories $m = 1, \ldots, M(j)$ and each of these categories can trigger the gate to be $R_k = 1$ with a probability $1 - q_{jm}^k$. To find the conditional probability distribution of R_k in this case, we start by splitting each input variables L_j up into several binary auxiliary variables L_{jm} where

$$
P(L_{jm} = 1 | L_j = a) = \begin{cases} 1, a = m \\ 0, a \neq m \end{cases} = \mathbb{1}_{L_j = m}(L_j) \ .
$$

These variables can then be connected to the ordinary binary–input noisy–or gate as visualized in Figure 1 with their corresponding inhibition probability q_{jm}^k. We will now show that this process extends the binary–input noisy–or gate to multicategorical input naturally. We start with the conditional probability via the above described structure with auxiliary variables.

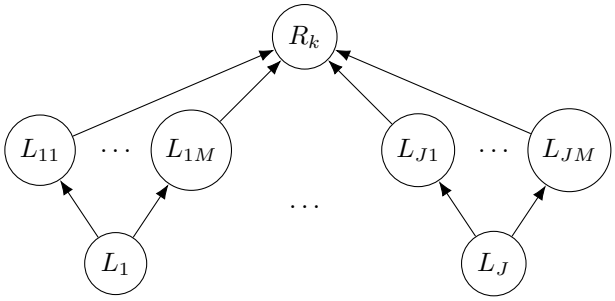

Figure 1: Decomposition of Multicategorical Inputs for a Binary–Input Noisy–Or Gate.

Algorithm 1 Simulation Dataset Generation

Require: nLabels, nRules, nData
1: pRSL $\leftarrow$ empty model
2: **for** i in $1, \ldots,$ nLabels **do**
3: nCategories $\sim U\{2, \ldots, 4\}$
4: Add label node with nCategories to pRSL
5: **for** i in $1, \ldots,$ nRules **do**
6: nCategories $\sim U\{2, \ldots, 5\}$
7: categories $\leftarrow$ Draw nCategories from all $\{1, \ldots, M(1), \ldots, 1, \ldots, M(J)\}$
8: **for** each categories **do**
9: inhProb $\sim$ Distribution with density $f(x) = 2 \cdot (1 - x) \cdot \mathbb{1}_{[0,1]}(x)$
10: Add rule with categories and inhProbs to pRSL
11: **for** i in $1, \ldots,$ nData **do**
12: **for** each label node in pRSL **do**
13: classifierOutput[label][i] $\sim$ Dir(1)
14: **return** pRSL, classifierOutput

$$
\begin{aligned}
& P(R_k = 0 | \boldsymbol{L} = \boldsymbol{\ell}) \\
= & \sum_{\ell_{11}, \ldots, \ell_{JM(J)} \; in \{0,1\}} \\
& P(R_k = 0 | L_{11} = \ell_{11}, \ldots, L_{JM(J)} = \ell_{JM(J)}) \cdot \\
& P(L_{11} = \ell_{11} | L_1 = \ell_1) \cdot \ldots \cdot \\
& P(L_{JM(J)} = \ell_{JM(J)} | L_J = \ell_J) \\
= & \sum_{\ell_{11}, \ldots, \ell_{JM(J)} \in \{0,1\}} \prod_{j=1}^{J} (q_{jm}^k)^{L_{jm}} (\mathbb{1}_{L_1 = 1}(L_1))^{\ell_{11}} \cdot \ldots \cdot \\
& (\mathbb{1}_{L_1 = M(1)}(L_1))^{\ell_{1M(1)}} \cdot \ldots \cdot (\mathbb{1}_{L_J = 1}(L_J))^{\ell_{J1}} \cdot \ldots \cdot \\
& (\mathbb{1}_{L_J = M(J)}(L_J))^{\ell_{JM(J)}} \; .
\end{aligned}
$$

With $0^0 := 1$, we can see that the only time the term inside the sum is not 0 is when $L_{jm} = 1$ iff $L_j = m$ for all $j = 1, \ldots, J$. This leaves open only one possible allocation of the binary auxiliary variables due to their XOR relation within j, so that the sum and the auxiliary variables vanish. We finally get a familiar expression that naturally extends the binary noisy–or to the multicategorical case:

$$
P(R_k = 0 | \boldsymbol{L} = \boldsymbol{\ell}) = \prod_{j=1}^{J} q_{j\ell_j}^k \; .
$$

3 SIMULATION DATASET GENERATION

Algorithm 1 describes the sampling procedure used to generate the simulation datasets used in Section 4.1. In the first ten lines of code, a pRSL model containing nLabels label nodes and nRules rule nodes is randomly generated. In lines 11 to 13, the classifier outputs for each classifier node are simulated by dirichlet noise for nData observations. Both the simulated data and the data–generating pRSL model are returned to allow performing approximate marginal and MPE queries on the correct model in Section 4.1.

References

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

Table 1: Performance of pRSL on Train, Validation, and Test Data. Mean $\pm$ Standard Deviation Between Folds.

	Emotions	Yeast	Birds	Medical	Enron	Mediamill
Joint Accuracy (higher = better)						
Train	0.351 ± 0.015	0.227 ± 0.010	0.516 ± 0.010	0.500 ± 0.019	0.153 ± 0.010	0.146 ± 0.001
Validation	0.339 ± 0.031	0.251 ± 0.026	0.516 ± 0.042	0.497 ± 0.027	0.154 ± 0.010	0.149 ± 0.006
Test	0.348 ± 0.067	0.236 ± 0.015	0.507 ± 0.032	0.491 ± 0.031	0.153 ± 0.020	0.149 ± 0.002
	Joint log–Likelihood (higher = better)			Label–wise log–Likelihood (higher = better)		
Train	-1.802 ± 0.077	-3.589 ± 0.054	-2.534 ± 0.060	-1.587 ± 0.022	-6.598 ± 0.079	-6.585 ± 0.006
Validation	-1.921 ± 0.241	-3.538 ± 0.177	-2.532 ± 0.269	-1.625 ± 0.051	-6.564 ± 0.178	-6.563 ± 0.005
Test	-1.839 ± 0.273	-3.592 ± 0.085	-2.458 ± 0.156	-1.565 ± 0.120	-6.479 ± 0.242	-6.532 ± 0.061
Label–wise Hamming Loss (lower = better)						
Train	0.182 ± 0.005	0.191 ± 0.003	0.043 ± 0.001	0.015 ± 0.001	0.046 ± 0.001	0.027 ± 0.000
Validation	0.181 ± 0.018	0.188 ± 0.004	0.042 ± 0.004	0.015 ± 0.001	0.046 ± 0.001	0.027 ± 0.000
Test	0.182 ± 0.022	0.190 ± 0.005	0.043 ± 0.002	0.015 ± 0.001	0.046 ± 0.001	0.027 ± 0.000