Unbiased Gradient Estimation with Balanced Assignments for Mixtures of Experts

Wouter Kool[*]
University of Amsterdam

Chris J. Maddison
DeepMind

Andriy Mnih
DeepMind

Abstract

Training large-scale mixture of experts models efficiently on modern hardware requires assigning datapoints in a batch to different experts, each with a limited capacity. Recently proposed assignment procedures lack a probabilistic interpretation and use biased estimators for training. As an alternative, we propose two unbiased estimators based on principled stochastic assignment procedures: one that skips datapoints which exceed expert capacity, and one that samples perfectly balanced assignments using an extension of the Gumbel-Matching distribution [29]. Both estimators are unbiased, as they correct for the used sampling procedure. On a toy experiment, we find the 'skip'-estimator is more effective than the balanced sampling one, and both are more robust in solving the task than biased alternatives.

1 Introduction

A mixture of experts (MoE) model can be used to implement conditional computation by processing different datapoints by different expert modules. This enables increasing the MoE models' representational capacity by adding more experts, without increasing the amount of computation per datapoint, which remains the same as for a single expert. Recently, this idea has been combined with deep neural networks, where each layer can be a separate MoE model, resulting in large-scale MoE's that yield state-of-the-art performance in various tasks [38, 25, 8, 26, 35, 45]. Training an MoE model involves training a *routing network* to assign datapoints to experts, and training individual experts to perform well on the datapoints assigned to them. In practice, for computational efficiency, we train on minibatches of datapoints, and as each expert has a limited capacity, we either have to *skip* datapoints exceeding the capacity of the experts they are assigned to, or *balance* the assignments of datapoints to different experts [8, 26]. The effect of skipping datapoints or balancing their assignments is typically not accounted for when training the routing network. Recent work has also challenged this approach by showing that in some cases better performance can be achieved using a fixed hashing-based routing strategy [36], which suggests that there is room for improvement in training the routing network. As a step towards this goal, we present two principled methods for optimizing MoE's under limited expert capacity. Specifically, we propose two sampling procedures and corresponding unbiased estimators in this paper: a simple one based on skipping datapoints that exceed expert capacity, and a more advanced one based on balanced datapoint assignment using an extension of the Gumbel-Matching distribution [29]. Whereas these sampling procedures ensure that each sample respects the expert capacity, we also propose to use the Sinkhorn algorithm to balance the assignment *in expectation* before sampling, and we connect this procedure to the Gumbel-Matching distribution, which has such Sinkhorn balancing built-in.

In this paper, we consider a single-layer MoE model, but this can be easily generalized. Formally, the problem we consider is to predict a label y for a datapoint x using an MoE model consisting of k individual experts $p_\theta(y|x, z)$ indexed by z and selected using the routing network $p_\theta(z|x)$. At test time, we select the most probable expert $z^* = \arg\max_z p_\theta(z|x)$, while for training we optimize

[*]Corresponding author: w.w.m.kool@uva.nl. Work done on an internship at DeepMind.

I (Still) Can't Believe It's Not Better Workshop at NeurIPS 2021.

a smoothed objective obtained by taking the expectation over the routing decisions. The resulting objective, ELBO, is a variational lower bound on the marginal log-likelihood:

$$\log p_\theta(y|x, z^*) \approx \underbrace{\mathbb{E}_{z\sim p_\theta(z|x)}[\log p_\theta(y|x, z)]}_{\text{ELBO}} \leq \log \mathbb{E}_{z\sim p_\theta(z|x)}[p_\theta(y|x, z)] = \log p_\theta(y|x). \quad (1)$$

We optimize this objective using minibatches of n datapoints, while respecting the expert capacity by assigning at most $c = \frac{n}{k}$ datapoints to each expert (for simplicity, we assume no *slack* capacity). We evaluate our estimators on a toy experiment, where we find that the simple 'skip'-estimator is more effective than the one based on balanced sampling using the Gumbel-Matching distribution. This is a surprising result, as we designed balanced sampling (with importance weights) as a better alternative to wastefully skipping data, but it seems that the benefit of using all data is outweighed by the added variance due to the importance weights. We do however find that both estimators, which are based on REINFORCE [11, 44], are more robust than the biased alternatives using differentiable gating.

2 Unbiased Estimation using Balanced Assignment

For simplicity, we consider the problem of optimizing a general function $f(x, z)$ using gradient descent using minibatches of datapoints $\mathbf{x} = (x_1, ..., x_n)$ and expert assignments $\mathbf{z} = (z_1, ..., z_n)$. To simplify notation, we omit the dependence of $f(x, z)$ on y and parameters θ (this can be added easily). By combining importance sampling with REINFORCE [11, 44], we can sample from any joint proposal distribution $q(\mathbf{z}|\mathbf{x})$ with marginals $q(z_i|\mathbf{x})$ to estimate the gradient for a minibatch $\mathbf{x}$:

$$\nabla\mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i f(x_i, z_i)\right] = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i \frac{p_\theta(z_i|x_i)}{q(z_i|\mathbf{x})}\nabla \log p_\theta(z_i|x_i)(f(x_i, z_i) - b)\right]. \quad (2)$$

Here b is a baseline which can reduce the estimator variance (see Appendix B). Taking $q(\mathbf{z}|\mathbf{x}) = \prod_i p_\theta(z_i|x_i)$ recovers the standard 'on-policy' REINFORCE estimator.

2.1 Skipping Datapoints as the Simple Solution

Our 'skip'-estimator respects expert capacity by sampling expert assignments independently and randomly subsampling the datapoints assigned to experts for which capacity is exceeded. If we assume a random order of the datapoints (or shuffle them first), simply skipping the last assignments per expert is equivalent to uniform subsampling. Let $\mathbf{z}$ be the vector of expert assignments, sampled independently from a proposal distribution $q(\mathbf{z}|\mathbf{x}) = \prod_i q(z_i|x_i)$. Let $n_j = \sum_i \mathbb{1}_{\{z_i=j\}}$ be the number of datapoints assigned to expert j (before subsampling) and $c = \frac{n}{k}$ be the expert capacity. Let $\boldsymbol{\delta} = (\delta_1, ..., \delta_n)$ with $\delta_i \in \{0, 1\}$ represent which datapoints are kept after per-expert subsampling. Correcting for the fact that the probability of datapoint i being kept after subsampling is $\min\{n_{z_i}, c\}/n_{z_i}$, we obtain (see Appendix C):

$$\nabla\mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i f(x_i, z_i)\right] = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\left[\mathbb{E}_{\boldsymbol{\delta}|\mathbf{z}}\left[\frac{1}{n}\sum_i \delta_i \frac{n_{z_i}}{\min\{n_{z_i}, c\}} \cdot \frac{\nabla p_\theta(z_i|x_i)}{q(z_i|x)} f(x_i, z_i)\right]\right]. \quad (3)$$

Here we have omitted the baseline b and used $\nabla p_\theta(z_i|x_i) = p_\theta(z_i|x_i)\nabla \log p_\theta(z_i|x_i)$ for brevity. Note that, while we skip datapoints in the gradient estimate (3), we can still propagate them to subsequent layers, e.g. by using skip connections or *no-token-left-behind* routing [8]. In paricular, we can also apply (3) to multilayer MoE's, where we can skip different datapoints in different layers.

2.2 Balanced Assignment using Gumbel-Matching

As an alternative to skipping, we use the $n \times k$ Gumbel-Matching distribution, a strict generalization of the $(n \times n)$ Gumbel-Matching distribution [29], to sample perfectly balanced assignments. We derive it by using the Gumbel-Max trick [28, 14] to view sampling of independent expert assignments as an optimization problem, and adding constraints to this problem to respect the expert capacity. Let $z_{ij} = \mathbb{1}_{\{z_i=j\}}$ be the one-hot representation of z_i, a_{ij} the unnormalized log-probability (logit) of assigning datapoint i to expert j, and $g_{ij} \sim \text{Gumbel}(0)$ i.i.d. standard Gumbel variables. Then $\mathbf{z}$ has

the Gumbel-Matching distribution if it is the solution to the following optimization problem:

$$\max_{\mathbf{z}} \sum_{ij} z_{ij}(a_{ij}/\tau + g_{ij}) \qquad \text{s.t. } \sum_j z_{ij} = 1 \; \forall i, \quad \sum_i z_{ij} \leq c \; \forall j, \quad z_{ij} \in \{0,1\} \; \forall i,j, \tag{4}$$

where τ is a *temperature* parameter and $c = \frac{n}{k}$ is the expert capacity. If we remove the balancing/capacity constraint $\sum_i z_{ij} \leq c$, the solution decomposes over i and is given by $z_i = \arg\max_j(a_{ij}/\tau + g_{ij})$ which is equivalent to $z_i \sim \text{Categorical}(\exp(a_{ij}/\tau)/\sum_j \exp(a_{ij}/\tau))$ as a result of the Gumbel-Max trick. Thus, adding the constraint can be seen as a natural way of enforcing a balanced assignment to the otherwise independent sampling procedure. Generalizing the result from [29], the $n \times k$ Gumbel-Matching approximates the Gibbs distribution over $n \times k$ assignments $\mathbf{z}$ with potentials given by $\sum_{ij} z_{ij} a_{ij}$ and temperature τ (see Appendix D.1):

$$p(\mathbf{z}) \propto \exp\left(\frac{1}{\tau}\sum_{ij} z_{ij} a_{ij}\right). \tag{5}$$

For $\tau \to 0$ we obtain the deterministic assignment used in BASE layers [26]. We propose to solve the $n \times k$ assignment problem using a special cycle cancelling algorithm [20] (see Appendix D.2), which for $k \ll n$ is more efficient than $O(n^3)$ assignment using the Hungarian Algorithm [23].

The marginals $q_\theta(z_i|\mathbf{x})$ of the Gumbel-Matching distribution are intractable [29], but we can compute the *conditionals* $q_\theta(z_i|\mathbf{x}, \mathbf{G}_{-i})$, conditioned on the noise $\mathbf{G}_{-\mathbf{i}} = (\mathbf{g}_1, ..., \mathbf{g}_{i-1}, \mathbf{g}_{i+1}, ..., \mathbf{g}_n)$ of the other datapoints. These can be computed efficiently (see Appendix D.4) and used as a stochastic approximations to the marginals, which still yield unbiased gradients when used in (2). Formally, let $\mathbf{z} = \text{GM}(\log p(\cdot|\mathbf{x}), \mathbf{G})$ be the solution for the Gumbel-Matching problem with noise $\mathbf{G}$. We can then use the following estimator (see Appendix D.5):

$$\nabla \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i f(x_i, z_i)\right] = \mathbb{E}_{\mathbf{G}}\left[\frac{1}{n}\sum_i \frac{\nabla p_\theta(z_i|x_i)}{q_\theta(z_i|\mathbf{x}, \mathbf{G}_{-i})} f(x_i, z_i)\bigg|_{\mathbf{z}=\text{GM}(\log p(\cdot|\mathbf{x}), \mathbf{G})}\right]. \tag{6}$$

2.3 Bonus: Balancing Expectations using the Sinkhorn Algorithm

As an alternative to generating balanced samples, we can balance the distribution *in expectation* by using a generalization of the Sinkhorn algorithm [39, 40, 21] to non-square matrices, which iteratively normalizes the columns and rows of the probability matrix $p_{ij} = p_\theta(z_i = j|x_i)$ to sum to $\frac{n}{k}$ and 1 respectively, until convergence. We may then use this Sinkhorn-normalized distribution as the proposal $\mathbf{q}(\mathbf{z}|\mathbf{x})$ in (2) and (3), which yields more (but not perfectly) balanced samples. The Gumbel-Matching distribution is invariant to this normalization as it does not change the solution to (4). See Figure 1 for an overview.

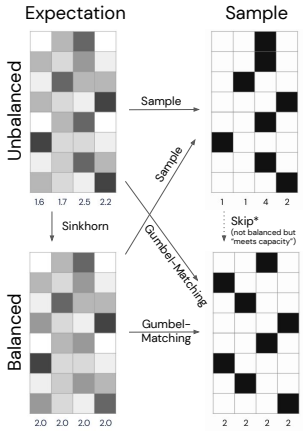
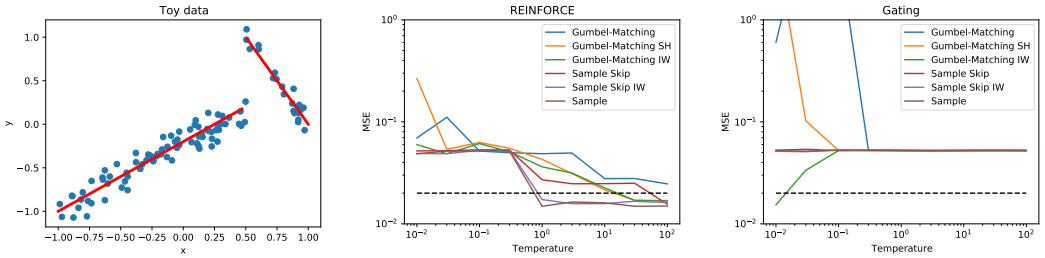


Figure 1: Overview of the methods that generate balanced and unbalanced samples and expectations.

The Sinkhorn algorithm is a direct extension of the softmax function, which solves a soft (entropy-regularized) version of the assignment problem [6, 30, 29]. As a result, it can be seen as approximating the marginals for the Gibbs distribution (5) [10, 29] (for $n = k$, but this can be generalized to $k < n$). Empirically, we find that the Sinkhorn algorithm (as a 'soft' matching algorithm) also closely approximates the marginals of the Gumbel-Matching distribution (itself an approximation of (5)), at least for $n \geq 4$. As such, we propose to heuristically use it with (2) as an alternative to (6).

The Sinkhorn algorithm yields a *balanced row-stochastic matrix* with probabilities/expectations $\tilde{p}_{ij}$, that can be seen as a balanced approximation of the unbalanced probabilities p_{ij}. Given such a balanced matrix, there exist many distributions over balanced assignments that have the (per datapoint) marginals equal to $\tilde{p}_{ij}$, which follows from generalizing the Birkhoff theorem [4, 43]. When using such a distribution with stochastic gradient descent, we would like to minimize the dependence between samples for different datapoints, to reduce the variance of the gradient estimates. This can be achieved by maximizing the entropy of the joint distribution over expert assignments with the given marginals $\tilde{p}_{ij}$. In Appendix D.6, we show that this maximum-entropy distribution has the form (5), again motivating the Gumbel-Matching distribution as a principled approximation.



(a) Toy dataset (b) Results using REINFORCE (c) Results using gating

Figure 2: Toy experiment data and results using REINFORCE (without balance loss) and training using gating (with balance loss weight 0.01). We report final mean squared error (MSE) when training with different temperatures/noise scales. The black dashed line indicates the success threshold 0.02.

3 Experiment

We evaluate the proposed estimators on a toy task of modelling a discontinuous function $y = \mathbb{1}_{x<0.5}(0.8x - 0.2) + \mathbb{1}_{x\geq 0.5}(-2.0x + 2.0)$. We sample a dataset of 100 datapoints $x \sim \text{Uniform}(-1, 1)$ and add noise $\epsilon \sim \text{Normal}(0, 0.1^2)$, see Figure 2a. We model this dataset using a mixture of two linear experts and a Bernoulli router distribution with probability given by the sigmoid of a linear function. This model is able to solve the task perfectly, but the training methods need to take into account the fact that the solution involves an imbalanced partitioning of the dataset between the experts. We train the model to minimize the mean squared error (MSE) under sampling of the experts, which corresponds to the objective (1) where $p_\theta(y|x, z)$ is Gaussian with a fixed variance. We consider the task solved successfully if the MSE < 0.02 after 10K steps of training using Adam [18] with the learning rate of 0.1 (found to work best overall using a grid search).

We compare the following sampling strategies combined with REINFORCE: **Sample Skip IW** is the skipping estimator with the importance-weighting correction (Equation (3)), whereas **Sample Skip** is the biased alternative that simply averages the gradient over the remaining datapoints. **Gumbel-Matching IW** uses the conditional distributions for the importance weights (Equation (6)), whereas **Gumbel-Matching SH** is the (biased) version that uses the 'Sinkhorn marginals' (see Section 2.3) with Equation (2), and **Gumbel-Matching** is the biased estimator that does not correct for balancing using importance weights. Lastly, to quantify the reduction in performance due to the expert capacity constraints, we also include the results for **Sample**, the ideal baseline that does *not* respect expert capacity. We run each experiment with 10 seeds, for a range of sampling temperatures τ, taking into account their effect on the proposal distribution in the importance weights for all estimators. To reduce variance, we include an exponential moving average baseline [41] with decay 0.99.

Figure 2b plots the final training MSE for different temperatures τ of the sampling (proposal) distribution, where we observe that **Sample Skip IW** is the only estimator that matches the imbalanced (unconstrained) **Sample** estimator, both solving the task for $\tau \geq 1$. The skipping estimator thus provides a simple and effective way to deal with limited expert capacity, but it is important to upweight the remaining samples for experts which have skipped datapoints, as **Sample Skip**, which does not use this weighting, does not achieve the same performance. Confounding our expectation, the Gumbel-Matching based estimators turned out to be less effective, because of the increased variance due to the importance weights. Investigating the issue, we found that a datapoint x can have high probability $p_\theta(z|x)$ for an expert z *according to the router*, but a low probability under the proposal $q(z|x)$ of *actually being assigned* to the expert z, due to the balancing constraint. While the latter probability is small, occasionally the datapoint will get assigned to z, resulting in a large importance weight $\frac{p_\theta(z|x)}{q(z|x)}$. This effect can be mitigated by increasing the temperature of the proposal distribution, making it more uniform and avoiding large importance weights, which explains the good results for large τ values for all estimators except **Gumbel-Matching**. We also experiment with all estimators with Sinkhorn balancing (Section 2.3) before sampling, which only works for high temperatures (see Appendix E).

3.1 Biased training using differentiable gating

Large scale MoE's used in practice [38, 25, 8, 26, 35, 45] do not use REINFORCE, but instead multiply the output of an expert by the router probability $p_\theta(z|x)$, which we refer to as *differentiable gating*. This way, the router becomes more coupled with the experts and gets a gradient signal directly from the objective. Different strategies for injecting noise to encourage exploration have been proposed, e.g. perturbing router inputs or (log-)probabilities with multiplicative or additive (Gaussian) noise [38, 8]. Empirically, we find that we get similar results by perturbing log-probabilities with (scaled) Gumbel noise, which, since $\arg\max_j(a_{ij} + \tau \cdot g_{ij}) = \arg\max_j(a_{ij}/\tau + g_{ij})$, has the advantage of being interpretable as sampling from a categorical distribution with the temperature τ (see Section 2.2). We find training using differentiable gating succeeds only if we additionally include a load balancing loss [8] with a weight of 0.01 or 0.03, and use balanced sampling with importance weights in a low temperature regime, as can be seen in Figure 2c. With differentiable gating, we may wonder if we need importance weights at all, since existing methods do not use importance weights to correct for the sampling temperature (noise scale) τ. In Appendix E we show however, that with differentiable gating, we fail to train the model when not using importance weights.

4 Discussion

In this paper we proposed several new estimators for training MoE models with a limited computational capacity per expert. We expected balanced sampling with importance weighting to correct for assignment of datapoints to low-probability experts to perform at least as well as skipping, which effectively uses a weight of 0 for the skipped datapoints. We found this not to be the case in practice, with the added variance from importance weights eliminating the benefit from the increased expert capacity utilization due to balanced sampling. Fortunately, the skipping estimator turned out to be a simple and effective alternative. We hope this work will be useful for training MoE models in practice.

Acknowledgments and Disclosure of Funding

We would like to thank Michalis Titsias, Jörg Bornschein, Matthias Bauer and Yee Whye Teh for helpful comments, discussions and support.

References

[1] Ryan Prescott Adams and Richard S Zemel. Ranking via sinkhorn propagation. *arXiv preprint arXiv:1106.1925*, 2011.

[2] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[4] Garrett Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucuman, Ser. A*, 5:147–154, 1946.

[5] Eric Budish, Yeon-Koo Che, Fuhito Kojima, and Paul Milgrom. Designing random allocation mechanisms: Theory and applications. *American economic review*, 103(2):585–623, 2013.

[6] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

[7] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*, 2013.

[8] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

[9] Robert W Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

[10] Amir Globerson and Tommi Jaakkola. Approximate inference using conditional entropy decompositions. In *Artificial Intelligence and Statistics*, pages 131–138. PMLR, 2007.

[11] Peter W Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84, 1990.

[12] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

[13] Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. In *International Conference on Learning Representations*, 2019.

[14] Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

[15] Tamir Hazan, Subhransu Maji, and Tommi Jaakkola. On sampling from the gibbs distribution with random maximum a-posteriori perturbations. *Advances in Neural Information Processing Systems*, 26:1268–1276, 2013.

[16] Mark Huber. Exact sampling from perfect matchings of dense regular bipartite graphs. *Algorithmica*, 44(3):183–193, 2006.

[17] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015.

[19] Louis Kirsch, Julius Kunze, and David Barber. Modular networks: learning to decompose neural computation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2414–2423, 2018.

[20] Morton Klein. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14(3):205–220, 1967.

[21] Philip A Knight. The sinkhorn–knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008.

[22] Jonathan Kuck, Tri Dao, Hamid Rezatofighi, Ashish Sabharwal, and Stefano Ermon. Approximating the permanent by sampling from adaptive partitions. *Advances in neural information processing systems*, 2019.

[23] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[24] Quoc Le and Alexander Smola. Direct optimization of ranking measures. *arXiv preprint arXiv:0704.3359*, 2007.

[25] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021.

[26] Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. *International Conference on Machine Learning*, 2021.

[27] Ke Li, Kevin Swersky, and Richard Zemel. Efficient feature learning using perturb-and-map. 2013.

[28] Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. *Advances in Neural Information Processing Systems*, 27:3086–3094, 2014.

[29] Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. *International Conference on Learning Representations*, 2018.

[30] Gonzalo Mena, David Belanger, Gonzalo Munoz, and Jasper Snoek. Sinkhorn networks: Using optimal transport techniques to learn permutations. In *NIPS Workshop in Optimal Transport and Machine Learning*, 2017.

[31] George Papandreou and Alan L Yuille. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models. In *2011 International Conference on Computer Vision*, pages 193–200. IEEE, 2011.

[32] Giorgio Patrini, Rianne van den Berg, Patrick Forre, Marcello Carioni, Samarth Bhargav, Max Welling, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. In *Uncertainty in Artificial Intelligence*, 2019.

[33] Pekka Raiko, Mathias Berglund, Guillaume Alain, and Laurent Dinh. Techniques for learning binary stochastic feedforward neural networks. In *International Conference on Learning Representations*, pages 1–10, 2015.

[34] Prajit Ramachandran and Quoc V Le. Diversity and depth in per-example routing models. In *International Conference on Learning Representations*, 2019.

[35] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *arXiv preprint arXiv:2106.05974*, 2021.

[36] Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models. *arXiv preprint arXiv:2106.04426*, 2021.

[37] Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*, 2019.

[38] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. 2017.

[39] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

[40] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

[41] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[42] Jakub M Tomczak. On some properties of the low-dimensional gumbel perturbations in the perturb-and-map model. *Statistics & Probability Letters*, 115:8–15, 2016.

[43] John Von Neumann. A certain zero-sum two-person game equivalent to the optimal assignment problem, contributions to the theory of games, vol. 2. *Ann. Math. Studies*, (28), 1953.

[44] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[45] An Yang, Junyang Lin, Rui Men, Chang Zhou, Le Jiang, Xianyan Jia, Ang Wang, Jie Zhang, Jiamang Wang, Yong Li, et al. Exploring sparse expert models and beyond. *arXiv preprint arXiv:2105.15082*, 2021.

A Related work

Mixtures of experts [7, 38] have a long history as a method for conditional computation [3, 2, 12] where different experts are used for different datapoints. Typically, the *router* or *gating* network that assigns datapoints to experts is learned jointly with the experts themselves. Mixtures of experts and the more general *routing* or *modular networks* [37, 19, 34], sometimes treat expert assignments as latent variables and use EM or variational methods [19] for training them. The benefit of optimizing a single-sample bound (ELBO) rather than marginal likelihood, is that in additional to being more tractable, it can help to avoid stochasticity [33] in datapoint assignments to experts.

Whereas the *conditional* distribution over experts given a datapoint should ideally have low entropy, the *marginal* distribution over experts should be balanced for efficient training and use of model capacity. In some cases this is explicitly encouraged by using a load balancing loss [2, 8]. Other methods algorithmically balance the assignment [26] or use a fixed distribution [36]. Many recent large scale MoE models use heuristics to train the router module (see Section 3.1) but have yielded state-of-the-art performance in different domains [38, 25, 8, 26, 35, 45].

Sampling and optimization of balanced assignments is closely related to the sampling and optimization of permutations ($n \times n$ assignments) [24, 1, 27, 30, 29, 13, 32], which relate to estimation of the matrix permanent [16, 17, 22]. In a different setting, the problem can also be seen as random fair assignment [5], with the difference being that in random fair assignment one typically is not concerned about dependence between the individual assignments.

B Derivation of (off-policy) REINFORCE

First, we derive (off-policy) REINFORCE for a single datapoint x:

$$\begin{aligned}
\nabla \mathbb{E}_{z\sim p_\theta(z|x)}[f(x,z)] &= \nabla \sum_z p_\theta(z|x) f(x,z) \\
&= \sum_z \nabla p_\theta(z|x) f(x,z) \\
&= \sum_z q(z|x) \frac{\nabla p_\theta(z|x)}{q(z|x)} f(x,z) \\
&= \mathbb{E}_{z\sim q(z|x)} \left[\frac{\nabla p_\theta(z|x)}{q(z|x)} f(x,z) \right] \\
&= \mathbb{E}_{z\sim q(z|x)} \left[\frac{p_\theta(z|x)}{q(z|x)} \nabla \log p_\theta(z|x) f(x,z) \right]
\end{aligned}$$

Now consider a minibatch $\mathbf{x} = (x_1, ..., x_n)$ and let $p_\theta(\mathbf{z}|\mathbf{x}) = \prod_i p_\theta(z_i|x_i)$ be the distribution that samples expert assignments independently. Let $q(z_i|\mathbf{x}) = \sum_{z_{-i}} q(\mathbf{z}|\mathbf{x})$ be the marginal of any joint proposal distribution $q(\mathbf{z}|\mathbf{x})$, which we can use to estimate the minibatch gradient:

$$\begin{aligned}
\nabla \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})} \left[\frac{1}{n} \sum_i f(x_i, z_i) \right] &= \frac{1}{n} \sum_i \nabla \mathbb{E}_{z_i \sim p_\theta(z_i|x_i)} \left[f(x_i, z_i) \right] \\
&= \frac{1}{n} \sum_i \mathbb{E}_{z_i \sim q(z_i|\mathbf{x})} \left[\frac{\nabla p_\theta(z_i|x_i)}{q(z_i|\mathbf{x})} f(x_i, z_i) \right] \\
&= \frac{1}{n} \sum_i \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} \left[\frac{\nabla p_\theta(z_i|x_i)}{q(z_i|\mathbf{x})} f(x_i, z_i) \right] \\
&= \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} \left[\frac{1}{n} \sum_i \frac{\nabla p_\theta(z_i|x_i)}{q(z_i|\mathbf{x})} f(x_i, z_i) \right] \\
&= \mathbb{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})} \left[\frac{1}{n} \sum_i \frac{p_\theta(z_i|x_i)}{q(z_i|\mathbf{x})} \nabla \log p_\theta(z_i|x_i) f(x_i, z_i) \right].
\end{aligned}$$

Since $\nabla \mathbb{E}_{z\sim p_\theta(z|x)}[b] = 0$, we can subtract any constant baseline b from $f(x, y)$, resulting in (2).

C Unbiasedness of the skipping estimator

First, consider a function $h(x_i, z_i)$. Let $z_{ij} = \mathbb{1}_{\{z_i=j\}}$ be the one-hot representation of z_i and let $h_{ij} = h(x_i, z_i)|_{z_i=j}$. Let $n_j = \sum_i z_{ij}$ be the number of datapoints assigned to expert j (before subsampling). Now let $\delta_i \in \{0, 1\}$ represent which datapoints are kept after we, for each expert j, uniformly subsample $\min\{n_j, c\}$ datapoints, where $c = \frac{n}{k}$ is the expert capacity. If $z_{ij} = 1$ (before subsampling), then the probability that datapoint i remains after subsampling is $\frac{\min\{n_j,c\}}{n_j}$, so we have $\mathbb{E}_{\boldsymbol{\delta}^*|\mathbf{z}}\left[\delta_i\right] \frac{n_j}{\min\{n_j,c\}} z_{ij} = z_{ij}$.

$$\begin{aligned}
\mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i h(x_i, z_i)\right] &= \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i\sum_j z_{ij} h_{ij}\right] \\
&= \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i\sum_j \mathbb{E}_{\boldsymbol{\delta}|\mathbf{z}}\left[\delta_i\right]\frac{n_j}{\min\{n_j, c\}} z_{ij} h_{ij}\right] \\
&= \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\mathbb{E}_{\boldsymbol{\delta}|\mathbf{z}}\left[\frac{1}{n}\sum_i \delta_i \frac{n_{z_i}}{\min\{n_{z_i}, c\}} h(x_i, z_i)\right]\right]
\end{aligned}$$

Now substituting $h(x_i, z_i) = \frac{\nabla p_\theta(z_i|x_i)}{q(z_i|x)} f(x_i, z_i)$ and combining with Equation (2) results in (3).

D The Gumbel-Matching distribution

D.1 Approximation to the Gibbs distribution

Here we essentially reproduce the argument from [29] for the $n \times k$ Gumbel-Matching distribution. By sampling i.i.d. Gumbel noise $g_{\mathbf{z}}$ for every assignment $\mathbf{z}$, we can sample from (5) by maximizing

$$\left(\frac{1}{\tau}\sum_{ij} z_{ij} a_{ij}\right) + g_{\mathbf{z}}$$

subject to the constraints given by (4). Comparing this to the objective for the Gumbel-Matching problem (4):

$$\sum_{ij} z_{ij}(a_{ij}/\tau + g_{ij}) = \left(\frac{1}{\tau}\sum_{ij} z_{ij} a_{ij}\right) + \sum_{ij} z_{ij} g_{ij}$$

we observe how the Gumbel-Matching distribution approximates (5) through the use of rank-one perturbations [31, 15, 42] $\sum_{ij} z_{ij} g_{ij}$ instead of $g_{\mathbf{z}}$.

D.2 Solving $n \times k$ matching using cycle cancelling with Floyd-Warshall

The $n \times k$ assignment problem can be modelled as a minimum cost flow problem which can be solved using cycle cancelling [20] as follows:

- Find an initial (heuristic) feasible assignment $\mathbf{z}$. We use the auction algorithm used in [26] with $\epsilon = 1.0$ such that it finds a good (but suboptimal) solution quickly.
- For every combination of experts j, j', find the lowest cost $d_{jj'}$ to move a datapoint from expert j to j'. Let $s_{ij} = a_{ij}/\tau + g_{ij}$ be the *score* for assigning datapoint i to j, such that moving datapoint i from j to j' we lose s_{ij} but gain $s_{ij'}$, incurring a net 'cost' of $s_{ij} - s_{ij'}$. Therefore, the minimum cost to move any of the currently assigned datapoints from j to j' is $d_{jj'} = \min_{i:z_{ij}=1} s_{ij} - s_{ij'}$.
- Use the Floyd-Warshall algorithm[2] [9] to find all indirect shortest paths in the fully connected graph with k nodes (one per expert) and distance from j to j' given by $d_{jj'}$. Stop as soon as a negative cycle is found (distance from j to j smaller than 0).

[2]We use the parallel version: `https://en.wikipedia.org/wiki/Parallel_all-pairs_shortest_path_algorithm#Floyd_algorithm` which runs in $O(k^3)$ and k sequential steps.

- If no negative cycle exists, the assignment is optimal, stop.
- For each edge (j, j') in the negative cycle,[3] move the datapoint i that minimizes $s_{ij} - s_{ij'}$ from j to j'. This will improve the assignment by incurring a negative total cost.
- Repeat until no negative cycle exists.

Depending on the initial assignment, only a small number of $O(k^3)$ improvements is needed and we find in practice for $k \ll n$ this algorithm is much faster than the $O(n^3)$ Hungarian [23] algorithm.

D.3 Computing conditionally optimal assignments

If the assignment is optimal, we denote the entries of the all-pairs shortest path matrix resulting from the Floyd-Warshall algorithm by $d^*_{jj'}$ (see Appendix D.2). We can use this to efficiently obtain *conditionally optimal assignments*, conditioning on $z_{ij} = 1$ for all i, j, as follows. If we condition on $z_{ij} = 1$, we may move datapoint i from the globally optimal assignment j^* to a suboptimal assignment j, incurring a cost $s_{ij^*} - s_{ij}$, and move another datapoint from j to j^* for a cost of $d^*_{jj^*}$, indirectly via the path found by the Floyd-Warshall algorithm. As such, the total cost of enforcing $z_{ij} = 1$ is $s_{ij^*} - s_{ij} + d^*_{jj^*}$. We denote the value of the globally optimal assignment by v^*. Subtracting the cost for enforcing $z_{ij} = 1$, we find that the value $v^*_{|z_{ij}=1}$ of the conditionally optimal assignment is given by

$$v^*_{|z_{ij}=1} = v^* - (s_{ij^*} - s_{ij} + d^*_{jj^*}) = v^* - s_{ij^*} + s_{ij} - d^*_{jj^*}. \tag{7}$$

D.4 Computation of the conditionals

The dependence of the Gumbel-Matching distribution on $\mathbf{x}$ is only through the logits $\mathbf{A} = (a_{ij})$, which allows us to slightly simplify notation in the rest of this section. Let $v^*(\mathbf{A}, \mathbf{G})$ be the value of the optimal assignment for the Gumbel-Matching problem with logits $\mathbf{A} = (a_{ij})$ and noise $\mathbf{G} = (g_{ij})$, and let $v^*_{|z_{ij}=1}(\mathbf{A}, \mathbf{G})$ be the value of the conditionally optimal assignment with the additional constraint that $z_{ij} = 1$ (see Appendix D.3). Assuming a capacity $c = \frac{n}{k}$, then given that $z_{ij} = 1$, the problem reduces to assigning the remaining $n - 1$ datapoints to the remaining $(k-1) \cdot \frac{n}{k} + (\frac{n}{k} - 1) = n - 1$ 'slots' ($\frac{n}{k}$ for experts $j' \neq j$ and $\frac{n}{k} - 1$ for expert j). As this reduced problem does not depend on datapoint i, we denote with $\mathbf{A}_{-i}$ and $\mathbf{G}_{-i}$ the logits and Gumbels with row i removed and we let $v^*_{-ij}(\mathbf{A}_{-i}, \mathbf{G}_{-i})$ be the value of the optimal assignment of the reduced problem. From the principle of optimality it follows that

$$v^*_{|z_{ij}=1}(\mathbf{A}, \mathbf{G}) = v^*_{-ij}(\mathbf{A}_{-i}, \mathbf{G}_{-i}) + a_{ij}/\tau + g_{ij}. \tag{8}$$

We can use this to compute the desired conditionals. In a slight abuse of notation[4] we write

$$\begin{aligned}
&q(z_{ij}|\mathbf{A}, \mathbf{G}_{-i}) \\
=&P(z_{ij} = 1|\mathbf{A}, \mathbf{G}_{-i}) \\
=&P\left(v^*_{|z_{ij}=1}(\mathbf{A}, \mathbf{G}) > \max_{j' \neq j} v^*_{|z_{ij'}=1}(\mathbf{A}, \mathbf{G}) \middle| \mathbf{A}, \mathbf{G}_{-i}\right) \\
=&P\left(v^*_{-ij}(\mathbf{A}_{-i}, \mathbf{G}_{-i}) + a_{ij}/\tau + g_{ij} > \max_{j' \neq j} v^*_{-ij'}(\mathbf{A}_{-i}, \mathbf{G}_{-i}) + a_{ij'}/\tau + g_{ij'} \middle| \mathbf{A}, \mathbf{G}_{-i}\right) \\
=&\frac{\exp(v^*_{-ij}(\mathbf{A}_{-i}, \mathbf{G}_{-i}) + a_{ij}/\tau)}{\sum_{j'} \exp(v^*_{-ij'}(\mathbf{A}_{-i}, \mathbf{G}_{-i}) + a_{ij'}/\tau)} \\
=&\frac{\exp\left(v^*_{|z_{ij}=1}(\mathbf{A}, \mathbf{G}) - g_{ij}\right)}{\sum_{j'} \exp\left(v^*_{|z_{ij'}=1}(\mathbf{A}, \mathbf{G}) - g_{ij'}\right)}
\end{aligned} \tag{9}$$

where we have used the Gumbel-max trick. Although g_{ij} appears in (9), $q(z_{ij}|\mathbf{A}, \mathbf{G}_{-i})$ does *not* depend on $\mathbf{g}_i$ as this value cancels against g_{ij} in (8). The values $v^*_{|z_{ij}=1}(\mathbf{A}, \mathbf{G})$ can be computed efficiently using Equation (7) with the method described in Appendix D.3.

[3] Can be reconstructed by using Floyd-Warshall with path reconstruction: `https://en.wikipedia.org/wiki/Floyd-Warshall_algorithm#Pseudocode_[11]`.

[4] $q(z_{ij})$ corresponds to $q(z_i)$ where $z_i = j$ but assumes a one-hot representation.

D.5 Unbiased of the Gumbel-Matching estimator

Let $\mathbf{z} = \text{GM}(\log p(\cdot|\mathbf{x}), \mathbf{G})$ be the solution for the Gumbel-Matching problem with noise $\mathbf{G}$. The idea behind the Gumbel-Matching estimator is that we can derive an unbiased estimate of the gradient *for each datapoint* as follows. First sample the Gumbel noise $\mathbf{G}_{-i}$ for all datapoints except i, and compute the conditional distribution $q_\theta(z_i|\mathbf{x}, \mathbf{G}_{-i})$ (see Appendix D.4 where $\mathbf{A} = \log p(\cdot|\mathbf{x})$ is the matrix with log-probabilities and $z_{ij} = 1 \Leftrightarrow z_i = j$). We can then use this conditional distribution over z_i as a proposal distribution in (2), which we can then reparameterize in terms of the Gumbel noise $\mathbf{g}_i$ for the i-th datapoint. Finally, we can use this estimator for all datapoints i, where we may reuse the same Gumbel noise and use their average as the estimate:

$$
\begin{aligned}
&\nabla \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}\left[\frac{1}{n}\sum_i f(x_i, z_i)\right] \\
=&\frac{1}{n}\sum_i \mathbb{E}_{z_i\sim p_\theta(z_i|x_i)}\left[\nabla \log p_\theta(z_i|x_i) f(x_i, z_i)\right] \\
=&\frac{1}{n}\sum_i \mathbb{E}_{\mathbf{G}_{-i}}\left[\mathbb{E}_{z_i\sim q_\theta(z_i|\mathbf{x},\mathbf{G}_{-i})}\left[\frac{p_\theta(z_i|x_i)}{q_\theta(z_i|\mathbf{x},\mathbf{G}_{-i})}\nabla \log p_\theta(z_i|x_i) f(x_i, z_i)\right]\right] \\
=&\frac{1}{n}\sum_i \mathbb{E}_{\mathbf{G}_{-i}}\left[\mathbb{E}_{\mathbf{g}_i}\left[\frac{p_\theta(z_i|x_i)}{q_\theta(z_i|\mathbf{x},\mathbf{G}_{-i})}\nabla \log p_\theta(z_i|x_i) f(x_i, z_i)\Big|_{z_i=\text{GM}(\log p(\cdot|\mathbf{x}),\mathbf{G})_i}\right]\right] \\
=&\mathbb{E}_{\mathbf{G}}\left[\frac{1}{n}\sum_i \frac{\nabla p_\theta(z_i|x_i)}{q_\theta(z_i|\mathbf{x},\mathbf{G}_{-i})} f(x_i, z_i)\Big|_{\mathbf{z}=\text{GM}(\log p(\cdot|\mathbf{x}),\mathbf{G})}\right].
\end{aligned}
$$

D.6 Maximum-entropy distribution over balanced assignments

For simplicity, we assume $n = k$, but this can be easily generalized. Assume that we have a balanced (square) matrix (see Section 2.3) $\mathbf{P} = (p_{ij})$ with probabilities $p_{ij} \geq 0$ such that $\sum_i p_{ij} = \sum_j p_{ij} = 1$, i.e. the matrix $\mathbf{P}$ is *doubly stochastic*. The Birkhoff decomposition [4, 43] decomposes such a matrix as a convex combination over *permutation matrices* $\mathbf{Z} = (z_{ij})$, for which $z_{ij} \in \{0, 1\}$ and $\sum_i z_{ij} = \sum_j z_{ij} = 1$. Such permutation matrices represent $n \times n$ matchings $\mathbf{z}$ in one-hot encoding (i.e. $z_{ij} = 1 \Leftrightarrow z_i = j$), which is why we will use $\mathbf{z}$ to denote them. A Birkhoff decomposition $\alpha_\mathbf{z} > 0, \sum_\mathbf{z} \alpha_\mathbf{z} = 1$ thus represents a joint probability distribution over matchings $\mathbf{z}$ (represented as permutation matrices) with marginal distributions (per datapoint) given by $\mathbf{P}$:

$$
p_{ij} = \sum_\mathbf{z} \alpha_\mathbf{z} z_{ij} \quad \forall i, j. \tag{10}
$$

Many such decompositions/distributions exist and can be found using the Birkhoff algorithm[4], but these will yield sparse $\alpha_\mathbf{z}$ and thus have low entropy and high dependence between the marginal distributions for different i. We aim to minimize this dependence between the marginal distributions, which is achieved by maximizing the entropy $-\sum_\mathbf{z} \alpha_\mathbf{z} \log \alpha_\mathbf{z}$ subject to the constraint (10) and $\sum_\mathbf{z} \alpha_\mathbf{z} = 1$, which has the Lagrangian:

$$
\mathcal{L}(\boldsymbol{\alpha}, \eta, \boldsymbol{\lambda}) = -\sum_\mathbf{z} \alpha_\mathbf{z} \log \alpha_\mathbf{z} - \eta\left(1 - \sum_\mathbf{z} \alpha_\mathbf{z}\right) - \sum_{ij} \lambda_{ij}\left(p_{ij} - \sum_\mathbf{z} \alpha_\mathbf{z} z_{ij}\right).
$$

This has first order conditions

$$
\begin{aligned}
\frac{\partial}{\partial \alpha_\mathbf{z}}\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) &= -\log \alpha_\mathbf{z} - 1 + \eta + \sum_{ij} \lambda_{ij} z_{ij} = 0 \quad \forall \mathbf{z} \\
\frac{\partial}{\partial \lambda_{ij}}\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) &= p_{ij} - \sum_\mathbf{z} \alpha_\mathbf{z} z_{ij} = 0 \quad \forall i, j \\
\frac{\partial}{\partial \eta}\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\lambda}) &= 1 - \sum_\mathbf{z} \alpha_\mathbf{z} = 0.
\end{aligned}
$$

If we let $u_{ij} = \exp(\lambda_{ij})$ and $w = \exp(\eta - 1)$, and convert from the 'one-hot' representation z_{ij} to the 'indexing' representation z_i (i.e. $\sum_{ij} \lambda_{ij} z_{ij} = \sum_i \lambda_{i,z_i}$), we find the solution

$$\alpha_{\mathbf{z}} = \exp\left(-1 + \eta + \sum_{ij} \lambda_{ij} z_{ij}\right) = \exp(\eta - 1) \prod_i \exp(\lambda_{i,z_i}) = w \prod_i u_{i,z_i}$$

$$p_{ij} = \sum_{\mathbf{z}} \alpha_{\mathbf{z}} z_{ij} = \sum_{\mathbf{z}: z_i = j} \alpha_{\mathbf{z}} = w \sum_{\mathbf{z}: z_i = j} \prod_{i'} u_{i',z_{i'}} = w u_{ij} \sum_{\mathbf{z}: z_i = j} \prod_{i' \neq i} u_{i',z_{i'}} = w u_{ij} \cdot \text{Perm}(U_{-ij})$$

$$1 = \sum_{\mathbf{z}} \alpha_{\mathbf{z}} = \sum_j \sum_{\mathbf{z}: z_i = j} \alpha_{\mathbf{z}} = \sum_j w u_{ij} \cdot \text{Perm}(U_{-ij}) = w \cdot \text{Perm}(U).$$

Here $\text{Perm}(U)$ is the *permanent* of the matrix $U = (u_{ij})$, and U_{-ij} is the matrix U with rows i and j removed. Since $w = 1/\text{Perm}(U)$ we find

$$p_{ij} = \frac{u_{ij} \text{Perm}(U_{-ij})}{\text{Perm}(U)} \Rightarrow u_{ij} = \frac{p_{ij} \cdot \text{Perm}(U)}{\text{Perm}(U_{-ij})}$$

which can, in theory, be solved using a (very expensive) fixed point iteration scheme. Empirically we found that the solution takes the form $u_{ij} \approx p_{ij}^{1/\tau}$ for some τ, such that the probability of an assignment $\mathbf{z}$ is given by

$$\alpha_{\mathbf{z}} = w \prod_i u_{i,z_i} = \frac{\prod_i u_{i,z_i}}{\text{Perm}(U)} \approx \frac{\prod_i p_{i,z_i}^{1/\tau}}{\text{Perm}(\mathbf{P}^{(1/\tau)})} \propto \left(\prod_i p_{i,z_i}\right)^{1/\tau} = \exp\left(\frac{1}{\tau} \sum_i \log p_{i,z_i}\right). \tag{11}$$

Here $\mathbf{P}^{(1/\tau)}$ is the *Hadamard power*, which raises the entries p_{ij} to the power $1/\tau$ element-wise. By generalizing permanents (sums over permutations) to non-square matrices as sums over balanced assignment matrices, the same result can be derived for $n \neq k$. Using $a_{ij} = \log p_{ij}$ and converting to 'one-hot' representation z_{ij}, we find that (11) is equal to (5).

If we do not use the approximation $u_{ij} \approx p_{ij}^{1/\tau}$, then it still holds that the maximum entropy distribution has the form (5), but we should let $\tau = 1$ and $a_{ij} = \log u_{ij}$, so in this case $a_{ij} \neq \log p_{ij}$ are *not* the marginal log-probabilities we aim to sample from.

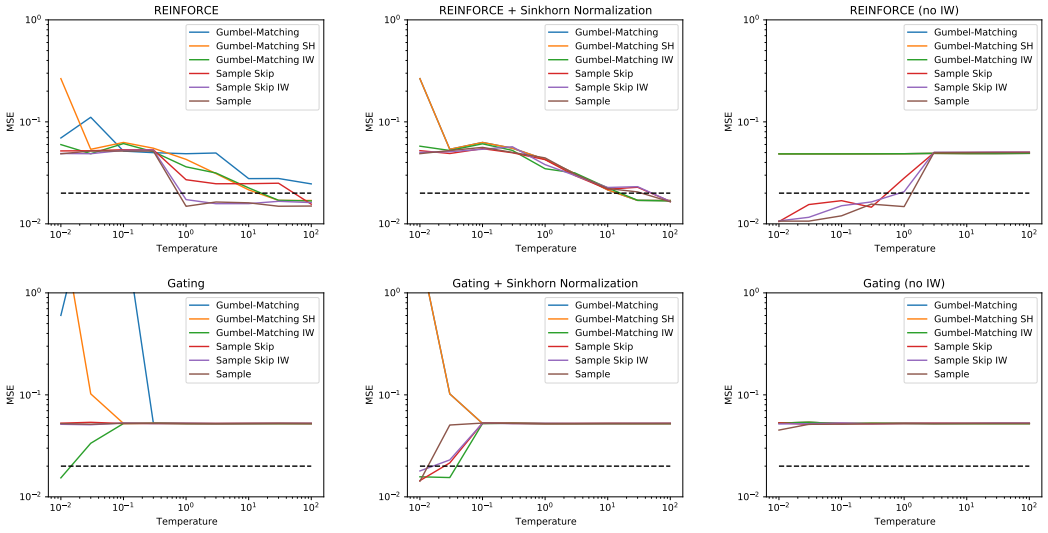


Figure 3: Results using REINFORCE (with no balance loss) and differentiable gating (with 0.01 balance loss weight) loss functions; both also shown in a version that applies Sinkhorn normalization before sampling, as well as a (biased) version that does not apply importance weights.

E Experiment

Figure 3 presents results for both REINFORCE (top) and differentiable gating (bottom) both with (middle) and without (left) the Sinkhorn normalization before sampling. When using Sinkhorn normalization before sampling, we take it into account when computing the importance weights. With REINFORCE, not using Sinkhorn normalization works better, as using Sinkhorn normalization requires a high sampling temperature to work well, i.e. close to uniform proposal samples.

With differentiable gating (and load balancing loss with weight 0.01), we observe the opposite: the results are better *with* Sinkhorn normalization than without Sinkhorn normalization (Section 3.1), but, contrasting REINFORCE, require a very low sampling temperature to succeed, so close to deterministic training.

Existing methods often do not use importance weights to take into account the noise scale or temperature [38, 8], so we experiment with this as well by completely dropping importance weights for all estimators (right column in Figure 3). We find this only works for REINFORCE, when sampling with a low temperature but not when we use the Gumbel-Matching estimators.

In all cases, we found REINFORCE succeeds both with and without a load balancing loss, whereas differentiable gating requires a load balancing loss with a weight of 0.01 to succeed at all. If the load balance loss is too high (0.1) all methods fail to model the unbalanced data.