Sinkhorn Networks: Using Optimal Transport Techniques to Learn Permutations

Gonzalo Mena[1]*, **David Belanger**[2], **Gonzalo Muñoz**[3], **Jasper Snoek**[2]
1. Columbia University, New York, NY, USA.
2. Google Brain, Cambridge, MA, USA.
3. Polytechnique Montréal, Montréal, Québec, Canada.

Abstract

Recently, Optimal Transport (OT) has received significant attention in the Machine Learning community. It has been shown to be useful as a tool for generative modeling, in which the density estimation problem is cast as the minimization of a linear function on the transportation polytope. Entropy regularization of this problem (Cuturi, 2013) has been demonstrated to be particularly useful, as its solution can be characterized in terms of the Sinkhorn operator, which i) can be computed more efficiently than the original problem and ii) enables efficient automatic differentiation (AD). We show that this technique extends to the Birkhoff polytope, and we use it to understand the solution of the linear assignment problem as a limit of the Sinkhorn operator. This observation justifies and enables the use of AD in computation graphs containing permutations as intermediate representations. As a result, we are able to introduce Sinkhorn networks for learning permutations, extending the work of Adams & Zemel (2011), and apply them to a variety of tasks. The success of our extension suggests entropy regularization might be used in other polytopes as well, enabling AD in other discrete structures.

1 Introduction

Optimal Transport (OT) Villani (2003) has received increased interest among the Machine Learning community, as it provides a renewed perspective to the question on how to compare two distributions. Indeed, the interpretation of the OT program as the minimum amount of total mass moved in order to transform one distribution into another Arjovsky et al. (2017) provides two advantages over the classical information paradigm for learning, based on minimization of KL divergence (e.g. maximum likelihood): first, it is not ill-posed when the true distribution lies on a low-dimensional manifold (Montavon et al., 2016; Genevay et al., 2017), and second, it provides a rich parameterization of the distance between distributions, given by the 'schedule' that minimizes the moved mass, the transportation plan.

The main drawback to applying OT is that it requires solving a linear problem that, although having polynomial complexity, in practice entails a substantial computational burden. An appealing solution was proposed by Cuturi (2013), where the original problem is replaced by an entropy-regularized version, whose solution is shown to be equivalent to the application of the so-called *Sinkhorn operator* (Sinkhorn, 1964), with a reduced computational cost. Since then, entropy regularization has gained popularity among practitioners, and more recently, has enabled automatic differentiation (AD) for the training of generative models based on OT Genevay et al. (2017), thanks to the differentiability of the Sinkhorn operator.

*gem2131@columbia.edu, Work done during an internship at Google Brain, Cambridge, MA.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

In this work, we extend this entropy regularization technique to show that the solution of the linear assignment problem can also be approximated with the Sinkhorn operator, enabling AD in computation graphs containing parameterized permutations as intermediate nodes, by replacing them by their differentiable approximations (section 2). Notably, by doing this we introduce *Sinkhorn networks* (section 3) which are able to learn the right permutation from training examples. We apply Sinkhorn networks to a variety of tasks, where we achieve state-of-the-art results (section 4).

2 An analog of the softmax for permutations

In this section we state our theoretical contribution, a rule to approximate matchings with the Sinkhorn operator, based on entropy regularization. We motivate it as an extension of a most elementary, discrete case, and defer a discussion of its relation to OT to section 5.

One sensible way to approximate a discrete category by continuous values is by using temperature-dependent softmax function, component-wise defined as $\text{softmax}_\tau(x_i) = \exp(x_i/\tau)/\sum_{j=1}\exp(x_j/\tau)$. For positive values of τ, $\text{softmax}_\tau(x_i)$ is a point in the probability simplex. Also, in the limit $\tau \to 0$, $\text{softmax}_\tau(x_i)$ converges to a vertex of the simplex, a one-hot vector, corresponding to the largest x_i [2]. This approximation is a key ingredient in the successful implementations by (Jang et al., 2016; Maddison et al., 2016), to enable AD in computation graphs containing discrete nodes, and here we extend it to permutations.

To do so, we first note that the Sinkhorn operator (the iterative normalization of rows and columns of a matrix) is an analogue of the softmax, for permutations. Specifically, for an N dimensional matrix X, we define [3] $\mathcal{T}_r(X) = X \oslash (X 1_N 1_N^\top)$, and $\mathcal{T}_v(X) = X \oslash (1_N 1_N^\top X)$ as the row and column-wise normalization operators of a matrix, with $\oslash$ denoting the element-wise division and 1_N a column vector of ones. Then, we define the Sinkhorn operator $S(\cdot)$ as follows:

$$\begin{aligned} S^0(X) &= \exp(X), \\ S^m(X) &= \mathcal{T}_c\left(\mathcal{T}_r(S^{n-1}(X))\right), \\ S(X) &= \lim_{m\to\infty} S^m(X). \end{aligned} \tag{1}$$

A theorem due to (Sinkhorn, 1964) [4] , proves that $S(X)$ must belong to the Birkhoff polytope, the set of doubly stochastic matrices, that we denote $\mathcal{B}_N$.

To continue our analogy with categories, first notice that choosing a category can always be cast as a maximization problem: the choice $\arg\max_i x_i$ is the one that maximizes the function $\langle x, v\rangle$ (with v being a one-hot vector), since the maximizing v^* is the one that indexes the largest x_i. By paralleling this construction, one may parameterize the choice of a permutation P through a matrix X, as the solution to the linear assignment problem (Kuhn, 1955), with $\mathcal{P}_N$ denoting the set of permutation matrices:

$$M(X) = \arg\max_{P\in\mathcal{P}_N} \text{trace}(P^\top X). \tag{2}$$

Our theoretical contribution is to notice that the hard choice of a permutation, $M(X)$, can be obtained as the limit of $S(X/\tau)$, meaning that one can approximate $M(X) \approx S(X/\tau)$ with a small τ. Theorem 1 summarizes our finding. We provide a rigorous proof in Appendix A, where we also comment on its relation to the simpler discrete case. This proof is based on showing that $S(X/\tau)$ solves a certain entropy-regularized problem in $\mathcal{B}_n$, which in the limit converges to the matching problem in equation 2.

Theorem 1. For a doubly-stochastic matrix P, define its entropy as $h(P) = -\sum_{i,j} P_{i,j}\log(P_{i,j})$. Then, one has,

$$S(X/\tau) = \arg\max_{P\in\mathcal{B}_N} \text{trace}(P^\top X) + \tau h(P). \tag{3}$$

Now, assume also the entries of X are drawn independently from a distribution that is absolutely continuous with respect to the Lebesgue measure in $\mathbb{R}$. Then, almost surely, the following convergence

[2] Unless the degenerate case of ties.

[3] Notation borrowed from Adams & Zemel (2011)

[4] This theorem requires certain technical conditions which are trivially satisfied if X has positive entries, motivating the use of the component-wise exponential $\exp(\cdot)$ in the first line of equation 1.

holds:

$$M(X) = \lim_{\tau \to 0^+} S(X/\tau). \tag{4}$$

3 Sinkhorn Networks

Now we show how to apply the approximation stated in Theorem 1 in the context of artificial neural networks. We construct a layer that encodes the representation of a permutation, and show how to train networks containing such layers as intermediate representations.

We define the components of this network through a minimal example: consider the supervised task of learning a mapping from scrambled objects $\tilde{X}$ to actual, non-scrambled X. Data, then, are M pairs $(X_i, \tilde{X}_i)$ where $\tilde{X}_i$ can be constructed by randomly permuting pieces of X^i. We state this problem as a permutation-valued regression $X^i = \pi^{-1}_{\theta,\tilde{X}_i}\left(\tilde{X}_i\right) + \varepsilon_i$, where ε_i is a noise term, and $\pi_{\theta,\tilde{X}_i}$ is the permutation (represented as a matrix) that maps X_i to $\tilde{X}_i$, and that depends on $\tilde{X}_i$ and some parameters θ. We are concerned with minimization of the reconstruction error [5]:

$$f(\theta, X, \tilde{X}) = \sum_{i=1}^{M} ||X_i - \pi^{-1}_{\theta,\tilde{X}_i}\left(\tilde{X}_i\right)||^2. \tag{5}$$

One way to express a complex parameterization of this kind is through a neural network: this network receives $\tilde{X}^i$ as input, which is then passed through some intermediate, feed-forward computations of the type $g_l(W_l x_l + b_l)$, where g_l are nonlinear activation functions, x_l is the output of a previous layer, and $\theta = \{(W_l, b_l)\}$ are the network parameters. To make the final network output be a permutation, we appeal to constructions developed in section 2: by assuming that the final network output $\pi_{\theta,\tilde{X}}$ can be parameterized as the solution of the assignment problem; i.e., $\pi_{\theta,\tilde{X}}\left(\tilde{X}\right) = M(g(\tilde{X}, \theta))$, where $g(\tilde{X}, \theta)$ is the output of the computations involving f_l.

Unfortunately, the above construction involves a non-differentiable f (in θ). We use Theorem 1 as a justification for replacing $M(g(\tilde{X}, \theta))$ by the differentiable $S(g(\tilde{X}, \theta)/\tau)$ in the computation graph. The value of τ must be chosen with caution: if τ is too small, gradients vanishes almost everywhere, as $S(g(\tilde{X}, \theta)/\tau)$ approaches the non-differentiable $M(g(\tilde{X}, \theta))$. Conversely, if τ is too large, $S(X/\tau)$ may be far from the vertices of the Birkhoff polytope, and reconstructions $\pi^{-1}_{\theta,\tilde{X}}\left(\tilde{X}\right)$ may be nonsensical (see Figure 1a). Importantly, we will always add noise to the output layer $g(\tilde{X}, \theta)$ as a regularization device: by doing so we ensure uniqueness of $M(g(\tilde{X}, \theta))$, which is required for convergence in Theorem 1.

3.1 Permutation equivariance

Among all possible architectures that respect the aforementioned parameterization, we will only consider networks that are *permutation equivariant*, the natural kind of symmetry arising in this context. Specifically, we require networks to satisfy:

$$\pi_{\theta,\pi'(\tilde{X})}\left(\pi'(\tilde{X})\right) = \pi'\left(\pi_\theta(\tilde{X}\right),$$

where π' is an arbitrary permutation. The underlying intuition is simple: reconstructions of objects should not depend on how pieces were scrambled, but only on the pieces themselves. We achieve permutation equivariance by using the same network to process each piece of $\tilde{X}$, which we require to have an N dimensional output. Then, these N outputs (each with N components) are used to conform the rows of the matrix $g(\tilde{X}, \theta)$, to which we finally apply the (differentiable) Sinkhorn operator. One can interpret each row as representing a vector of local likelihoods of assignment, but they might be inconsistent. The Sinkhorn operator, then, mixes those separate representations are mixed, and ensures that consistent (approximate) assignment are produced.

[5] This error arises from gaussian ε_i. Other choices may be possible, but here we stick to the straightest formulation

	$N = 80$		$N = 100$		$N = 120$	
Test distribution	P.W.	P.A.W.	P.W.	P.A.W.	P.W.	P.A.W.
$U(0, 1)$	.0	.0	.0	.0	.0	.01
$U(0, 10)$	.0	.0	.0	.02	.0	.03
$U(0, 1000)$	.0	.01	.0	.02	.0	.04
$U(1, 2)$	.0	.01	.0	.04	.0	.08
$U(10, 11)$	.0	.08	.0	.08	.1	.6
$U(100, 101)$	.02	.65	.09	.99	.12	1.
$U(1000, 1001)$	.22	1.	.39	1.	.49	1.

Table 1: Results in the number sorting task, for test data sampled from different uniform distributions. *Results from Vinyals et al. (2015)

With permutation equivariance, the only consideration that will be left to the practitioner is the choice of the particular architecture, which will depend on the particular kind of data. In Section 4 we illustrate the uses of Sinkhorn Networks with three examples, each of them using a different architecture.

4 Experiments

4.1 Sorting numbers

To illustrate the capabilities of Sinkhorn Networks in the most elementary case, we considered the task of sorting numbers using artificial neural networks, first described in (Vinyals et al., 2015). Specifically, we sample uniform random numbers $\tilde{X}$ in the $[0, 1]$ interval and we train our network with pairs $(\tilde{X}, X)$ where X are the same $\tilde{X}$ but in order. The network has a first fully connected layer that links a number with an intermediate representation (with 32 units), and a second (also fully connected) layer that turns that representation into a row of the matrix $g(\tilde{X}, \theta)$.

Table 1 shows our network learns to sort up to $N = 120$ numbers. We used two evaluation measures: the proportion of wrong responses (P.W.), and the proportion of sequences where there was at least one error (P.A.W.). Surprisingly, the network learns to sort number even when test examples are not sampled from $U(0, 1)$, but on a considerably different interval. This indicates the network is not overfitting. These results may be compared with the ones in Vinyals et al. (2015), where a much more complex (recurrent) network was used, but performance guarantees were obtained only with at most $N = 15$ numbers. In that case, the reported P.A.W is 0.9, ours is negligible for most test distributions.

4.2 Jigsaw Puzzles

A more complex scenario for learning a permutation relates to images, which has been addressed in (Noroozi & Favaro, 2016; Cruz et al., 2017). There, we would like to solve a Jigsaw puzzle, to recover an image X given their scrambled pieces $\tilde{X}$, at a certain level of atomicity. In this example, our network differs from the one in 4.1 slightly: now, the first layer is a simple CNN (convolution + max pooling), which maps the puzzle pieces to an intermediate representation.

For evaluation on test data we report $l1$ and $l2$ (train) losses and the Kendall tau, a 'correlation coefficient' for ranked data. In Table 2 we benchmark results for the MNIST, Celeba and Imagenet datasets, with puzzles between 2x2 and 6x6 pieces. In MNIST we achieve very low $l1$ and $l2$ errors on up to 6x6 puzzles, although a high proportion of errors. This is a consequence of our loss being agnostic to particular permutations, but only caring about reconstruction errors: as the number of black pieces increases with the number of puzzle pieces, most of them become unidentifiable under this loss.

In Celeba, we are able to solve puzzles of up to 5x5 pieces with only 21% of pieces of faces being incorrectly ordered (see Figure 1a for examples of reconstructions). However, learning in the the Imagenet dataset is much more challenging, as there isn't a sequential structure that generalizes among images, unlike Celeba and MNIST. In this dataset, our network ties with the .72 Kendall tau

score reported in (Cruz et al., 2017). Their network, named DeepPermNet, is based on stacking up to the sixth fully connected layer of AlexNet (Krizhevsky et al., 2012), which finally fully connects to a Sinkhorn layer through two additional layers. We note, however, that our network is much simpler, with only two layers and far fewer parameters.

6

	MNIST					Celeba				Imagenet	
	2x2	3x3	4x4	5x5	6x6	2x2	3x3	4x4	5x5	2x2	3x3
Prop. wrong	.0	.09	.45	.45	.59	.0	.03	.1	.21	.12	.26
Prop. any wrong	.0	.28	.97	1.	1.	.0	.09	.36	.73	.19	.53
Kendall tau	1.	.83	.43	.39	.27	1.0	.96	.88	.78	.85	**.73** (0.72*)
$l1$	.0	.0	.04	.02	.03	.0	.01	.04	.08	.05	.12
$l2$	.0	.0	.26	.18	.19	.0	.11	.18	.24	.22	.31

Table 2: Jigsaw puzzle results. *Result from Cruz et al. (2017)

4.3 Assembly of arbitrary MNIST digits from pieces

We also consider an original application, motivated by the observation that the Jigsaw Puzzle task becomes ill-posed if a puzzle contains too many pieces. Indeed, consider the binarized MNIST dataset: there, reconstructions are not unique if pieces are sufficiently atomic, and in the limit case of pieces of size 1x1 squared pixels, for a given scrambled MNIST digit there are as many valid reconstructions as MNIST digits are there with the same number of white pixels. In other words, reconstructions stop being probabilistic and become a multimodal distribution over permutations.

We exploit this intuition to ask whether a neural network can be trained to achieve arbitrary digit reconstructions, given their loose atomic pieces. To address this question, we slightly changed the network in 4.2, this time stacking several second layers linking an intermediate representation to the output. We trained the network to reconstruct a particular digit with each layer, by using digit identity to indicate which layer should activate with a particular training example.

Our results demonstrate a positive answer: Figure 1b shows reconstructions of arbitrary digits given 10x10 scrambled pieces. In general, they can be unambiguously identified by the naked eye. Moreover, this judgement is supported by the assessment of a neural network. Specifically, we trained a two-layer CNN [6] on MNIST (achieving a 99.2% accuracy on test set) and evaluated its performance on the test set generated by arbitrary transformations of each digit of the original test set into any other digit. We found the CNN made an appropriate judgement in 85.1% of the times.

5 Related work

We recognize two relevant sources of influence, besides the motivation given by the success of Jang et al. (2016) and Maddison et al. (2016): from OT and previous uses of the Sinkhorn operator for inference of permutations. Below, we explain how our work benefits from them, and elaborate on the relation between them.

The first part of Theorem 1 is very similar to an intermediate result from Cuturi (2013), linking the entropy-regularized transportation problem and the computation of a 'Sinkhorn distance'. Our contribution is to verify that the same argument does not only apply to the transportation polytope, the optimization set that arises in the mostly adopted Kantorovich formulation of OT (Genevay et al., 2016), but can also be applied to the Birkhoff polytope as well. Also, unlike usual OT that is bound to specific choices of cost functions (or transportation metrics), we are freed from attaching such an interpretation to our matrix X, which for us simply parameterizes a matching.

There are clear connections between the usual Kantorovich OT problem and ours, though: first, the Birkhoff polytope simply corresponds to the transportation polytope between uniform histograms $U(1_N/N, 1_N/N)$, multiplied by N. Also, the set of permutations is the one that naturally appears

[6] Specifically, we used the one described in the Deep MNIST for experts tutorial. in Tensorflow's (Abadi et al., 2016) online documentation.

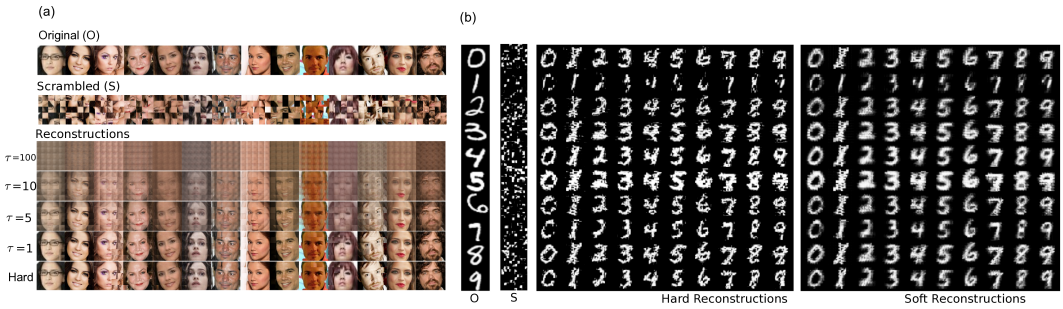

Figure 1: Results on images. (a) Sinkhorn networks can be trained to solve Jigsaw Puzzles. Given a trained model, 'soft' reconstructions are shown at different τ using $S(X/\tau)$. We also show hard reconstructions, made by computing $M(X)$ with the Hungarian algorithm (Munkres, 1957). (b) Sinkhorn networks can also be used to learn to transform any MNIST digit into another. We show hard and soft reconstructions, with $\tau = 1$.

when stating the discrete OT problem with the slightly stricter Monge formulation (Villani, 2008), which explicitly requires a one-to-one assignment. However, to our understanding, this extension has not been established elsewhere, with the exception of recent work by Ferradans et al. (2014). That work, however, used a different penalization, losing access to the Sinkhorn operator.

Connections between permutations and the Sinkhorn operator have been known for at least twenty years. Indeed, first, the limit stated in Theorem 1 was also presented in Kosowsky & Yuille (1994); however, their interpretation and motivation were more linked to statistical physics and economics. Second, in Gold et al. (1996) a similar theorem is introduced, but their proof is not rigorous. Third, we understand our work as extending the pioneering contribution of Adams & Zemel (2011), which enabled neural networks to learn a permutation-like structure; a ranking. However, there, as in Helmbold & Warmuth (2009) as well, the objective function was linear, and the Sinkhorn operator was instead used to approximate the expectation of the objective. In consequence, there was no need to introduce a temperature parameter and consider a limit argument, which is critical to our case.

The extension from Adams & Zemel (2011); i.e., training neural networks to learn permutations with nonlinear objectives, was simultaneously introduced in Cruz et al. (2017), although their work substantially differs from ours: while their interest lies on the representational aspects of CNN's, we are more concerned with the more fundamental properties. In their work, they don't consider a temperature parameter τ, but their network still successfully learns, as $\tau = 1$ happens to fall within the range of reasonable values. We hope our more general theory; particularly, our limit argument and the notion of equivariance, may aid further developments aligned with the work of Cruz et al. (2017).

Finally, we mention that a connection between the approach by Kosowsky & Yuille (1994) and OT exists: In Genevay et al. (2016), related formulations of the entropy regularized OT were given, based on the notion of Fenchel duality (Rockafellar, 1970). One of these formulations, named 'semi-dual' in Genevay et al. (2016), essentially corresponds to the one presented in Kosowsky & Yuille (1994).

6 Discussion

We have shown that techniques that improved computational aspects of OT can also be used to tackle problems involved in permutations. This is critical in the AD era, which requires us to formulate ways in which the computation of a discrete entity can be thought of as the limit of another lying in the continuum. We hypothesize other extensions of the entropy regularization method may be possible to address more general structures: for example, the ones that put cardinality constraints on objects (Swersky et al., 2012; Tarlow et al., 2012).

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

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

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Garrett Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucumán. Revista A*, 5: 147–151, 1946.

Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. Deeppermnet: Visual permutation learning. *arXiv preprint arXiv:1704.02729*, 2017.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pp. 2292–2300, 2013.

Sira Ferradans, Nicolas Papadakis, Gabriel Peyré, and Jean-François Aujol. Regularized discrete optimal transport. *SIAM Journal on Imaging Sciences*, 7(3):1853–1882, 2014.

Aude Genevay, Marco Cuturi, Gabriel Peyré, and Francis Bach. Stochastic optimization for large-scale optimal transport. In *Advances in Neural Information Processing Systems*, pp. 3440–3448, 2016.

Aude Genevay, Gabriel Peyré, and Marco Cuturi. Sinkhorn-autodiff: Tractable wasserstein learning of generative models. *arXiv preprint arXiv:1706.00292*, 2017.

Steven Gold, Anand Rangarajan, et al. Softmax to softassign: Neural network algorithms for combinatorial optimization. *Journal of Artificial Neural Networks*, 2(4):381–399, 1996.

David P Helmbold and Manfred K Warmuth. Learning permutations with exponential weights. *Journal of Machine Learning Research*, 10(Jul):1705–1736, 2009.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

JJ Kosowsky and Alan L Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural networks*, 7(3):477–490, 1994.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2):83–97, 1955.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Grégoire Montavon, Klaus-Robert Müller, and Marco Cuturi. Wasserstein training of restricted boltzmann machines. In *Advances in Neural Information Processing Systems*, pp. 3718–3726, 2016.

James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pp. 69–84. Springer, 2016.

C Radhakrishna Rao. Convexity properties of entropy functions and analysis of diversity. *Lecture Notes-Monograph Series*, pp. 68–77, 1984.

Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 1970.

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

Kevin Swersky, Ilya Sutskever, Daniel Tarlow, Richard S Zemel, Ruslan R Salakhutdinov, and Ryan P Adams. Cardinality restricted boltzmann machines. In *Advances in neural information processing systems*, pp. 3293–3301, 2012.

Daniel Tarlow, Kevin Swersky, Richard S Zemel, Ryan P Adams, and Brendan J Frey. Fast exact inference for recursive cardinality models. *arXiv preprint arXiv:1210.4899*, 2012.

Cédric Villani. *Topics in optimal transportation*. Number 58. American Mathematical Soc., 2003.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

Acknowledgements

The authors would like to thank Ryan Adams, Kevin Swersky and Scott Linderman for valuable discussions, and Danny Tarlow also for revision of the manuscript.

A Proof of Theorem 1

In this section we give a rigorous proof of Theorem 1. Also, in **??** we briefly comment on how Theorem 1 extend a perhaps more intuitive results, in the probability simplex.

Before stating Theorem 1 we need some preliminary definitions. We start by recalling a well-known result in matrix theory, the Sinkhorn theorem.

Sinkhorn's theorem, (Sinkhorn, 1964): Let A be an N dimensional square matrix with positive entries. Then, there exists two diagonal matrices D_1, D_2, with positive diagonals, so that $P = D_1 A D_2$ is a doubly stochastic matrix. These D_1, D_2 are unique up to a scalar factor. Also, P can be obtained through the iterative process of alternatively normalizing the rows and columns of A. For our purposes, it is useful to define the Sinkhorn operator $S(\cdot)$ as follows:

Definition 1: Let A be an arbitrary matrix with dimension N. Denote $\mathcal{T}_r(X) = X \oslash (X 1_N 1_N^\top)$, $\mathcal{T}_c(X) = X \oslash (1_N 1_N^\top A)$ (with $\oslash$ representing the element-wise division and 1_n the n dimensional vector of ones) the row and column-wise normalization operators, respectively. Then, we define the Sinkhorn operator applied to A; $S(X)$, as follows:

$$\begin{aligned} S^0(X) &= \exp(X), \\ S^n(X) &= \mathcal{T}_c\left(\mathcal{T}_r(S^{n-1}(X))\right), \\ S(X) &= \lim_{n\to\infty} S^n(X). \end{aligned}$$

Here, the $\exp(\cdot)$ operator is interpreted as the component-wise exponential. Notice that by Birkhoff's theorem, $S(X)$ is a doubly stochastic matrix.

Finally, we review some key properties related to the space of doubly stochastic matrices.

Definition 2: We denote by $\mathcal{B}_N$ the N-Birkhoff polytope, i.e., the set of doubly stochastic matrices of dimension N. Likewise, we denote $\mathcal{P}_n$ be the set of permutation matrices of size N. Alternatively,

$$\mathcal{B}_N = \{P \in [0,1] \in \mathbb{R}^{N,N} \; P^\top 1_N = 1_N, P^\top 1_N = 1_N\},$$

$$\mathcal{P}_N = \{P \in \{0,1\} \in \mathbb{R}^{N,N} \; P^\top 1_N = 1_N, P^\top 1_N = 1_N\}.$$

(Birkhoff's Theorem, Birkhoff (1946)) $\mathcal{P}_N$ is the set of extremal points of $\mathcal{B}_N$. In other words, the convex hull of $\mathcal{B}_N$ equals $\mathcal{P}_N$.

A.1 An approximation theorem for the matching problem

Let's now focus on the standard combinatorial assignment (or matching) problem, for an arbitrary N dimensional matrix X. We aim to maximize a linear functional (in the sense of the Frobenius norm) in the space of permutation matrices. In this context, let's define the matching operator $M(\cdot)$ as the one that returns the solution of the assignment problem:

$$M(X) \equiv \arg\max_{P \in \mathcal{P}_N} trace\left(P^\top X\right). \tag{6}$$

Likewise, we define $\tilde{M}(\cdot)$ as a related operator, but changing the feasible space by the Birkhoff polytope:

$$\tilde{M}(X) \equiv \arg\max_{P \in \mathcal{B}_N} trace\left(P^\top X\right). \tag{7}$$

Notice that in general $\tilde{M}(X), M(X)$ might not be unique matrices, but a face of the Birkhoff polytope, or a set of permutations, respectively (see Lemma 2 for details). In any case, the relation $M(X) \subseteq \tilde{M}(X)$ holds by virtue of Birkhoff's theorem, and the fundamental theorem of linear programming.

Now we state the main theorem of this work:

Theorem 1. For a doubly stochastic matrix P define its entropy as $h(P) = -\sum_{i,j} P_{i,j} \log\left(P_{i,j}\right)$. Then, one has,

$$S(X/\tau) = \arg\max_{P \in \mathcal{B}_N} trace(P^\top X) + \tau h(P). \tag{8}$$

Now, assume also the entries of X are drawn independently from a distribution that is absolutely continuous with respect to the Lebesgue measure in $\mathcal{R}$. Then, almost surely the following convergence holds:

$$M(X) = \lim_{\tau \to 0^+} S(X/\tau). \tag{9}$$

We divide the proof of Theorem 1 in three steps. First, in Lemma 1 we state a relation between $S(X/\tau)$ and the entropy regularized problem in equation (8). Then, in Lemma 2 we show that under our stochastic regime, uniqueness of solutions holds. Finally, in Lemma 3 we show that in this well-behaved regime, convergence of solutions holds. states that and Lemma 2b endows us with the tools to make a limit argument.

A.1.1 Intermediate results for Theorem 1

Lemma 1:

$$S(X/\tau) = \arg\max_{P \in \mathcal{B}_N} trace(P^\top X) + \tau h(P).$$

Proof: We first notice that the solution P_τ of the above problem exists, and it is unique. This is a simple consequence of the strict concavity of the objective (recall the entropy is strictly concave Rao (1984)).

Now, let's state the Lagrangian of this constrained problem

$$\mathcal{L}(\alpha, \beta, P) = trace(P^\top X) + \tau h(P) + \alpha^\top (P 1_N - 1_N) + \beta^\top (P^\top 1_N - 1_N),$$

It is easy to see, by stating the equality $\partial \mathcal{L}/\partial P = 0$ that one must have for each i, j,

$$p_\tau^{i,j} = \exp(\alpha_i/\tau - 1/2) \exp(X_{i,j}/\tau) \exp(\beta_j/\tau - 1/2),$$

in other words, $P_\tau = D_1 \exp(X_{i,j}/\tau) D_2$ for certain diagonal matrices D_1, D_2, with positive diagonals. By Sinkhorn's theorem, and our definition of the Sinkhorn operator, we must have that $S(X/\tau) = P_\tau$.

Lemma2: Suppose the entries of X are drawn independently from a distribution that is absolutely continuous with respect to the Lebesgue measure in $\mathbb{R}$. Then, almost surely, $\tilde{M}(X) = M(X)$ is a unique permutation matrix.

Proof: This is a known result from sensibility analysis on linear programming which we prove for completeness. Notice first that the problem in (2) is a linear program on a polytope. As such, by the fundamental theorem of linear program, the optimal solution set must correspond to a face of

the polytope. Let $\mathcal{F}$ be a face of $\mathcal{B}_N$ of dimension ≥ 1, and take $P_1, P_2 \in \mathcal{F}$, $P_1 \neq P_2$. If $\mathcal{F}$ is an optimal face for a certain $X_{\mathcal{F}}$, then $X_{\mathcal{F}} \in \{X \,:\, trace(P_1^T X) = trace(P_2^T X)\}$. Nonetheless, the latter set does *not* have full dimension, and consequently has measure zero, given our distributional assumption on X. Repeating the argument for every face of dimension ≥ 1 and taking a union bound we conclude that, almost surely, the optimal solution lies on a face of dimension 0, i.e, a vertex. From here uniqueness follows.

Lemma 3 Call P_τ the solution to the problem in equation 8, i.e. $P_\tau = P_\tau(X) = S(X/\tau)$. Under the assumptions of Lemma 2, $P_\tau \to P_0$ when if $\tau \to 0^+$.

Proof Notice that by Lemmas 1 and 2, P_τ is well defined and unique for each $\tau \geq 0$. Moreover, at $\tau = 0$, $P_0 = M(X)$ is the unique solution of a linear program. Now, let's define $f_\tau(\cdot) = trace(\cdot^\top X) + \tau h(\cdot)$. We observe that $f_0(P_\tau) \to f_0(P_0)$. Indeed, one has:

$$\begin{aligned} f_0(P_0) - f_0(P_\tau) &= trace(P_0^\top X) - trace(P_\tau^\top X) \\ &= trace(P_0^\top X) - f_\tau(P_\tau) + \tau h(P_\tau) \\ &< trace(P_0^\top X) - f_\tau(P_0) + \tau h(P_\tau) \\ &< \tau \left(h(P_\tau) - h(P_0)\right) \\ &< \tau \max_{P \in \mathcal{B}_N} h(P). \end{aligned}$$

From which convergence follows trivially. Moreover, in this case convergence of the values implies the converge of P_τ: suppose P_τ does not converge to P_0. Then, there would exist a certain δ and sequence $\tau_n \to 0$ such that $\|P_{\tau_n} - P_0\| > \delta$. On the other hand, since P_0 is the unique maximizer of an LP, there exists $\varepsilon > 0$ such that $f_0(P_0) - f_0(P) > \varepsilon$ whenever $\|P - P_0\| > \delta$, $P \in \mathcal{B}_N$. This contradicts the convergence of $f_0(P_{\tau_n})$.

A.1.2 Proof of Theorem 1

The first statement is Lemma 1. Convergence (equation 9) is a direct consequence of Lemma 3, after noticing $P_\tau = S(X/\tau)$ and $P_0 = M(X)$.

A.2 Relation to softmax

Finally, we notice that all of the above results can be understood as a generalization of the well-known approximation result $\arg\max_i x_i = \lim_{\tau \to 0^+} softmax(x/\tau)$. To see this, treat a category as a one-hot vector. Then, one has

$$\arg\max_i x_i = \arg\max_{e \in \mathcal{S}_N} \langle e, x \rangle, \tag{10}$$

where $\mathcal{S}_n$ is the probability simplex, the convex hull of the one-hot vectors (denoted $\mathcal{H}_n$). Again, by the fundamental theorem of linear algebra, the following holds

$$\arg\max_i x_i = \arg\max_{e \in \mathcal{H}_N} \langle e, x \rangle. \tag{11}$$

On the other hand, by a similar (but simpler) argument than of the proof of theorem 4 one can easily show that

$$softmax(x/\tau) \equiv \frac{\exp(x/\tau)}{\sum_{i=1} \exp(x_i/\tau)} = \arg\max_{e \in \mathcal{S}_n} \langle e, x \rangle + \tau h(x), \tag{12}$$

where the entropy $h(\cdot)$ is not defined as $h(x) = -\sum_{i=1}^n x_i \log(x_i)$