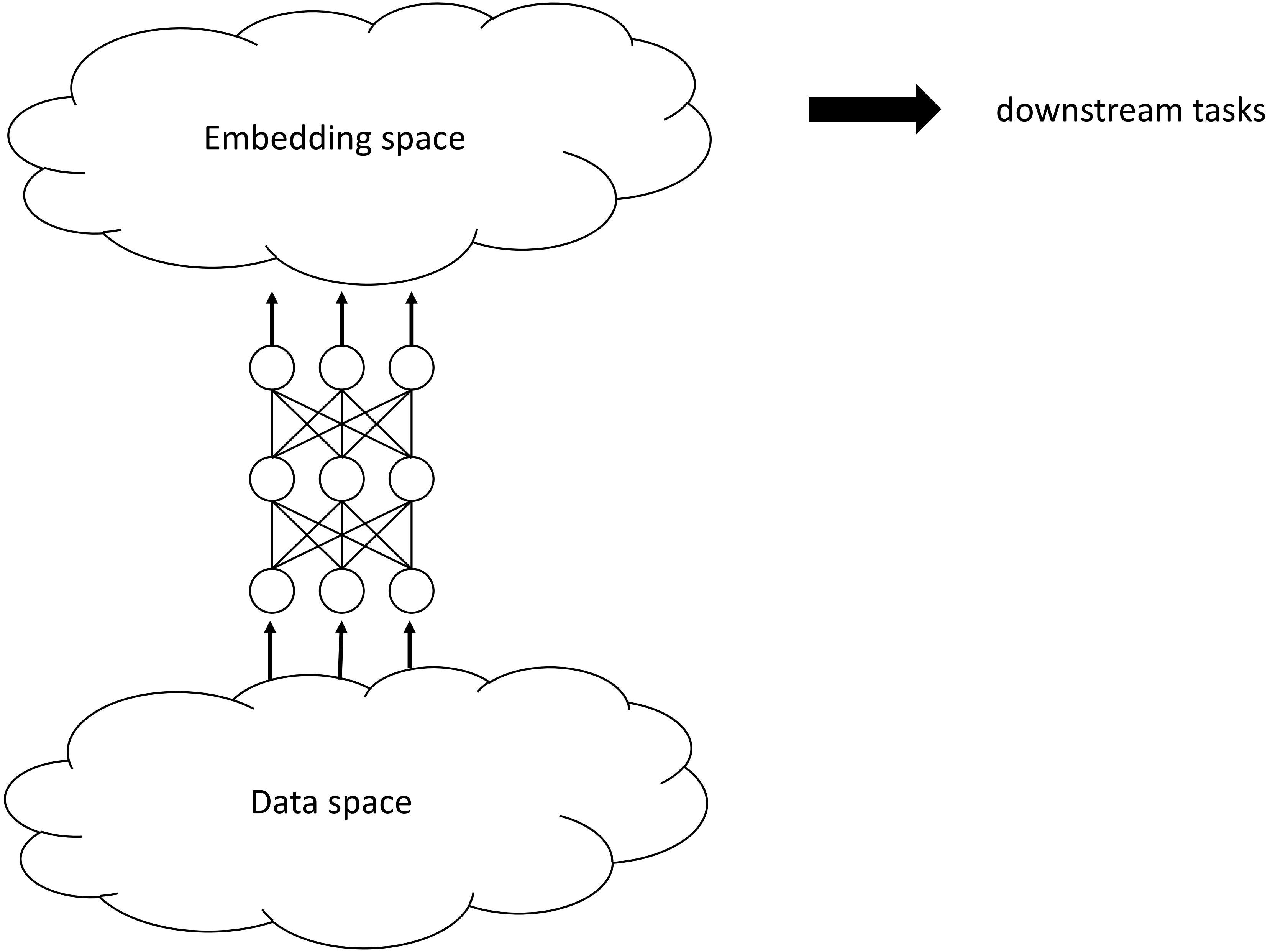
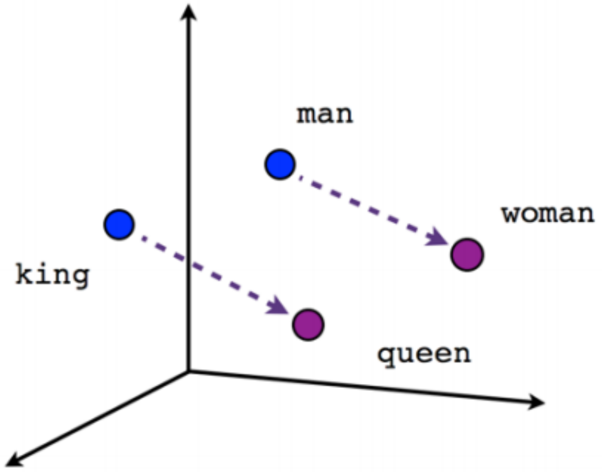
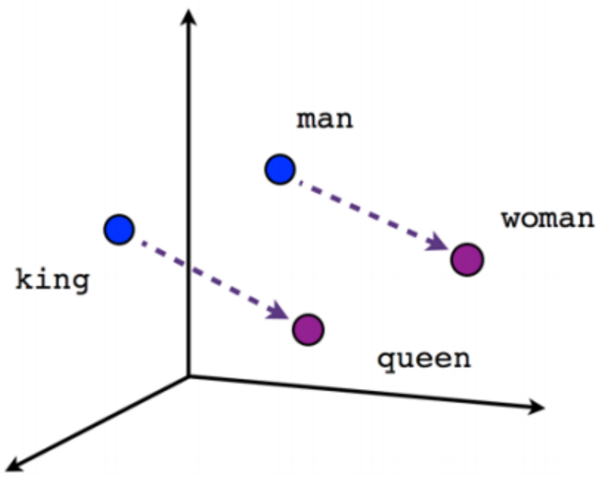
Learning Embeddings into Entropic Wasserstein Spaces

Charlie Frogner, Farzaneh Mirzazadeh, Justin Solomon


Embedding space
downstream tasks
Data space

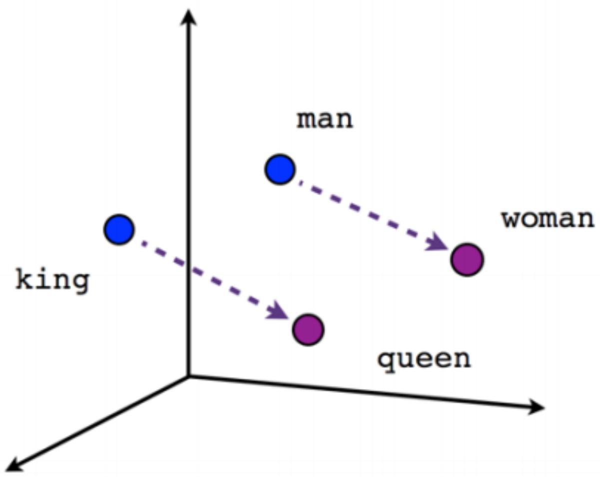


Word embedding: word2vec *(Mikolov 2013)*, GloVe *(Pennington 2014)*, fastText *(Bojanowski 2017)*, ELMo *(Peters 2018)*



Word embedding: word2vec *(Mikolov 2013)*, GloVe *(Pennington 2014)*, fastText *(Bojanowski 2017)*, ELMo *(Peters 2018)*

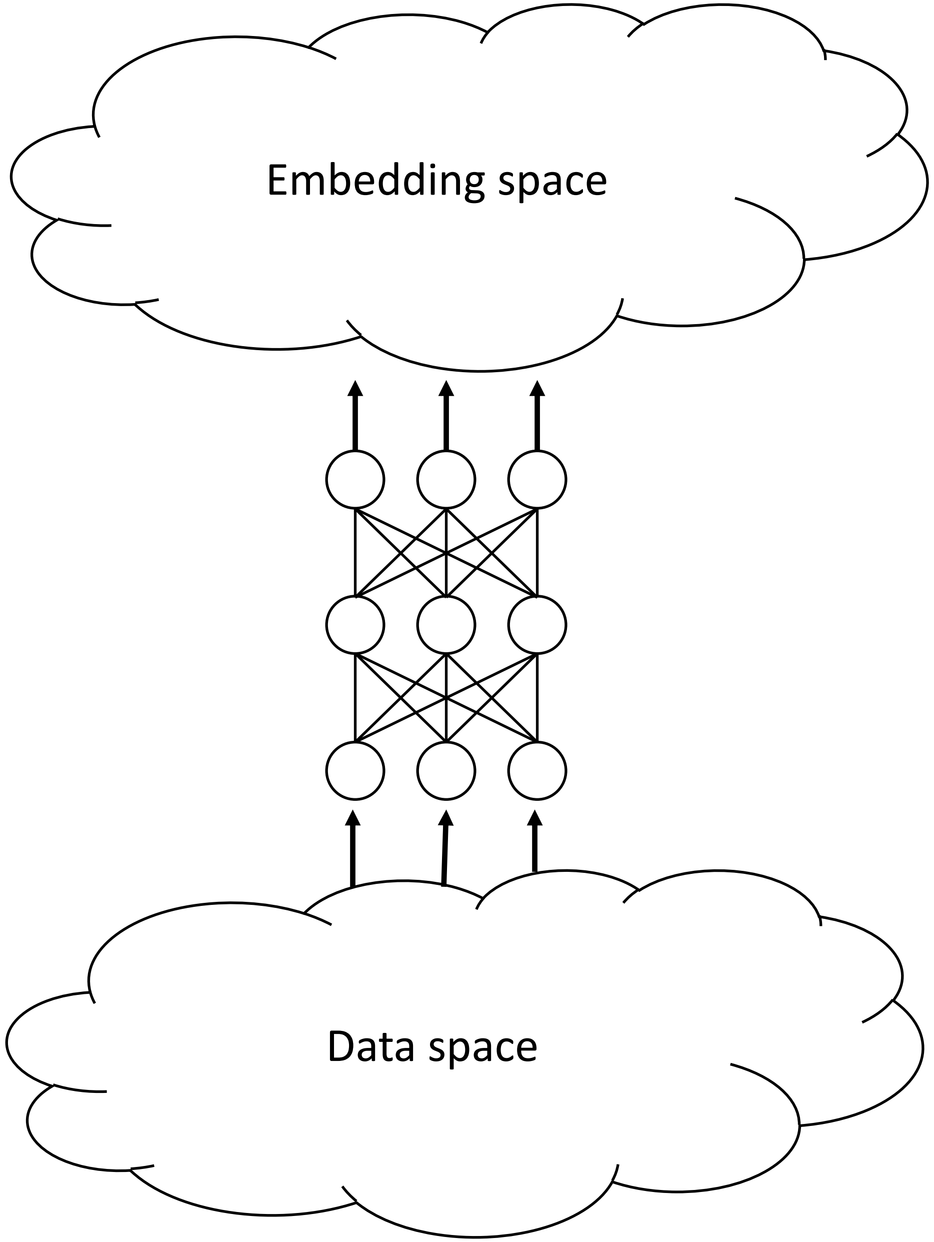
Graph embedding: Laplacian eigenmaps *(Belkin & Niyogi 2001)*, DeepWalk *(Perozzi 2014)*, node2vec *(Grover & Leskovec 2016), G*raph Convolutional Networks *(Kipf 2017),* Poincare embedding *(Nickel 2017)*

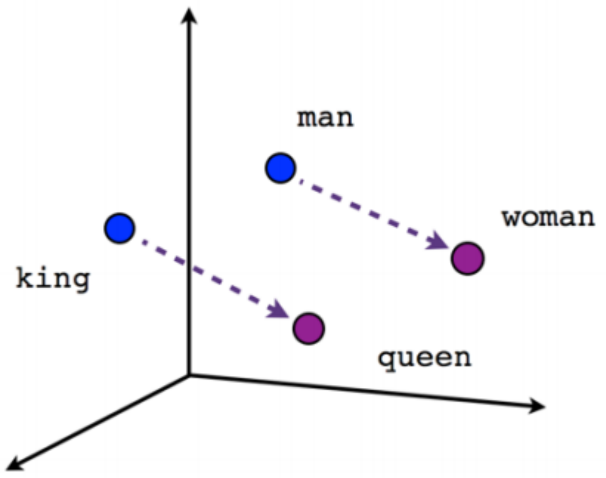


Word embedding: word2vec *(Mikolov 2013)*, GloVe *(Pennington 2014)*, fastText *(Bojanowski 2017)*, ELMo *(Peters 2018)*

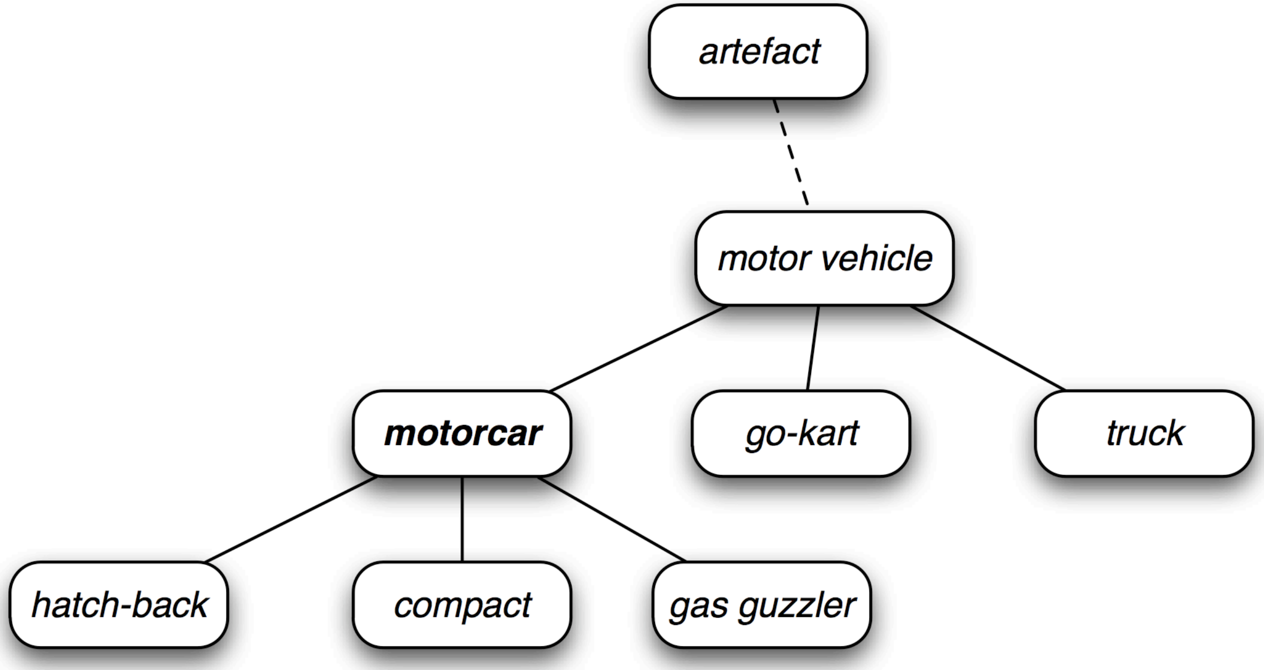
Graph embedding: Laplacian eigenmaps *(Belkin & Niyogi 2001)*, DeepWalk *(Perozzi 2014)*, node2vec *(Grover & Leskovec 2016)*, Graph Convolutional Networks *(Kipf 2017)*, Poincare embedding *(Nickel 2017)*

Image representation: AlexNet *(Krizhevsky 2012)*, VGG *(Simonyan 2014)*, ResNet *(He 2015)*.

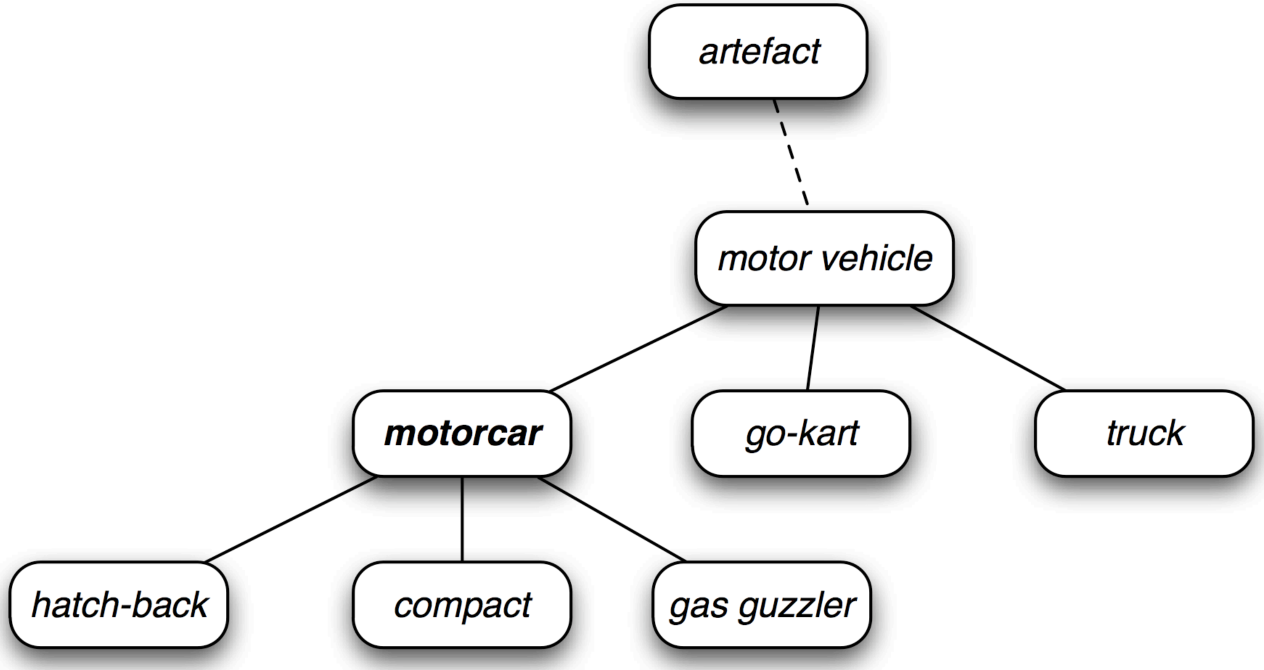




Euclidean:
word2vec *(Mikolov 2013)*
GloVe *(Pennington 2014)*
fastText *(Bojanowski 2017)*
ELMo *(Peters 2018)*
Laplacian eigenmaps *(Belkin & Niyogi 2001)*
DeepWalk *(Perozzi 2014)*
node2vec *(Grover & Leskovec 2016)*
Graph Convolutional Nets *(Kipf 2017)*
…


artefact
motor vehicle
motorcar
go-kart
truck
hatch-back
compact
gas guzzler

Euclidean



Hyperbolic

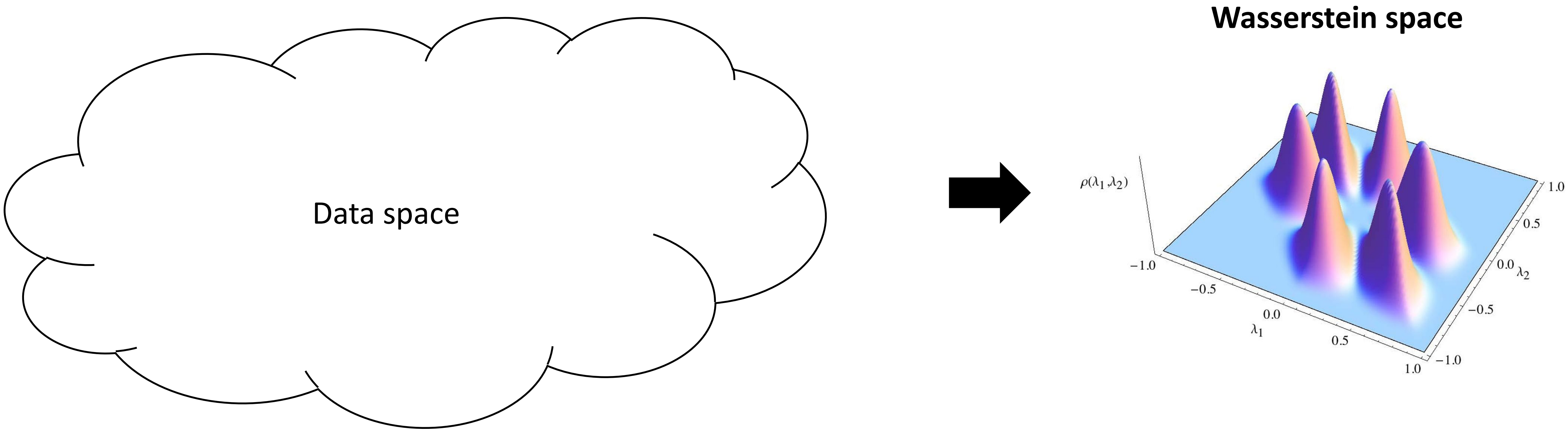
(Nickel & Kiela 2017)

(Chamberlain et al. 2017)

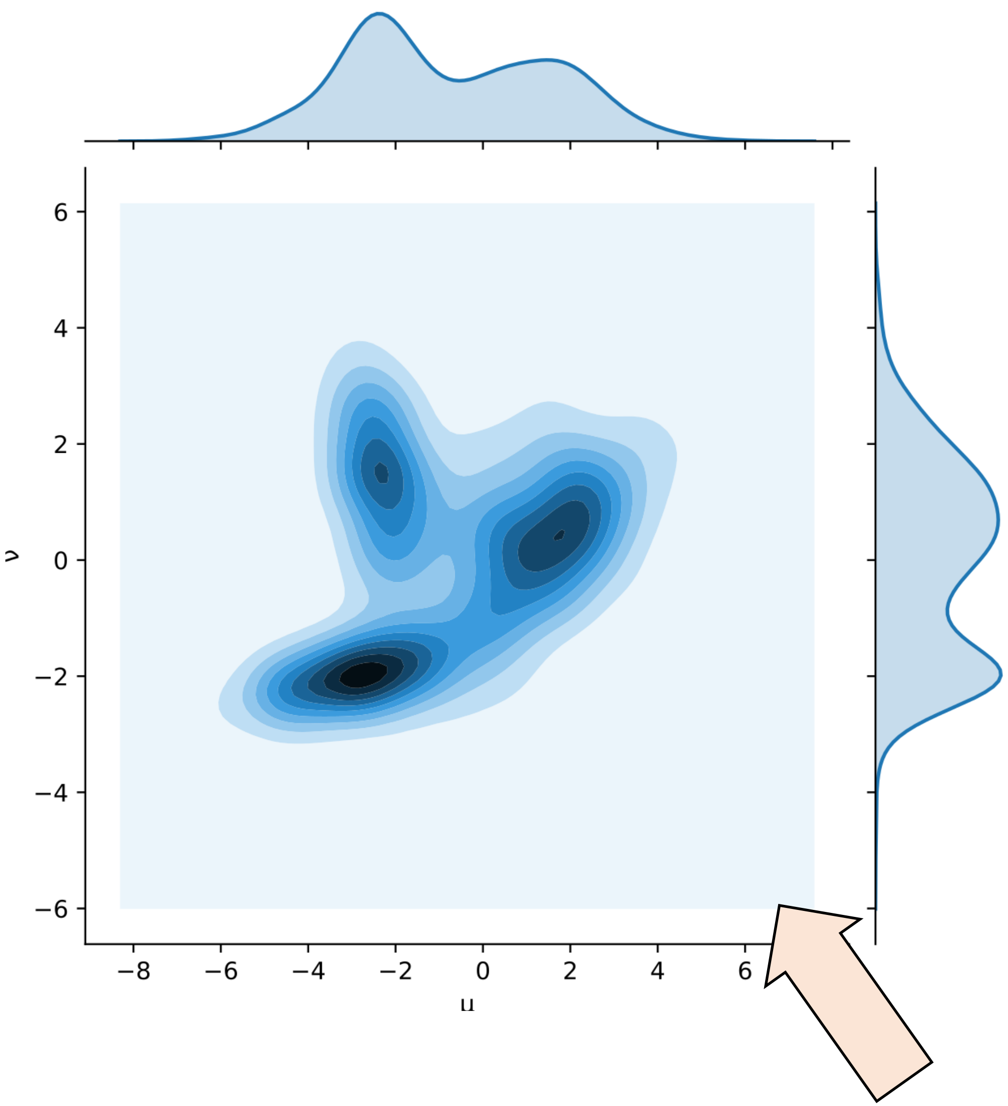
(Nickel & Kiela 2018)

(Sala et al. 2018)

(Le et al. 2019)


Data space
Wasserstein space
$\rho(\lambda_1, \lambda_2)$
λ_1
λ_2
−1.0
−0.5
0.0
0.5
1.0

1st PDF



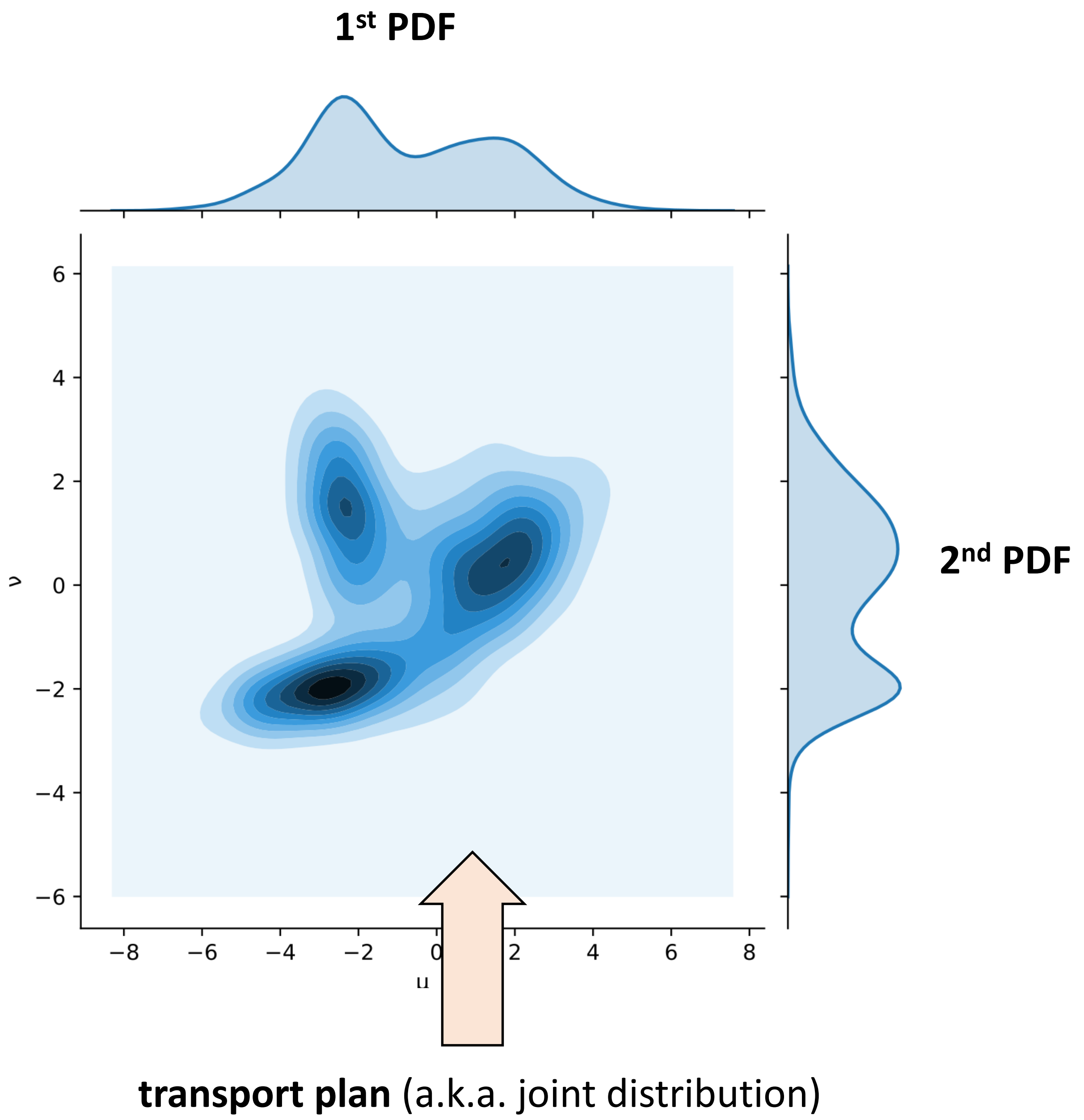
2nd PDF

transport plan (a.k.a. joint distribution)

$$W_p(\mu,\nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \, \mathrm{d}\gamma(x,y) \right)^{1/p}$$

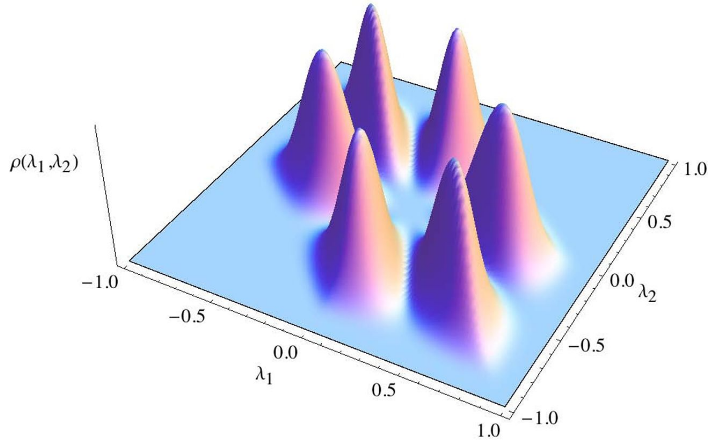
$\Gamma(\mu,\nu)$ joint distributions with marginals μ, ν

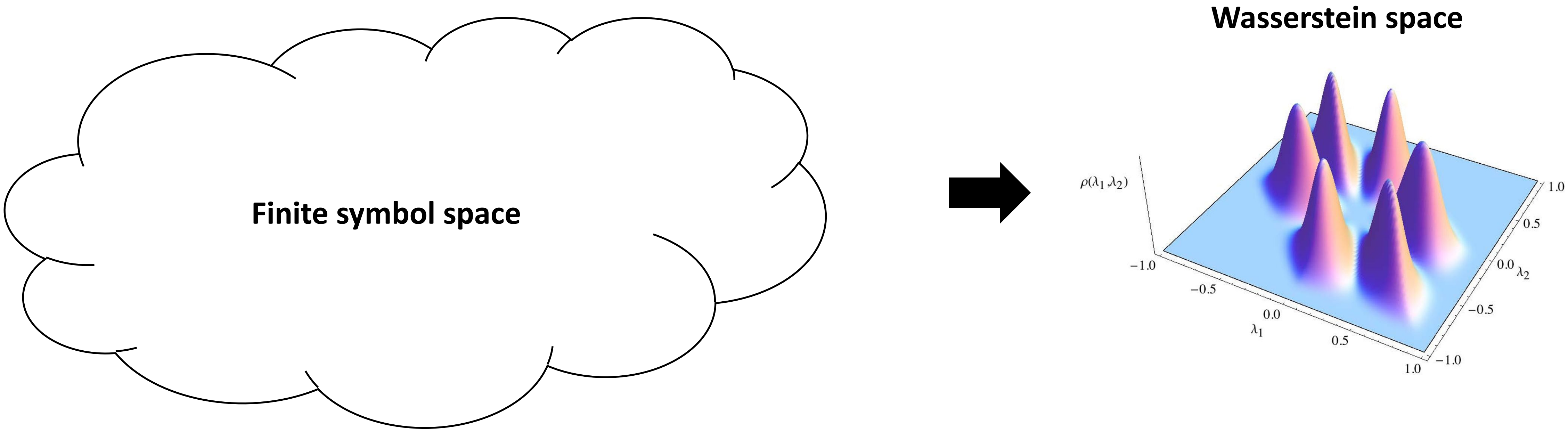
$d(x,y)$ ground metric



X^n with X arbitrary metric space

Wasserstein space




Finite symbol space
Wasserstein space
$\rho(\lambda_1, \lambda_2)$
λ_1
λ_2
−1.0
−0.5
0.0
0.5
1.0

Can we learn useful embeddings into Wasserstein spaces?

given only:

samples $\left\{\left(u^{(i)}, v^{(i)}, r(u^{(i)}, v^{(i)})\right)\right\}$

data space	$\mathcal{C}$
target relationship	$r : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$

learn:

map $\phi : \mathcal{C} \to \mathcal{W}_p(\mathbb{R}^k)$

such that:

(metric learning) $\mathcal{W}_p(\phi(u), \phi(v)) \approx r(u, v)$

(graph embedding) $\mathcal{W}_p(\phi(u), \phi(v))$ small iff graph adjacency

(word embedding) $\mathcal{W}_p(\phi(u), \phi(v))$ predicts semantic similarity

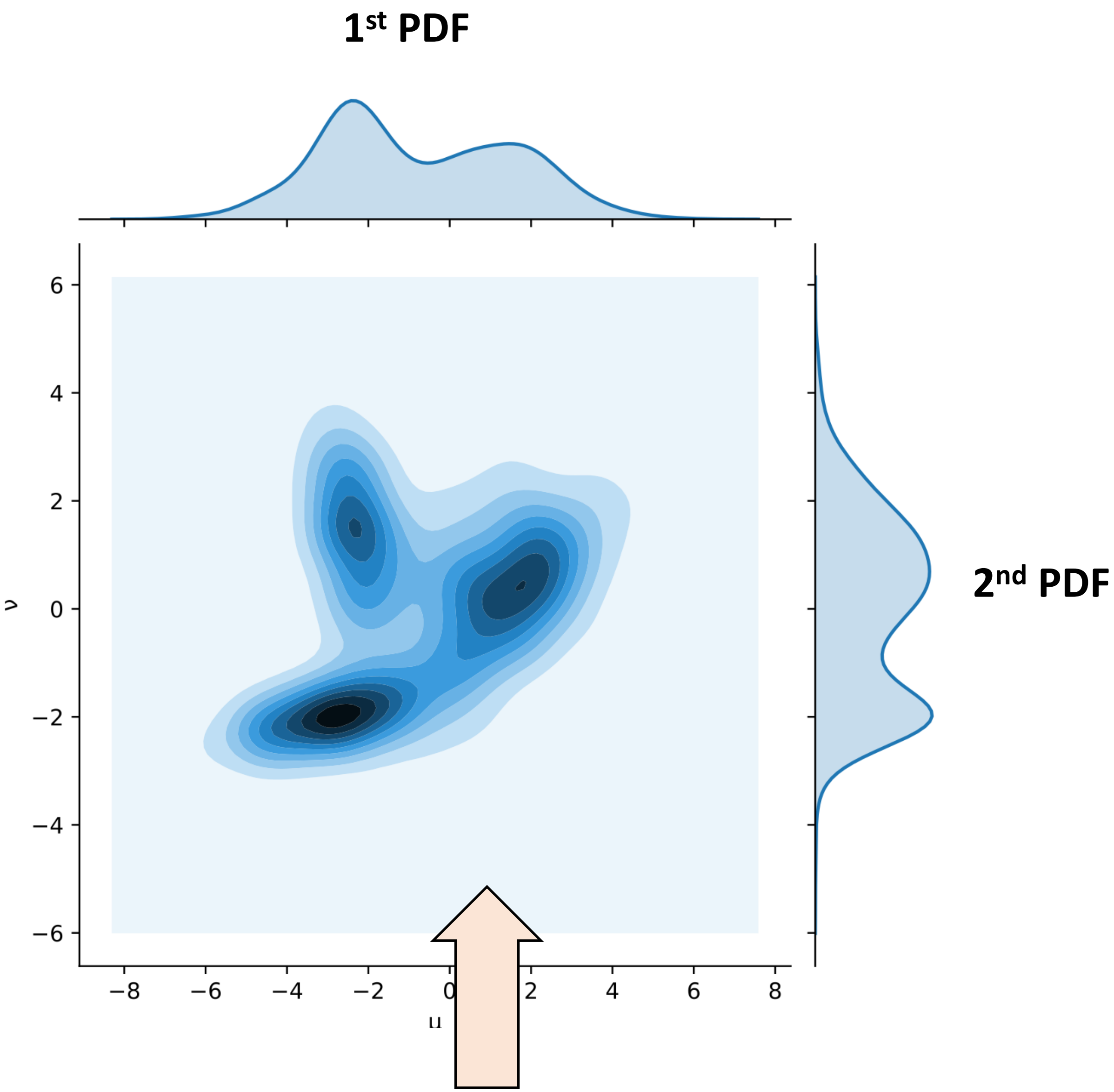
$$\phi : \mathbf{c} \in \mathcal{C} \mapsto \rho \in \mathcal{W}_p(\mathbb{R}^k)$$

$$\boxed{\rho = \frac{1}{M} \sum_{j=1}^{M} \delta_{\mathbf{x}^{(j)}}}$$

$$W_p(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \,\mathrm{d}\gamma(x,y)\right)^{1/p}$$

$\Gamma(\mu,\nu)$ joint distributions with marginals μ, ν

$d(x,y)$ ground metric



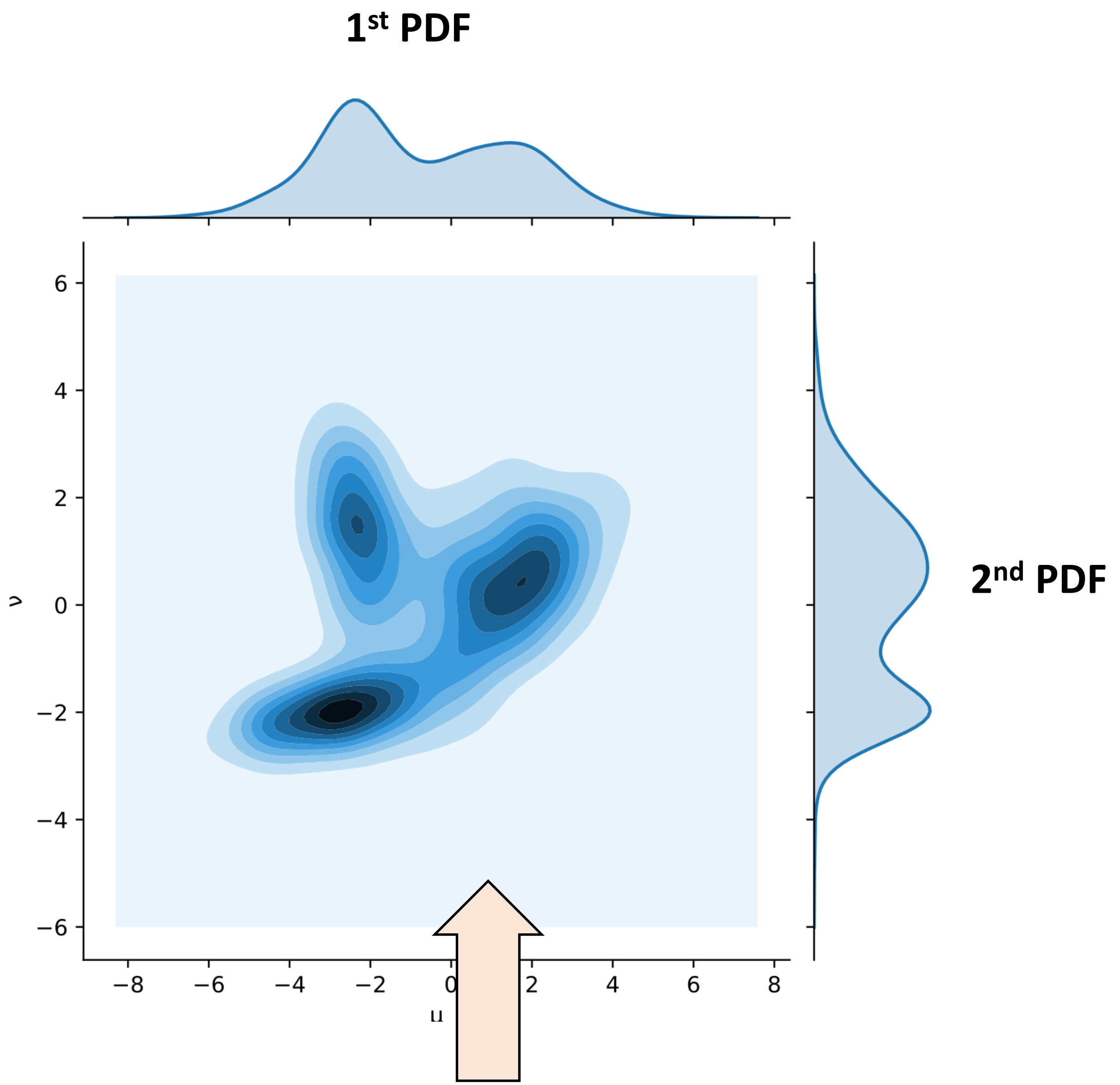
transport plan (a.k.a. joint distribution)

$$\mu = \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathbf{x}^{(i)}}, \quad \nu = \frac{1}{M}\sum_{j=1}^{M}\delta_{\mathbf{y}^{(j)}}$$

$$\mathcal{W}_p(\mu,\nu) = \left(\inf_{\pi\in\Pi(\mathbf{1},\mathbf{1})}\sum_{i,j=1}^{M} d(\mathbf{x}^{(i)},\mathbf{y}^{(j)})^p \pi_{ij}\right)^{1/p}$$

$\Pi(\mathbf{1},\mathbf{1})$ doubly-stochastic matrices of size M x M

$d(x,y)$ ground metric



transport plan (a.k.a. joint distribution)

$\mu = \frac{1}{M} \sum_{i=1}^{M} \delta_{\mathbf{x}^{(i)}}, \quad \nu = \frac{1}{M} \sum_{j=1}^{M} \delta_{\mathbf{y}^{(j)}}$

entropic regularizer

$$\mathcal{W}_p^\lambda(\mu, \nu) = \left(\sum_{i,j=1}^{M} d(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})^p \pi_{ij}^\lambda \right)^{1/p}$$

$$\pi^\lambda = \inf_{\pi \in \Pi(\mathbf{1}, \mathbf{1})} \sum_{i,j=1}^{M} d(\mathbf{x}^{(i)}, \mathbf{y}^{(j)})^p \pi_{ij} + \lambda \sum_{i,j=1}^{M} \pi_{ij} \log(\pi_{ij})$$

$\Pi(\mathbf{1}, \mathbf{1})$ doubly-stochastic matrices of size M x M

$d(x, y)$ ground metric

Marco Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport.* NIPS (2013).

$\mu = \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathbf{x}^{(i)}}, \quad \nu = \frac{1}{M}\sum_{j=1}^{M}\delta_{\mathbf{y}^{(j)}}$

entropic regularizer

$$\mathcal{W}_p^{\lambda}(\mu,\nu) = \left(\sum_{i,j=1}^{M} d(\mathbf{x}^{(i)},\mathbf{y}^{(j)})^p \pi_{ij}^{\lambda}\right)^{1/p}$$

$$\pi^{\lambda} = \inf_{\pi\in\Pi(\mathbf{1},\mathbf{1})} \sum_{i,j=1}^{M} d(\mathbf{x}^{(i)},\mathbf{y}^{(j)})^p \pi_{ij} + \lambda \sum_{i,j=1}^{M} \pi_{ij}\log(\pi_{ij})$$

$\Pi(\mathbf{1},\mathbf{1})$ doubly-stochastic matrices of size M x M

$d(x,y)$ ground metric

Sinkhorn iteration

$$\pi^{\lambda} = \mathsf{diag}(\alpha)\mathbf{K}\mathsf{diag}(\beta)$$

$$\alpha = \mathbf{1} \oslash \mathbf{K}\beta$$

$$\beta = \mathbf{1} \oslash \mathbf{K}^{\mathsf{T}}\alpha$$

Marco Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport.* NIPS (2013).

$\mu = \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathbf{x}^{(i)}}, \quad \nu = \frac{1}{M}\sum_{j=1}^{M}\delta_{\mathbf{y}^{(j)}}$

entropic regularizer

$$\mathcal{W}_p^{\lambda}(\mu,\nu) = \left(\sum_{i,j=1}^{M} d(\mathbf{x}^{(i)},\mathbf{y}^{(j)})^p \pi_{ij}^{\lambda}\right)^{1/p}$$

$$\pi^{\lambda} = \inf_{\pi\in\Pi(\mathbf{1},\mathbf{1})} \sum_{i,j=1}^{M} d(\mathbf{x}^{(i)},\mathbf{y}^{(j)})^p \pi_{ij} + \lambda \sum_{i,j=1}^{M} \pi_{ij}\log(\pi_{ij})$$

$\Pi(\mathbf{1},\mathbf{1})$ doubly-stochastic matrices of size M x M

$d(x,y)$ ground metric

Sinkhorn iteration

$$\pi^{\lambda} = \mathrm{diag}(\alpha)\mathbf{K}\mathrm{diag}(\beta)$$

$$\alpha = \mathbf{1} \oslash \mathbf{K}\beta$$

$$\beta = \mathbf{1} \oslash \mathbf{K}^{\mathsf{T}}\alpha$$

unroll and differentiate *(Genevay 2018)*

Marco Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport.* NIPS (2013).

Can we learn useful embeddings into Wasserstein spaces?

given only:

samples $\left\{\left(u^{(i)}, v^{(i)}, r(u^{(i)}, v^{(i)})\right)\right\}$

data space $\mathcal{C}$

target relationship $r : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$

learn:

map $\phi : \mathcal{C} \to \mathcal{W}_p^\lambda(\mathbb{R}^k)$

$$\phi_* = \arg\min_{\phi \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(\mathcal{W}_p^\lambda\left(\phi(u^{(i)}), \phi(v^{(i)})\right), r^{(i)}\right)$$

such that:

(metric learning) $\mathcal{W}_p^\lambda(\phi(u), \phi(v)) \approx r(u, v)$

(graph embedding) $\mathcal{W}_p^\lambda(\phi(u), \phi(v))$ small iff graph adjacency

(word embedding) $\mathcal{W}_p^\lambda(\phi(u), \phi(v))$ predicts semantic similarity

Representational capacity

Generate **random network**. ($\mathcal{C}$ = nodes.)

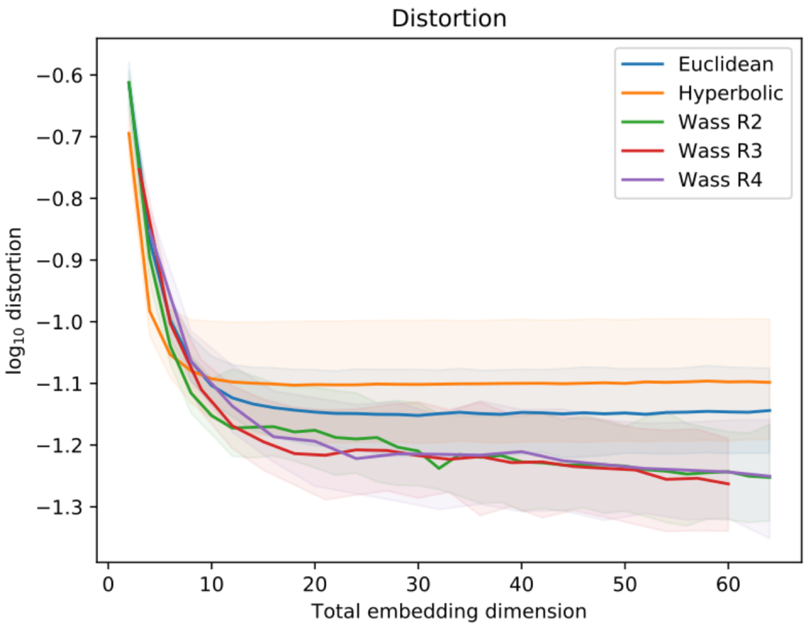
Compute **input metric** = shortest path distance.

Learn **embedding** $\phi : \mathcal{C} \to \mathcal{W}_p^\lambda(\mathbb{R}^k)$ such that $\mathcal{W}_p^\lambda(\phi(u), \phi(v))$ matches input metric.

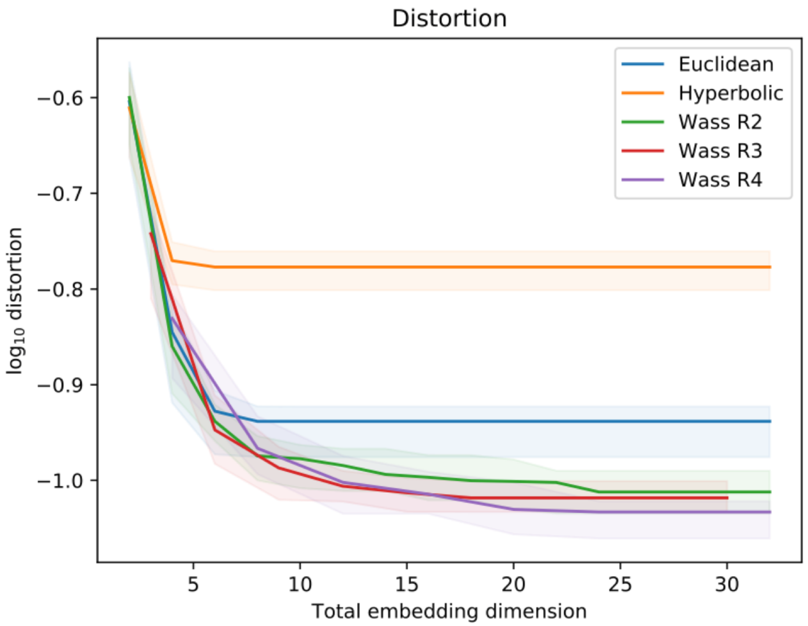
Minimize **distortion**:

$$\phi_* = \arg\min_{\phi} \frac{1}{\binom{n}{2}} \sum_{j>i} \frac{|\mathcal{W}_1^\lambda(\phi(v_i), \phi(v_j)) - d_{\mathcal{C}}(v_i, v_j)|}{d_{\mathcal{C}}(v_i, v_j)}$$

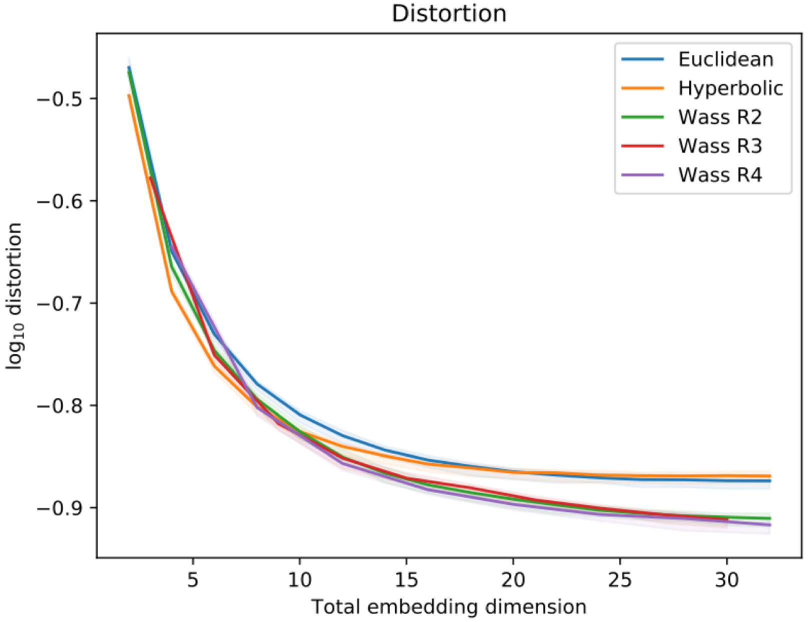
Compare with **Euclidean** and **hyperbolic** *(Nickel & Kiela 2017)* embeddings.



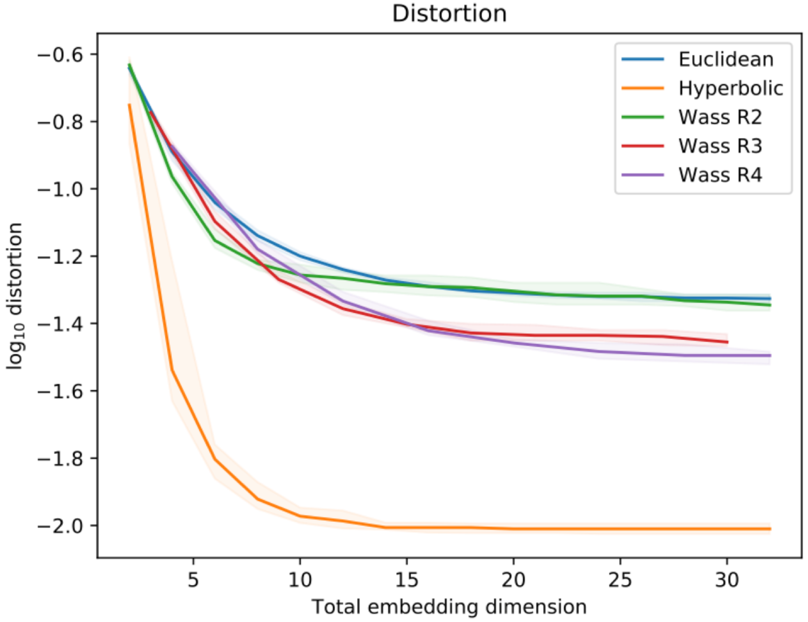
(a) Random scale-free networks.



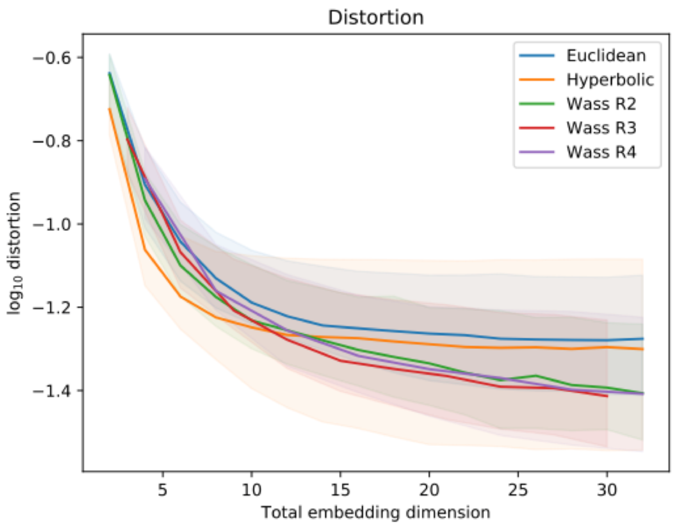
(b) Random small-world networks.



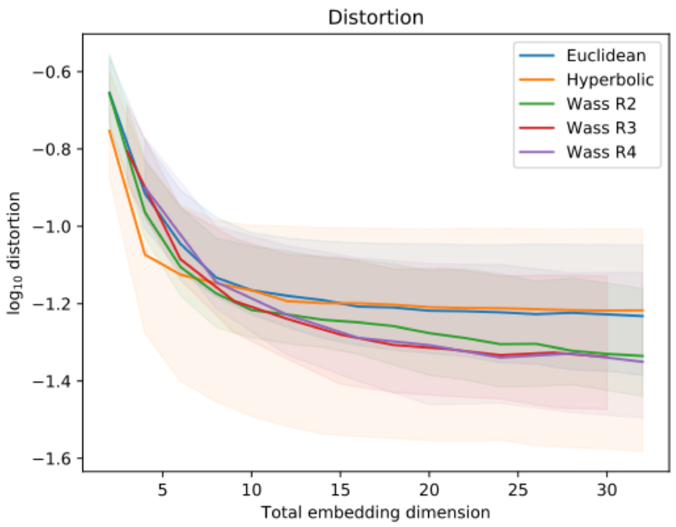
(c) Random community-structured networks.



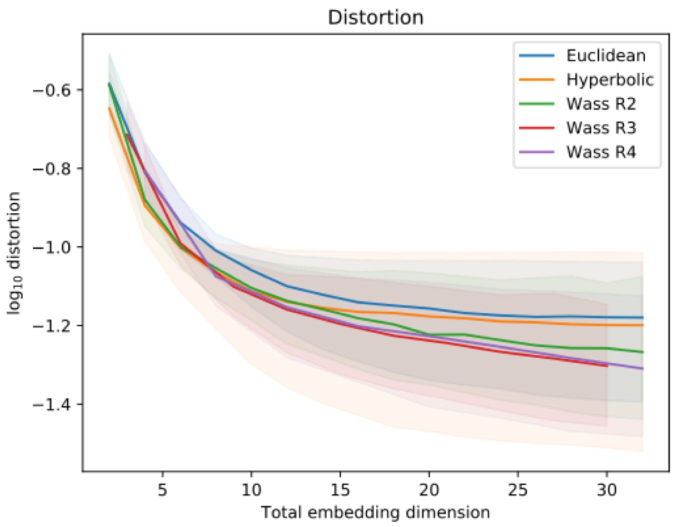
(d) Random trees.



(a) arXiv co-authorship.



(b) Amazon product co-purchases.



(c) Google web graph.

Word embedding

Define a **vocabulary** of many words. ($\mathcal{C}$ = words.)

Given a **corpus** (many sentences).

Define a **positive example =** pair of words co-occurring in a sentence

For each positive example, define a **negative example** = same first word, second word not in sentence.

Learn **embedding** $\phi : \mathcal{C} \to \mathcal{W}_p^\lambda(\mathbb{R}^k)$ s.t. positive pairs are close, negative pairs are far.

Minimize **contrastive divergence:**

$$\phi_* = \arg\min_{\phi} \sum_{\mathbf{x}_i, \mathbf{x}_j} r_{\mathbf{x}_i, \mathbf{x}_j} \Big(\mathcal{W}_1^\lambda(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \Big)^2 + (1 - r_{\mathbf{x}_i, \mathbf{x}_j}) \Big(\Big[m - \mathcal{W}_1^\lambda(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) \Big]_+ \Big)^2$$

$\mathcal{W}_1^\lambda(\mathbb{R}^2)$	one:	f, two, i, after, four
	united:	series, professional, team, east, central
	algebra:	skin, specified, equation, hilbert, reducing
$\mathcal{W}_1^\lambda(\mathbb{R}^3)$	one:	two, three, s, four, after
	united:	kingdom, australia, official, justice, officially
	algebra:	binary, distributions, reviews, ear, combination
$\mathcal{W}_1^\lambda(\mathbb{R}^4)$	one:	six, eight, zero, two, three
	united:	army, union, era, treaty, federal
	algebra:	tables, transform, equations, infinite, differential

Task Name	# Pairs	# Found	$\mathcal{W}_1^\lambda(\mathbb{R}^2)$ 17M	$\mathcal{W}_1^\lambda(\mathbb{R}^3)$ 17M	$\mathcal{W}_1^\lambda(\mathbb{R}^4)$ 17M	R —	M 63M	S 631M	G 900M	W 100B
RG-65	65	64	0.18	0.56	0.69	0.27	-0.02	0.50	0.66	0.54
Verb-143	143	144	0.12	0.14	0.29	0.29	0.06	0.36	0.44	0.27
WS-353	353	351	0.14	0.22	0.37	0.24	0.10	0.49	0.62	0.64
WS-353-S	203	201	0.19	0.35	0.47	0.36	0.15	0.61	0.70	0.70
WS-353-R	252	252	0.05	0.12	0.24	0.18	0.09	0.40	0.56	0.61
MC-30	30	30	-0.04	0.43	0.48	0.47	-0.14	0.57	0.66	0.63
Rare-Word	2034	1159	0.08	0.27	0.11	0.29	0.11	0.39	0.06	0.39
MEN	3000	2915	0.20	0.26	0.31	0.24	0.09	0.57	0.31	0.65
MTurk-287	287	284	0.30	0.30	0.43	0.33	0.09	0.59	0.36	0.67
MTurk-771	771	770	0.10	0.24	0.27	0.26	0.10	0.50	0.32	0.57
SimLex-999	999	998	0.06	0.09	0.13	0.23	0.01	0.27	0.10	0.31

R: RNN (80D) *(Kombrink et al. 2011)*
M: Metaoptimize (50D) *(Turian et al. 2010)*
S: SENNA (50D) *(Collobert 2011)*
G: Global Context (50D) *(Huang et al. 2012)*
W: word2vec (80D) *(Mikolov 2013)*

Baseline tasks from *(Faruqui & Dyer 2014)*.

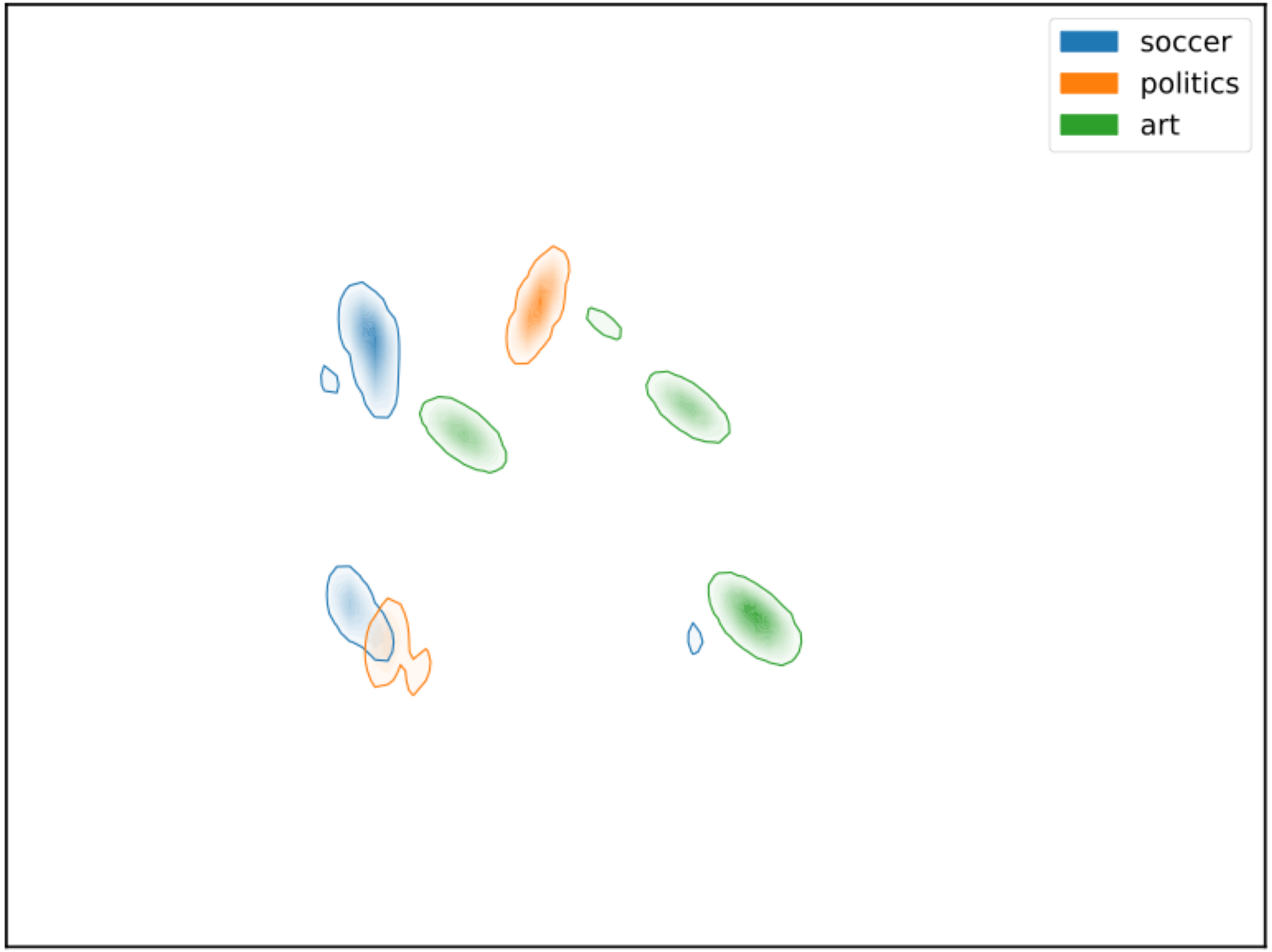
Direct visualization

Learn **word embedding**.

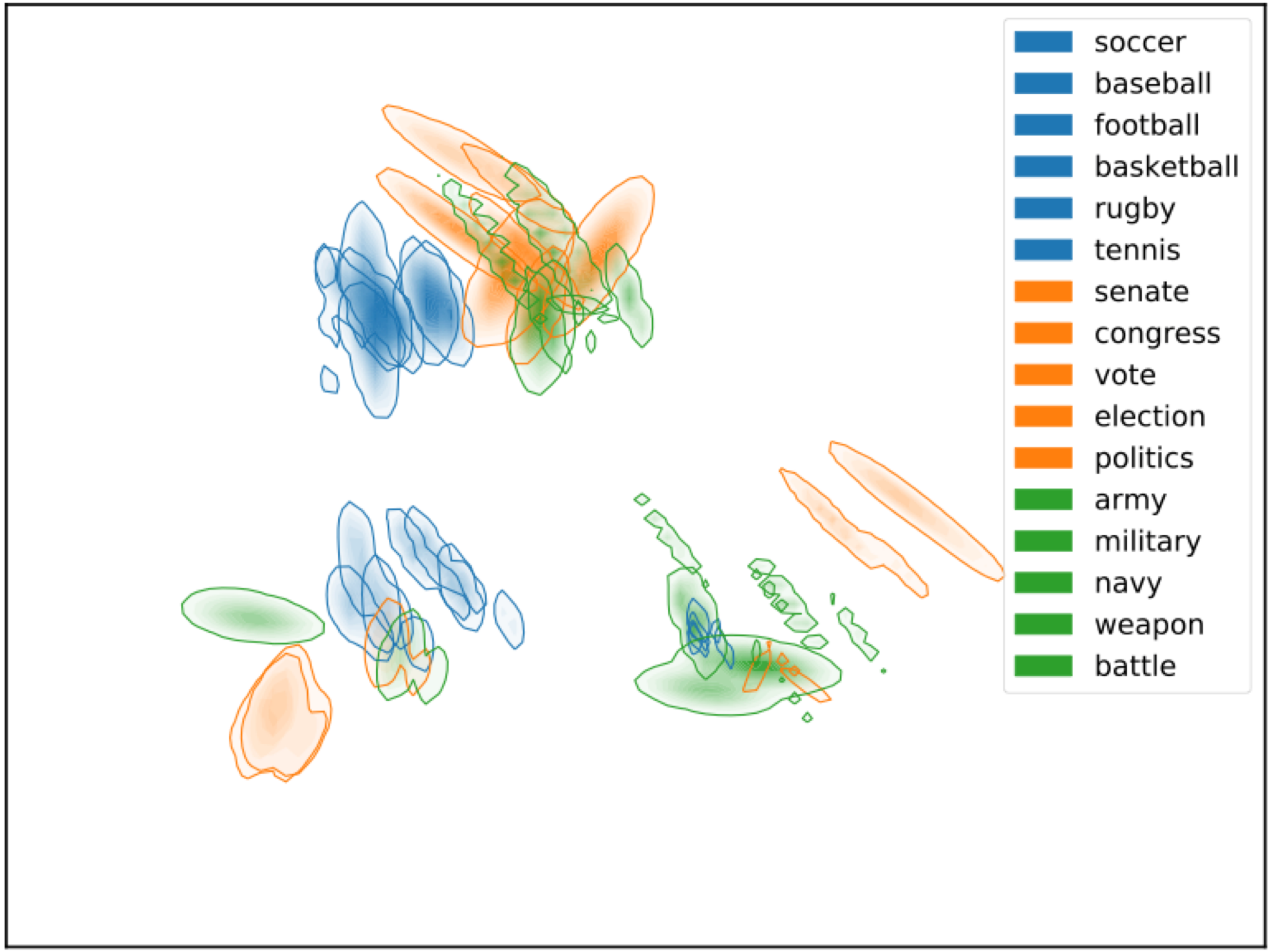
For a single word: apply **KDE** to point cloud.

Threshold the density estimate.

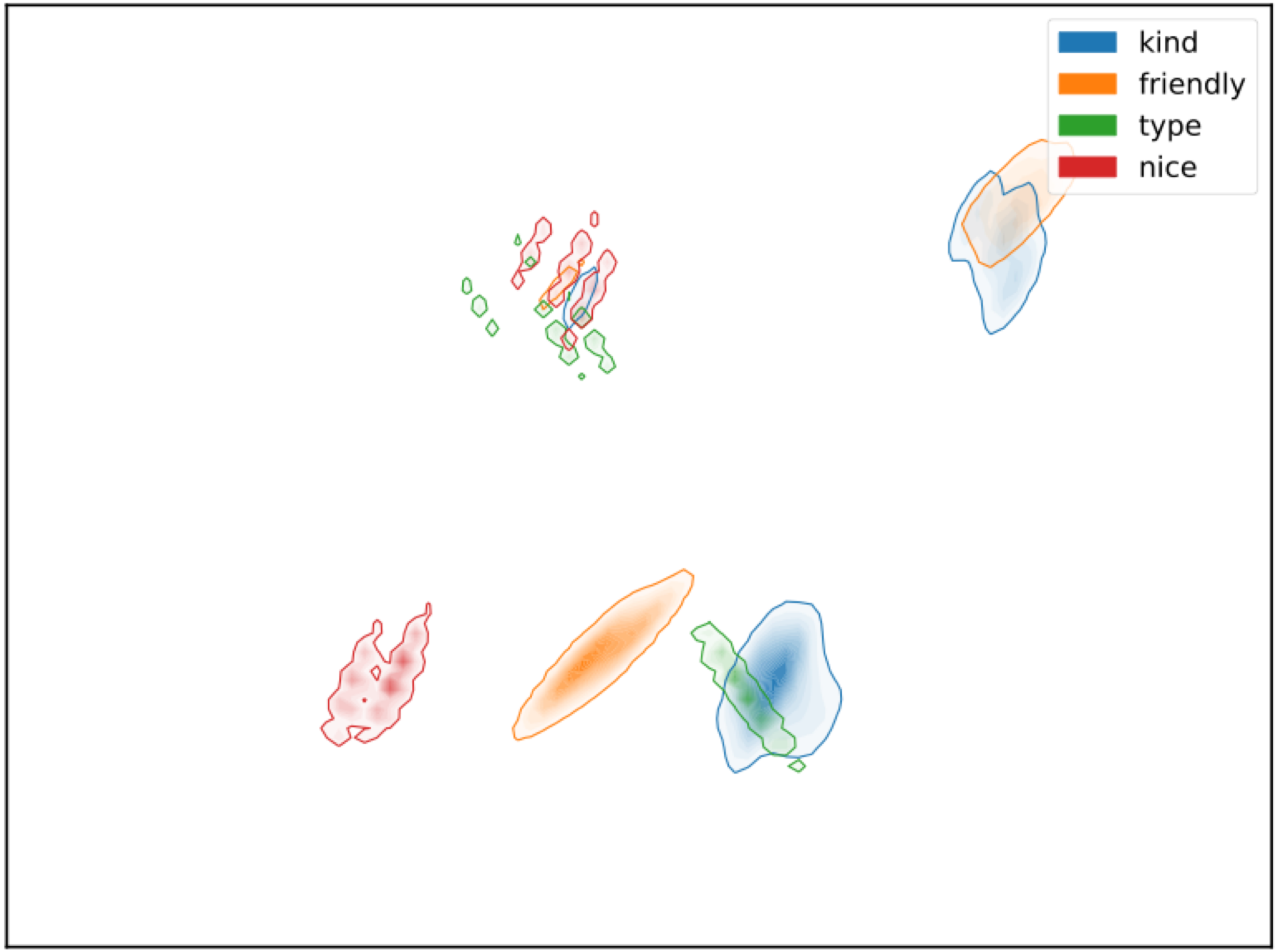
Show both **upper level set** and **density**.



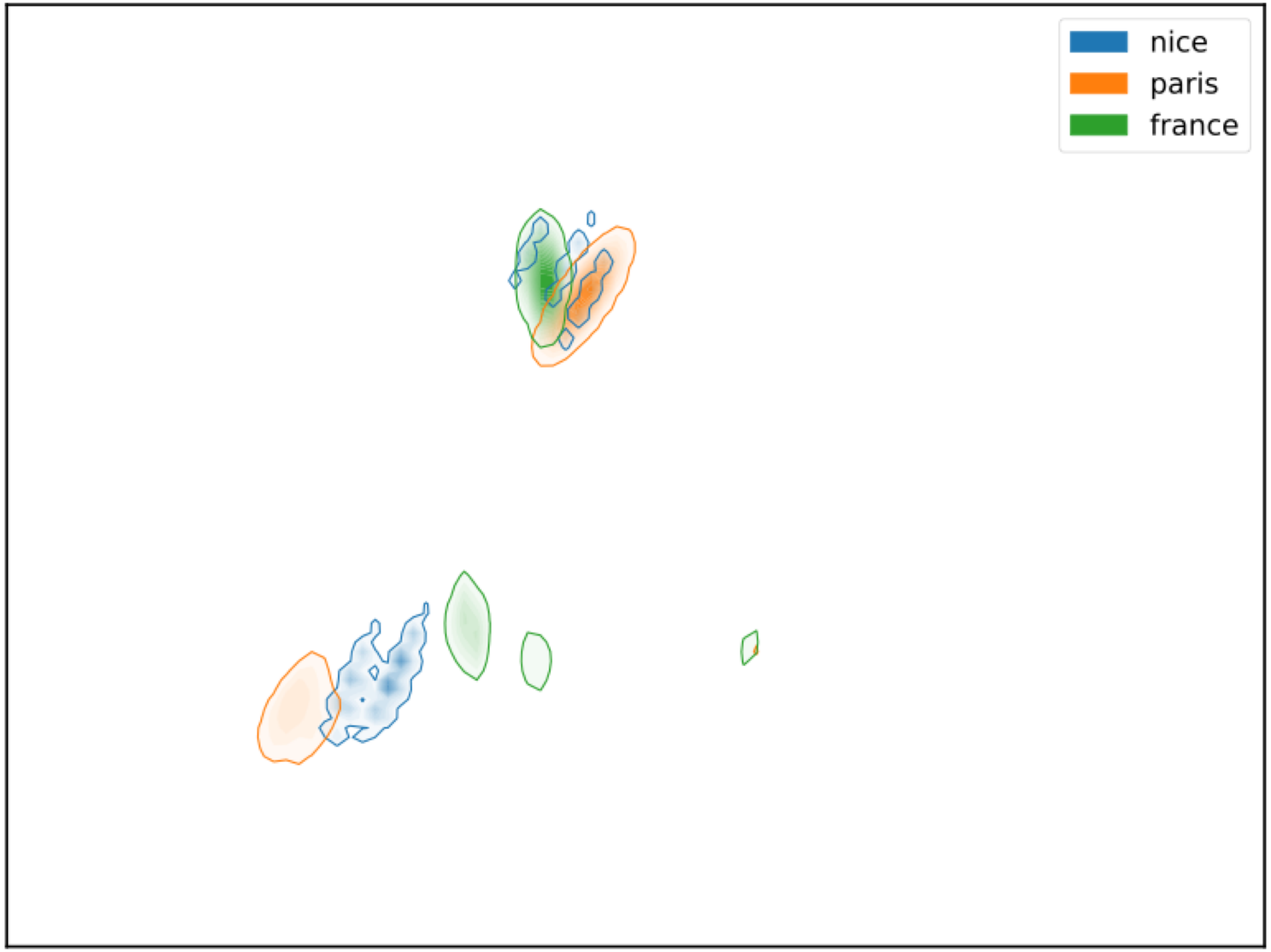
(a) Densities of three embedded words.



(b) Class separation.



(c) Word with multiple meanings: `kind`.



(d) Explaining a failed association: `nice`.

Learning Entropic Wasserstein embeddings

Wasserstein spaces can embed a wide variety of metrics.

Can **learn embeddings** into (entropic) Wasserstein spaces.

Learned embeddings of **complex networks** can achieve **lower distortion** than Euclidean.

Learned **word embeddings** comparable to existing work in replicating human similarity judgments.

Can **directly visualize** the embedding (unlike most methods).