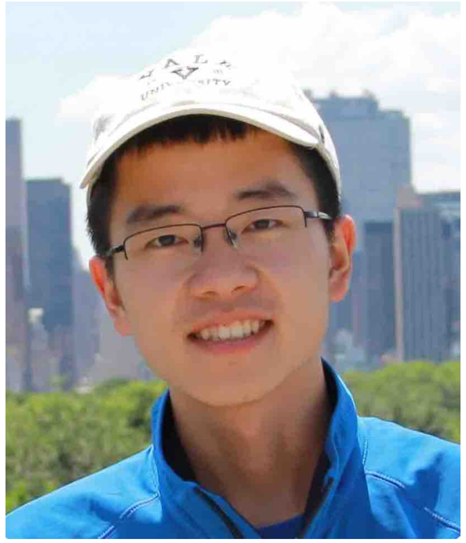
Learning about Language with Normalizing Flows

Graham Neubig

Language Technologies Institute, Carnegie Mellon University

Chunting Zhou   Junxian He



Di Wang, Xuezhe Ma, Daniel Spokoyny, Taylor Berg-Kirkpatrick

Language Technologies Institute

Learning about Language?

Learning about Language?

- **Syntactic structure**

Learning about Language?

- **Syntactic structure**

The cat sat on a green wall

Learning about Language?

- **Syntactic structure**

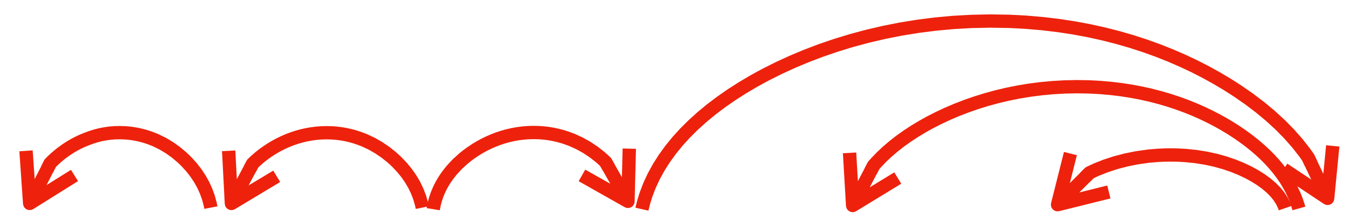
	The	cat	sat	on	a	green	wall
Parts-of-speech:	DT	NN	VBD	IN	DT	JJ	NN

Learning about Language?

- **Syntactic structure**

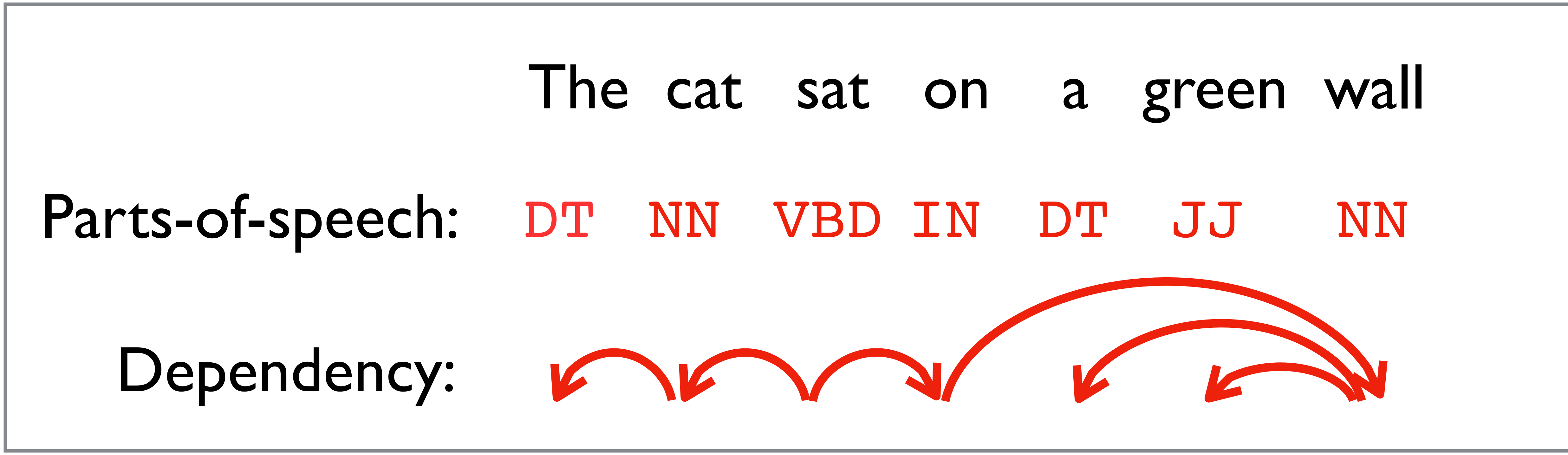
	The	cat	sat	on	a	green	wall
Parts-of-speech:	DT	NN	VBD	IN	DT	JJ	NN

Dependency:



Learning about Language?

- **Syntactic structure**



- **Cross-lingual correspondences**

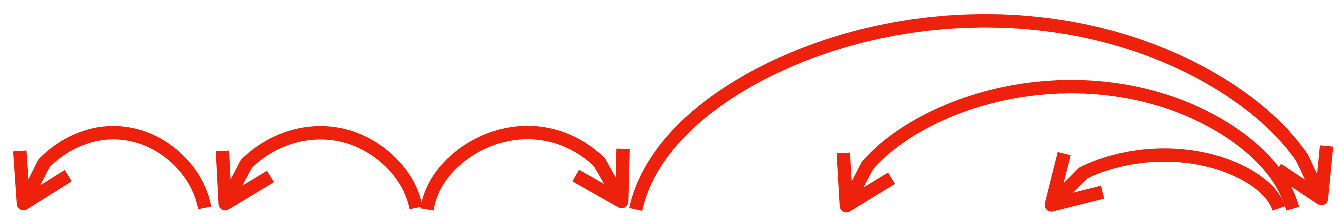
Learning about Language?

- **Syntactic structure**

The cat sat on a green wall

Parts-of-speech: DT NN VBD IN DT JJ NN

Dependency:



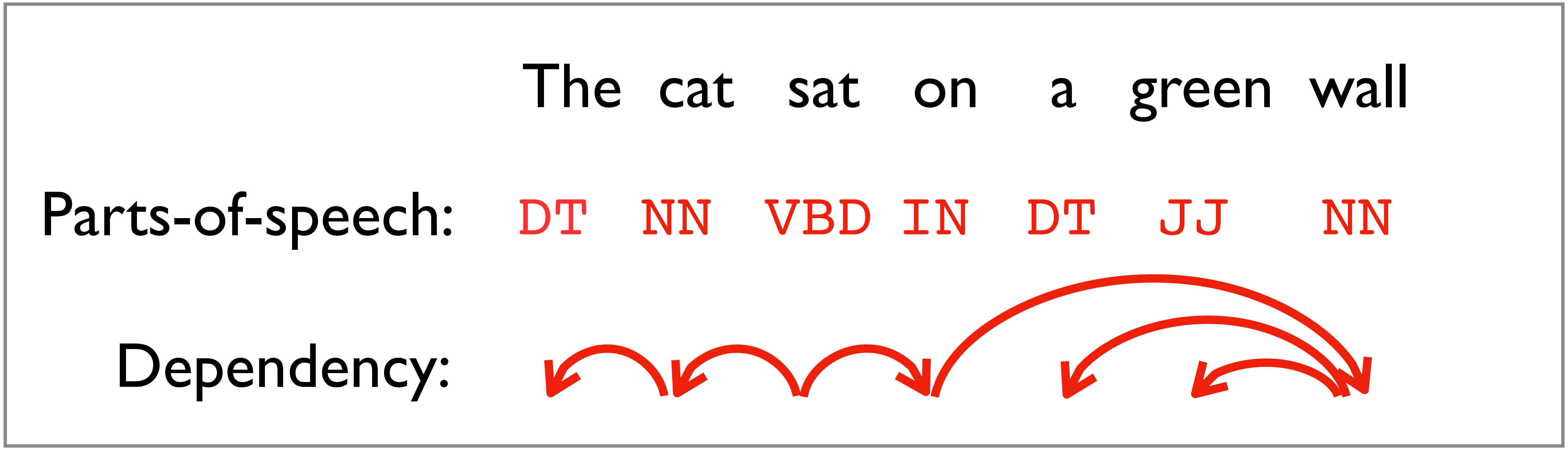
- **Cross-lingual correspondences**

a cat green on sat the wall

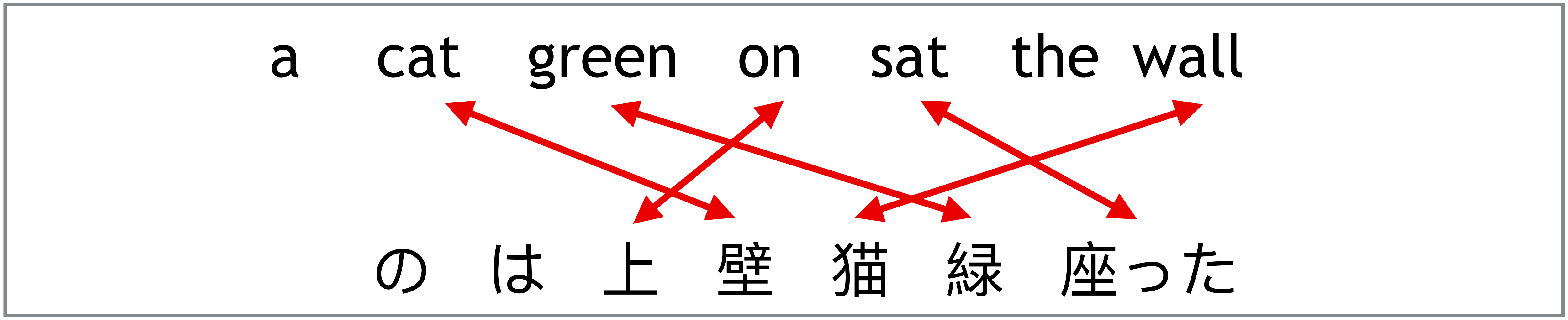
の は 上 壁 猫 緑 座った

Learning about Language?

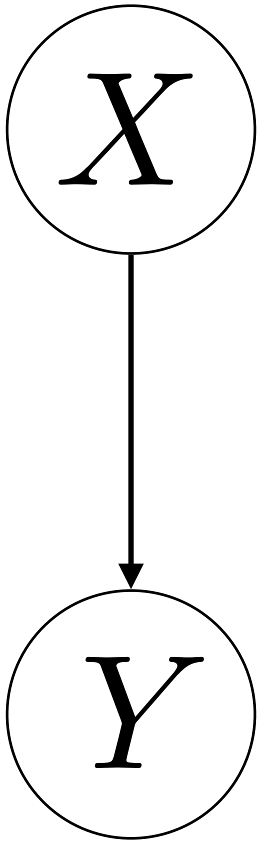
- **Syntactic structure**



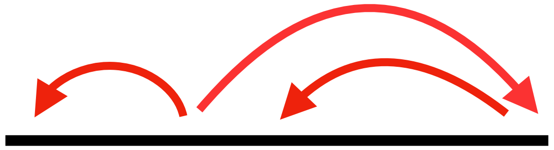
- **Cross-lingual correspondences**



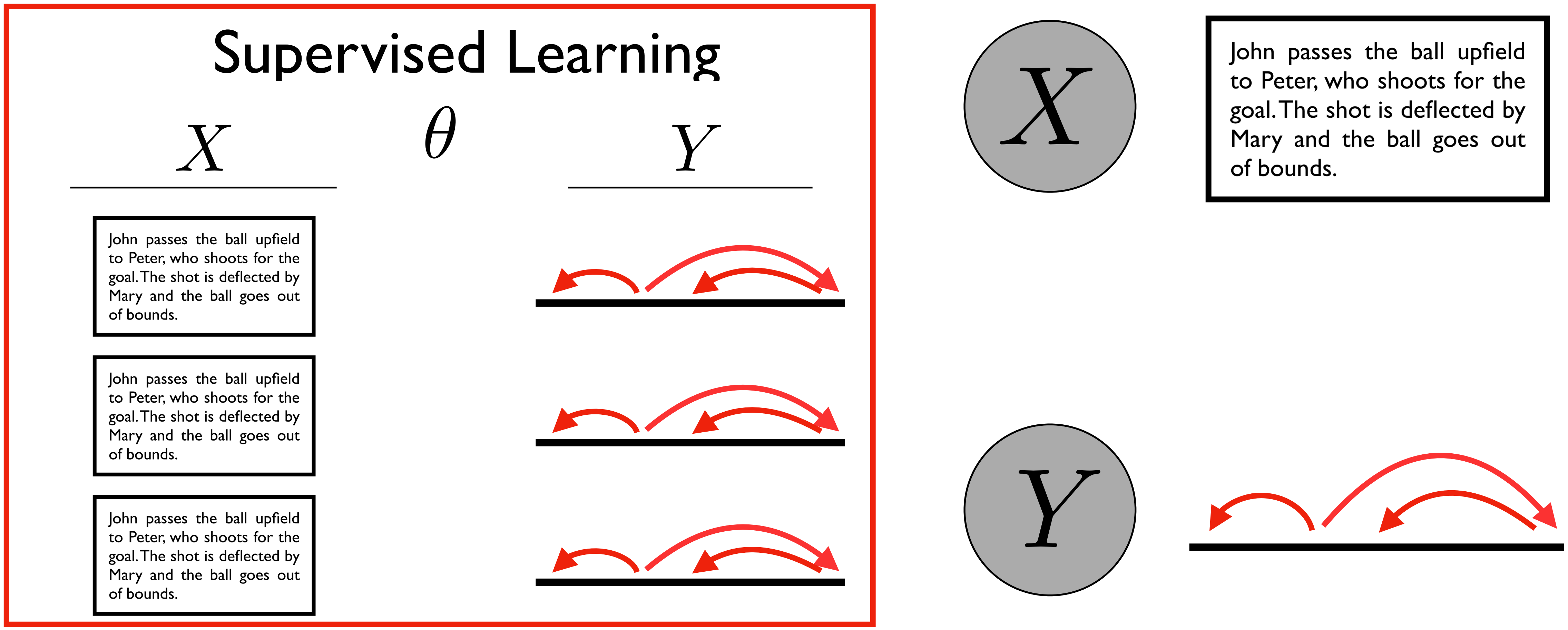
Supervised Approaches



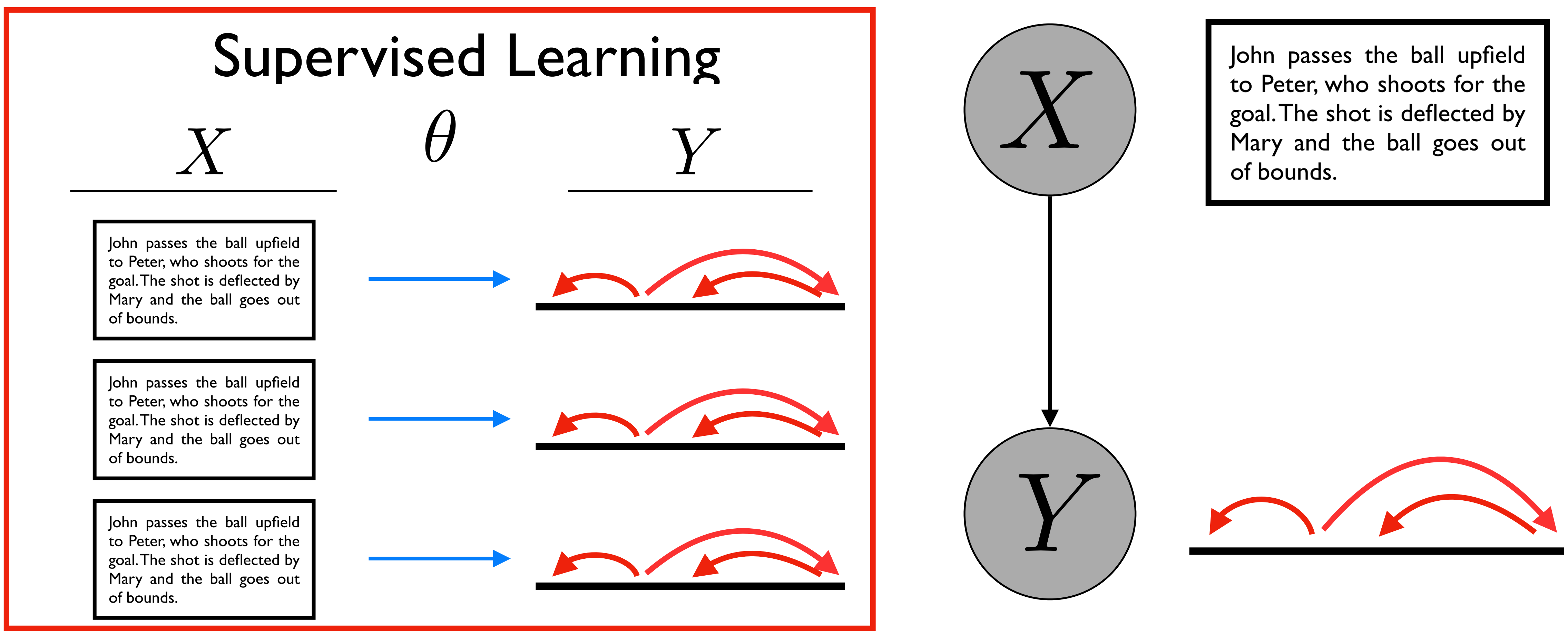
John passes the ball upfield to Peter, who shoots for the goal. The shot is deflected by Mary and the ball goes out of bounds.



Supervised Approaches

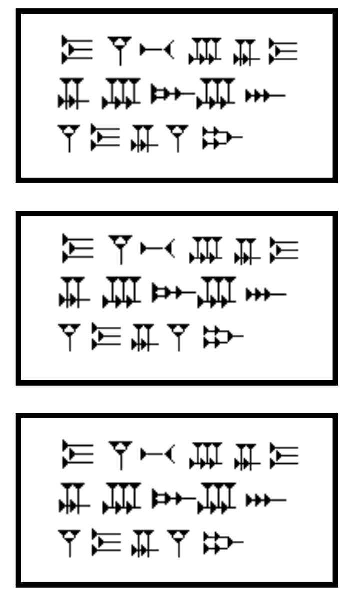


Supervised Approaches

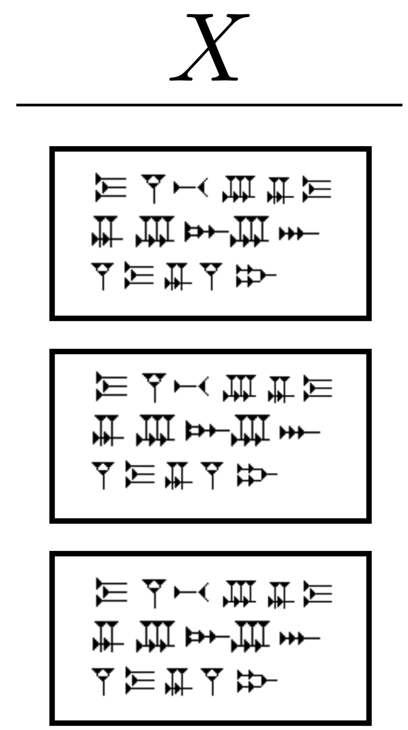


Unsupervised Approaches

X

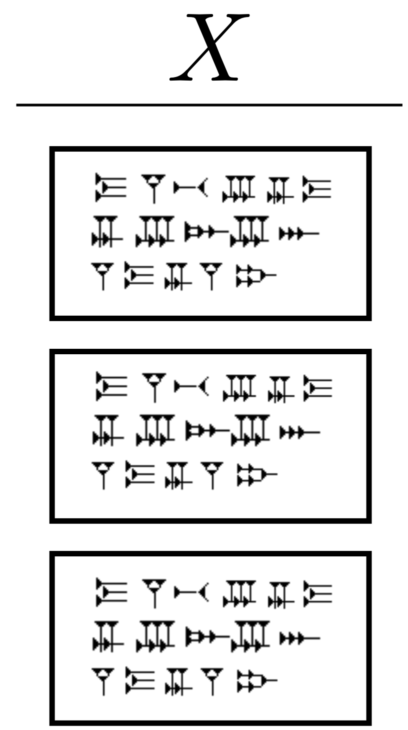


Unsupervised Approaches



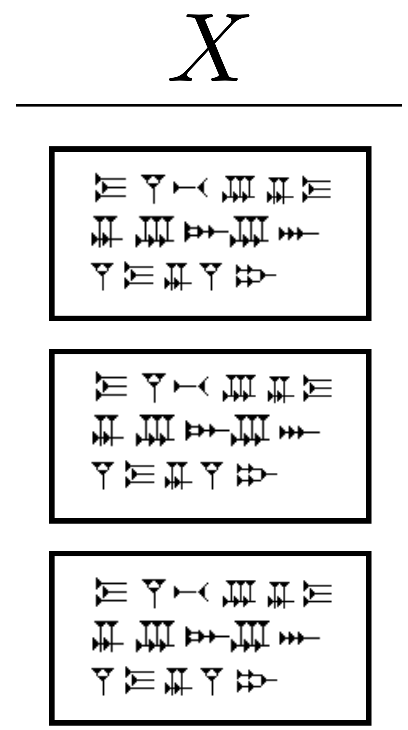
- Learning language models P(X)

Unsupervised Approaches



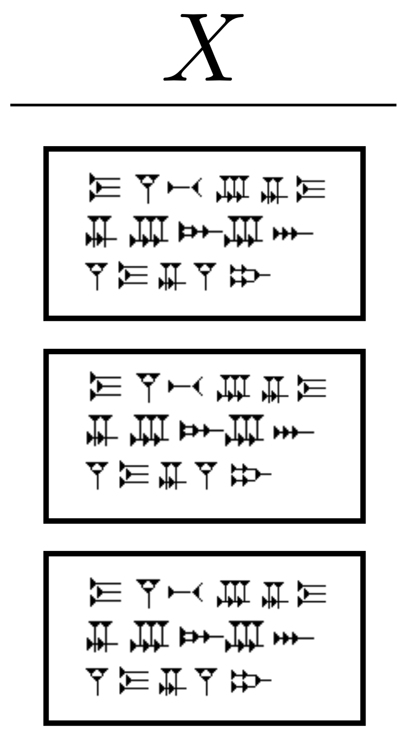
- Learning language models P(X)
- Learning continuous features from language models (e.g. word2vec, skipthought, BERT)

Unsupervised Approaches



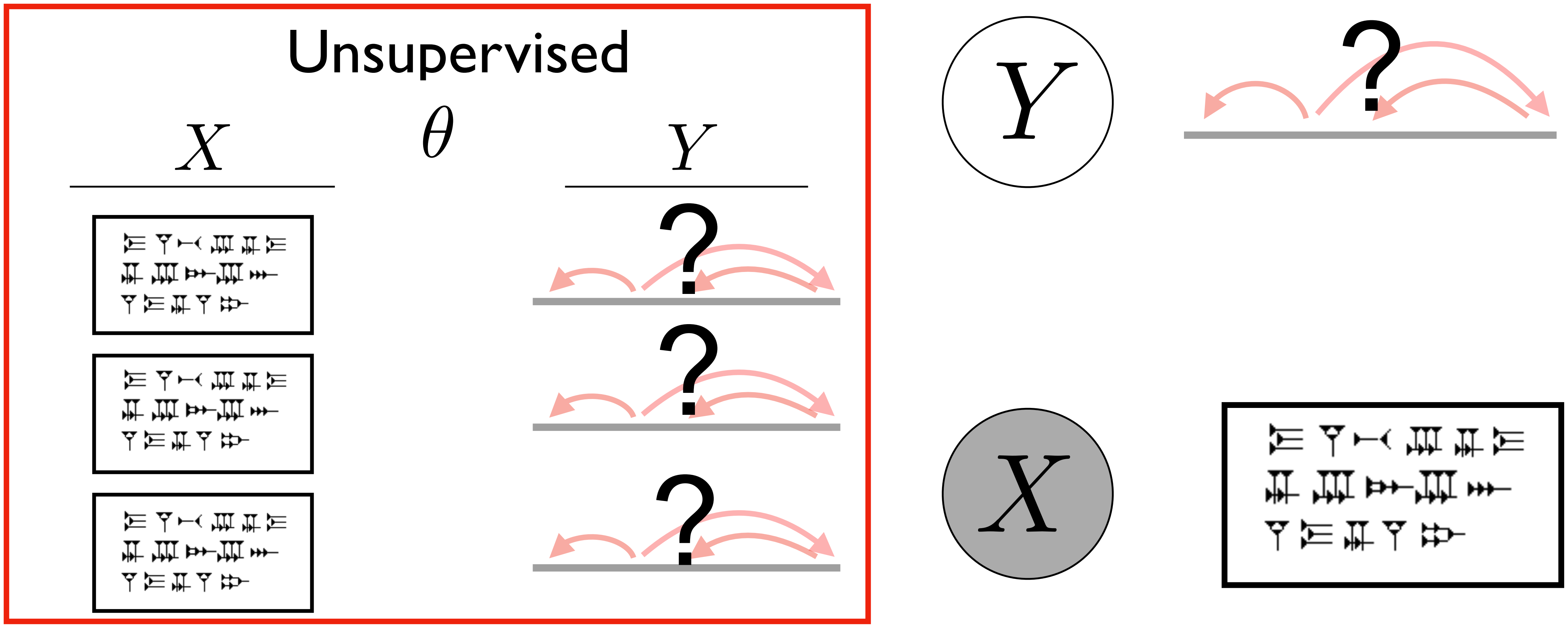
- Learning language models P(X)
- Learning continuous features from language models (e.g. word2vec, skipthought, BERT)
- But how do we turn this into **interpretable structure**?

Unsupervised Approaches

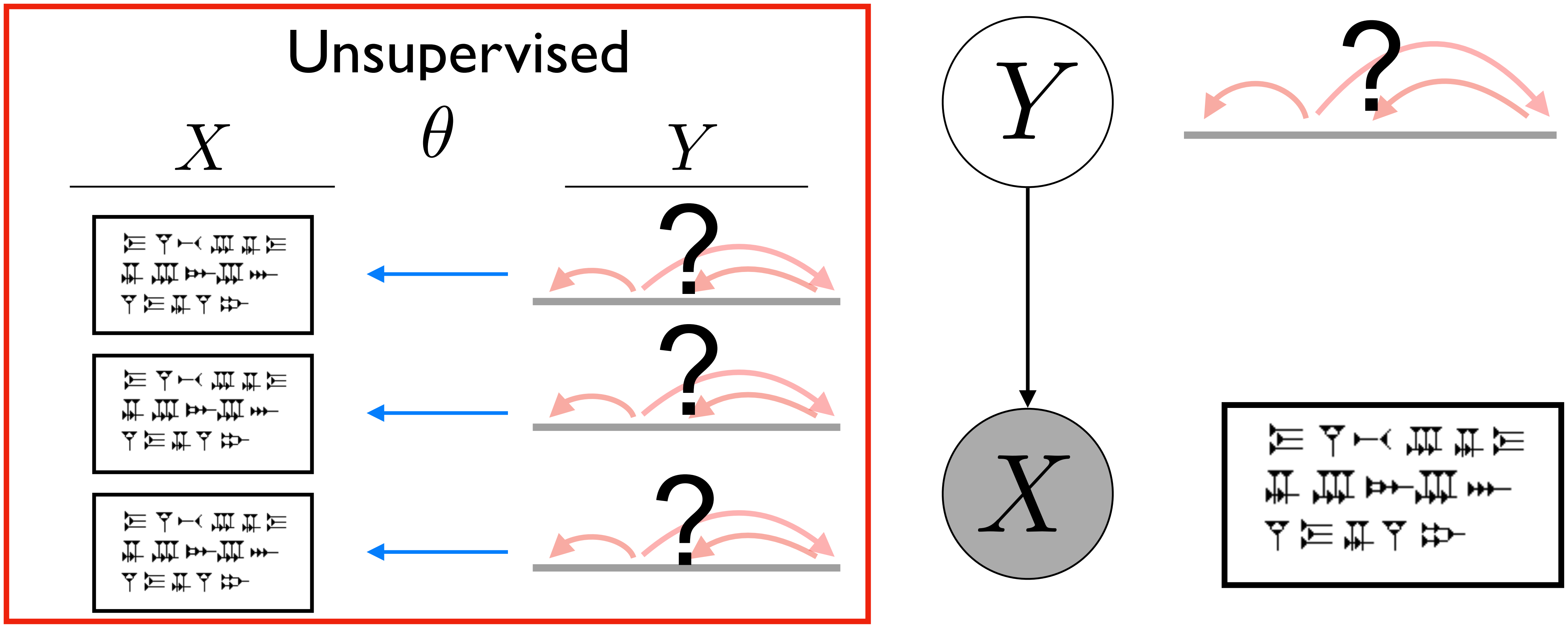


- Learning language models P(X)
- Learning continuous features from language models (e.g. word2vec, skipthought, BERT)
- But how do we turn this into **interpretable structure**?
- How do we do it while **taking advantage of continuous features**?

Latent Variable Approaches



Latent Variable Approaches



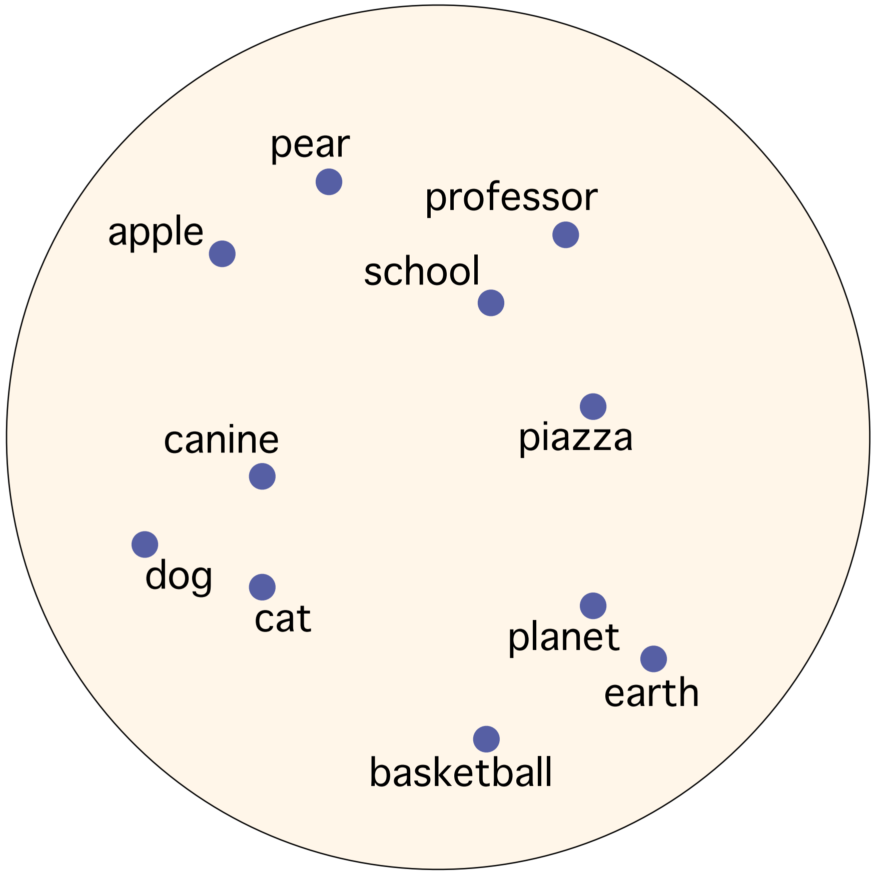
Density Matching for Bilingual Word Embedding

Chunting Zhou, Xuezhe Ma, Di Wang, Graham Neubig

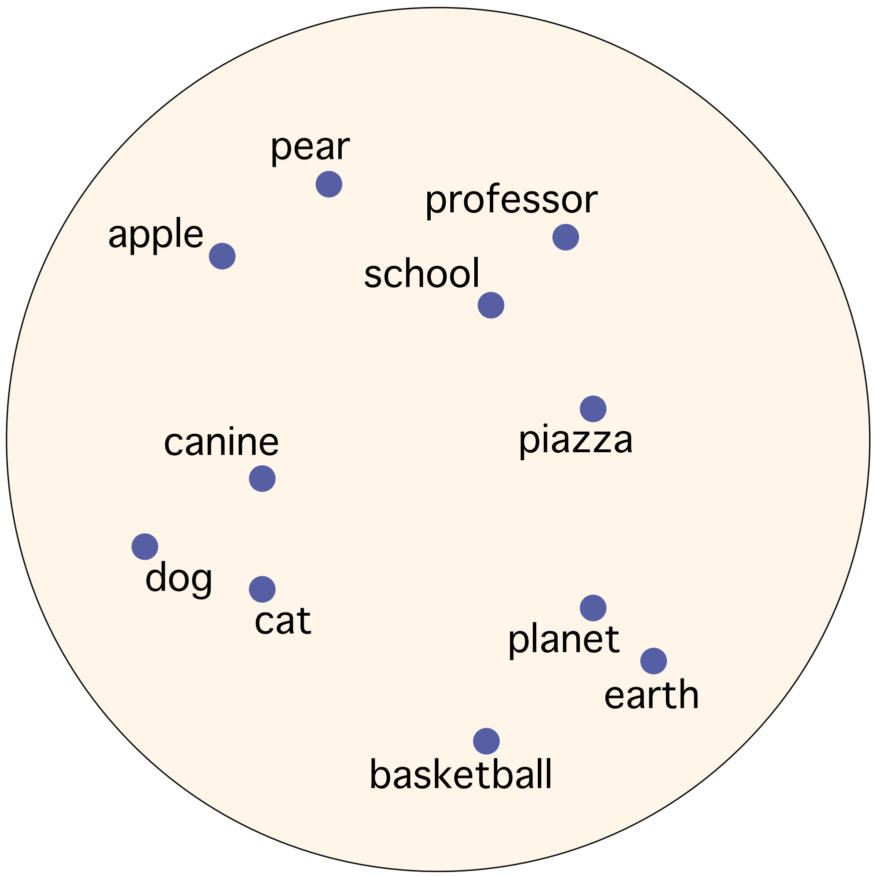
(NAACL 2019)

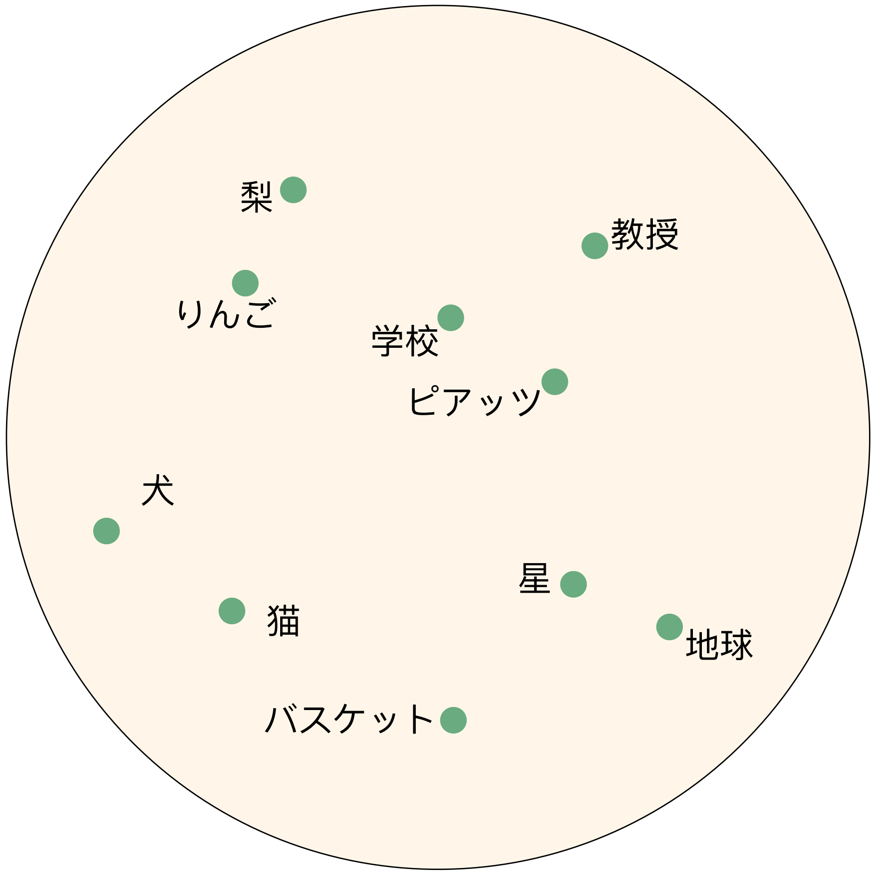
Bilingual Word Embedding

Bilingual Word Embedding

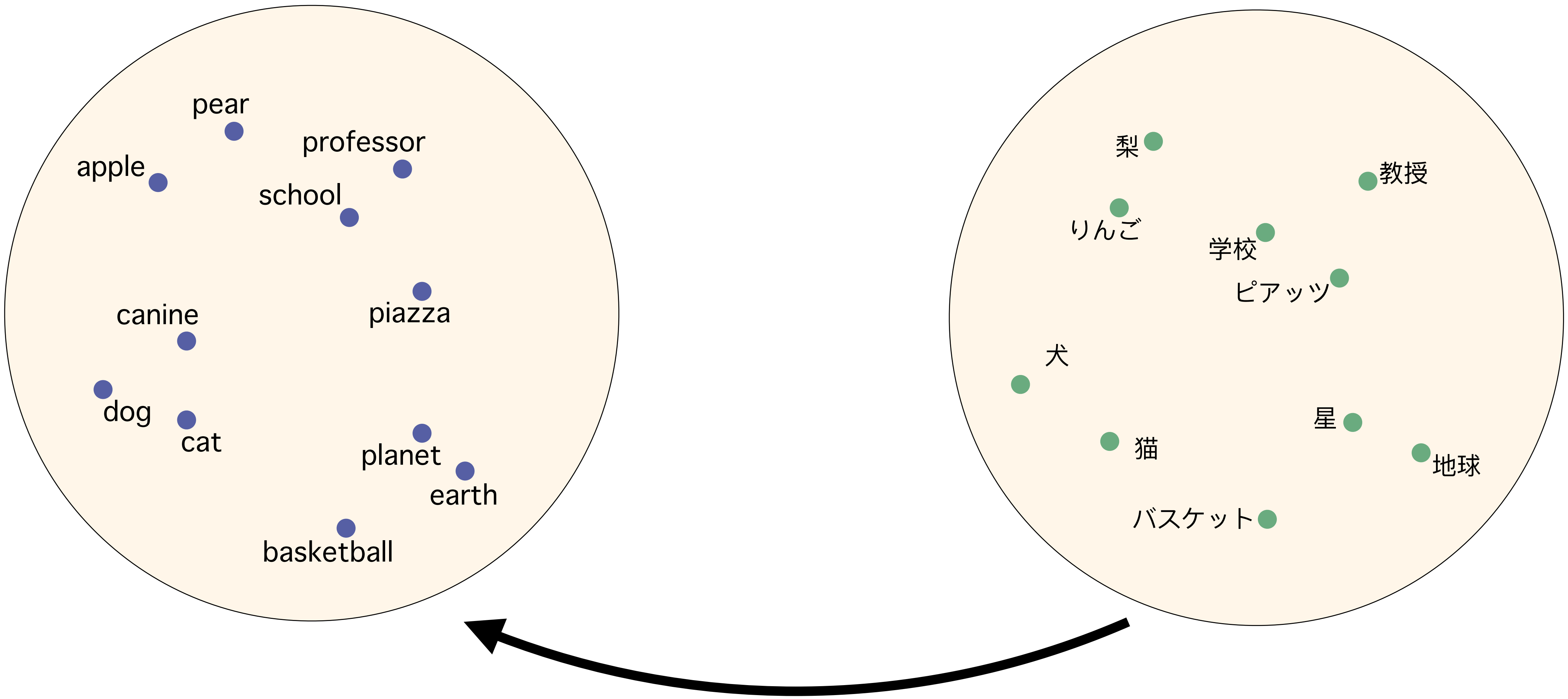


Bilingual Word Embedding



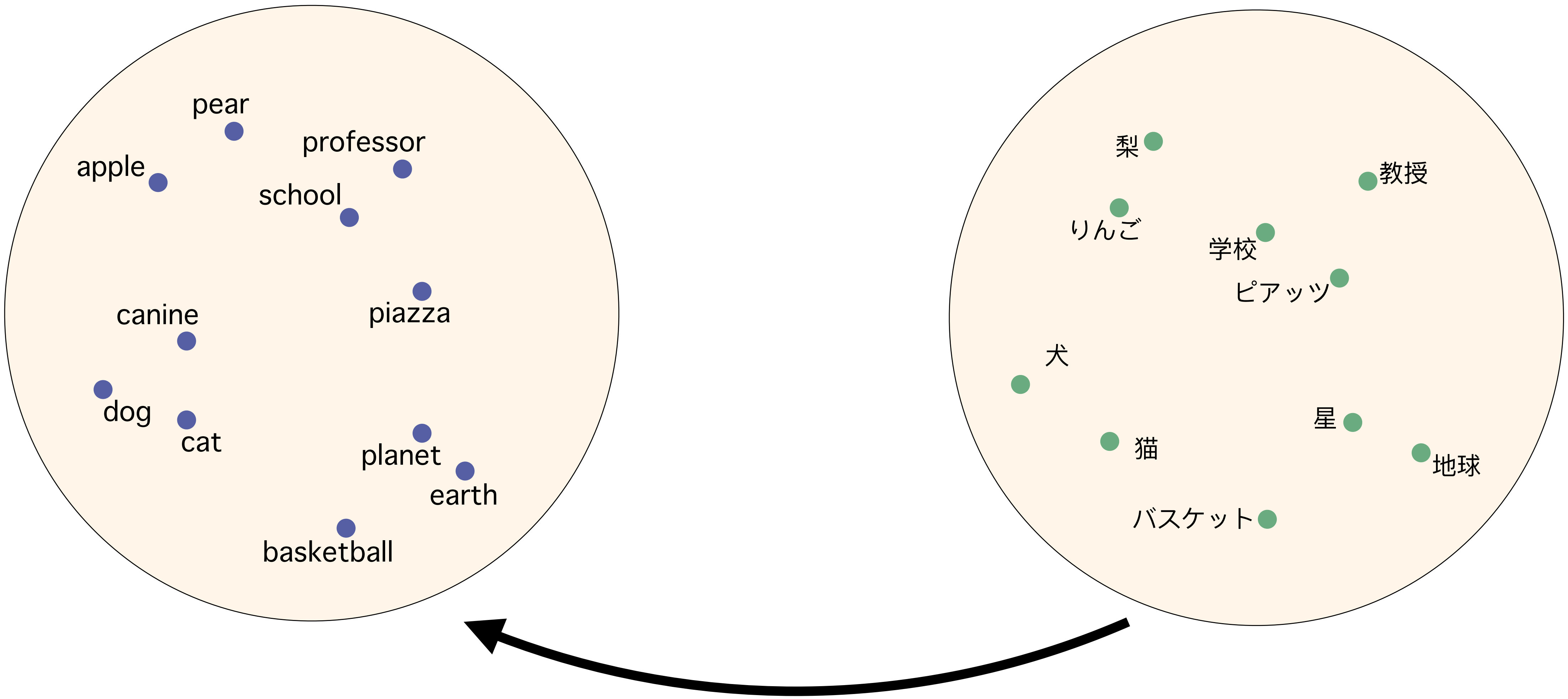


Bilingual Word Embedding



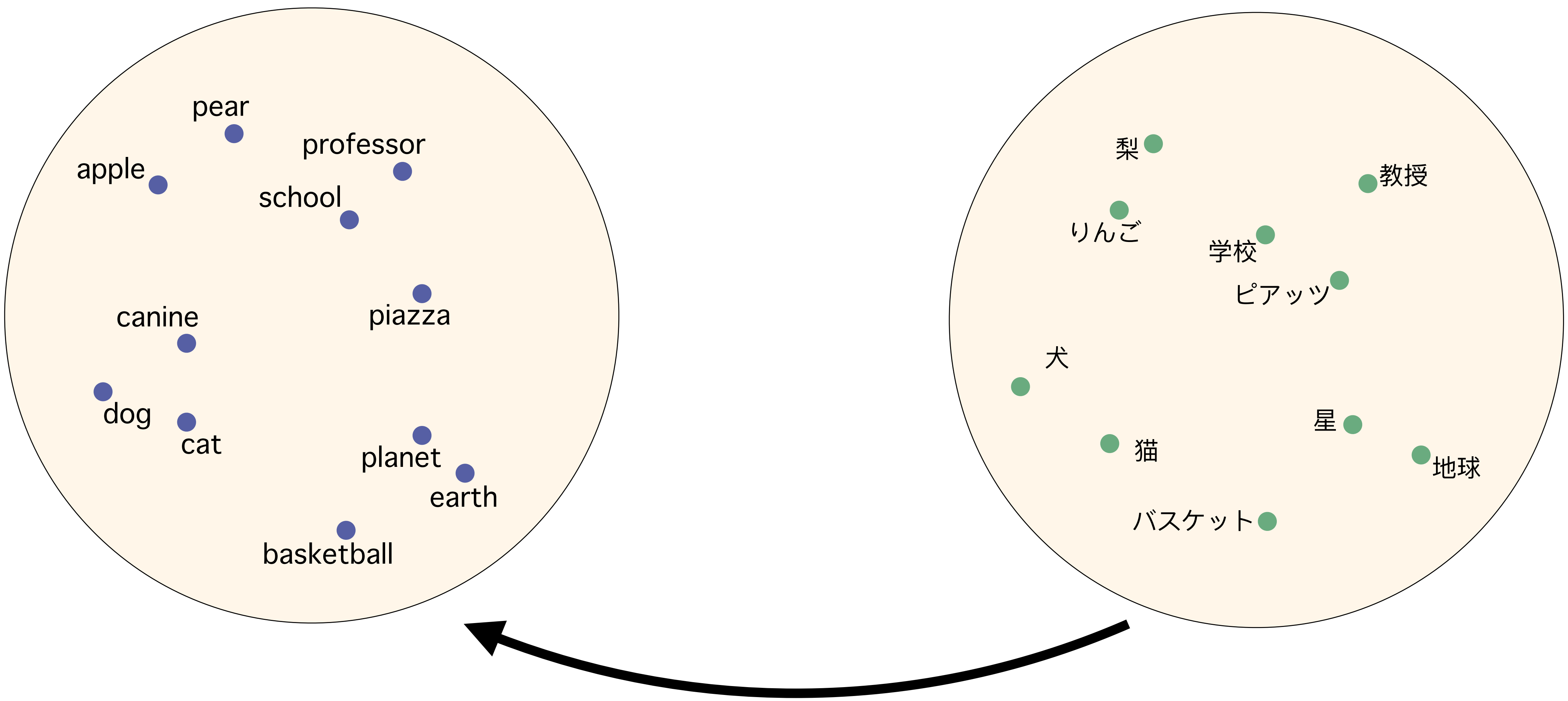
- Map word embeddings from different languages into a single vector space

Bilingual Word Embedding



- Map word embeddings from different languages into a single vector space
  - -Cross-lingual transfer

Bilingual Word Embedding



- Map word embeddings from different languages into a single vector space
  - -Cross-lingual transfer
  - -Cross-lingual NLP tasks

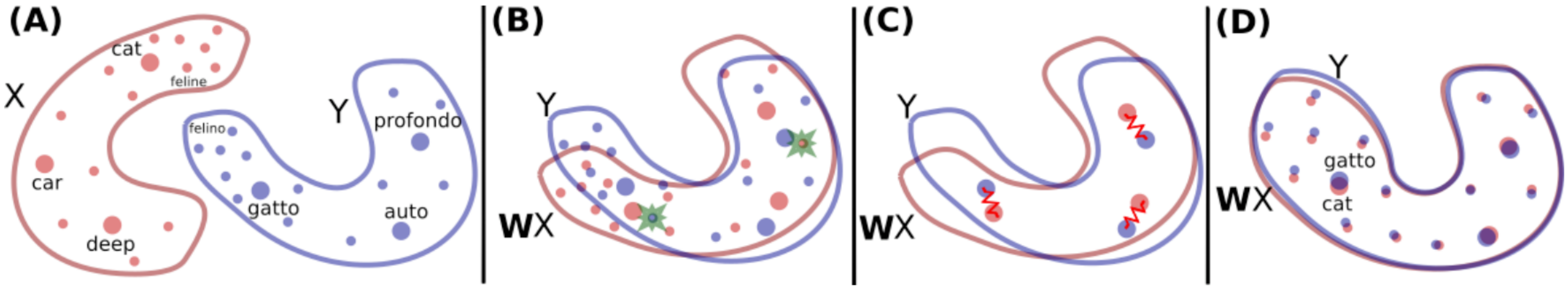
Previous Work on Unsupervised BWE

Previous Work on Unsupervised BWE

- Unsupervised methods of minimization some form of distance between distributions of discrete vector sets:

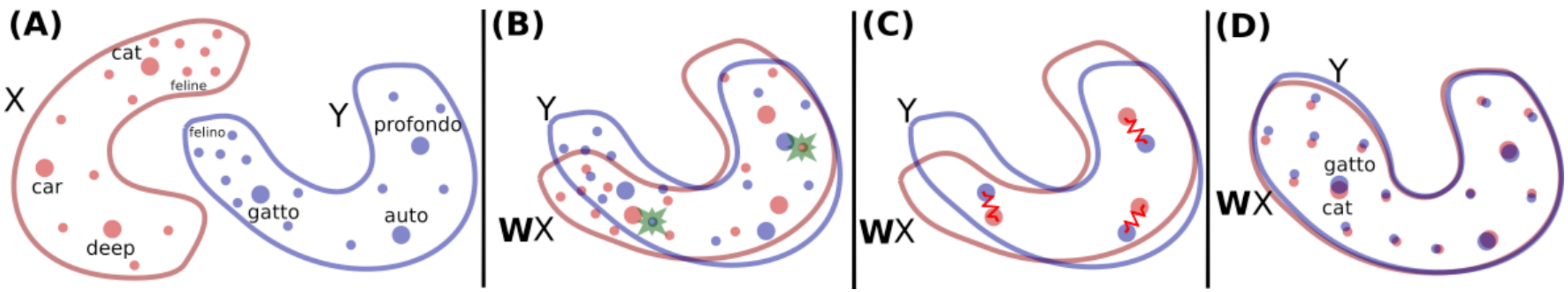
Previous Work on Unsupervised BWE

- Unsupervised methods of minimization some form of distance between distributions of discrete vector sets:



Previous Work on Unsupervised BWE

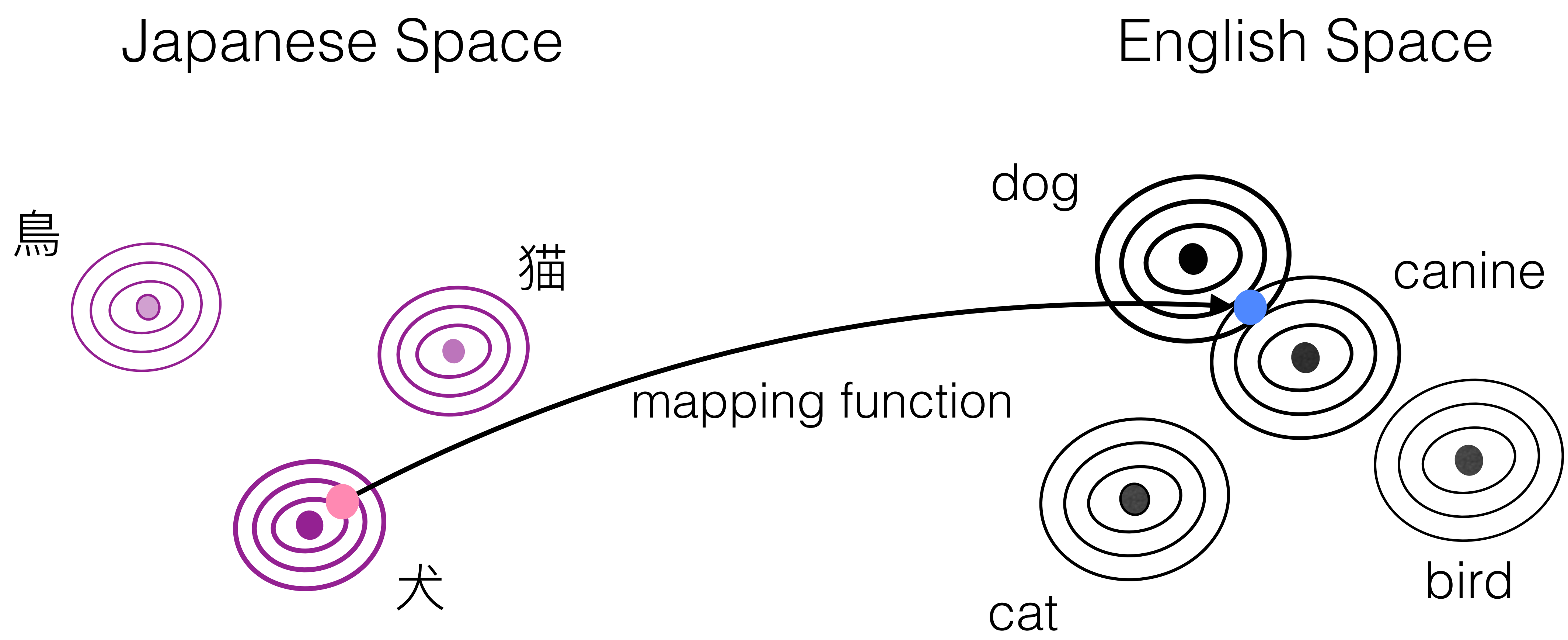
- Unsupervised methods of minimization some form of distance between distributions of discrete vector sets:



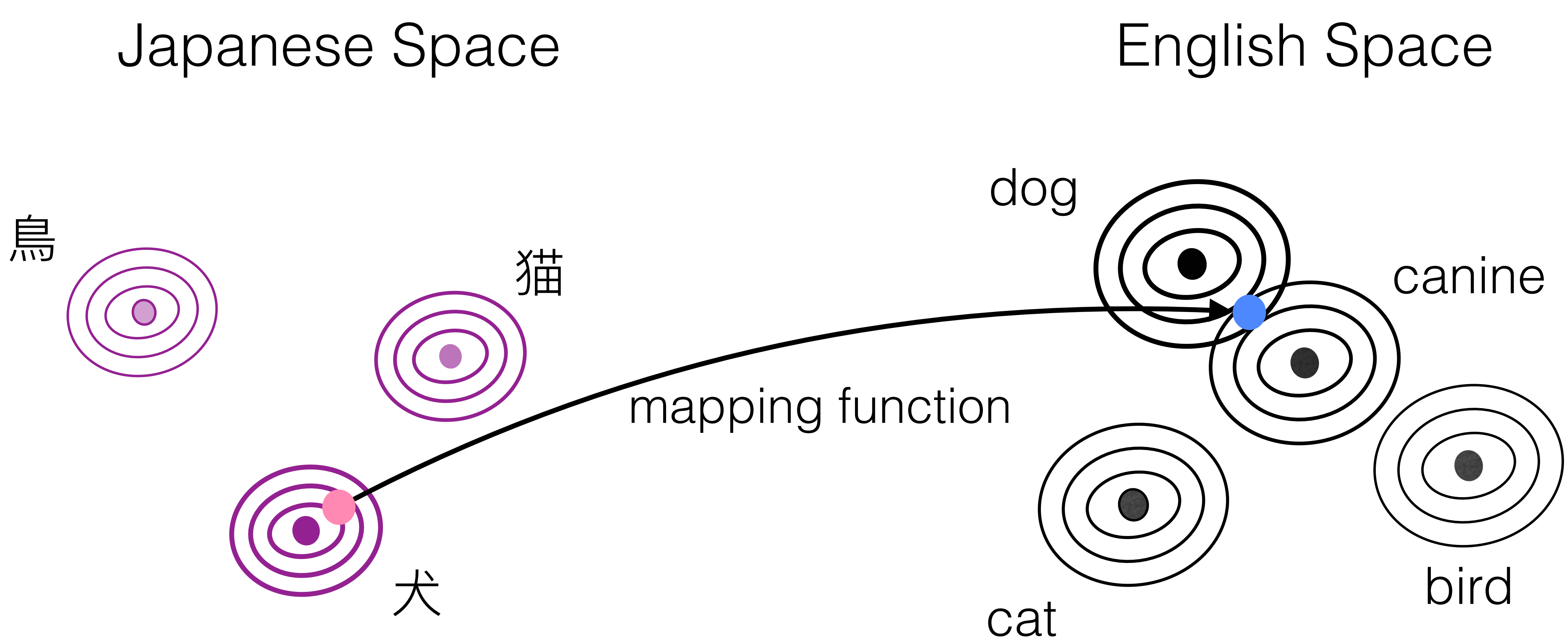
- No direct probabilistic interpretation, not a "typical" unsupervised generative model

Density Mapping for Bilingual Word Embedding (DeMa-BWE)

Density Mapping for Bilingual Word Embedding (DeMa-BWE)



Density Mapping for Bilingual Word Embedding (DeMa-BWE)



- Mapping function is learned with **normalizing flow**

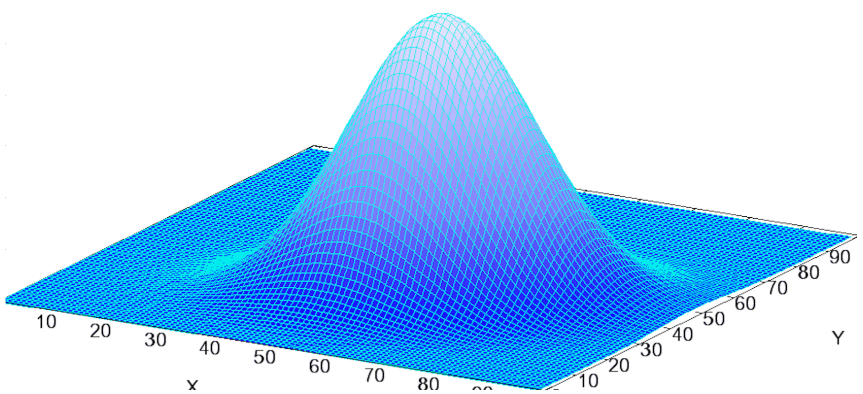
Normalizing Flows

Normalizing Flows

$$X \sim P(X)$$

Normalizing Flows

$$X \sim P(X)$$

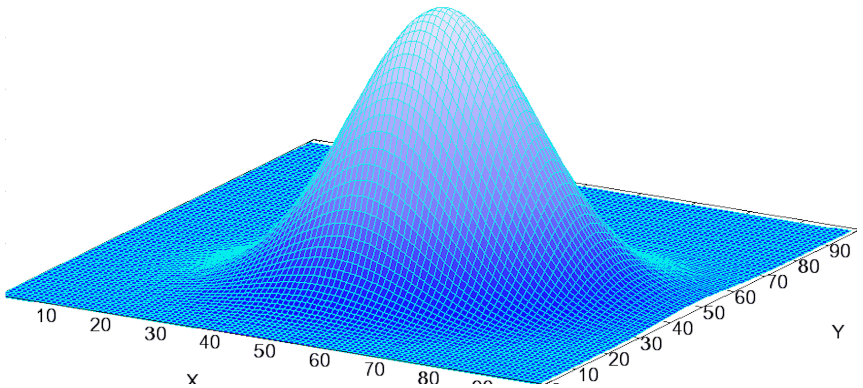


$$Z \sim N(0, I)$$

Normalizing Flows

$$X = f_\theta^{-1}(Z)$$

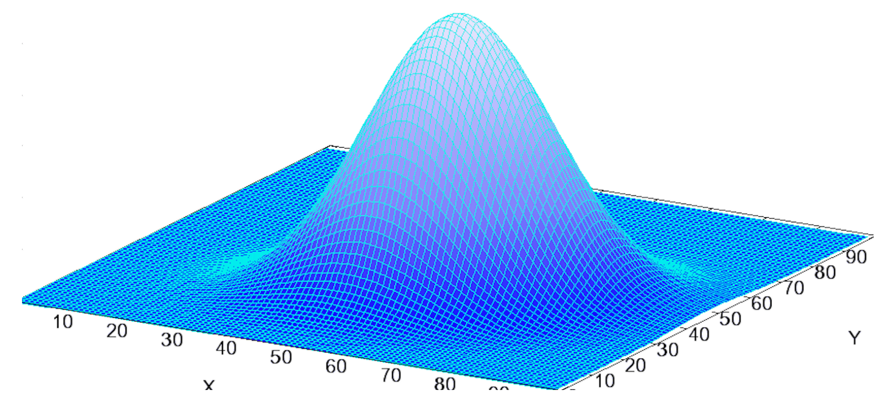
$$Z = f_\theta(X)$$



$$X \sim P(X)$$

$$Z \sim N(0, I)$$

Normalizing Flows



$$X = f_\theta^{-1}(Z)$$

$$Z = f_\theta(X)$$

$$X \sim P(X)$$

$$Z \sim N(0, I)$$

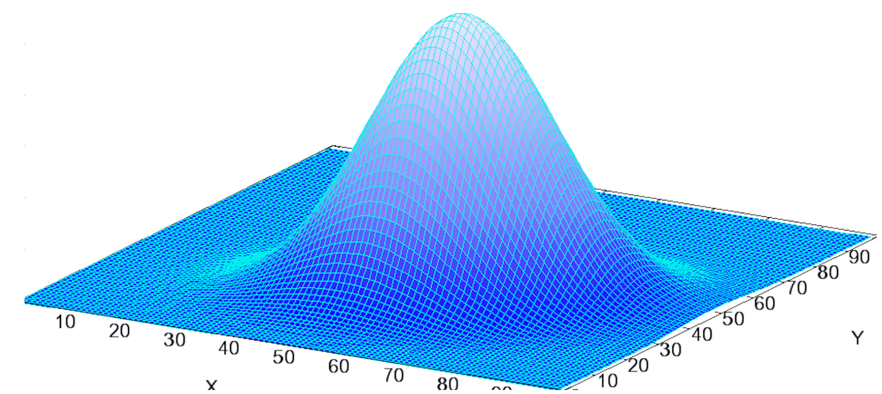
Change of variable formula:

$$p_\theta(x) = p_Z(f_\theta(x)) \left| \det\left(\frac{\partial f_\theta(x)}{\partial x}\right) \right|$$

Normalizing Flows

$$X = f_\theta^{-1}(Z)$$

$$Z = f_\theta(X)$$



$X \sim P(X)$ $Z \sim N(0, I)$

Change of variable formula:

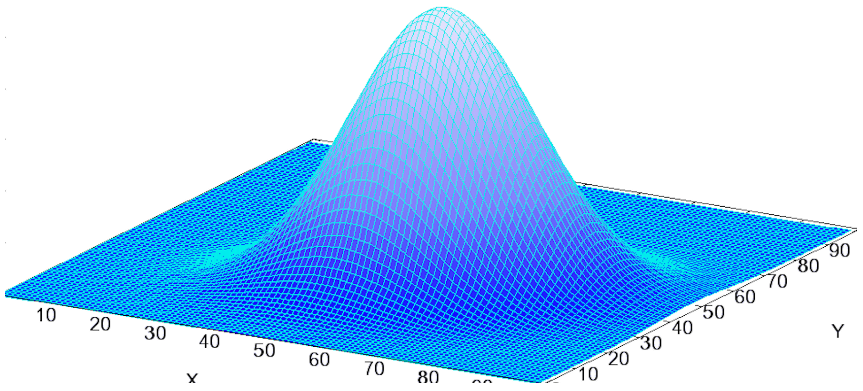
$$p_\theta(x) = p_Z(f_\theta(x)) \left| \det\left(\frac{\partial f_\theta(x)}{\partial x}\right) \right|$$

Intuitively, prevents degenerative mapping of everything to zero vector

Normalizing Flows

$$X = f_\theta^{-1}(Z)$$

$$Z = f_\theta(X)$$



$X \sim P(X)$ $\quad$ $Z \sim N(0, I)$

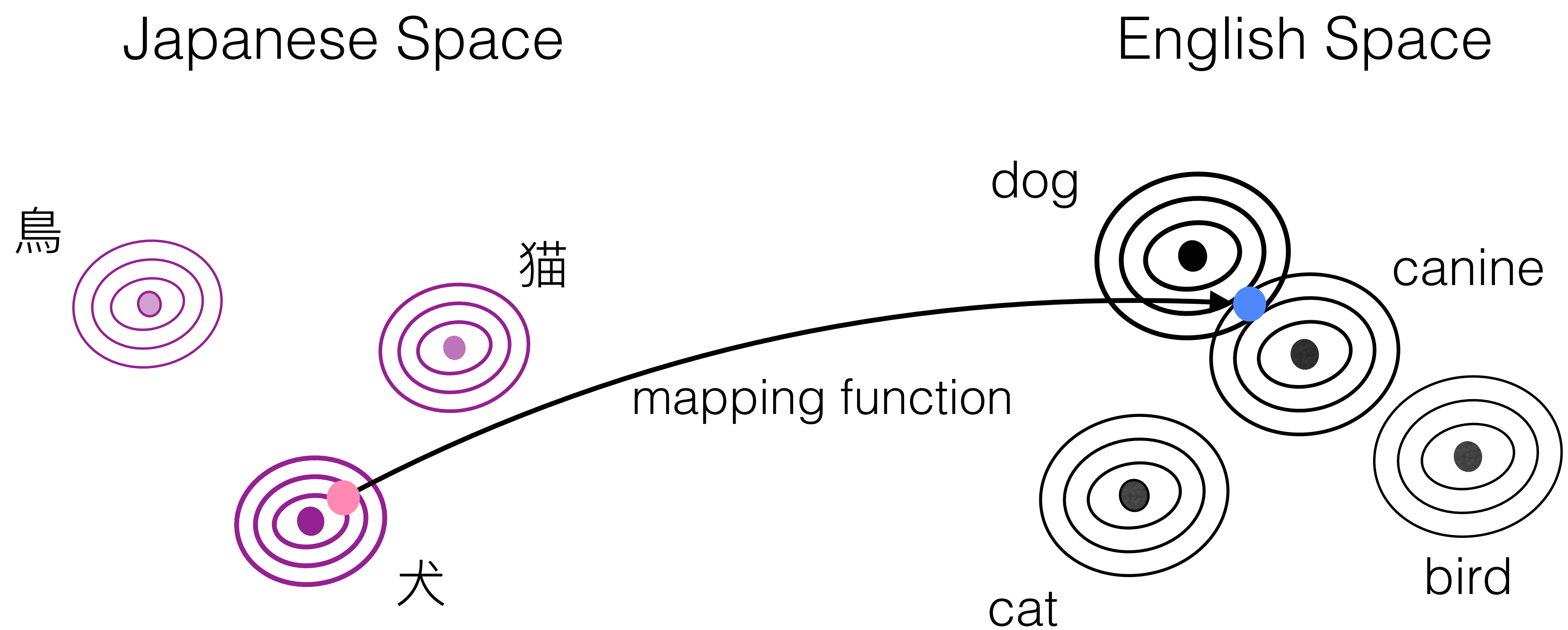
Change of variable formula:

$$p_\theta(x) = p_Z(f_\theta(x)) \left| \det\left(\frac{\partial f_\theta(x)}{\partial x}\right) \right|$$

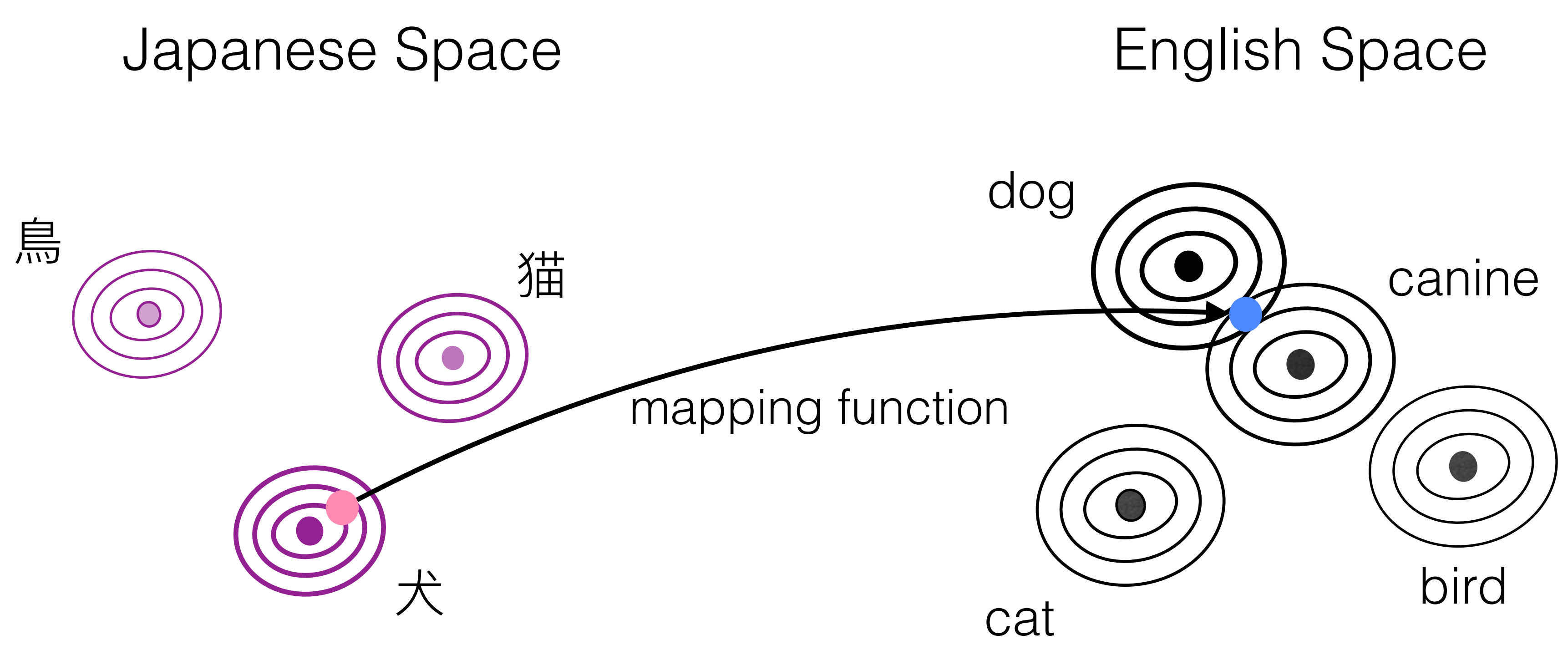
Intuitively, prevents degenerative mapping of everything to zero vector

Normalizing Flow: A series of such invertible transformations f

DeMa-BWE: Preliminaries

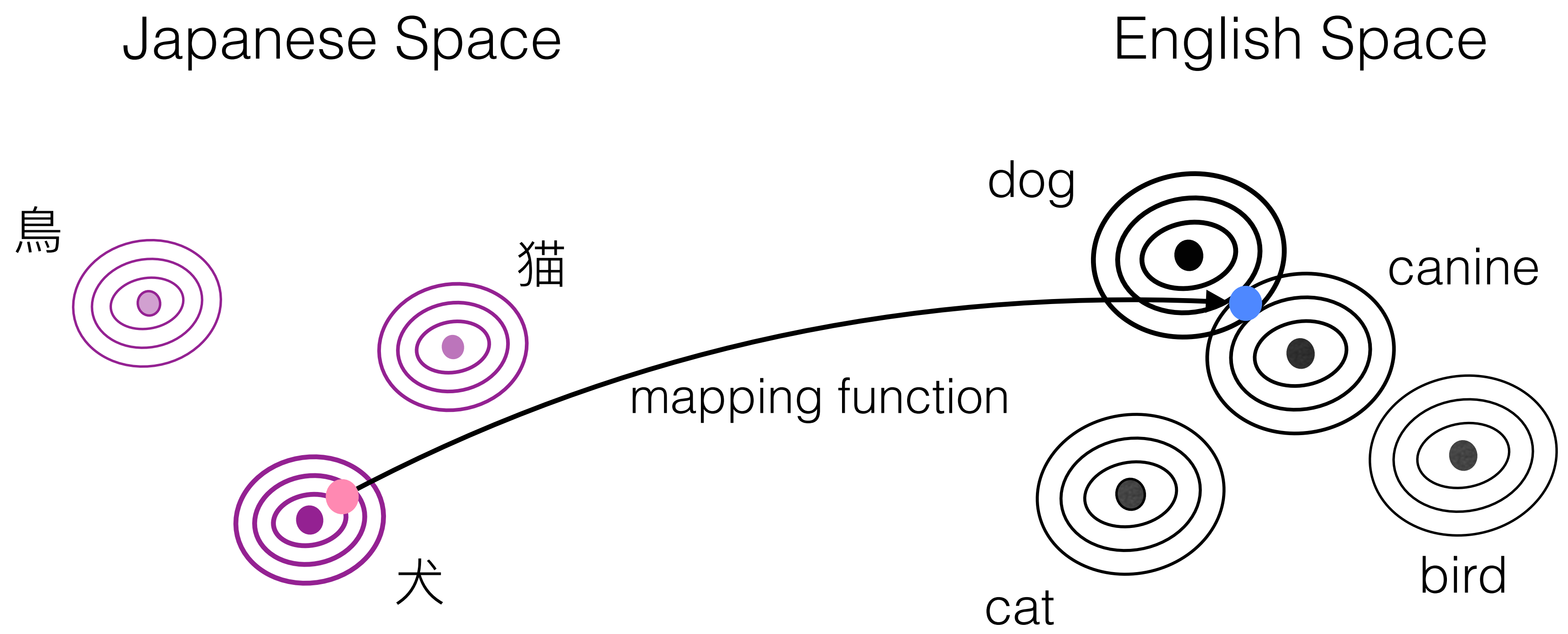


DeMa-BWE: Preliminaries



Notations:

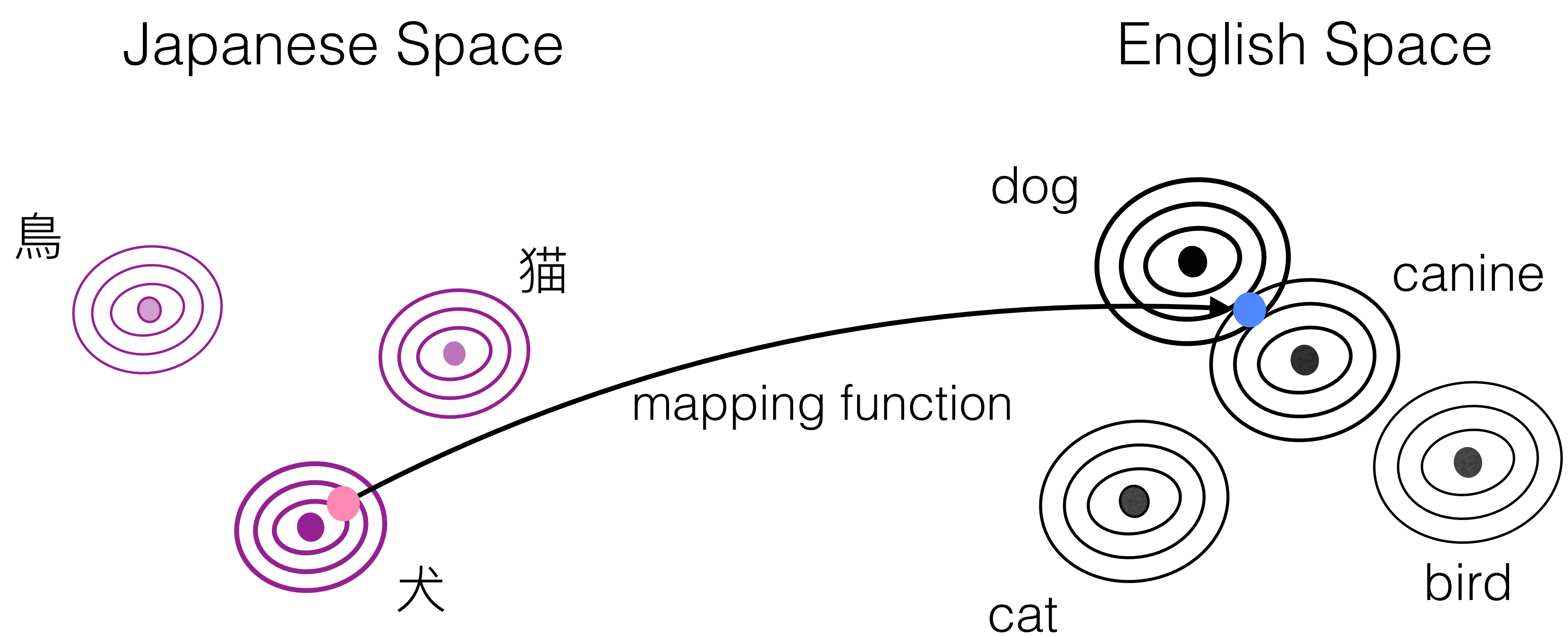
DeMa-BWE: Preliminaries



Notations:

$\mathbf{x} \in \mathbb{R}^d, \quad \mathbf{y} \in \mathbb{R}^d$: denote vectors in the src and tgt embedding space

DeMa-BWE: Preliminaries

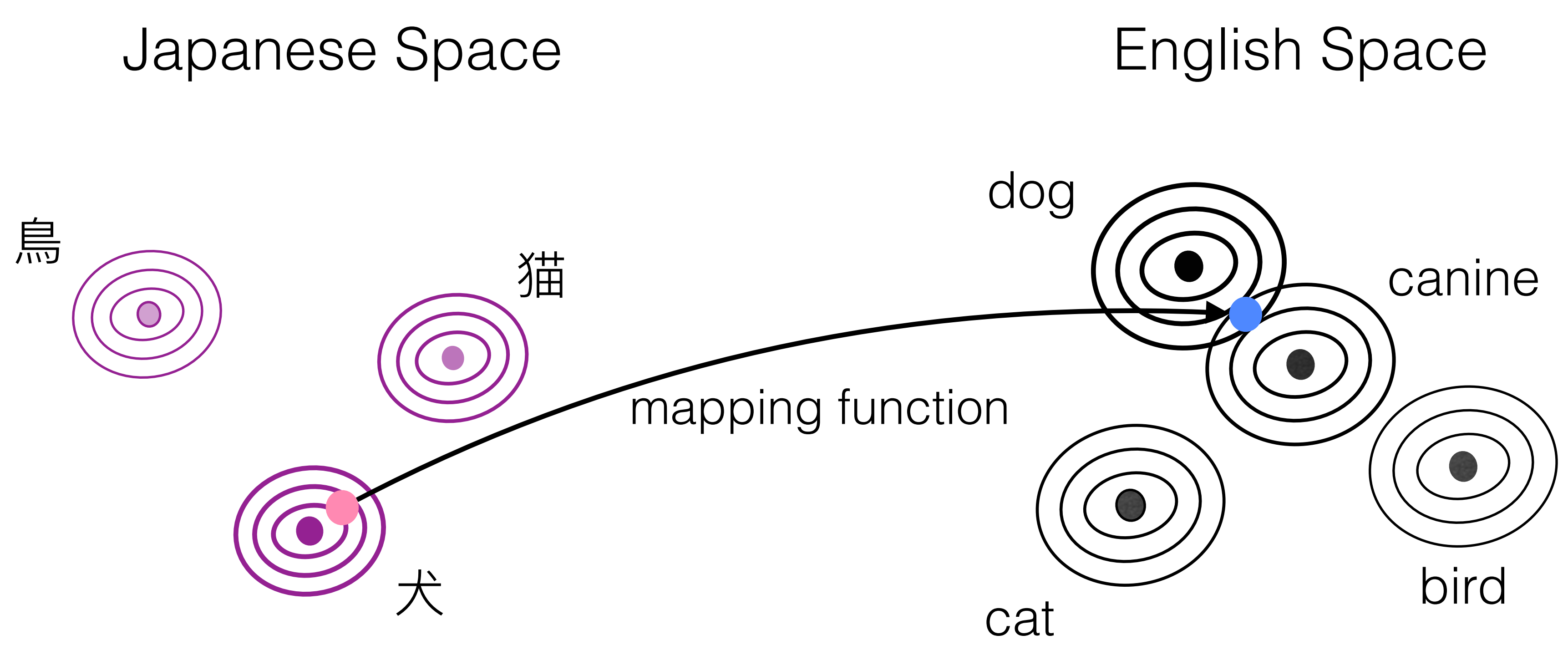


Notations:

$\mathbf{x} \in \mathbb{R}^d, \quad \mathbf{y} \in \mathbb{R}^d$: denote vectors in the src and tgt embedding space

$x_i, \quad y_j$: denote an actual word in src and tgt vocabularies

DeMa-BWE: Preliminaries



Notations:

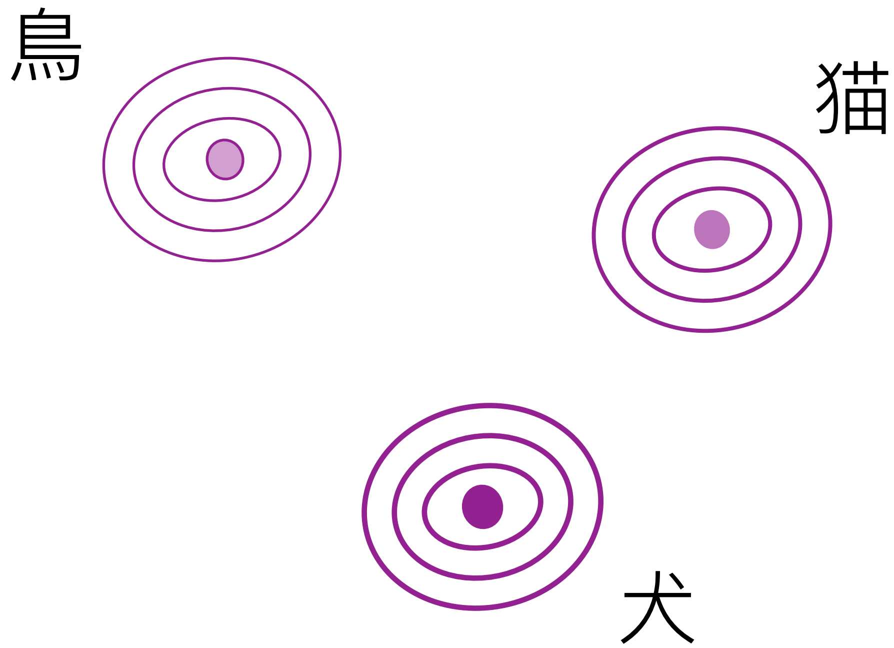
$\mathbf{x} \in \mathbb{R}^d, \quad \mathbf{y} \in \mathbb{R}^d$: denote vectors in the src and tgt embedding space

$x_i, \quad y_j$: denote an actual word in src and tgt vocabularies

$f_{xy}, \quad f_{yx}$: denote src->tgt, and tgt-src mapping functions

Prior Distribution

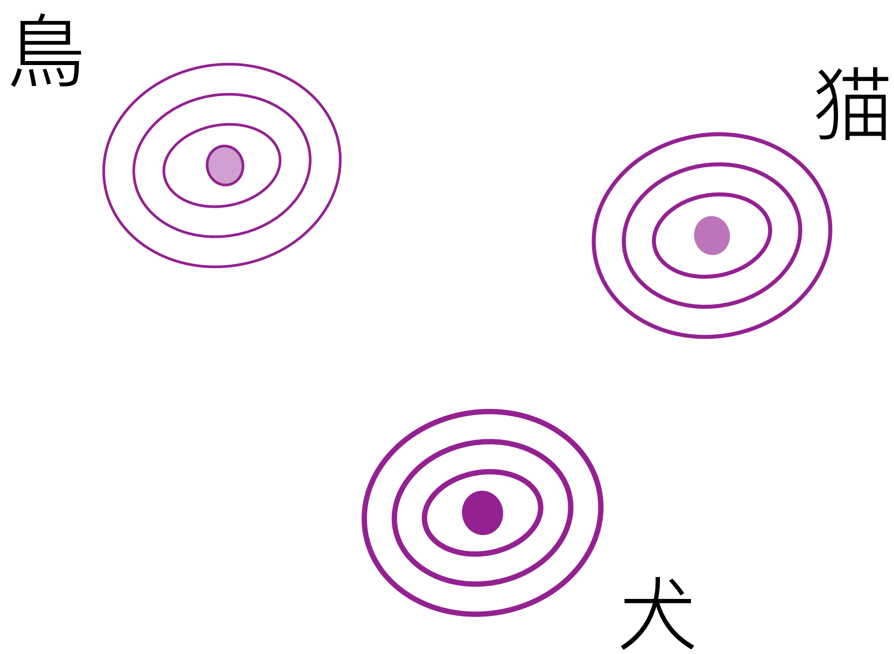
- Assumption on the monolingual word embedding space: Gaussian mixture model



Prior Distribution

- Assumption on the monolingual word embedding space: Gaussian mixture model

$$p(\mathbf{x}) = \sum_{i \in \{1,\ldots,N_x\}} \pi(x_i)\tilde{p}(\mathbf{x}|x_i)$$

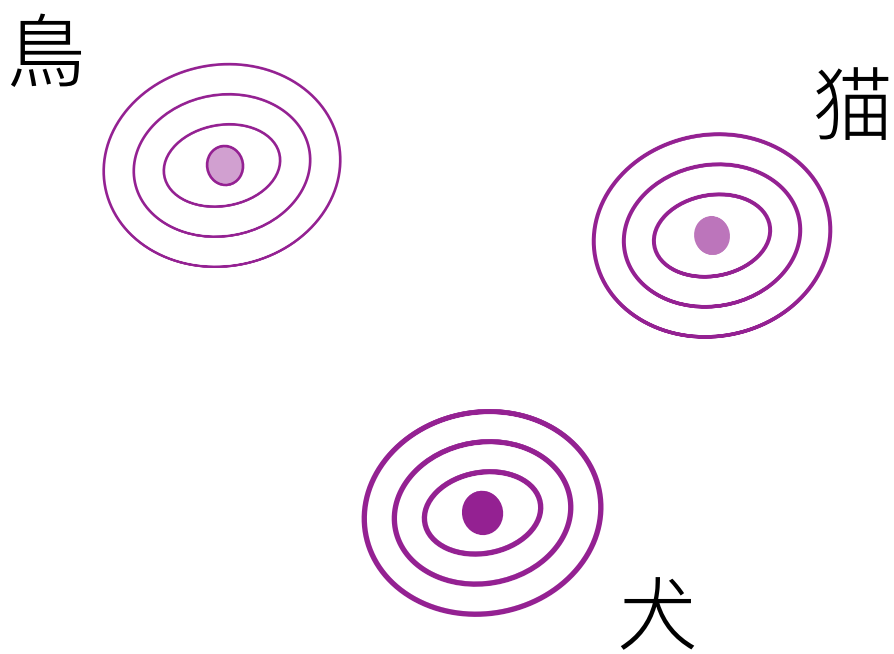


Prior Distribution

- Assumption on the monolingual word embedding space: Gaussian mixture model

$$p(\mathbf{x}) = \sum_{i \in \{1,\ldots,N_x\}} \pi(x_i)\tilde{p}(\mathbf{x}|x_i)$$

$$\tilde{p}(\mathbf{x}|x_i) = \mathcal{N}(\mathbf{x}|\mathbf{x}_i, \sigma_x^2\mathbf{I})$$



DeMa-BWE: Density Matching

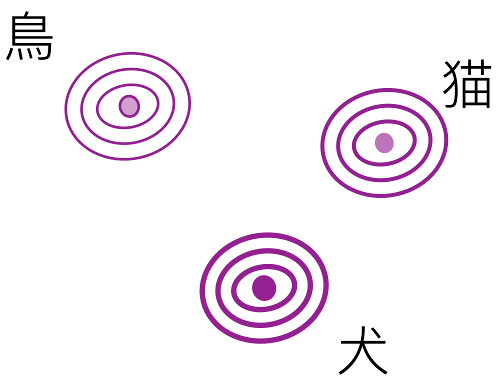
DeMa-BWE: Density Matching

- Sampling a continuous vector from the GMM

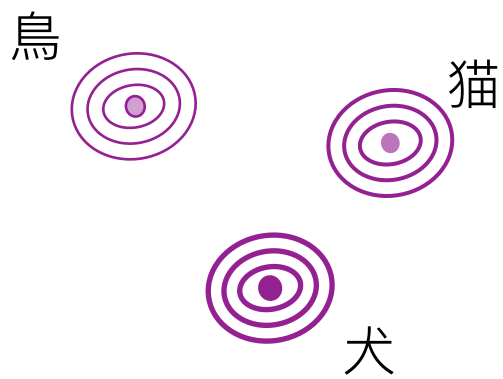
DeMa-BWE: Density Matching

- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$



DeMa-BWE: Density Matching



- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

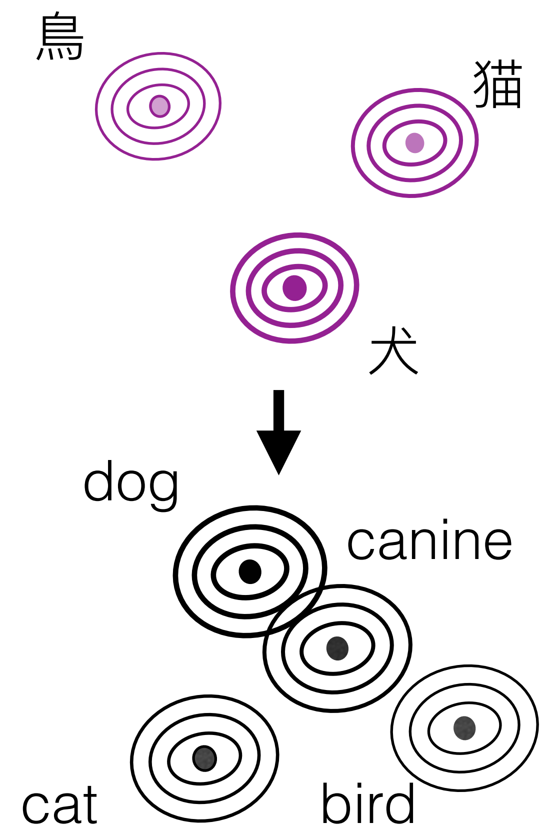
DeMa-BWE: Density Matching

- Sampling a continuous vector from the GMM

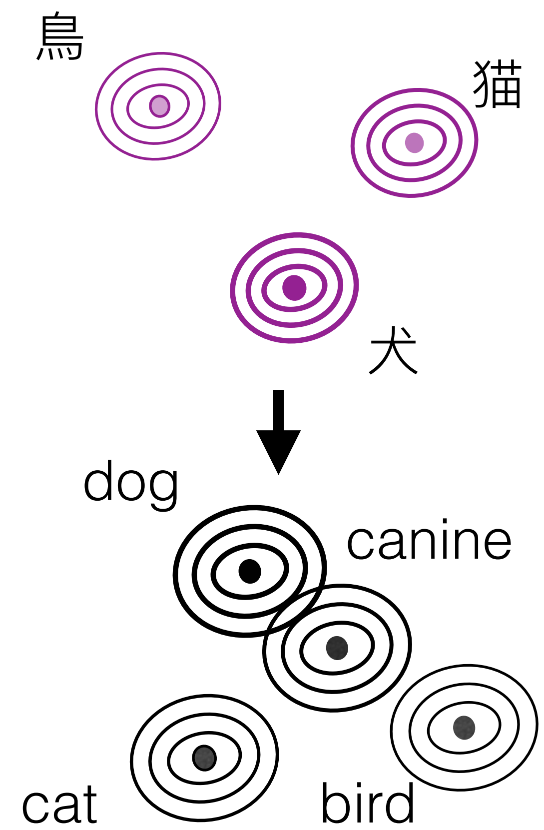
$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy} \cdot$$



DeMa-BWE: Density Matching



- Sampling a continuous vector from the GMM

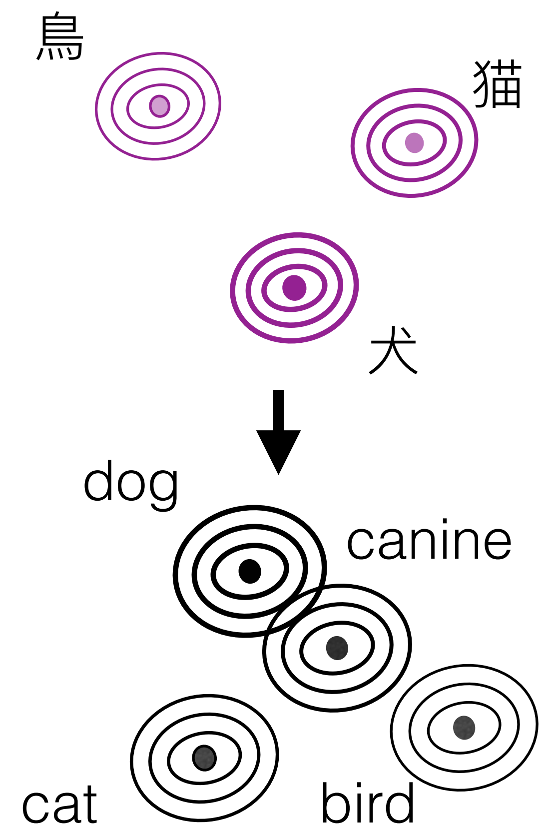
$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy}\cdot$$

- Computing the density of x in the mapped target space

DeMa-BWE: Density Matching



- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

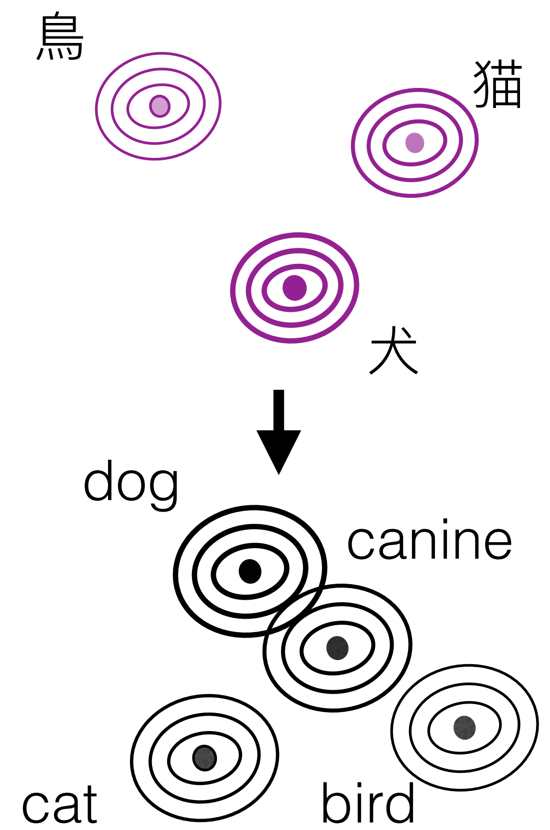
- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy}\cdot$$

- Computing the density of x in the mapped target space

$$\log p(\mathbf{x}; \mathbf{W}_{xy}) = \log p(\mathbf{y}) + \log \left|\det(\mathbf{W}_{xy})\right|$$

DeMa-BWE: Density Matching



- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy}\cdot$$

- Computing the density of x in the mapped target space

$$\log p(\mathbf{x}; \mathbf{W}_{xy}) = \log p(\mathbf{y}) + \log \left|\det(\mathbf{W}_{xy})\right|$$

$$\text{minimize:} \;\; \mathrm{KL}(p(\mathbf{x}) || p(\mathbf{x}; \mathbf{W}_{xy}))$$

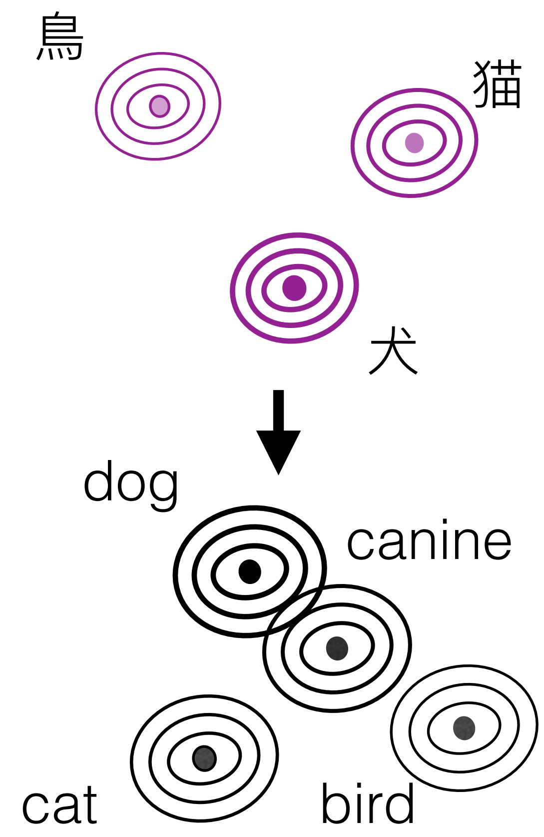
DeMa-BWE: Density Matching

- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy}\cdot$$

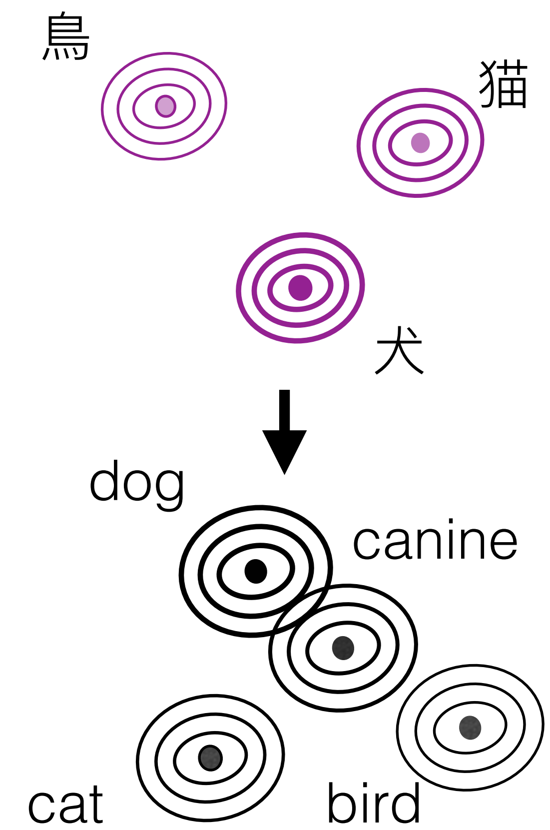


- Computing the density of x in the mapped target space

$$\log p(\mathbf{x}; \mathbf{W}_{xy}) = \log p(\mathbf{y}) + \log \left|\det(\mathbf{W}_{xy})\right|$$

- Objective: $\text{minimize: } \mathrm{KL}(p(\mathbf{x})||p(\mathbf{x}; \mathbf{W}_{xy}))$

DeMa-BWE: Density Matching



- Sampling a continuous vector from the GMM

$$x_i \sim \pi(x_i) \qquad \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)$$

- Apply the mapping function f_{xy} to obtain the transformed vector in the target space.

$$f_{xy}(\cdot) = \mathbf{W}_{xy}\cdot$$

- Computing the density of x in the mapped target space

$$\log p(\mathbf{x}; \mathbf{W}_{xy}) = \log p(\mathbf{y}) + \log |\det(\mathbf{W}_{xy})|$$

- Objective: $\text{minimize: } \mathrm{KL}(p(\mathbf{x})||p(\mathbf{x}; \mathbf{W}_{xy}))$

$$\mathcal{L}_{xy} = \mathrm{E}_{\mathbf{x}\sim p(\mathbf{x})}[\log p(\mathbf{y}) + \log |\det(\mathbf{W}_{xy})|]$$

Method Details

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

$$\mathcal{L}_{bt} = \mathrm{E}_{x_i \sim \pi(x_i), \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)} \left[g(\mathbf{W}_{yx} \mathbf{W}_{xy} \mathbf{x}, \mathbf{x}) \right] + \mathrm{E}_{y_j \sim \pi(y_j), \mathbf{y} \sim \tilde{p}(\mathbf{y}|x_j)} \left[g(\mathbf{W}_{xy} \mathbf{W}_{yx} \mathbf{y}, \mathbf{y}) \right]$$

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

$$\mathcal{L}_{bt} = \mathrm{E}_{x_i \sim \pi(x_i), \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)} \left[g(\mathbf{W}_{yx}\mathbf{W}_{xy}\mathbf{x}, \mathbf{x}) \right] + \mathrm{E}_{y_j \sim \pi(y_j), \mathbf{y} \sim \tilde{p}(\mathbf{y}|x_j)} \left[g(\mathbf{W}_{xy}\mathbf{W}_{yx}\mathbf{y}, \mathbf{y}) \right]$$

- **Weak Supervision w/ Identical Strings:** Take advantage of the fact that identical strings are usually the same word in both languages

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

$$\mathcal{L}_{bt} = \mathrm{E}_{x_i \sim \pi(x_i), \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)} \left[g(\mathbf{W}_{yx}\mathbf{W}_{xy}\mathbf{x}, \mathbf{x}) \right] + \mathrm{E}_{y_j \sim \pi(y_j), \mathbf{y} \sim \tilde{p}(\mathbf{y}|x_j)} \left[g(\mathbf{W}_{xy}\mathbf{W}_{yx}\mathbf{y}, \mathbf{y}) \right]$$

- **Weak Supervision w/ Identical Strings:** Take advantage of the fact that identical strings are usually the same word in both languages

$$\mathcal{L}_{sup} = \sum_{v \in \mathcal{W}_{id}} g(\mathbf{v}_x \mathbf{W}_{xy}^T, \mathbf{v}_y) + g(\mathbf{v}_y \mathbf{W}_{yx}^T, \mathbf{v}_x)$$

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

$$\mathcal{L}_{bt} = \mathrm{E}_{x_i \sim \pi(x_i), \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)} \left[g(\mathbf{W}_{yx} \mathbf{W}_{xy} \mathbf{x}, \mathbf{x}) \right] + \mathrm{E}_{y_j \sim \pi(y_j), \mathbf{y} \sim \tilde{p}(\mathbf{y}|x_j)} \left[g(\mathbf{W}_{xy} \mathbf{W}_{yx} \mathbf{y}, \mathbf{y}) \right]$$

- **Weak Supervision w/ Identical Strings:** Take advantage of the fact that identical strings are usually the same word in both languages

$$\mathcal{L}_{sup} = \sum_{v \in \mathcal{W}_{id}} g(\mathbf{v}_x \mathbf{W}_{xy}^T, \mathbf{v}_y) + g(\mathbf{v}_y \mathbf{W}_{yx}^T, \mathbf{v}_x)$$

- **Alignment Selection Methods:** Use cross-domain similarity local scaling (CSLS)

Method Details

- **Weak Orthogonality Constraint:** Try to make sure that the transformation is close to orthogonal

$$\mathcal{L}_{bt} = \mathrm{E}_{x_i \sim \pi(x_i), \mathbf{x} \sim \tilde{p}(\mathbf{x}|x_i)} \left[g(\mathbf{W}_{yx} \mathbf{W}_{xy} \mathbf{x}, \mathbf{x}) \right] + \mathrm{E}_{y_j \sim \pi(y_j), \mathbf{y} \sim \tilde{p}(\mathbf{y}|x_j)} \left[g(\mathbf{W}_{xy} \mathbf{W}_{yx} \mathbf{y}, \mathbf{y}) \right]$$

- **Weak Supervision w/ Identical Strings:** Take advantage of the fact that identical strings are usually the same word in both languages

$$\mathcal{L}_{sup} = \sum_{v \in \mathcal{W}_{id}} g(\mathbf{v}_x \mathbf{W}_{xy}^T, \mathbf{v}_y) + g(\mathbf{v}_y \mathbf{W}_{yx}^T, \mathbf{v}_x)$$

- **Alignment Selection Methods:** Use cross-domain similarity local scaling (CSLS)

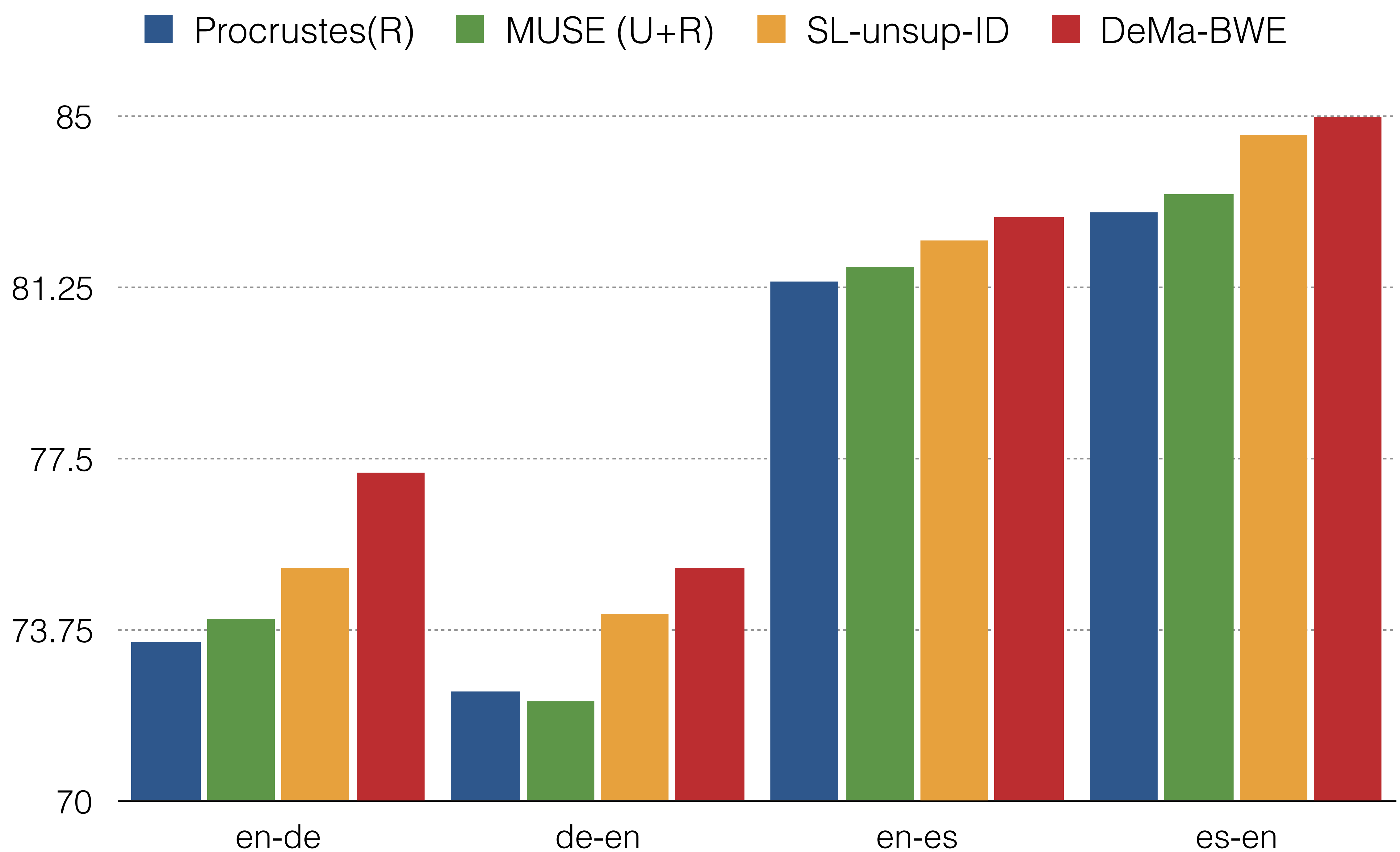
$$\mathrm{CSLS}(\mathbf{x}', \mathbf{y}) = 2\cos(\mathbf{x}', \mathbf{y}) - r_T(\mathbf{x}') - r_S(\mathbf{y})$$

Experiments

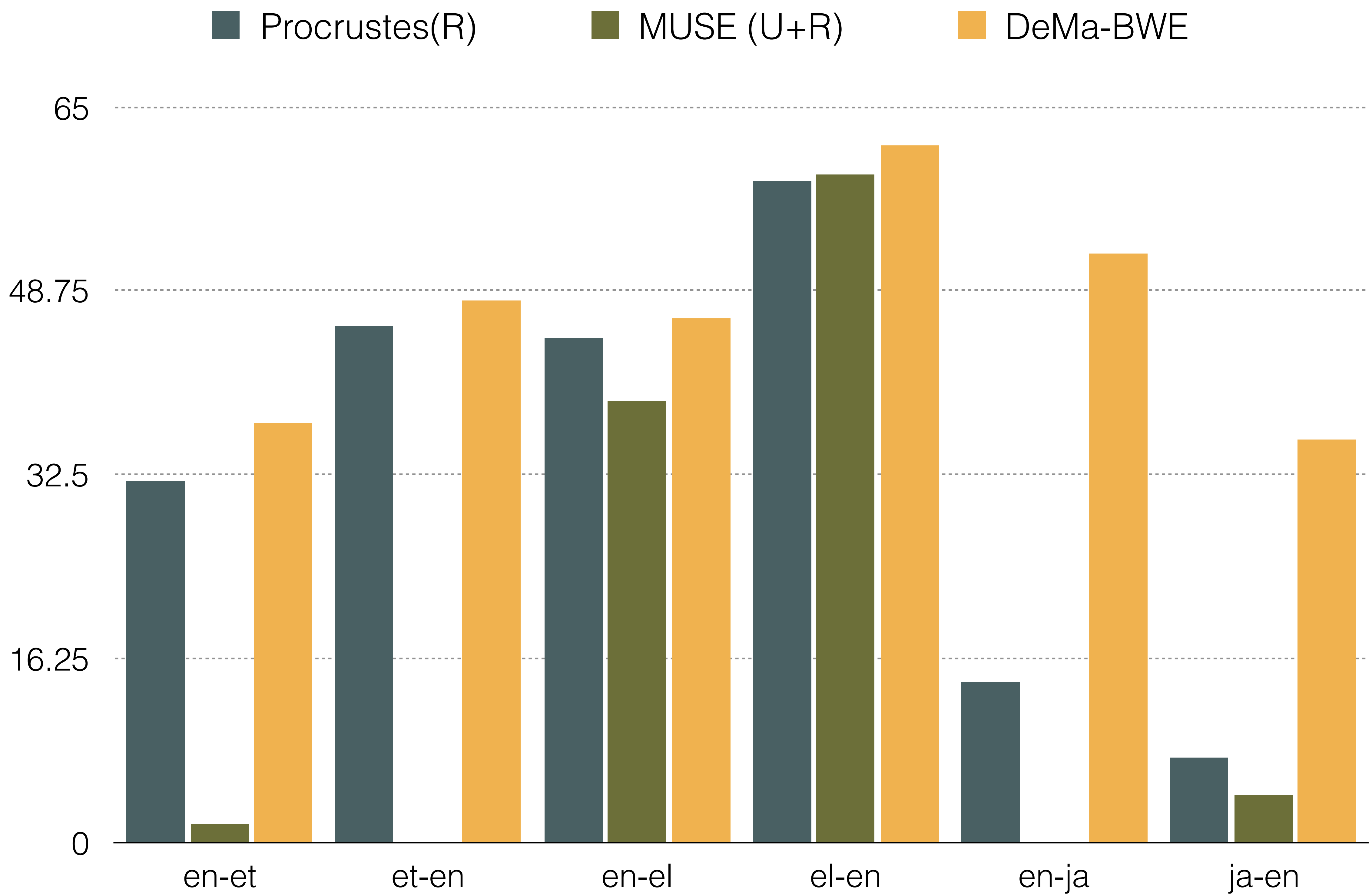
- **Dataset and Tasks**
  - Bilingual Lexicon Induction Task: MUSE dataset (Conneau el al., 2017)
  - Cross-lingual Word Similarity Task: SemEval 2017

- **Languages**
  - Baseline languages: en - es, de, fr, ru, zh, ja
  - Morphologically rich languages: en - et, fi, el, hu, pl, tr

Main Results on BLI (close languages)



Main Results on BLI (distant languages)



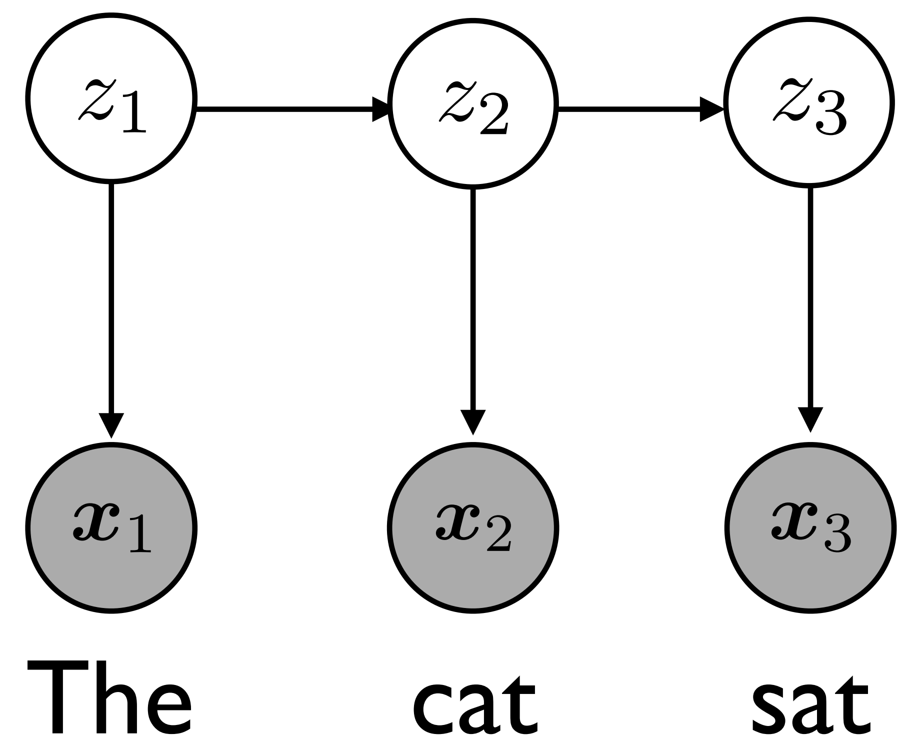
Language Technologies Institute

Unsupervised Learning of Syntactic Structure w/ Invertible Neural Projections

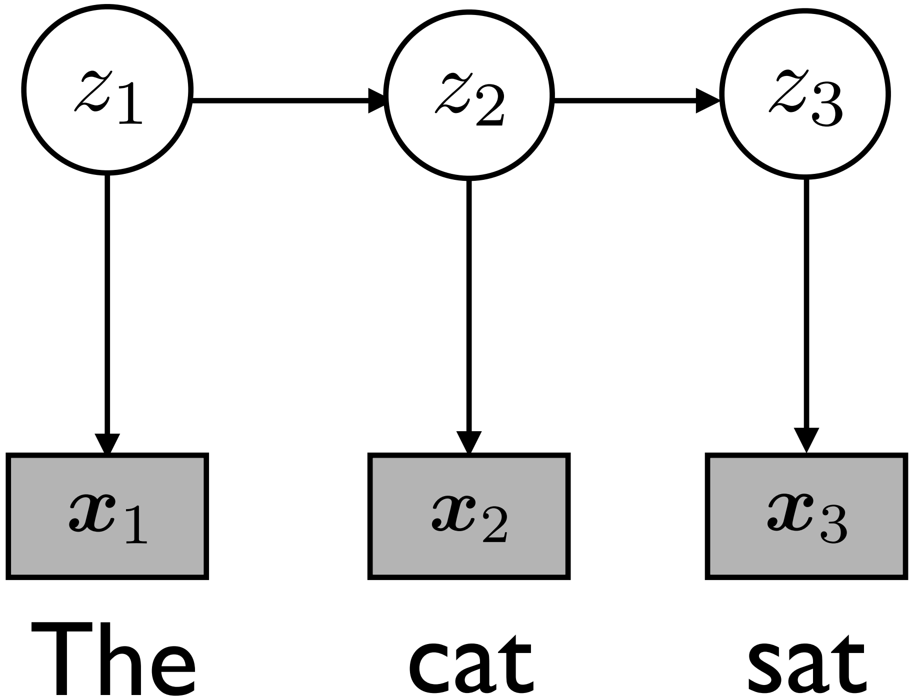
Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick

(EMNLP 2018)

HMM for Part-of-Speech Induction



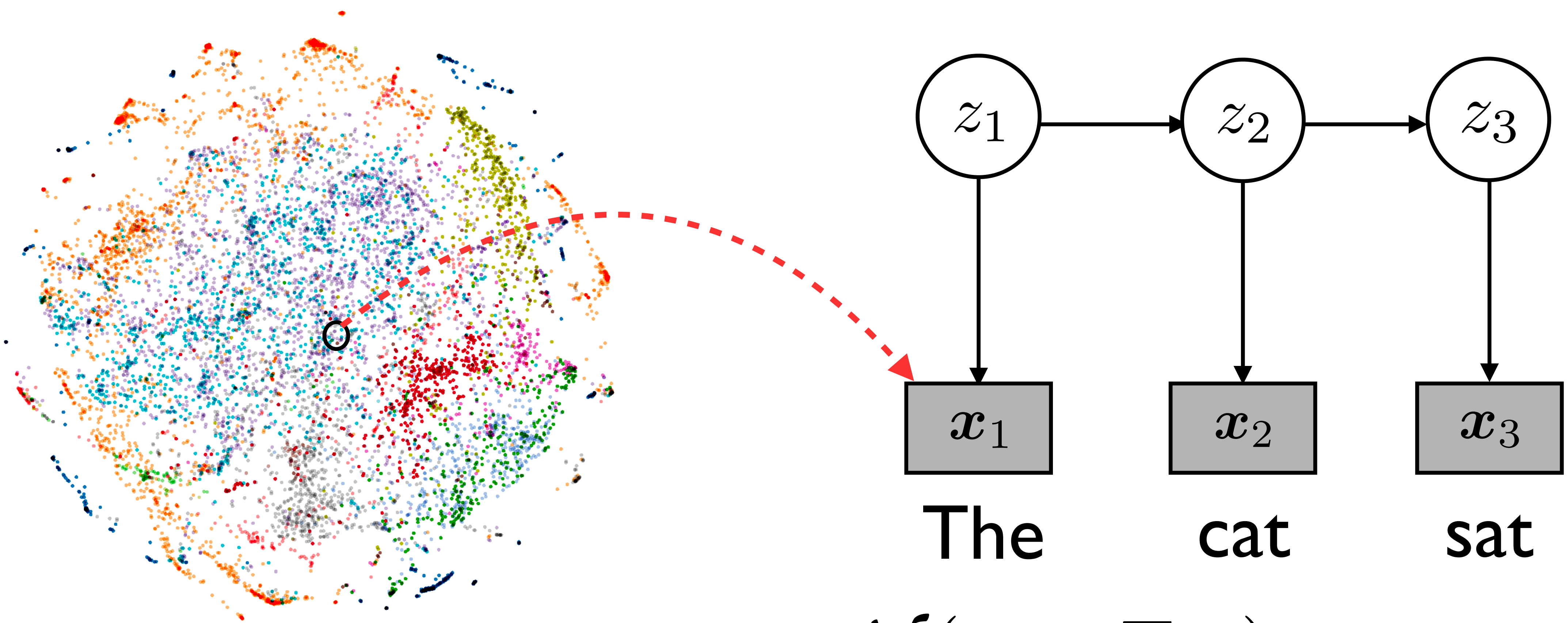
Gaussian HMM for POS Induction



$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \Sigma_{z_i})$$

[Lin et al. 2015]

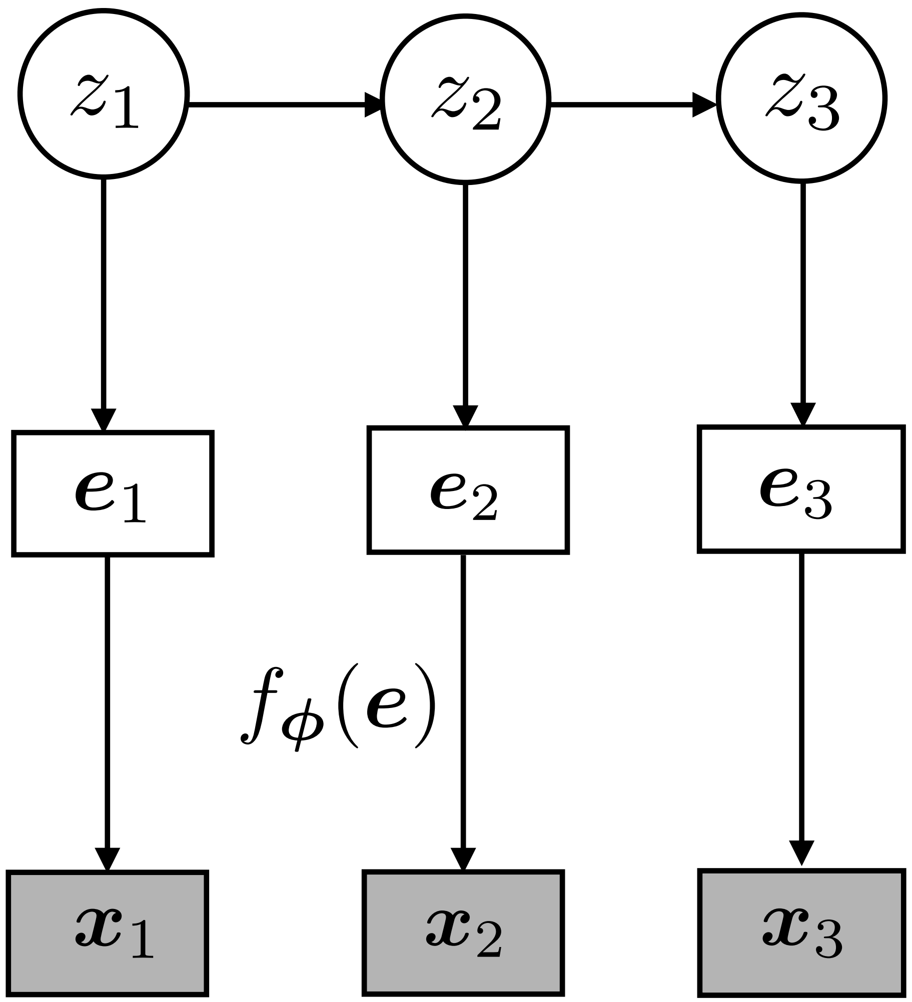
Gaussian HMM for POS Induction



$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \Sigma_{z_i})$$

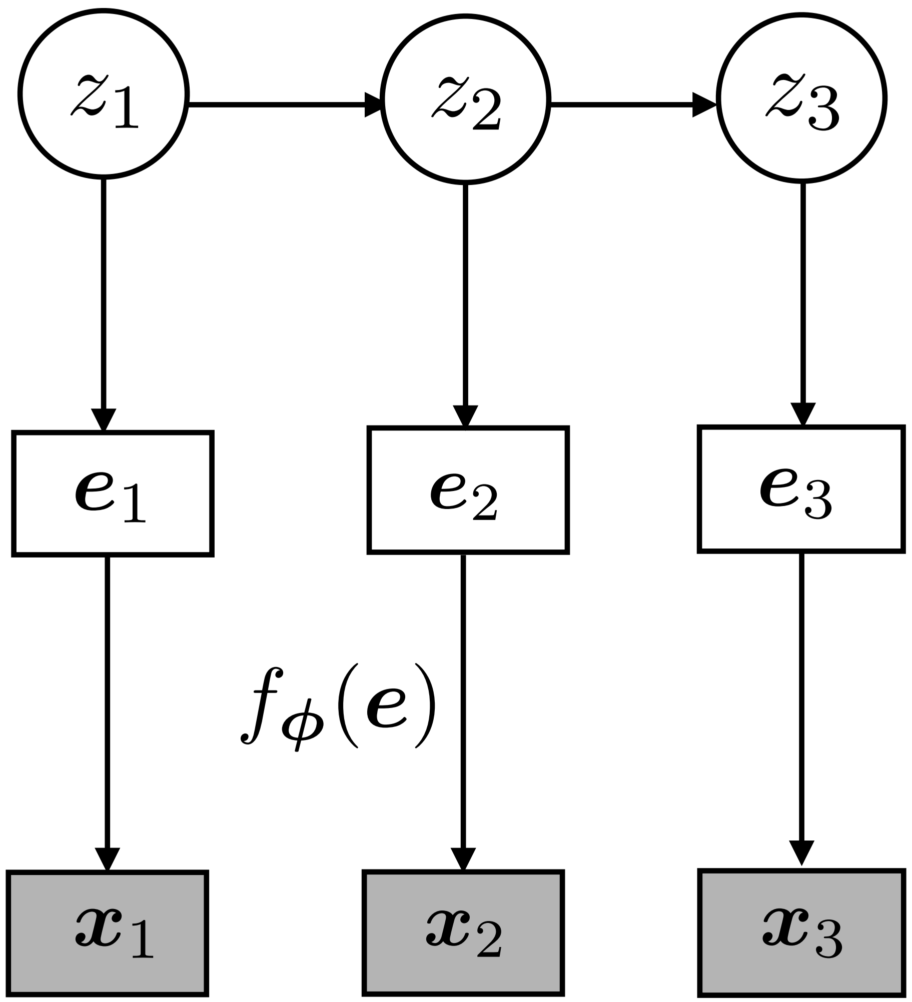
[Lin et al. 2015]

Latent Embeddings w/ Neural Projection



Latent Embeddings w/ Neural Projection

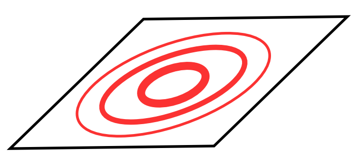
$z_i \sim$ Markov Structure

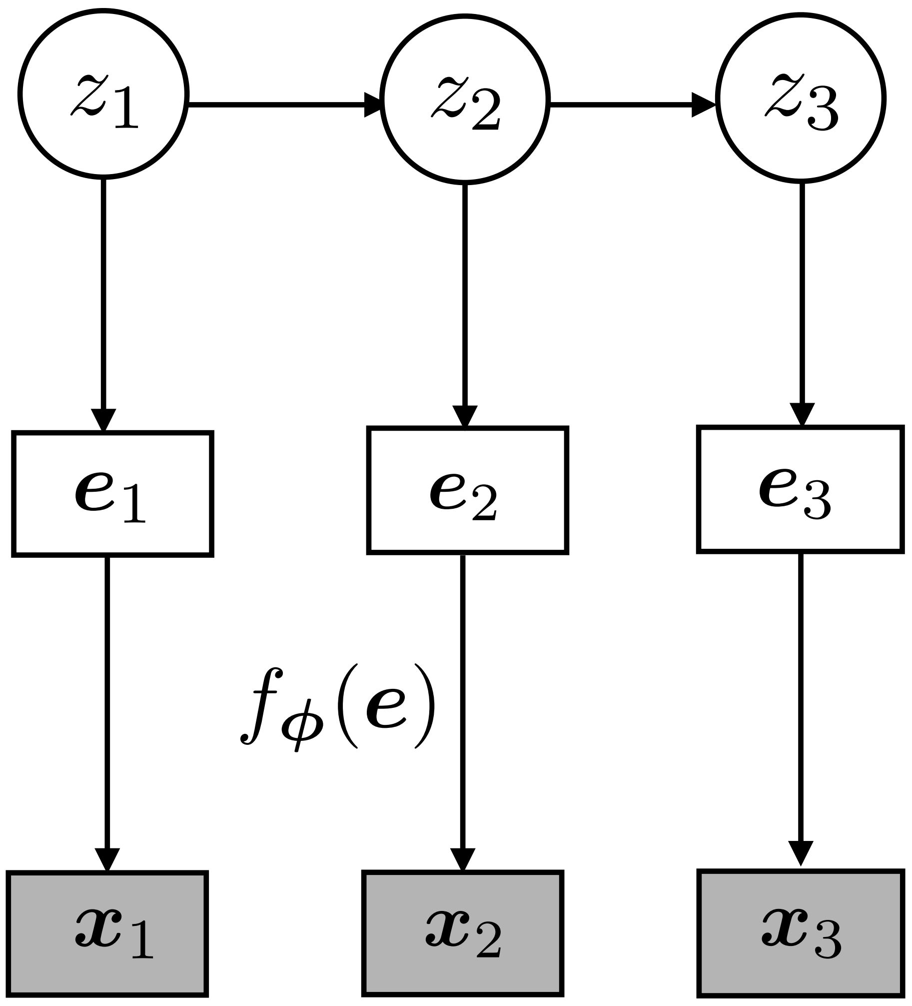


Latent Embeddings w/ Neural Projection

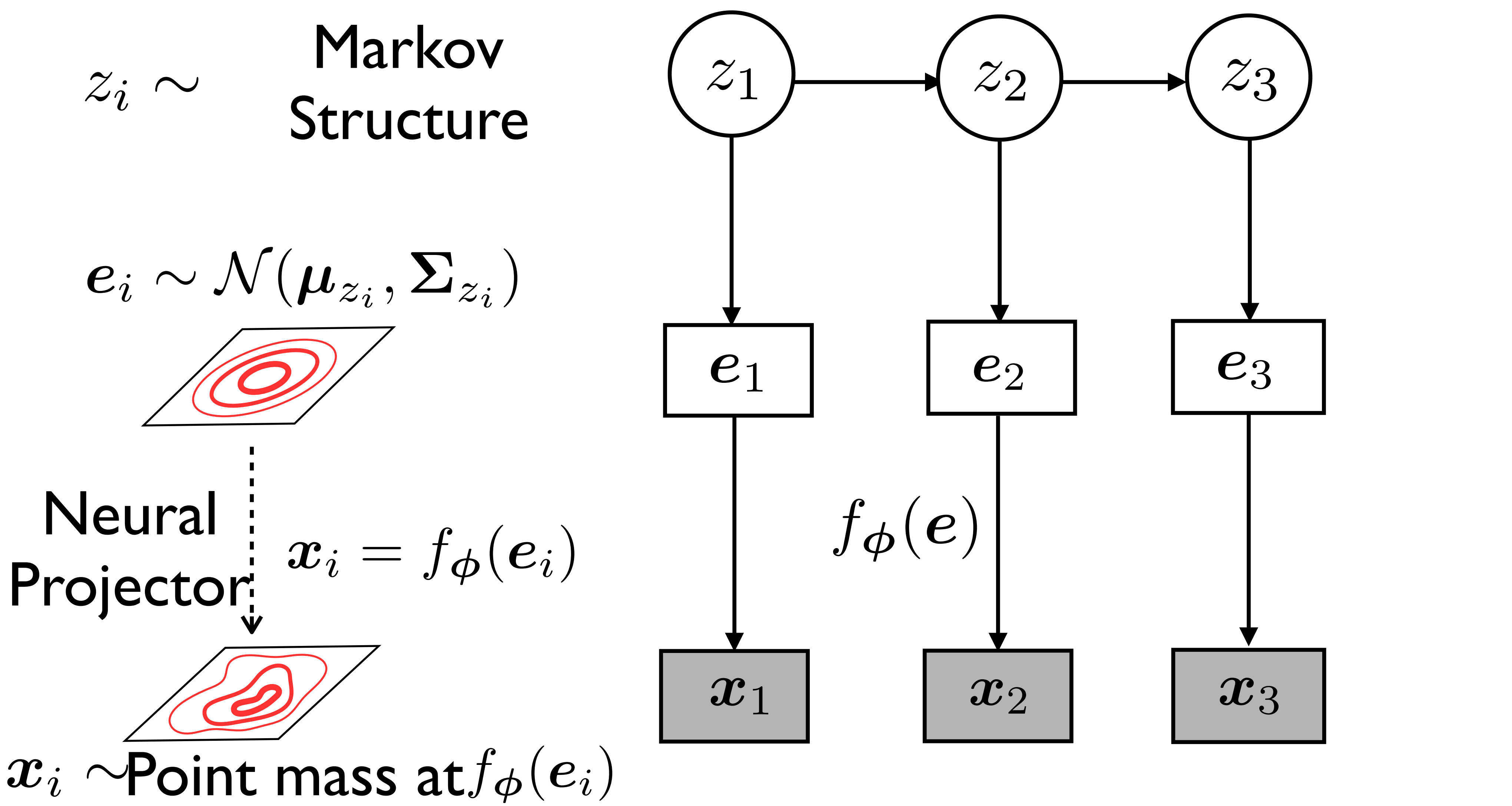
$z_i \sim$ Markov Structure

$\boldsymbol{e}_i \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$





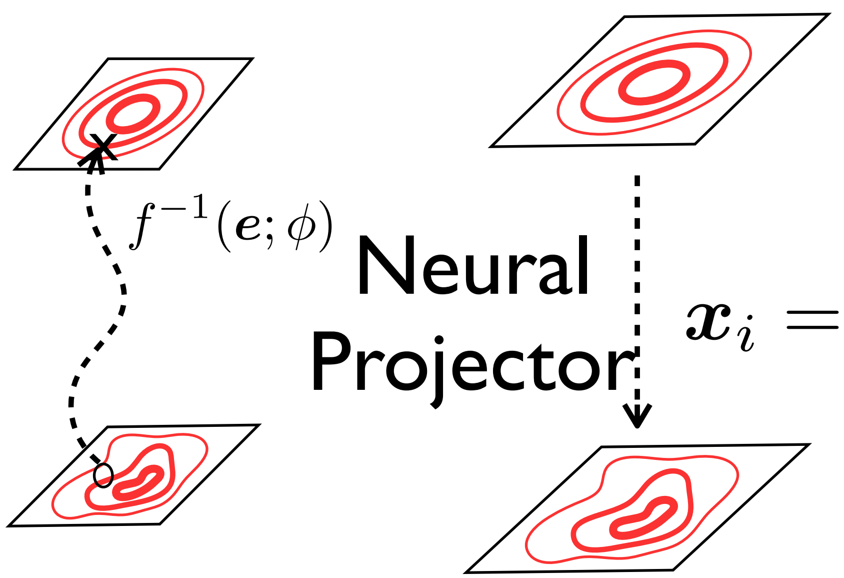
Latent Embeddings w/ Neural Projection



Latent Embeddings w/ Neural Projection

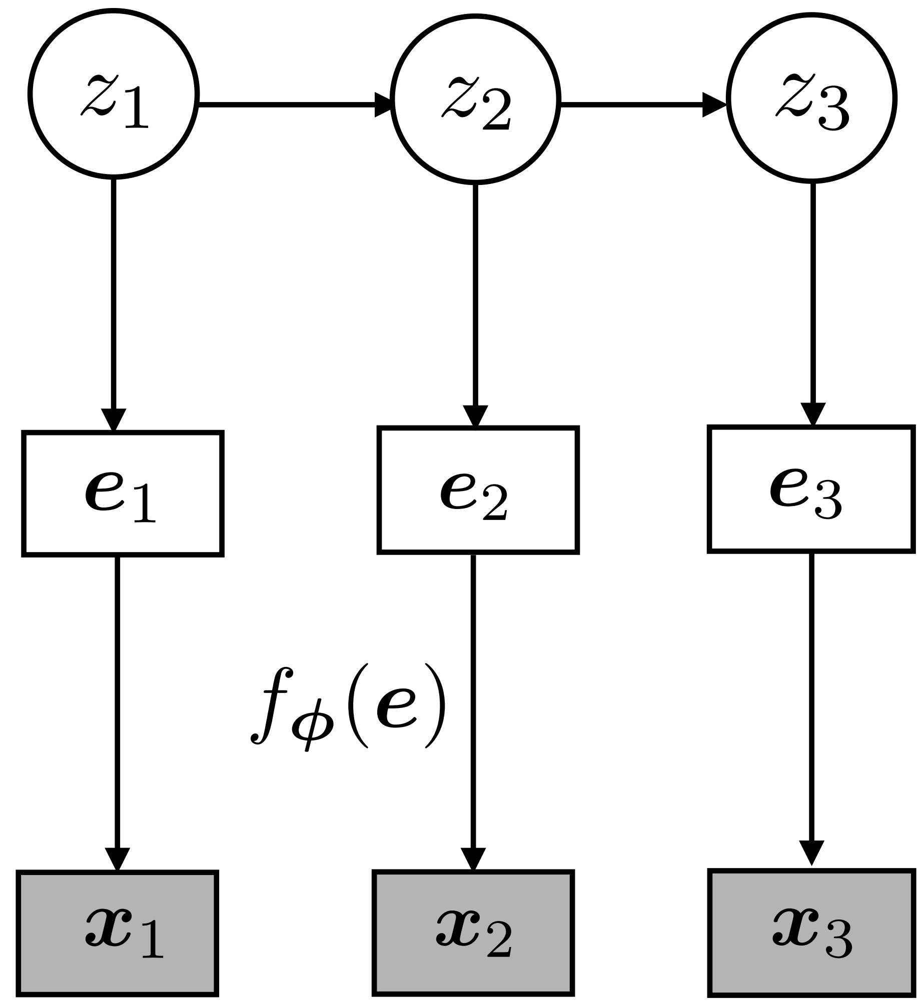
$z_i \sim$ Markov Structure

$\boldsymbol{e}_i \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$

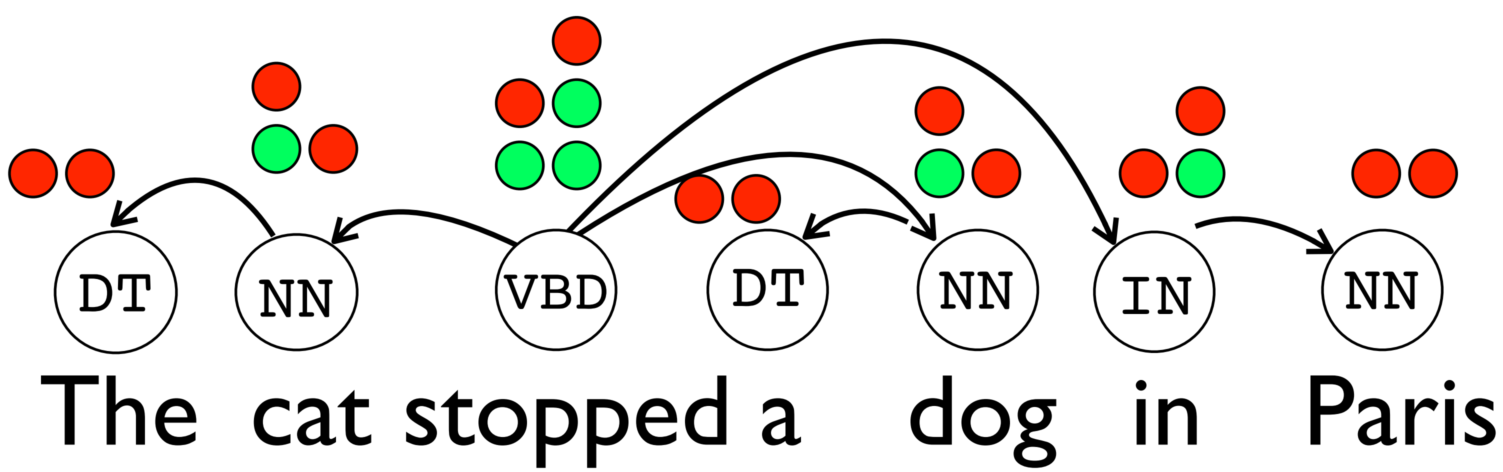


$\boldsymbol{x}_i = f_\phi(\boldsymbol{e}_i)$

$\boldsymbol{x}_i \sim$ Point mass at $f_\phi(\boldsymbol{e}_i)$

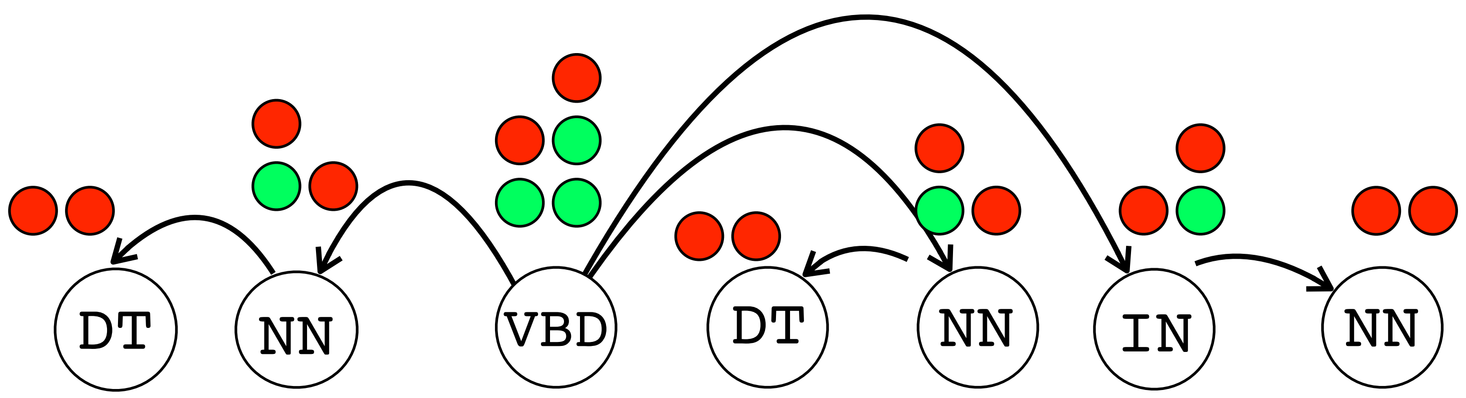


Dependency Model with Valence



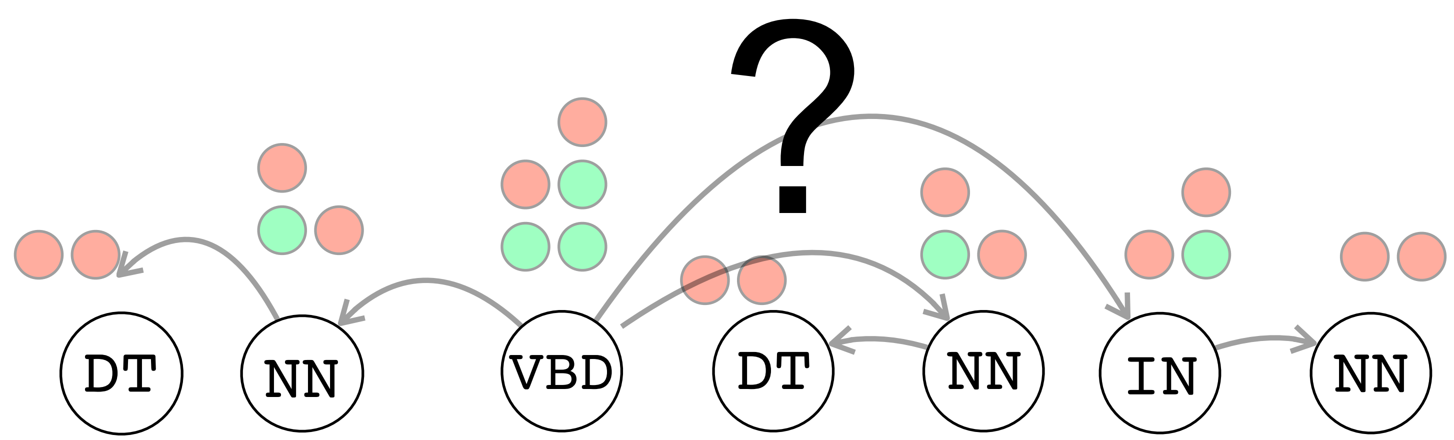
[Klein and Manning 2004]

Dependency Model with Valence

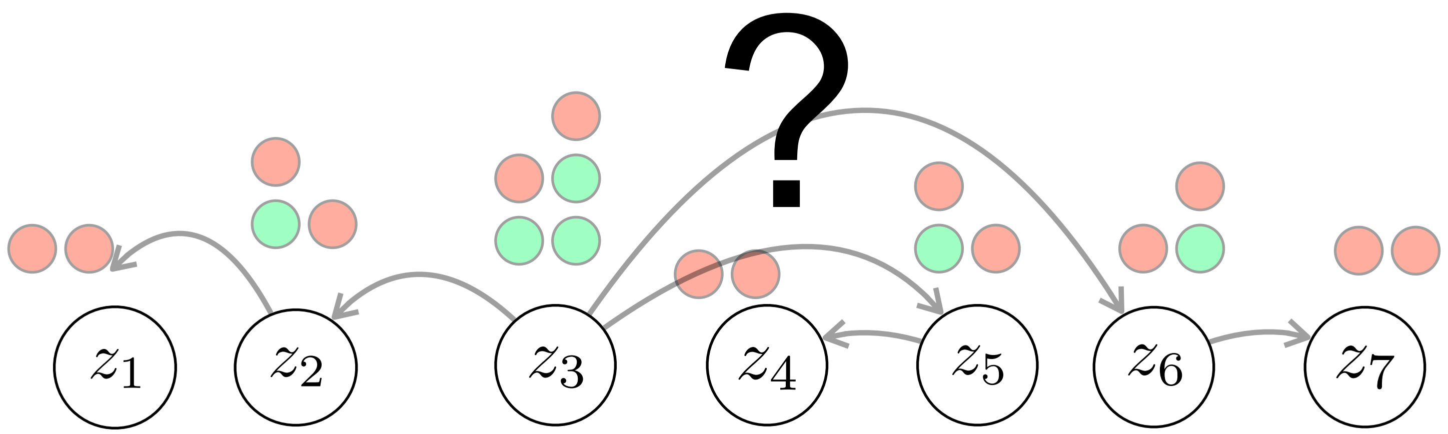


[Klein and Manning 2004]

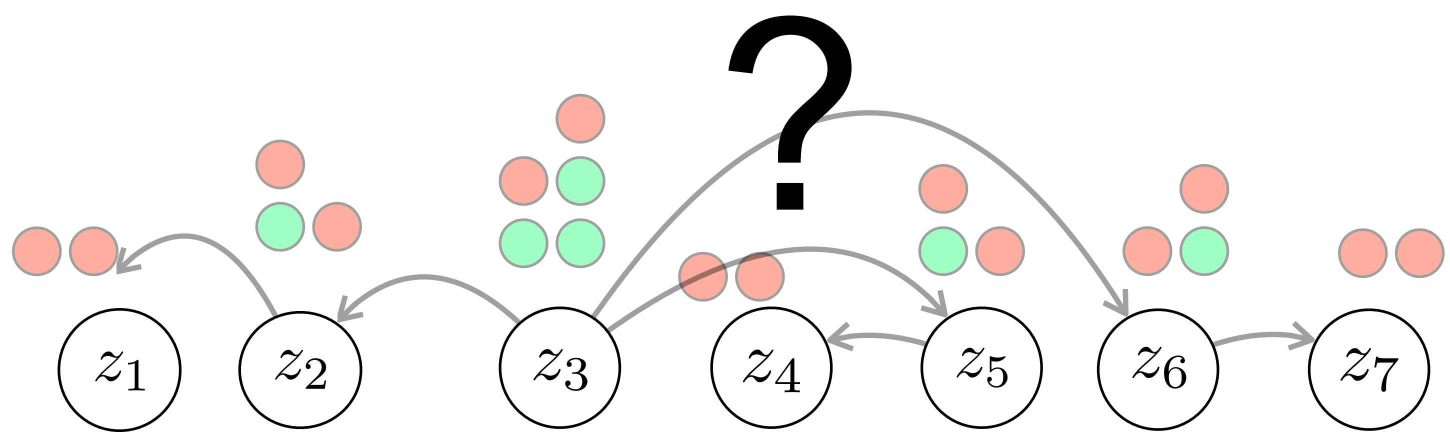
Dependency Parse Induction from POS



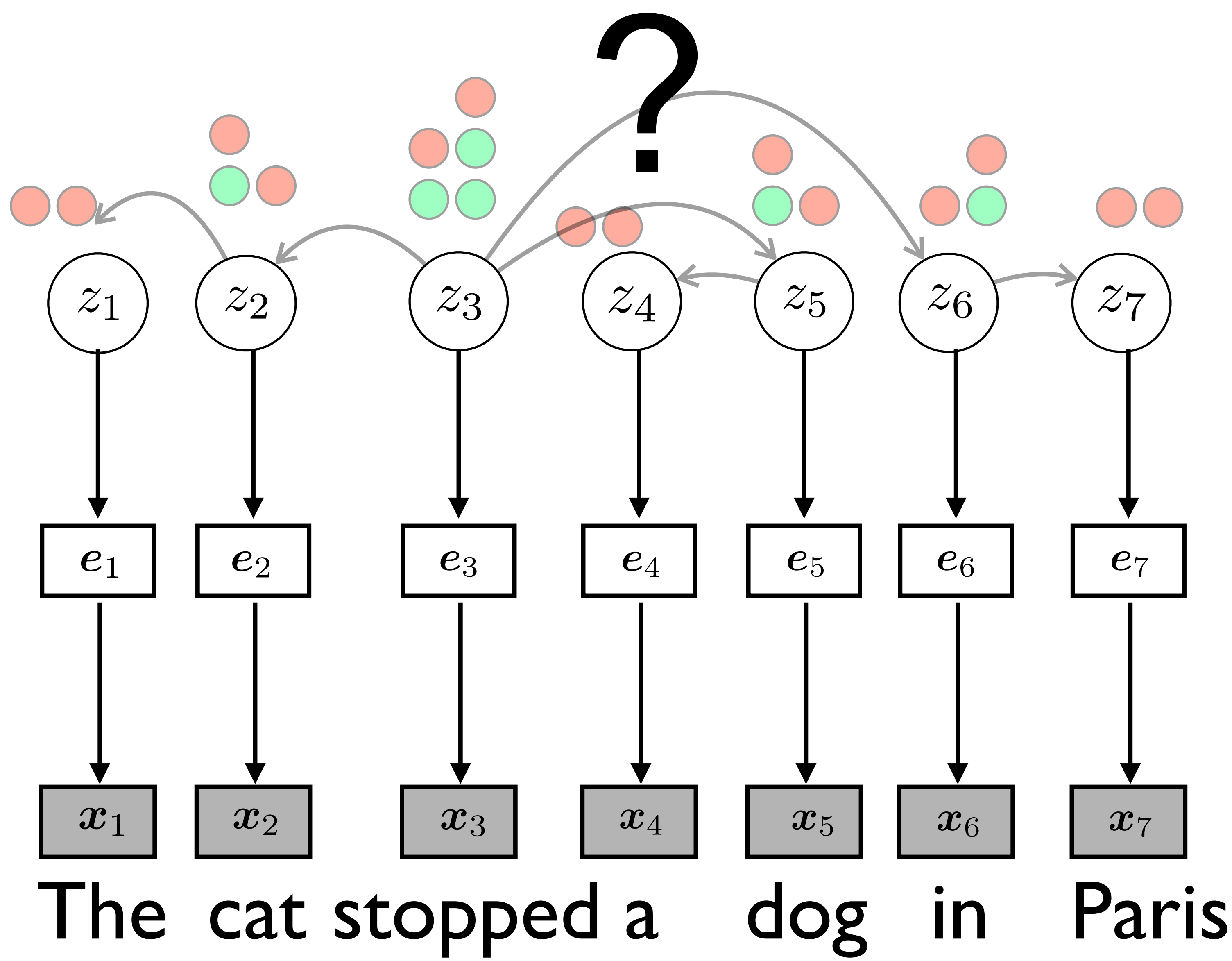
Grammar Induction from Raw Text



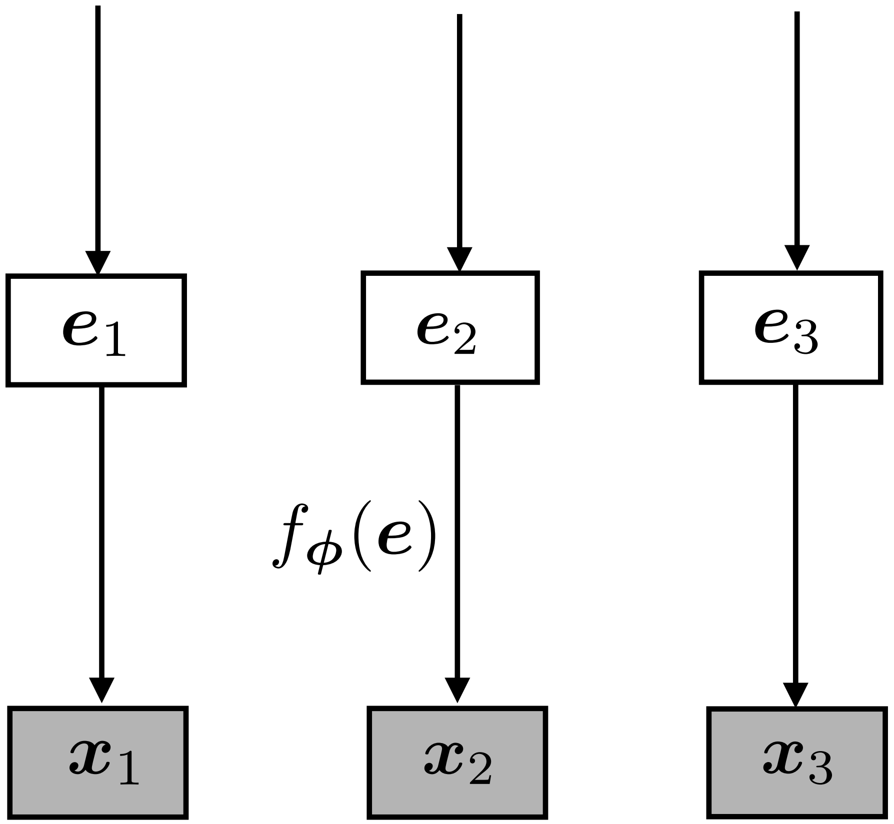
Grammar Induction from Raw Text



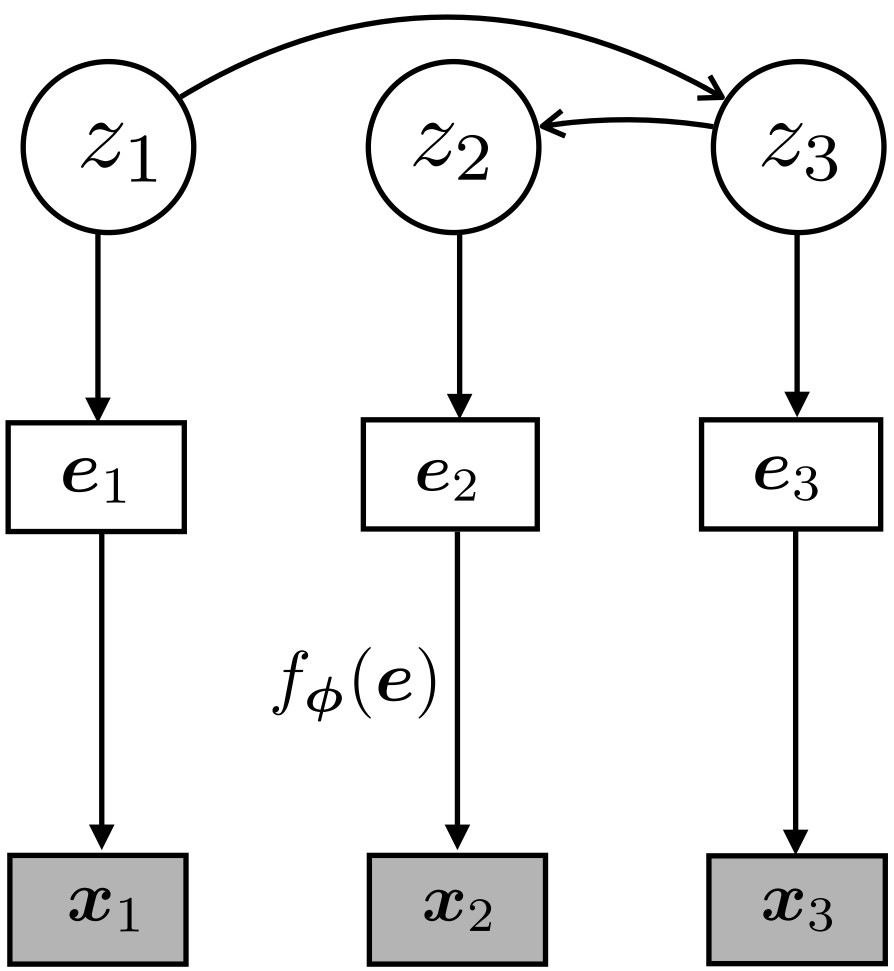
Grammar Induction from Raw Text



Latent Embeddings w/ Neural Projection



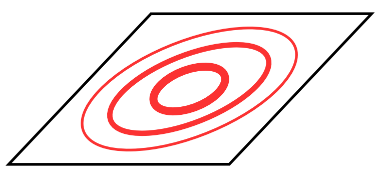
Latent Embeddings w/ Neural Projection



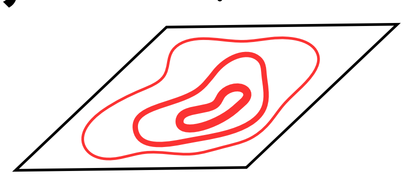
Latent Embeddings w/ Neural Projection

$z_i \sim$ Syntax Model

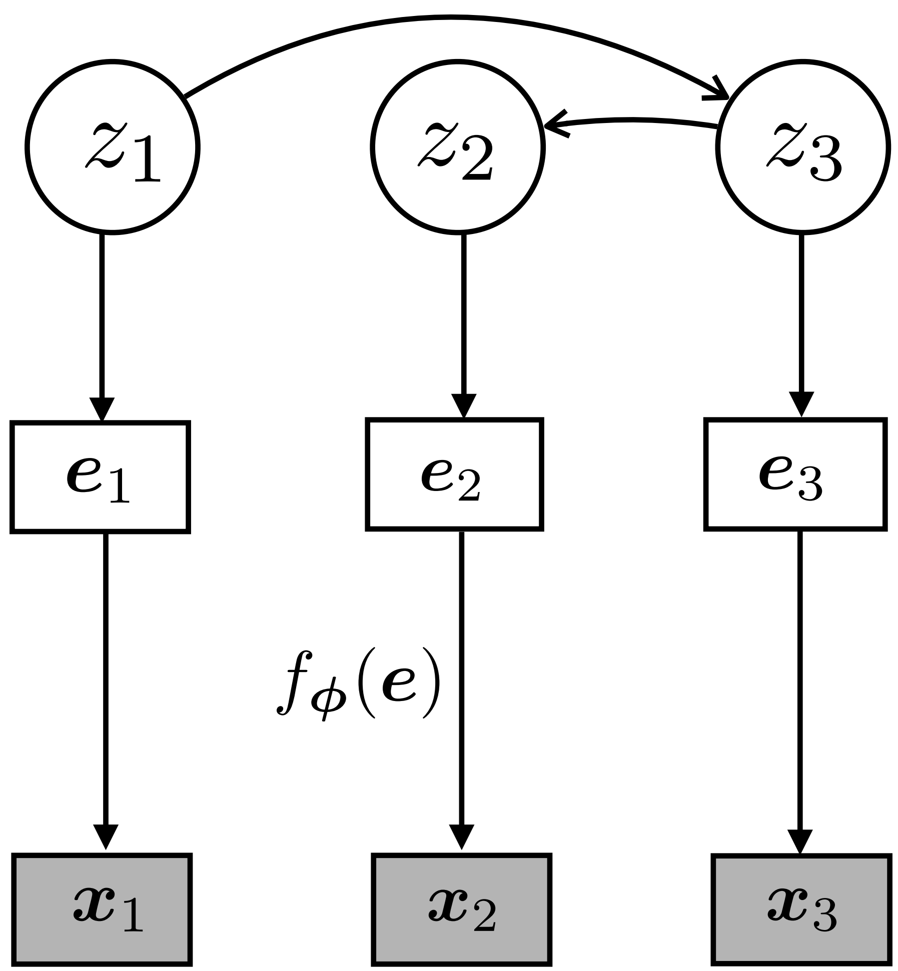
$\boldsymbol{e}_i \sim \mathcal{N}(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$



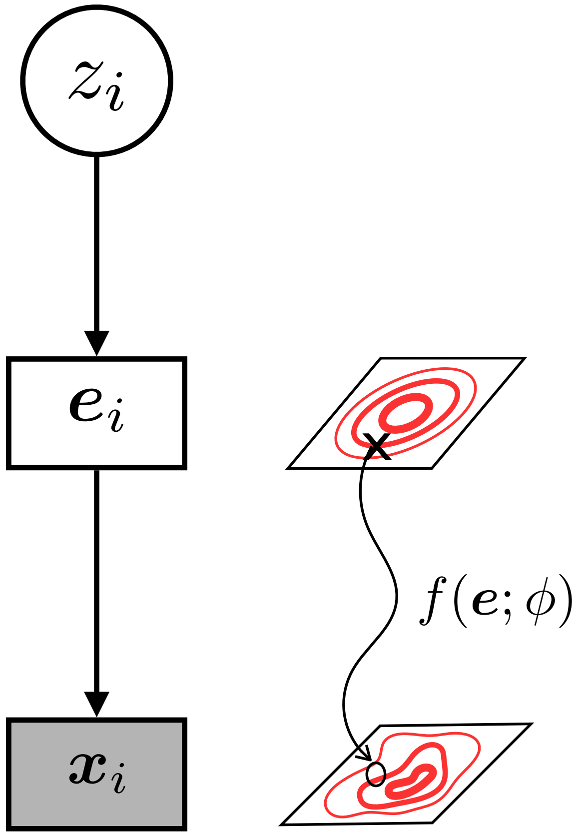
Neural Projector $\boldsymbol{x}_i = f_{\boldsymbol{\phi}}(\boldsymbol{e}_i)$



$\boldsymbol{x}_i \sim$ Point mass at $f_{\phi}(e_i)$

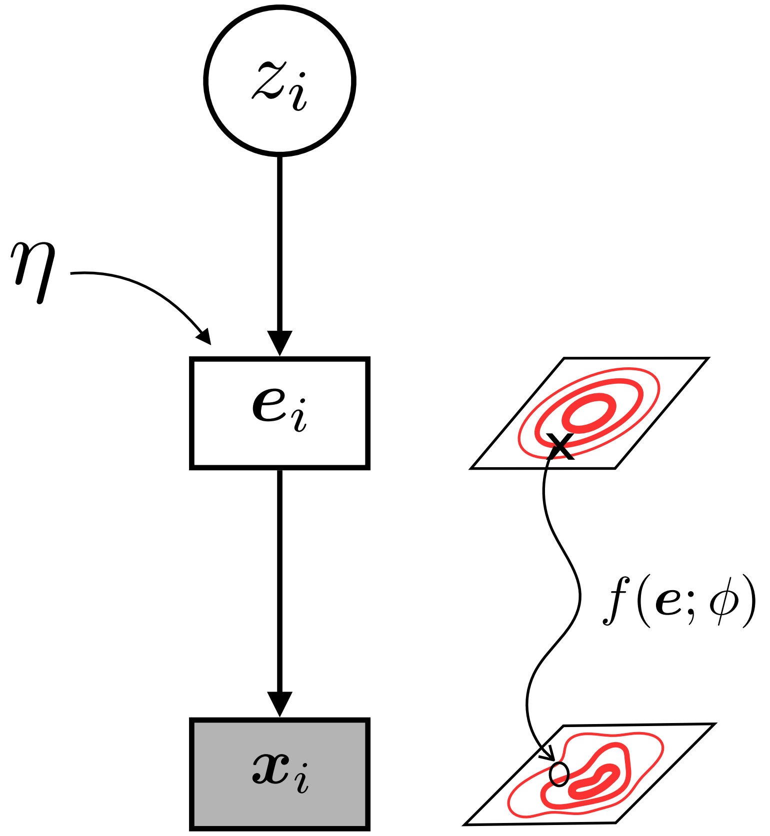


Learning and Inference



$$p(\boldsymbol{x}_i | z_i; \eta, \phi)$$

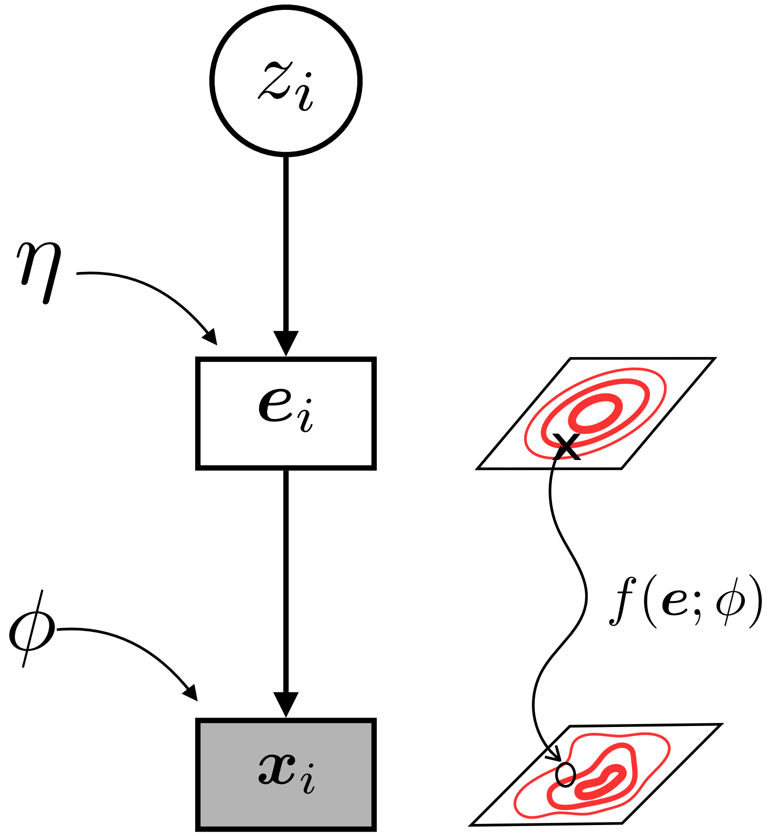
Learning and Inference



$$p(\boldsymbol{x}_i | z_i; \eta, \phi)$$

Gaussian embedding parameters

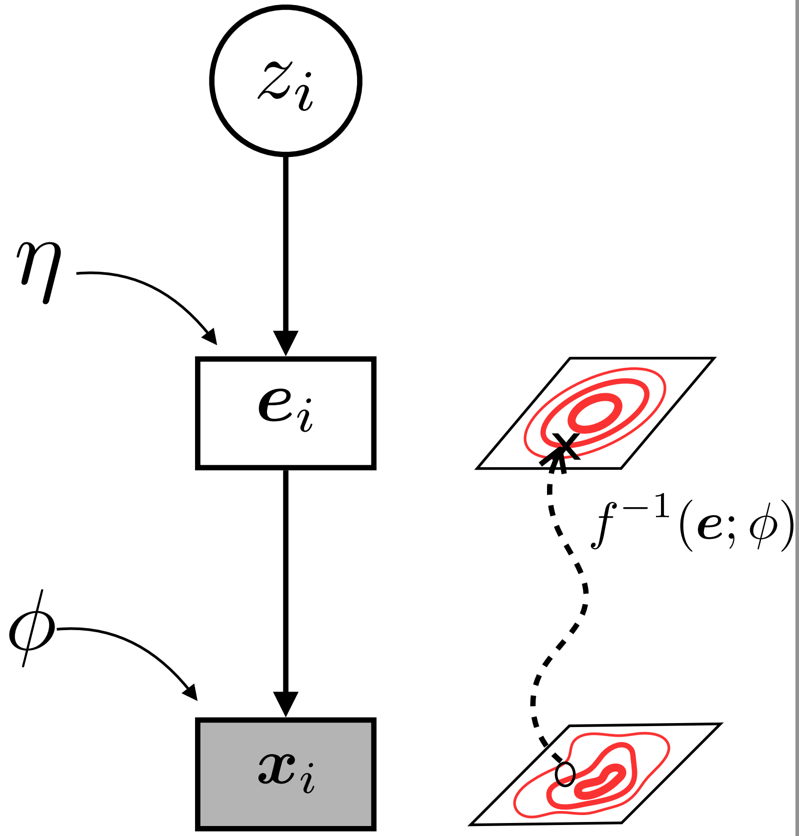
Learning and Inference



$$p(\boldsymbol{x}_i | z_i; \eta, \phi)$$

Projection parameters

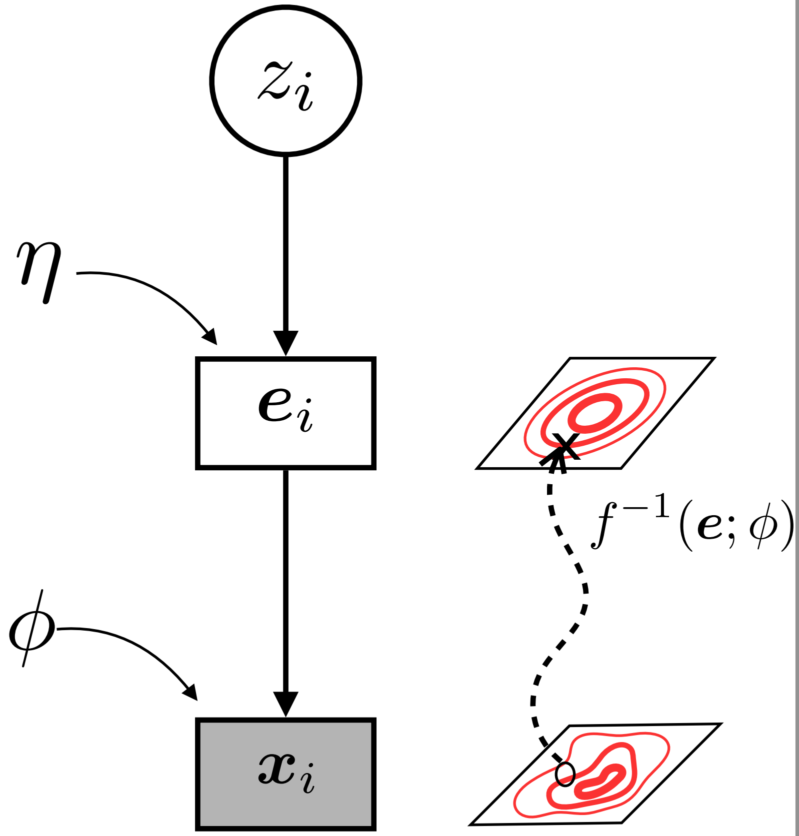
Learning and Inference



dim(x) = dim(e) and f is invertible

$$p(\boldsymbol{x}_i | z_i; \eta, \phi)$$

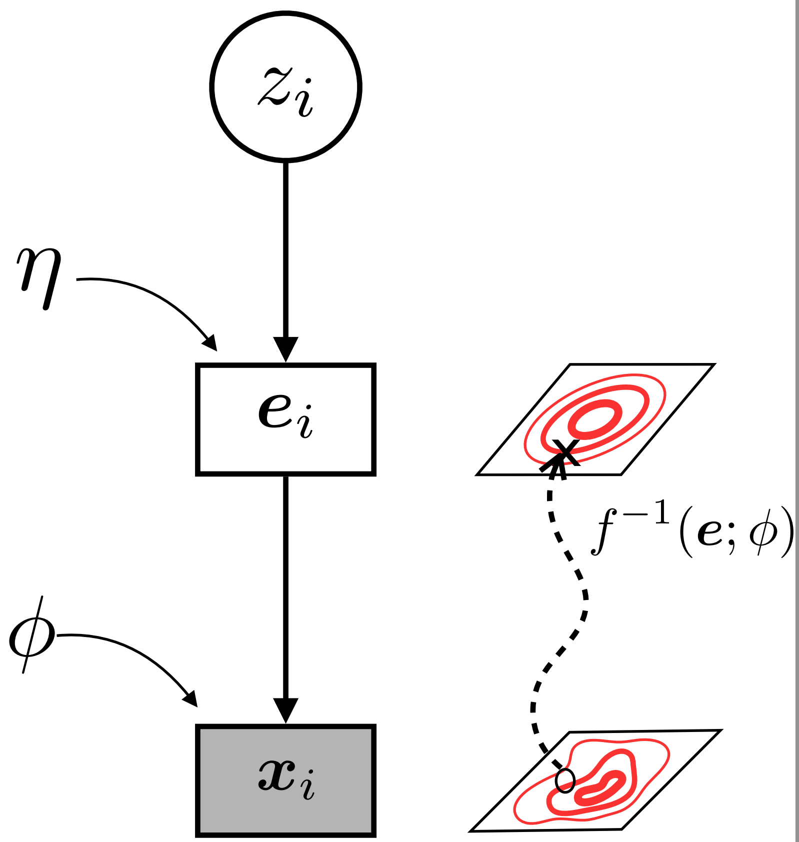
Learning and Inference



dim(x) = dim(e) and f is invertible

$$p(\boldsymbol{x}_i|z_i;\eta,\phi)$$
$$= p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Learning and Inference



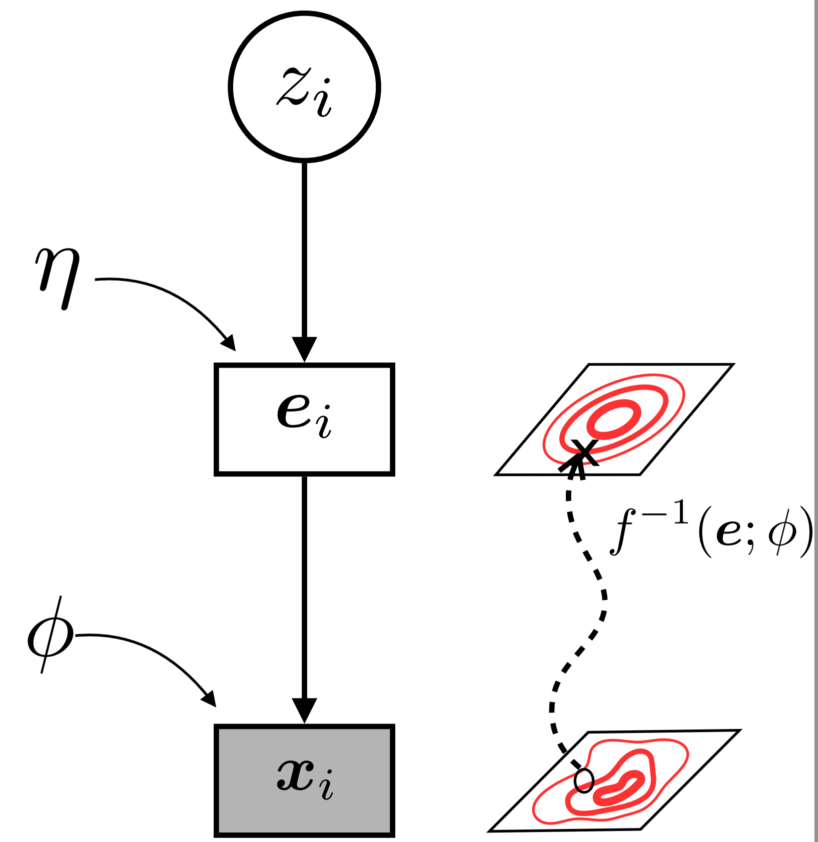
dim(x) = dim(e) and f is invertible

$$p(\boldsymbol{x}_i|z_i;\eta,\phi)$$

$$= p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Determinant of Jacobian matrix

Learning and Inference



dim(x) = dim(e) and f is invertible

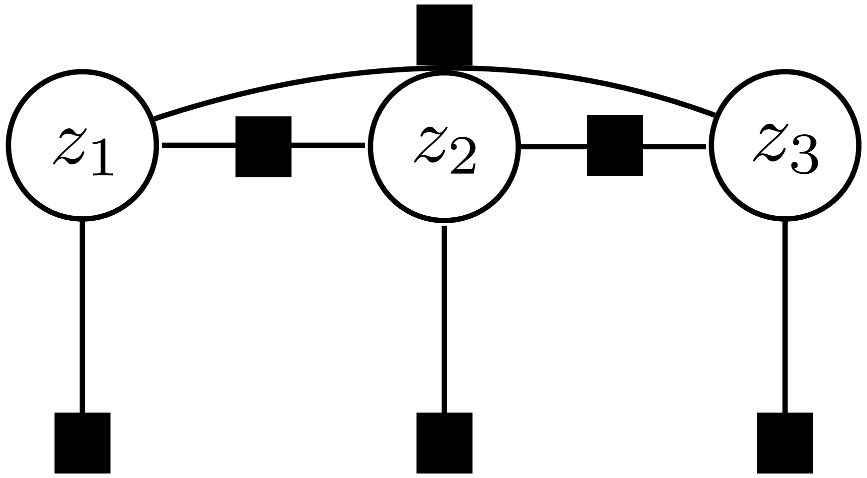
$$p(\boldsymbol{x}_i | z_i; \eta, \phi)$$

$$= p(f_\phi^{-1}(\boldsymbol{x}_i) | z_i; \eta) \left| \det \frac{\partial f^{-1}}{\partial \boldsymbol{x}_i} \right|$$

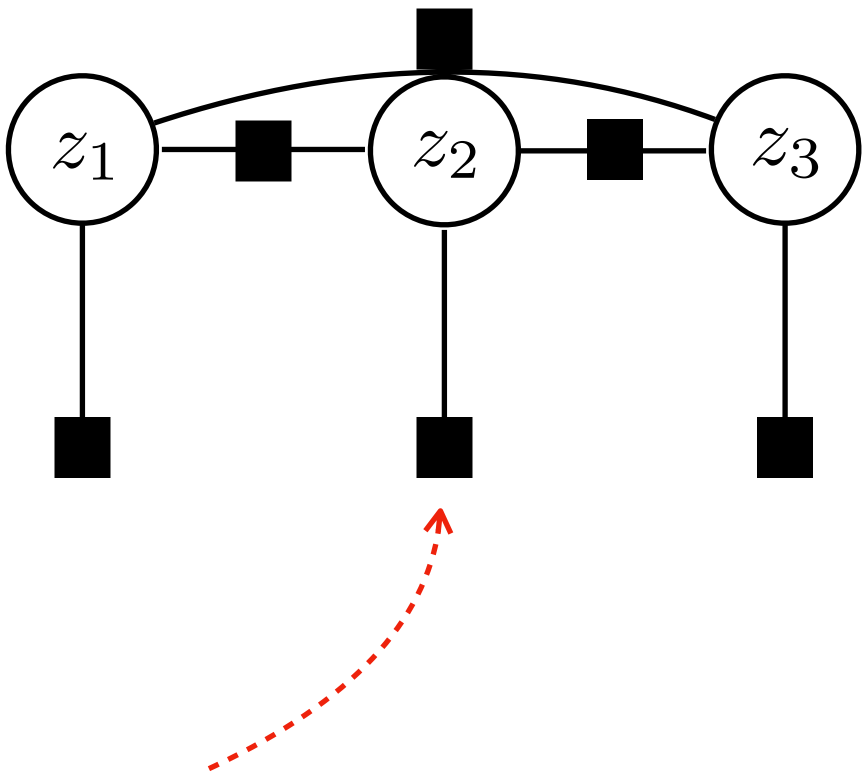
Gaussian distribution

Determinant of Jacobian matrix

Learning and Inference

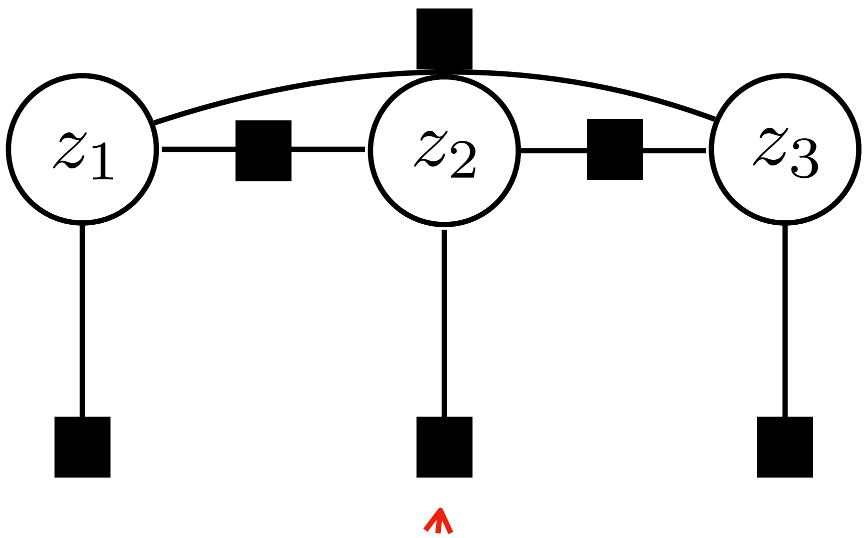


Learning and Inference



$$p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Learning and Inference

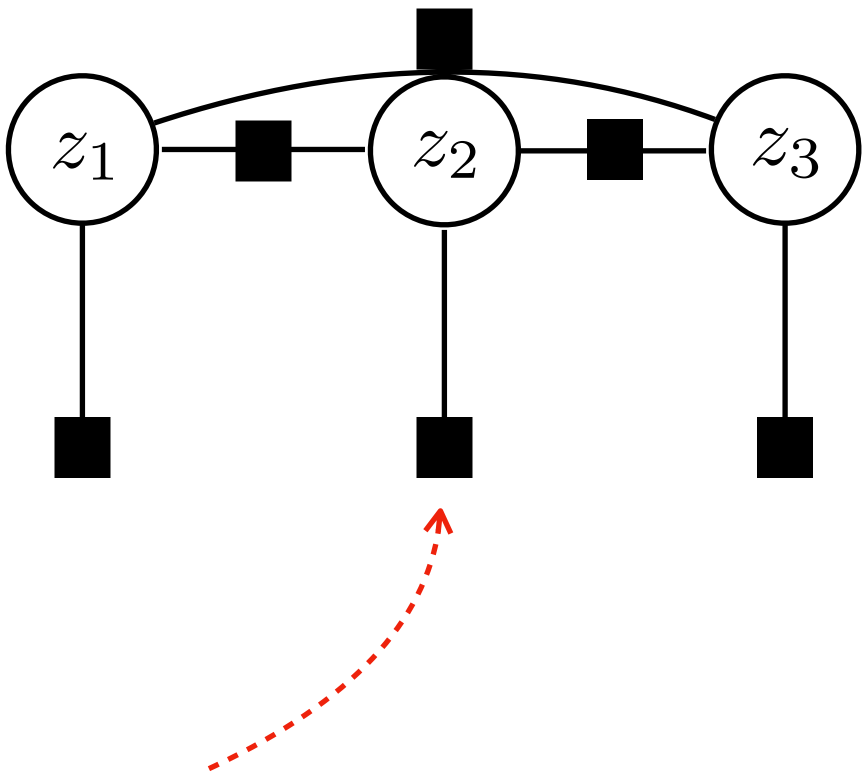


$$p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det \frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Example of Markov prior

$$\log p(\boldsymbol{x}) = \log p_{\text{GHMM}}(f_\phi^{-1}(\boldsymbol{x}))$$

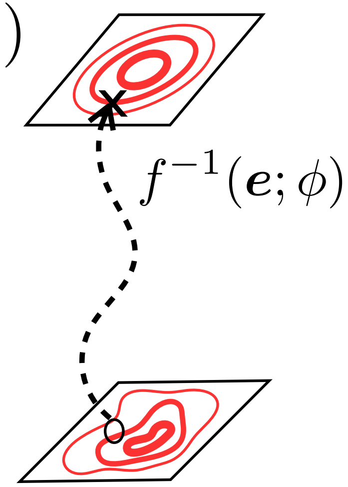
Learning and Inference



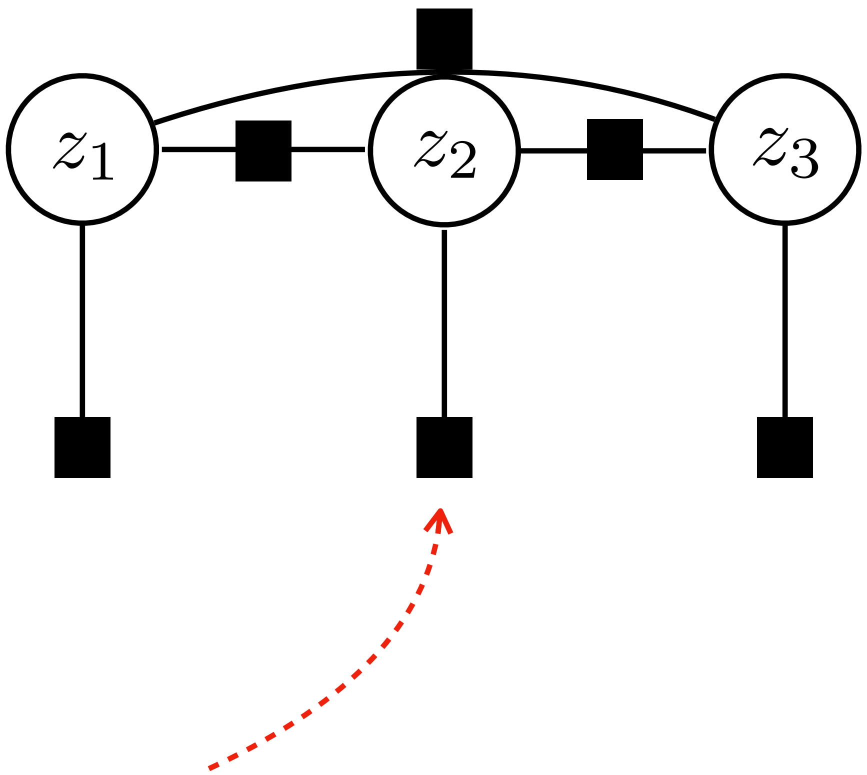
$$p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Example of Markov prior

$$\log p(\boldsymbol{x}) = \log p_{\mathrm{GHMM}}(f_\phi^{-1}(\boldsymbol{x}))$$



Learning and Inference

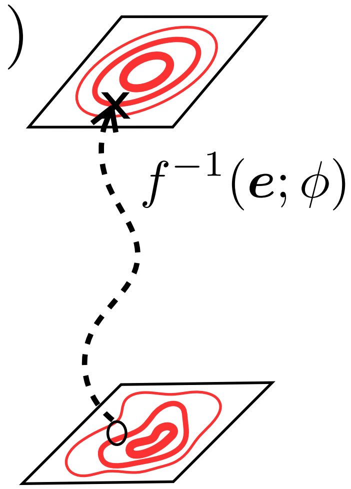


$$p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

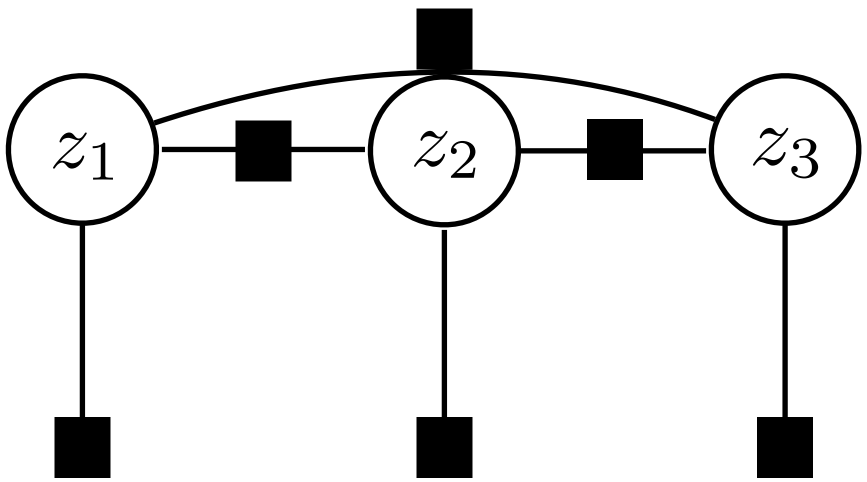
Example of Markov prior

$$\log p(\boldsymbol{x}) = \log p_{\mathrm{GHMM}}(f_\phi^{-1}(\boldsymbol{x}))$$

$$+\sum \log\left|\det\frac{\partial f_\phi^{-1}}{\partial \boldsymbol{x}_i}\right|$$



Learning and Inference



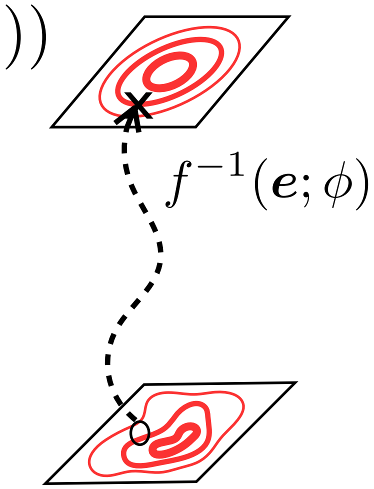
$$p(f_\phi^{-1}(\boldsymbol{x}_i)|z_i;\eta)\left|\det\frac{\partial f^{-1}}{\partial \boldsymbol{x}_i}\right|$$

Example of Markov prior

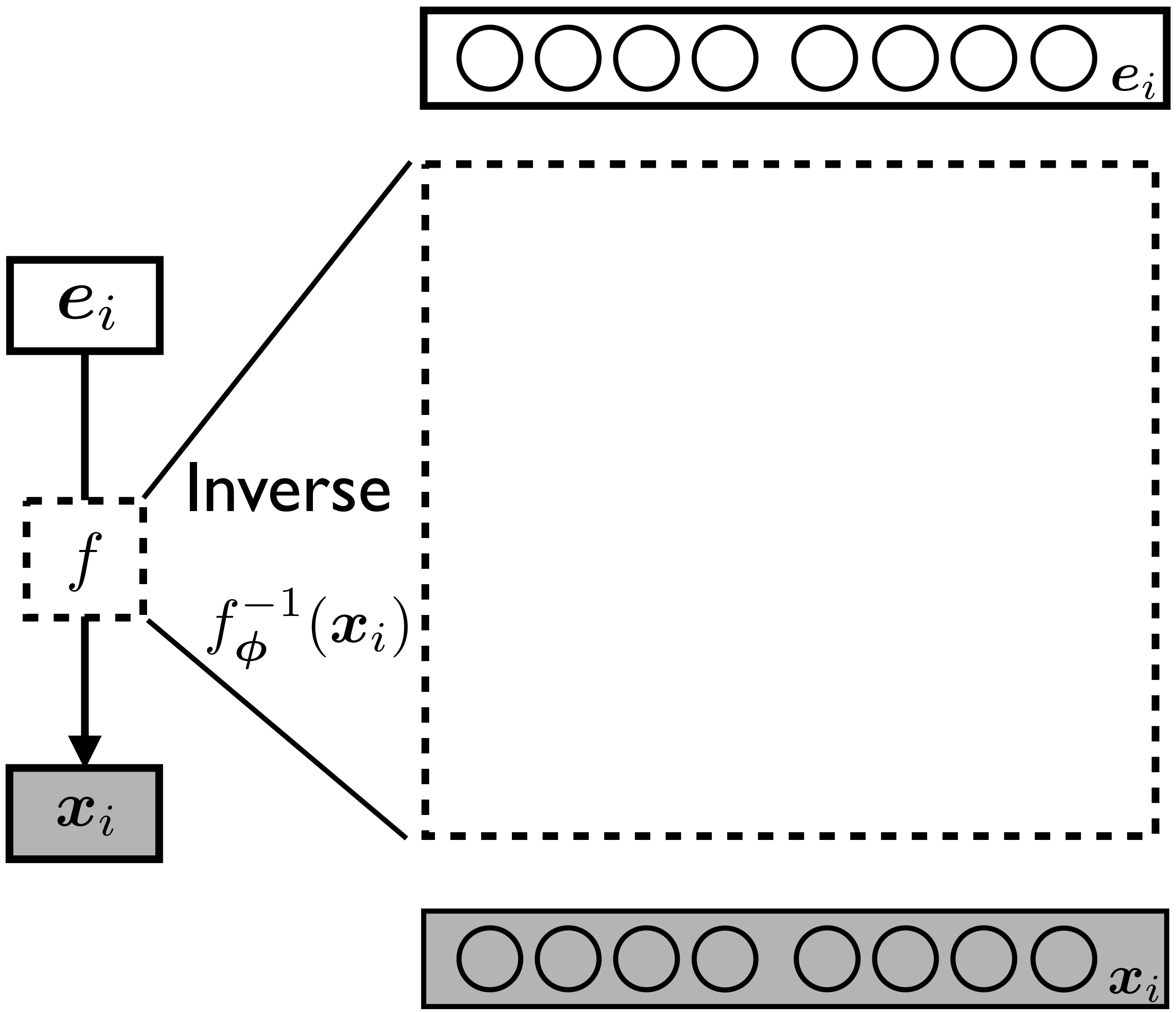
$$\log p(\boldsymbol{x}) = \log p_{\mathrm{GHMM}}(f_\phi^{-1}(\boldsymbol{x}))$$

$$+\sum \log \left|\det \frac{\partial f_\phi^{-1}}{\partial \boldsymbol{x}_i}\right|$$

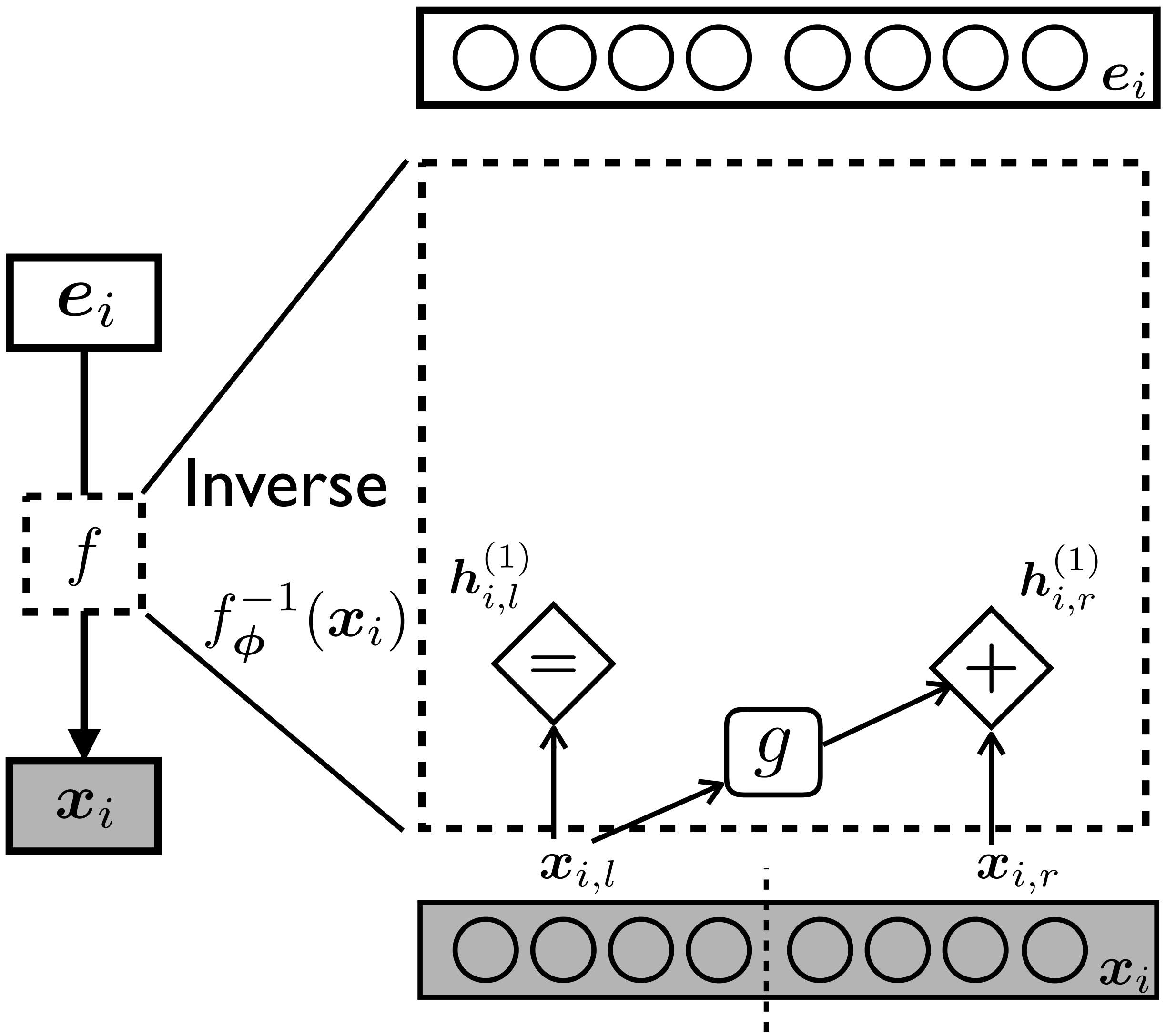
$-\infty$ when f is not invertible



Learning with Inverse Projection



Learning with Inverse Projection

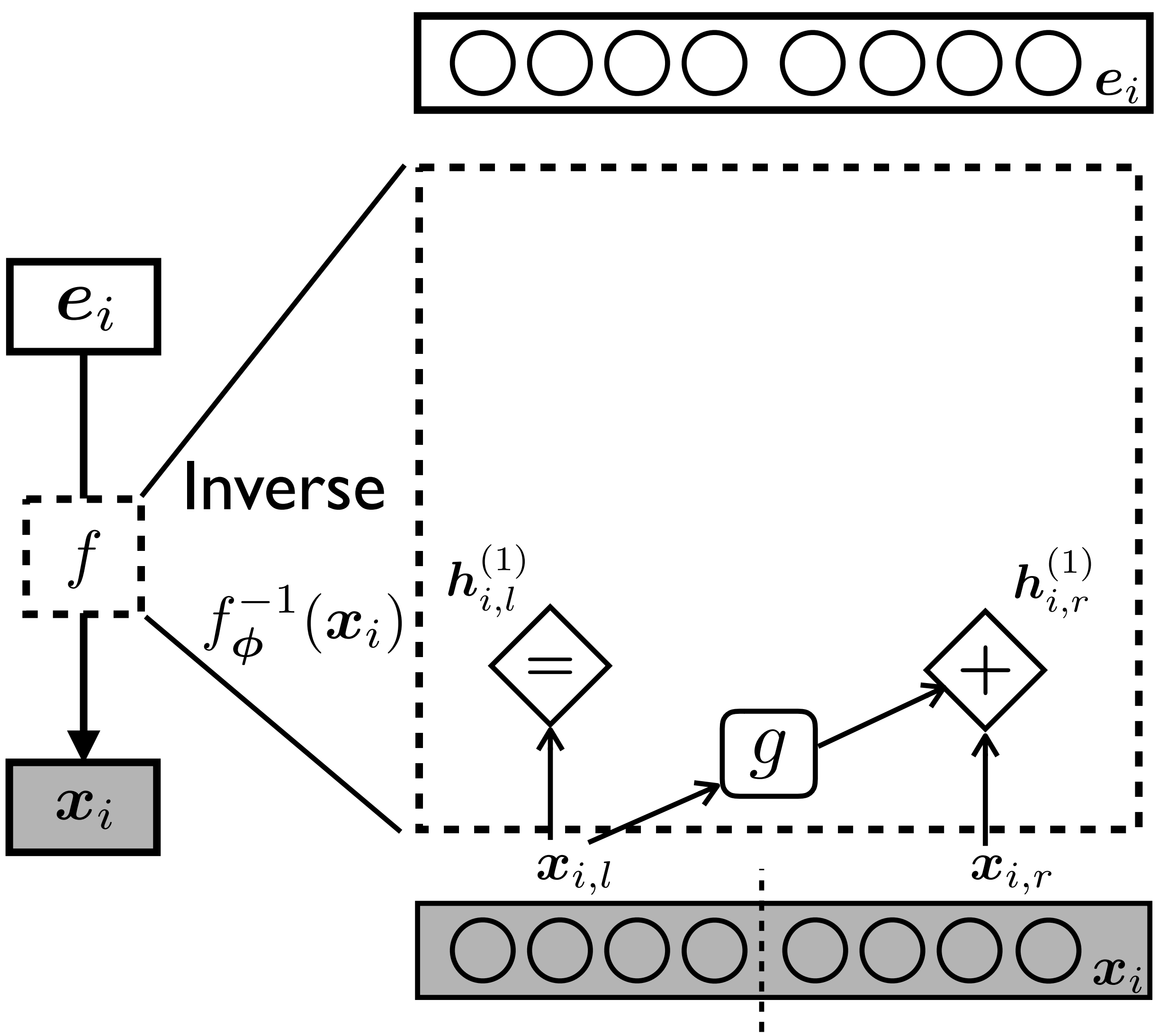


$$h_{i,l}^{(1)} = x_{i,l}$$

$$h_{i,r}^{(1)} = x_{i,r} + g(x_{i,l})$$

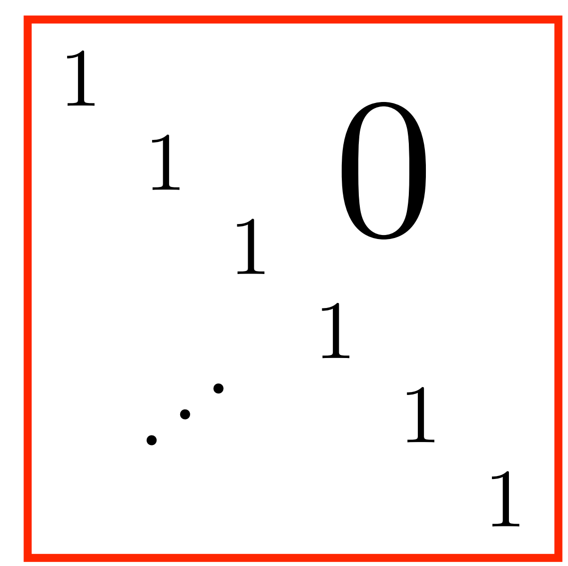
[Dinh et al. 2014]

Learning with Inverse Projection



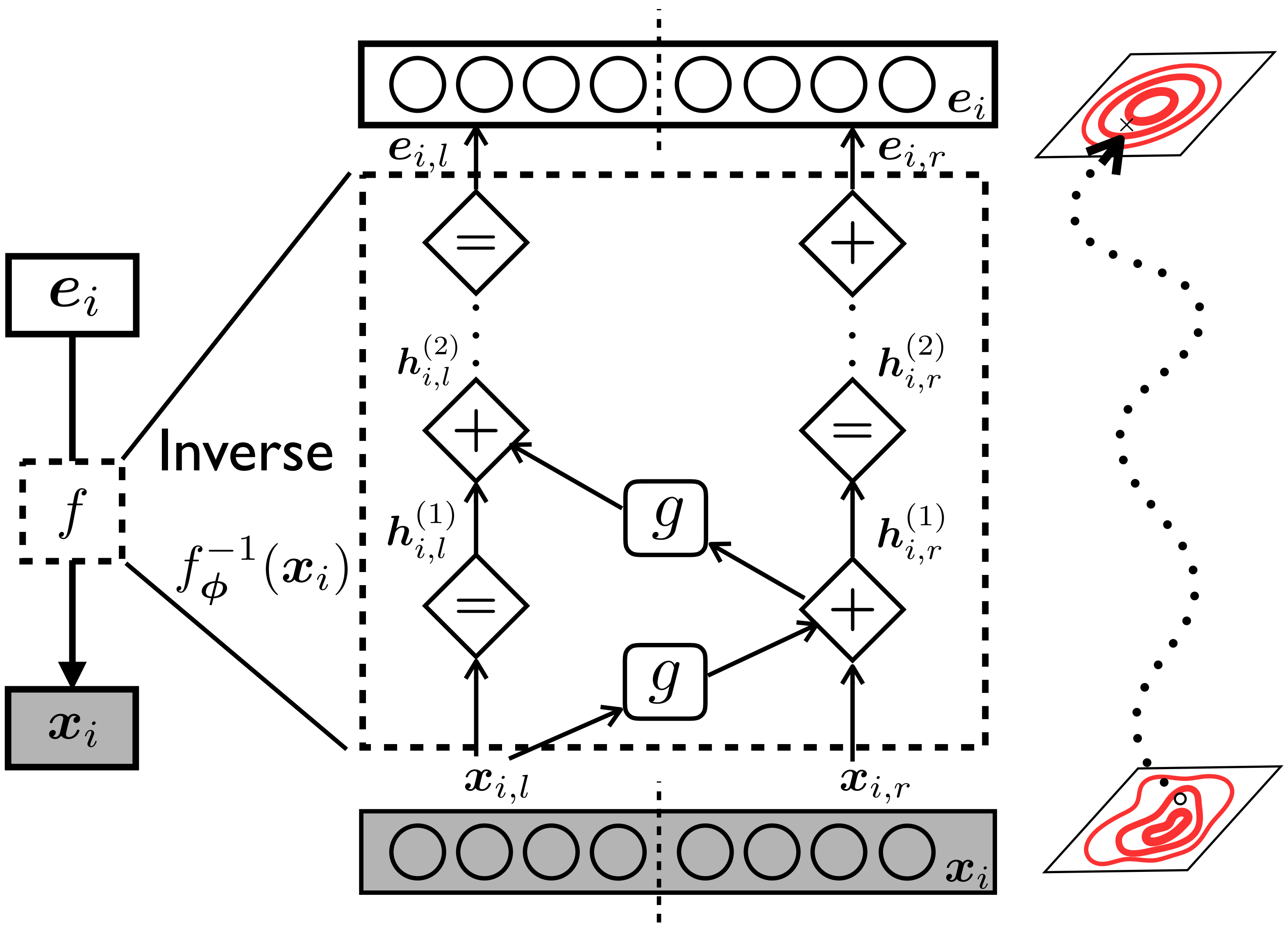
$$h_{i,l}^{(1)} = x_{i,l}$$

$$h_{i,r}^{(1)} = x_{i,r} + g(x_{i,l})$$



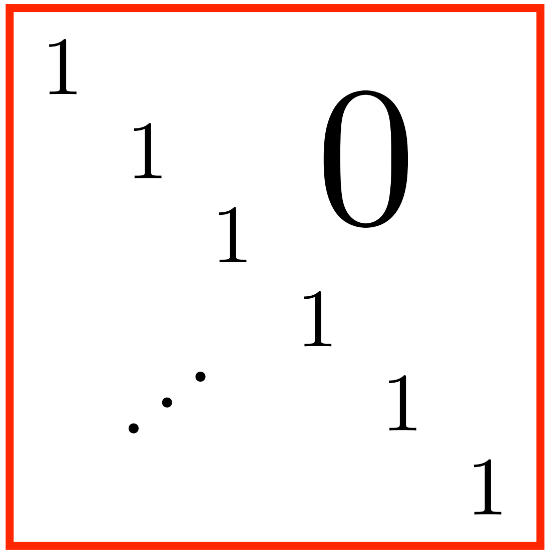
[Dinh et al. 2014]

Learning with Inverse Projection



$$h_{i,l}^{(1)} = x_{i,l}$$

$$h_{i,r}^{(1)} = x_{i,r} + g(x_{i,l})$$



[Dinh et al. 2014]

Experiments

- Dataset: English Penn Treebank
- POS tagging

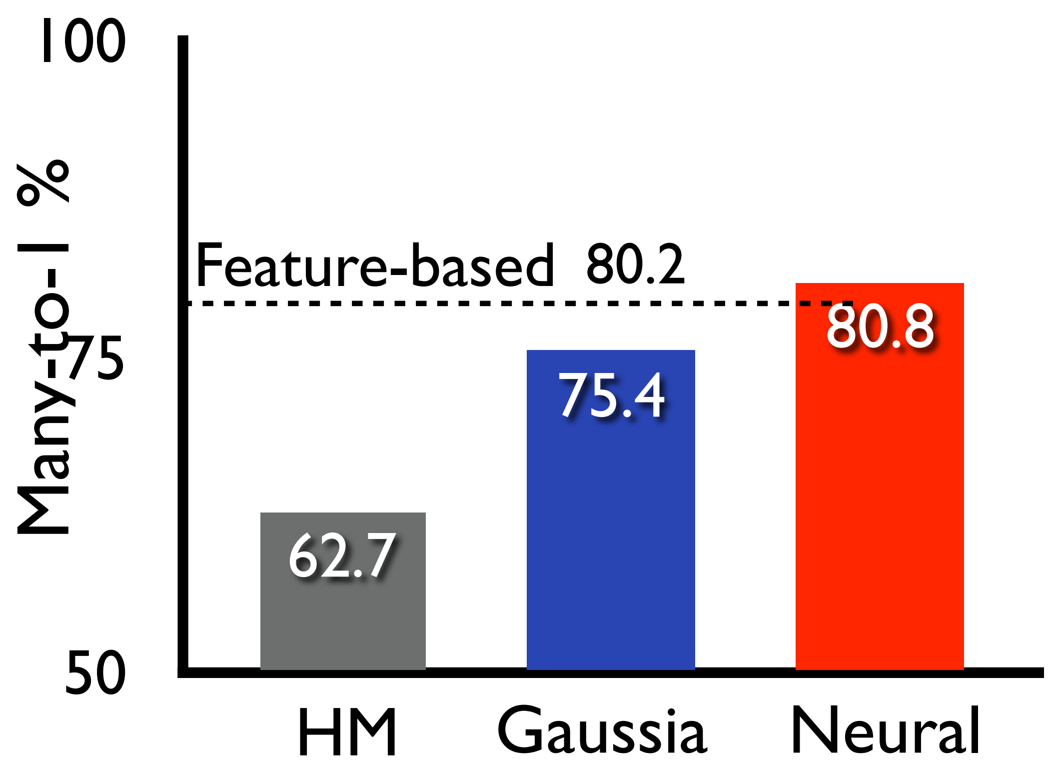
  Trained and tested on whole PTB

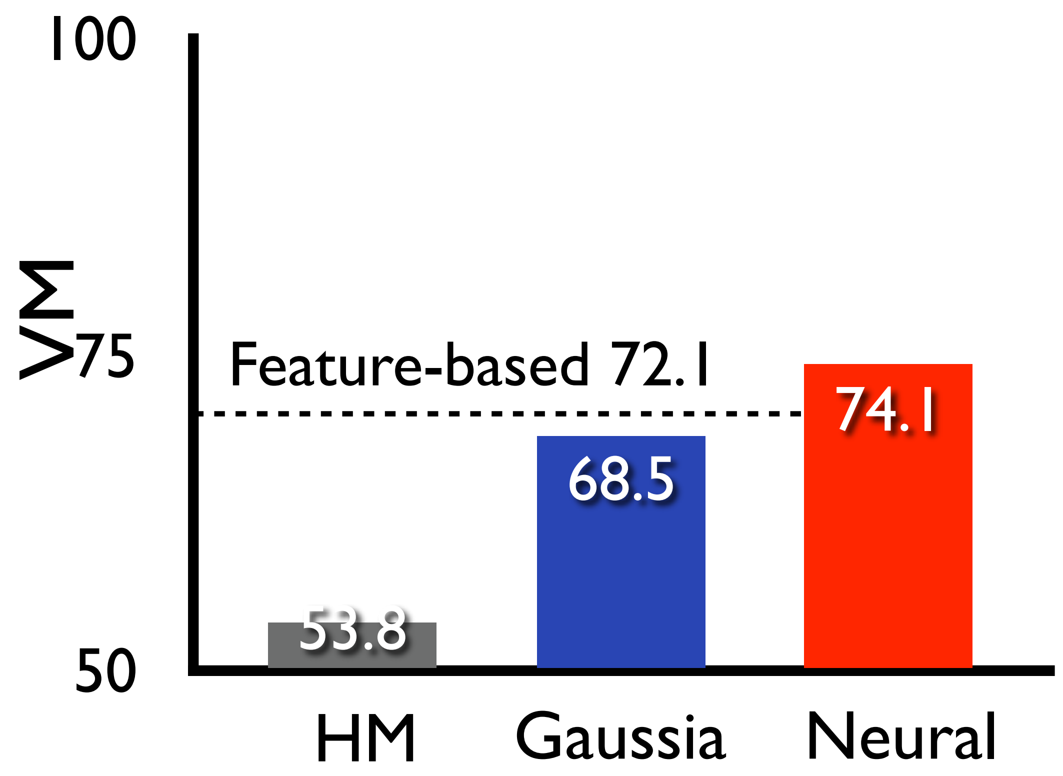
- Grammar induction

  Trained on sentences of length <= 10 in section 2-21

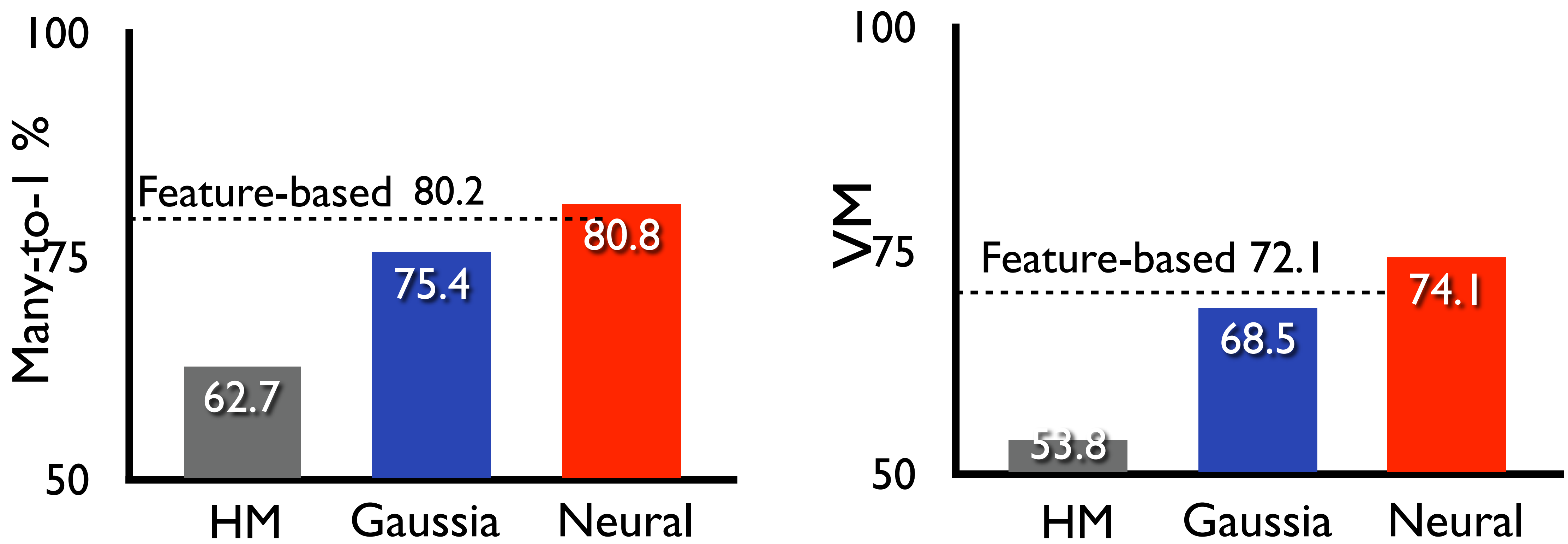
  Tested on sentences in section 23

Part-of-speech Induction



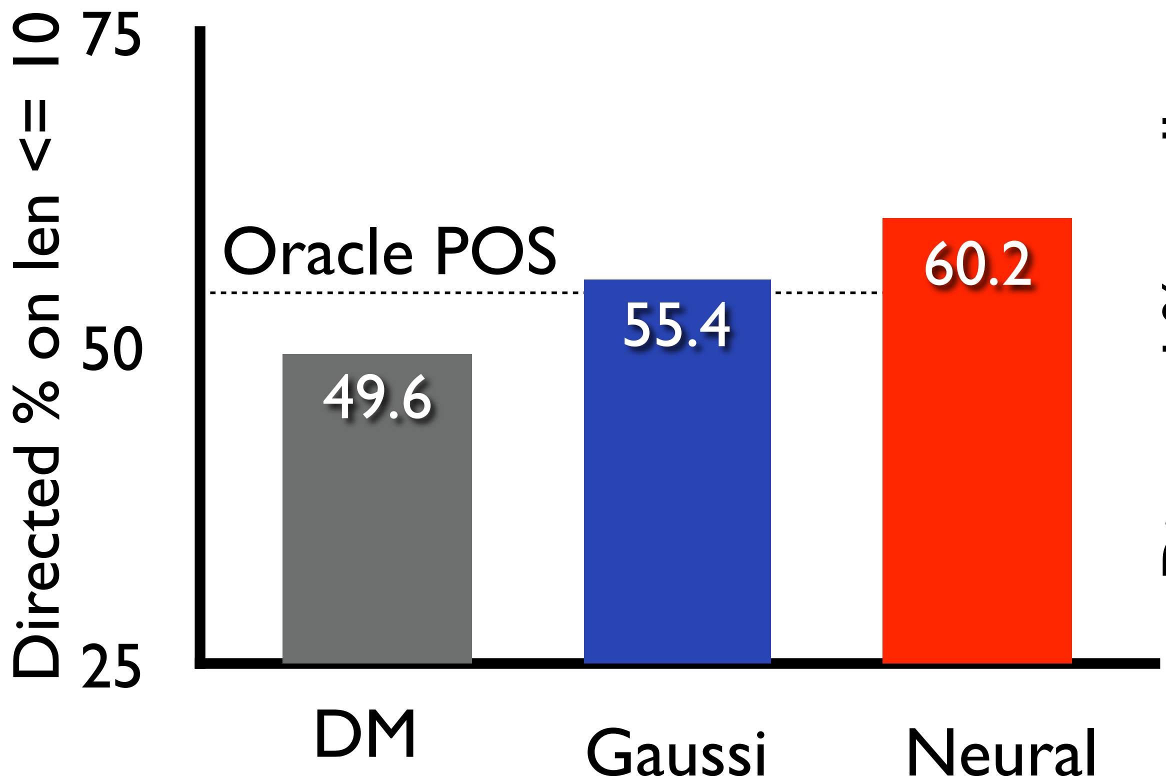


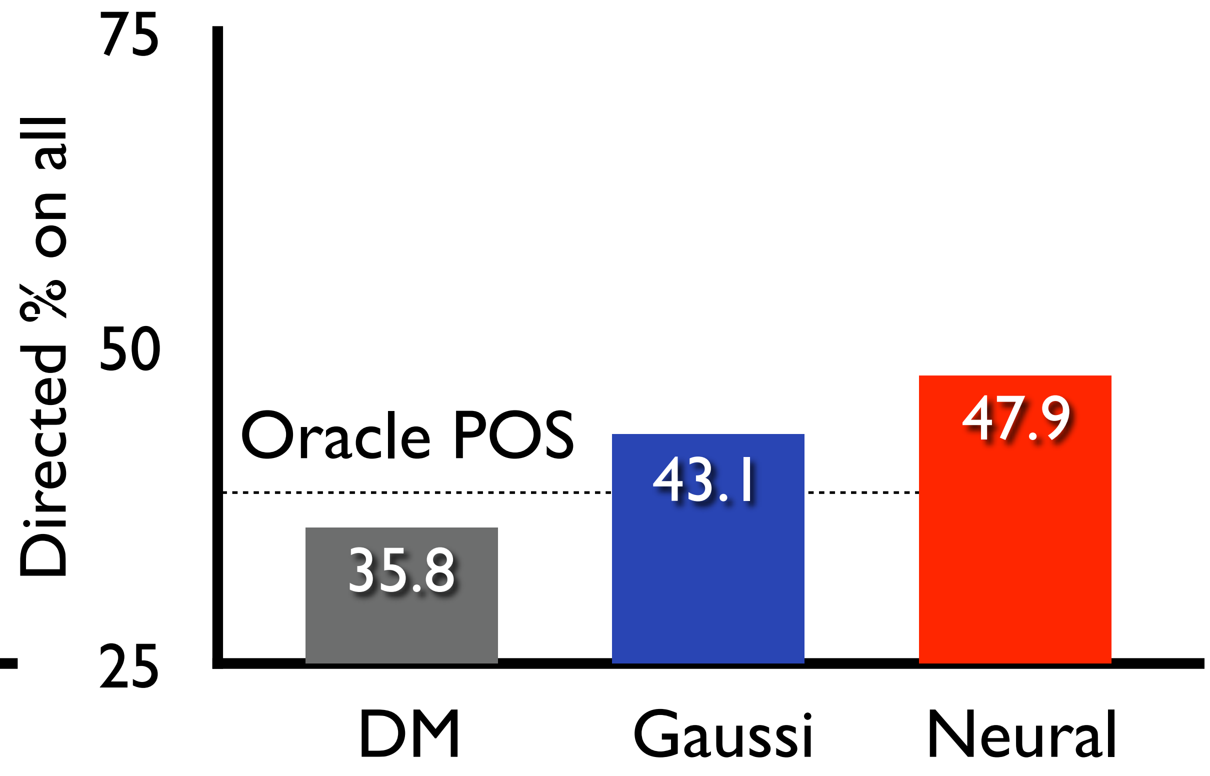
Part-of-speech Induction



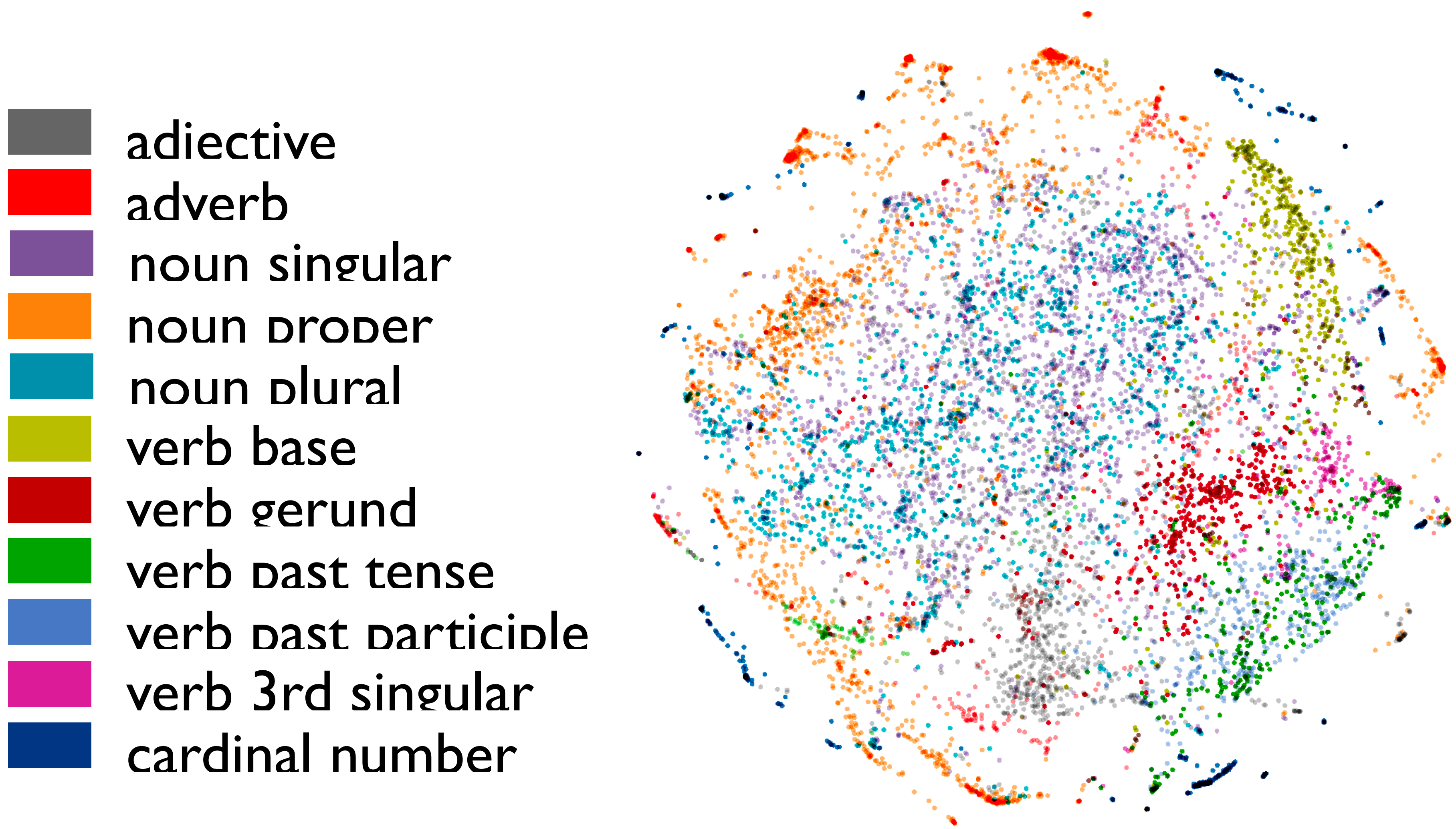
Outperform feature-based SOTA

Dependency Parse Induction

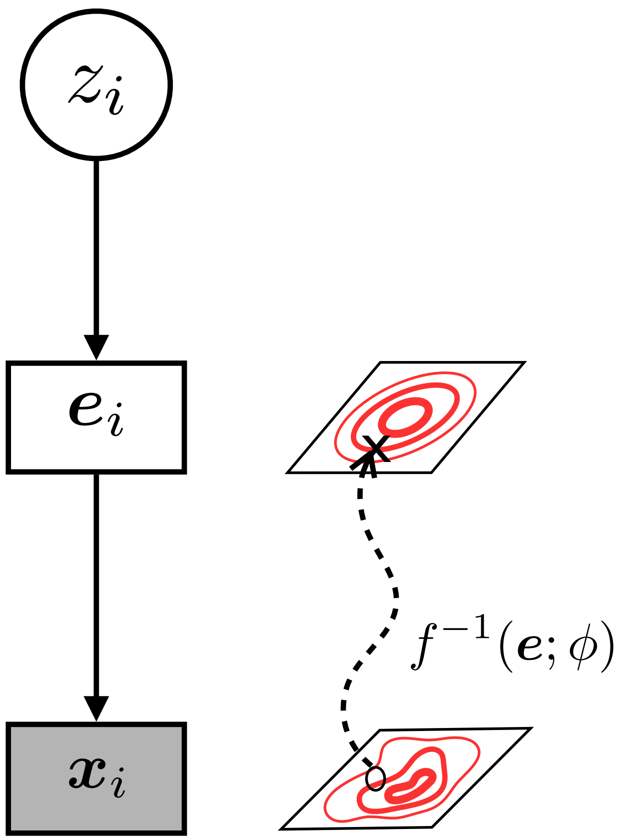




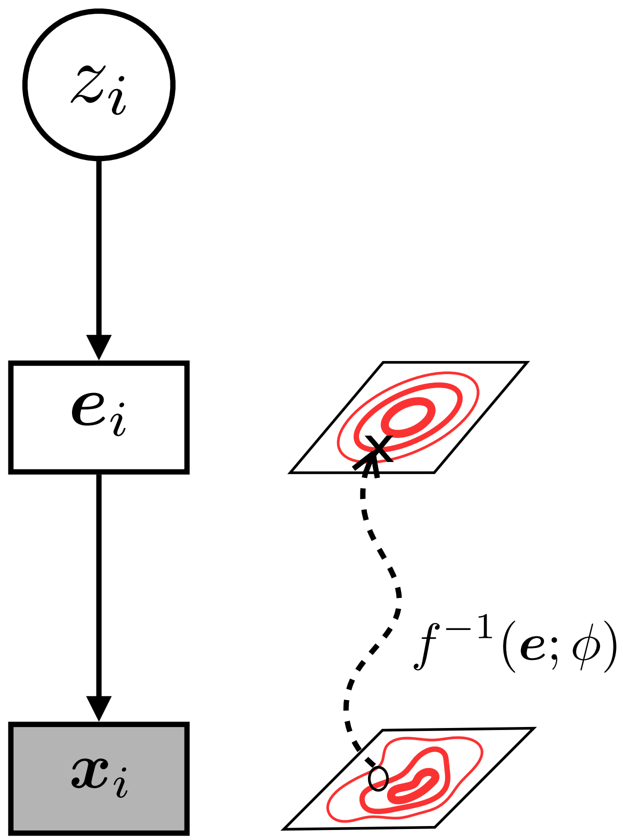
Original Embedding Space

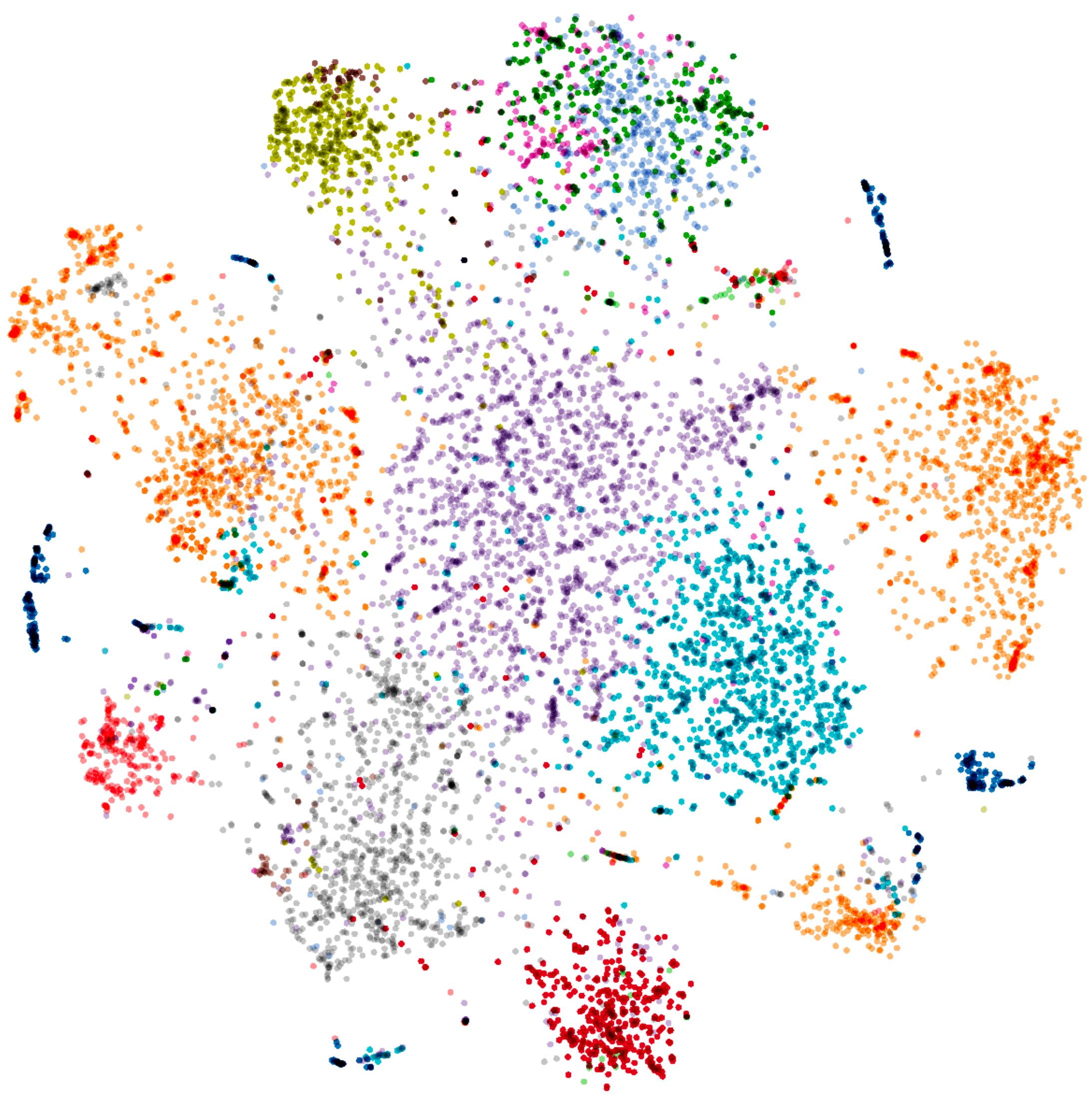


Projected Embedding Space w/ Markov Prior

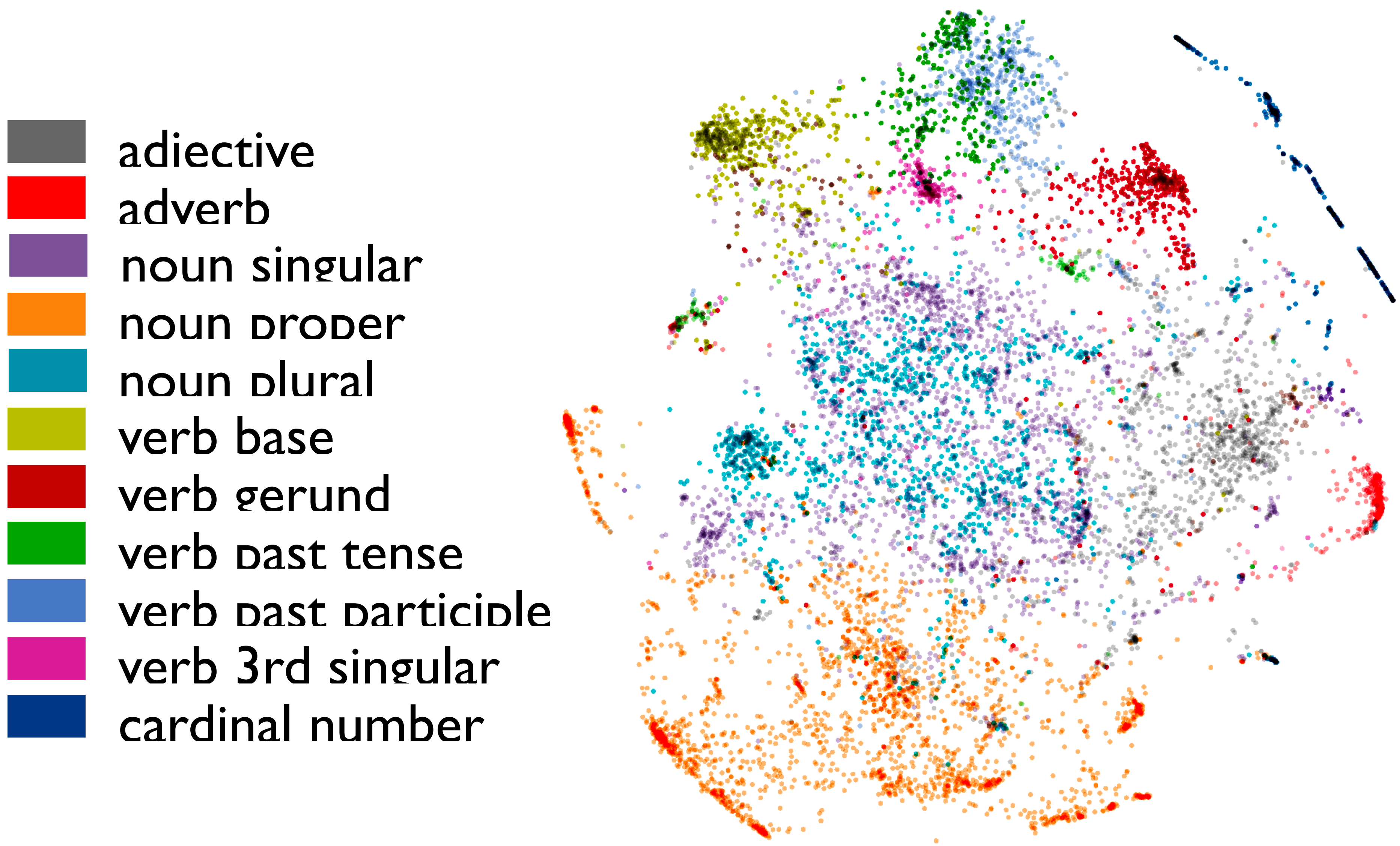


Projected Embedding Space w/ Markov Prior

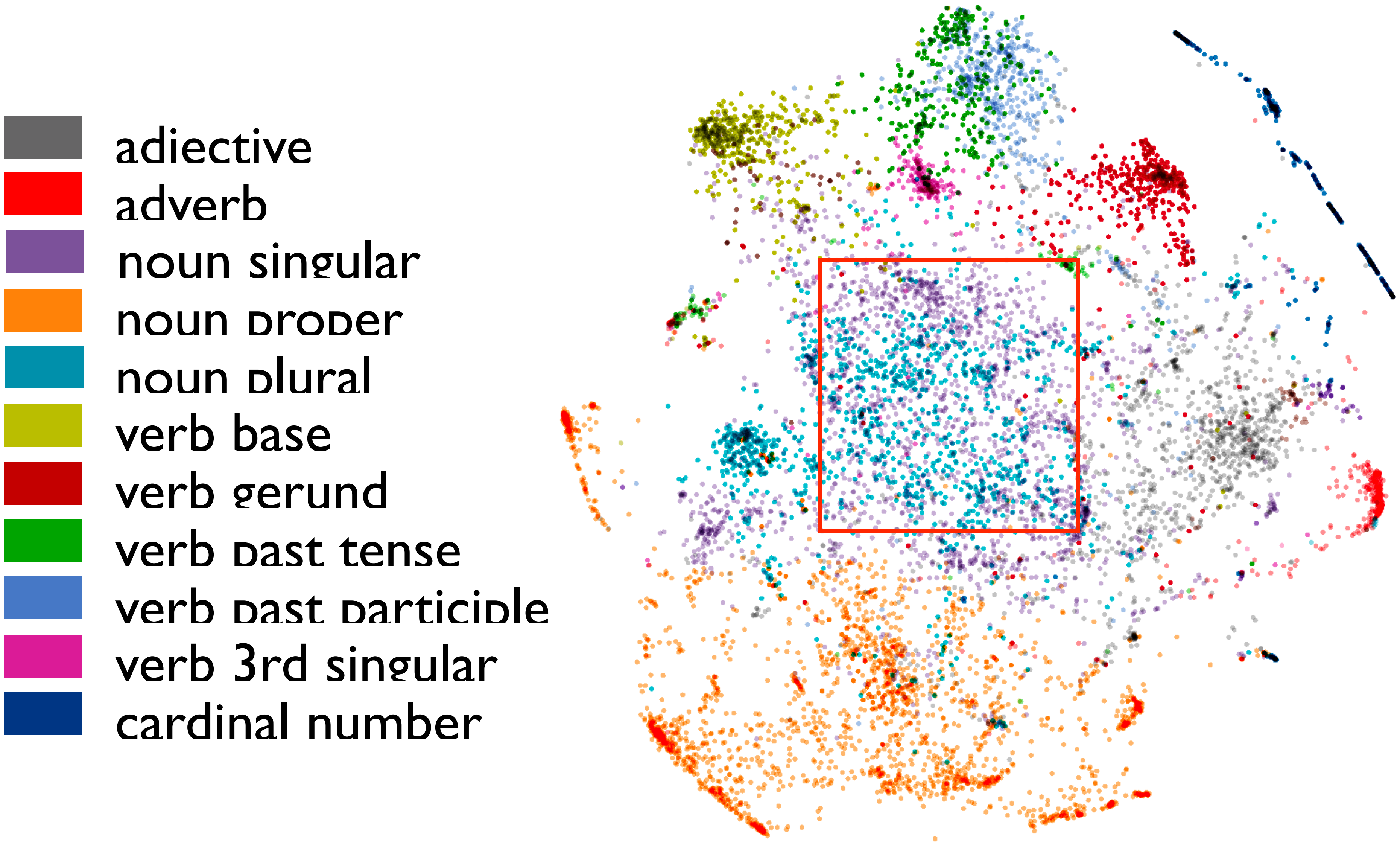




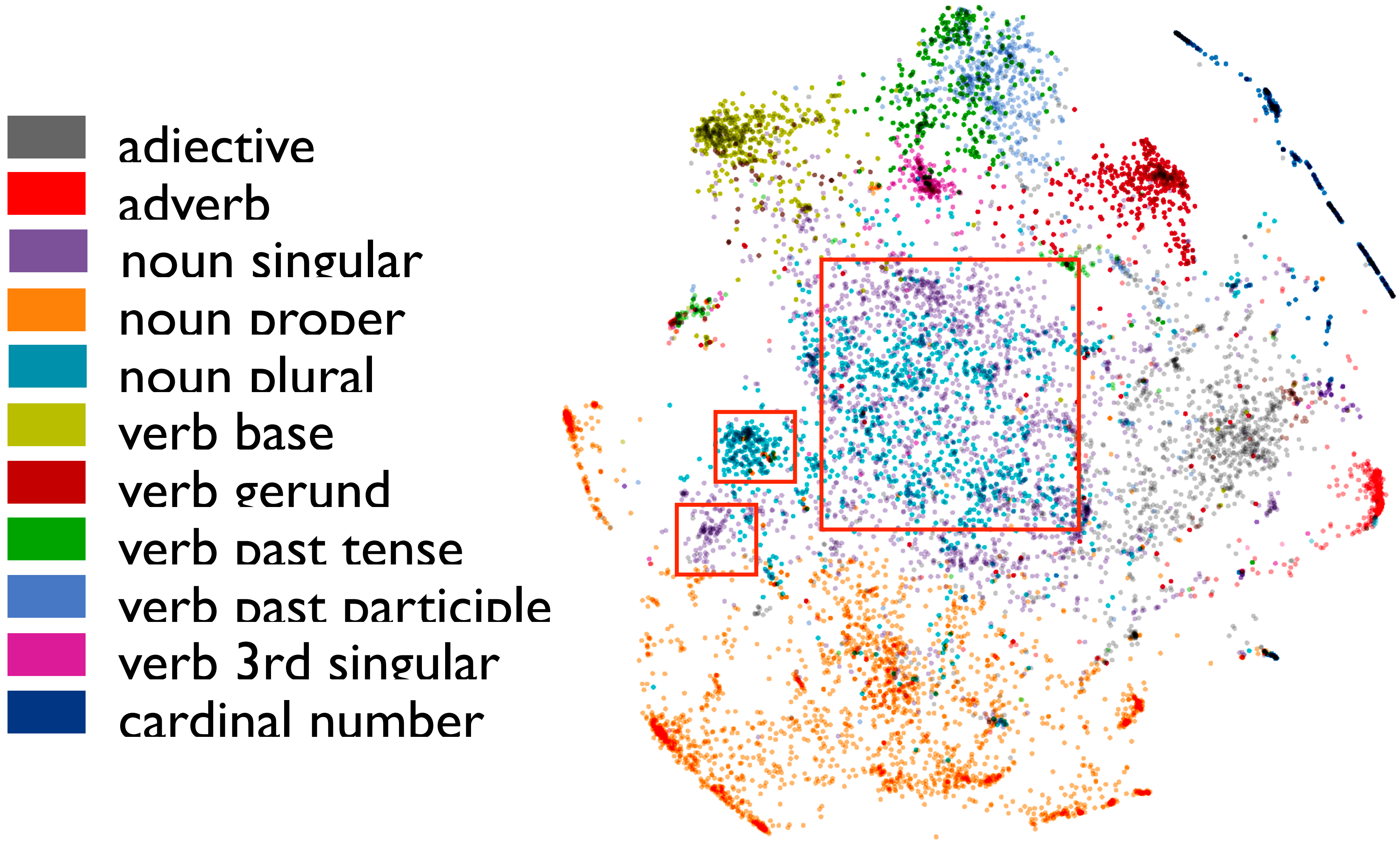
Projected Embedding Space w/ DMV Prior



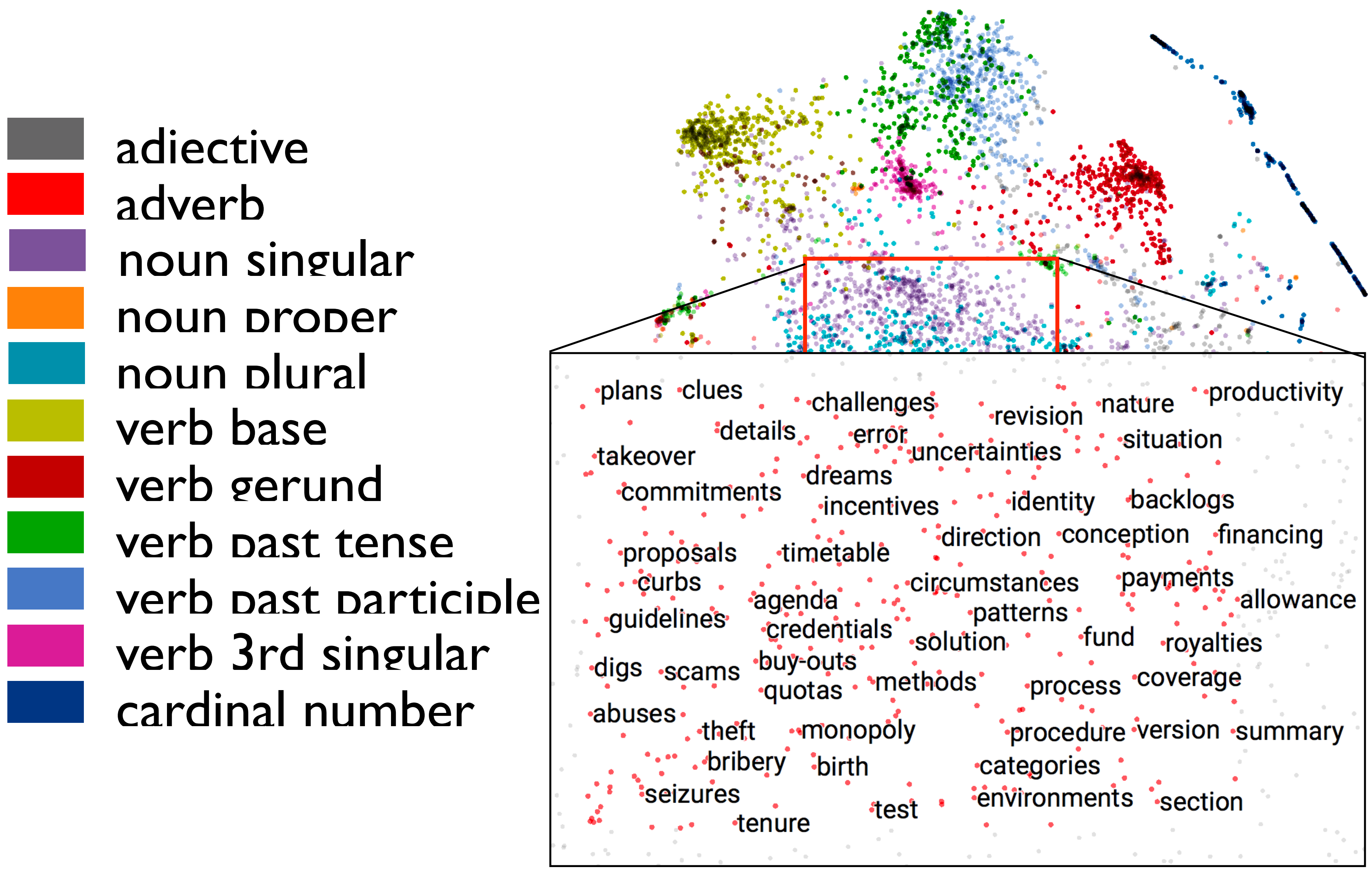
Projected Embedding Space w/ DMV Prior



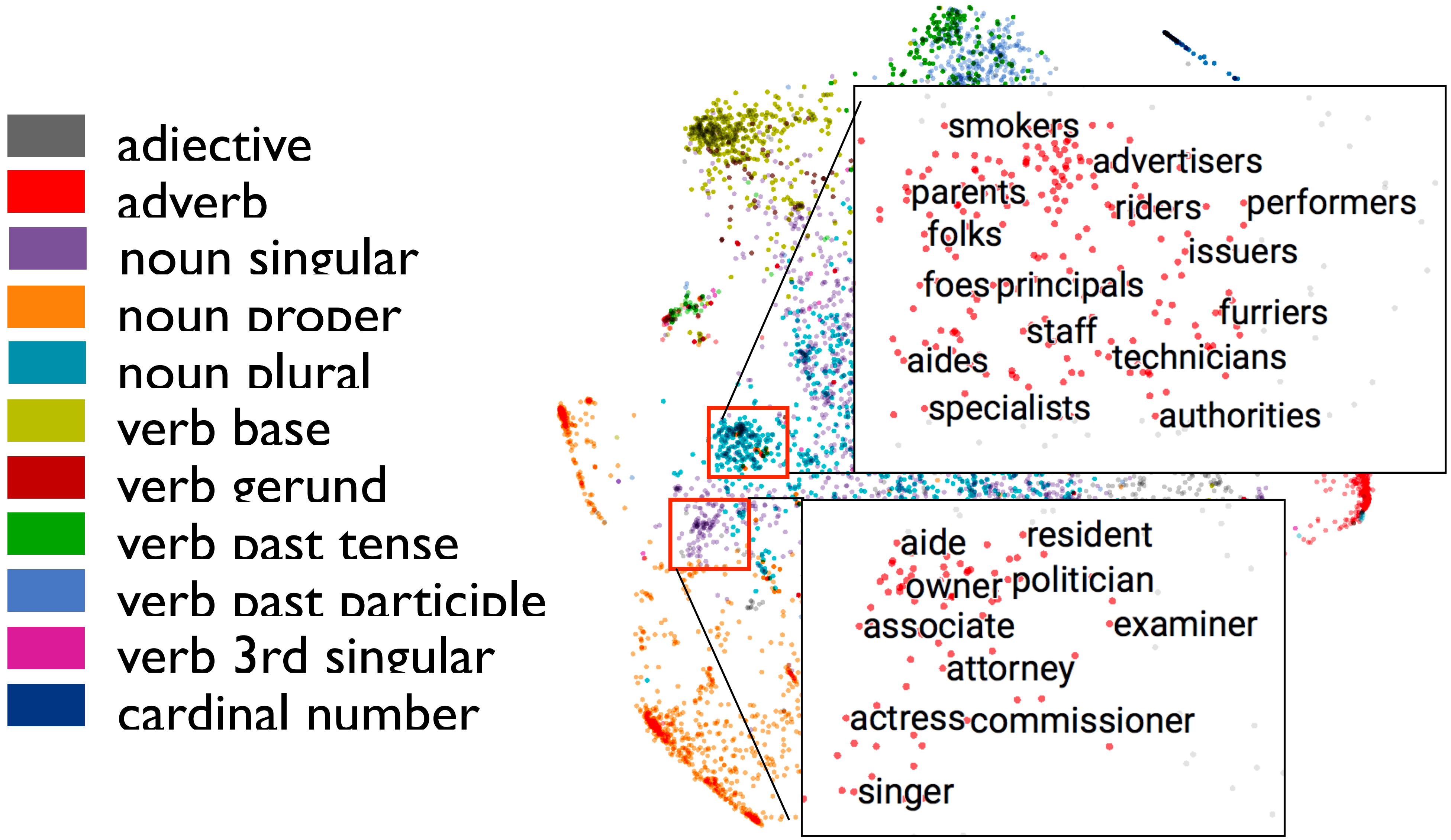
Projected Embedding Space w/ DMV Prior



Projected Embedding Space w/ DMV Prior



Projected Embedding Space w/ DMV Prior



Conclusion

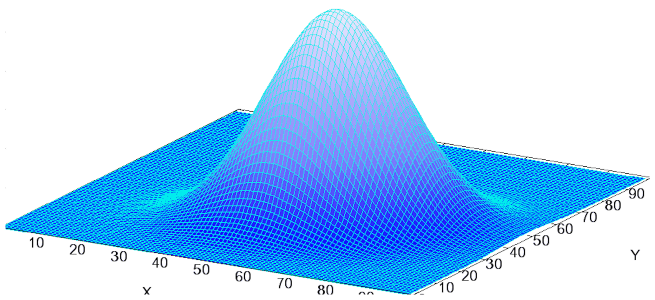
Conclusion

Conclusion

- Normalizing flows for unsupervised learning

$$X = f_\theta^{-1}(Z)$$

$$Z = f_\theta(X)$$

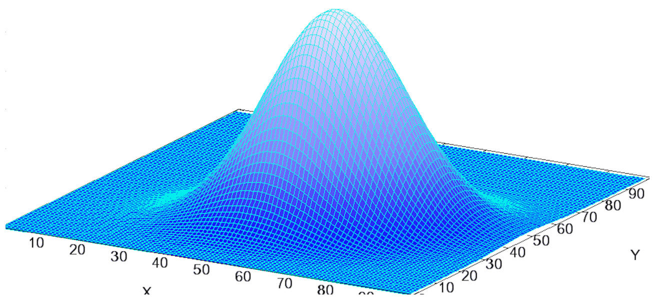


Conclusion

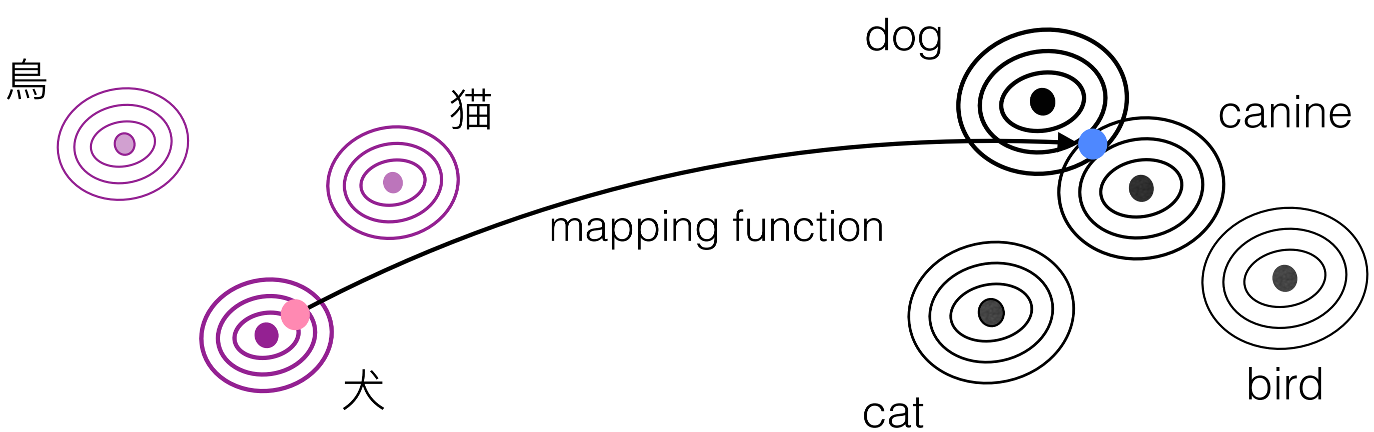
- Normalizing flows for unsupervised learning

$$X = f_\theta^{-1}(Z)$$

$$Z = f_\theta(X)$$



- Learning of bilingual lexicons

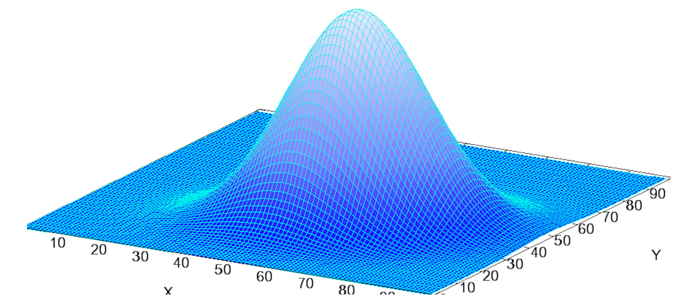


Conclusion

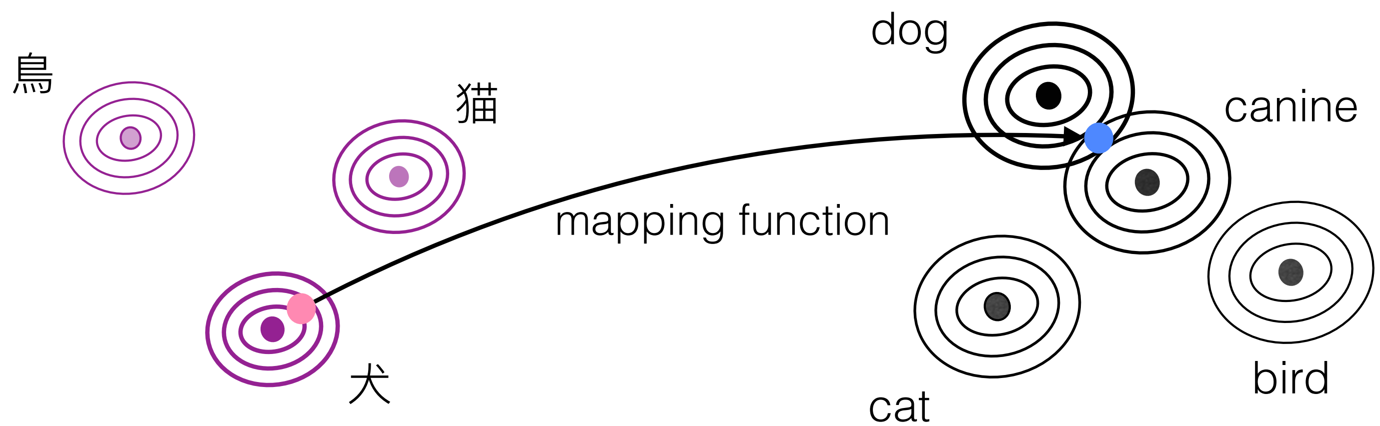
- Normalizing flows for unsupervised learning

$$X = f_\theta^{-1}(Z)$$

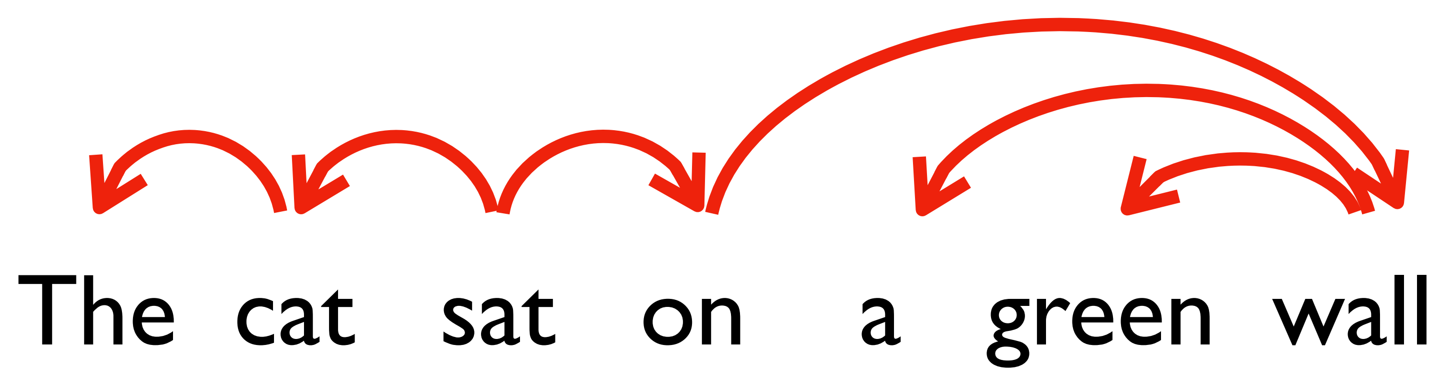
$$Z = f_\theta(X)$$



- Learning of bilingual lexicons



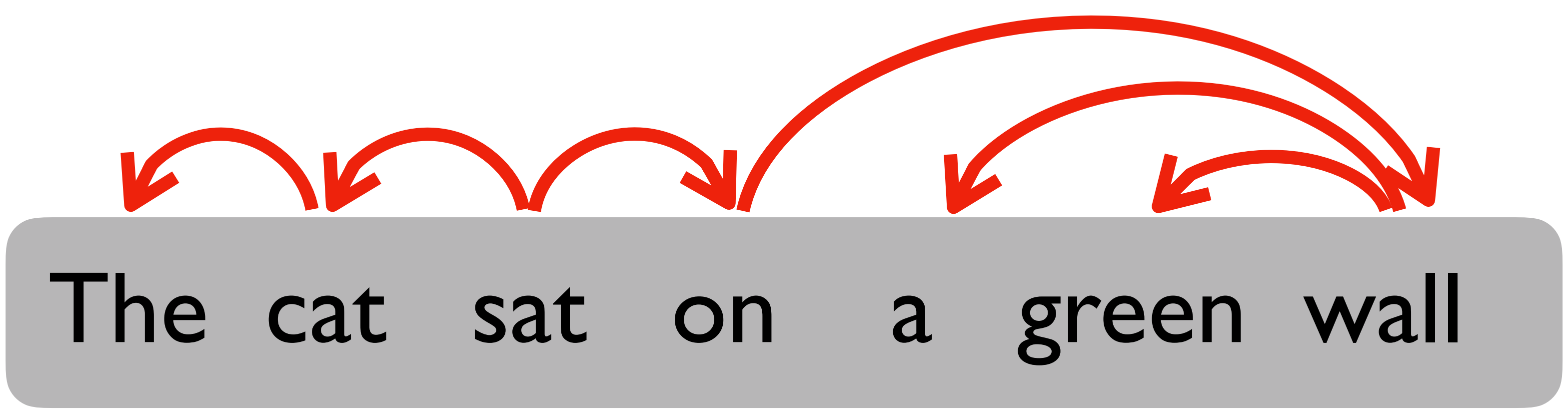
- Learning of syntactic structure



Thank You! Questions?

DeMa-BWE

https://github.com/violet-zct/DeMa-BWE



https://github.com/jxhe/struct-learning-with-flow