Zero-Shot Learning for Word Translation: Successes and Failures

Ndapa Nakashole,
University of California, San Diego
05 June 2018

Outline

- Introduction
- Successes
- Limitations

Zero-shot learning

- Zero-shot learning:

  $\Longrightarrow$ at test time can encounter an instance whose corresponding label was not seen at training time

  $x_j \in \mathcal{X}_{test}$

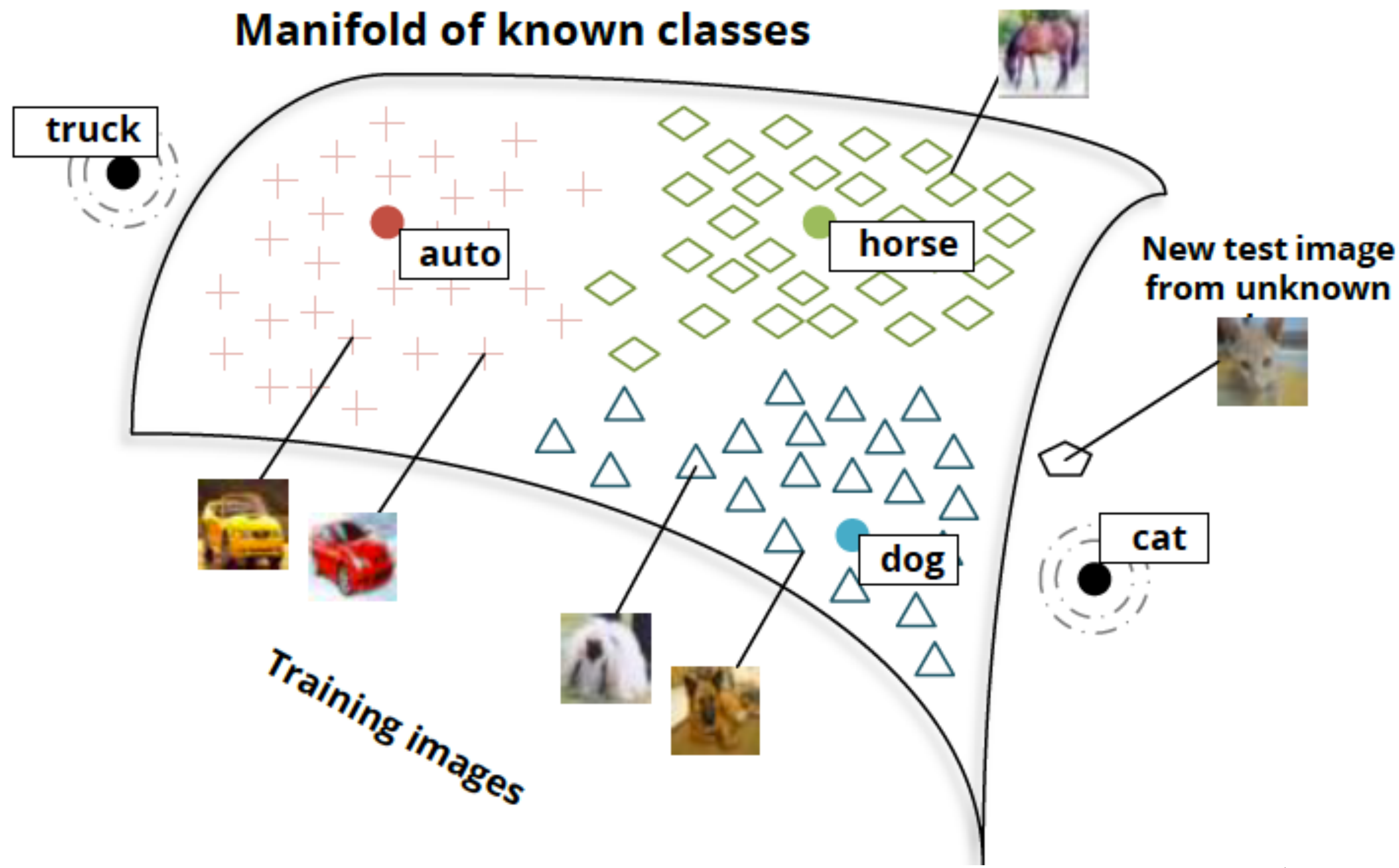
  $y_j \notin \mathcal{Y}$

- ZL setting occurs in domains with many possible labels

Zero-shot learning: Unseen labels

- To deal with labels that have no training data
  - ▷ Instead of learning parameters associated with each label $y_{\in}\mathcal{Y}$
  - ▷ Treat as problem of learning a single projection function
- Resulting function can then map input vectors to label space

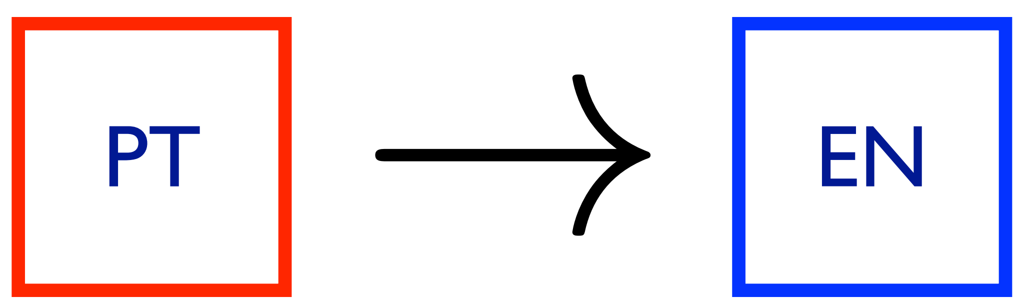
Zero-shot Learning: Cross-Modal Mapping



- Socher et al. 2013

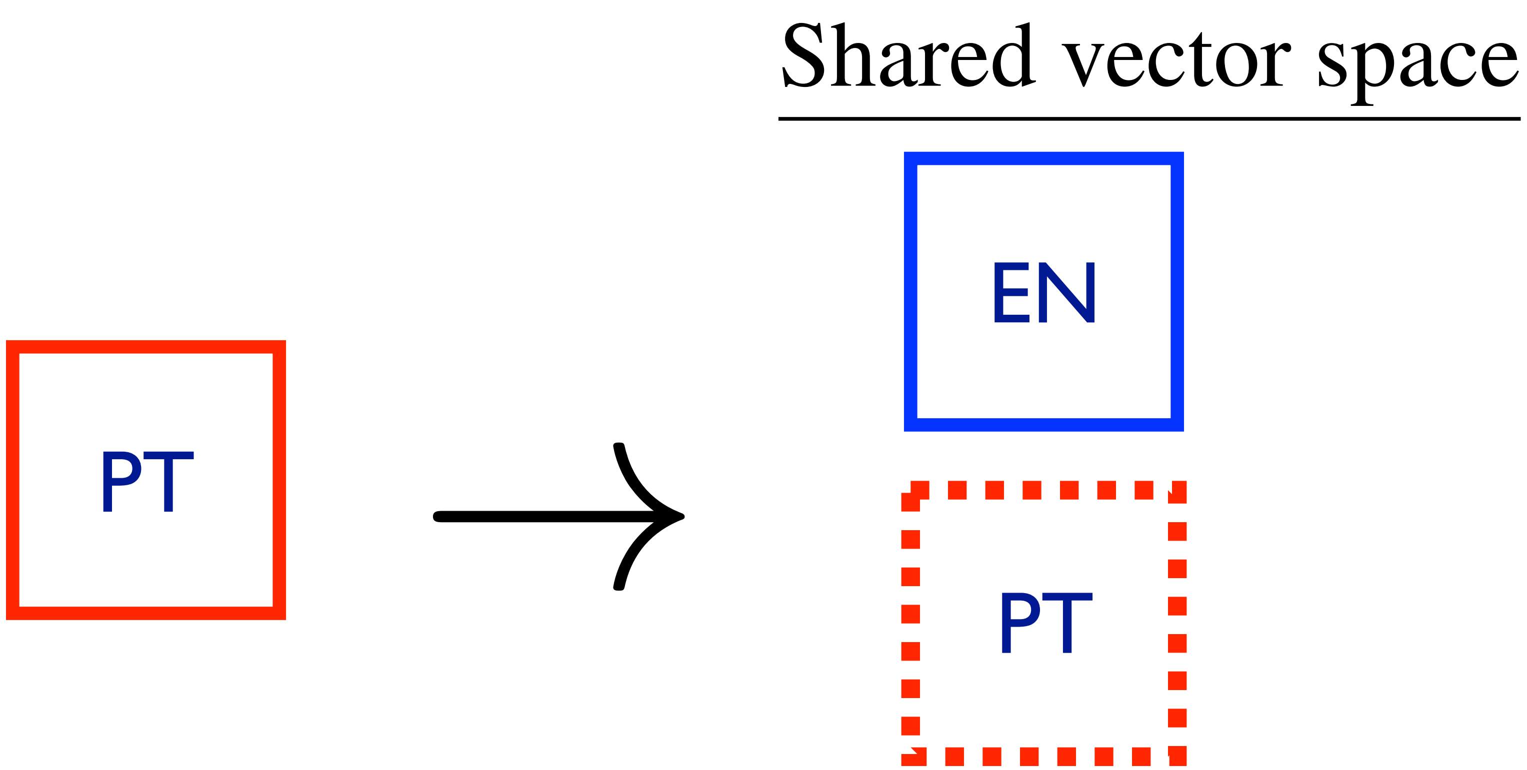
Cross-lingual mapping

- First generate monolingual word embeddings for each language
- Second, learn to map between embedding spaces of different languages



Multilingual word embeddings

- Creates multilingual word embeddings



- Multilingual word embeddings uses:
  - ▷ Model transfer
  - ▷ Recent: initialize unsupervised machine translation

Problem

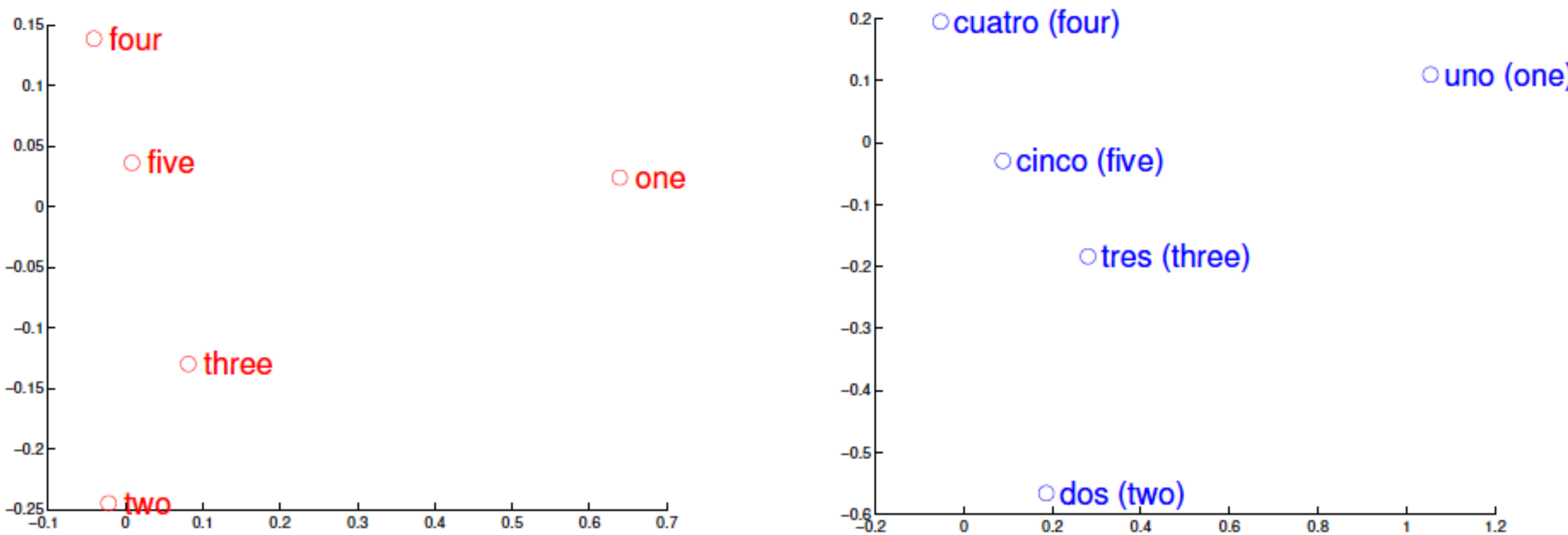
- Learn cross-lingual mapping function
  - that projects vectors from embedding space of one language to another

Outline

- early work & assumptions
- improving precision
- reducing supervision

Early work & assumptions

- Concepts have similar geometric arrangements in vector spaces of different languages (Mikolov et al. 2013)



Linear Mapping Function

- Mikolov et al. 2013
  - Mapping function/translation matrix learned with least squares loss

$$\hat{\mathbf{M}} = \arg\min_{\mathbf{M}} ||\mathbf{M}\mathbf{X} - \mathbf{Y}||_F + \lambda||\mathbf{M}||$$

$$y = \arg\max_y \cos(\mathbf{M}x, y)$$

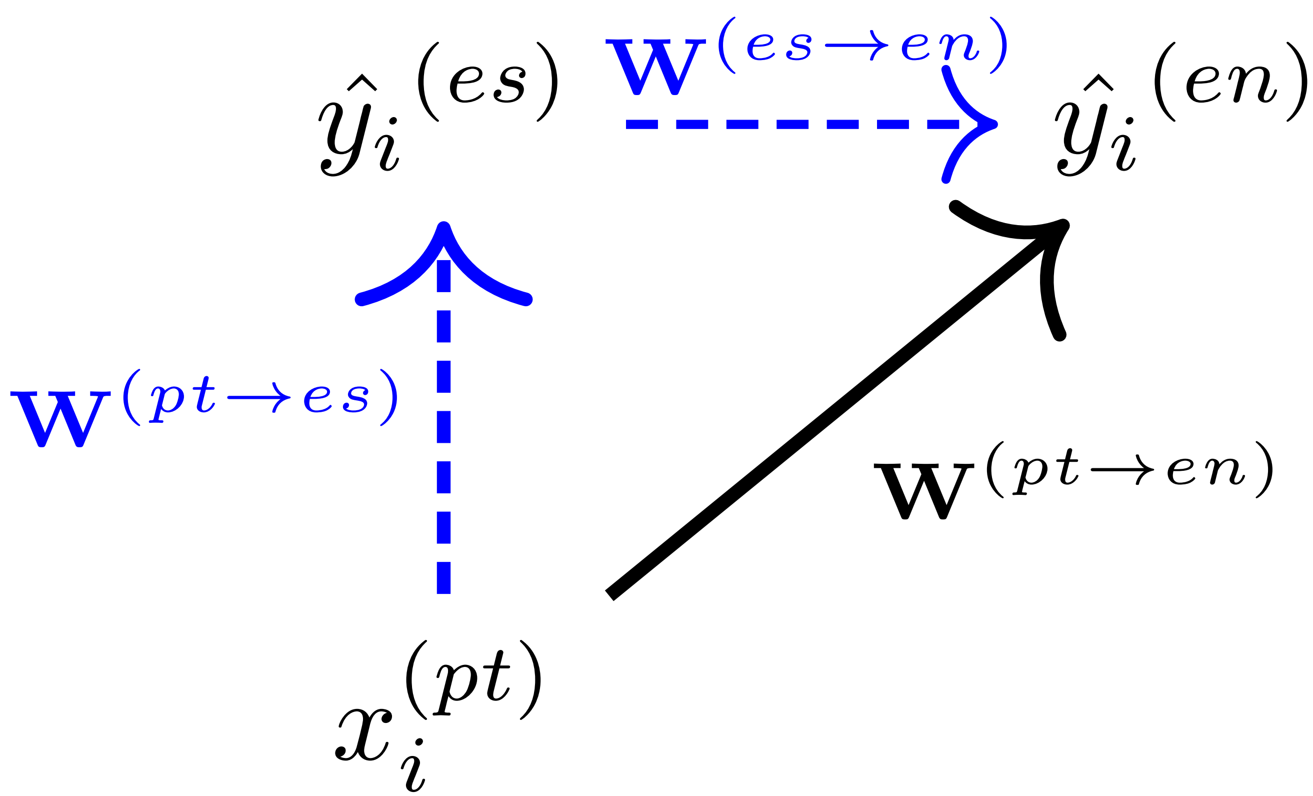
Improving accuracy

- Impose orthogonality constraint on learned map
  - Xing et al. 2015, Zhang et al 2016

- Ranking loss to learn map
  - Lazaridou et al. 2015

Reducing supervision

- Our own work: teacher-student framework (Nakashole EMNLP 2017)



- (Artetxe et al., 2017) bootstrap approach
  - Start with a small dictionary
  - Iteratively build it up while learning map function

No supervision

- Unsupervised training of mapping function (Barone 2016, Zhang et al., 2017; Conneau et al., 2018)
  - Adversarial training
  - **Discriminator**: separate mapped vectors **Mx** from targets **Y**
  - **Generator** (learned map): prevent discriminator from succeeding

Success Summary

- With no supervision current methods obtain high accuracy
  - However, there's room for improvement

Outline

Limitations

Assumptions

- Limitations tied to assumptions made by current methods
  - A1. Maps are linear (linearity)
  - A2. Embedding spaces are similar (isomorphism)

Assumption of Linearity

- SOTA methods learn linear maps
  – Artexte et al. 2018, Conneau et al. 2018, …, Nakashole 2017, … Mikolov et al. 2013

- Although assumed by SOTA & large body of work
  – Unclear to what extent the assumption of linearity holds

- Non-linear methods have been proposed
  – Currently not SOTA
  – Trying to optimize multi-layer neural networks for this zero-shot learning problem largely fails

Testing Linearity

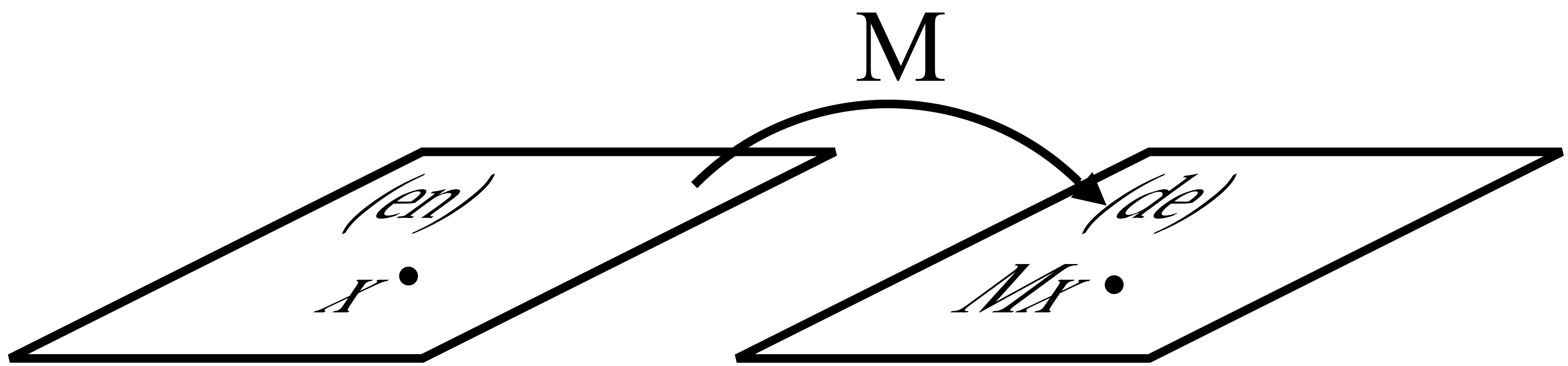
- To what extent does the assumption of linearity hold?

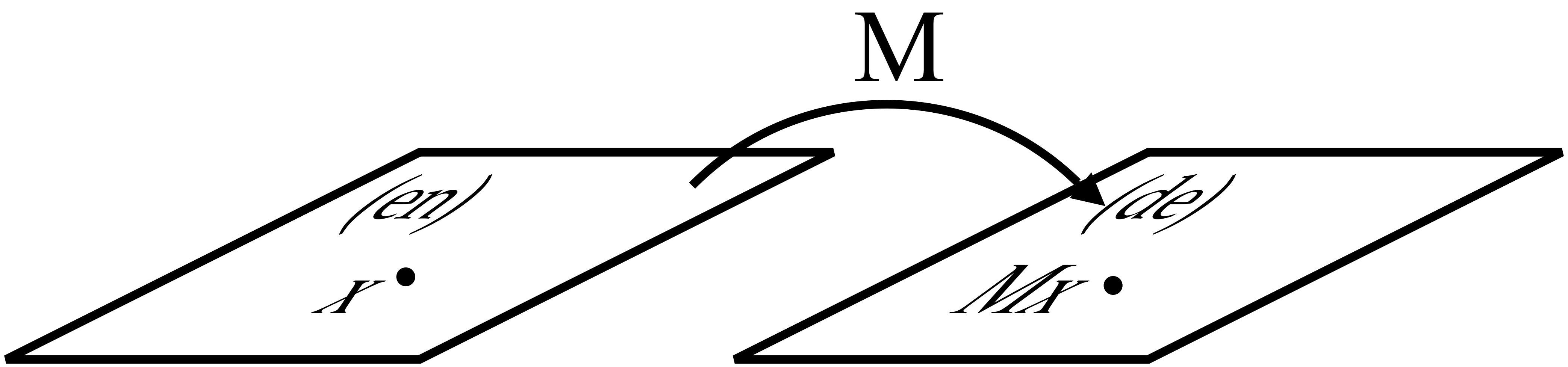
Testing Linearity

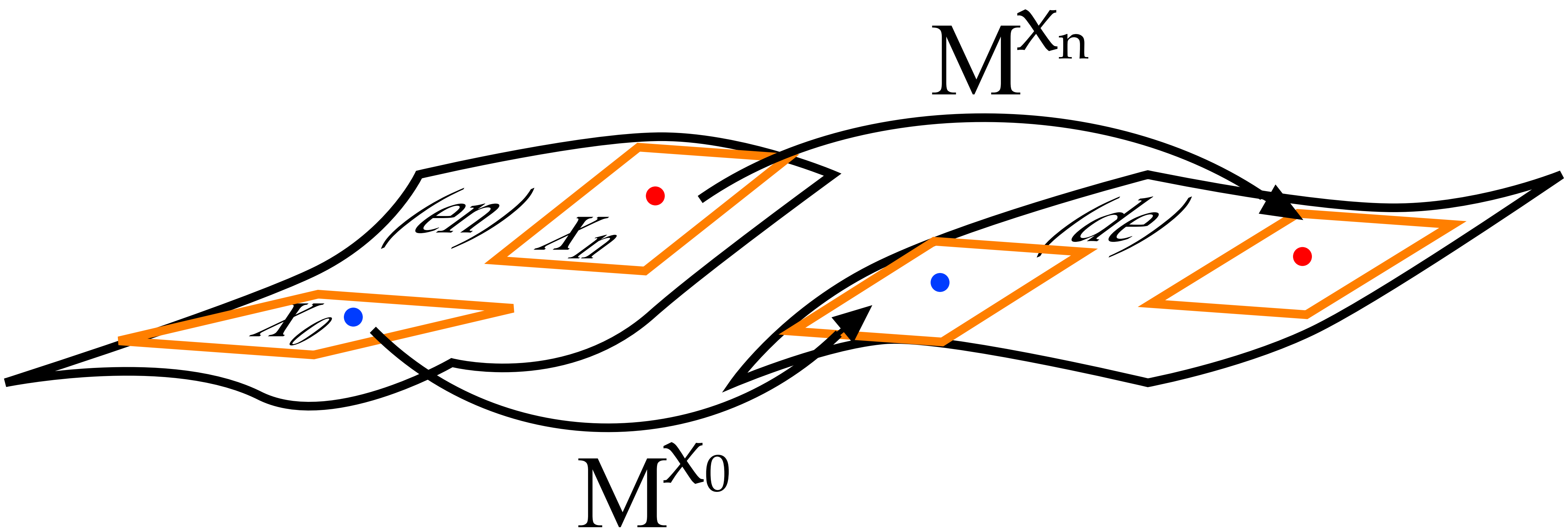
- Assume underlying mapping function is non-linear
  - but can be approximated by linear maps in small enough neighborhoods

- If the underlying map is linear
  - **local approximations should be identical or similar**

- If the underlying map is non-linear
  - **local approximations will vary across neighborhoods**


M
(en)
x
(de)
Mx

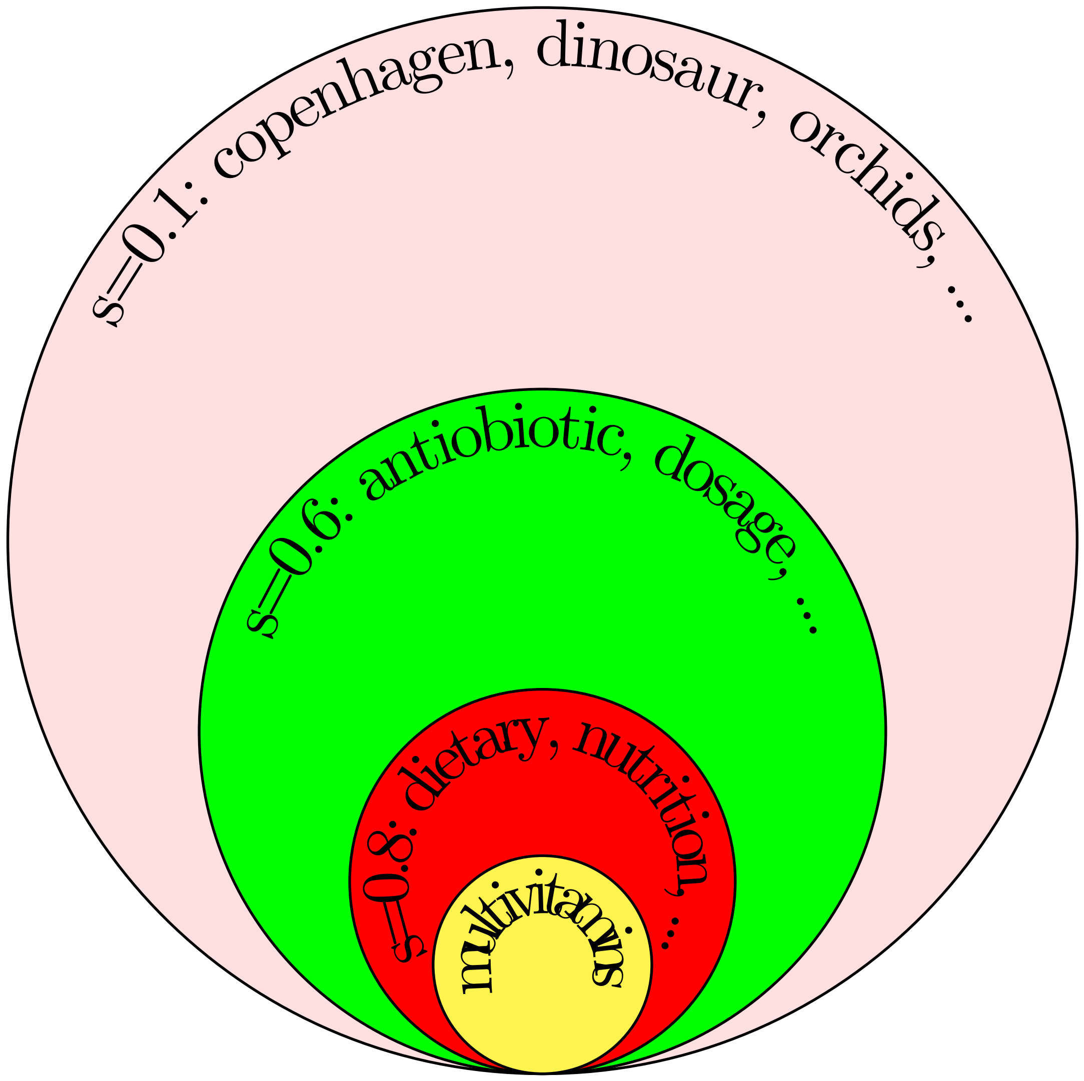

M
(en)
x
(de)
Mx


M^xn
(en)
xn
x0
(de)
M^x0

Neighborhoods in Word Vector Space

- To perform linearity test, need to define neighborhood

  – Pick an ‘anchor’ word, consider all nearby words (cos sim>=0.5) to be in its neighborhood



Neighborhoods: en-de

	$\cos(x_0, x_i)$
x_0:multivitamins	1.0
x_1:antibiotic	0.60
x_2:disease	0.45
x_3:blowflies	0.33
x_4:dinosaur	0.24
x_5:orchids	0.19
x_6:copenhagen	0.11

Neighborhood maps

- We consider three training settings:

1. Train a single map on one of the neighborhoods (1 Map)
2. Train a map for every neighborhood (N maps)
3. Train a global map (1 Map) : this is the typical setting

Setting 1: train a single map (M^{X0})

- Translate words from all neighborhoods using M^{X0}

	x_0 Similarity	Translation Accuracy
	$\cos(x_0, x_i)$	$\mathbf{M}^{\mathbf{x_0}}$
x_0:multivitamins	1.0	**68.2**
x_1:antibiotic	0.60	67.3
x_2:disease	0.45	59.2
x_3:blowflies	0.33	28.4
x_4:dinosaur	0.24	14.7
x_5:orchids	0.19	19.3
x_6:copenhagen	0.11	31.2

Setting 2: a map for every neighborhood (M^{x_i})

	x_0 Similarity	Translation Accuracy		
	$\cos(x_0, x_i)$	M^{x_0}	M^{x_i}	Δ
x_0:multivitamins	1.0	**68.2**	**68.2**	0
x_1:antibiotic	0.60	67.3	**72.7**	5.4 ↑
x_2:disease	0.45	59.2	**73.4**	14.2 ↑
x_3:blowflies	0.33	28.4	**73.2**	44.8 ↑
x_4:dinosaur	0.24	14.7	**77.1**	62.4 ↑
x_5:orchids	0.19	19.3	**78.0**	58.7 ↑
x_6:copenhagen	0.11	31.2	**67.4**	36.2 ↑

Testing Linearity Assumption

- If the underlying map is linear
  - **local approximations should be identical or similar**

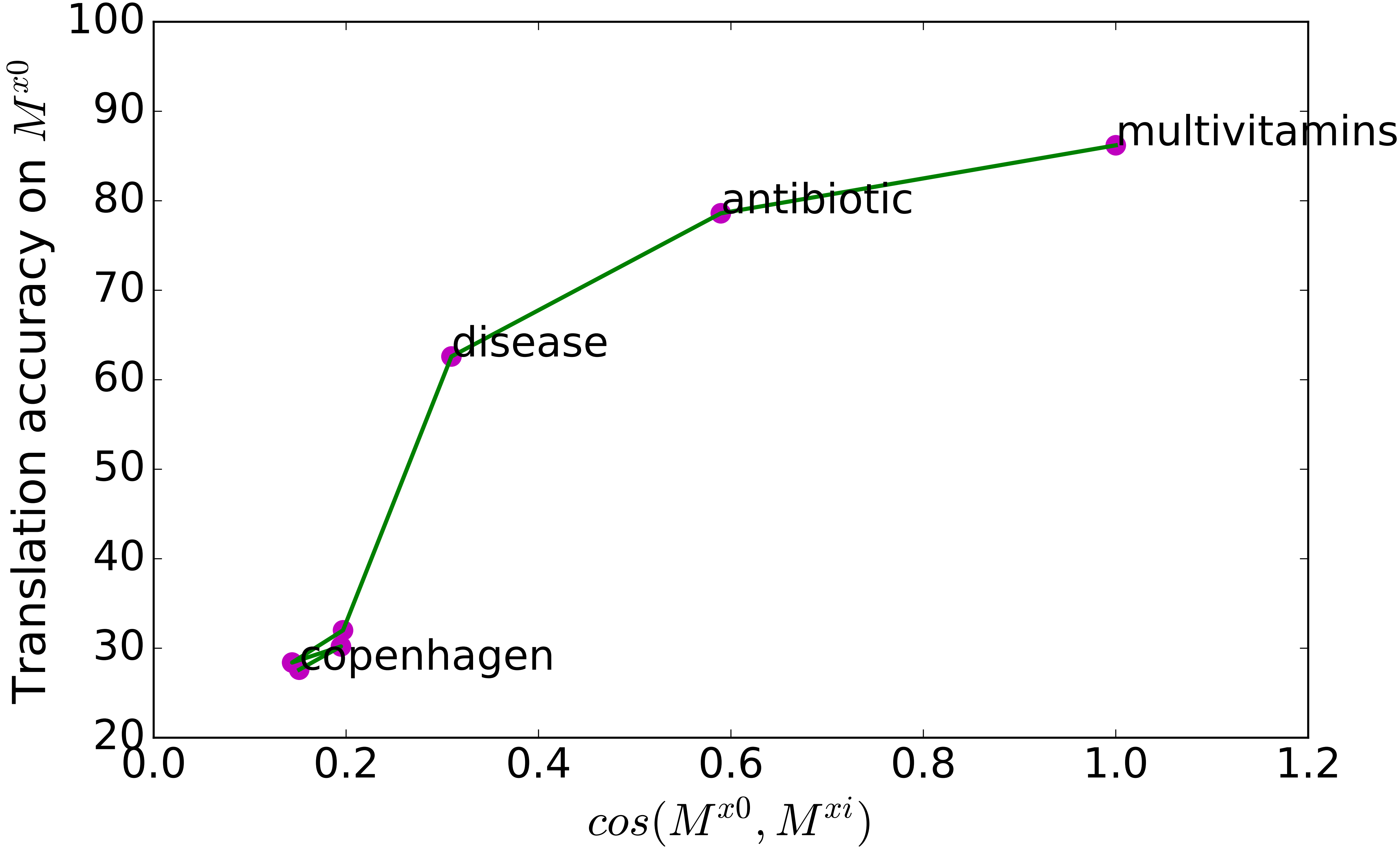
- If the underlying map is non-linear
  - **local approximations will vary across neighborhoods**

Map Similarity

$$\cos(\mathbf{M_1}, \mathbf{M_2}) = \frac{tr(\mathbf{M_1}^T\mathbf{M_2})}{\sqrt{tr(\mathbf{M_1}^T\mathbf{M_1})tr(\mathbf{M_2}^T\mathbf{M_2})}}$$

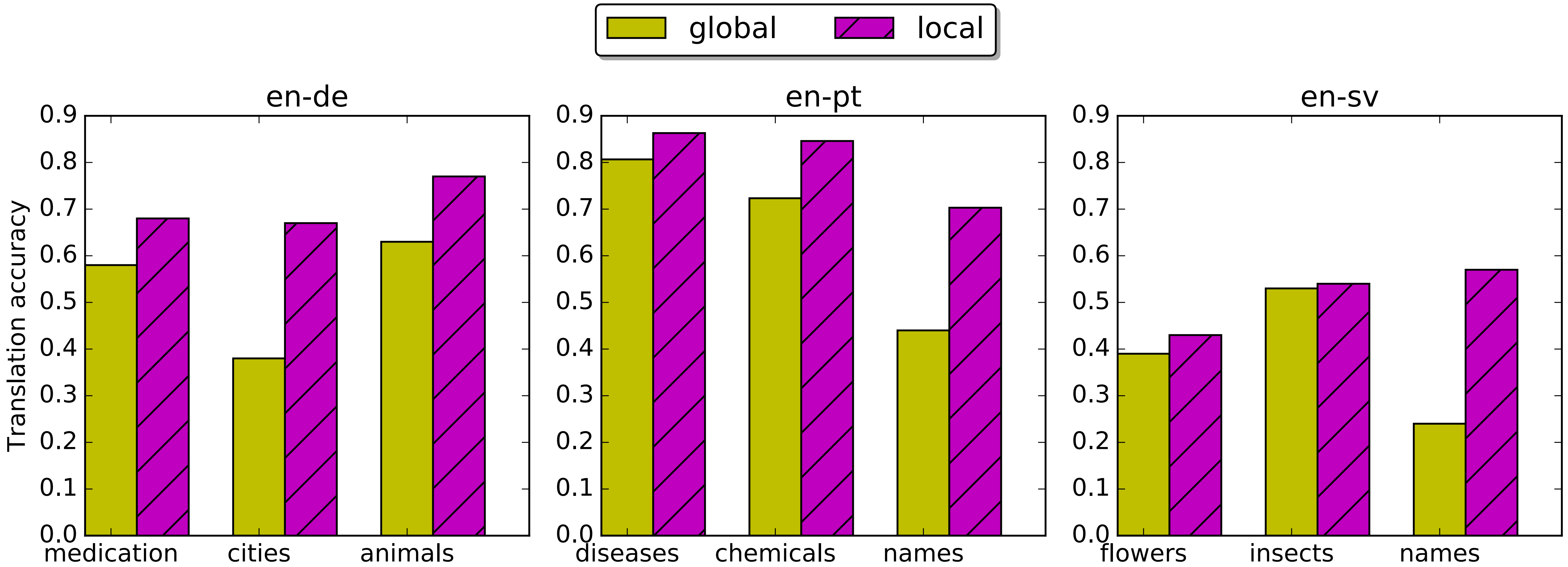
	x_0 Similarity	
	$\cos(x_0, x_i)$	$\cos(\mathbf{M^{x_0}}, \mathbf{M^{x_i}})$
x_0:multivitamins	1.0	1.0
x_1:antibiotic	0.60	0.59
x_2:disease	0.45	0.31
x_3:blowflies	0.33	0.20
x_4:dinosaur	0.24	0.14
x_5:orchids	0.19	0.20
x_6:copenhagen	0.11	0.15

Translate (^{Xi}) neighborhood using (M^{X0})



Setting 3: train a single global map (M)

	x_0 Similarity	Translation Accuracy		
	$\cos(x_0, x_i)$	M	M^{x_0}	M^{x_i}
x_0:multivitamins	1.0	58.3	**68.2**	**68.2**
x_1:antibiotic	0.60	61.1	67.3	**72.7**
x_2:disease	0.45	69.3	59.2	**73.4**
x_3:blowflies	0.33	71.4	28.4	**73.2**
x_4:dinosaur	0.24	63.2	14.7	**77.1**
x_5:orchids	0.19	73.7	19.3	**78.0**
x_6:copenhagen	0.11	38.5	31.2	**67.4**


global
local
en-de
en-pt
en-sv
Translation accuracy
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
medication
cities
animals
diseases
chemicals
names
flowers
insects
names

Linearity Assumption: Summary

- Provided evidence that linearity assumption does not hold
- Locally linear maps vary
  - by an amount tightly correlated with distance between neighborhoods on which they were trained

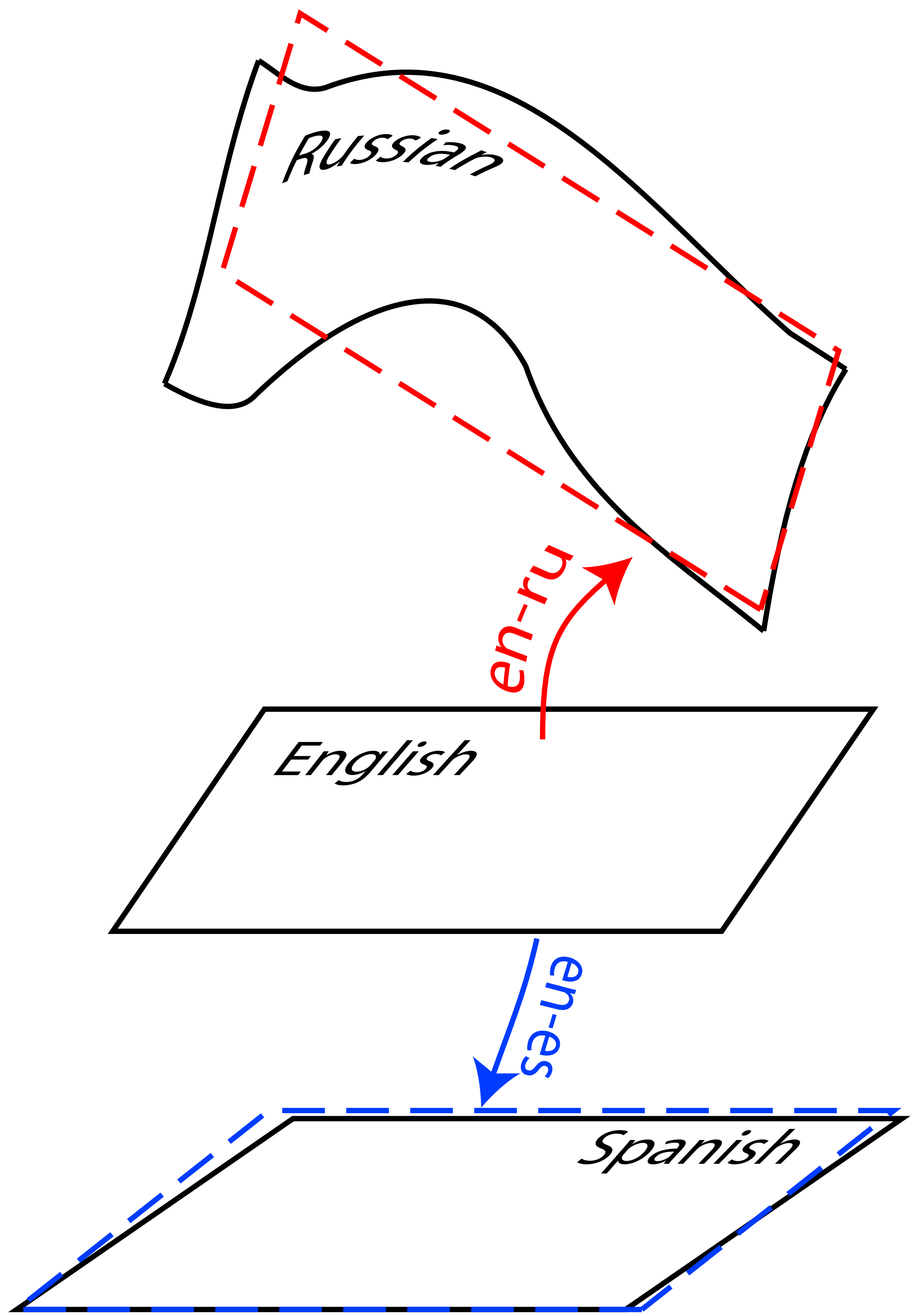
But SOTA achieves remarkable precision

- SOTA unsupervised, precision@1 ~80% (Conneau et al. ICLR 2018)
  - BUT only for closely related languages, e.g, EN-ES
- Distant languages?
  - Precision much lower, ~ 40% EN-RU, ~30% EN-ZH

Assumptions

- Limitations tied to assumptions made by current methods
  - A1. Maps are linear (linearity)
  - A2. Embedding spaces are similar (isomorphism)

close vs distant language translation



State-of-the-Art

	en-ru	en-zh	en-de	en-es	en-fr
Artetxe et al . 2018	47.93	20.4	70.13	**79.6**	**79.30**
Conneau et al. 2018	37.30	30.90	**71.30**	79.10	78.10

- Datasets: FAIR MUSE lexicons
- 5k train/1.5k test

Proposed approach

- To capture differences in embedding spaces
  - learn neighborhood sensitive maps

Learn neighborhood sensitive maps

- In principle can do this by learning a non-linear map
  - Currently not SOTA
  - Trying to optimize multi-layer neural networks for this zero-shot learning problem largely fails

Jointly discover neighborhoods & translate

- We propose to jointly discover neighborhoods
  - while learning to translate

Reconstructive Neighborhood Discovery

- Discovered by learning a reconstructive dictionary of neighborhoods
  - Reconstruct word vector x_i using a linear combination of K neighborhoods.
  - Dictionary that minimizes reconstruction error (Lee et al 2007)

$$\mathbf{D}, \mathbf{V} = \arg\min_{\mathbf{D},\mathbf{V}} ||\mathbf{X} - \mathbf{V}\mathbf{D}||_2^2$$

$$\mathbf{X}_{\mathcal{F}} = \mathbf{X}\mathbf{D}^T$$

Maps

- Use neighborhood aware representation to learn maps

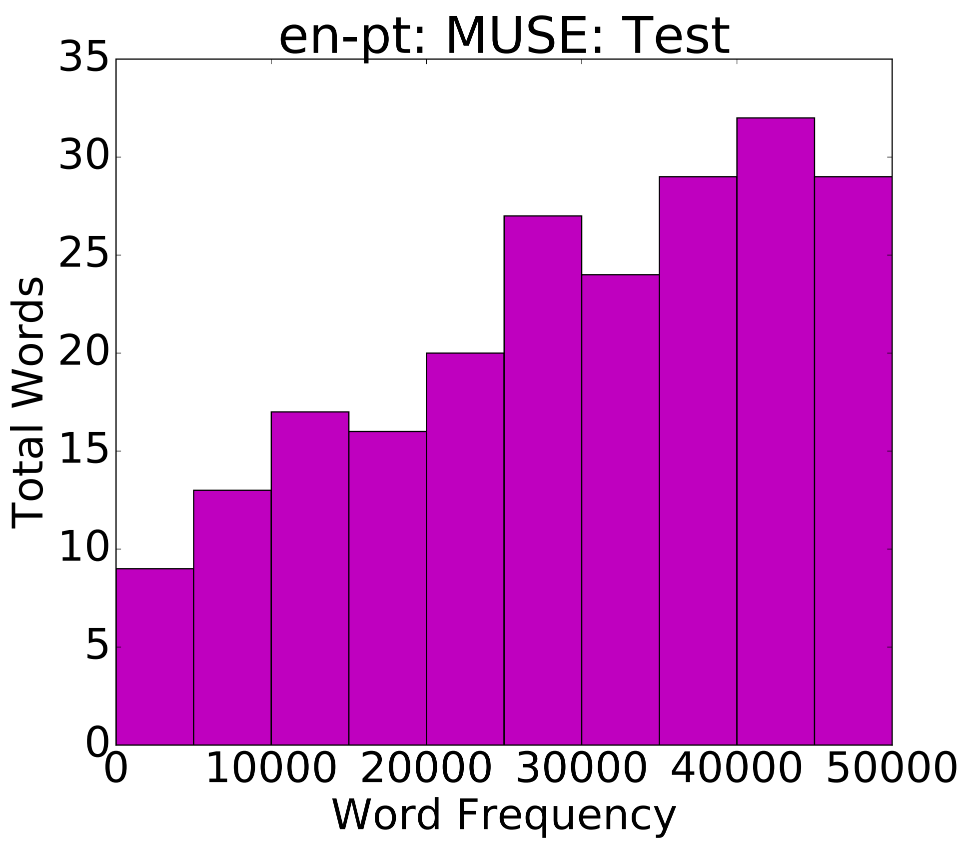
$$\hat{y}_i^{linear} = \mathbf{W} x_{\mathcal{F}_i}$$

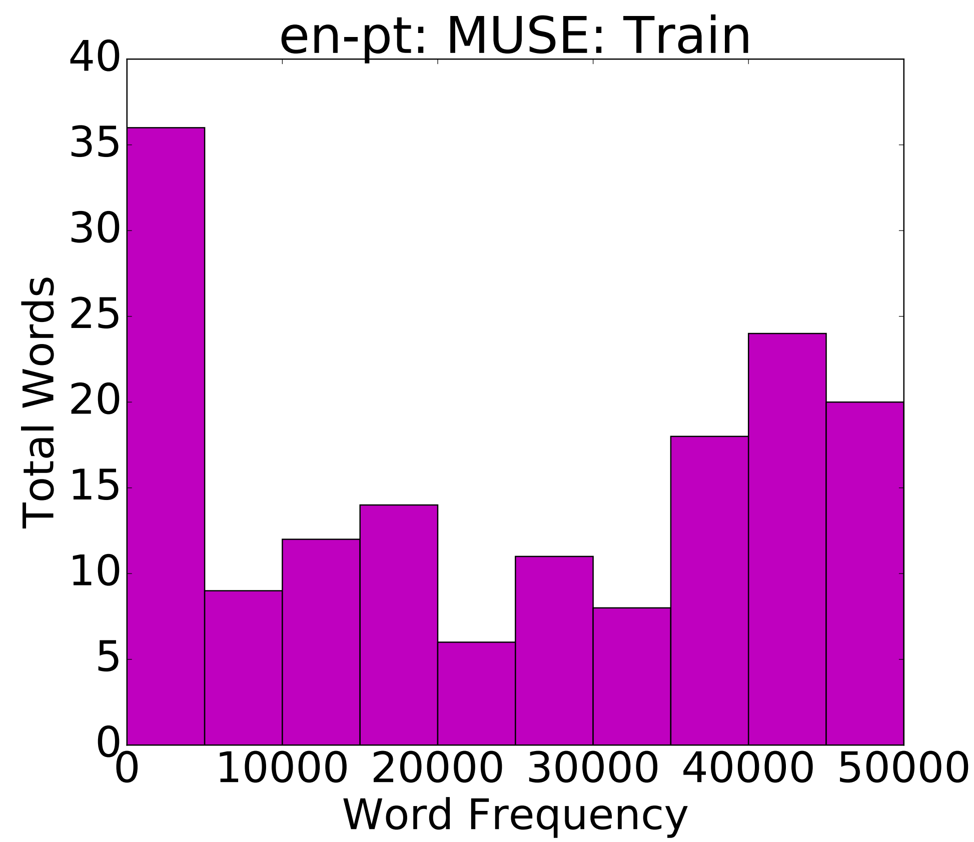
$$h_i = \sigma_1(x_{\mathcal{F}_i}\mathbf{W})$$
$$t_i = \sigma_2(x_{\mathcal{F}_i}\mathbf{W^t})$$
$$\hat{y}_i^{nn} = t_i \times h_i + (1.0 - t_i) \times x_{\mathcal{F}_i}$$

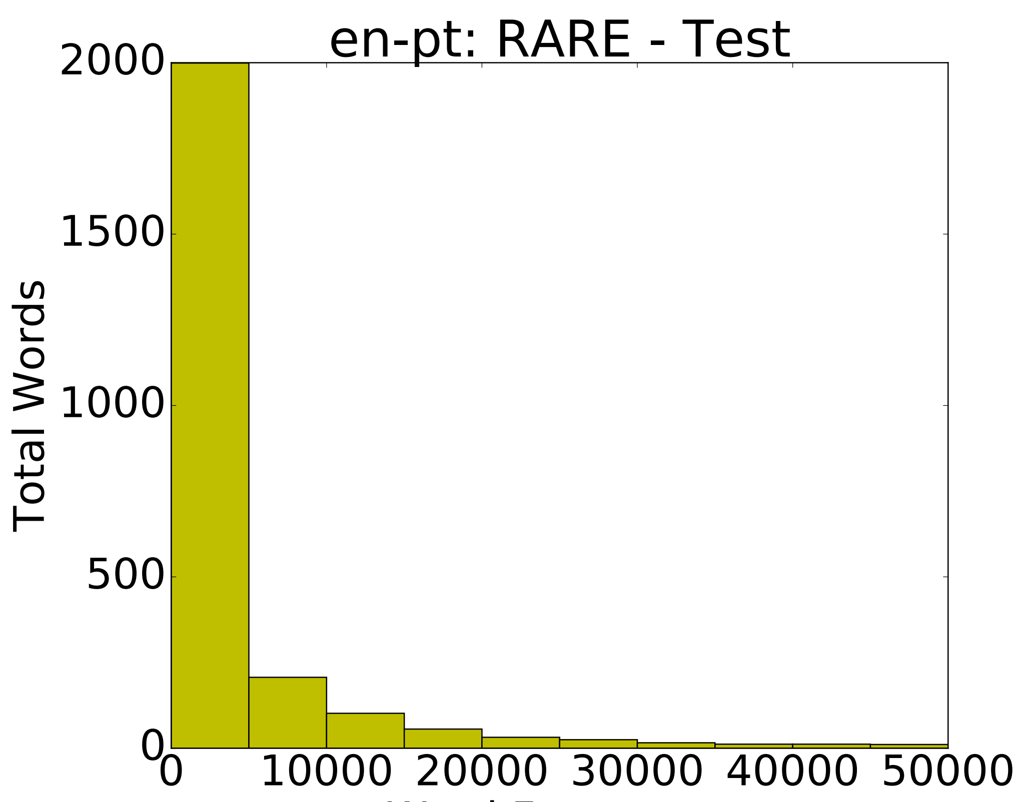
$$L(\theta) = \sum_{i=1}^{m} \sum_{j \neq i}^{k} \max\Big(0, \gamma + d(y_i, \hat{y}_i^g) - d(y_j, \hat{y}_i^g)\Big),$$

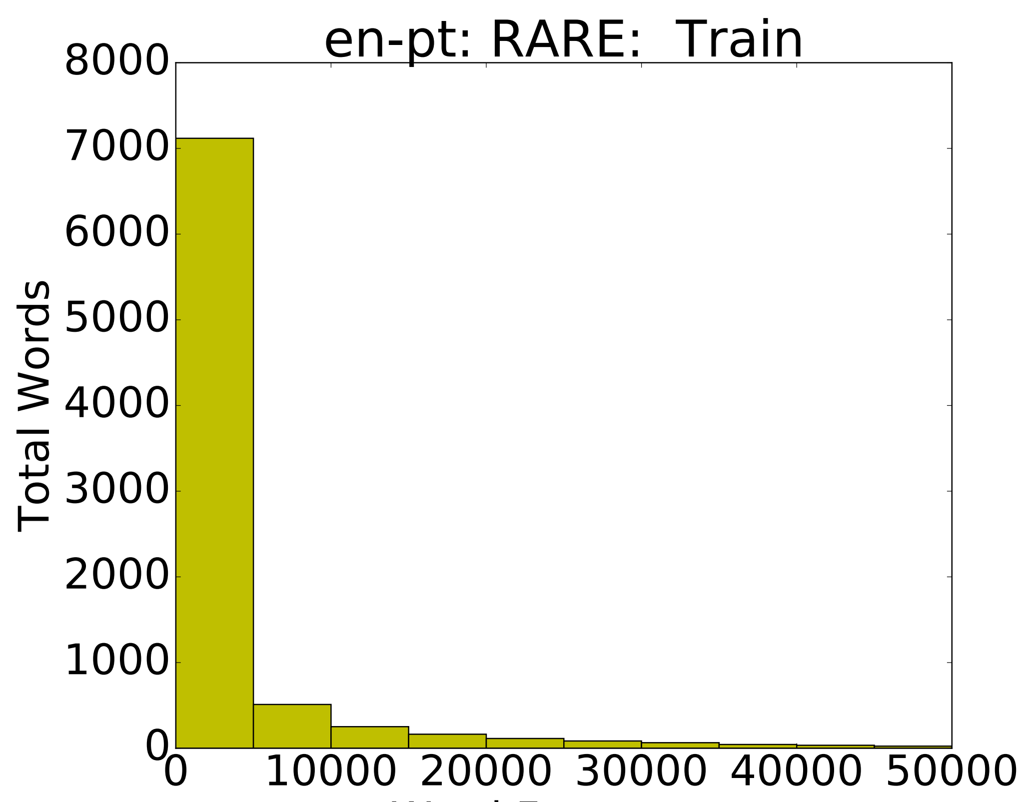
	en-ru	en-zh	en-de	en-es	en-fr
	50.33	**43.27**	68.50	77.47	76.10
Artetxe et al . 2018	47.93	20.4	70.13	**79.6**	**79.30**
Conneau et al. 2018	37.30	30.90	**71.30**	79.10	78.10

Rare Words


en-pt: MUSE: Test
Total Words
Word Frequency
0
5
10
15
20
25
30
35
0
10000
20000
30000
40000
50000


en-pt: MUSE: Train
Total Words
Word Frequency
0
5
10
15
20
25
30
35
40
0
10000
20000
30000
40000
50000


en-pt: RARE - Test
Total Words
0
500
1000
1500
2000
0
10000
20000
30000
40000
50000


en-pt: RARE: Train
Total Words
0
1000
2000
3000
4000
5000
6000
7000
8000
0
10000
20000
30000
40000
50000

Rare vs frequent words: en-pt

	en-pt	
	RARE	MUSE
	57.67	72.60
Artetxe et al . 2018	47.00	**77.73**

Neighborhood interpretability

Neighborhood			
51	134	162	7
drugs zonisamide cocaine ritalin hospitalized pheniprazine overdose disorientation focusyn alfaxalone	criminally judicature prosecutory derogation restitutionary derogative jailable extradition sodomy crimes	chuanyao chuanyan zhiang thanong qiangbing pengpeng nguyan yuning liheng thanong	khoisan bantu sepedi otjiherero ndebeles hereros otjinene shona hutu witotoan

Conclusion

1. Success on close languages

2. Distant languages still far behind

- assumptions responsible?