Meta Text Embeddings — Issues and Use cases

Danushka Bollegala
Professor@University of Liverpool

Meaning Representations

- How do you find the meaning of *Pho*?
  - Look it up in a dictionary
    - *Pho is a Vietnamese soup consisting of broth, rice noodles, herbs and meat.*
  - See how it is being used in the language
    - *I had a delicious Pho yesterday at Hanoi.*
  - Ask someone?
    - *Can you tell me what Pho is?*
  - Look at its pre-trained word embedding??
    - $\overrightarrow{Pho} = (0.31, -0.21, 0.41, \ldots, 0.11)^\top$
  - Just eat it!

Do words have meanings?

Not really. They just borrow meanings from their neighbours
Distributional Hypothesis

J. R. Firth

"You shall know a word by the company it keeps"

Possibly the most widely (over) used hypothesis in NLP

Image credit: www.odlt.org

Quiz

- X is a device that is easy to carry around, you can speak using X, watch the Internet. What could X be?
  - a dog
  - an airplane
  - an iPhone
  - a banana

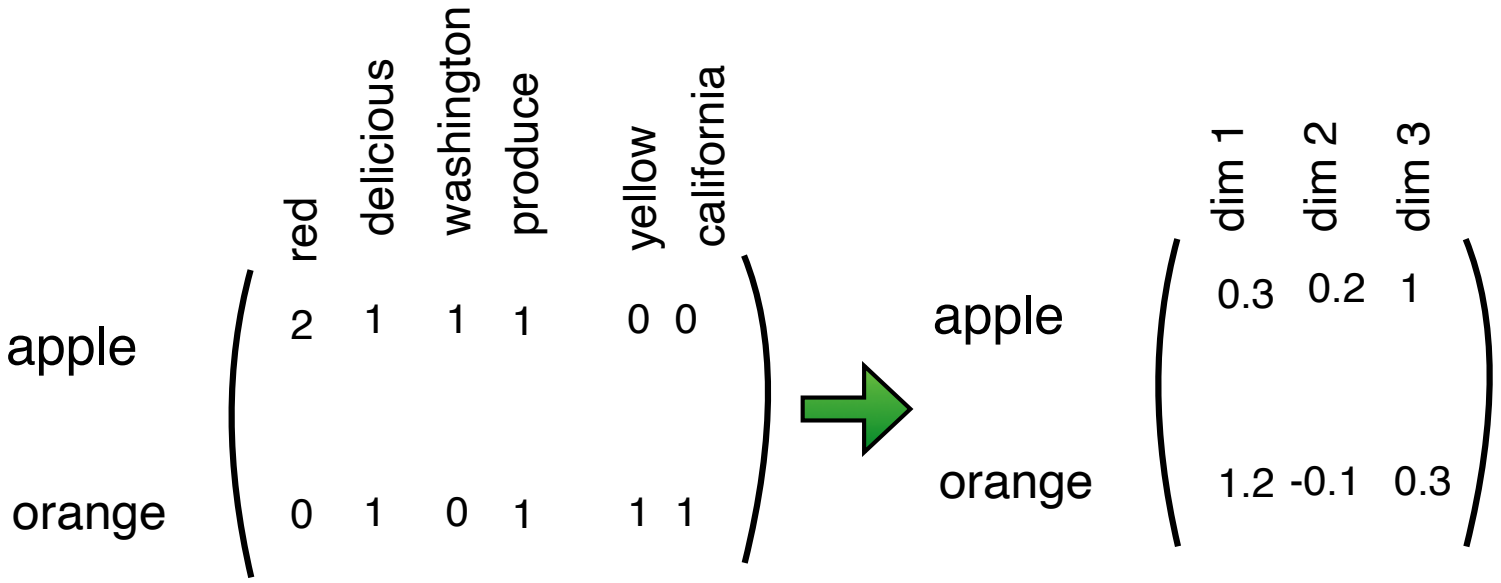
Word Embeddings

- Representing the meaning of words is a fundamental task in NLP
  - Lexical semantics
    - Represent the meaning of individual words
  - Compositional semantics
    - Using word-level representations, compose representations for larger lexical units such as phrases, sentences, or documents.
- Two main approaches
  - *Counting-based* Embeddings
  - *Prediction-based* Embeddings

Counting-based Embeddings

- Let us create a representation for “apple”
- S_1=Apples are red.
- S_2=Red apples are delicious.
- S_3=Apples are produced in Washington.

apple=[(red,2),(delicious,1),(washington,1),(produce,1)]



Prediction-based Embeddings

- A Top-down approach for word embedding learning

    1. Initialise (possibly randomly) word embeddings

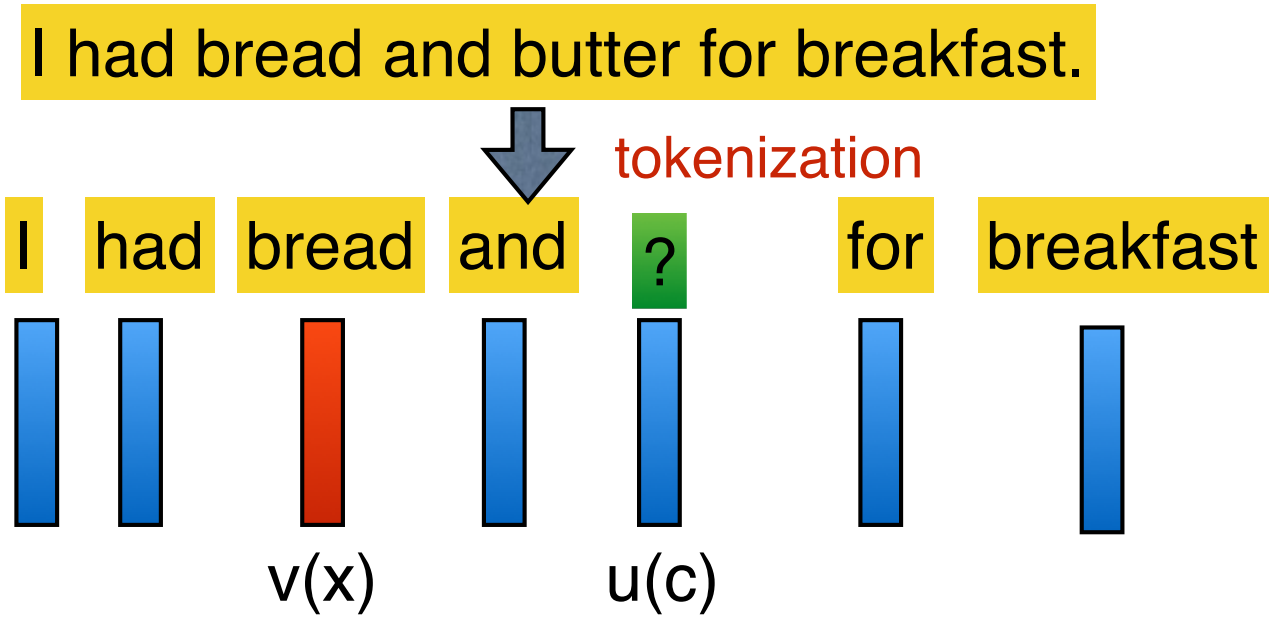
    2. Use the current word embeddings to do some task (preferably unsupervised, if you have lots of labelled data then you may do supervised tasks)

    3. Did we do well on the task used in the previous step?

        1. If yes, terminate. You have good word embeddings.

        2. If not, update your word embeddings and goto step 2.

Example — skip-gram model



(x=bread, c=butter) is more natural than (x=bread, c'=cake) in English.
Can we learn vectors v(x), u(c), and u(c') that encode this knowledge?

Log bi-linear model

probability of observing c in the context of x

co-occurrences

$$p(c|x) = \frac{\exp(\boldsymbol{v}(x)^\top \boldsymbol{u}(c))}{\sum_{c' \in \mathcal{V}} \exp(\boldsymbol{v}(x)^\top \boldsymbol{u}(c'))}$$

Normalize by dividing from the sum taken over all words in the vocabulary.

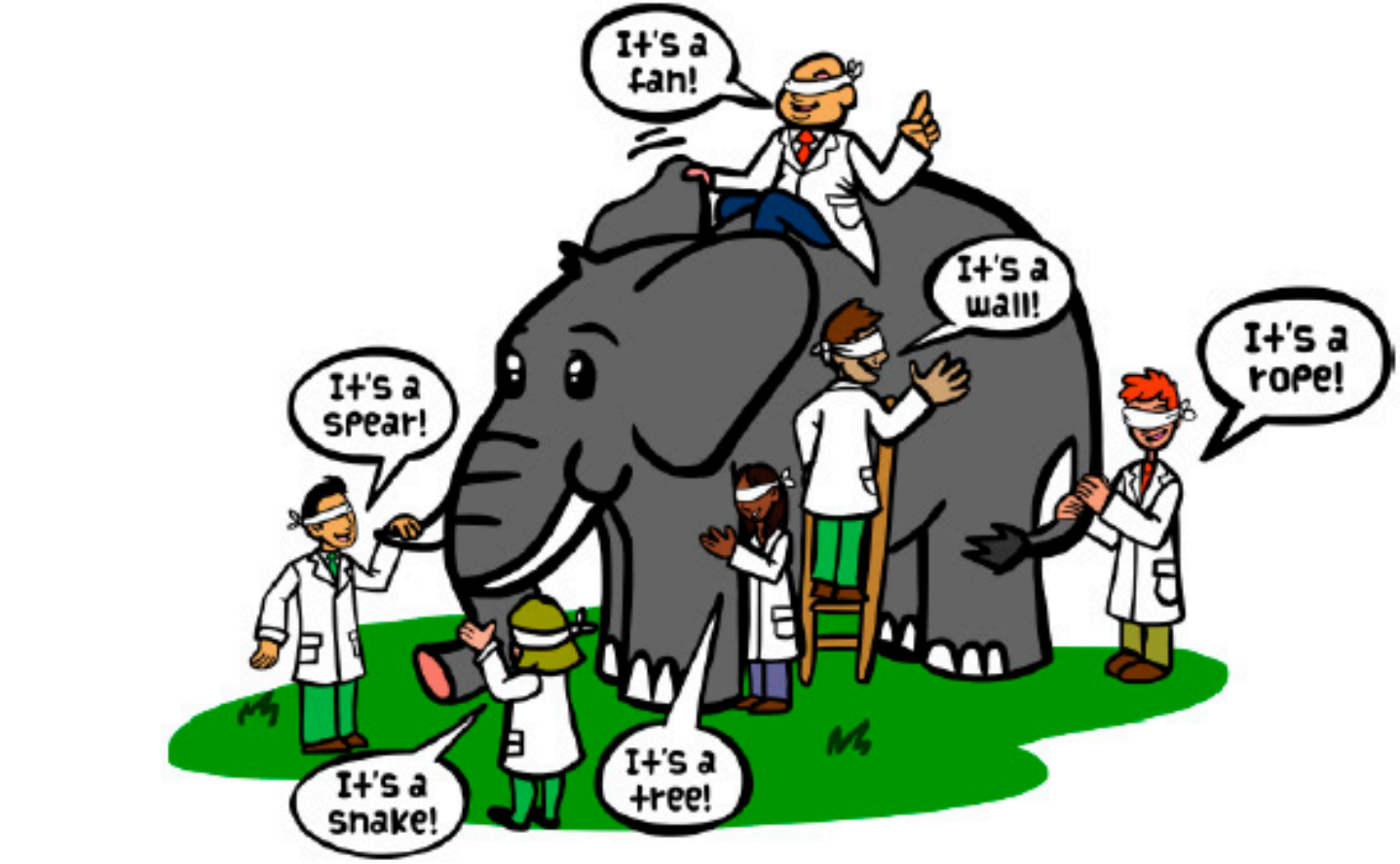
Training is similar to that of logistic regression. If we fix one of u or v, then the function becomes convex in the other. It is similar to solving alternative logistic regression problems.

[Mnih+Hinton ICML'07]

Prediction-based Embeddings

- Word/sub-word embeddings
  - SGNS/CBOW [Mikolov+13],
  - Global Vector Representation (GloVe) [Pennington+14]
  - Structured SGNS/CBOW [Shen+Liu 16]
  - HLBL (Hierarchical log-bilinear embeddings) [Turian+10]
  - FastText [Mikolov+17]
- Knowledge Graph Embedding methods
  - TransE [Bordes+13], TransH [Wang+14], TransD [Ji+15], TransR [Lin+15], DistMult [Yang+15], HolE [Nickel+16], ComplEx [Trouillon+16], TransSparse [Ji+16], STransE [Nguyen+16], RESCAL [Nickel+11], NTN [Socher+13], RelWalk [Bollegala+19]
- Joint approaches:
  - Retro fitting [Faruqui+15], JointReps [Bollegala+16, Mohammed+17]
- Sense embeddings:
  - SCWS [Huang+12], MSSG [Neelakantan+14], EWISE [Kumar+19], Resinger+Mooney [2010], ...
- Contextualised embeddings [BERT, Elmo, CoVe, XLNet, RoBERTa, ELECTRA, ALBERT, ...]
  -  

Who is correct?



Meta Embedding Learning

- Different word embeddings capture different aspects of lexical semantics such as sense, context, morphology, etc.
- Some word embeddings are good for some tasks, whereas other word embeddings are good for other tasks.
- It is time consuming to try each and every embedding available for a given task.

Research Question

Can we benefit from the complementary strengths in multiple pre-trained word embeddings, without having to re-train the **source** word embeddings, to create better **meta** word embeddings?

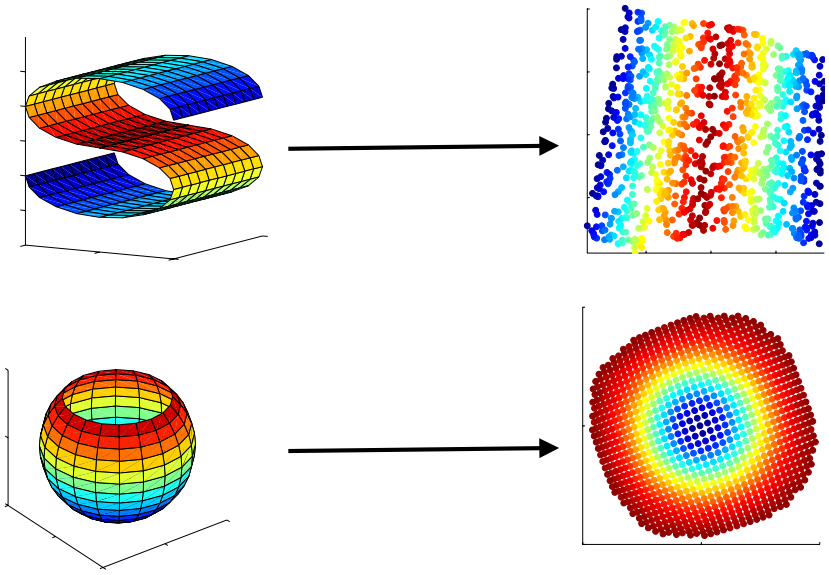
Meta-Embedding Learning

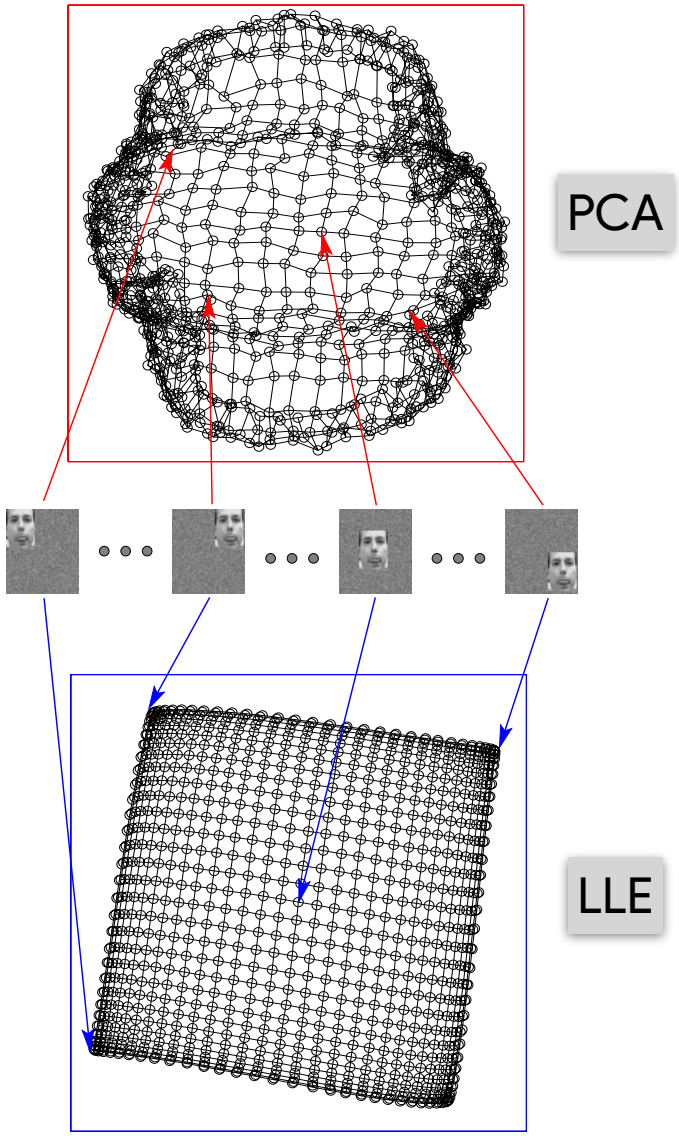
- Problem setting
  - We are given a set of pre-trained (source) word embeddings for a vocabulary.
  - The source embeddings may have been learnt using different algorithms, from different language resources and/or can have different dimensionalities.
  - We do not assume any knowledge about the algorithms used to train the source embeddings.
  - We do not assume the availability of the language resources used to train the source embeddings.
  - We will not re-train any of the source embeddings (why? expensive, access to hardware, time consuming, etc.)
- Task
  - Taking pre-trained source embeddings as the input we must learn more accurate and wide-coverage meta embeddings than the individual source embeddings

Locally Linear Embedding

- Locally Linear Embedding (LLE) [Saul+Rowies 03]
- Nearest neighbour reconstruction step.
  - For each point, construct its representation as the linearly weighted sum of the representations of its nearest neighbours.
  - Nearest neighbour computation followed by solving a set of linear equations.
- Lower-dimensional projection step.
  - Project the weights computed in the previous step such that points that are close to each other in the source space are projected proximally in the target space.
  - Find the smallest eigenvalues of a matrix

Examples of LLE

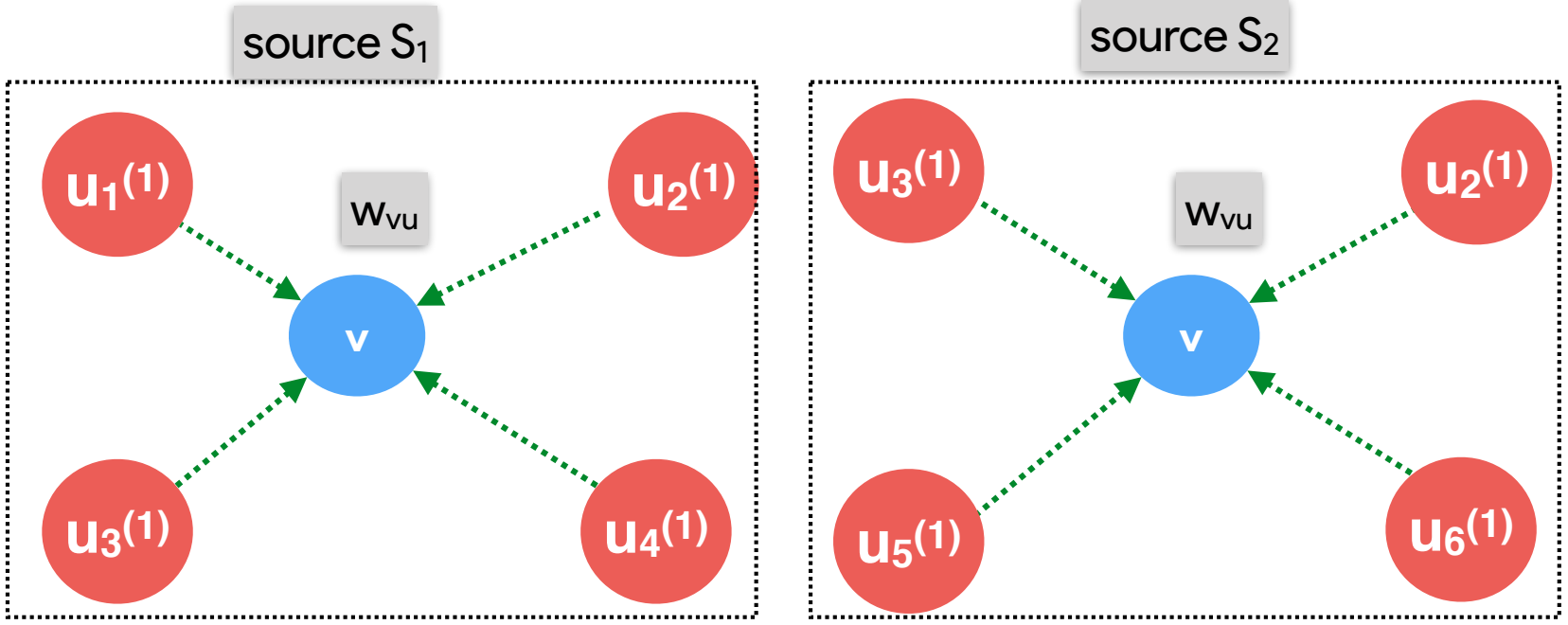




Locally Linear Meta Embeddings

Given two sets of source embeddings S_1 and S_2, we compute the neighbourhood reconstruction weights w_{vu} by minimising the following objective.

$$\Phi(\mathbf{W}) = \sum_{i=1}^{2} \sum_{v \in \mathcal{V}} \left\| \boldsymbol{v}^{(i)} - \sum_{u \in \mathcal{N}_i(v)} w_{vu} \boldsymbol{u}^{(i)} \right\|_2^2$$



Reconstruction weights are common across the embedding spaces.

Bollegala+ IJCAI-18

Neighbourhood Reconstruction

- We compute the gradient of the objective and apply stochastic gradient descent (SGD) to find the optimal weights.
- Weights are randomly initialised
- Ball Tree Algorithm (Complexity O($n$log(n)) is used to compute the k-NN.
- Optimisation converges quickly (ca. after 5 iterations)

Projection to Meta-Embedding Space

$$\Psi(\mathcal{P}) = \sum_{v \in \mathcal{V}} \left\| \boldsymbol{v}^{(\mathcal{P})} - \sum_{u \in \mathcal{N}_{\mathcal{S}}(v)} w_{vu} \boldsymbol{u}^{(\mathcal{P})} - \sum_{u \in \mathcal{N}_{\mathcal{T}}(v)} w_{vu} \boldsymbol{u}^{(\mathcal{P})} \right\|_2^2$$

meta-embeddings to be learnt

reconstruction weights learnt in the previous step

Meta-embeddings can be learnt by the eigendecomposition of the matrix M given by

$$\mathbf{M} = (\mathbf{I} - \mathbf{W}')^\top (\mathbf{I} - \mathbf{W}')$$

$$w'_{vu} = \left(\mathbb{I}[u \in \mathcal{N}_{\mathcal{S}}(v)] + \mathbb{I}[u \in \mathcal{N}_{\mathcal{T}}(v)]\right) w_{vu}$$

The required k projection dimensions are given by the smallest k+1 (first one is null vector) eigenvectorsof M.

Experiments

- Five source word embeddings are used in the experiments
  - HLBL (Hierarchical log-bilinear embeddings) [Turian+10]
  - Huang (Sense-sensitive embeddings) [Huang+12]
  - GloVe (Global Vector Prediction) [Pennington+14]
  - CW (Rank-loss embeddings) [Collobert+Weston+08]
  - CBOW (Continuous Bag-of-Words) [Mikolov+13]
- Covering 2.8m words.

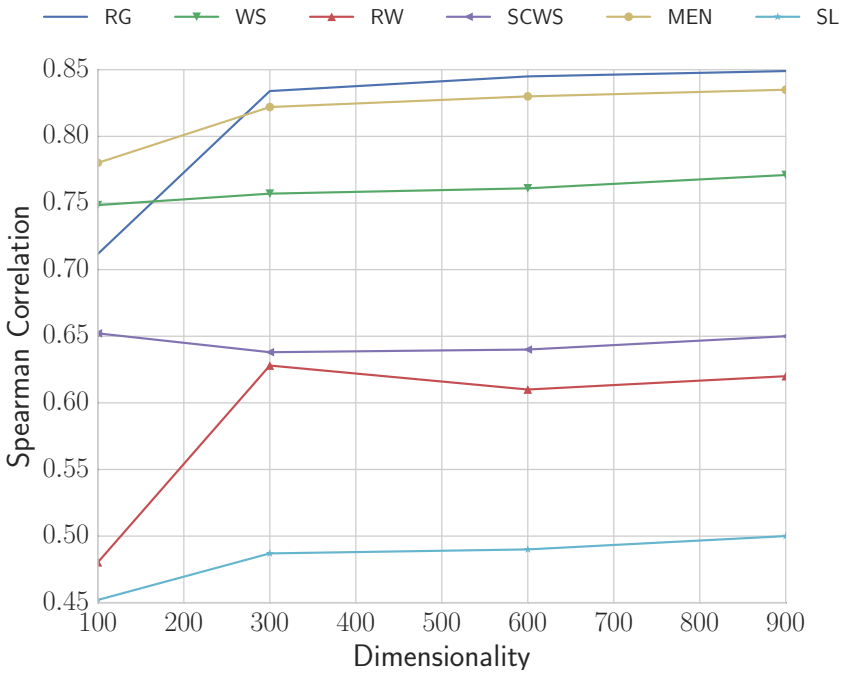
Tasks

- Semantic similarity measurement
  - Measure cosine similarity between two words using their word embeddings and compare that against human similarity ratings using Sperman correlation coefficient.
- Word analogy detection
  - Find a word that completes an analogy such as Tokyo is to Japan as Hanoi is to X. Use **cos(Japan - Tokyo + Hanoi, X)** to score each candidate X.
- Relation classification
  - Predict the semantic relation type between two words in a word-pair (x,y), where the word-pair is represented by the vector **x-y**. Train a k-NN classifier to predict the relations for the test word-pairs.
- Short-text classification
  - Classify sentiment (binary) of a given short-text. Use the centroid of word embeddings to represent the text and train a binary logistic regression classifier.

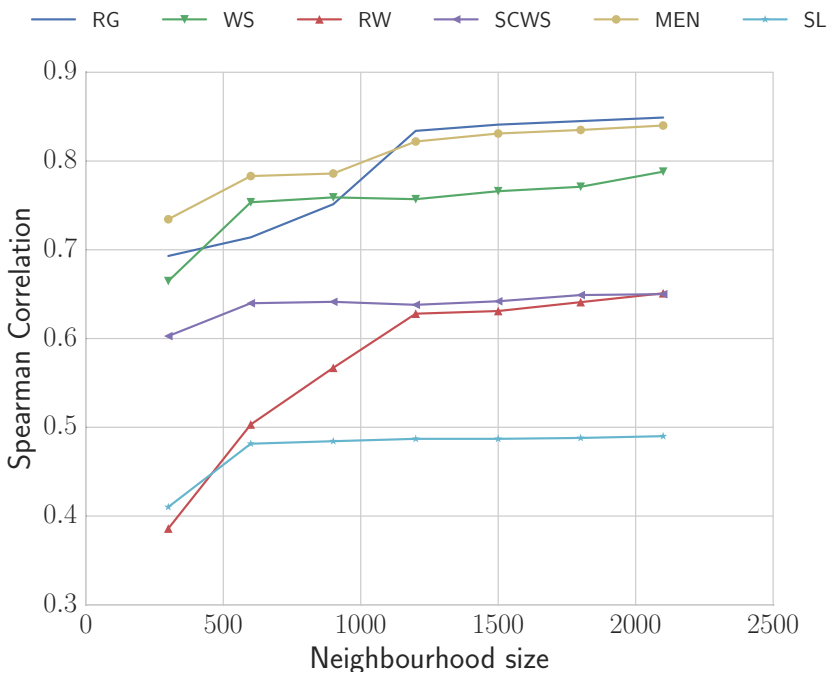
Results

		Model	RG	MC	WS	RW	SCWS	MEN	SL	GL	SE	DV	SA	MR
sources	1	GloVe	81.7	80.8	64.3	38.4	54.0	74.3	37.4	70.5	39.9	87.7	73.4	70.0
	2	CBOW	76.0	82.2	69.8	53.4	53.4	78.2	44.2	75.2	39.1	87.4	73.6	71.0
	3	HLBL	35.3	49.3	35.7	19.1	47.7	30.7	22.1	16.6	34.8	72.0	62.6	61.6
	4	Huang	51.3	58.8	58.0	36.4	63.7	56.1	21.7	8.3	35.2	76.0	64.8	60.9
	5	CW	29.9	34.3	28.4	15.3	39.8	25.7	15.6	4.7	34.6	75.6	62.7	61.4
ablation	6	CONC (-GloVe)	75.0	79.0	70.0	55.3	62.9	77.7	41.5	64.0	38.7	82.9	72.1	69.1
	7	CONC (-CBOW)	80.8	81.0	65.2	46.0	56.3	74.9	37.3	70.0	38.8	86.0	71.6	69.9
	8	CONC (-HLBL)	83.0	84.0	71.9	53.4	61.4	80.1*	41.6	72.7	39.5	84.9	71.0	69.4
	9	CONC (-Huang)	83.0	84.0	71.6	48.8	60.8	80.1*	41.9	72.8	40.0	86.7	71.2	69.1
	10	CONC (-CW)	82.9	84.0	71.9	53.3	61.6	80.2*	41.6	72.6	39.6	84.9	72.3	69.9
	11	SVD (-GloVe)	78.6	79.9	68.4	53.9	61.6	77.5	40.1	61.7	38.5	84.1	71.6	69.8
	12	SVD (-CBOW)	80.5	81.2	64.4	45.3	55.3	74.2	35.7	70.9	38.7	86.7	73.4	69.1
	13	SVD (-HLBL)	82.7	83.6	70.3	52.6	60.1	79.9*	39.6	73.5	39.8	87.3	73.2	70.4
	14	SVD (-Huang)	82.5	85.0	70.3	48.6	59.8	79.9*	39.9	73.7	40.0	87.3	73.5	70.8
	15	SVD (-CW)	82.5	83.9	70.4	52.5	60.1	80.0*	39.7	73.3	39.8	87.2	73.1	70.7
	16	Proposed (-GloVe)	79.8	79.7	71.1	54.7	62.3	78.2	46.1	84.2*	39.8	85.4	72.2	70.2
	17	Proposed (-CBOW)	80.9	82.1	67.4	58.7*	58.7	75.7	45.2	85.2*	40.1	87.1	73.8	70.1
	18	Proposed (-HLBL)	82.1	86.1	71.3	58.3*	62.1	81.9*	34.8	86.3*	40.3	87.7	73.7	71.1
	19	Proposed (-Huang)	81.2	85.2	73.1	55.1	63.7	81.4*	42.3	82.6*	41.1	87.5	73.9	71.2
	20	Proposed (-CW)	83.1	84.8	72.5	58.5	62.3	81.1*	43.5	88.4*	41.9	87.8	71.6	71.1
ensemble	21	CONC	82.9	84.1	71.9	53.3	61.5	80.2*	41.6	72.9	39.6	84.9	72.4	69.9
	22	SVD	82.7	83.9	70.4	52.6	60.0	79.9*	39.7	73.4	39.7	87.2	73.4	70.7
	23	1TON	80.7	80.7	74.5	60.1*	61.6	73.5	46.4	76.8	42.3*	87.6	73.8	70.3
	24	1TON+	82.7	85.0	75.3	61.6*	60.2	74.1	46.3	77.0	40.1	83.9	73.9	69.2
	25	Proposed	**83.4**	**86.2**	**75.7***	**62.8***	**63.8**	**82.2***	**48.7**	**89.9***	**43.1***	**88.7***	**74.0**	**71.3**

Parameter sensitivity



Effect of dimensionality on semantic similarity prediction performance



Effect of neighbourhood size on semantic similarity prediction performance

Meta-embedding by Averaging

- Concatenation increases the dimensionality of the meta-embedding space.
- Can we “average” the source embeddings instead?
  - At first sight NO, because different source embeddings live in different vector spaces
- However, this turns out to be YES if the expected difference between two word embeddings in a source is a random vector.
- *Frustratingly Easy Meta-Embedding — Computing Meta-Embeddings by Averaging Source Word Embeddings* [Coates+Bollegala NAACL-18]

Concat. vs. Avg.

- Consider two source embedding spaces S_1 and S_2 with dimensionalities respectively d_{S1} and d_{S2}. We zero-pad the embeddings of two words u and v and compute their concat as their sum

$$\boldsymbol{u}_{\mathcal{S}_1}^{zero} + \boldsymbol{u}_{\mathcal{S}_2}^{zero} = \begin{bmatrix} \boldsymbol{u}_{\mathcal{S}_2(1)} \\ \boldsymbol{u}_{\mathcal{S}_2(2)} \\ \vdots \\ \boldsymbol{u}_{\mathcal{S}_2(d_{\mathcal{S}_2})} \\ \boldsymbol{u}_{\mathcal{S}_1(1)} \\ \boldsymbol{u}_{\mathcal{S}_1(2)} \\ \vdots \\ \boldsymbol{u}_{\mathcal{S}_1(d_{\mathcal{S}_1})} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_{\mathcal{S}_2} \\ \boldsymbol{u}_{\mathcal{S}_1} \end{bmatrix}$$

$$E_{\mathcal{S}_1} = ||\boldsymbol{u}_{\mathcal{S}_1} - \boldsymbol{v}_{\mathcal{S}_1}||_2 = \left|\left|\boldsymbol{u}_{\mathcal{S}_1}^{zero} - \boldsymbol{v}_{\mathcal{S}_1}^{zero}\right|\right|_2$$
$$E_{\mathcal{S}_2} = ||\boldsymbol{u}_{\mathcal{S}_2} - \boldsymbol{v}_{\mathcal{S}_2}||_2 = \left|\left|\boldsymbol{u}_{\mathcal{S}_2}^{zero} - \boldsymbol{v}_{\mathcal{S}_2}^{zero}\right|\right|_2$$

- Because the zero-padded vectors are orthogonal we can write the Euclidean distance in the concat (meta) space as follows

$$\begin{aligned} & E_{CONC} \\ &= \left|\left| \begin{bmatrix} \boldsymbol{u}_{\mathcal{S}_2} \\ \boldsymbol{u}_{\mathcal{S}_1} \end{bmatrix} - \begin{bmatrix} \boldsymbol{v}_{\mathcal{S}_2} \\ \boldsymbol{v}_{\mathcal{S}_1} \end{bmatrix} \right|\right|_2 \\ &= \left|\left|(\boldsymbol{u}_{\mathcal{S}_1}^{zero} + \boldsymbol{u}_{\mathcal{S}_2}^{zero}) - (\boldsymbol{v}_{\mathcal{S}_1}^{zero} + \boldsymbol{v}_{\mathcal{S}_2}^{zero})\right|\right|_2 \\ &= \left|\left|(\boldsymbol{u}_{\mathcal{S}_1}^{zero} - \boldsymbol{v}_{\mathcal{S}_1}^{zero}) - (\boldsymbol{v}_{\mathcal{S}_2}^{zero} - \boldsymbol{u}_{\mathcal{S}_2}^{zero})\right|\right|_2 \\ &= \sqrt{(E_{\mathcal{S}_1})^2 + (E_{\mathcal{S}_2})^2 - 2E_{\mathcal{S}_1}E_{\mathcal{S}_2}cos(\theta)} \\ &= \sqrt{(E_{\mathcal{S}_1})^2 + (E_{\mathcal{S}_2})^2 - 2E_{\mathcal{S}_1}E_{\mathcal{S}_2}(0)} \\ &= \sqrt{(E_{\mathcal{S}_1})^2 + (E_{\mathcal{S}_2})^2} \end{aligned}$$

Concat. vs. Avg.

- Similarly, the Euclidean distance in the average (meta) space can be written as

$$
\begin{aligned}
& E_{AVG} \\
&= \left\| \frac{(\boldsymbol{u}_{\mathcal{S}_1} + \boldsymbol{u}_{\mathcal{S}_2})}{2} - \frac{(\boldsymbol{v}_{\mathcal{S}_1} + \boldsymbol{v}_{\mathcal{S}_2})}{2} \right\|_2 \\
&= \frac{1}{2} \left\| (\boldsymbol{u}_{\mathcal{S}_1} - \boldsymbol{v}_{\mathcal{S}_1}) - (\boldsymbol{v}_{\mathcal{S}_2} - \boldsymbol{u}_{\mathcal{S}_2}) \right\|_2 \\
&\propto \sqrt{(E_{\mathcal{S}_1})^2 + (E_{\mathcal{S}_2})^2 - 2E_{\mathcal{S}_1} E_{\mathcal{S}_2} \cos(\theta)}
\end{aligned}
$$

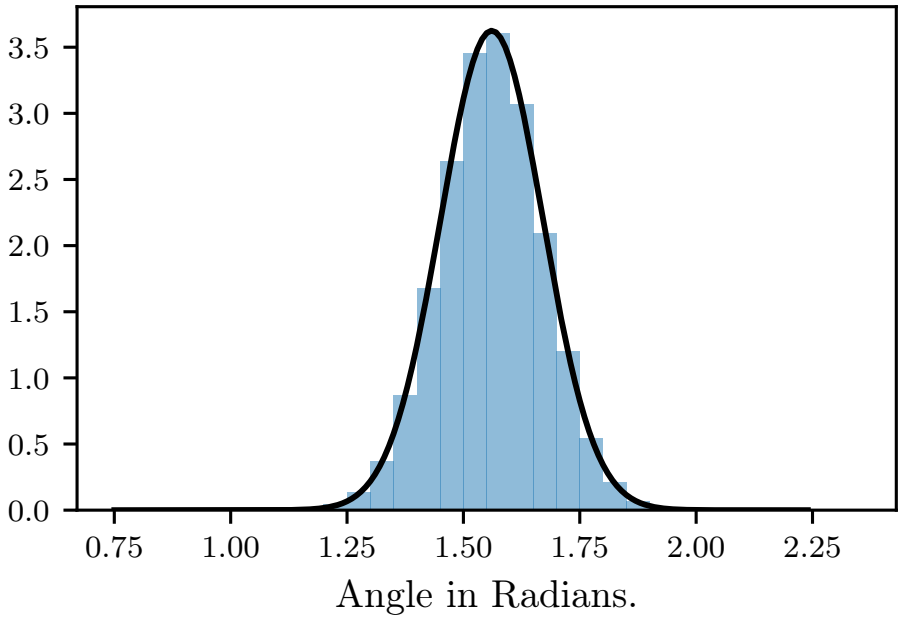
Theorem (Cai+2013)

Let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independently chosen random points with the uniform distribution on the p-dimensional unit sphere $\mathbf{S}^{p-1}$ in $\mathbf{R}^p$. The angle between $\mathbf{X}_i$ and $\mathbf{X}_j$ is $\Theta_{ij} (\in [0,\pi])$. Let the empirical distribution μ_n of the angles $\Theta_{ij}, 1 \le i < j \le n$, defined as

$\mu_n = \frac{1}{\binom{n}{2}} \sum_{1 \le i < j \le n} \delta_{\Theta_{ij}}, n \ge 2$. Then, as $n \to \infty$ with probability one, μ_n converges weakly

to the distribution with density $h(\theta) = \frac{1}{\pi} \frac{\Gamma\left(\frac{p}{2}\right)}{\Gamma\left(\frac{p-1}{2}\right)} (\sin\theta)^{p-2}, \theta \in [0,\pi]$

Orthogonality



Distribution of angle between CBOW and GloVe embeddings for a randomly selected set of words common to both embeddings.

Embeddings	μ	σ^2
GloVe & CBOW	1.5609	0.0121
GloVe & HLBL	1.5709	0.0129
CBOW & HLBL	1.5740	0.0126

$90^0 = \frac{\pi}{2}$ radian = 1.5707

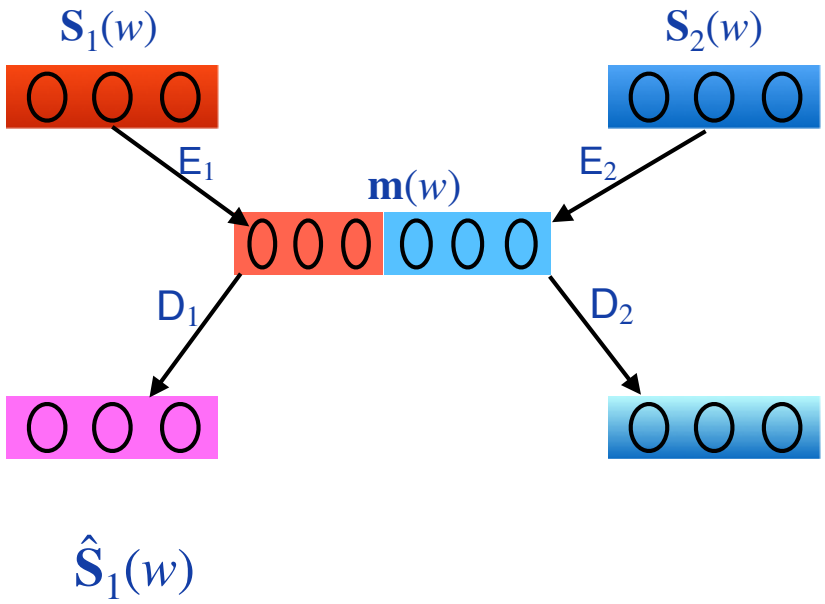
Conct. vs Avg

Embeddings	RG	MC	WS	RW	SL	GL
sources						
HLBL 100	35.3	49.3	35.7	19.1	22.1	15.0
CBOW 300	76.0	82.2	69.8	53.4	44.2	67.1
GloVe 300	82.9	87.0	75.4	48.7	45.3	68.7
AVG						
CBOW+HLBL 300	69.2	81.0	60.1	48.7	37.3	49.4
GloVe+CBOW 300	82.2	87.0	74.5	52.9	**46.5**	73.8
GloVe+HLBL 300	73.7	74.1	64.2	44.6	38.8	49.5
CONC						
CBOW+HLBL 400	68.7	80.2	62.9	49.1	39.6	53.2
GloVe+CBOW 600	**83.0**	**88.8**	**76.4**	**54.8**	46.3	**75.5**
GloVe+HLBL 400	73.7	80.1	65.5	46.4	40.0	53.8

Results on word similarity (RG, MC, WS, RW, SL) and analogy (GL) tasks. Best performances are bolded per task. Dimensionaltiy of the meta-embedding space is shown next to the source names.

Meta-Embedding as Autoencoding

- In meta-embedding, we would like to learn intermediate representations that can preserve/reconstruct the source embeddings.
- This can be seen as an autoencoder learning problem
- Given two sources S_1 and S_2, we can learn encoders E_1, E_2 and decoders D_1, D_2 following different strategies.
  - Decoupled Autoencoded Meta-Embedding (DAEME)



$$\boldsymbol{m}(w) = E_1(\boldsymbol{s}_1(w)) \oplus E_2(\boldsymbol{s}_2(w))$$

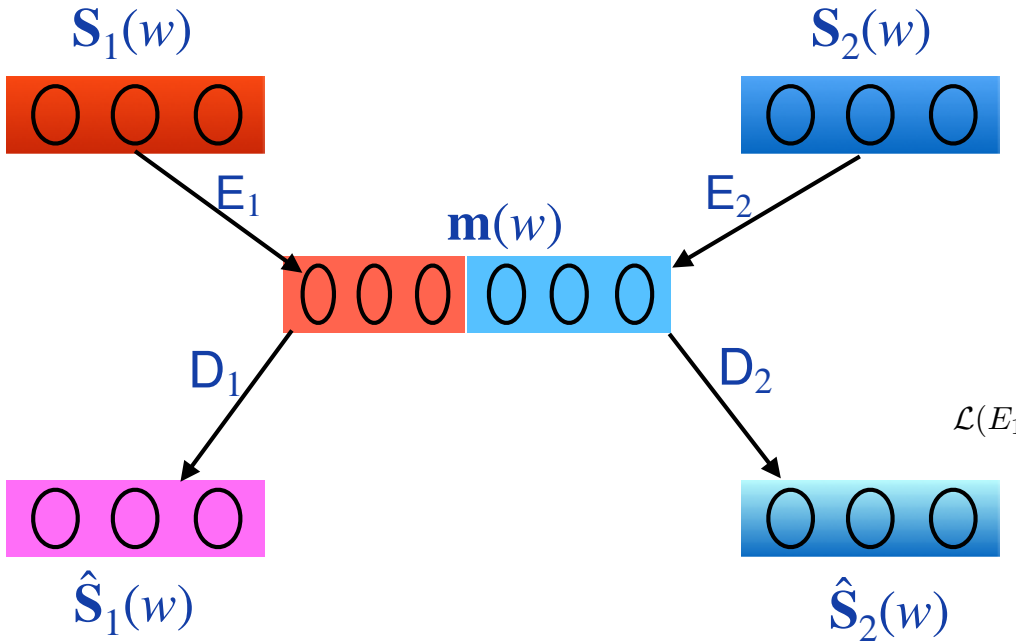
$$\hat{\boldsymbol{s}}_1(w) = D_1(E_1(\boldsymbol{s}_1(w)))$$
$$\hat{\boldsymbol{s}}_2(w) = D_2(E_2(\boldsymbol{s}_2(w)))$$

$$\mathcal{L}(E_1, E_2, D_1, D_2) = \sum_{w \in \mathcal{V}} \Big(\lambda_1 \, ||E_1(s_1(w)) - E_2(s_2(w))||^2 + \lambda_2 \, ||\hat{\boldsymbol{s}}_1(w) - \boldsymbol{s}_1(w)||^2 + \lambda_3 \, ||\hat{\boldsymbol{s}}_2(w) - \boldsymbol{s}_2(w)||^2 \Big)$$

Learning Word Meta-Embeddings by Autoencoding [Bao+Bollegala COLING-18]

Meta-Embedding as Autoencoding

- In meta-embedding, we would like to learn intermediate representations that can preserve/reconstruct the source embeddings.
- This can be seen as an autoencoder learning problem
- Given two sources S_1 and S_2, we can learn encoders E_1, E_2 and decoders D_1, D_2 following different strategies.
  - Concatenated Autoencoded Meta-Embedding (CAEME)



$$\boldsymbol{m}(w) = E_1(\boldsymbol{s}_1(w)) \oplus E_2(\boldsymbol{s}_2(w))$$

$$\hat{\mathbf{s}}_1(w) = D_1(m(w))$$

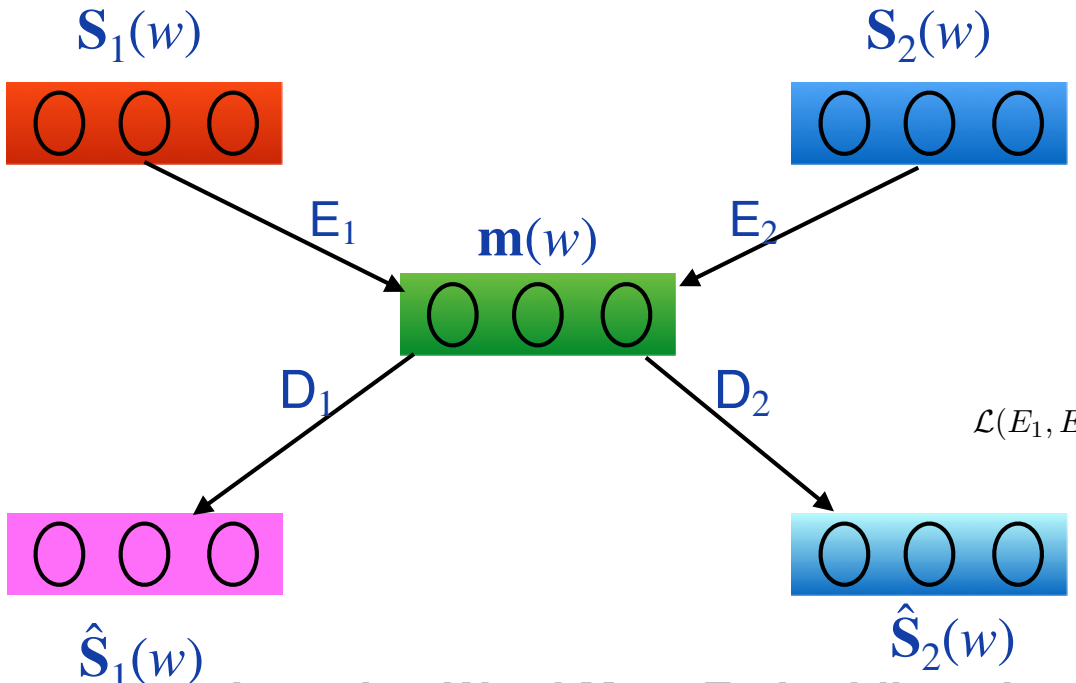
$$\hat{\mathbf{s}}_2(w) = D_2(m(w))$$

$$\mathcal{L}(E_1, E_2, D_1, D_2) = \sum_{w \in \mathcal{V}} \left(\lambda_1 \, ||\hat{\boldsymbol{s}}_1(w) - \boldsymbol{s}_1(w)||^2 + \lambda_2 \, ||\hat{\boldsymbol{s}}_2(w) - \boldsymbol{s}_2(w)||^2 \right)$$

Learning Word Meta-Embeddings by Autoencoding [Bao+Bollegala COLING-18]

Meta-Embedding as Autoencoding

- In meta-embedding, we would like to learn intermediate representations that can preserve/reconstruct the source embeddings.
- This can be seen as an autoencoder learning problem
- Given two sources S_1 and S_2, we can learn encoders E_1, E_2 and decoders D_1, D_2 following different strategies.
  - Averaged Autoencoded Meta-Embedding (AAEME)



$$\boldsymbol{m}(w) = \frac{E_1(\boldsymbol{s}_1(w)) + E_2(\boldsymbol{s}_2(w))}{||E_1(\boldsymbol{s}_1(w)) + E_2(\boldsymbol{s}_2(w))||_2}$$

$$\hat{\mathbf{s}}_1(w) = D_1(m(w))$$

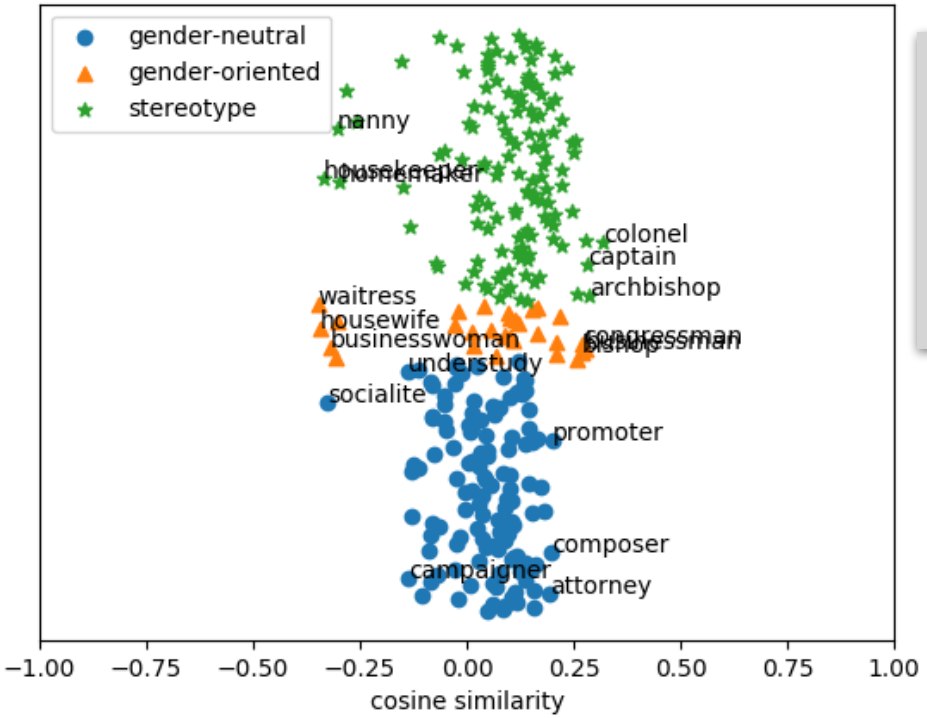
$$\hat{\mathbf{s}}_2(w) = D_2(m(w))$$

$$\mathcal{L}(E_1, E_2, D_1, D_2) = \sum_{w \in \mathcal{V}} \left(\lambda_1 \, ||\hat{\boldsymbol{s}}_1(w) - \boldsymbol{s}_1(w)||^2 + \lambda_2 \, ||\hat{\boldsymbol{s}}_2(w) - \boldsymbol{s}_2(w)||^2 \right)$$

Learning Word Meta-Embeddings by Autoencoding [Bao+Bollegala COLING-18]

Bias in Word Embeddings

- Word embeddings contain unfair social biases such as gender, racial and religious biases.



Cosine similarity between the vector ***he*** - ***she*** and the GloVe word embedding of each word. We see that *gender-neutral* professions such as *captain, colonel, composer, attorney,* etc. have positive (biased towards male gender) scores.

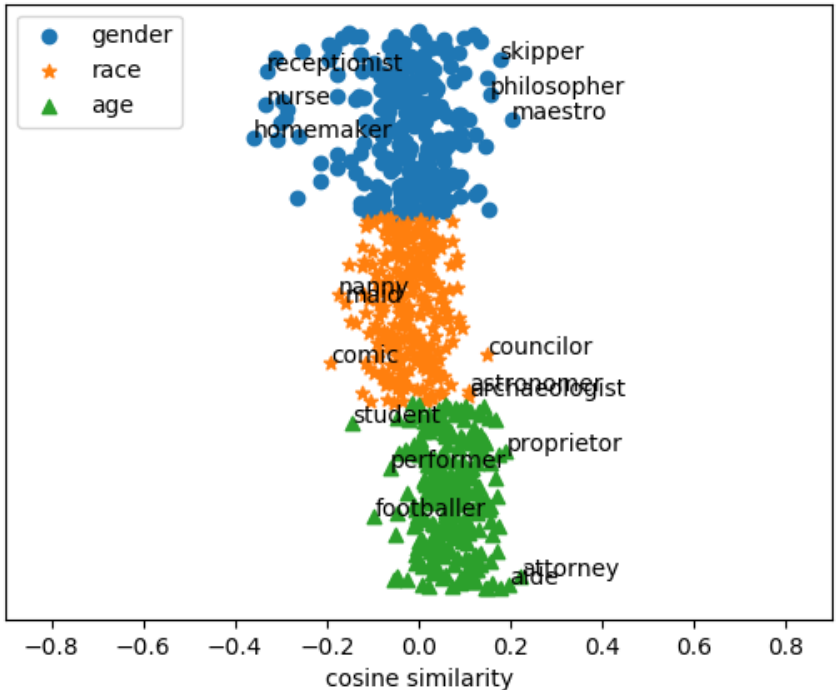
How can we remove biases?

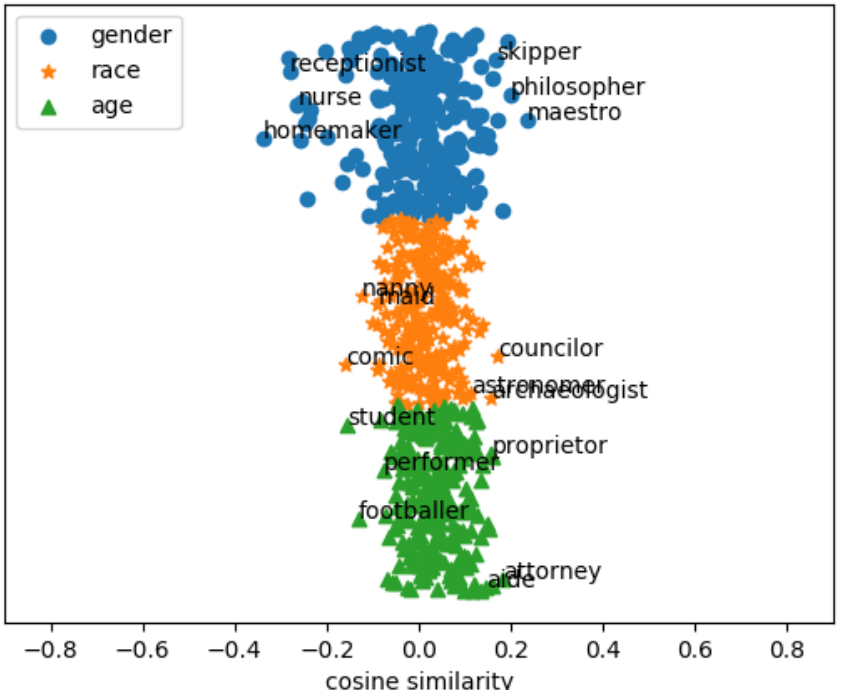
- Gender-Preserving Debiasing [Kaneko+Bollegala ACL'19]
  - Fine-tune pretrained static word embeddings such that the following constraints are *jointly* satisfied.
  - For feminine words, require that they are similar to the female direction.
  - For masculine words, require that they are similar to the male direction.
  - For gender-neutral words, require that they are equally similar to both male and female directions.
  - For stereotypical words, require that they do not have any similarities with both genders.

How can we remove biases?

- Dictionary-based Debiasing [Kaneko+Bollegala, COLING'20]
  - Dictionary-definitions of words can be used as bias-free text for debiasing word embeddings
  - **Semantic Presevation**
    - Do not change the pretrained embeddings a lot during debiasing
  - **Dictionary agreement**
    - Debiased word embeddings must be similar to their dictionary definitions

Results





Cosine similarity between neutral occupation words for vector directions on gender (***he*** - ***she***), race (***Caucasoid*** - ***Negroid***) and age (***elder*** - ***youth***) vectors

My amazing meta-embedders...

K. Hayashi
PFI

J. Coates
UoL/PhD

K. Kawayabayashi
Prof@NII

C. Bao
UCL/MSc

T. Zhang
UoL/UG

J. O'Neill
UoL/PhD

References

[1] Cong Bao and Danushka Bollegala. Learning word meta-embeddings by autoencoding. In Proc. of the 27th International Conference on Computational Linguistics (COLING), pages 1650–1661, 2018.

[2] Danushka Bollegala and Cong Bao. Learning word meta-embeddings by autoencoding. In Proceedings of the 27th International Conference on Computational Linguistics, pages 1650–1661, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[3] Danushka Bollegala, Kohei Hayashi, and Ken-ichi Kawarabayashi. Think globally, embed locally — locally linear meta-embedding of words. In Proc. of IJCAI-EACI, pages 3970–3976, 2018.

[4] Joshua Coates and Danushka Bollegala. Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. In Proc. of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 194–198, 2018.

[5] Masahiro Kaneko and Danushka Bollegala. Gender-preserving debiasing for pre-trained word embed- dings. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1641–1650, Florence, Italy, 2019. Association for Computational Linguistics.

[6] Masahiro Kaneko and Danushka Bollegala. Dictionary-based debiasing of pre-trained word embeddings. In Proc. of the 16th European Chapter of the Association for Computational Linguistics (EACL, 2021.

Thank You

メター⊕ meta ⊕ मेटा ⊕ 元 ⊕ میتا ⊕ મેટા ⊕ మెటా ⊕ මෙටා


Danushka Bollegala
http://danushka.net

danushka@liverpool.ac.uk

@Bollegala