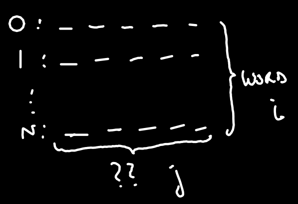
X:



Word Embedding Matrices

X, Z

X: row X_i / Z_i = embedding of i'th word in language.

Steps

- normalize the embeddings.
- unsupervised initialization scheme to create initial solution
- self learning procedure to iteratively improve solution
- Final refinement step.

Goal:
- learn linear transformation matrices W_x, W_z s.t. XW_x, ZW_z in same space.
- Build a dictionary between both
  - encoded as D, $D_{ij} = 1$ if $trans(i_{source}) = j_{target}$

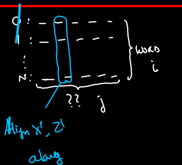
Embedding Normalization

- length normalise embeddings
- mean-center each dimension
- length normalise embeddings.

⇒ dot product ≃ cosine similarity ∝ euclidian distance

Fully unsupervised initialization

Construct X', Z' s.t ∃ alignment along X'_{*j}, Z'_{*j}



Idea:

$M_x = XX^T$
$M_z = ZZ^T$ ⇒

$$\begin{matrix} w_0^2 & w_0w_1 & w_0w_2 & w_0w_3 \\ w_1w_0 & w_1^2 & w_1w_2 & w_1w_3 \\ w_2w_0 & w_2w_1 & w_2^2 & w_2w_3 \\ w_3w_0 & w_3w_1 & w_3w_2 & w_3^2 \end{matrix}$$

$\boxed{\text{Assuming isometry}}$, M_x and M_z have a match in some combination of rows, columns.

→ Cannot try all permutations, computational limits.

So:
- Sort each row of $\sqrt{M_x}, \sqrt{M_z}$
- normalize $\underbrace{sorted(\sqrt{M_x})}_{X'}, \underbrace{sorted(\sqrt{M_z})}_{Z'}$

Robust self-learning

Algorithm:
- Computes optimal ortho mapping to maximize similarities for dictionary D.
- Computes optimal dictionary over sim. mat. of mapped embeddings. $(XW_xW_z^TZ^T)$

Init Dict
- Compute X', Z'

$$\text{argmax} \sum_i \sum_j D_{ij}\left((X_{i*}W_x)\cdot(Z_{j*}W_z)\right)$$

$$D_{ij} = \begin{cases} 1 & \text{if } j = \text{argmax}_k (X_{i*}W_x)\cdot(Z_{k*}W_z) \\ 0 & \text{otherwise.} \end{cases}$$

Dictionary was created in previous step only so it gets not stuck in poor local optima.

IMPROVEMENTS

- Stochastic dictionary induction
- Freq-vocab cutoff.
- CSLS retrieval
- Bidirectional dictionary induction.

Symmetric Re-weighting

Evaluating Embeddings

$$
\begin{array}{l}
\text{English words} \longrightarrow X \text{ (english embeddings)} \\
\qquad\qquad\qquad\quad \downarrow W_Y \\
\qquad\qquad\qquad\quad \text{embed\_X / embed\_Y} \\
\qquad\qquad\qquad\quad \uparrow W_X \\
\text{Hindi words} \longrightarrow Y \text{ (hindi embeddings)}
\end{array}
$$

1. ### Intrinsic evaluation

Word translation

- Take equivalent words in both languages and map into vec. space.
  - hi, en :

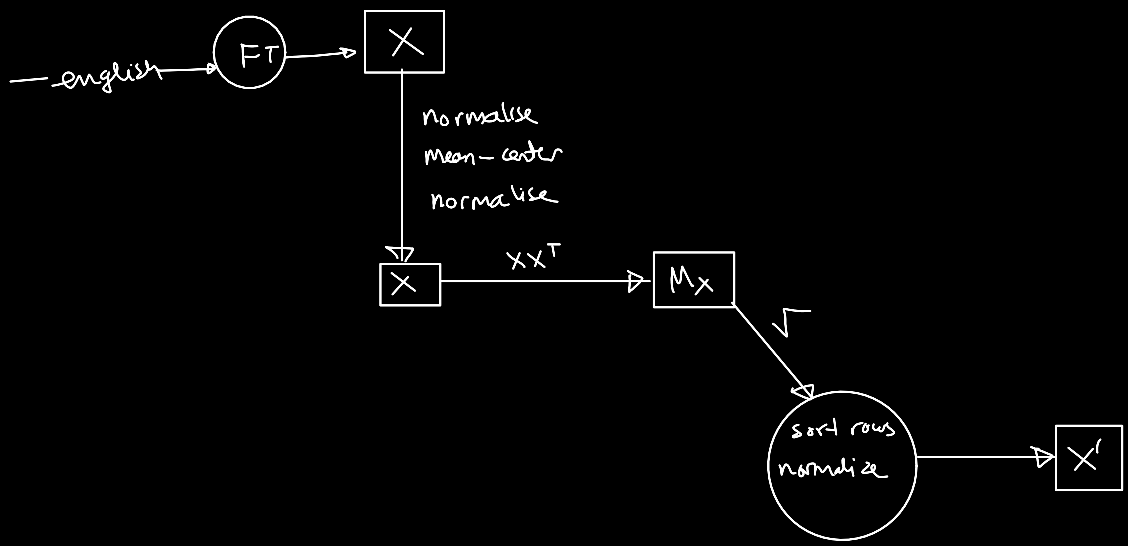
    _hi = (hi)

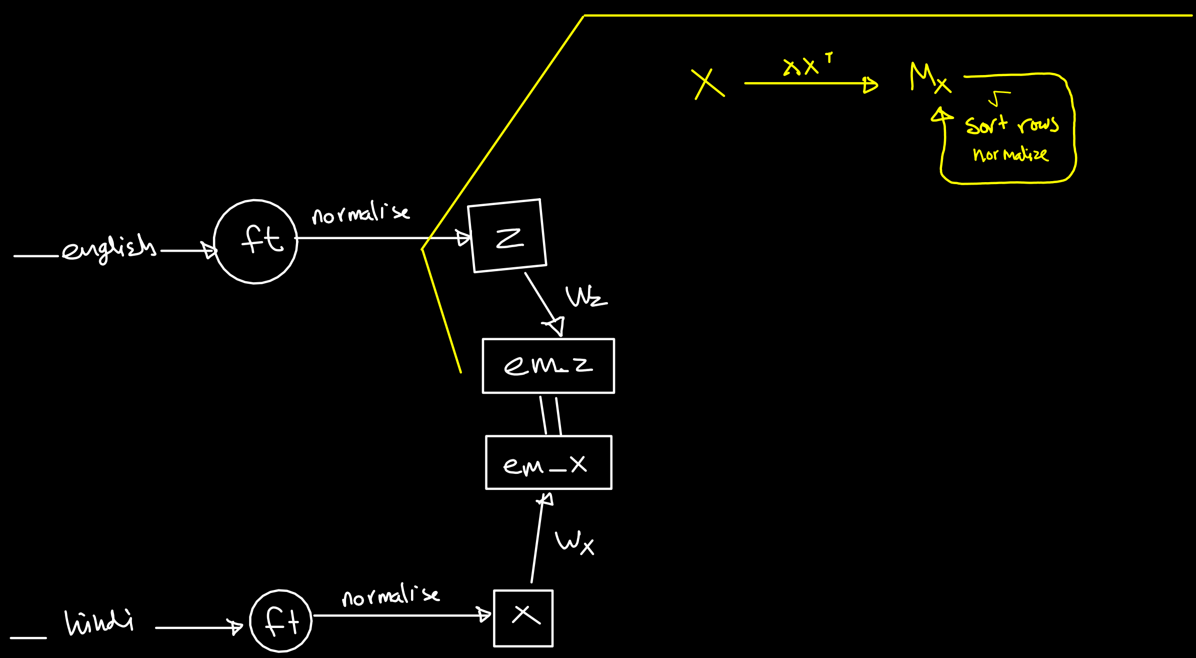
    _en = (en)

    cosine (_hi, _en).

$$
\text{score}((l_1, w_1), (l_2, w_2)) = \begin{cases} 1 & \text{if } \cos(E(l_1, w_1), E_2(l_2, w_2)) \geq \cos(E(l_1, w_1), E(l_2, w_2')) \quad \forall\, w_2' \in G^{l_2} \\ 0 & \text{otherwise.} \end{cases}
$$

2. ### Extrinsic Evaluation


english
FT
X
normalise
mean-center
normalise
X
XX^T
Mx
√
sort rows
normalize
X'


X
XX^T
Mx
√
sort rows
normalize
english
ft
normalise
Z
Wz
em_z
em_x
Wx
hindi
ft
normalise
X

norm(X)   sim_matrix   dist_m

X : hi_matrix

Y : en_matrix

MX, MY : similarity $(\sqrt{?\,?^T})$

X1, Y1 : sorted, norm MX, MY

D : dictionary

W_X, W_Y : transform vector.

Possible Improvements

1. Most frequent Content Words instead of most freq. words. Will not get embeddings Accurately enough maybe?
2. Solutions of Wahba's problem.

I will get some water from the fridge.

मै Fridge से पानि ले कर आऊंगा

Imul → Iml

~~frequent~~

~~1 0~~ → 1 0

0 1 → 0 1

$X^T X$ → all pas of dot prod.

$U^T U$ → all pas

yy → 0 1

yz → 0.7

y.x → 0.3

$\begin{bmatrix} xx & xy & xz \\ yx & yy & yz \\ zx & zy & zz \end{bmatrix}$

$\begin{bmatrix} \alpha\alpha & \alpha\beta & \alpha\gamma \\ \beta\alpha & \beta\beta & \beta\gamma \\ \gamma\alpha & \gamma\beta & \gamma\gamma \end{bmatrix}$

x.y = 0.3

y.y = 0 1

0.1 = yy

0 3 = xy

0.7 = z.y

y → (y.xi)

y xi

"function words" → the, a, an, …

nouns, words + the p.b.

$B \approx A$