Page No.
Date

Week 2: NLP & Word embeddings

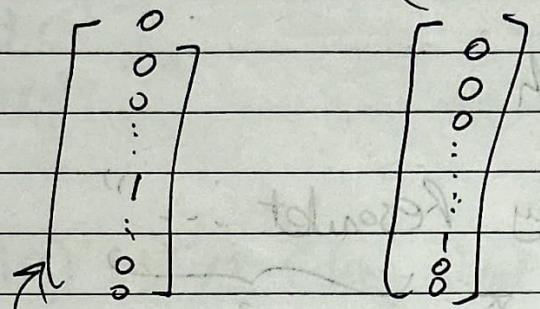
Intro to word embeddings:

1) Word representation:

$V = [a, aaron, \ldots, zulu, <UNK>]$

$|V| = 10{,}000$

Man (5391) — Woman (9853)



O_{5391} (one-hot vector) O_{9853}

I want a glass of orange juice

I want a glass of apple ?

Featurized representation: word embedding

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.7	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97
⋮	⋮	⋮				
Size						
Cost						
Noun						
Verb...						

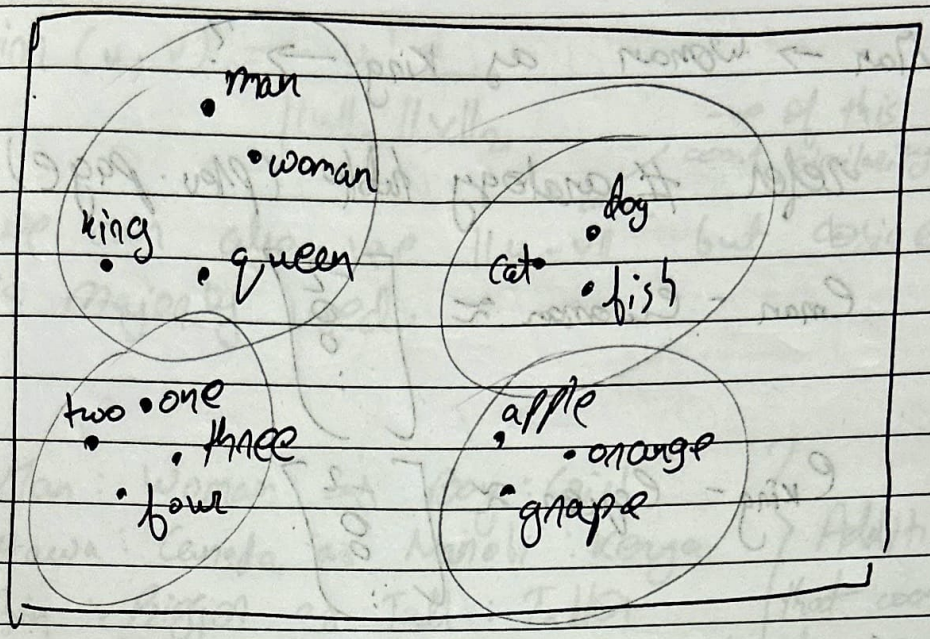
300

Analogy table

$e_{5391} = e_{man}$ $e_{9853} = e_{woman}$

Page No.
Date

Visualizing word embeddings:



t-SNE

2) Using word embeddings

Name entity recognition example

Sally Johnson is an orange farmer

Using a model that uses word embeddings, we will be able to predict that Robert Lin is also a person.

Robert Lin is an apple farmer

The algorithm will identify orange and apple belonging to the same class hence output Robert Lin as a person.

Instead of apple farmer, if we use durian (another fruit) cultivator, our algo will still be able to give us correct outputs

3) Properties of ~~most~~ word embeddings:

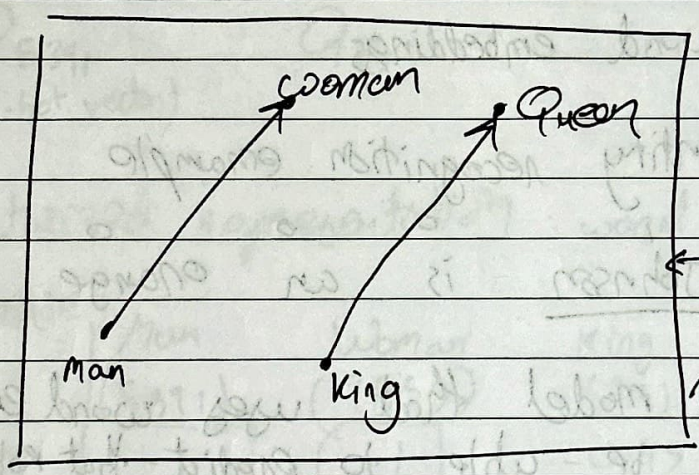
a) Man → Woman as King → ?

refer # analogy table (prev. page)

$$e_{man} - e_{woman} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

$$e_{king} - e_{queen} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

Compute $e_{man} - e_{woman} \approx e_{king} - e_?$ ← queen



Find word w : $\arg\max_w \text{sim}(e_w, e_{king} - e_{man} + e_{woman})$

maximizes similarity

Returns the word w which has maximum similarity to the equation $e_{king} - e_{man} + e_{woman}$ and it should be the word queen with accuracy of 30 - 75%.

Page No.
Date

Cosine similarity :

$$\text{sim}(u, v) = \frac{u^T v}{\|u\|_2 \|v\|_2}$$

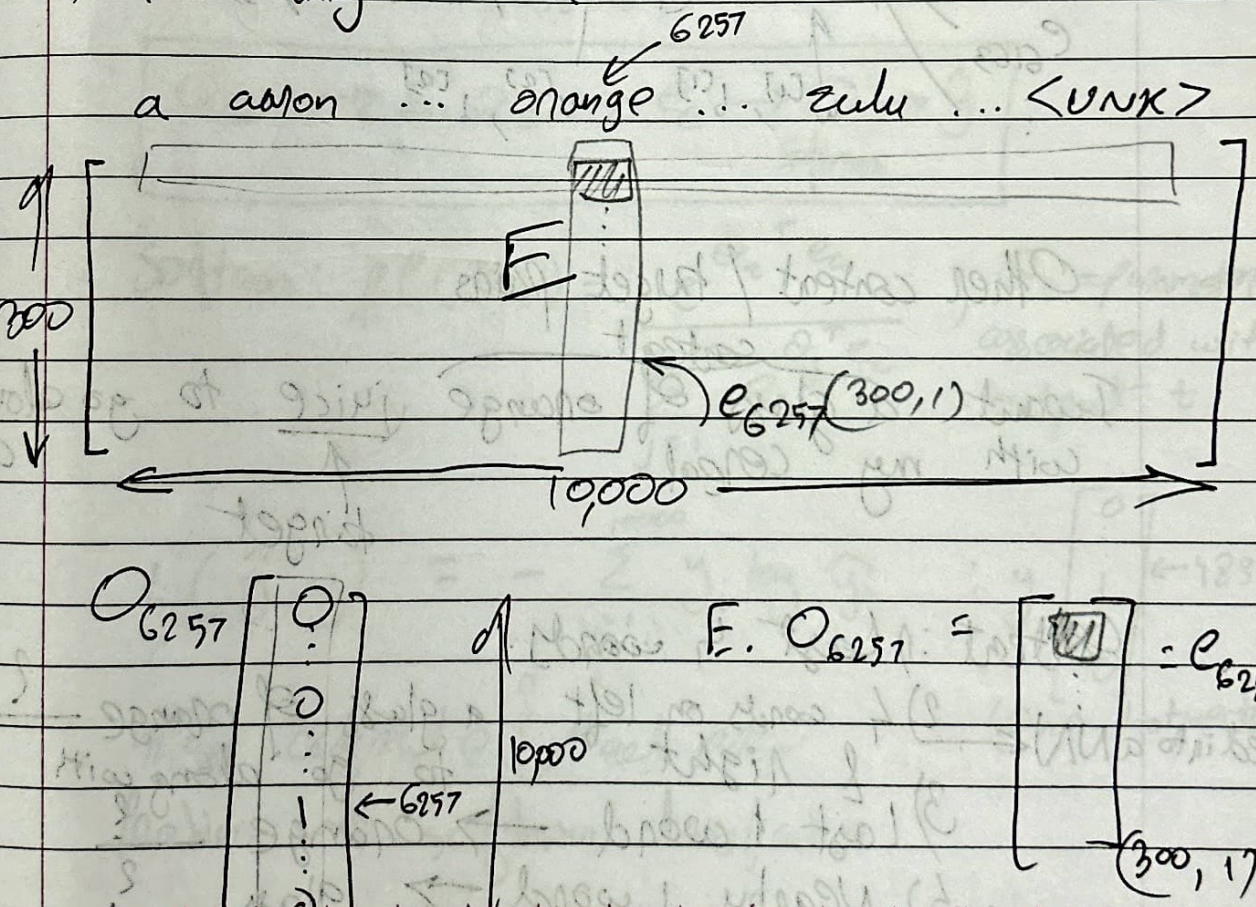
-ve of this as we count similarity not dissimilarity

We can also use $\|u - v\|_2$ but cosine similarity is majorly used.

Man : Woman as Boy : Girl
Ottawa : Canada as Nairobi : Kenya
Big : Bigger as Tall : Taller
Yen : Japan as Ruble : Russia

} Addition examples that word emb. algo can do.

4) Embedding matrix



1) Learning word embeddings:

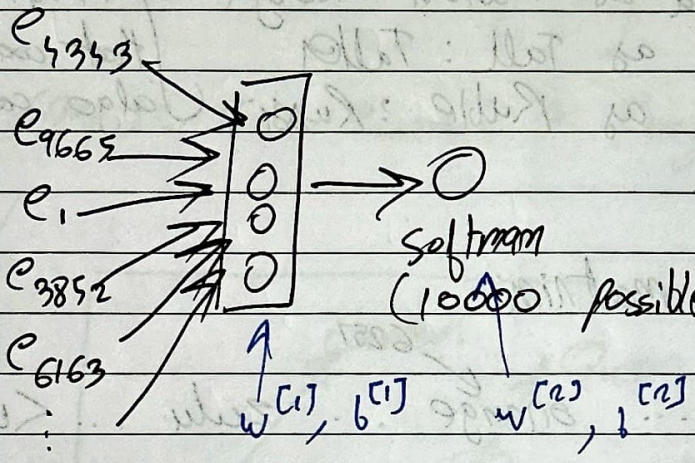
I want a glass of orange ____ ?

I $O_{4343} \rightarrow E \cdot O_{4343} \rightarrow e_{4343}$

want $O_{9665} \rightarrow E \cdot O_{9665} \rightarrow e_{9665}$

a $O_1 \rightarrow E \cdot O_1 \rightarrow e_1$

embedding vectors (300, 1)



Other context / target pairs

context

I want a glass of orange juice to go along with my cereal.

target

Context: 1) Last 4 words.

Feed into a NN ← 2) 4 words on left & right } a glass of orange ____ ?

3) Last 1 word → orange ____ ?

4) Nearby 1 word → glass ____ ?

to go along with

skip-gram

Page No.
Date

2) Word 2 Vec Skip-grams Model:

~~Skip-grams~~

I want a glass of orange juice to go along with my cereal.

Content	Target
orange	juice
orange	glass
orange	my

Model

Vocab size = 10,000

$x \longrightarrow \hat{y}$

Content c ("orange") $\longrightarrow$ Target t ("juice")

6257 4834

$O_c \rightarrow E \cdot O_c \rightarrow e_c \rightarrow O \rightarrow \hat{y}$ (softmax)

Softmax: $p(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10000} e^{\theta_j^T e_c}}$, θ_t = parameter associated with output t.

$L(\hat{y}, y) = -\sum_{i=1}^{10000} y_i \log \hat{y}_i$; y = [0 ... 1 ... 0] ← 4834 (one-hot vector)

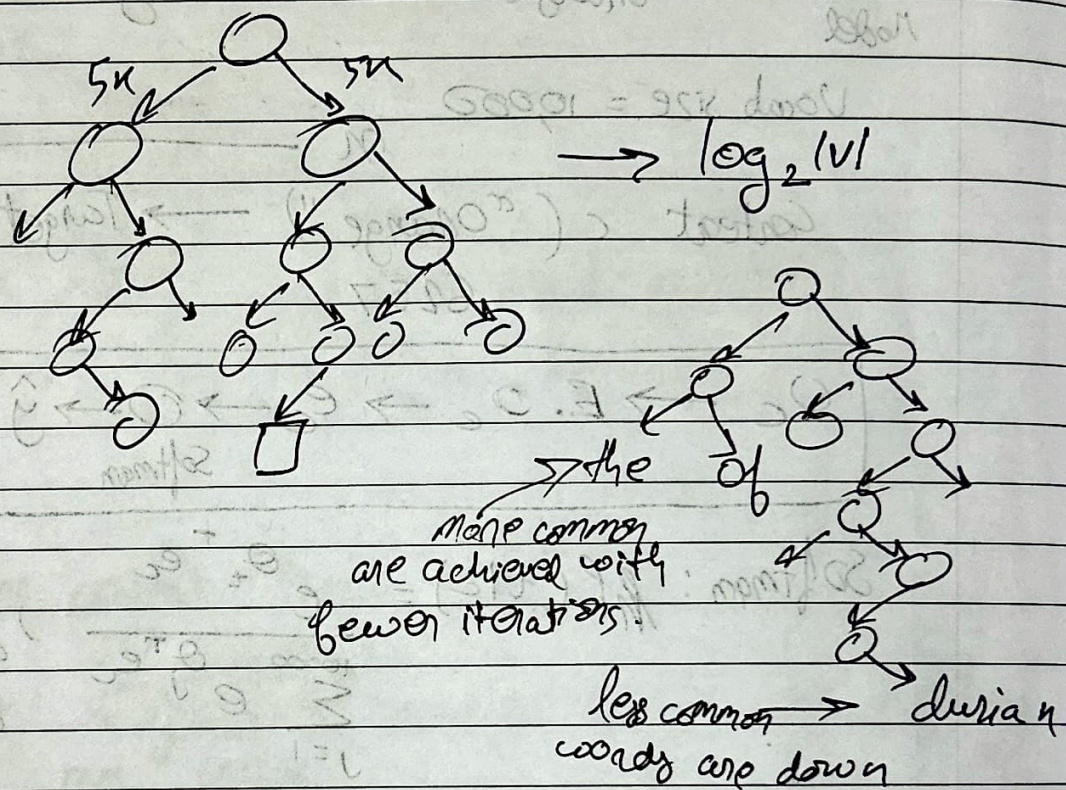
Then compute Back prop

Problems with softmax classification:

$$P(+|c) = \frac{e^{\theta_i^T e_c}}{\sum_{j=1}^{10000} e^{\theta_j^T e_c}}$$

because of the denominator, this would become computationally time consuming as 10000 iterations make it slow. 1M or 1B words would make it even slower.

Heirarchical softmax classifier:



3) Negative Sampling

Defining a new learning problem

I want a glass of orange juice to go along with my cereal.

Pick a content-target pair from ~~sentence~~ and set the row, target to 1

content	word	target?
orange	juice	1
orange	of	0
orange	king	0
orange	the	0
orange	book	0

← positive example

K

random words from the dictionary

Set these to 0 even if any word appears in the sentence (negative examples)

Create a supervised learning problem, where the algorithm inputs x (context and word) and has to predict the target y(0) i.e. ~~make~~ to distinguish between positive & negative examples

Model

~~Softmax~~ Binary classification

Using logistic regression,

$$P(y=1 \mid (c,t)) = \sigma(\theta_t^T e_c)$$

θ_t: parameter vector θ; e_c: embedding vector

orange → 6257

$O_{6257} \rightarrow E \cdot O_{6257} \rightarrow e_{6257}$

e_{6257} → 0 · juice? 0 · 0 · king 0 · (100 ∞)

* instead of 10,000 we can update k+1 units.

Page No.
Date

Selecting negative examples:

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=1}^{} f(w_j)^{3/4}}$$

$f(w_i) \rightarrow$ observed frequency

4) GloVe word vectors

→ global vectors for word representations

I want a glass of orange juice to go along with my cereal.

c, t

X_{ij} = # times j appears in context of i

(i: c, j: t; j: t; i: c)

$X_{ij} = X_{ji}$ ← may / may not be true, depends on the no. of words you're considering i.e. either in the range of ± 10 words or the word immediately next

Model:

$$\text{Minimize} \sum_{i=1}^{10000} \sum_{j=1}^{10000} f(X_{ij}) \left(\theta_i^T e_j + b_i + b_j' - \log X_{ij} \right)^2$$

(i: c, j: t; θ_i: c, e_j: t; b_i: c; b_j': t; X_{ij}: c, t)

weighting term

$f(X_{ij}) = 0$ if $X_{ij} = 0$ "$0 \log 0$" $= 0$

$f(x_{ij})$ → this, is, of, a, ... frequent words
↳ weight not too big
durian, ... infrequent words
↳ weight is not too small.
given by $f(x_{ij})$

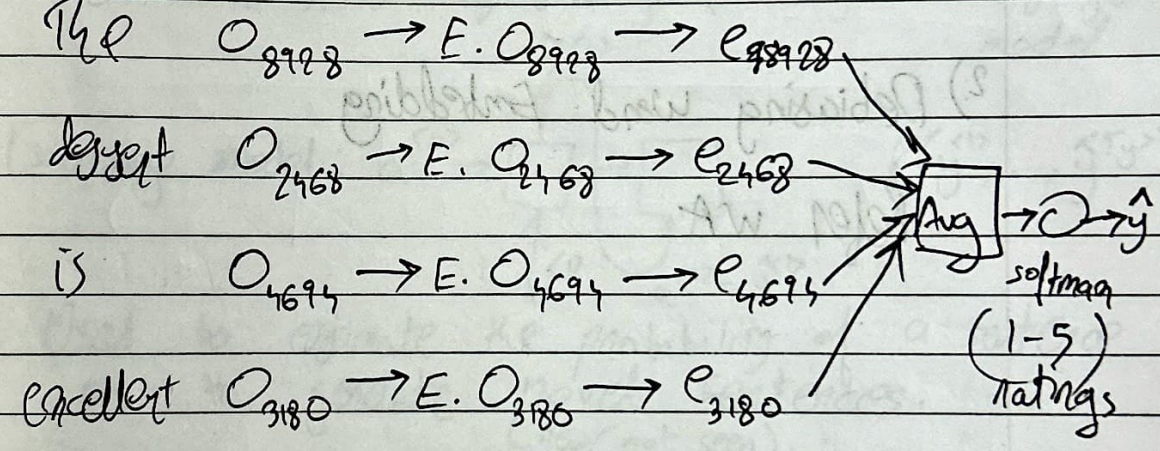
O_i & e_j are symmetric

$$e_w^{(final)} = \frac{e_w + O_w}{2}$$

Applications of Using Word Embeddings

1) sentiment classification (like or dislike)

eg: The dessert is excellent 4-★ Review
8928 2468 4694 3180

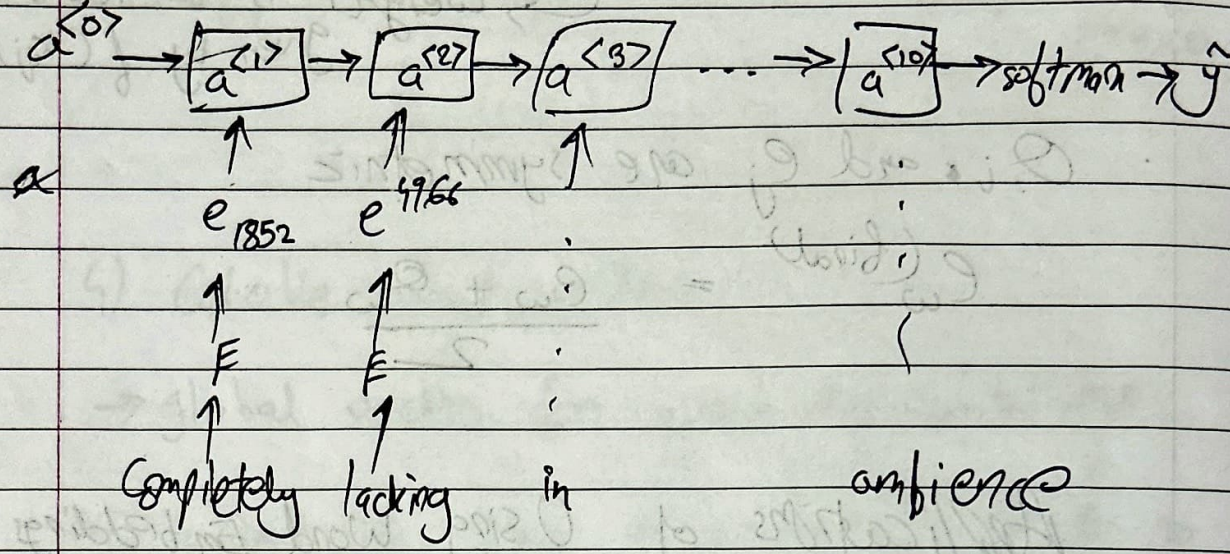


Drawback:
completely lacking in good taste, good service, and ~~got~~ good ambience

Since, there are so many goods, the average will orient itself to good but in reality its a 1-★ review

Page No.
Date

RNN for sentiment classification



This algorithm will do a better job at classification as it can work on words that are not in the training set too. This is a type of many-to-one architecture

2) Debiasing word Embedding

Refer WA