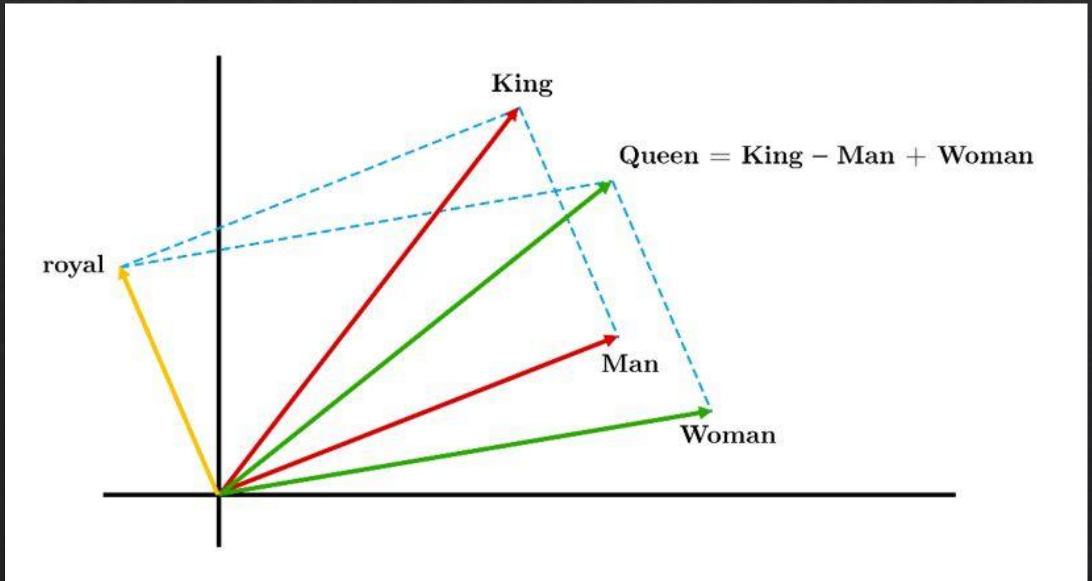
A peek into word embeddings using word2vec

Yue Ka Leung (1155214424 - CSCI)

Tang Yu Hin (1155211754 - AIST)

Word embeddings & Word2Vec

- Represent words as vectors. (String → double[])
  - Here we have an example.
  - Basis: <👑, ⚥> (instead of the standard <1, 1>)
  - *Man* = <1, 1>
  - *Woman* = <1, 2>
  - *King* = <2, 1>
  - *Queen* = *King* – *Man* + *Woman*
    = <2, 1> - <1, 1> + <1, 2>
    = <2, 2>

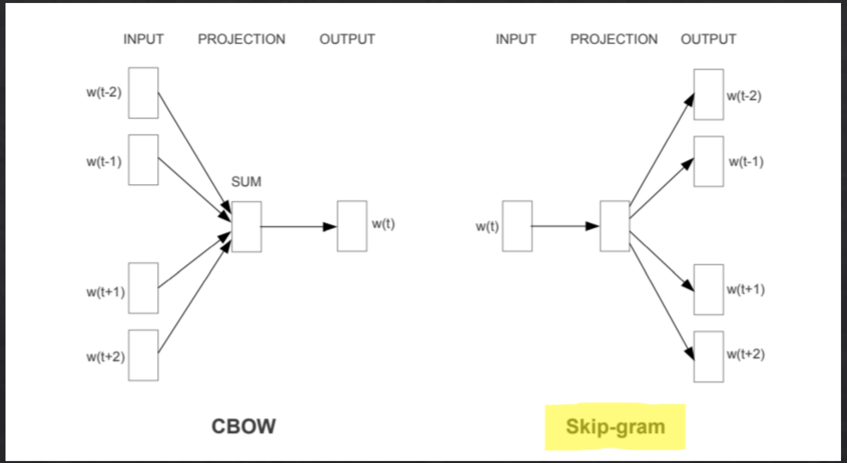


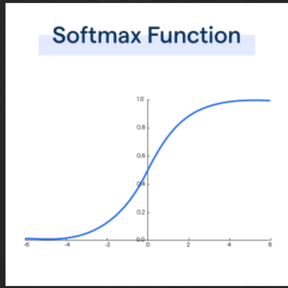
- The word vectors depend on the words frequently appearing near the word in the data.
- In real life applications, the vector spaces usually have much higher dimensions.

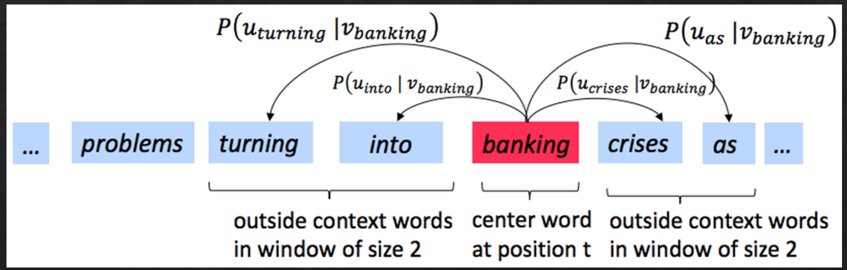
Probability in Word2Vec models

- CBOW: input Outside context words → output Center word
- Skip-gram: input Center word → output Outside context words

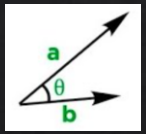
- Let's focus on Skip-gram.
- We calculate the probability in the training of models:
  - $P(O=o|C=c) = \frac{e^{\vec{u_o}^T \cdot \vec{v_c}}}{\sum_{w \in Vocab} e^{\vec{u_w}^T \cdot \vec{v_c}}} \in [0,1]$
  - The softmax function gives a probability distribution.
  - Probability of words appearing together in the data → semantic meanings
  - The word vectors are continuously adjusted to maximize the probability.

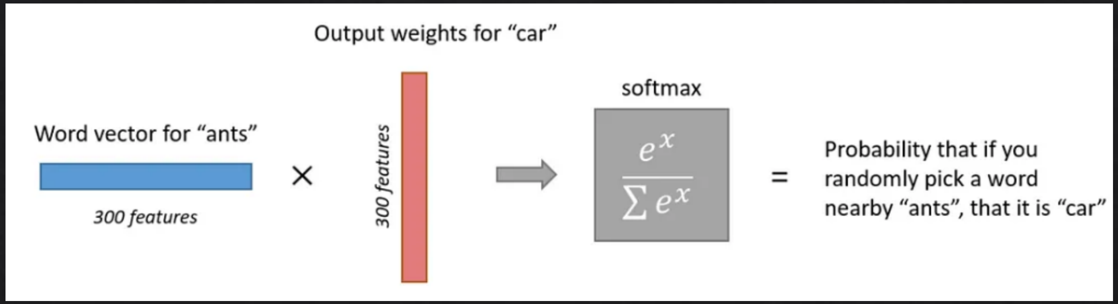






- The trained models compute the cosine similarity $\cos\theta$
  - Recall that $\cos\theta = \frac{\vec{a} \cdot \vec{b}}{|\vec{a} \cdot \vec{b}|} \in [-1,1]$
  - Similarity of the semantic meanings of the words
  - Note that dot products are involved in both the calculations of $\cos\theta$ and P…





Code implementation

- Implementing a basic Skip-Gram model:

```
epoch  999  loss =  15919.783211167764
Top 3 words most similar to 'food' in the essay: ['i', 'nourish', 'plant-based']
```

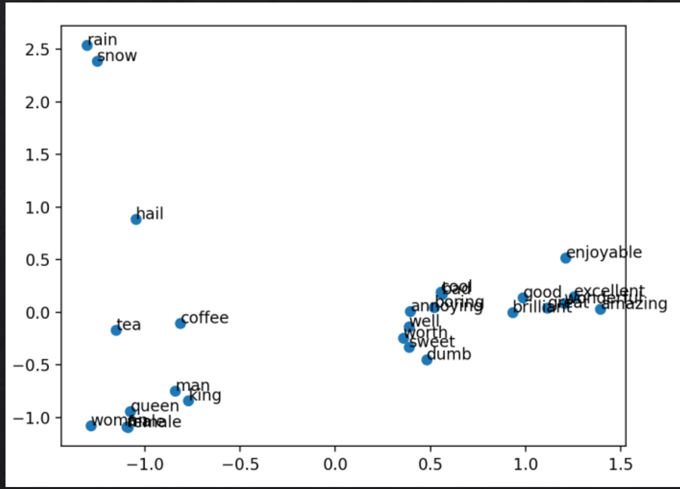
- Using a pre-trained model (word2vec-google-news-300):

```
Most similar words to the word "food":
foods 0.6804922819137573
Food 0.6538903713226318
foodstuffs 0.642582893371582
meals 0.616668701171875
food_stuffs 0.5928642153739929
nourishing_meals 0.5847609043121338
foodstuff 0.5835224986076355
staple_foods 0.5535788536071777
nutritious 0.5466451644897461
meal 0.5433712601661682
```

```
wv.most_similar("good")

[('great', 0.7291510105133057),
 ('bad', 0.7190051078796387),
 ('terrific', 0.6889115571975708),
 ('decent', 0.6837348341941833),
 ('nice', 0.6836092472076416),
 ('excellent', 0.644292950630188),
 ('fantastic', 0.6407778263092041),
 ('better', 0.6120728850364685),
 ('solid', 0.5806034803390503),
 ('lousy', 0.576420247554779)]
```

- Factors affecting prediction: size of corpus, preprocessing techniques (stop word removal, tokenization), epochs, sampling etc.



- Please visit the GitHub Repository* to read the more detailed explanation and download the full source code. Thank you.

* Link: https://github.com/yueagar/ESTR2018-project