Deep Learning

Big Data & Machine Learning Bootcamp - Keep Coding

Outline

1. Word representation
2. Using word embeddings
3. Properties of word embeddings
4. Learning word embeddings
5. Word2Vec & Skip-gram
6. Word2Vec & Skip-gram (Negative sampling)
7. GloVe word vectors
8. Sentiment classification
9. Debiasing word embeddings

1. Word representation

How we can represent words?

So far we've shown how we can represent words in **one hot representation.**

There is also another more powerful word representation (word embedding) that allows us to find similarity between words.

Using one hot representation we can't find similarities between the word orange and apple. This means, the word apple is as probable as queen or castle.

Word embedding or FEATURIZED REPRESENTATION allows us to cluster words by their similarity. It is also called word encoding!

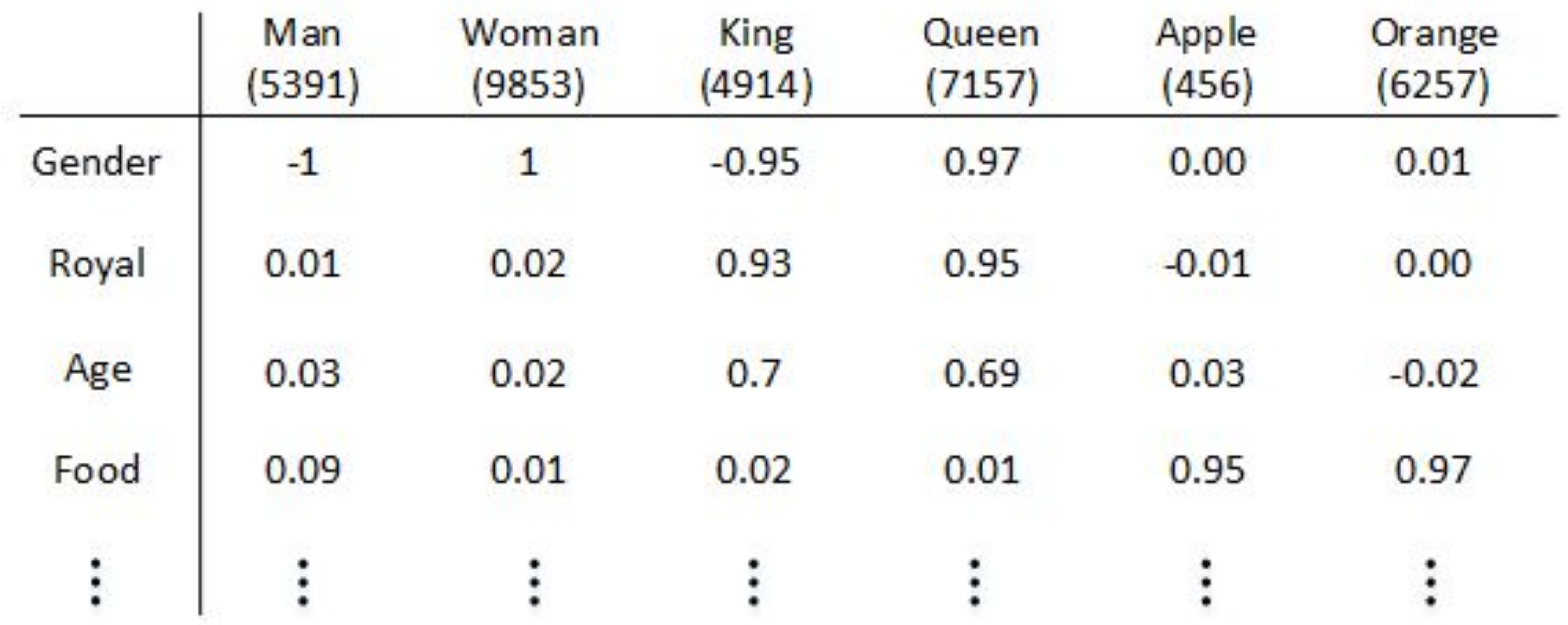
Sources:
- Coursera

1. Word representation

Word embedding: Featurized representation

Feature

Word and the number in the dictionary



	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
⋮	⋮	⋮	⋮	⋮	⋮	⋮

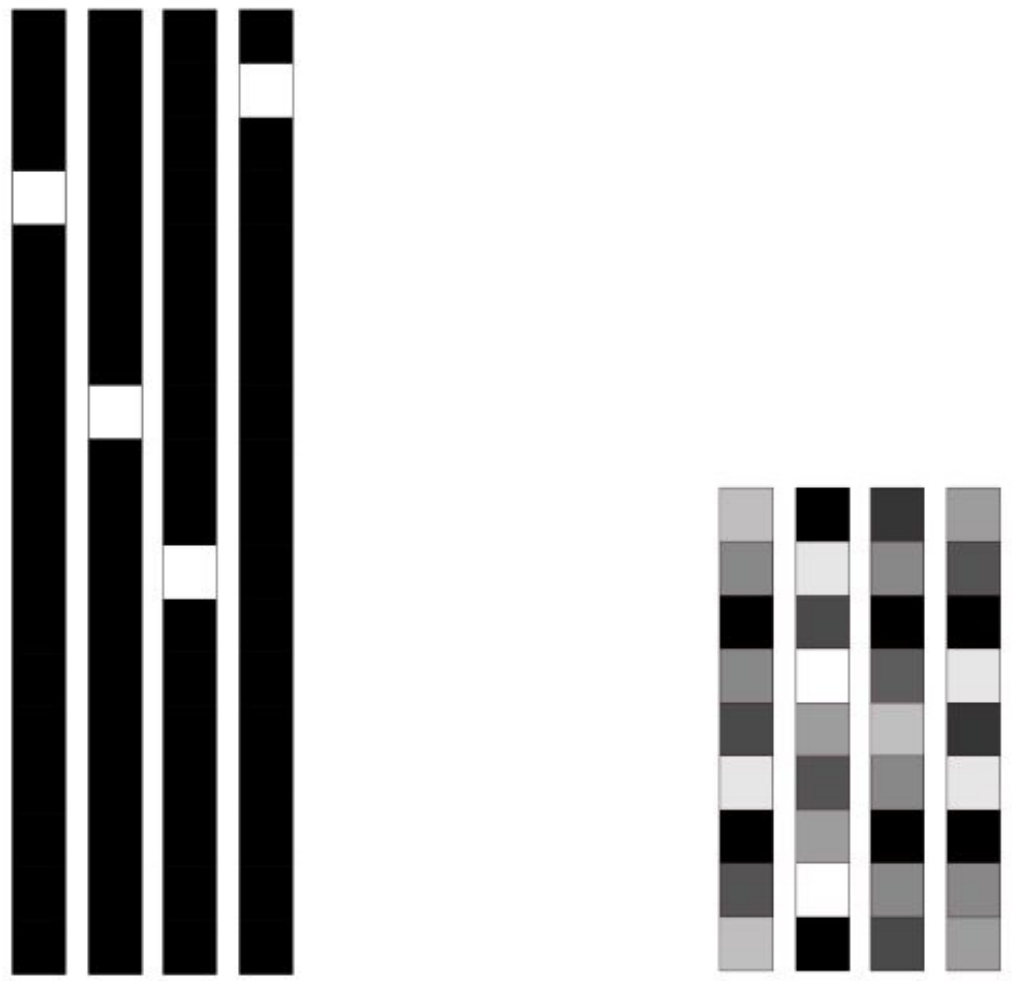
We represent the words by using vectors that are close when the words are similar or they have some semantic similarity.

This table is just to show an example. Usually the vectors are automatically learned and they don’t have obvious distinctions as gender, Royal, Age, etc

Sources:
- Coursera

1. Word representation

Word embedding: Featurized representation



```
One-hot word vectors:
- Sparse
- High-dimensional
- Hard-coded
```

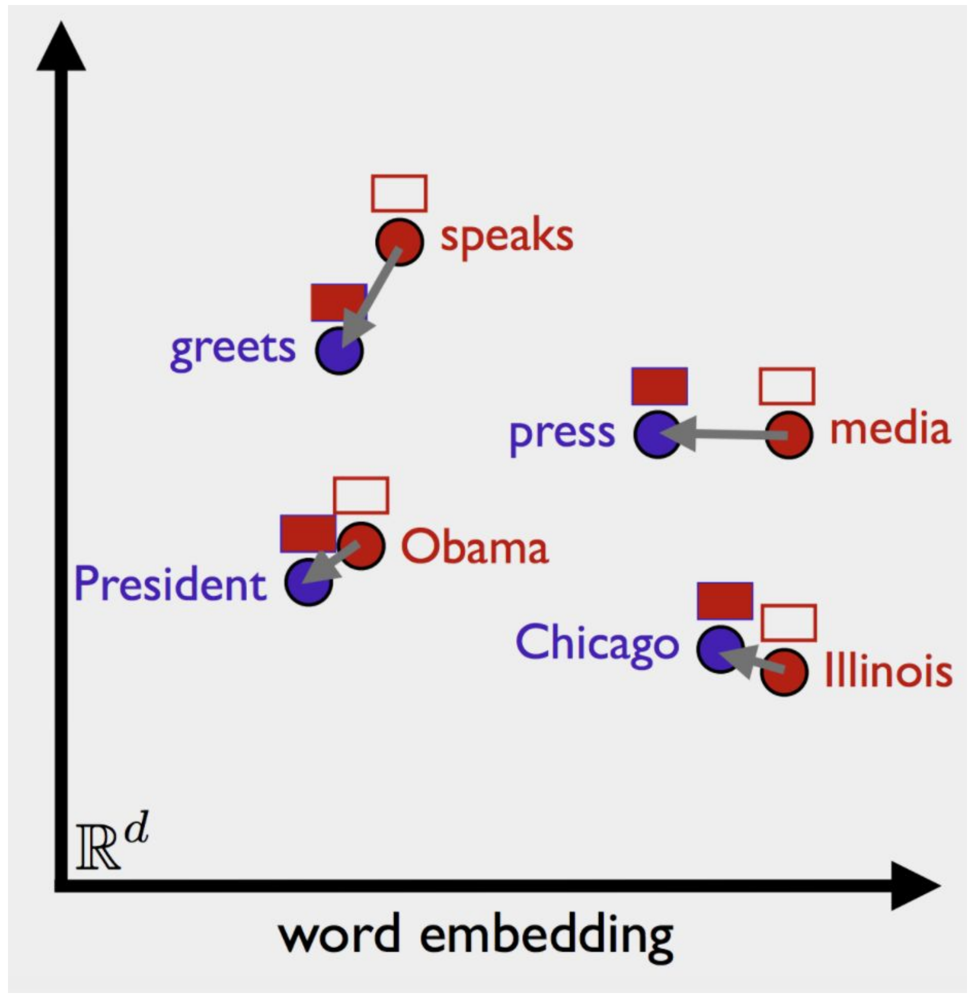
```
Word embeddings:
- Dense
- Lower-dimensional
- Learned from data
```

Sources:
- Coursera
- https://jjallaire.github.io/deep-learning-with-r-notebooks/notebooks/6.1-using-word-embeddings.nb.html

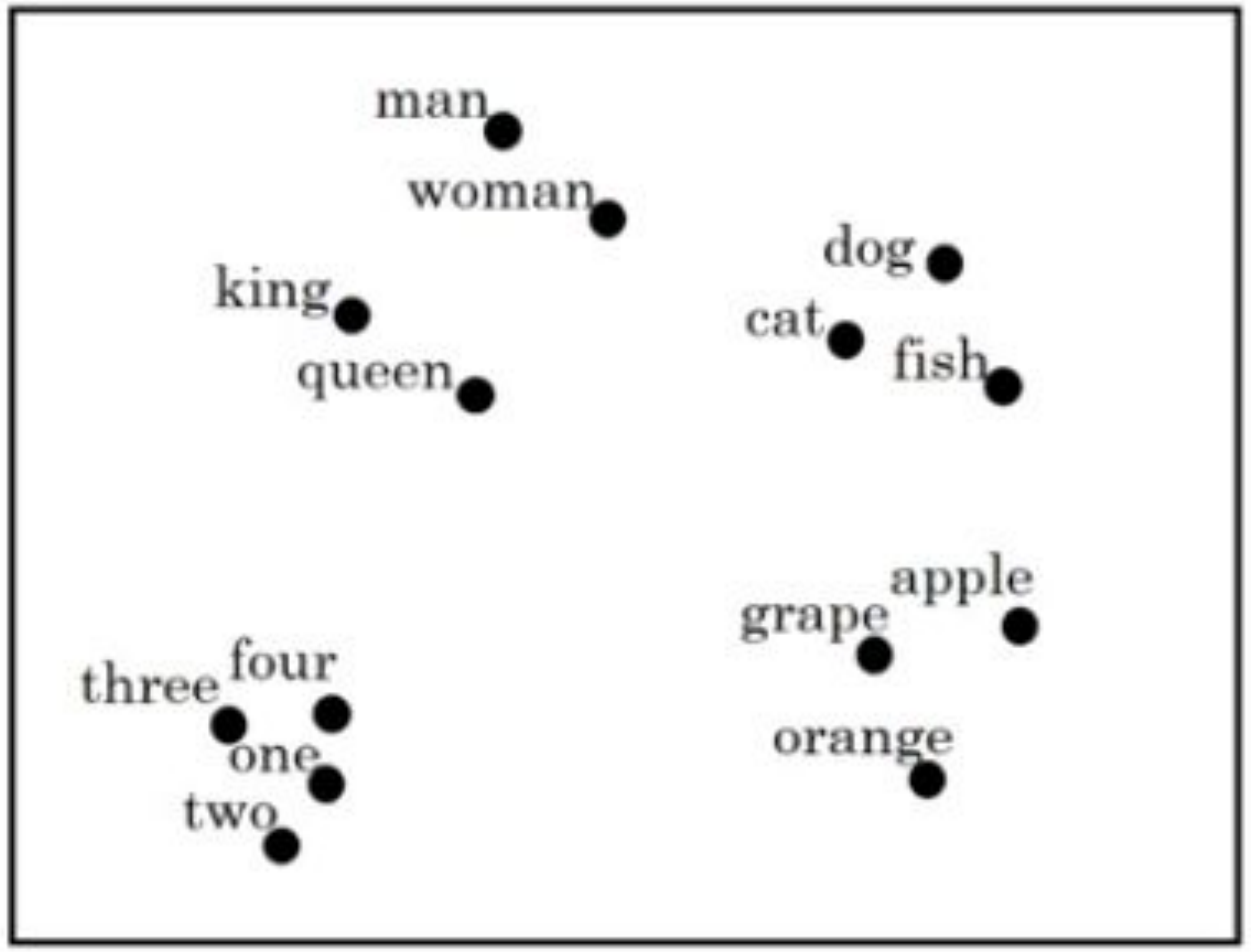
1. Word representation

Word embedding: Featurized representation

Example 1



Example 2



They are called embeddings because these vectors are represented by a number of variables that can be embedded in a space with the same number of dimensions.

These vectors are embedded in a space. This space can be a 300 dimensional space

Maaten, Laurens van der, and Geoffrey Hinton. "Visualizing data using t-SNE." *Journal of machine learning research* 9, no. Nov (2008): 2579-2605.

Sources:
- Coursera
- https://mlwhiz.com/blog/2017/04/09/word_vec_embeddings_examples_understanding/

2. Using word embeddings

Word embeddings has many advantages when compared to one hot representation of words. Probably the most important one is transfer learning.

1. You can learn embeddings from a large text corpus (1 to 100 billion words) or download pre-trained embedding online

2. Transfer embedding to new task with smaller training set. (Say 100.000 words)
   Really useful for most Natural Language Processing (NLP) tasks but less useful for language modelling and machine translation settings

3. If you want, fine tune the word embedding with new data.

Sources:
- Coursera

3. Properties of word embeddings

The most fascinating property of word embedding is that it helps with analogy reasoning.

This may not be the most relevant Natural Language Processing (NLP) application, but word embeddings help to convey a sense of what word embeddings are doing or what they can do.

BUT what is analogy reasoning?

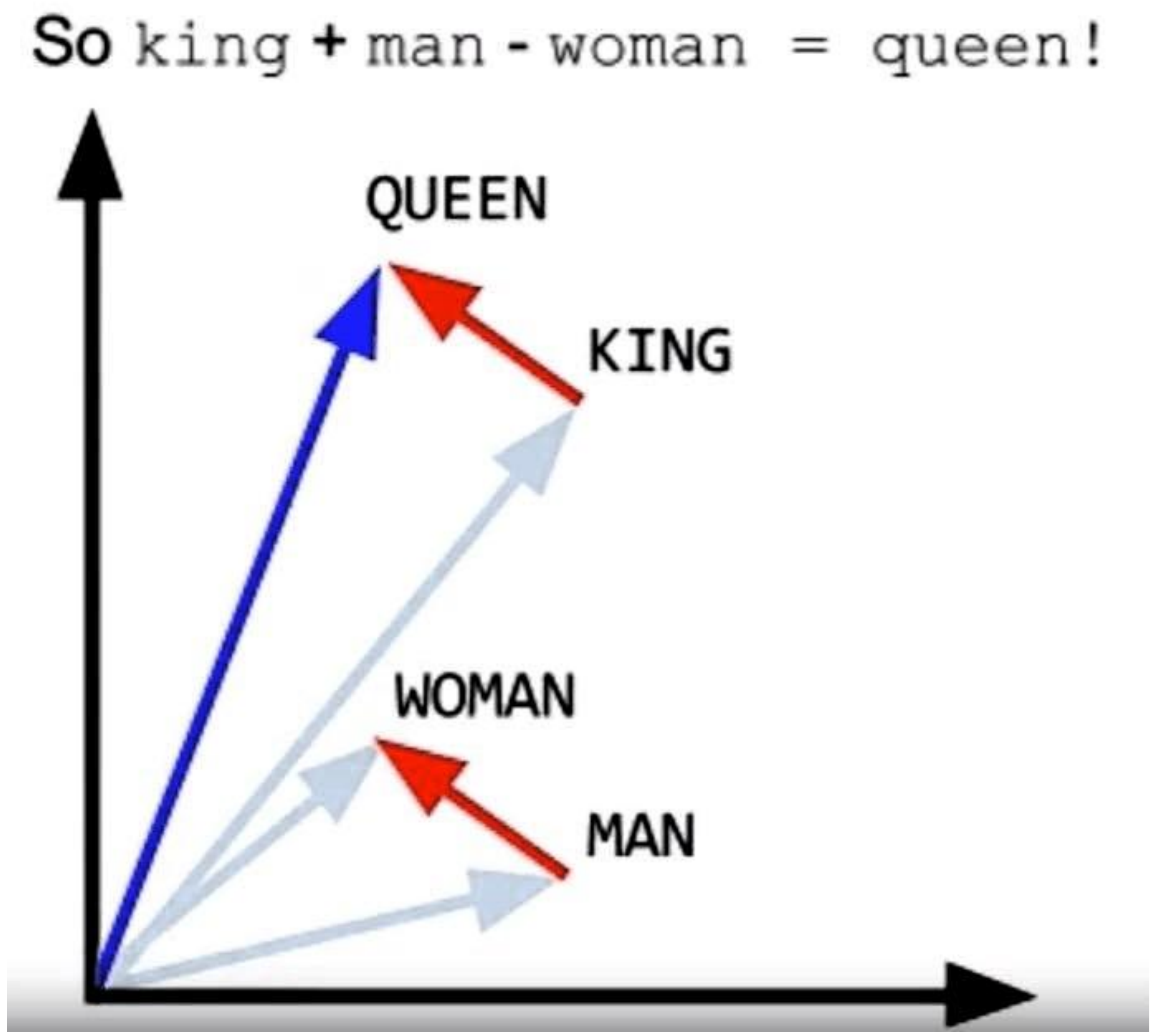
Basically it helps us to find relationships: **Man** is to a **Woman** as **King** is to **?**

This means, we subtract the vector representing the word **Man** to the vector representing the word **Woman** and then, find a word that subtracting the the word **King** we obtain a similar output.

Sources:
- Coursera

3. Properties of word embeddings

Analogy reasoning.



Red vectors are the vector differences between word pairs ***Man-Woman*** *and* ***King-Queen****.*

Mikolov, Tomáš, Wen-tau Yih, and Geoffrey Zweig. "Linguistic regularities in continuous space word representations." In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746-751. 2013.

Sources:
- Coursera
- https://twitter.com/kirkdborne/status/1080500437520924672

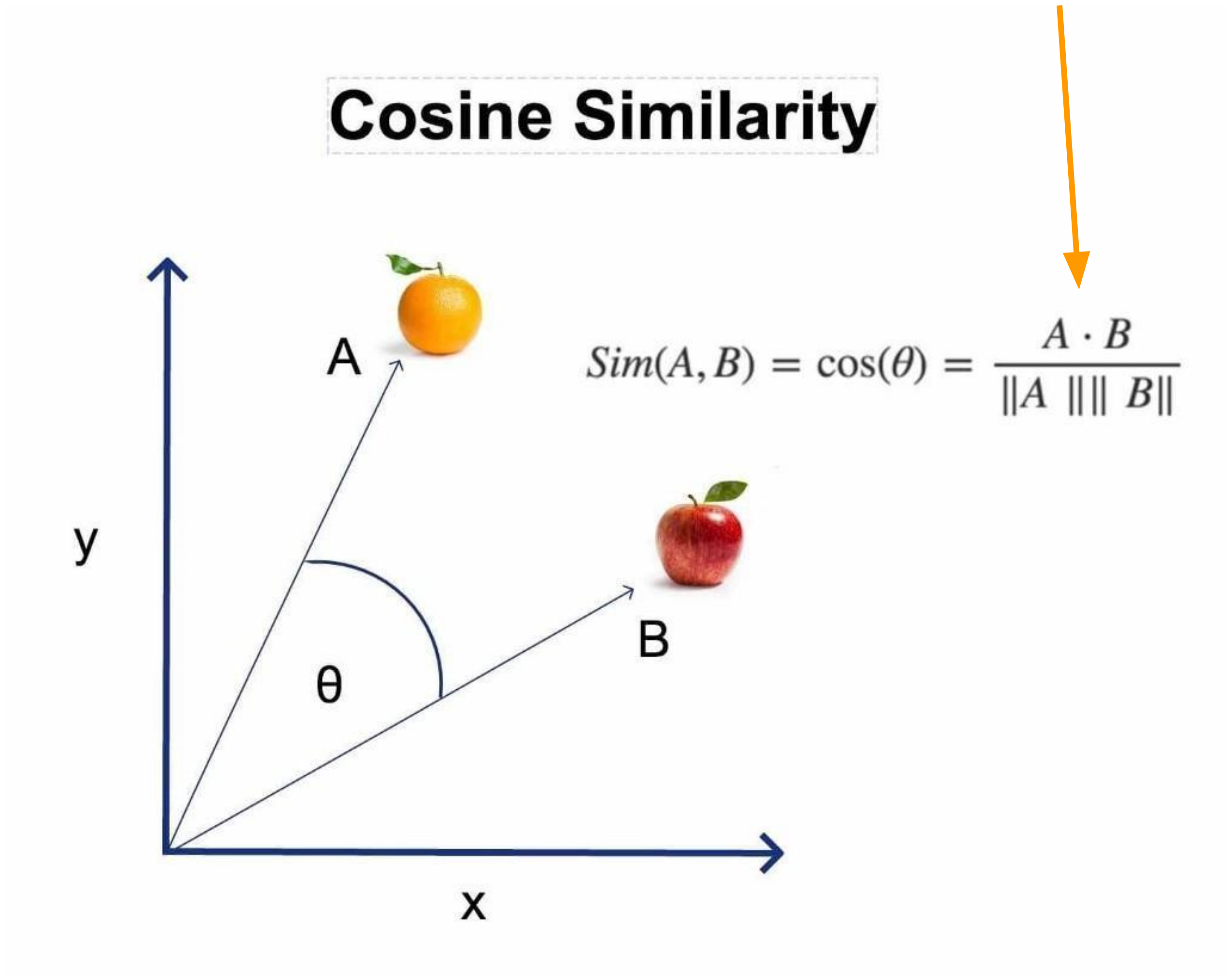
3. Properties of word embeddings

Analogy reasoning - Cosine similarity

The most commonly used formula to compute similarity between words is the **Cosine similarity.**

We can also used Euclidean distance, but that will be a measure of dissimilarity instead of similarity, as Euclidean distance is how separated are two vectors.

Inner product



$$Sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Mikolov, Tomáš, Wen-tau Yih, and Geoffrey Zweig. "Linguistic regularities in continuous space word representations." In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746-751. 2013.

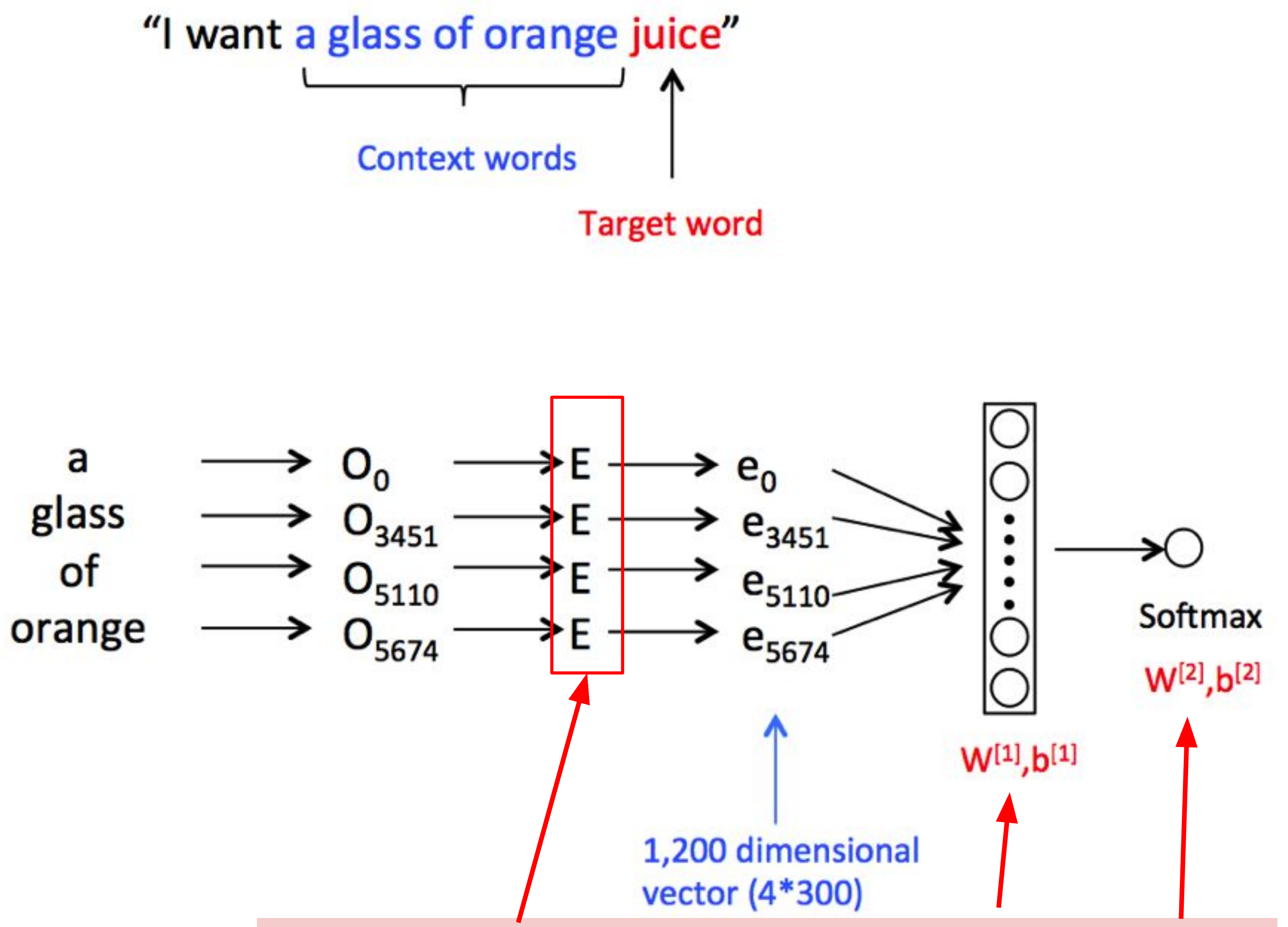
Sources:
- Coursera
- https://laptrinhx.com/how-recommendation-systems-work-1825654218/

4. Learning word embeddings

Now, let's talk about how we can build word embeddings.

Building a neural language model is a reasonable way to learn a set of word embeddings

*We used 4 words history (**context words**) to predict the **target word**.*
By learning those network parameters, this algorithm learns pretty decent word embeddings (Matrix E).



These are the parameters of the network

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. "A neural probabilistic language model." *Journal of machine learning research* 3, no. Feb (2003): 1137-1155.

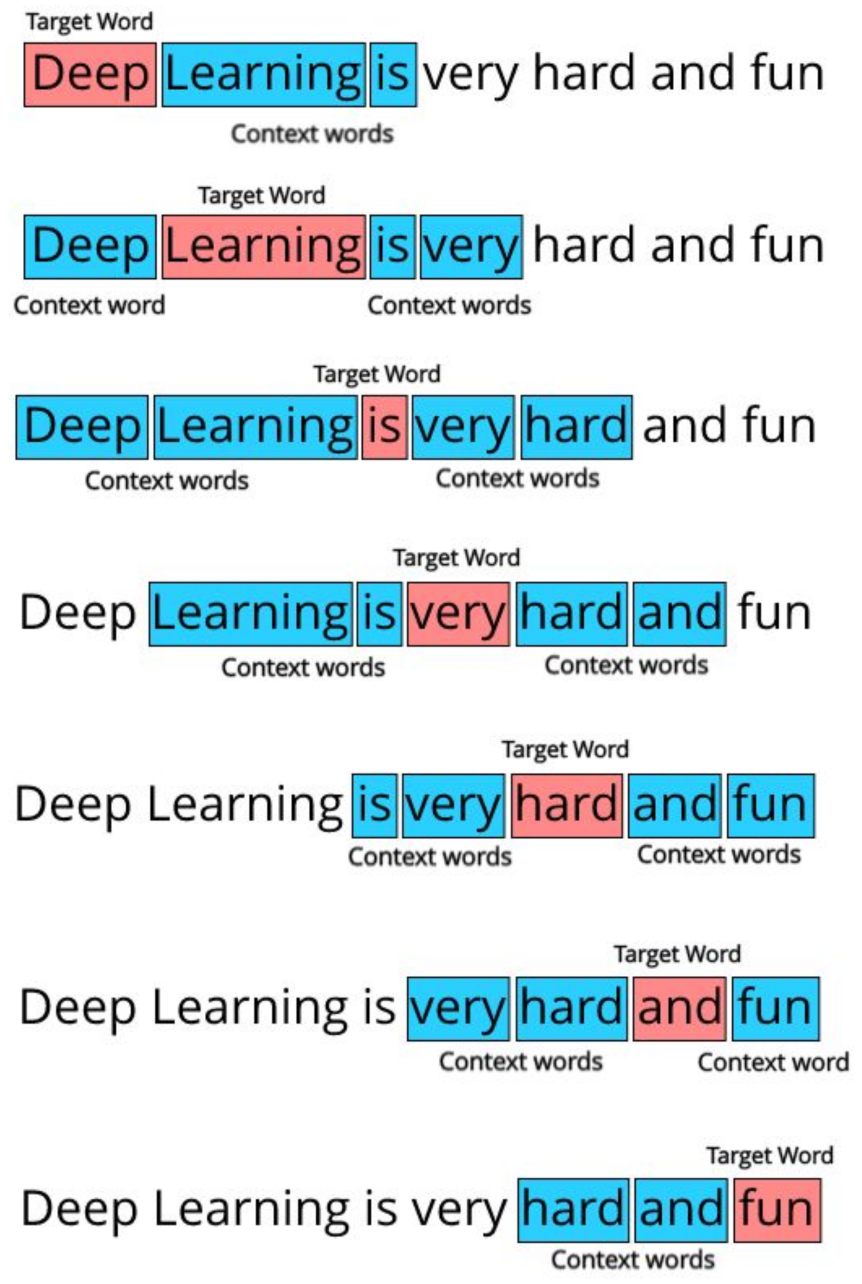
Sources:
- Coursera

4. Learning word embeddings

Using other context/target pairs.

We can also use other context words instead of using only the previous 4 words. These are the options:

- **Last 4 words**
- **4 words on left and right**
- **Last 1 word**
- **Nearby 1 word (or Skip-gram). More simple but it works really well.**


Target Word
Deep Learning is very hard and fun
Context words
Target Word
Deep Learning is very hard and fun
Context word
Context words
Target Word
Deep Learning is very hard and fun
Context words
Context words
Target Word
Deep Learning is very hard and fun
Context words
Context words
Target Word
Deep Learning is very hard and fun
Context words
Context words
Target Word
Deep Learning is very hard and fun
Context words
Context word
Target Word
Deep Learning is very hard and fun
Context words

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. "A neural probabilistic language model." *Journal of machine learning research* 3, no. Feb (2003): 1137-1155.

Sources:
- Coursera
- https://deeplearningdemystified.com/article/nlp-1

5. Word2Vec & Skip-gram

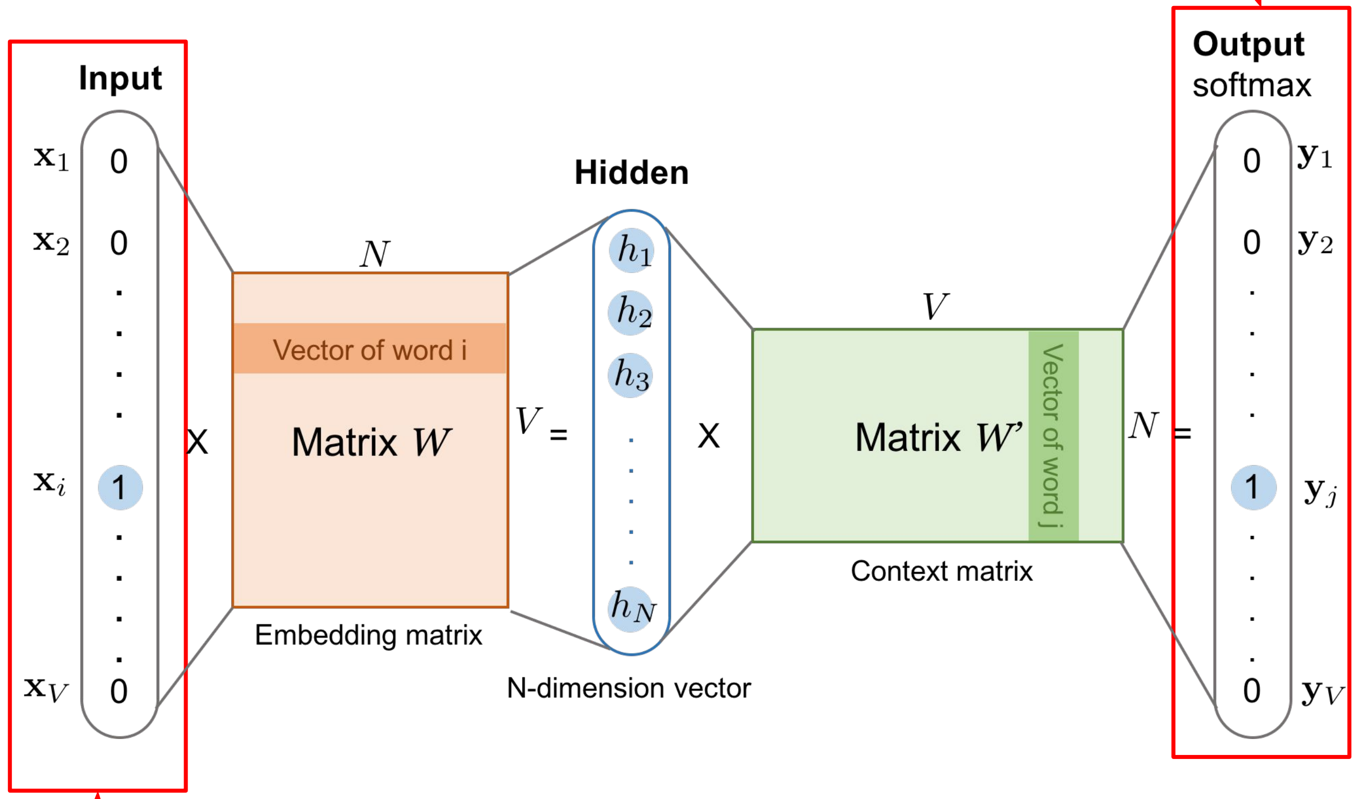
Skip-gram: The simpler and more efficient way to learn word embeddings.

Context and target words are selected randomly

*Instead of using the 4 previous words as the context, we randomly choose a word from the phrase as **the context word** and randomly choose **the target word within a window** of 3 or 5 words close to the context word.*

In this way, we can have more samples (pairs target context) to train the supervised problem.

Softmax output wiht the same number of words in the dictionary



Word in one hot representation

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

Sources:
- Coursera

5. Word2Vec & Skip-gram

Disadvantage of this skip-gram algorithm:

- **Computational speed.** *It performs the softmax for the 10.000 words in the dictionary. Sometimes the dictionary can even have 1 million words*

$$p(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10,000} e^{\theta_j^T e_c}}$$

This problem can be mitigated by using hierarchical softmax, which is a way of making softmax faster.

OR we can make this method even faster by using a different way to sample the context and the target words.

Let's see that!

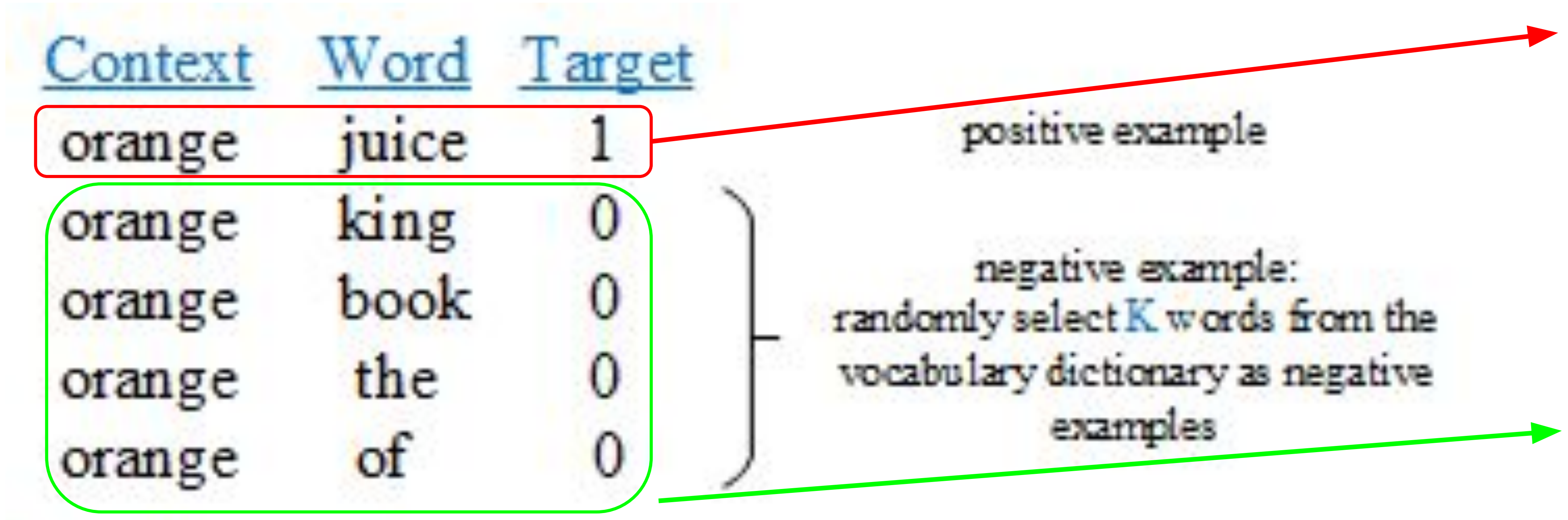
Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

Sources:
- Coursera

6. Word2Vec & Skip-gram (Negative sampling)

Negative Sampling method basically defines a new learning problem to train the supervised architecture that learns the word embedding.

Say for instance we have the phrase/sentence ***"I want a glass of orange juice to go along with my cereal"***



Context	Word	Target
orange	juice	1
orange	king	0
orange	book	0
orange	the	0
orange	of	0

positive example
negative example: randomly select K words from the vocabulary dictionary as negative examples

Pick a random context and target word that is positive. That is the first row.

Then, pick the same context word and randomly choose K times words from the dictionary as negative words.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems 26 (2013): 3111-3119.

Sources:
- Coursera
- https://createmomo.github.io/2018/01/23/Super-Machine-Learning-Revision-Notes/

6. Word2Vec & Skip-gram (Negative sampling)

Binary classification *after creating context/target words*

Once you create the positive and negative samples, you used that database to train a binary classification system.

Making thousands of binary classifications is faster than doing softmax of 10.000 neurons (the size of the dictionary)

In the original paper they proposed a way of sampling negative samples that is based on word frequency

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems 26 (2013): 3111-3119.

Sources:
- Coursera
- https://createmomo.github.io/2018/01/23/Super-Machine-Learning-Revision-Notes/

7. GloVe word vectors

GloVe (global vectors for word representation) is not as used as Skip-gram but it has some momentum.

This algorithm works a bit different than previous ones as the loss function is different. It first defines the term:

X_{ij} = ***It is the number of times word j occurs in the context of word i.***

Loss function to get the word embeddings

Weighting term

$$J = \sum_{i,j=1}^{V} \underbrace{f\left(X_{ij}\right)}_{} \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

$$f(x) = \begin{cases} (x/\overset{100}{x_{\max}})^{\overset{3/4}{\alpha}} & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543. 2014.

Sources:
- Coursera
- https://jonathan-hui.medium.com/nlp-word-embedding-glove-5e7f523999f6

8. Sentiment classification

Sentiment classification is the task of looking at a piece of text and telling if someone likes or dislikes the thing they're talking about.

It is one of the most important building blocks of Natural Language Processing (NLP)

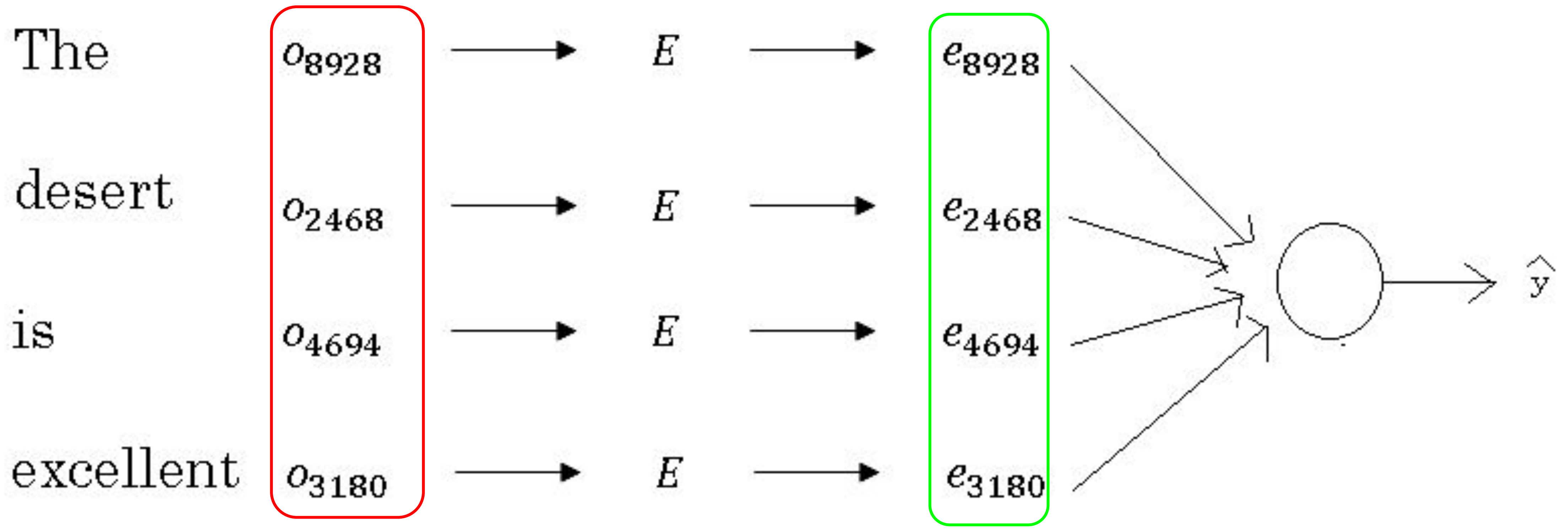
Inputs	***Labels***
The dessert is excellent.	★★★★☆
Service was quite slow.	★★☆☆☆
Good for a quick meal, but nothing special.	★★★☆☆
Completely lacking in good taste, good service, and good ambience.	★☆☆☆☆

Sources:
- Coursera

8. Sentiment classification

Word embeddings in sentiment classification

Using a simple model to classify sentiment in a phrase



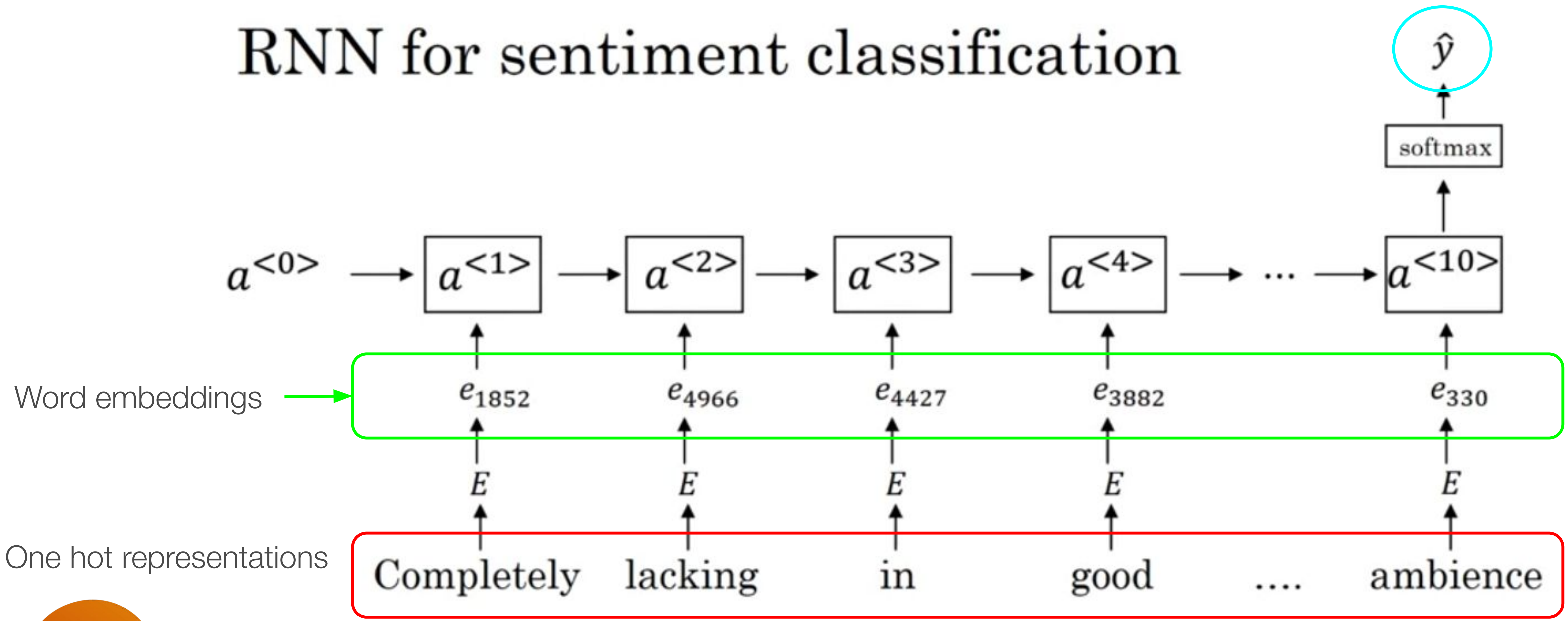
1. Very bad
2. Bad
3. Neutro
4. Good
5. Very good

However, this model ignores word order.
For instance, the phrase **"Completely lacking in good taste, good service, and good ambience"** will be classified as a "good" comment because of the number of "good" words in the phrase

Sources:
- Coursera

8. Sentiment classification

Many-to-one architecture



Sources:
- Coursera

9. Debiasing word embeddings

Language models and word embeddings reflect gender, ethnicity, age, sexual orientation, and other biases of the text used to train the model!

Because machine learning tools are being used to make decisions such as loan applications and criminal justice system (sentencing guidelines), **it is important to remove these negative, undesirable and toxic biases.**

Authors of the paper **"Man is to computer programmer as woman is to homemaker? debiasing word embeddings"** proposed a way to do this.

IT IS STILL AN AREA OF RESEARCH

Bolukbasi, Tolga, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. "Man is to computer programmer as woman is to homemaker? debiasing word embeddings." *Advances in neural information processing systems* 29 (2016): 4349-4357.

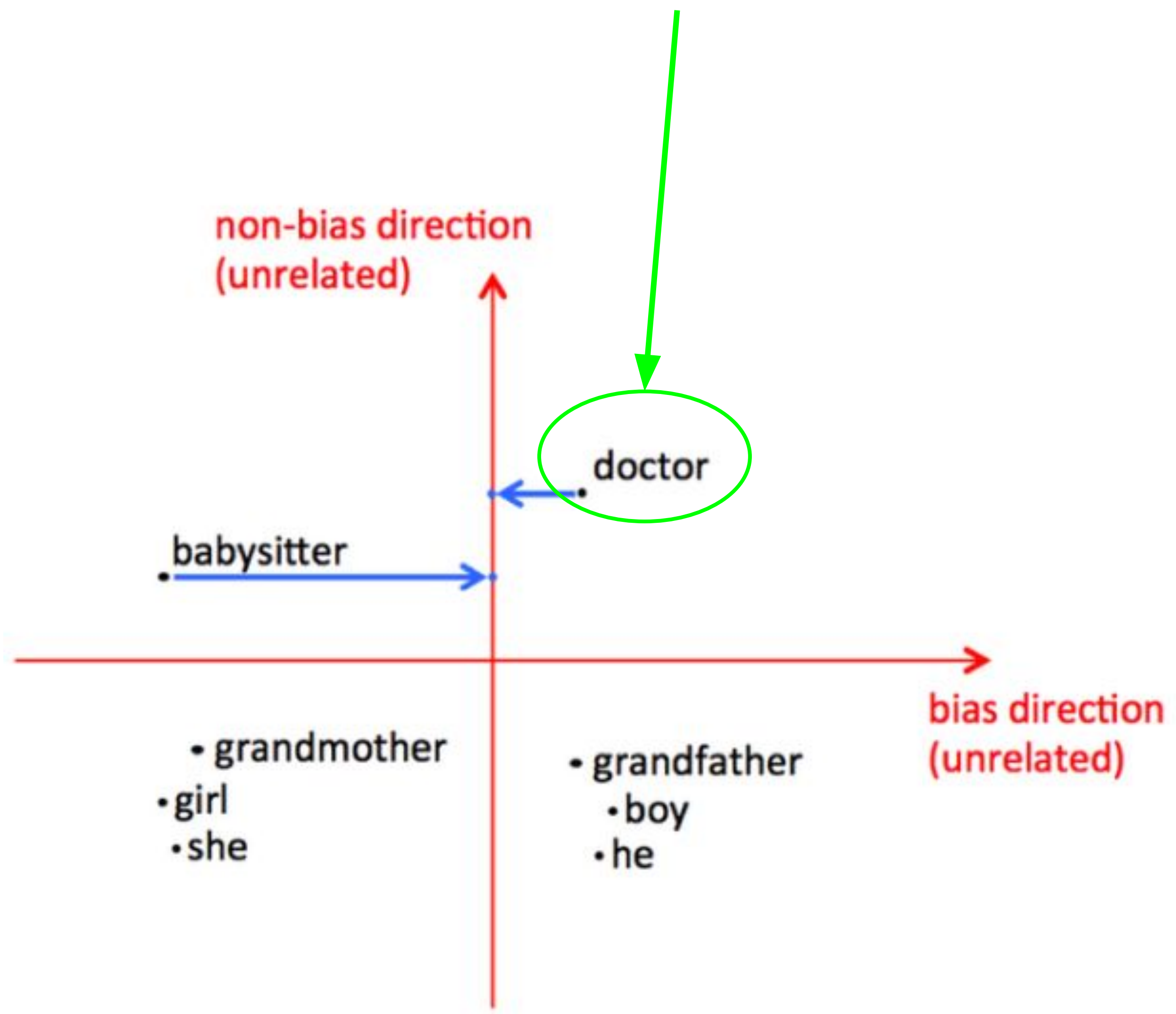
Sources:
- Coursera

9. Debiasing word embeddings

The word doctor is biased to man

Steps to address bias:

1. Identify bias direction
2. Neutralize: For every word that is not definitional, project to get rid of bias.
   A definitional word doesn't have, by definition, a gender. For instance, doctor or receptionist are non definitional words as they are roles that can be occupied by any gender.
3. Equalize pairs



Bolukbasi, Tolga, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. "Man is to computer programmer as woman is to homemaker? debiasing word embeddings." *Advances in neural information processing systems* 29 (2016): 4349-4357.

Sources:
- Coursera
- https://vagdevik.wordpress.com/2018/07/08/debiasing-word-embeddings/

Let’s move to Google Colab!

Notebooks:

- *12_Emojify.ipynb*

Sources:
- Coursera