CS224N : Lecture 1 - Intro & Word Vectors

📌 Word meaning can be representing rather well by a large vector of real numbers.

---

1. Human language and word meaning

- **How do we represent the meaning of a word?**
  - **WordNet** : A thesaurus containing lists of **synonym sets and hypernyms**
    - ▼ Problems with resources like WordNet

      (1) Missing nuance

      (2) Missing new meanings of words

      (3) Subjective

      (4) Requires human labor to create and adapt

      (5) Can't compute accurate word similarity
    - ▼ Problems of the traditional NLP
      - Regard words as discrete symbols → Represent words by **one-hot vectors**
      - Very huge vector dimension (= number of words in vacabulary)
      - If vectors of the word are orthogonal → There is no natural notion of similarity for one-hot vectors
        - Solution : Learn to encode similarity in the vectors themselves

- **Distributional semantics** : A word's meaning is given by the words that frequently appear close-by (**contexts**)

2. Word2vec algorithm introduction

- **Word vectors**
  - **Distribted** Representation
    - = Word embeddings
    - = (Neural) Word representations
  - **Similar contexts** appears as **similar vector of words**
- **Word2vec** : framework for learning word vectors
  - User the **similarity of the word vectors** for certain words to **calculate the probability** of them
  - Keep adjusting the word vectors to **maximize the probability**

3. Word2vec objective function gradients

- **Likelihood** : **predict context words** within a window of fixed size *m*
- **Objective function** : (average) **negative log likelihood**

> Minimizing objective function ⇔ Maximizing predictive accuracy

▼ ***Slides***

For each position $t = 1, \dots, T$, predict context words within a window of fixed size *m*, given center word w_j. Data likelihood:

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} \mid w_t; \theta)$$

θ is all variables to be optimized

sometimes called a *cost* or *loss* function

The objective function $J(\theta)$ is the (average) negative log likelihood:

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log P(w_{t+j} \mid w_t; \theta)$$

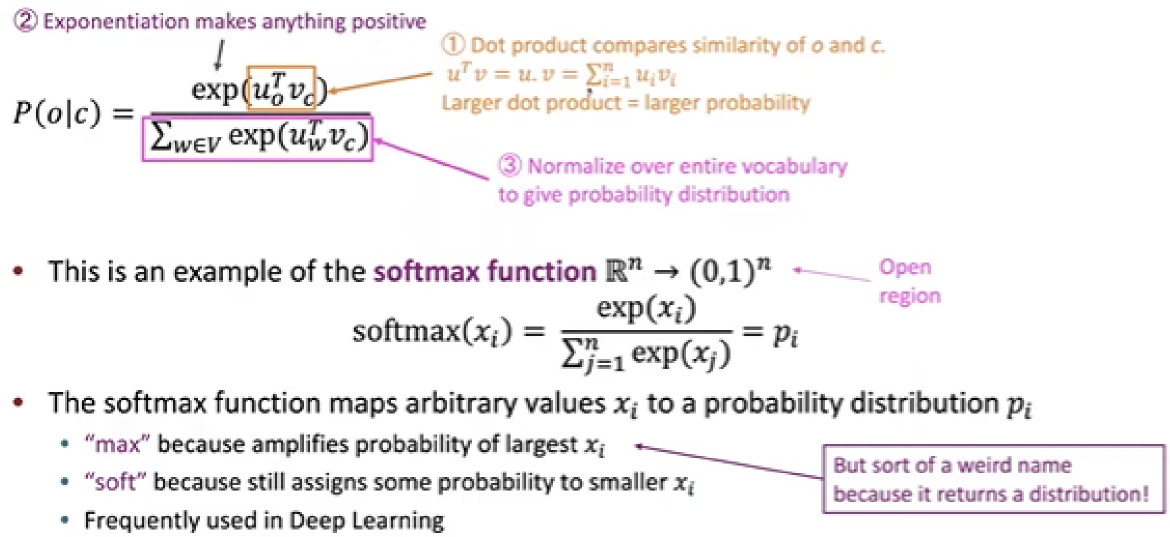
Minimizing objective function ⟺ Maximizing predictive accuracy

- We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log P(w_{t+j} \mid w_t; \theta)$$

- **Question:** How to calculate $P(w_{t+j} \mid w_t; \theta)$?
- **Answer:** We will *use two* vectors per word *w*:
  - v_w when *w* is a center word
  - u_w when *w* is a context word
- Then for a center word *c* and a context word *o*:

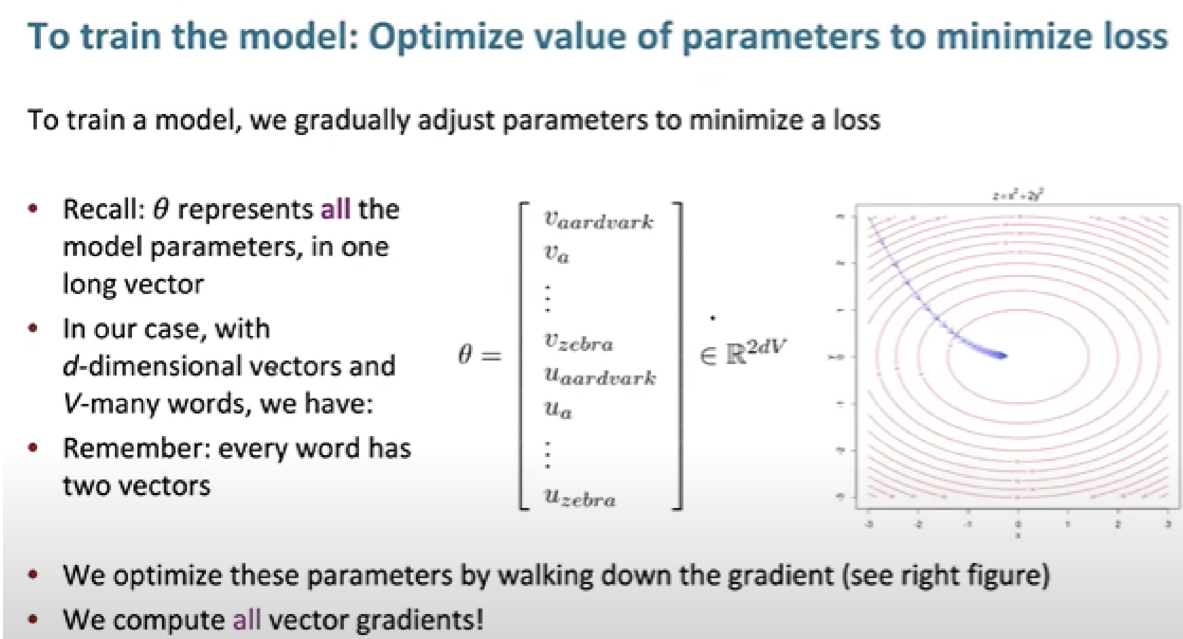
$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$



4. Optimization basics

- **Optimization**
  - Adjust parameters to **minimize a loss**
    - Optimize parameters by walking down the gradient (compute all vector gradient)

▼ *Slides*



5. Looking at word vectors