Neural Language Model, RNNs

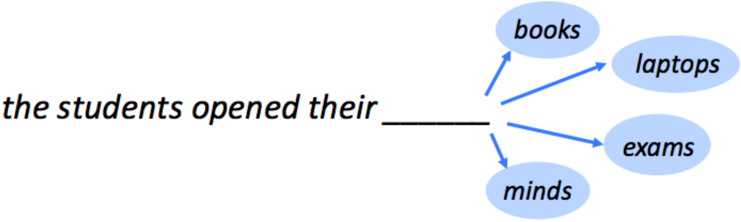
Pawan Goyal

CSE, IIT Kharagpur

CS60010

Language Modeling

Language Modeling is the task of predicting what word comes next.



- **Goal:** Compute the probability of a sentence or sequence of words:

$$\boxed{P(W) = P(w_1, w_2, w_3, \ldots, w_n)}$$

n-gram LMs

- **Related Task:** probability of an upcoming word:

$$P(w_4 | w_1, w_2, w_3)$$

- A model that computes either of these is called a **language model**

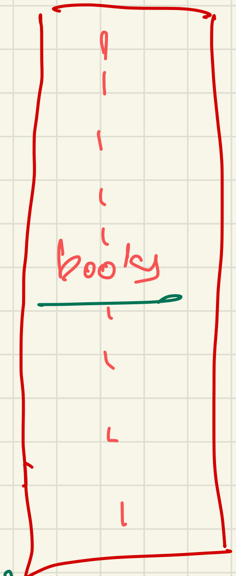
Why should we care about language modeling?

- Language Modeling is a benchmark task that helps us measure our progress on understanding language
- Language Modeling is fundamental to many NLP tasks, especially those involving generating text or estimating the probability of text:
  - Predictive typing
  - Speech recognition
  - Handwriting recognition
  - Spelling/grammar correction

How will I build an LM?

Large corpus

the students opened their ____



Vocabulary
[Every unique word in the corpus]

$$P(\text{books} \mid \text{the students opened their})$$

$$= \frac{\text{Count}(\text{the students opened their books})}{\text{Count}(\text{the students opened their})}$$

n-gram LM

$O(V) \leftarrow P(\text{books}) \rightarrow$ unigram LM

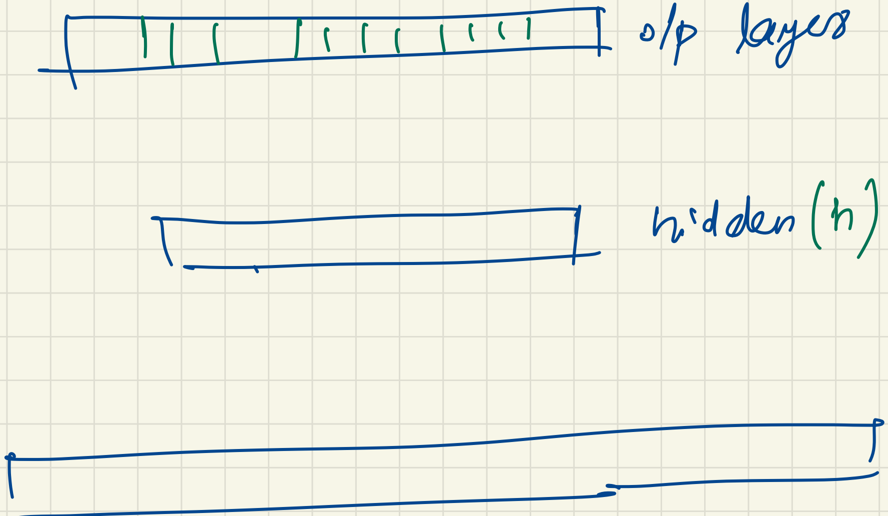
$O(V^2)$ ← $P(\text{books} \mid \text{their})$ → bigram LM

$O(V^3)$ ← $P(\text{books} \mid \text{opened their})$ → trigram

- Storage issues

- Does not generalize

NN

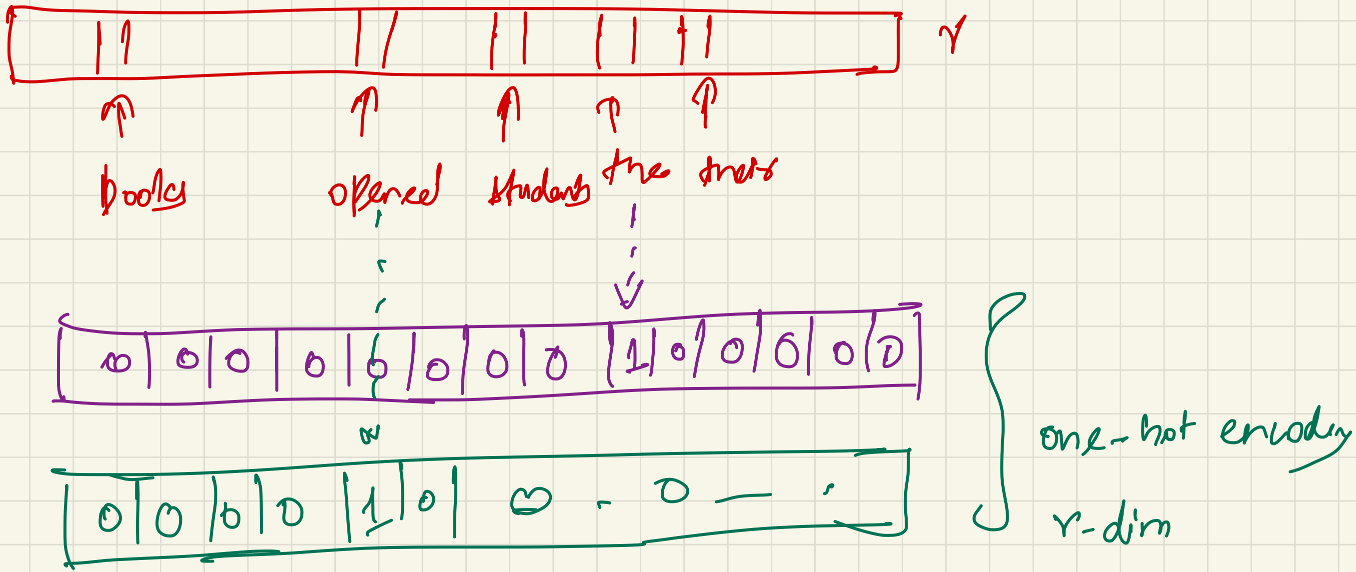


o/p layer (V - dim)

hidden (h)

I/p (the students opened their)

→ How to feed this as i/p?


V
books
opened
students
the
one-hot encoding
V-dim

A fixed-window neural language model

Students

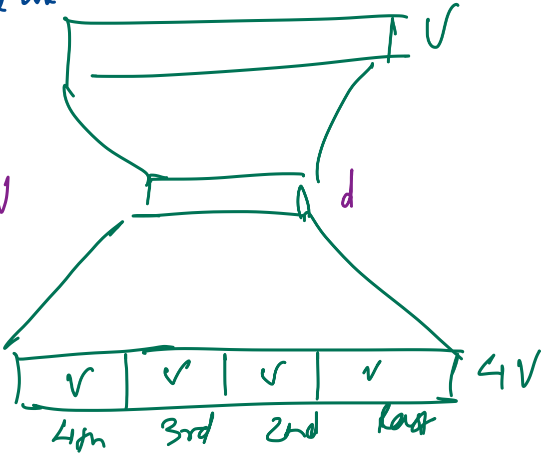
student → separate one-hot vectors

Can't generalize

$4V \times d + dV$

$= 5dV$

exponential → linear



~~as the proctor started the clock~~ the students opened their ______

discard

fixed window

pupils

A fixed-window neural language model

output distribution

$\hat{\boldsymbol{y}} = \mathrm{softmax}(\boldsymbol{U}\boldsymbol{h} + \boldsymbol{b}_2) \in \mathbb{R}^{|V|}$

hidden layer

$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{e} + \boldsymbol{b}_1)$

$d << |V|$

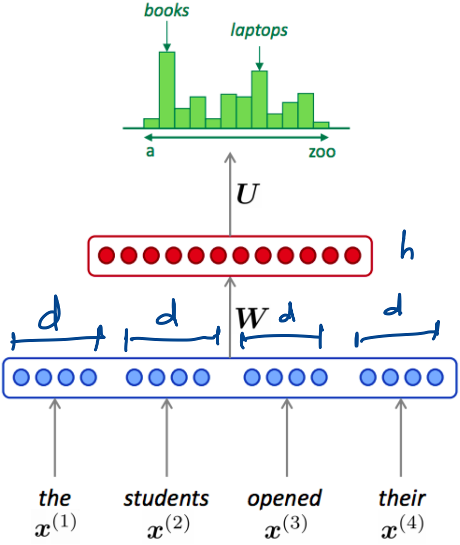
concatenated word embeddings

$4d$

$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$

words / one-hot vectors

$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$



How do we obtain word representations?

In traditional NLP / IR, words are treated as discrete symbols.

One-hot representation

Words are represented as one-hot vectors: one 1, the rest 0s

motel [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0] AND
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0] = 0

What is the problem?

- Vector dimension = number of words in vocabulary (e.g., 500,000)
- The vectors are orthogonal, and there is no natural notion of similarity between one-hot vectors!

motel [0.1 0.5 | | | |] d

hotel [0.2 0.7] d

Self-supervision :-

Take a large corpus

Given a word, predict surroundings

Distributional representation – word embedding?

Any word w_i in the corpus is given a distributional representation by an embedding

$$w_i \in R^d$$

i.e., a $d-$dimensional vector, which is mostly learnt!

$$\textit{linguistics} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Learning Word Vectors: Overview

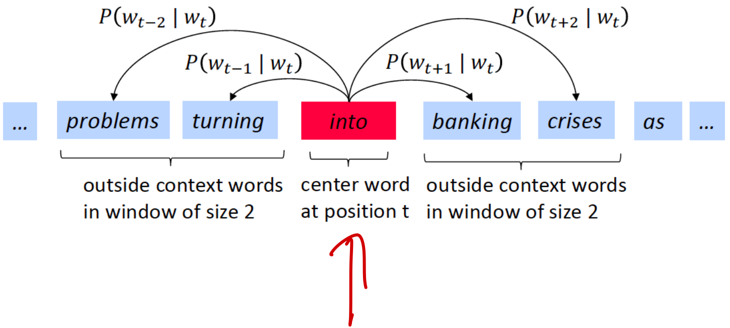
in-vector $V \times d$ out-vector $V \times d$

Basic Idea: Use self-supervision

- We have a large corpus of text
- Every word in a fixed vocabulary is represented by a *vector*
- Go through each position t in the text, which has a center word c and context ("outside") words o
- Use the similarity of the word vectors for c and o to calculate the probability of o given c (or vice versa)
- Keep adjusting the word vectors to maximize this probability

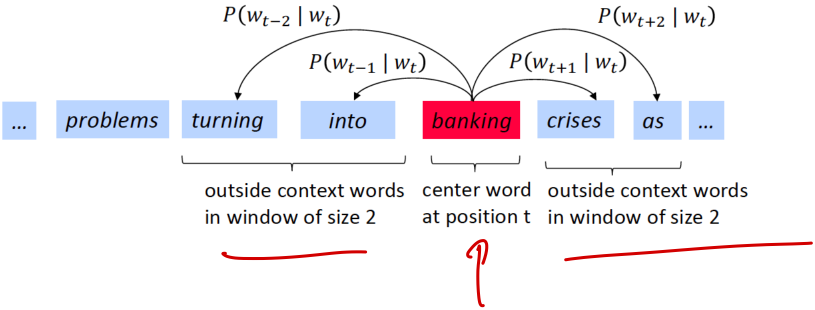
Word2Vec (Skip-gram) Overview

Example windows and process for computing $P(w_{t+j}|w_t)$



Word2Vec Overview

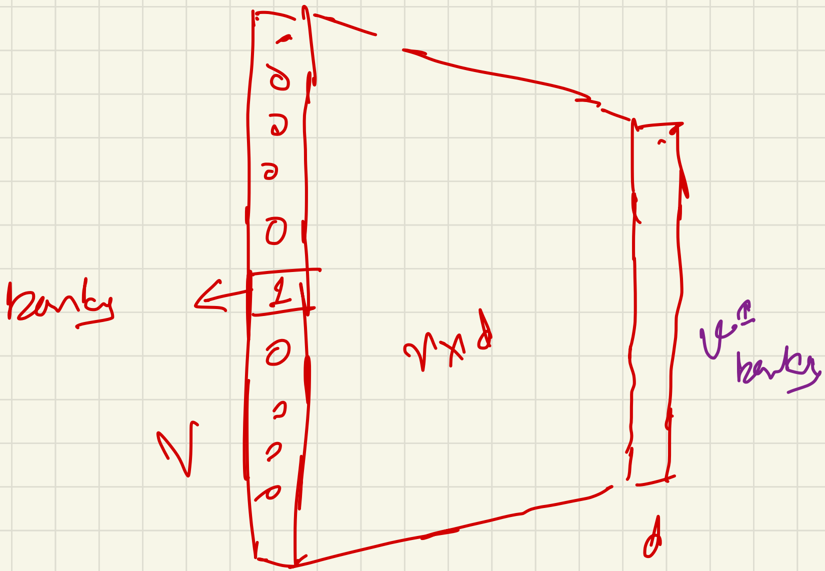
Example windows and process for computing $P(w_{t+j}|w_t)$

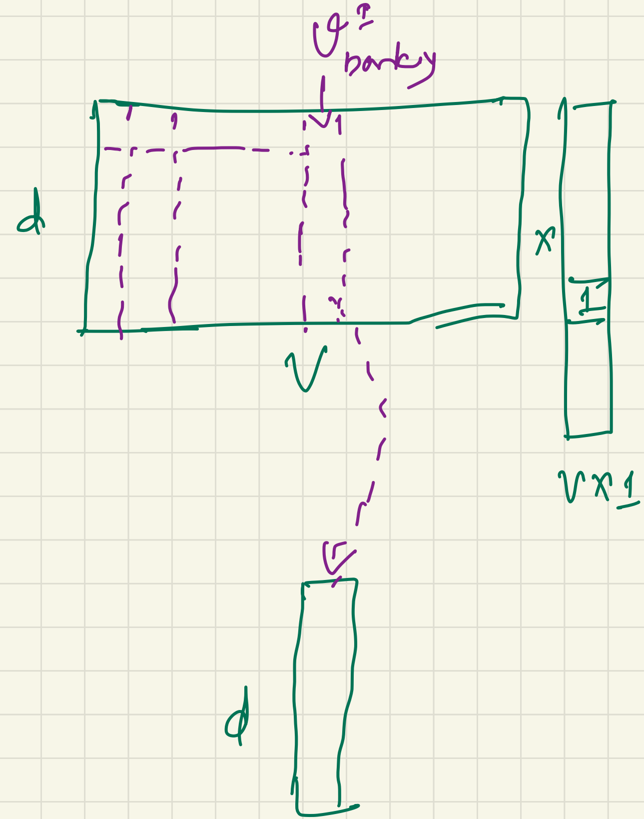


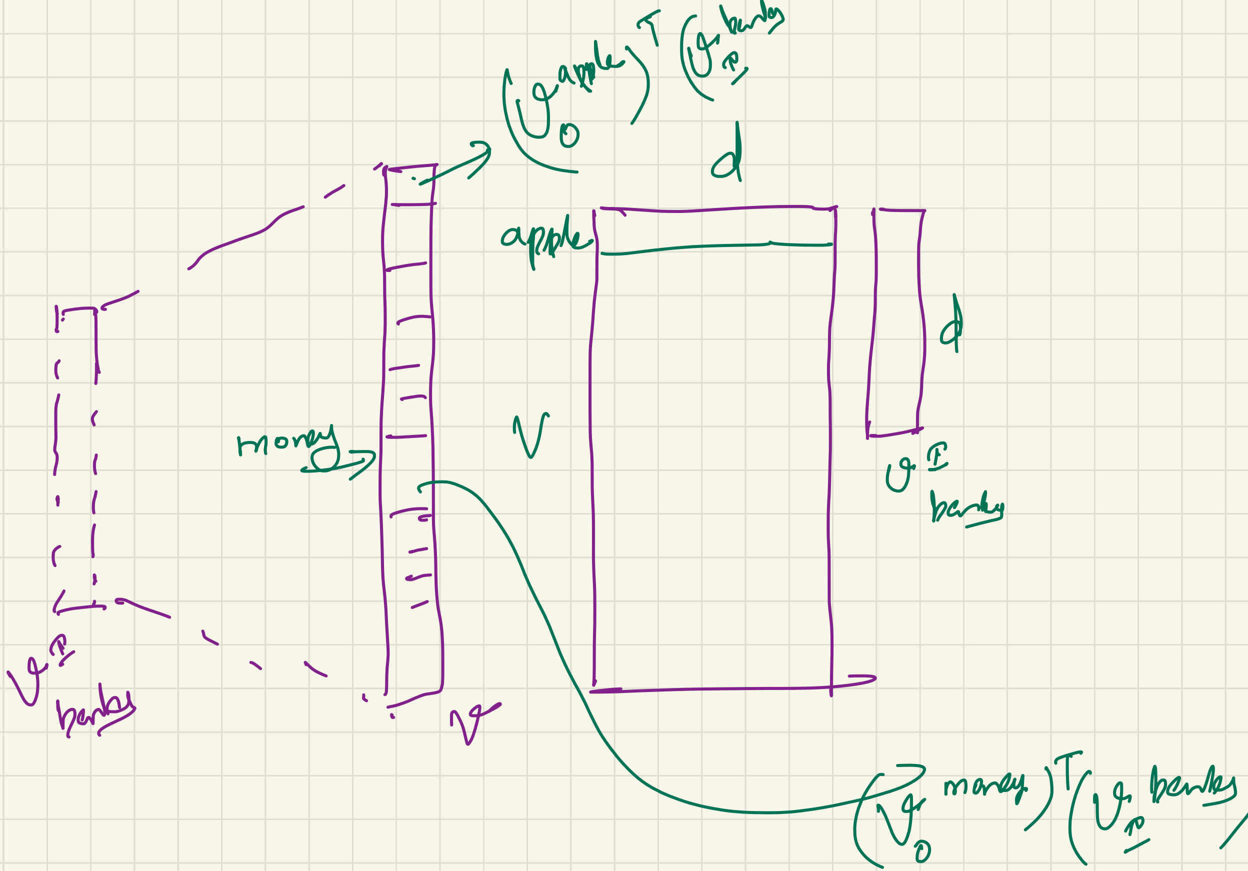
banking $\rightarrow$ v^{I}_{banking} $\rightarrow$ money v^{O}_{money}

finance $\rightarrow$ v^{I}_{finance}

hot cold


banky
1
0
V
V x d
d
v^T banky


v^T banky
d
V
x
1
V x 1
v
d


$(v_o^{apple})^T (v_I^{banky})$
d
apple
V
d
money
v_I^{banky}
v_I^{banky}
V
$(v_o^{money})^T (v_I^{banky})$