Lecture:
Natural Language Processing

Chapter 06: Feedforward and Recurrent Neural Language Models

Tim Metzler, Jörn Hees

2024-05-14

Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Fachbereich Informatik
Department of Computer Science

Plan (Preliminary)

Date	Lec.	Topics
2024-04-09	1	Org, Introduction to NLP, Text Processing
2024-04-16	2	Text Processing and Tokenization
2024-04-23	3	Statistical Language Models
2024-04-30	4	Static Embeddings (1-Hot, TF-IDF)
2024-05-07	5	Static Embeddings (Word2Vec)
2024-05-14	6	Feedforward & Recurrent Neural Language Models
2024-05-21	7	Attention and the Transformer Architecture
2024-05-28	8	Transformers and Applications
2024-06-04	9	Generative Pretrained Transformers and Large Language Models
2024-06-11	10	Instruction-Following Language Models
2024-06-18	11	Model Adaptation, Retrieval Augmented Generation
2024-06-25	12	Project: Run your own LLM
2024-07-02	13	Project: Presentations
2024-07-09	14	Summary & Questions

Plan updated!

Feedforward Neural Language Models

Quiz: How would you build a simple sentiment classifier?

- Input: Document (e.g., amazon review)
  - Examples:
    - “This blender is great, it shredded my iphone without any problems”
    - “The blender is no good ☹”
    - “Broken after first use!”
    - “The dessert was delicious”
    - “I wouldn’t recommend the desserts at this place”
- Output: Sentiment classification (binary: positive / negative)

Simple Text Classifiers: Ideas

- Manual Feature Engineering
- Embeddings

Sentiment Classifier: Manual Feature Engineering

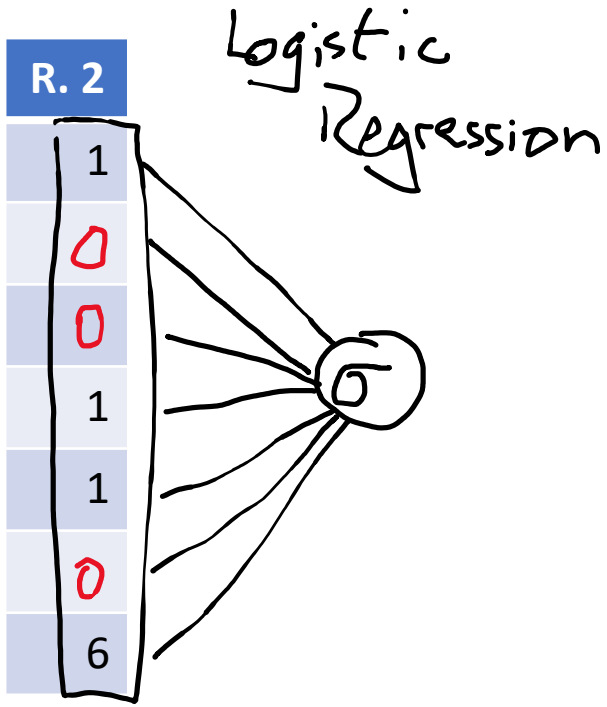
1. “This blender is great, it shredded my iphone without any problems”
2. “The blender is no good ☹”
3. “Broken after first use!”
4. “The dessert was delicious”
5. “I wouldn’t recommend the desserts at this place”

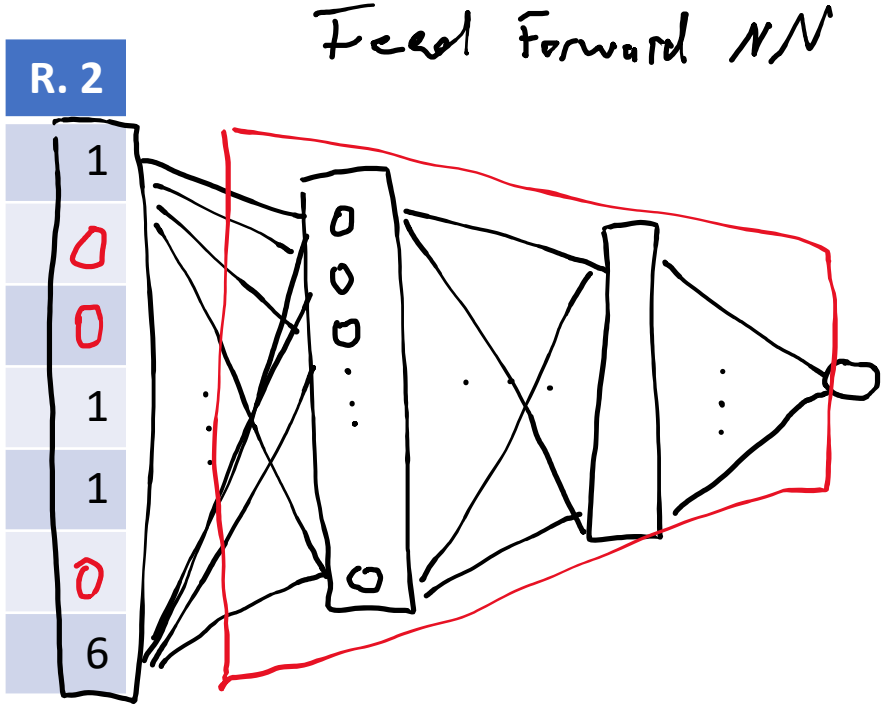
- Feature ideas:
  - Count positive words
  - Count negative words
  - Count positive emoticons
  - Count negative emoticons
  - Does the review contain negations?
  - Does the review include “!”?
  - Length of review

R. 1	R. 2	R. 3	R. 4	R. 5
1	1		1	1
2		1		
	1			
1	1			1
		1		
11	6	5	4	8

Sentiment Classifier: Manual Feature Engineering

- Manual Feature Engineering turned each doc into a feature vector
- Let's plug them into a simple Feed Forward Neural Network





Sentiment Classifier: Embeddings

- Hand crafted features
  - often useful
  - might reach their limits
- Why not let the network learn what's important?
- Textual feature representations / Embeddings:
  - One-Hot
  - TF-IDF
  - Word2Vec
  - …

Sentiment Classifier: NN based on Embeddings

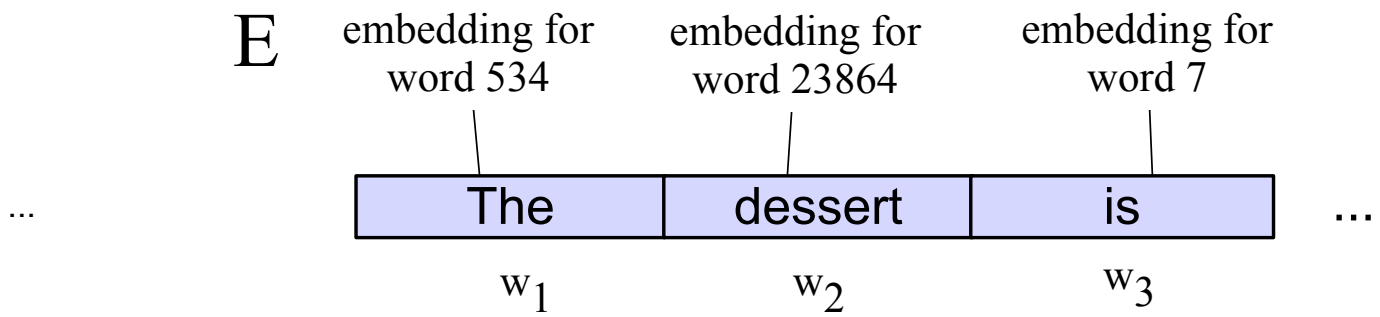


Image Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Sentiment Classifier: NN based on Embeddings

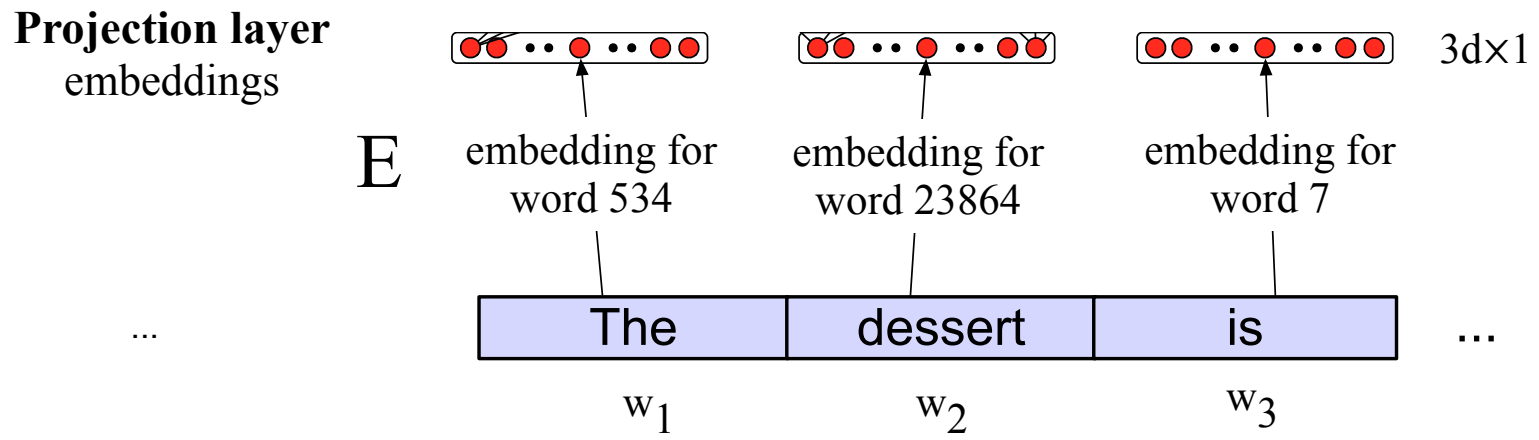


Image Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Sentiment Classifier: NN based on Embeddings

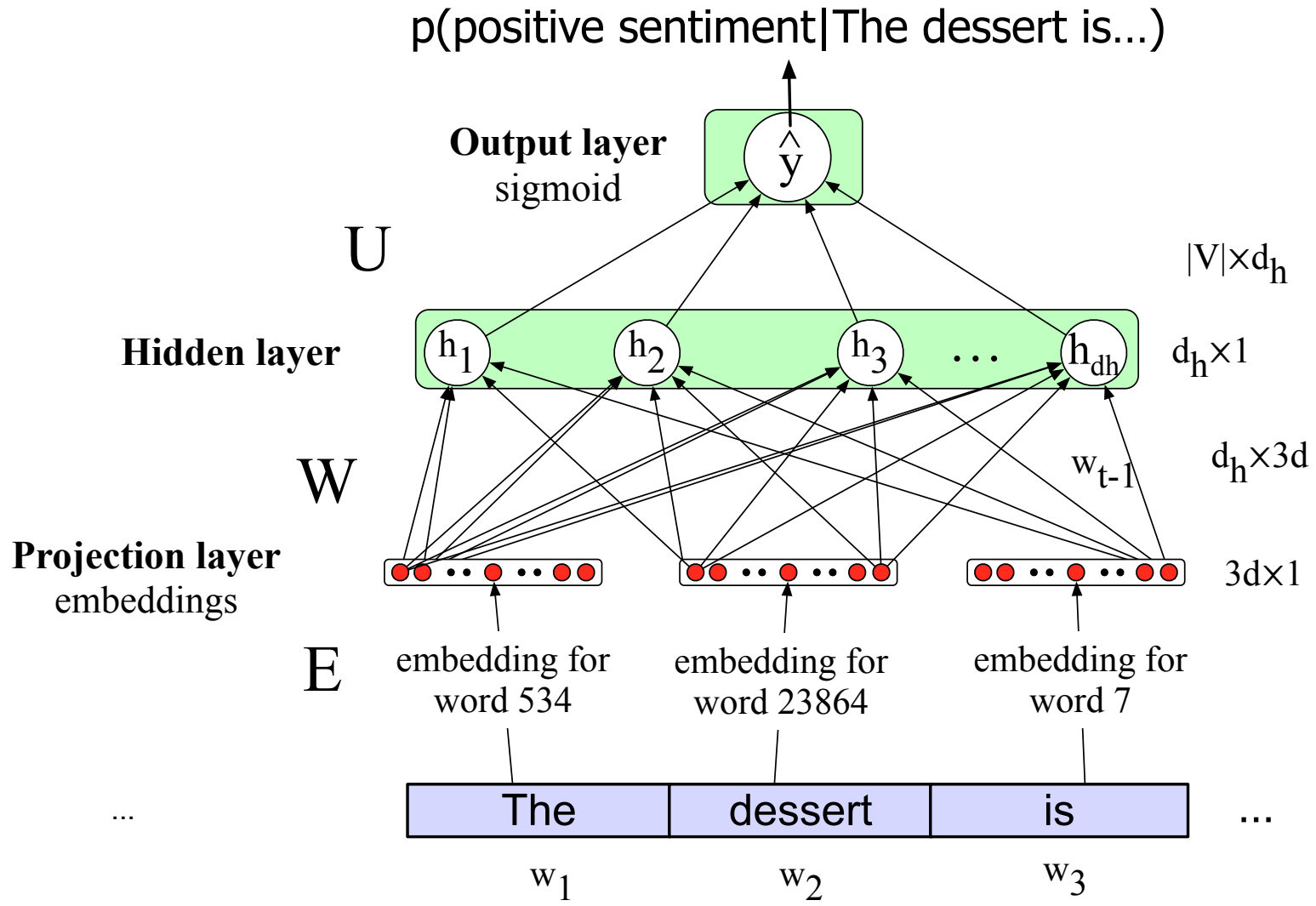
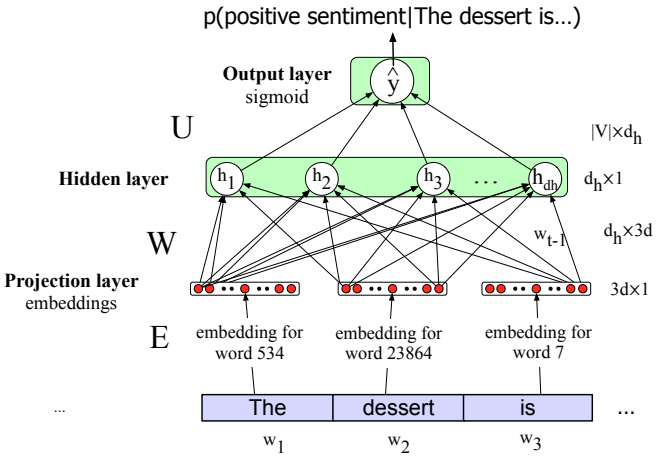


Image Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

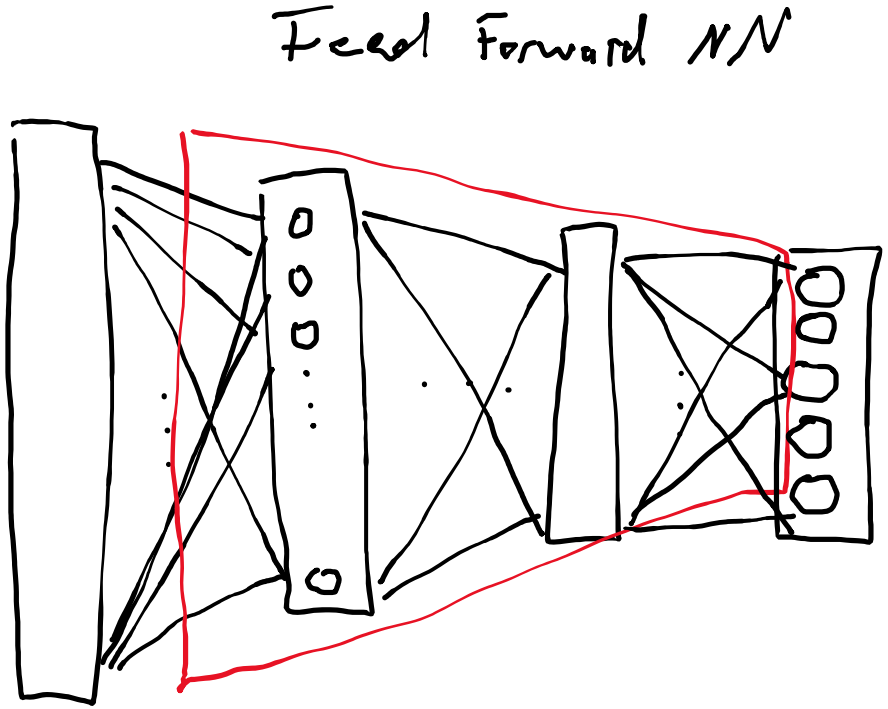
Sentiment Classifier: NN based on Embeddings

- Issues?
  - Only works for fixed length inputs!
  - In this case: 3 words
- Workarounds?
  - Set input size to fixed length (e.g., that of longest review)
    - Pad shorter with 0 vectors
    - Truncate longer ones
  - Create single fixed size review embedding (sentence / paragraph / document)
    - Mean of all word embeddings
    - Element-wise max of all word embeddings



Simple Text Classifiers: More Classes

- General idea applicable not only to (binary) sentiment classification



Simple Language Modeling Task

Neural Language Models (LMs)

Language Modeling: Calculating the probability of the next word in a sequence given some history.

- We've seen N-gram based LMs
- But neural network LMs far outperform n-gram language models

State-of-the-art neural LMs are based on more powerful neural network technology like Transformers

But **simple feedforward LMs** can do almost as well!

Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Simple feedforward Neural Language Models

Task: predict next word w_t

given prior words w_{t-1}, w_{t-2}, w_{t-3}, …

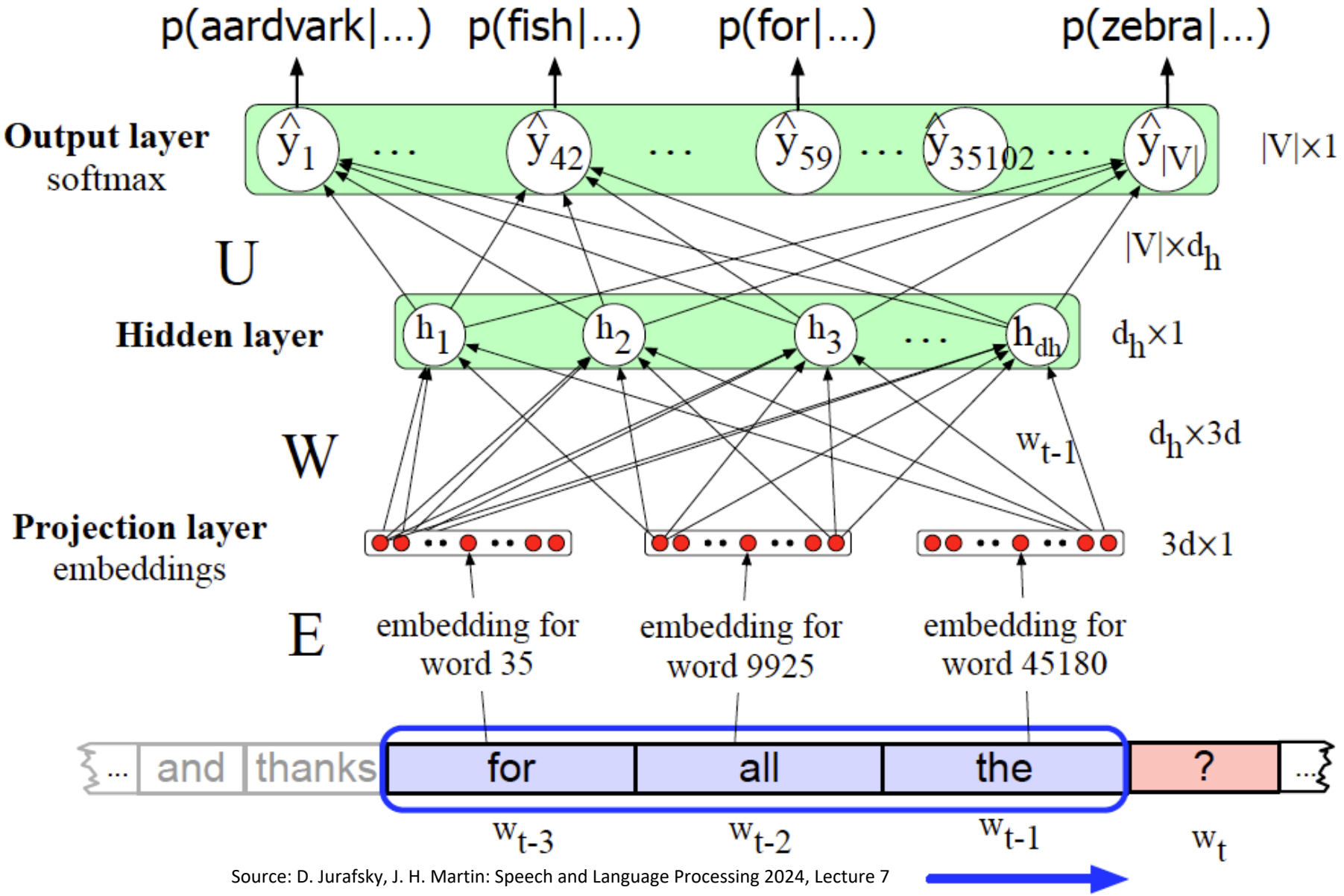
Problem: Now we're dealing with sequences of arbitrary length.

Solution: Sliding windows (of fixed length)

$$P(w_t|w_1^{t-1}) \approx P(w_t|w_{t-N+1}^{t-1})$$

Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Neural Language Model



Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Why Neural LMs work better than N-gram LMs

Training data:

We've seen: I have to make sure that the cat gets fed.

Never seen: dog gets fed

Test data:

I forgot to make sure that the dog gets ___

N-gram LM can't predict "fed"!

Neural LM can use similarity of "cat" and "dog" embeddings to generalize and predict “fed” after dog

Source: D. Jurafsky, J. H. Martin: Speech and Language Processing 2024, Lecture 7

Next

Recurrent Neural Language Models