CS 378 Lecture 15: Language Modeling, n-grams, RNNs

Announcements

- Midterm back soon (Mon?)
- A4 released Tues
- Vicente Ordóñez talk Fri 11am 6.302
- Custom FP proposals due in one week (email to me)

Today

- Intro to language modeling
- N-gram LMs
- Neural LMs
- RNNs (brief!)
- Transformers on Tuesday

Recap (course so far)

Input text → classify (sentiment)
→ POS
→ syntactic parsing

text → label/structure

Next few weeks:

text → text (seq2seq)

machine translation, dialogue systems, summarization, QA, ...

Today Language Modeling

"autocomplete" / predictive text

RNNs / Transformers

LM: a distribution $P(\overline{w})$ over sequences of words

Assign high prob. to real, grammatical, natural sentences

Grammatical error correction:

$\overline{w}$, try changing some function words, find $\overline{w}'$

$P(\overline{w}') > P(\overline{w}) \Rightarrow$ we fixed some errors?

N-gram language modeling

$\overline{w} = (w_1, \dots, w_m)$ m words

By the chain rule of probability:

$$P(\bar{w}) = P(w_1)\, P(w_2 | w_1)\, P(w_3 | w_1 w_2) \cdots P(w_n | w_1 \cdots w_{m-1})$$

no assumptions

$$P(\bar{w}) = P(w_1)\, P(w_2 | w_1)\, P(w_3 | w_2)$$

2-gram LM

n-gram LM: depend on past n-1 words

$$P(\bar{w}) = \prod_{i=1}^{n} P(w_i | w_{i-n+1}, \ldots, w_{i-1})$$

$n=4$: $w_{i-3}\, w_{i-2}\, w_{i-1}$

the cat ran

2-gram LM: start of sequence

$$P_2(\overline{w}) = P(\text{the} \mid <s>)\, P(\text{cat} \mid \text{the})$$
$$P(\text{ran} \mid \text{cat})\, \cancel{P(STOP \mid \text{ran})}$$

3-gm

$$P_3(\overline{w}) = P(\text{the} \mid <s> <s>) \cdot$$
$$P(\text{cat} \mid <s>\ \text{the})$$
$$P(\text{ran} \mid \text{the cat})$$

Ex

I saw the dog ______ lots of completions

yesterday bark owner

2^{-3} sleep store

in

at

P(_ | Texas is)
P(_ | cap. of TX is)

The capital of Texas is ______

fewer completions
trickier!
≥ 4 words

Austin
far
hot

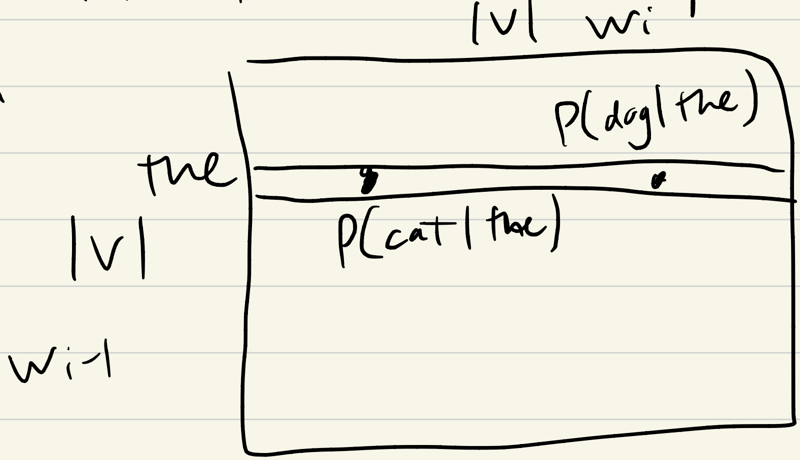
LM is a very fundamental task

N-gram parameterization

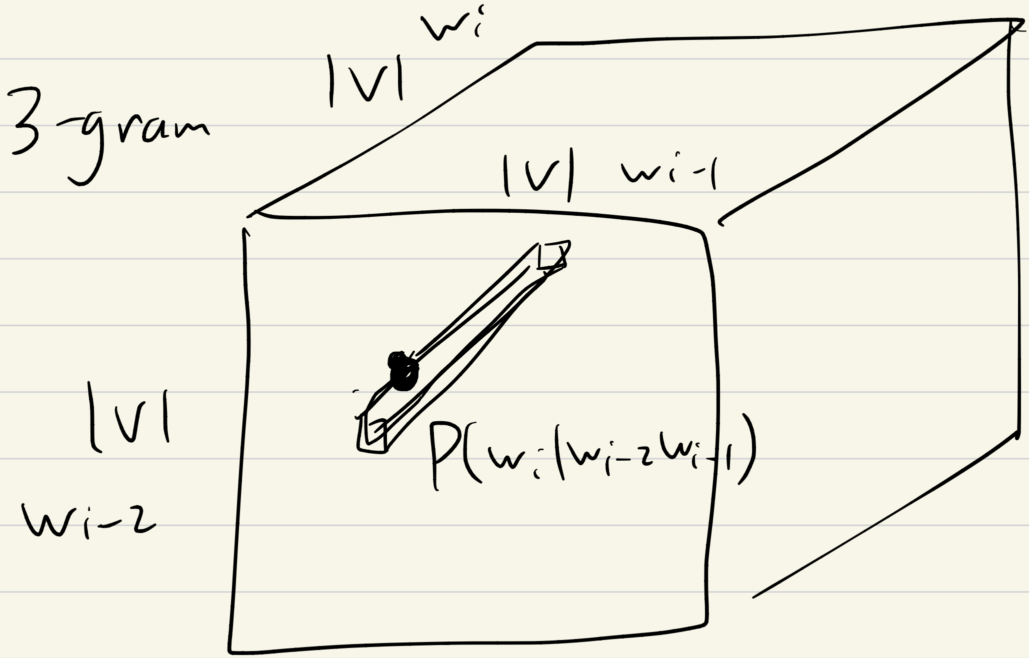
V vocab

Like HMM transitions: lookup table

2-gram



3-gram



sparse: not all 3-grams exist
actual size $<< |V|^3$

$n \sim 7$ is about as high as you can go

Parameters: Count + normalize over a big corpus

the cat
the cat
the dog
the snake

$P(\text{cat} \mid \text{the}) = \frac{1}{2}$

$P(\text{snake} \mid \text{the}) = \frac{1}{4}$

Ex

$n = 5$, I $\underbrace{\text{hate to go to Maui}}_{5 \text{ words}}$

$$P(\text{Maui} \mid \text{hate to go to}) = \frac{\text{count}(\text{hate to go to Maui})}{\text{count}(\text{hate to go to})} = 0$$

P^{raw}_5 //

Smoothing

$$P_5(w_i \mid w_{i-4}\ w_{i-3}\ w_{i-2}\ w_{i-1})$$

$$= \lambda \cdot P_5^{raw}(w_i \mid w_{i-4}\ {}_{-3}\ {}_{-2}\ {}_{-1})$$

$$+ (1-\lambda)\boxed{P_4}(w_i \mid w_{i-3}\ {}_{-2}\ {}_{-1})$$

$\lambda = 0.9$

$$\boxed{P_4} = \lambda \cdot P_4^{raw} + (1-\lambda)\boxed{P_3}$$ recursive

$\vdots$

$$\boxed{P_1} = P_1^{raw}(w_i) > 0$$ any $w_i \in V$ is seen ≥ 1 time

What do we store:

P_5^{raw} "$|V|^5$" $P_3^{raw}, P_2^{raw}, P_1^{raw}$

P_4^{raw} "$|V|^4$" tiny

Counts are sparse

distribution is not sparse

Many smoothing schemes

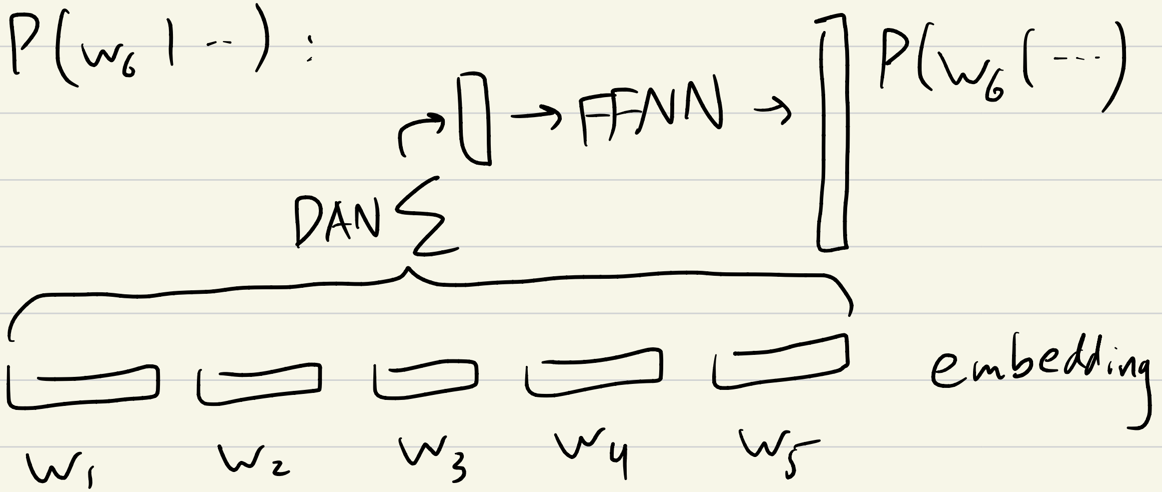
absolute discounting: sets the λ dynamically

Kneser-Ney smoothing

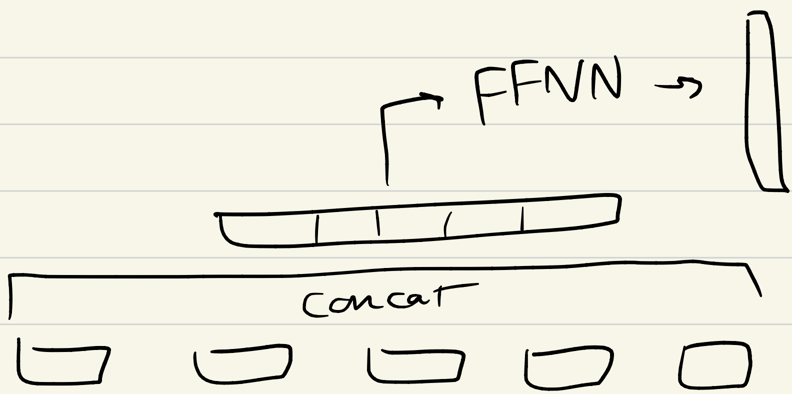
Neural Language Modeling

$P(w_i \mid w_1, \ldots, w_{i-1}) \Rightarrow$ model w/a NN

$P(w_6 \mid \ldots)$:

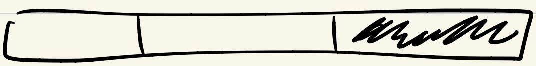


Con: ignore ordering



Cons: require weights for every word in every position

the movie was ___



___

my favorite movie

May be hard to scale / different sizes of input

Two solutions:

① RNN: process sequences "uniformly"

② Transformer: halfway between DAN + concat

RNN will maintain a hidden state $\overline{h}$ after processing n words

$$\overline{h}_n = \text{neural net}(w_1, \dots, w_n)$$

$$P(w_{n+1} \mid w_1 \dots w_n) = \text{softmax}(W \overline{h}_n)$$

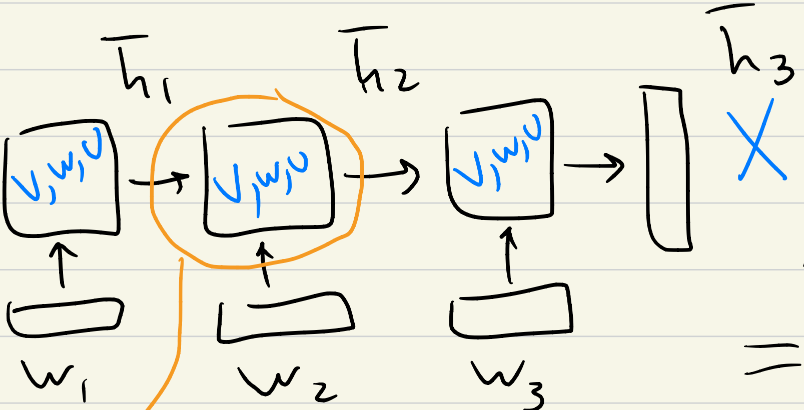
W : $|V| \times d$ matrix

$\overline{h}$: d-dimensional vector

$$\overline{h}_{n+1} = \text{neural net}(\overline{h}_n, w_{n+1})$$

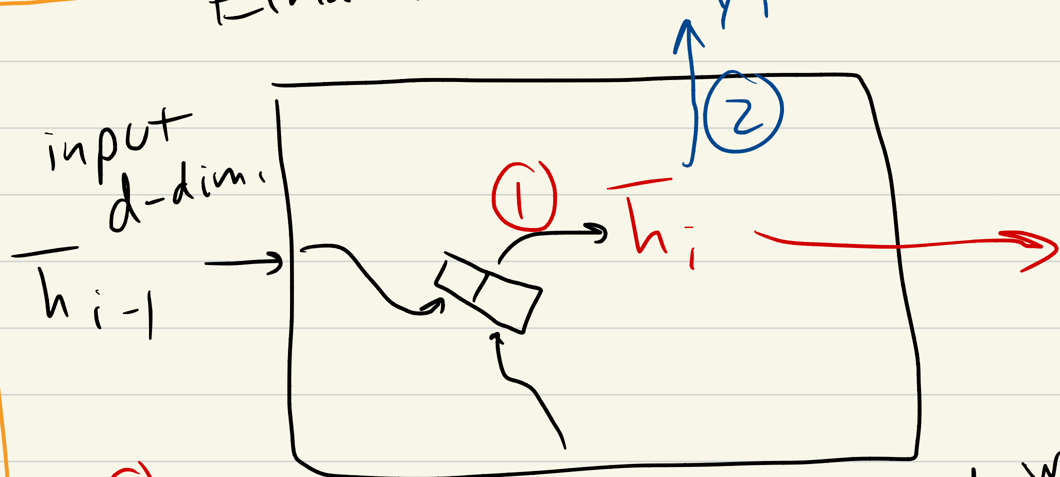
easy to update

<u>RNN</u> encodes a sequence of vecs. into a single vector



$$P(w_4 | w_{1..3}) = \mathrm{softmax}(X \bar{h}_3)$$

Elman network:



$$\bar{h}_i = \tanh(W \bar{w}_i + V \bar{h}_{i-1}) \quad (1)$$

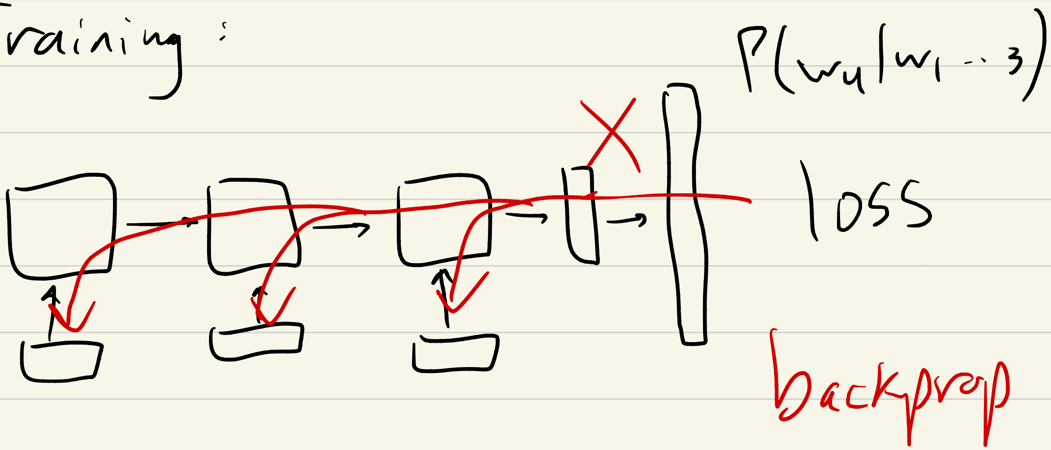
$$\bar{y}_i = \tanh(U \bar{h}_i) \quad (2)$$

W: d × word emb.

V: d × d

U: ? × d

Training:



updates V, W, U

update term from every cell

Elman network

LSTM