Language Modeling, Smoothing, and the Noisy Channel Model

Dr. Sambit Praharaj, Assistant Professor (II), KIIT

1 Introduction to Language Modeling

A **Language Model (LM)** assigns a probability to a sequence of words. Given a sentence:

$$W = w_1, w_2, \dots, w_n$$

a language model estimates:

$$P(W) = P(w_1, w_2, \dots, w_n)$$

Higher probability implies a more natural or fluent sentence.

Applications

- Speech recognition
- Machine translation
- Spell correction
- Text prediction and autocomplete

2 Chain Rule of Probability

Using the chain rule:

$$P(w_1, w_2, \dots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \cdots P(w_n|w_1, \dots, w_{n-1})$$

Problem: Conditioning on all previous words is computationally infeasible.
Solution: Use the **Markov assumption** $\Rightarrow$ N-gram models.

3 N-Gram Language Models

An **N-gram model** assumes the probability of a word depends only on the previous $N - 1$ words.

Model	Probability Assumption
Unigram	$P(w_i)$
Bigram	$P(w_i\|w_{i-1})$
Trigram	$P(w_i\|w_{i-2}, w_{i-1})$

4 Bigram Language Model

4.1 Definition

The bigram approximation is:

$$P(w_i|w_1,\ldots,w_{i-1}) \approx P(w_i|w_{i-1})$$

4.2 Example Corpus

```
I love NLP
I love AI
```

Vocabulary:

$$V = \{\text{I, love, NLP, AI}\}$$

4.3 Counts

Bigram	Count
(I, love)	2
(love, NLP)	1
(love, AI)	1

4.4 Bigram Probability

$$P(love|I) = \frac{2}{2} = 1$$

$$P(NLP|love) = \frac{1}{2}$$

Sentence probability:

$$P(\text{I love NLP}) = P(I|\langle s\rangle) \times P(love|I) \times P(NLP|love)$$

5 Trigram Language Model

5.1 Definition

$$P(w_i|w_1,\ldots,w_{i-1}) \approx P(w_i|w_{i-2},w_{i-1})$$

5.2 Formula

$$P(w_i|w_{i-2},w_{i-1}) = \frac{\text{Count}(w_{i-2},w_{i-1},w_i)}{\text{Count}(w_{i-2},w_{i-1})}$$

5.3 Advantage and Limitation

- Better contextual modeling
- Suffers from data sparsity

6 Zero Probability Problem

If an N-gram never appears in training:

$$\text{Count} = 0 \Rightarrow P = 0$$

This causes the entire sentence probability to become zero, even if the sentence is reasonable.

7 Laplace (Add-One) Smoothing

7.1 Idea

Assign a small probability to unseen events by adding 1 to all counts.

7.2 Bigram Laplace Formula

$$P(w_i|w_{i-1}) = \frac{\text{Count}(w_{i-1}, w_i) + 1}{\text{Count}(w_{i-1}) + V}$$

7.3 Numerical Example

Given:

- Vocabulary size $V = 5$
- $\text{Count}(I) = 2$
- $\text{Count}(I, hate) = 0$

$$P(hate|I) = \frac{0+1}{2+5} = \frac{1}{7}$$

7.4 Effect of Laplace Smoothing

- Avoids zero probability
- Redistributes probability mass
- Over-smooths frequent events

8 Backoff Smoothing

8.1 Core Idea

If a higher-order N-gram is unseen, back off to a lower-order model.

$$\text{Trigram} \rightarrow \text{Bigram} \rightarrow \text{Unigram}$$

8.2 Formal Definition

$$P(w_i|w_{i-2}, w_{i-1}) = \begin{cases} P_{\text{tri}}(w_i|w_{i-2}, w_{i-1}) & \text{if count} > 0 \\ \alpha P_{\text{bi}}(w_i|w_{i-1}) & \text{otherwise} \end{cases}$$

8.3 Example

Sentence:

I love AI

- Trigram $(I, love, AI)$ unseen
- Back off to $P(AI|love)$
- If unseen again, use unigram $P(AI)$

Thus, the sentence gets a non-zero probability.

9 Noisy Channel Model

9.1 Motivation

We observe a noisy sentence O and want to recover the intended sentence S.

$$\hat{S} = \arg\max_S P(S|O)$$

Using Bayes' rule:

$$\hat{S} = \arg\max_S P(O|S)P(S)$$

9.2 Components

- $P(S)$: Language model (fluency)
- $P(O|S)$: Channel model (noise)

9.3 Example

Observed sentence:

I sea you

Candidates:

- I see you
- I sea you
- Language model prefers *see*
- Channel model explains typo: see $\rightarrow$ sea

Final decision:

$$\arg\max_S P(O|S)P(S)$$

10 Key Takeaways

- Language models assign probabilities to word sequences
- N-grams approximate long histories
- Smoothing avoids zero probabilities
- Backoff balances context and reliability
- Noisy channel separates fluency and noise