Language Modeling

Michael Collins, Columbia University

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

The Language Modeling Problem

- We have some (finite) vocabulary,
  say $\mathcal{V} = \{\text{the, a, man, telescope, Beckham, two, } \ldots\}$

- We have an (infinite) set of strings, $\mathcal{V}^\dagger$

  the STOP
  a STOP
  the fan STOP
  the fan saw Beckham STOP
  the fan saw saw STOP
  the fan saw Beckham play for Real Madrid STOP

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English
- We need to "learn" a probability distribution p
  i.e., p is a function that satisfies

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \quad p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

The Language Modeling Problem (Continued)

- We have a *training sample* of example sentences in English
- We need to "learn" a probability distribution p
  i.e., p is a function that satisfies

$$\sum_{x \in \mathcal{V}^\dagger} p(x) = 1, \quad p(x) \geq 0 \text{ for all } x \in \mathcal{V}^\dagger$$

$p(\text{the STOP}) = 10^{-12}$
$p(\text{the fan STOP}) = 10^{-8}$
$p(\text{the fan saw Beckham STOP}) = 2 \times 10^{-8}$
$p(\text{the fan saw saw STOP}) = 10^{-15}$
...
$p(\text{the fan saw Beckham play for Real Madrid STOP}) = 2 \times 10^{-9}$
...

Why on earth would we want to do this?!

- **Speech recognition** was the original motivation. (Related problems are optical character recognition, handwriting recognition.)

Why on earth would we want to do this?!

- **Speech recognition** was the original motivation. (Related problems are optical character recognition, handwriting recognition.)
- The estimation techniques developed for this problem will be **VERY** useful for other problems in NLP

A Naive Method

- We have N training sentences
- For any sentence $x_1 \ldots x_n$, $c(x_1 \ldots x_n)$ is the number of times the sentence is seen in our training data
- A naive estimate:

$$p(x_1 \ldots x_n) = \frac{c(x_1 \ldots x_n)}{N}$$

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Markov Processes

- Consider a sequence of random variables $X_1, X_2, \ldots X_n$. Each random variable can take any value in a finite set $\mathcal{V}$. For now we assume the length n is fixed (e.g., $n = 100$).

- Our goal: model

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

First-Order Markov Processes

一阶马尔科夫

$$P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n)$$

马尔可夫模型本质上是一个加权的有限状态机，它描述了不同状态之间的转换关系以及转换概率（这里的权重就是状态转移概率）。

常见的是一阶马尔科夫：当前状态出现的概率，只取决于上一个状态

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1})
\end{aligned}
$$

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}) \\
= \quad & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})
\end{aligned}
$$

First-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}) \\
= \quad & P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})
\end{aligned}
$$

The first-order Markov assumption: For any $i \in \{2 \ldots n\}$, for any $x_1 \ldots x_i$,

$$P(X_i = x_i | X_1 = x_1 \ldots X_{i-1} = x_{i-1}) = P(X_i = x_i | X_{i-1} = x_{i-1})$$

Second-Order Markov Processes

$$P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n)$$

Second-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \times P(X_2 = x_2 | X_1 = x_1) \\
& \times \prod_{i=3}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

Second-Order Markov Processes

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= \quad & P(X_1 = x_1) \times P(X_2 = x_2 | X_1 = x_1) \\
& \times \prod_{i=3}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1}) \\
= \quad & \prod_{i=1}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

(For convenience we assume $x_0 = x_{-1} =$ *, where * is a special "start" symbol.)

Modeling Variable Length Sequences

- We would like the length of the sequence, n, to also be a random variable
- A simple solution: always define $X_n = \text{STOP}$ where STOP is a special symbol

Modeling Variable Length Sequences

- We would like the length of the sequence, n, to also be a random variable
- A simple solution: always define $X_n = \text{STOP}$ where STOP is a special symbol
- Then use a Markov process as before:

$$
\begin{aligned}
& P(X_1 = x_1, X_2 = x_2, \ldots X_n = x_n) \\
= & \prod_{i=1}^{n} P(X_i = x_i | X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})
\end{aligned}
$$

(For convenience we assume $x_0 = x_{-1} = *$, where * is a special "start" symbol.)

Trigram Language Models

- A trigram language model consists of:
  1. A finite set $\mathcal{V}$
  2. A parameter $q(w|u,v)$ for each trigram u,v,w such that $w \in \mathcal{V} \cup \{\text{STOP}\}$, and $u,v \in \mathcal{V} \cup \{\text{*}\}$.

Trigram Language Models

- A trigram language model consists of:
  1. A finite set $\mathcal{V}$
  2. A parameter $q(w|u,v)$ for each trigram u,v,w such that $w \in \mathcal{V} \cup \{\text{STOP}\}$, and $u,v \in \mathcal{V} \cup \{*\}$.

- For any sentence $x_1 \ldots x_n$ where $x_i \in \mathcal{V}$ for $i = 1 \ldots (n-1)$, and $x_n = \text{STOP}$, the probability of the sentence under the trigram language model is

$$p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i|x_{i-2}, x_{i-1})$$

where we define $x_0 = x_{-1} = *$.

The dog barks STOP

An Example

For the sentence

the dog barks STOP

we would have

$$
\begin{aligned}
p(\text{the dog barks STOP}) = \; & q(\text{the}|*, *) \\
& \times q(\text{dog}|*, \text{the}) \\
& \times q(\text{barks}|\text{the, dog}) \\
& \times q(\text{STOP}|\text{dog, barks})
\end{aligned}
$$

The Trigram Estimation Problem

Remaining estimation problem:

$$q(w_i \mid w_{i-2}, w_{i-1})$$

For example:

$$q(\text{laughs} \mid \text{the, dog})$$

怎么算?

The Trigram Estimation Problem

Remaining estimation problem:

$$q(w_i \mid w_{i-2}, w_{i-1})$$

For example:

$$q(\text{laughs} \mid \text{the, dog})$$

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Sparse Data Problems

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Say our vocabulary size is $N = |\mathcal{V}|$, then there are N^3 parameters in the model.

e.g., $N = 20,000 \quad \Rightarrow \quad 20,000^3 = 8 \times 10^{12}$ parameters

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

PPL 困惑度

Evaluating a Language Model: Perplexity

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

Evaluating a Language Model: Perplexity

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

- We could look at the probability under our model $\prod_{i=1}^{m} p(s_i)$. Or more conveniently, the *log probability*

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i)$$

Evaluating a Language Model: Perplexity

混淆度 (Perplexity) 用来衡量一个语言模型在未见过的的字符串S上的表现。

- We have some test data, m sentences

$$s_1, s_2, s_3, \ldots, s_m$$

- We could look at the probability under our model $\prod_{i=1}^{m} p(s_i)$. Or more conveniently, the *log probability*

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i)$$

- In fact the usual evaluation measure is *perplexity*

$$\text{Perplexity} = 2^{-l} \quad \text{where} \quad l = \frac{1}{M} \sum_{i=1}^{m} \log p(s_i)$$

and M is the total number of words in the test data.

Some Intuition about Perplexity

- Say we have a vocabulary $\mathcal{V}$, and $N = |\mathcal{V}| + 1$ and model that predicts

$$q(w|u, v) = \frac{1}{N}$$

  for all $w \in \mathcal{V} \cup \{\text{STOP}\}$, for all $u, v \in \mathcal{V} \cup \{*\}$.
- Easy to calculate the perplexity in this case:

$$\text{Perplexity} = 2^{-l} \quad \text{where} \quad l = \log \frac{1}{N}$$

$\Rightarrow$

$$\text{Perplexity} = N$$

Perplexity is a measure of effective "branching factor"

平均分支系数

示例:训练好的bigram语言模型的困惑度为?

- $p(w_1|BOS)=0, p(w_2|BOS)=1, p(w_3|BOS)=0$;
- $p(w_1|w_1)=\frac{1}{3}, p(w_2|w_1)=\frac{1}{3}, p(w_3|w_1)=\frac{1}{3}$;
- $p(w_1|w_2)=\frac{1}{3}, p(w_2|w_2)=\frac{1}{3}, p(w_3|w_2)=\frac{1}{3}$;
- $p(w_1|w_3)=\frac{1}{3}, p(w_2|w_3)=\frac{1}{3}, p(w_3|w_3)=\frac{1}{3}$;
- $p(EOS|w_1)=\frac{1}{3}, p(EOS|w_2)=\frac{1}{3}, p(EOS|w_3)=\frac{1}{3}$;

计算$perplexity(w_2, w_1, w_3)$的值

$$
\begin{aligned}
perplexity(w_2, w_1, w_3)^{-\frac{1}{3}} &= \sqrt[3]{\frac{1}{p(w_2|BOS) \times p(w_1|w_2) \times p(w_3|w_1) \times p(EOS|w_3)}} \\
&= \sqrt[3]{\frac{1}{1 \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3}}} \\
&= 3
\end{aligned}
$$

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$
- A bigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-1})$. Perplexity $= 137$

Typical Values of Perplexity

- Results from Goodman ("A bit of progress in language modeling"), where $|\mathcal{V}| = 50,000$
- A trigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-2}, x_{i-1})$. Perplexity $= 74$
- A bigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i | x_{i-1})$. Perplexity $= 137$
- A unigram model: $p(x_1 \ldots x_n) = \prod_{i=1}^{n} q(x_i)$. Perplexity $= 955$

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Sparse Data Problems

A natural estimate (the "maximum likelihood estimate"):

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

$$q(\text{laughs} \mid \text{the, dog}) = \frac{\text{Count(the, dog, laughs)}}{\text{Count(the, dog)}}$$

Say our vocabulary size is $N = |\mathcal{V}|$, then there are N^3 parameters in the model.

e.g., $N = 20,000 \quad \Rightarrow \quad 20,000^3 = 8 \times 10^{12}$ parameters

The Bias-Variance Trade-Off

- Trigram maximum-likelihood estimate

$$q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) = \frac{\mathsf{Count}(w_{i-2}, w_{i-1}, w_i)}{\mathsf{Count}(w_{i-2}, w_{i-1})}$$

- Bigram maximum-likelihood estimate

$$q_{\mathsf{ML}}(w_i \mid w_{i-1}) = \frac{\mathsf{Count}(w_{i-1}, w_i)}{\mathsf{Count}(w_{i-1})}$$

- Unigram maximum-likelihood estimate

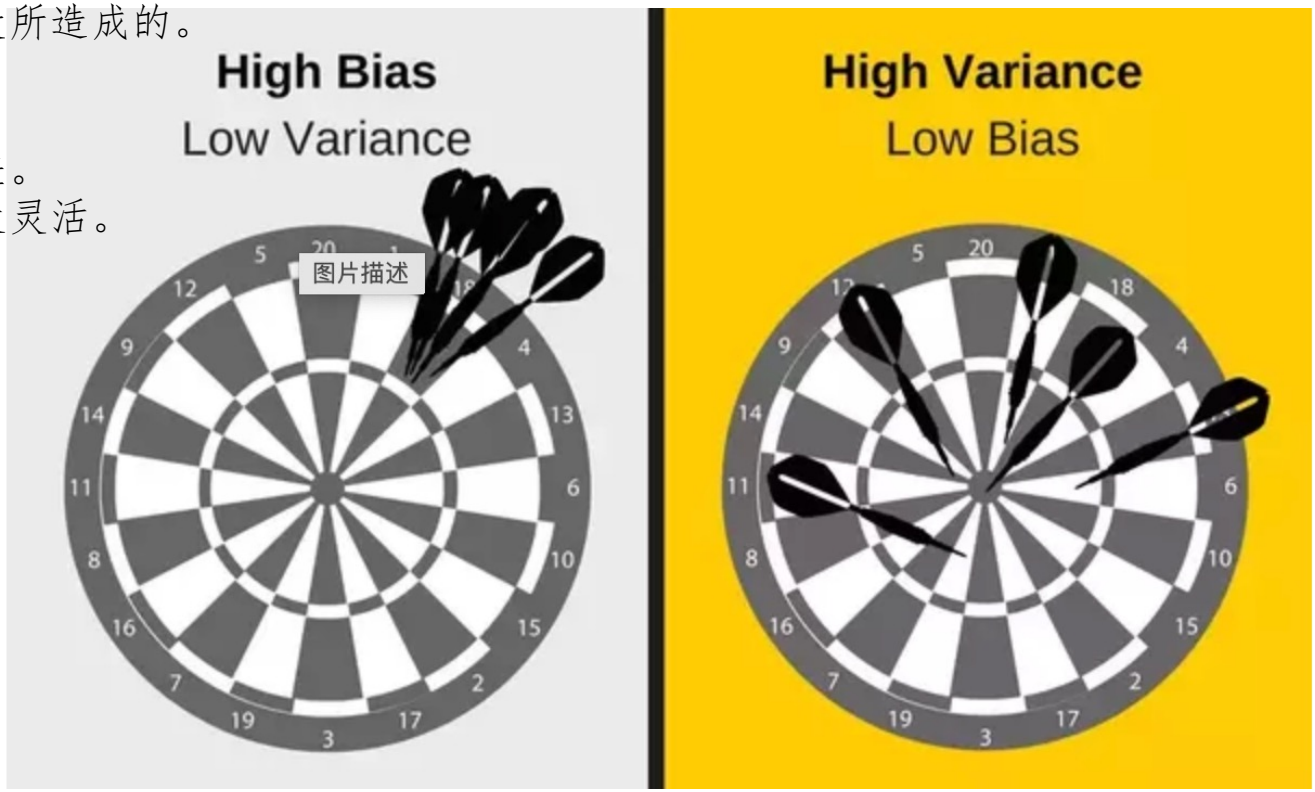
$$q_{\mathsf{ML}}(w_i) = \frac{\mathsf{Count}(w_i)}{\mathsf{Count}()}$$

偏差与方差之间的平衡

机器学习全部是关于给定输入数据（X）和给定输出数据（Y），然后去寻找一个最佳映射函数（F），这个映射函数通常也被叫做目标函数。

偏差误差（bias）：是由于简单的假设所造成的。
假设越简单，模型更加容易去训练。

一般而言，参数化算法具有较高的偏差。
学习速度非常快，容易理解，通常不太灵活。



低偏差：对目标函数提出更少的假设；

高偏差：对目标函数提出更多的假设；

低偏差模型例子：？

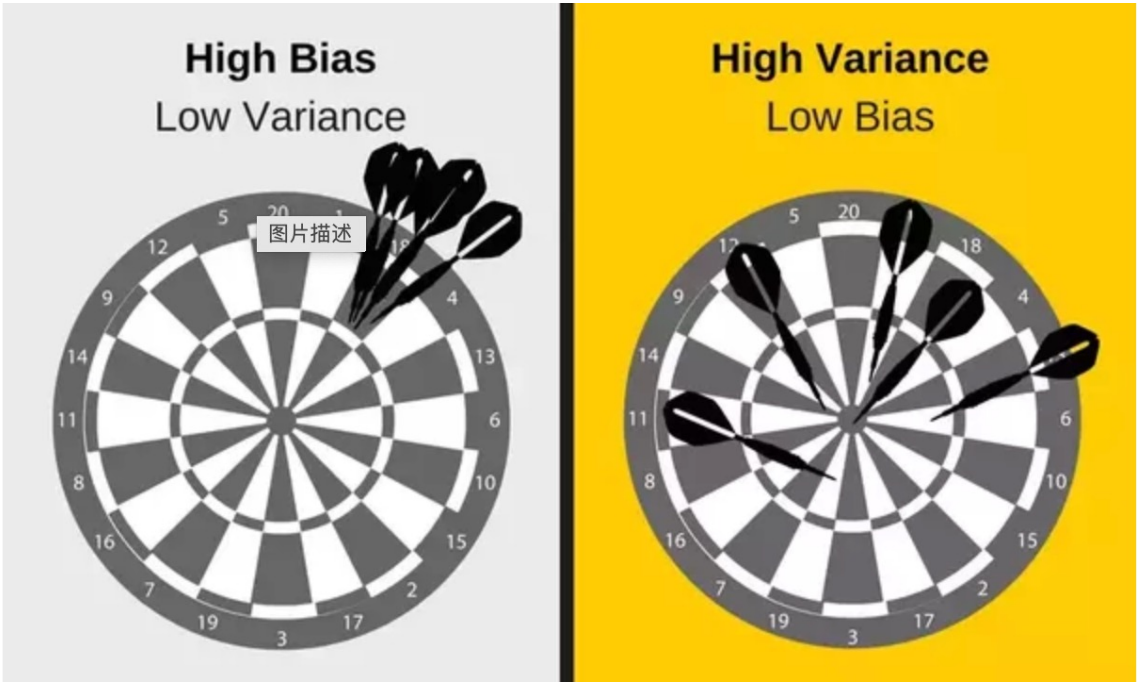
高偏差模型例子：？

低偏差和高偏差模型举例

- 低偏差模型例子：KNN 和 SVM；
- 高偏差模型例子：线性回归和逻辑斯特回归；

偏差与方差之间的平衡

- 使用不同的数据去训练同一个模型，我们得到的目标函数估计会改变。
- 目标函数是由机器学习的训练数据所估计得到的，所以我们期望训练数据拥有一定的方差variance。
- 理想情况下，不希望目标函数从一个训练数据集到另一个训练数据集有太大的变化，也就是说我们的算法需要很好的从训练数据中找到一些映射的特征关系，这样可以保证不同训练集都有一个差不多的目标函数。



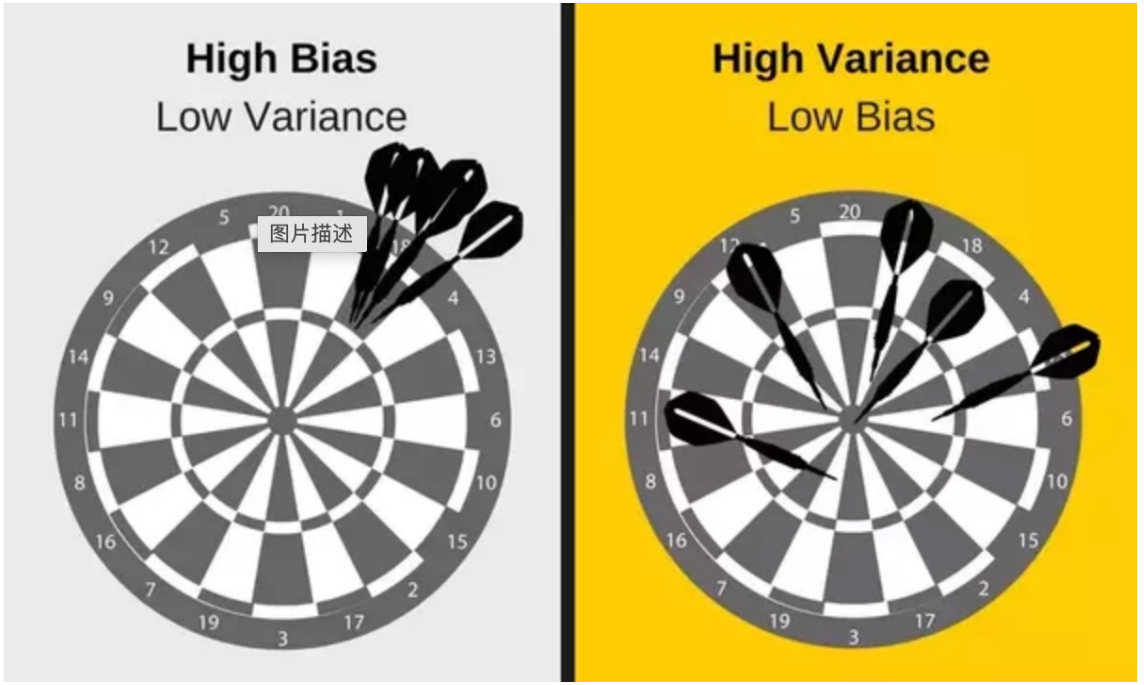
- 低方差：随着训练数据集变化，对目标函数估计值的变化非常小；
- 高方差：随着训练数据集变化，对目标函数估计值的变化非常大；
- 一般而言，具有很大灵活性的非参数学习算法都具有很高的方差。
- 高方差例子：？

高方差举例

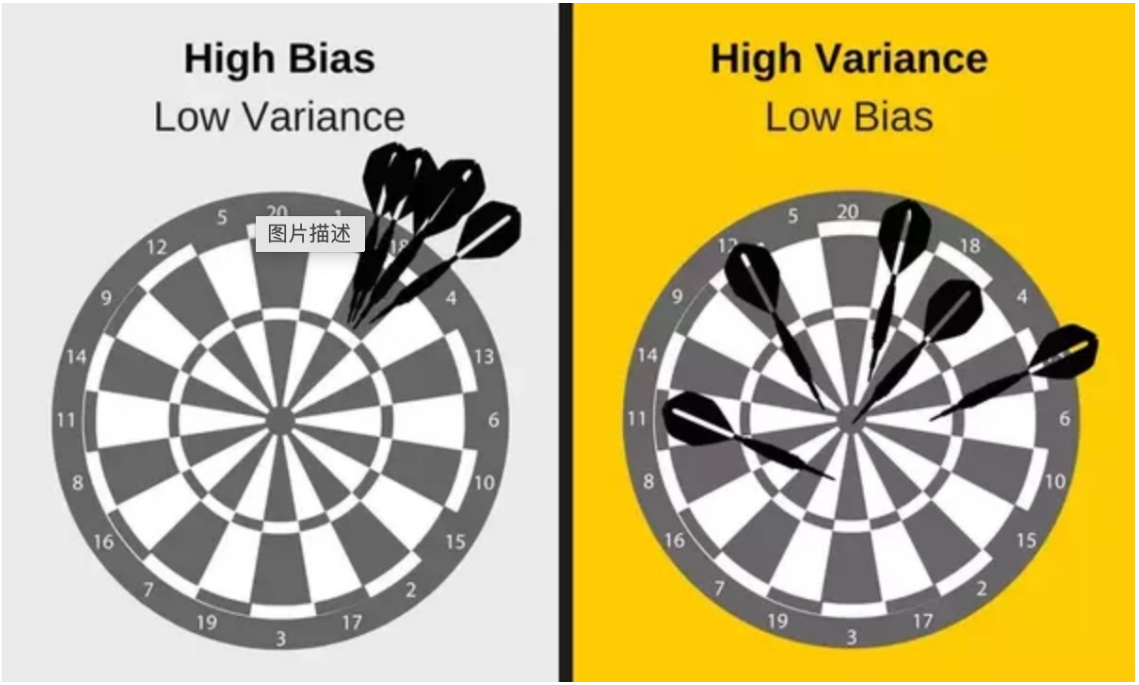
- 低方差：随着训练数据集变化，对目标函数估计值的变化非常小；
- 高方差：随着训练数据集变化，对目标函数估计值的变化非常大；
- 一般而言，具有很大灵活性的非参数学习算法都具有很高的方差。
- 高方差例子：KNN 和 SVM。

偏差与方差之间的平衡

- Why?



- 参数或者线性的机器学习算法一般会有一个很高的偏差和一个很低的方差。
- 非参数或者非线性的机器学习算法一般会有一个很低的偏差和一个很高的方差。
- 需要在这两者之间找到一个平衡点，来优化我们的算法



Q&A

- KNN 算法有_____偏差和_____方差，但是我们可以通过调整_____的值来改变偏差和方差之间的权衡关系，从而达到一个比较平衡的状态。

- KNN 算法有很低的偏差和很高的方差，但是我们可以通过调整 k 的值来改变偏差和方差之间的权衡关系，从而达到一个比较平衡的状态。

Linear Interpolation

- Take our estimate $q(w_i \mid w_{i-2}, w_{i-1})$ to be

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2 \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3 \times q_{\mathsf{ML}}(w_i) \end{aligned}$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i.

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\text{ML}}(w \mid u, v) + \lambda_2 \times q_{\text{ML}}(w \mid v) + \lambda_3 \times q_{\text{ML}}(w)]$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\text{ML}}(w \mid u, v) + \lambda_2 \times q_{\text{ML}}(w \mid v) + \lambda_3 \times q_{\text{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\text{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\text{ML}}(w \mid v) + \lambda_3 \sum_w q_{\text{ML}}(w)$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\text{ML}}(w \mid u, v) + \lambda_2 \times q_{\text{ML}}(w \mid v) + \lambda_3 \times q_{\text{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\text{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\text{ML}}(w \mid v) + \lambda_3 \sum_w q_{\text{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\text{ML}}(w \mid u, v) + \lambda_2 \times q_{\text{ML}}(w \mid v) + \lambda_3 \times q_{\text{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\text{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\text{ML}}(w \mid v) + \lambda_3 \sum_w q_{\text{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

$$= 1$$

Linear Interpolation (continued)

Our estimate correctly defines a distribution (define $\mathcal{V}' = \mathcal{V} \cup \{\text{STOP}\}$):

$$\sum_{w \in \mathcal{V}'} q(w \mid u, v)$$

$$= \sum_{w \in \mathcal{V}'} [\lambda_1 \times q_{\text{ML}}(w \mid u, v) + \lambda_2 \times q_{\text{ML}}(w \mid v) + \lambda_3 \times q_{\text{ML}}(w)]$$

$$= \lambda_1 \sum_w q_{\text{ML}}(w \mid u, v) + \lambda_2 \sum_w q_{\text{ML}}(w \mid v) + \lambda_3 \sum_w q_{\text{ML}}(w)$$

$$= \lambda_1 + \lambda_2 + \lambda_3$$

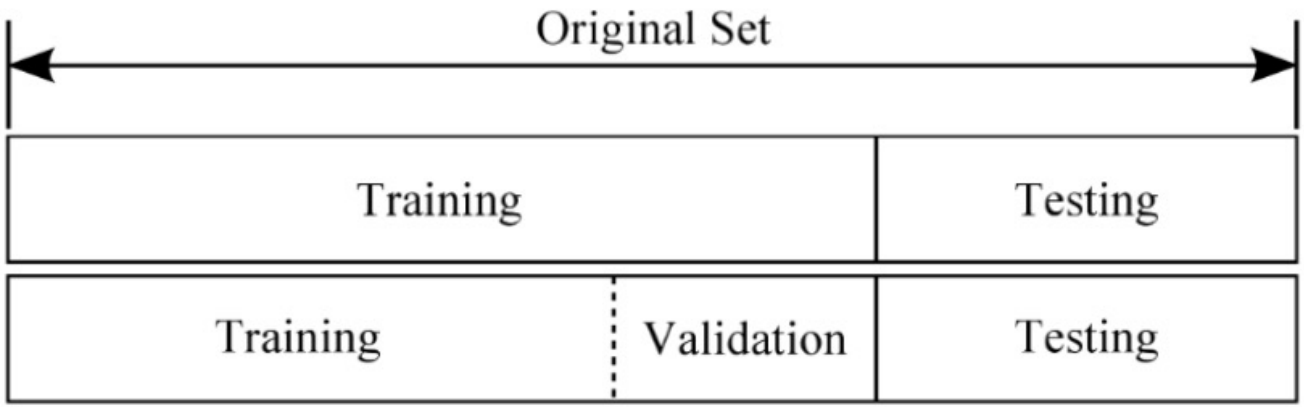
$$= 1$$

(Can show also that $q(w \mid u, v) \geq 0$ for all $w \in \mathcal{V}'$)

How to estimate the λ values?

- Hold out part of training set as "validation" data

训练集 验证集 测试集



- 训练集（train set）：用于训练模型以及确定参数。
  - 相当于老师教学生知识的过程。
- 验证集（validation set）：用于确定网络结构以及调整模型的超参数。
  - 相当于月考等小测验，用于学生对学习的查漏补缺。
- 测试集（test set）：用于检验模型的泛化能力。
  - 相当于大考，上战场一样，真正的去检验学生的学习效果。

参数（parameters）是指由模型通过学习得到的变量，如权重和偏置。

超参数（hyperparameters）是指根据经验进行设定的参数，如迭代次数，隐层的层数，每层神经元的个数，学习率等。

How to estimate the λ values?

- Hold out part of training set as "validation" data
- Define $c'(w_1, w_2, w_3)$ to be the number of times the trigram (w_1, w_2, w_3) is seen in validation set

How to estimate the λ values?

- Hold out part of training set as "validation" data
- Define $c'(w_1, w_2, w_3)$ to be the number of times the trigram (w_1, w_2, w_3) is seen in validation set
- Choose $\lambda_1, \lambda_2, \lambda_3$ to maximize:

$$L(\lambda_1, \lambda_2, \lambda_3) = \sum_{w_1, w_2, w_3} c'(w_1, w_2, w_3) \log q(w_3 \mid w_1, w_2)$$

such that $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i, and where

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = \quad & \lambda_1 \times q_{\mathsf{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2 \times q_{\mathsf{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3 \times q_{\mathsf{ML}}(w_i) \end{aligned}$$

Allowing the λ's to vary

- Take a function Π that partitions histories
  e.g.,

$$\Pi(w_{i-2}, w_{i-1}) = \begin{cases} 1 & \text{If Count}(w_{i-1}, w_{i-2}) = 0 \\ 2 & \text{If } 1 \leq \text{Count}(w_{i-1}, w_{i-2}) \leq 2 \\ 3 & \text{If } 3 \leq \text{Count}(w_{i-1}, w_{i-2}) \leq 5 \\ 4 & \text{Otherwise} \end{cases}$$

- Introduce a dependence of the λ's on the partition:

$$\begin{aligned} q(w_i \mid w_{i-2}, w_{i-1}) = & \quad \lambda_1^{\Pi(w_{i-2},w_{i-1})} \times q_{\text{ML}}(w_i \mid w_{i-2}, w_{i-1}) \\ & + \lambda_2^{\Pi(w_{i-2},w_{i-1})} \times q_{\text{ML}}(w_i \mid w_{i-1}) \\ & + \lambda_3^{\Pi(w_{i-2},w_{i-1})} \times q_{\text{ML}}(w_i) \end{aligned}$$

  where $\lambda_1^{\Pi(w_{i-2},w_{i-1})} + \lambda_2^{\Pi(w_{i-2},w_{i-1})} + \lambda_3^{\Pi(w_{i-2},w_{i-1})} = 1$,
  and $\lambda_i^{\Pi(w_{i-2},w_{i-1})} \geq 0$ for all i.

Overview

- The language modeling problem
- Trigram models
- Evaluating language models: perplexity
- Estimation techniques:
  - Linear interpolation
  - Discounting methods

Discounting Methods

- Say we've seen the following counts:

x	Count(x)	$q_{\text{ML}}(w_i \mid w_{i-1})$
the	48	
the, dog	15	15/48
the, woman	11	11/48
the, man	10	10/48
the, park	5	5/48
the, job	2	2/48
the, telescope	1	1/48
the, manual	1	1/48
the, afternoon	1	1/48
the, country	1	1/48
the, street	1	1/48

- The maximum-likelihood estimates are high (particularly for low count items)

Discounting Methods

- Now define "discounted" counts, $\text{Count}^*(x) = \text{Count}(x) - 0.5$
- New estimates:

x	$\text{Count}(x)$	$\text{Count}^*(x)$	$\frac{\text{Count}^*(x)}{\text{Count(the)}}$
the	48		
the, dog	15	14.5	14.5/48
the, woman	11	10.5	10.5/48
the, man	10	9.5	9.5/48
the, park	5	4.5	4.5/48
the, job	2	1.5	1.5/48
the, telescope	1	0.5	0.5/48
the, manual	1	0.5	0.5/48
the, afternoon	1	0.5	0.5/48
the, country	1	0.5	0.5/48
the, street	1	0.5	0.5/48

Discounting Methods (Continued)

- We now have some "missing probability mass":

$$\alpha(w_{i-1}) = 1 - \sum_{w} \frac{\text{Count}^*(w_{i-1}, w)}{\text{Count}(w_{i-1})}$$

e.g., in our example, $\alpha(the) = 10 \times 0.5/48 = 5/48$

Katz Back-Off Models (Bigrams)

- For a bigram model, define two sets

$$\begin{aligned} \mathcal{A}(w_{i-1}) &= \{w \ : \ \text{Count}(w_{i-1}, w) > 0\} \\ \mathcal{B}(w_{i-1}) &= \{w \ : \ \text{Count}(w_{i-1}, w) = 0\} \end{aligned}$$

- A bigram model

$$q_{BO}(w_i \mid w_{i-1}) = \begin{cases} \dfrac{\text{Count}^*(w_{i-1}, w_i)}{\text{Count}(w_{i-1})} & \text{If } w_i \in \mathcal{A}(w_{i-1}) \\ \alpha(w_{i-1}) \dfrac{q_{\text{ML}}(w_i)}{\sum_{w \in \mathcal{B}(w_{i-1})} q_{\text{ML}}(w)} & \text{If } w_i \in \mathcal{B}(w_{i-1}) \end{cases}$$

where

$$\alpha(w_{i-1}) = 1 - \sum_{w \in \mathcal{A}(w_{i-1})} \frac{\text{Count}^*(w_{i-1}, w)}{\text{Count}(w_{i-1})}$$

Katz Back-Off Models (Trigrams)

- For a trigram model, first define two sets

$$\mathcal{A}(w_{i-2}, w_{i-1}) = \{w : \text{Count}(w_{i-2}, w_{i-1}, w) > 0\}$$
$$\mathcal{B}(w_{i-2}, w_{i-1}) = \{w : \text{Count}(w_{i-2}, w_{i-1}, w) = 0\}$$

- A trigram model is defined in terms of the bigram model:

$$q_{BO}(w_i \mid w_{i-2}, w_{i-1}) = \begin{cases} \dfrac{\text{Count}^*(w_{i-2}, w_{i-1}, w_i)}{\text{Count}(w_{i-2}, w_{i-1})} & \text{If } w_i \in \mathcal{A}(w_{i-2}, w_{i-1}) \\ \dfrac{\alpha(w_{i-2}, w_{i-1}) q_{BO}(w_i \mid w_{i-1})}{\sum_{w \in \mathcal{B}(w_{i-2}, w_{i-1})} q_{BO}(w \mid w_{i-1})} & \text{If } w_i \in \mathcal{B}(w_{i-2}, w_{i-1}) \end{cases}$$

where

$$\alpha(w_{i-2}, w_{i-1}) = 1 - \sum_{w \in \mathcal{A}(w_{i-2}, w_{i-1})} \frac{\text{Count}^*(w_{i-2}, w_{i-1}, w)}{\text{Count}(w_{i-2}, w_{i-1})}$$

Summary

- Three steps in deriving the language model probabilities:
  1. Expand $p(w_1, w_2 \dots w_n)$ using Chain rule.
  2. Make Markov Independence Assumptions
     $p(w_i \mid w_1, w_2 \dots w_{i-2}, w_{i-1}) = p(w_i \mid w_{i-2}, w_{i-1})$
  3. Smooth the estimates using low order counts.

- Other methods used to improve language models:
  - "Topic" or "long-range" features.
  - Syntactic models.

It's generally hard to improve on trigram models though!!