

# Modelos de Lenguaje


Fernanda Sobrino

6/22/2021

# Introducción

# Aplicaciones



san f 

- san francisco weather
- san francisco
- san francisco giants
- san fernando valley
- san francisco state university
- san francisco hotels
- san francisco 49ers
- san fernando
- san fernando mission
- san francisco zip code

[Google Search](#) [I'm Feeling Lucky](#)

# Aplicaciones



# Aplicaciones

- ▶ autofill
- ▶ reconocimiento de voz
- ▶ traducciones
- ▶ POS (part of speech tagging)
- ▶ resumir textos
- ▶ son el primer paso para modelos de NLP más avanzados

# Qué es un modelo de lenguaje?

- ▶ una distribución de probabilidad sobre palabras o secuencias de palabras
- ▶ nos da la probabilidad de que cierta secuencia de palabras sea válida
- ▶ válida: se parece a como habla/escribe la gente

# Tipos de modelos de lenguaje

- ▶ probabilísticos: n-grams, Hidden Markov Models (HMM)
- ▶ redes neuronales: por lo general son mejores que los probabilísticos

# Modelos Probabilísticos



# Modelos Probabilísticos

- Objetivo: calcular la probabilidad de una oración o conjunto de palabras

$$p(\mathbf{W}) = p(w_1, w_2, \dots, w_n)$$

- Tarea relacionado: la probabilidad de la siguiente palabra

$$p(w_5 | w_1, w_2, w_3, w_4)$$

- Los modelos que calculan cualquiera de estas dos son modelos de lenguaje (ML)

## Cómo calculamos $p(\mathbf{W})$ ?

- ▶ recordemos que:  $p(B|A) = \frac{p(A,B)}{p(A)}$
- ▶  $p(A, B) = p(A)p(B|A)$
- ▶ si tengo tres eventos de probabilidad  
 $P(A, B, C) = P(A, B)P(C|A, B) = p(A)p(B|A)p(C|A, B)$
- ▶ en general

$$p(x_1, \dots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\dots p(x_n|x_1, \dots, x_{n-1})$$

## Cómo calculamos  $p(\mathbf{W})$ ?

$$P(w_1, w_2, \dots, w_n) = \prod_i p(w_i|w_1, \dots, w_{i-1})$$

$$p(\text{"hola como estas hoy"}) = p(hola)p(como|hola)p(estas|hola como)p(hoy|hola como estas)$$

# Cómo estimamos estas probabilidades?

- Podríamos solo contar y dividir?

$$p(\text{hoy}|\text{hola como estas}) = \frac{\text{Count}(\text{hola como estas hoy})}{\text{Count}(\text{hola como estas})}$$

## Podríamos solo contar y dividir?

- No! Hay demasiados grupos de palabras posibles
- El lenguaje es creativo, hay nuevas oraciones todo el tiempo
- Incluso oraciones sencillas usando un corpus muy grande van a tener  $\text{count} = 0$
- nunca tendríamos suficientes datos para entrenar este modelo

# Supuesto de Markov

- Podríamos asumir que

$$p(\text{hoy}|\text{hola como estas}) \approx p(\text{hoy}|\text{estas})$$

- \* O algo como:

$$p(\text{hoy}|\text{hola como estas}) \approx p(\text{hoy}|\text{como estas})$$

## Supuesto de Markov

$$p(w_1 w_2 \dots w_n) \approx \prod_i p(w_i | w_{i-k} \dots w_{i-1})$$

- Es decir aproximamos cada una de las partes de la oración por su producto

$$p(w_i | w_1 w_2 \dots w_n) \approx p(w_i | w_{i-k} \dots w_{i-1})$$

## Uni-gram model (caso más sencillo)

$$p(w_1 w_2 \dots w_n) \approx \prod_i p(w_i)$$

## Bi-gram model \* condicionamos en la palabra anterior

$$p(w_i | w_1 w_2 \dots w_n) \approx p(w_i | w_{i-1})$$

# N-gram model

- ▶ podemos extender este modelo a 3, 4, 5
- ▶ en general es un modelo insuficiente de language
  - ▶ el lenguaje tiene dependencias largas
  - ▶ “La cadena argentina TyC Sports publicó una foto de la increíble obra de arte en su página de Instagram, donde Messi la vio y comentó que le gustaría firmarla.”
- ▶ pero en muchas ocasiones es buena idea usar un n-gram model

## Estimación

# Bi-gram model

- ▶ usamos MLE

$$p(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

$$p(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$



## Ejemplo

Supongamos que nuestro corpus es de 3 oraciones \* <s> Yo soy Sam <s> \* <s> Sam yo soy <s> \* <s> no me gustan los huevos verdes con jamón <s>

Entonces

$$p(yo | <s>) = 1/3 \quad p(Sam | <s>) = 1/3 \quad p(soy | yo) = 2/2 = 1$$

$$p(<s> | Sam) = 1/2 \quad p(Sam | soy) = 1/2$$

## Ejemplo 2

Oraciones del proyecto de restaurantes de Berkeley (el proyecto tiene 9222 oraciones). Ejemplos: \* can you tell me about any good cantonese restaurants close by \* mid priced tai food is what i'm looking for \* tell me about chez panisse \* can you give me a listing of the kinds of food that are available \* i'm looking for a good place to eat breakfast \* when is caffe venezia open during the day

# Bi-grams

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

# Bi-grams

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

# Bi-grams

- podemos estimar la probabilidad de oraciones

$$p(< s > | I \text{ want english food } < s >) =$$

$$p(I | < s >) p(want | I) p(english | want) p(food | english) p(< s > | food) = 0.0$$

## En la práctica \* Usar log space \* nos ayuda a lidiar con ps muy pequeñas \* es más fácil sumar que multiplicar

## Evalución del modelo

# Cómo sabemos que tan bien lo hace el modelo?

- ▶ Prefiere buenas oraciones o malas?
  - ▶ asigna probabilidades altas a oraciones “frecuentes” o “reales”
  - ▶ asigna poca probabilidad a oraciones “infrecuentes” o “mal escritas”
- ▶ Entrenamos los parámetros utilizando un training set
- ▶ Probamos el modelo en el test set
- ▶ Escogemos alguna métrica de evaluación

# Evaluación entre dos n-gram models

- ▶ 'Externa'
  - ▶ corres dos modelos para el mismo problema
  - ▶ obtienes la accuracy para ambos
  - ▶ comparamos estas medidas
  - ▶ **Problema**: esto lleva demasiado tiempo



# Evaluación entre dos n-gram models

- ▶ 'Interna'
  - ▶ perplejidad
  - ▶ es una mala aproximación
  - ▶ solo es buena si el conjunto de entrenamiento es muy parecido al de evaluación
  - ▶ es bueno tenerla presente

# Perplejidad

- ▶ el mejor modelo de lenguaje es aquel que mejor predice un conjunto de evaluación nunca antes visto
- ▶ nos da la mayor  $p(\text{sentence})$
- ▶ la perplejidad es 1 entre la probabilidad del conjunto de aprendizaje normalizada por el número de palabras
- ▶ minimizar la perplejidad es lo mismo que maximizar la probabilidad

# Perplejidad

$$\begin{aligned} PP(\mathbf{W}) &= p(w_1 \dots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{p(w_1 \dots w_N)}} = \sqrt[N]{\prod_i^N \frac{1}{p(w_1 \dots w_N)}} \end{aligned}$$

Si tenemos bi-grams

$$PP(\mathbf{W}) = \sqrt[N]{\prod_i^N \frac{1}{p(w_i | w_{i-1})}}$$

## Perplejidad Intuición \* podemos pensar en ella como un promedio ponderado del factor de ramificación \* el factor de ramificación de un language es la cantidad de palabras que pueden seguir a una palabra

## Perplejidad Intuición: ejemplo

- ▶ Supongamos que tenemos una oración que consiste de dígitos al azar
- ▶ Cada dígito ocurre en el conjunto de entrenamiento  $\frac{1}{10}$

$$PP(\mathbf{W}) = p(w_1 \dots w_N)^{-\frac{1}{N}} = \left( \left( \frac{1}{N} \right)^N \right)^{-1/N}$$

$$PP(\mathbf{W}) = \left( \frac{1}{10} \right)^{-1} = 10$$

## Problemas con los modelos n-gram \* entre mas grande la n mejor pero esto implica más operaciones previas  $\implies$  necesitas mas RAM \* n-grams son sparse la probabilidad de muchas palabras ocurriendo juntas va a ser cero  $\implies$  problemas computacionales

# Modelos de Aprendizaje profundo

