

Clasificación de textos

Opción 1 → uso de reglas y expresiones regulares → en casos muy particulares

Opción 2 → Supervised machine learning

SML:

Input → $\{(d_1, c_1), \dots, (d_n, c_n)\}$

② Nóive Bayes → planteamiento

- Input → $\{(d_1, c_1), \dots, (d_n, c_n)\}$

- Dado un nuevo $(d, ?)$ estima $P(\text{pos}|d) \times P(\text{neg}|d)$

- Selecciona la clase de mayor probabilidad

(Maximum a posteriori)

$$c_{MAP} = \arg \max_{C \in \{\text{pos, neg}\}} P(C|d) \quad \times \text{Bayes}$$

$$= \arg \max_{C \in \{\text{pos, neg}\}} \frac{P(d|C) P(C)}{P(d)} \rightarrow \text{no depende de } C$$

$$c_{MAP} = \arg \max_{C \in \{\text{pos, neg}\}} \frac{P(d|C) P(C)}{\#C} \quad \text{se estima } \frac{\#C}{\#D}$$

$$= \arg \max_{C \in \{\text{pos, neg}\}} \underbrace{P(x_1, x_2, \dots, x_n | C)}_{\text{difícil de estimar}} \hat{P}(C)$$

→ se aproxima
key words

① Multinomial Naive Bayes

- Bag of words: posición de las palabras no importa
- Independencia condicional: $P(X_i | c)$ son independientes

$$\Rightarrow P(X_1, \dots, X_n | c) = P(X_1 | c) P(X_2 | c) \dots P(X_n | c)$$

$$\therefore C_{MNB} = \underset{c \in C}{\arg \max} P(c) \prod_i P(X_i | c)$$

$$\hat{P}(c) = \frac{N_c}{N_{\text{doc}}} \quad \hat{P}(w_i | c) = \underbrace{\text{count}(w_i, c)}_{\text{una palabra}} \quad \frac{\text{count}(w_i, c)}{\sum_{w \in V} \text{count}(w, c)}$$

② Para evitar dividir por 0 se usa un smoothing

La place smoothing ~~(add-1)~~
 $(\text{add}-\alpha) \rightarrow \text{generalización}$

$$\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + \alpha}{\left[\sum_{w \in V} \text{count}(w, c) \right] + \alpha |V|}$$

③ Para evitar problemas numéricos

$P(c) \prod_i P(X_i | c) \rightarrow$ producto de números chicos da un número muy chico

se usa el log que es creciente \Rightarrow extremismo max

$$\therefore C_{MNB} = \underset{c \in C}{\arg \max} \log(P(c)) + \sum_i \log(P(X_i | c))$$

④ Tips para mejorar la clasificación

- Normalización ~~o better normalization~~
 - tokenizar
 - lematizar o stemming
- Aumentar el peso de ciertos tokens
 - tokens de título
 - primera oración de C/parrafo
 - en oraciones q' contienen palabras del título
- Descartar palabras
 - muy poco frecuentes
 - muy frecuentes
 - stop words
- Probar mejoras → en sentiment analysis
- Extraer n-gramos

⑤

Niveles Básicos

↑ no over-fitca, high-bias

↑ pocos parámetros o ajustar

↑ intuitivo

↑ rápido

↓ no funcionan bien

↓ otros métodos sejan más eficaces

Es un punto de inicio para comenzar a probar

④ Evaluación de los métodos

$$\left\{ \begin{array}{l} \text{Precision} \quad \frac{TP}{TP+FP} \\ \text{Recall} \quad \frac{TP}{TP+FN} \end{array} \right.$$

{ F-measure (f1-score) }

$$F = \frac{2}{\frac{1}{Pr} + \frac{1}{Rc}} = \frac{2 Pr R_c}{Pr + R_c}$$

{ Cross Validation

(5-fold CV)

me quedo con el ~~avg~~ (f-score) de cross

{ Cross Validation + test set

me quedo con el f-score sobre el test set

⑤ Regresión logística \rightarrow discriminativos

- Se computa directamente $P(Y|D)$ sin tener que calcular el likelihood
- Puede usar features poco importantes para la generación

- correctrizadores que permiten discriminar

- Componer

+ Representación de Features

$$x_i^{(i)} = [x_1^{(i)}, \dots, x_n^{(i)}] \rightarrow \text{soñando o una clase } y$$

+ Función de clasificación que compute \hat{y} estimado

$$f(x^{(i)}) = f([x_1^{(i)}, \dots, x_n^{(i)}]) = \hat{y}$$

+ Función objetivo para el aprendizaje

+ Algoritmo para optimizar la función objetivo.

Como entrenar el algoritmo

{ función objetivo \rightarrow costos de errores

\rightarrow encontrar los pesos que generen buenas predicciones

\rightarrow calcular el error J (clásico)

\rightarrow modificar los pesos para disminuir el error

\rightarrow se usa Cross-Entropy: dif entre el valor real y y el predicho \hat{y}

Como minimizar la pérdida (costo)

{ algoritmo de optimización

\rightarrow modificar pesos y bias

\rightarrow derivar la func de costo J con respecto a la dirección en la que hace el algoritmo la pérdida

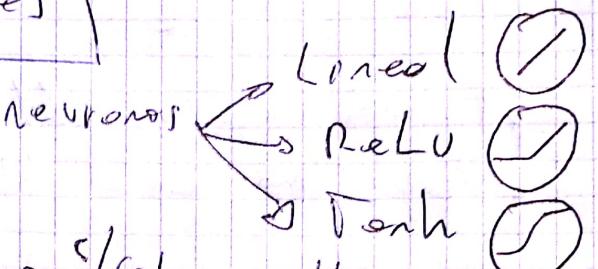
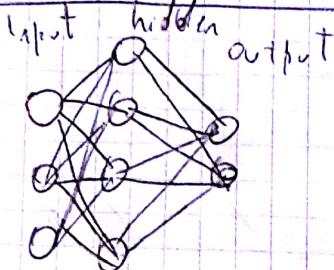
Cross-entropy con la func sigmoidal es convexa y derivable

Se obtiene una discriminación lineal

⑥ Regularizaciones

- + Pesos de los features muy grandes \Rightarrow es probable que se esté overfitando
- + La regularización penaliza los pesos altos - se hace de calcular el costo

⑦ Redes Neuronales \rightarrow combinación de logísticos



c/ capa oculta
es como una regresión logística

Tienen activación sigmoidal

usa backpropagation para entrenamiento \rightarrow learning rate para optimizar pesos
se usa Stochastic Gradient Descent
 \rightarrow muy costoso \Rightarrow se realiza

entrenamiento de mini batches
calcula costo medio

se actualiza sobre gradiente del minibatch

\rightarrow como no es suave pero más eficiente

⑧ Pre Procesamiento

Term-document matrix \rightarrow matriz que contiene # filas = # de documentos

columnas = # de palabras f

Categoría tiene la cantidad de apariciones de esa palabra en el documento

Se normaliza por # de palabras en el doc

Ej: 2 documentos:

$d_1 \rightarrow \{1, 1, 2, 1, 0, 0, 0, 0, 0, 0\}$
 $d_2 \rightarrow \{0, 1, 0, 0, 1, 1, 2, 1, 1, 1\}$

la Normalización $L_1 \rightarrow d = \frac{d}{\|d\|_1} = \sqrt{\sum d_i^2}$

Term Frequency weight \rightarrow resaltar los
palabras más significativas

$$tf-idf(t, d) = tf(t, d) \cdot idf(t)$$

$$idf(t) = \log \left(\frac{|D| + 1}{\frac{|id:t|}{|D|}} \right) + 1$$

documentos totales en el set de train
documentos que aparece t

Se suele realizar smoothing

1º) tf-idf

2º) normalizar

~~3º) medir~~

NLP - Close 5

HOJA N.

FECHA

Similaridad Semántica

La similaridad semántica puede deducirse de su contexto
similaridad entre palabras

④ Word embeddings

→ calcular la cercanía entre palabras

→ usa tf-idf

→ calcula la similaridad coseno entre las palabras → no tiene en cuenta el largo de los ~~vectores~~ vectores (palabras que aparecen más)

⑤ Información retrieval

Query: "algún string" → similaridad entre documentos

Se extrae el TF-IDF del dataset

Se obtienen los pesos entrenados del query

⑥ Term-context matrix

term-document matrix

coocurrencia de primer orden

asociaciones sintácticas

sintácticos (observables) (se observan)

term-context matrix

coocurrencia de 2 orden

asociaciones

porciones (no observables) (se observan)

⑦ Pointwise Mutual Information (PMI)

$$\text{Defn PMI}(w, c) = \log \left(\frac{P(w, c)}{P(w)P(c)} \right)$$

Si $\text{PMI} < 0$ es que w o c tienden a ocurrir individualmente. Muestran co-ocurrencias no informativas

$\text{PMI} < 0 \rightarrow$ no hay asociación \rightarrow son independientes

NOTA

④ ~~PPMI~~

④ Positive Pointwise Mutual Information (PPMI)

$$\text{PPMI}(w, c) = \max(0, \log\left(\frac{P(w, c)}{P(w)P(c)}\right))$$

2 palabras totalmente correlacionadas $P(w, c) = P(w)P(c)$

$$\begin{aligned} P(w) = P(c) \Rightarrow PMI(w, c) &= \log\left(\frac{P(w, c)}{P(w)P(c)}\right) = \log\left(\frac{1}{P(w, c)}\right) \\ &= -\log(P(w, c)) \end{aligned}$$

\Rightarrow ~~P(w, c)~~ más alta dan PPMI más alta

Solución →

→ Filtrar $fij \in K$

→ Formato $P(c)$ $P_\alpha(c) = \frac{\text{count}(c)^\alpha}{\sum_c \text{count}_j(c)^\alpha}$

→ add -~~α~~ smoothing

$$\rightarrow \text{Normalized PMI} \rightarrow NPMI(w, c) = \frac{\log\left(\frac{P(w, c)}{P(w)P(c)}\right)}{-\log(P(w, c))}$$

⑤ Applications → Collocations (expresiones)

~~Tipos de collocations~~ → tokenización (norton + serro, boy + toy)
~~Collocations~~ → Identificar collocations

→ lista

→ buscar bigramas ~~de alto~~
en el corpus con alto PPMI

NLP - clase 6

HOJA N.

FECHA

Word-embeddings

• Heredado de SVD → sobre term document context matrix

~~Perdida de~~ LSA se utiliza la transformación SVD sobre la TF-IDF. Antes de utilizar SVD se puede aplicar una transformación sobre TF-IDF llamada Log-Entropy

-Tips: + no estándar de dimensión ($K=300$)

$X = \sum_{i=1}^n \begin{matrix} \text{row} \\ \text{matrix} \end{matrix} \begin{matrix} \text{column} \\ \text{matrix} \end{matrix} + K \text{ óptimo depende de las tareas}$
+ veces conviene tirar hasta los primeros 50 dimensiones

similitud cosine
similaridad entre palabras
ver la similitud
 $X^T = \begin{matrix} \text{row} \\ \text{matrix} \end{matrix} \begin{matrix} \text{row} \\ \text{matrix} \end{matrix} \begin{matrix} \text{row} \\ \text{matrix} \end{matrix}$
+ similaridad cosine no es absoluta,
mayor $K \Rightarrow$ palabras más distantes)

Word2Vec (skip-gram)

/ Factorización

→ Bow (bag of words)

Factoriza 3 red neuronas
top PMI-C.V. phot input x palabra
negative sampling para fa objetivo

• No existe una técnica de word-embedding que sea mejor en todos los tareas

target embedding
context embedding
No existe tarea en particular, se debe elegir el embedding que mejor se adapte
Existen embeddings ya preentrenados

② Embeddings

pose pueden usarse como features

- para investigar un corpus
- métrica de similaridad semántica externa
- no son comparables embeddings de distintos corpora

Modelos de lenguaje

⑥ Aplicaciones

→ traducción

→ reconocimiento del habla

→ corrección en la escritura

→ sistema de ayuda

$$p(w|h) = \text{probabilidad de la palabra } w \text{ dada su historia } h$$

⑦ N-gramas

$$p(w|h) = \#(h, w) / \#(h)$$

cuantas veces aparece una frase en el lenguaje

$$p(w_1^n) = \prod_{k=1}^n p(w_k | w_1^{k-1})$$

Problema: (h, w) es probable que no exista

Ventaja: buscar n más chico si permite

calcular las probabilidades y así

$$\Rightarrow p(w_1^n) = \prod_{k=1}^n p(w_k | w_{n-k+1}^{n-1})$$

usar N chicos en lugar de todos lo close

↑ aparece el n-grama

↑ aparece el (n-1)-grama

④ Evaluación

- + Extrínsecos: usar resultados de nuestro algoritmo en alguna aplicación externa
- + Entrenar modelos para despacharlos en otra aplicación es costoso
- + Intrínsecos: evaluar en un modelo de test
 - operaciones de test no están en forma set
 - se usa perplexity

⑤ Perplexity

- + Inversa de la probabilidad en un set de testeo normalizada por la cantidad de palabras
- + Solo es comparable si dos modelos fueron entrenados con el mismo corpus.
 - ↳ debajo = los cunkas

⑥ Metodos implementacion

- by prob.
- n-gram x Fm operación
- unids genérico
- unids selector
- tratamiento de palabras no vistas en train
 - ↳ n-gram cunk de baja freq
 - ↳ todos los cunk tratamiento igual
- tratamiento de prob = 0 con smoothing
- n-gram no puede generalizar palabras similares

↳ Hockoff and interpolation
Caso no hay prob n-gram, probar 1-1, 1-2, ..., 1-primer

① LM con redes

Neural LM

- ↳ mejor embeddings para entornos → en lugar de los palabros
- ↳ tienen codificado los relaciones semánticas
- ↳ Long option calculadas para optimizar la tarea para la cual creamos la red (predicción)
- ↳ sirve para una copia para calcular el unpadding
- ↳ back propagation incluye los embeddings para que se calcule teniendo en cuenta la tarea

② USOS

- + Predictibilidad: prob de conocer un palabro antes de leerlo, basada en su contexto precedente
- ↳ ↓ medir es costoso
- ↳ ↓ se mide para cada corpus separado
- ↳ objetivos ↳ desarrollar rectificadores computacional de una medida costosa
 - ↳ tener mejor comprensión de las formas utilizadas para predecir
- ↳ Metodología [LSA, Word2Vec, N-gram] } UNIFICAR }
- ↳ Entrenamiento

Topic Models

- Descubrir temáticas ocultas
- Anotar documentos
- Utilizar las anotaciones para visualizar, organizar, resumir

④ **Topico**: distribución probabilística sobre un vocabulario. Aprobación de prob a cada palabra

Método LDA → idea de LSA

+ Modelos probabilísticos basados en los ojos de un lector generativo:

↳ 1) doc vacío

2) asignamos una prob de los temas

3) generamos palabras: 1) elegimos tema al azar

2) elegimos palabra del tema según sus prob

3) repetir n veces

→ 2 regula los temas

n regula palabras x tema

4) repetir n veces

+ → Inferir los parámetros de un modelo probabilístico

⑤ Uso → visualizar palabras más probables x tema
y probabilidad de temas en el documento

① Evaluación con métodos

+ cognitivo → global normal

+ intrínsecos → topic coherence → PMI entre todos los palabras

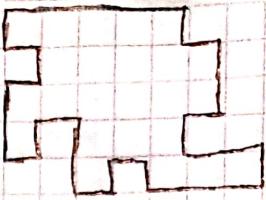
+ errores in human judgments → holismo intrínseco → topic intrusos

+ extrínsecos

Hay topicos preestablecidos

(o de acuerdo a la problemática local)

que sea útil utilizarlo



Sentiment Analysis

④ Sentiment lexicons

Como se forma:

- Anotaciones humanas
- De manera supervisada.

Tienen dimensiones

- (Más) → valencia (pos/neg)
- arousal (intensidad)
- dominancia
- objectividad
- relativa concordancia

- 1) seeds de sentimientos
- 2) definir métrica de cercanía
- 3) el polohra calcula su sentimiento

⑤ Semantic orientation → generar lexicones

$$\text{polaridad}(w) = \sum_{w_p \in \text{pos-words}} PMI(w, w_p) - \sum_{w_n \in \text{neg-words}} PMI(w, w_n)$$

Turney usa bigramos

La polaridad de un review es la polaridad promedio de sus bigramos

⑥ Preprocesamiento

- tokenización casual
- proposición de negaciones
- stopwords
- ortopíefice
- stopwords
- normalización de fechas, números, entódeos

① M étodos totalmente supervisados

Medidas de personalidad (Big 5)

por reprocessamiento:

- 1) hebbi y funtokenitz
- 2) bigramos y trigramos con m2ulca
- 3) distribución de tokens de los sujetos
- 4) transformación de ascensoe

Represión lineal

para cada test fito

$$x = w_0 + \sum_{i \in [1, N]} w_i p(\text{phrase}_i)$$

↓
valor
obtenido
del test

$p(\text{phrase}_i)$ → peso de la
aparición en el test

distribución de n-gramas

Luego con los w_i calculo los Big 5 de
personalidad de un nuevo documento

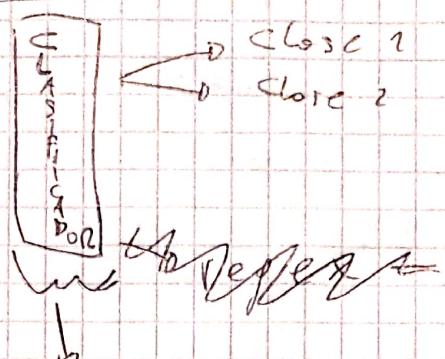
Vídeo 2 - 1

HOJA N°

FECHA

Clasificación de textos

Documento →



- SML

- RegEx → muchas veces sirve mal
de lo esperado
y es lento

SML → Naïve Bayes → buen punto de partida
para cualquier método simple

→ Multinomial Naïve Bayes → bag of words (posición de los palabras)
independencia condicional

→ Laplace smoothing ($\text{total} - \alpha$) $\alpha = 1$

→ se usa b_j para comparar $y \rightarrow$ que
evite errores y problemas numéricos

general
pero
depende
de coro
y método

→ TPS: normalización → reexp → tokens
lematizar o stemming

remover pero de tokens

descartar palabras / Stop words

proporcionales → sentiment
n-frames

Video 4-1

HOJA N°

FECHA

Clasificación
Regresión → generativo

Regressão logística → discriminativo encontra
características que servem
para diferenciar classes
→ separar em features

$$\text{features } [x_1, \dots, x_n] = c_1$$

[texto] → extroípo features

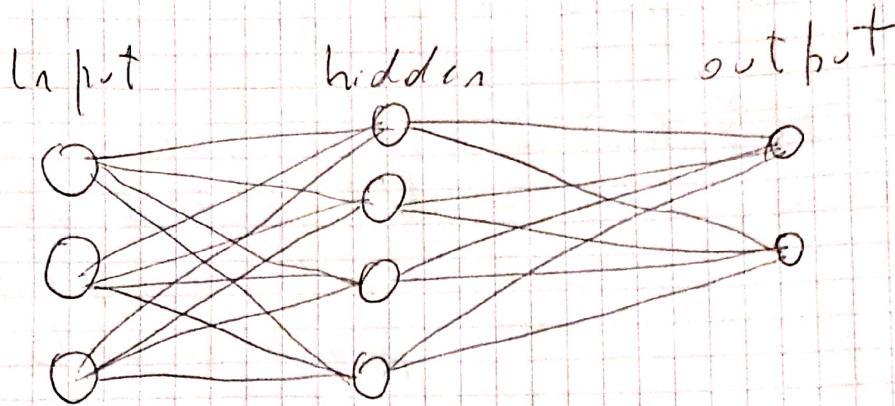
Atributo função

$$f(\text{Pos}) + f(\text{Neg})$$

Se busca la combinación de pesos y bias que
optimice la clasificación. Se usa la objetivo
como por ejemplo cross-entropy

Regularización → penaliza pesos altos → la hora de
calcular costos

Redes Neuronales → sumarizar los lógicos



Backpropagation (derivada en lo lógico)

Muy caro
computacionalmente

→ buscar los valores óptimos de los pesos y bias
→ tomar los errores y los propagar hacia atrás
para encontrar los óptimos
se usa Stochastic Gradient Descent. ~~para~~ → Entrenando
de a mini batches

Preprocesamiento → term-document matrix

(normalizar)

→ Term-Frequency weight → NO NORM.
ALITA

importancia
de las palabras
significativas

tf-idf
frequency
smooth

distintivo en el
documento

Video 5-1

Similaridad semántica

Word embeddings → vectorialmente los polímeros
→ distancias entre ellos

Vector Space Model → Term document matrix

+ first order occurrence

+ $S \times N$ legumbres

occurciones

+ mostaza - homologos

matriz muy esparsa

+ f-idf → No Normaliza

→ Similaridad coseno entre
vectores → son parecidos

son los polímeros

similaridad coseno entre
documentos porque
que tienen parecidos
se ve en IR

#polímeros x #polímeros

Term context matrix

→ simétrico

asimétricas

+ second order occurrence

→ 1 matrix for corpus

+ paradigmáticas ocurrencias

+ mostaza - ket chup

PMI → para medir similaridad semántica

2 palabras juntas en la misma ventana

$$PMI(w, c) = \log \frac{P(w, c)}{P(w)P(c)}$$

juntas

o - lejos

$\text{PMI} = 0 \rightarrow$ no hay asociación

$$\text{PMI}(w, c) = \text{PMI}(c, w)$$

Similitud semántica \leftrightarrow coocurrencia

$$\text{PPMI} \geq 0 = \max(0, \text{PMI})$$

↓
dependiente del corpus

Palabras muy correlacionadas dan PPMI altos

→ $P(w, c)$ cercas don PMI altos

↳ Soluciones: filtrar $f_{ij} < K$ (foco frecuentes)

Aumentar $P(c)$

(K-smoothing)

Normalized PMI

Collocations → buscar bigramas en corpus con

alto PPMI

ventana solo a derecha de Número 2

Video 6-1

HOJA N°

FECHA

Algoritmo

Word embeddings

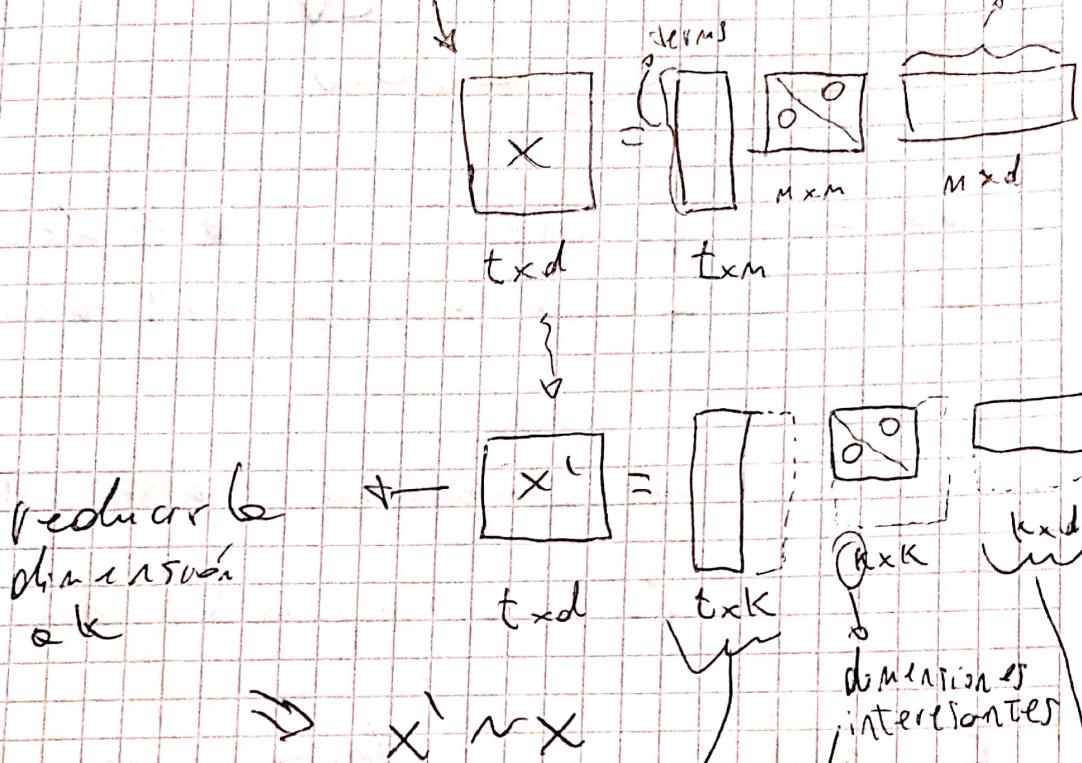
LSA → term document matrix

SVD → factorización

dimensionalidad

reducida

Objetivo → representar los palabras
en una dimensionalidad
reducida



reducir la
dimensionalidad
a k

$\Rightarrow X' \approx X$

topic

captura la
cantidad de tópicos
más comunes

embeddings in context

dimensiones interiores

antes de SVD → Tf-idf o log entropy

Alternativas

LSA sobre term document matrix

$k = 300$ seg si

Tipos

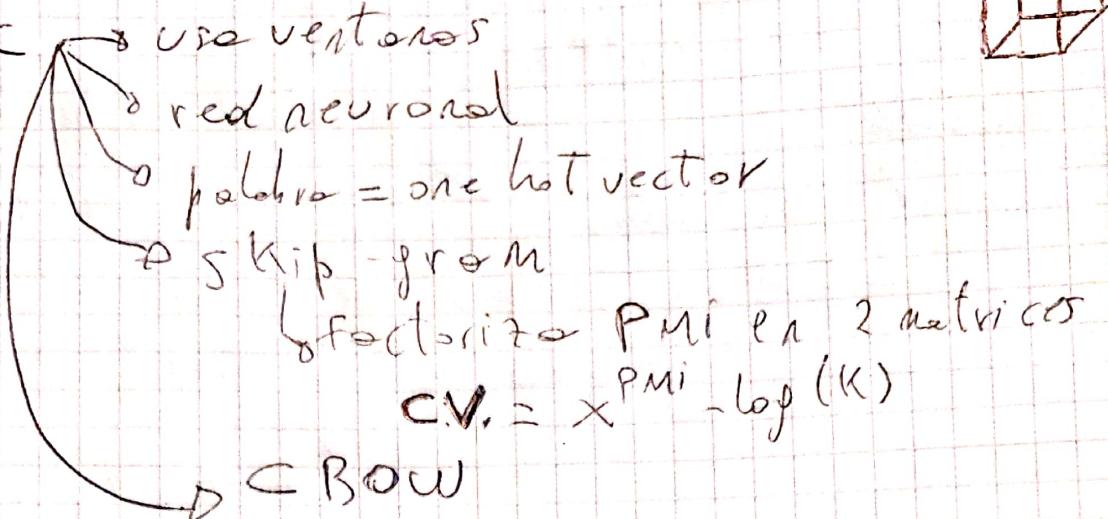
k óptimo depende de la tarea

→ tirar hasta que no sea dim. más alta que el centro de masa

→ similitud entre palabras no es absoluta

$> k \Rightarrow$ palabras diferentes \Rightarrow $<$ similitud

Word2Vec



Evaluación:
+ Pseffel test → % de aciertos
+ word similarity 333 → correlación (spearman)

+ categorización → silhouette coefficient
∴ no hay ninguna técnica óptima

Embeddings preentrenados → sirven en algunos contextos

No se pueden comparar embeddings de distintos corpus

Topic Models

↳ LDA

- descubrir
- motor
- visualizar

Tópicos → distribución probabilística sobre el vocabulario
 → asignación de palabras a tópico

→ modelo generativo y probabilístico

λ = cuentas tópicos se prenden x documento

η = cuentas palabras se prenden x tópico

Evaluación
métodos

- objetivo
- intrínseco → topic coherence → P_{Mi}
- heredos en human judgments
- extrínsecos → usando separado

Video 8-1

HOJA N°

FECHA

Resumen

Modelos de lenguaje

Traducción

→ reconocimiento del habla

→ corrección de la escritura

→ sistema de ayuda

N-grams → prob de un palabra dada su historio

solo dada su historio

→ usa ventanas chicas de N en lugar de usar toda frase

→ # aparece el n-grama

→ # aparece el (N-1)-grama

Evaluación → extrínseca → mejor resultado

en una tarea = 100%

diferente x

testcor

intrínseca

→ usar test-set
held-test

perplexity

Perplexity → usa el log para evitar problemas numéricos

→ plantean se tiene que hacer sobre el mismo

→ curiosos

→ texto dialecto

→ detectar

→ tratamientos de palabras no vistas

→ en el train

→ tratamiento de prob.

→ (SMoothing)

corpus

→ marcar bien y son oración es separadas

Smoothing

→ replace

→ K-smoothing

→ Backoff and Interpolation

→ específico para n-grams

→ usa menos contexto ($n - m$ chico)

→ cuando no existe n -grama

→ si no existe n -grama calculo

para $(n-1)$ -grams, sino al $(n-2)$ -grams...

Generalizar N-gram

→ no puedes generalizar a palabras simples

→ embeddings x redes neuronales

(feed forward) pueden ayudar

Modelo de tiempo con redes → core computacionamente

→ embeddings preentrenados

→ extienden codificadores (es) relevantes
similares

→ no podes optimizar la forma

→ preferir una capa hidden donde se
calcula el embedding

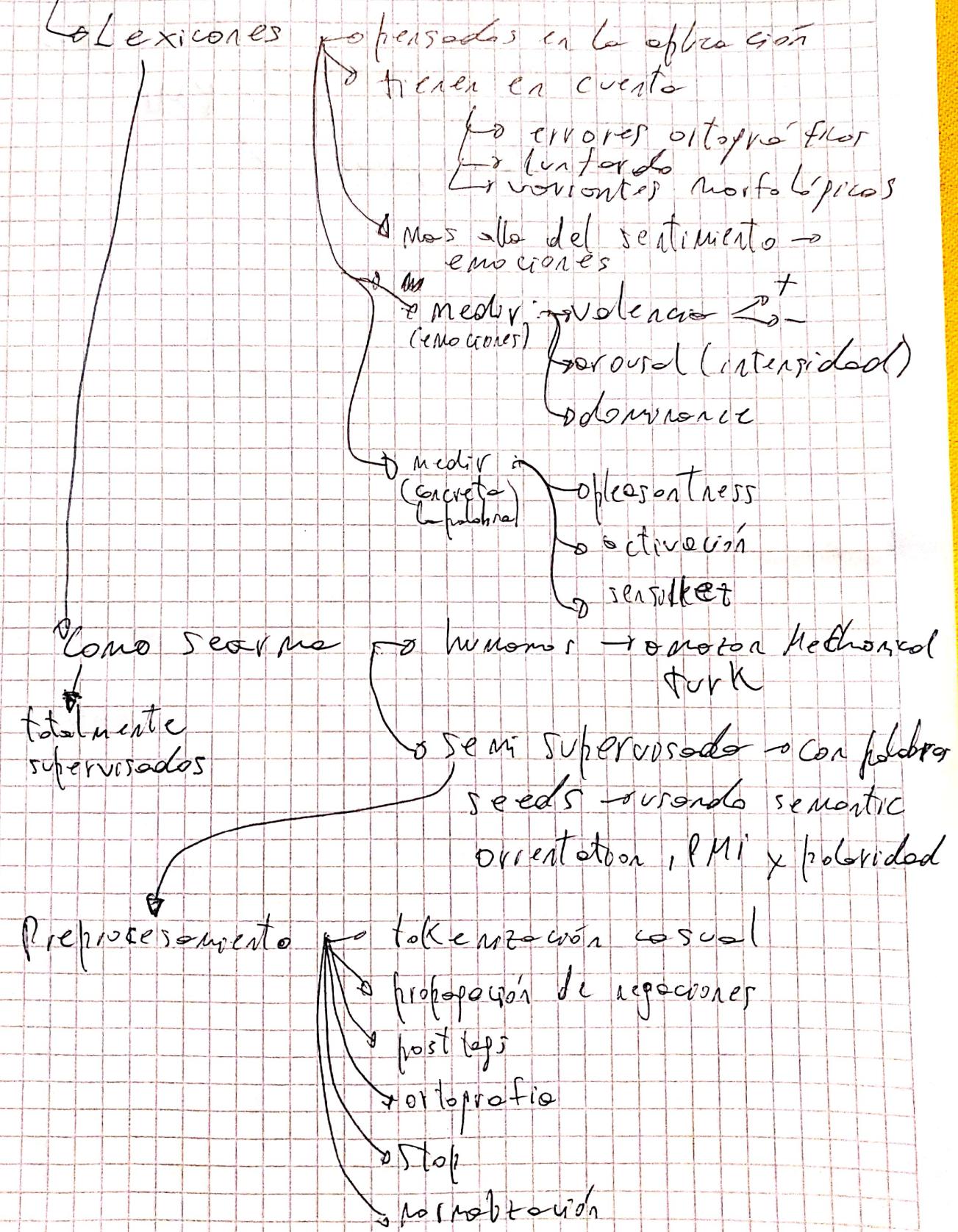
→ back propagation incluye a los
embeddings

Vídeo 10-1

HOJA N°

FECHA

Sentiment Analysis



NOTA

reAmo

Comparación de corpora

→ normalizar por conteo
→ prob. condicional
→ normalizar por categorías
o frecuencias

↓

normalizar por frecuencias
de los palabros y por } PMI
prob. de categorías