

Palabras a vectores

Fernanda Sobrino

6/22/2021

Co-occurrence matrix

Proceso de analisis de texto

1. seleccionar textos: corpus
2. definir los documentos: la unidad de análisis (tweets, oraciones, párrafos, guiones)
3. definir las características: tokens, frases, segmentos, lenguaje, etc
4. convertir estas características a una matriz
5. proceso cuantitativo o estadístico para extraer información de la matriz
6. resumen, datos nuevos, etc

Co-occurrence matrix

- ▶ matrices que representan la frecuencia con la que ocurren las palabras en un documento
- ▶ al menos dos tipos distintos:
 - ▶ term-document matrix
 - ▶ word-word matrix (term-context matrix)

Term-document matrix

- ▶ fila: palabras en el vocabulario
- ▶ columnas: documentos
- ▶ celdas: medida de frecuencia
- ▶ dim: V donde V es el tamaño del vocabulario

Term-document matrix

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

Word Vector (Passage Vector) →

Document Vector ↗

Word-Word matrix

- ▶ filas: palabras
- ▶ columnas: palabras
- ▶ dim: $V * V$
- ▶ celda: alguna medida de co-ocurrencia de las dos palabras (por lo general una ventana de i palabras)

Word-Word matrix

► Ejemplo:

- I like deep learning.
- I like NLP.
- I enjoy flying.

	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	0	1	0
enjoy	1	0	0	0	0	0	0	1
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Term-document matrix

Posibles medidas de co-ocurrencia

- ▶ binario: 1 si la palabra está
- ▶ frecuencia: cuantas veces aparece cada palabra
- ▶ ti-idf: contiene información de las palabras mas y menos importantes de todos los textos

Binario

1. the red dog →

the	red	dog	cat	eats	food
1	1	1	0	0	0
0	0	1	1	1	0
0	0	1	0	1	1
0	1	0	1	1	0

2. cat eats dog →

3. dog eats food →

4. red cat eats →

Frecuencia

Documents



Vector-space
representation

However, complexity
We will see how small
Given a function-based
Using entropy of traffic
We study the complexity
of influencing elections
through bribery: How
computationally complex
is it for an external actor
to determine whether by
a certain amount of
bribing voters a specified
candidate can be made
the election's winner? We
study this problem for
election systems as varied
as scoring ...

	D1	D2	D3	D4	D5
complexity	2		3	2	3
algorithm	3			4	4
entropy	1			2	
traffic		2	3		
network		1	4		

Term-document matrix

Term frequency - inverse document frequency (tf-idf)

- ▶ busca palabras importantes en el contenido del documento
- ▶ la multiplicación de dos métricas
 - ▶ la frecuencia en el documento
 - ▶ penaliza palabras muy frecuentes en todos los documentos

Definiciones

- ▶ t : palabra/término
- ▶ d : documento
- ▶ N : total de documentos en el corpus
- ▶ vocab : vocabulario del corpus, lista de todas las posibles palabras en el corpus

Frecuencia de un término (tf)

- ▶ mide la frecuencia de una palabra en un documento
- ▶ depende en la longitud del documento y que tan común es la palabra
- ▶ \implies aplicar normalización a la frecuencia
- ▶ los documentos van a ser vectorizados en el vocab

Frecuencia de un término (tf)

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t'} f_{t',d}}$$

donde $f_{t,d}$ es cuantas veces aparece el término t en el documento d

Frecuencia del documento (df)

- mide la importancia del documento en todo el corpus
- es la cantidad de documentos en los cuales se encuentra la palabra

$$df(t) = \begin{cases} 1 & t \in d \\ 0 & otherwise \end{cases}$$

##Frecuencia inversa de documento(idf): * mide que tan informativo es una palabra * el logaritmo evita que si N es muy grande idf explote

$$idf(t) = \log\left(\frac{N}{df(t) + 1}\right)$$

TD-IDF

$$td - idf(t, d) = tf(t, d) * idf(t)$$

TD-IDF

- ▶ nos permite asociar cada palabra en un documento con un número que representa que tan relevante es cada palabra en ese documento
- ▶ documentos con palabras relevantes similares van a tener vectores similares

Problemas con estas representaciones

- ▶ sparse matrixes (tienen muchos ceros)
- ▶ computacionalmente costosos
- ▶ vocabulario
- ▶ significado

Bag of Words

- ▶ vocabulario del corpus
- ▶ medida de frecuencia de las palabras
- ▶ cada palabra en el vocabulario es una característica
- ▶ los documentos son representados como un vector de la dimensión del vocabulario
- ▶ puede usar como medida de frecuencia: binaria, frecuencia, tf-idf

Word-Word matrix

Cuál es el significado de las palabras?

- ▶ Existe un área de la lingüística dedicada a esto: semántica léxica
- ▶ Esta rama tiene conceptos distintos que nos ayudan a definir una palabra

Lemas

- ▶ unidades semánticas más pequeñas
- ▶ cada uno posee un significante y un significado único
- ▶ sus denotaciones y connotaciones no pueden ser representadas por ningún otro lema

Lemas

rata¹

De or. inc.; cf. fr. *rat*, ingl. *rat*, al. *Ratte*.

1. f. Mamífero roedor, de unos 36 cm desde el hocico a la extremidad de la cola, muy larga, con cabeza pequeña, hocico puntiagudo, orejas tiesas, cuerpo grueso, patas cortas, pelaje gris oscuro, muy fecundo y voraz.
2. f. Hembra del ratón.
3. f. coloq. Persona despreciable.
4. f. rur. Coleta de pelo pequeña y muy delgada.
5. f. germ. Bolsillo del vestido.
6. m. coloq. *ratero* (|| ladrón).
7. m. y f. coloq. Persona tacaña.

Sónonimos

- ▶ tienen el mismo significado en algunos o todos los contextos
- ▶ abundante/mucho
- ▶ anteojos/gafas/lentes
- ▶ embrujar/hechizar
- ▶ agua/ H_2O

Sónonimos

- ▶ no existen sinónimos perfectos
- ▶ a pesar de tener significados prácticamente idénticos
- ▶ pueden variar dependiendo del contexto, jerga, formalidad, etc

Otras relaciones entre palabras

- ▶ Semejanza: palabras que se parecen en significado pero que no comparten exactamente la misma definición
 - ▶ vaca, caballo
 - ▶ auto, bicicleta
- ▶ Asociación: marcos semánticos
 - ▶ café, te : similares
 - ▶ café, taza: relacionadas pero no similares

Otras relaciones entre palabras

- ▶ antónimos: significados que son opuestos
 - ▶ apagar/prender
 - ▶ alto/bajo
- ▶ connotaciones: las palabras expresan sentimientos
 - ▶ a veces no es obvio el sentido de estas palabras

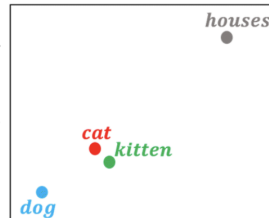
Semántica vectorial

- ▶ Idea central: 2 palabras son similares si tienen un contexto de palabras similar
- ▶ podemos definir el significado como un punto en un espacio multidimensional
- ▶ cada palabra es un vector
- ▶ palabras similares van a estar cerca

Semántica vectorial

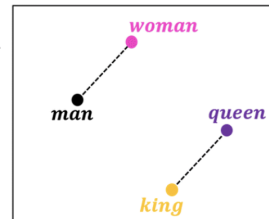
	living being	feline	human	gender	royalty	verb	plural
<i>cat</i> →	0.6	0.9	0.1	0.4	-0.7	-0.3	-0.2
<i>kitten</i> →	0.5	0.8	-0.1	0.2	-0.6	-0.5	-0.1
<i>dog</i> →	0.7	-0.1	0.4	0.3	-0.4	-0.1	-0.3
<i>houses</i> →	-0.8	-0.4	-0.5	0.1	-0.9	0.3	0.8

Dimensionality
reduction of
word
embeddings
from 7D to 2D



<i>man</i> →	0.6	-0.2	0.8	0.9	-0.1	-0.9	-0.7
<i>woman</i> →	0.7	0.3	0.9	-0.7	0.1	-0.5	-0.4
<i>king</i> →	0.5	-0.4	0.7	0.8	0.9	-0.7	-0.6
<i>queen</i> →	0.8	-0.1	0.8	-0.9	0.8	-0.5	-0.9

Dimensionality
reduction of
word
embeddings
from 7D to 2D



Embeddings

- ▶ la característica va a ser un vector
- ▶ va a ser mas fácil ver similitudes entre vectores
 - ▶ antes: perro (0, 0, 0, 0, 0, 0, 1, 0, ...) difícil de comparar con gato (1, 0, 0, 0, 0, 0, 0, 0, ..)
 - ▶ ahora: perro (22, 10, 1, 5, ...) comparable con gato (21, 11, 2, 5, ...)
- ▶ podemos generalizar para palabras similares que no hayamos visto antes

Word embeddings

- ▶ las palabras del vocabulario son mapeadas a vectores
- ▶ estos vectores son calculados de la probabilidad de que cada una de las palabras aparezca antes o después de otra (ventana de contexto)
- ▶ palabras con contextos similares van a tener representaciones vectoriales parecidas
- ▶ vectores densos (sin entradas iguales a cero) de dimensión reducida
- ▶ pueden ser aprendidos de los datos

Ventana de Contexto




- ▶ los embeddings identifican las palabras que ocurren en esta ventana
- ▶ se define como las palabras antes y después de la palabra 'central' (la que queremos entrenar)
- ▶ cada palabra 'central' y sus palabras de contexto se pueden representar como un vector que representa si las palabras están o no en un documento






Ventana de Contexto

 : Center Word

 : Context Word

c=0 The cute  jumps over the lazy dog.

c=1 The    over the lazy dog.

c=2      the lazy dog.

Variaciones de las matrices de co-ocurrencia

1. matrices de $V * V$, difíciles de manejar
2. matrices de $V * N$ donde $N \subset V$ y se puede obtener quitando palabras irrelevantes como las stopwords, etc.

Estas dos representaciones siguen siendo problemáticas: sparsity!!

- Solución: la matriz de co-ocurrencia se descomponen usando cosas como PCA, SVD, etc en factores. Cada palabra será un vector de dimensión k y no de dimensión V

Pasos para generar un vector de contexto (SVD)

- ▶ generar la matriz X de co-ocurrencia de dimensiones $|V| \times |V|$
- ▶ aplicar reducción de dimansionalidad $X = U\Sigma V^T$
- ▶ seleccionar k columnas de U para obtener vectores de esa dimensión
- ▶ varianza capturada por las k primeras dimensiones: $\frac{\sum_{i=1}^k \sigma_i}{\sum_{i=1}^{|V|} \sigma_i}$
- ▶ donde σ_i son los valores singulares de X
- ▶ $Xv = \sigma u$

Word embeddings

- ▶ existen dos maneras de aprenderlos:
 - ▶ junto a la tarea principal: los actualizas junto con tu problema (sentiment analysis, clasificación)
 - ▶ usas un modelo de word embeddings pre-entrenado

Embeddings pre entrenados

- ▶ Word2Vec (2013) Google
- ▶ GloVe (2014) Stanford
- ▶ FastText (2016) Facebook

Word2Vec

- ▶ combinación de dos técnicas:
 - ▶ CBOW (continuous bag of words)
 - ▶ Skip-gram
- ▶ redes neuronales de una sola capa
- ▶ mapean palabras a palabras
- ▶ los pesos de estas redes son la representación vectorial

CBOW

- ▶ predice la probabilidad de una palabra dado su contexto
- ▶ contexto: las palabras en la ventana de contexto
- ▶ supongamos por ahora solo hay 1
- ▶ usamos la palabra de contexto para predecir la palabra central/objetivo

CBOW: ejemplo matriz

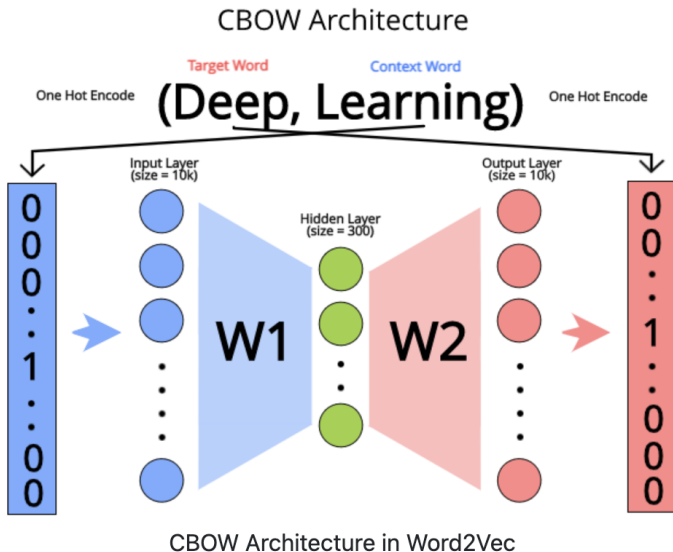
- ▶ Corpus: “Deep learning es difícil y divertido”
- ▶ context window: 1

input	output	Deep	Learning	es	difícil	y	divertido
Deep	Learning	1	0	0	0	0	0
Learning	Deeep	0	1	0	0	0	0
Learning	es	0	1	0	0	0	0
es	Learning	0	0	1	0	0	0
es	difícil	0	0	1	0	0	0
difícil	es	0	0	0	1	0	0
difícil	y	0	0	0	1	0	0
y	difícil	0	0	0	0	1	0
y	divertido	0	0	0	0	1	0
divertido	y	0	0	0	0	0	1

CBOW

- ▶ input layer: vector one-hot encoded (0's y 1's) de tamaño $1 \times V$
- ▶ hidden layer: función de activación lineal $x \cdot w$, la cantidad de nodos va a determinar el tamaño de los vectores de palabras (word vectors)
- ▶ output layer: vector one-hot encoded de tamaño $1 \times V$
activación softmax
- ▶ ejemplos de entrenamiento: los vectores obtenidos de las combinaciones de palabras
- ▶ función de costos: negative log likelihood

CBOW: ventana de contexto 1



CBOW: ventana de contexto $c > 1$

- ▶ lo unico que cambia es el input, en vez de un solo vector
- ▶ c vectores
- ▶ la activación de la capa intermedia $\frac{x \cdot w}{c}$
- ▶ el word vector son los pesos entre la capa intermedia y la capa de salida

CBOW:

- ▶ Ventajas:
 - ▶ modelo de probabilidades
 - ▶ no usa tanta RAM como una matriz de co-ocurrencias
- ▶ Problemas:
 - ▶ toma el contexto promedio de las palabras (función de activación lineal) entonces palabras que se usan en distintos contextos terminan en entre contextos. Ejemplo: naranja termina entre frutas y colores
 - ▶ entrenarlas desde cero es muy lento

Skip-gram

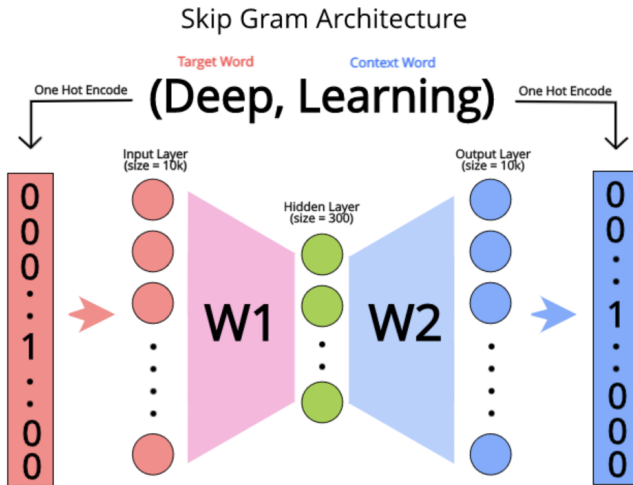
- ▶ intenta predecir cada palabra de contexto dada la palabra central

input	output(context1)	output(context1)
Deep	Learning	<SPACE>
Learning	Deep	es
es	Learning	difícil
difícil	es	y
y	difícil	divertido
divertido	y	<SPACE>

Skip-gram

- ▶ input: un vector one-hot encoded de dimensión $1 \times V$
- ▶ hidden: función de activación lineal $x \cdot w$
- ▶ output: vectores one-hot encoded con las palabras de contexto (ej. 2), función de activación softmax
- ▶ word vector: los pesos entre la capa de entrada y la capa intermedia

Skip-gram



Skip Gram architecture in Word2Vec

Qué están haciendo las redes neuronales en estos modelos?

- ▶ la red esta intentando capturar información semántica de las palabras
- ▶ la red aprende usando todas las palabras de contexto y centrales
- ▶ si tenemos suficientes ejemplos la red empezara a aprender que palabras son semánticamente similares

Pesos de una red neuronal

- ▶ podemos decir que las matrices de pesos entre capas están guardando información o patrones de lo que la red ya aprendió
- ▶ los pesos no tienen mucho sentido por si solos, solo en el contexto de la red en particular
- ▶ Recordemos nuestra red con una sola capa intermedia
- ▶ $H = X \cdot W_1$

Pesos de una red neuronal

- ▶ input: $X = ([0, 0, 0, 0, 0, 1, 0, 0, 0, \dots])$
- ▶ entonces $X \cdot W_1$ solo una fila en W_1 va a conservar sus valores, las demás se vuelven 0
- ▶ entonces podemos decir que la fila n (donde esta el 1 en el vector) contienen toda la información del aprendizaje de la red con respecto a esa palabra
- ▶ \implies que W_1 (o W_2 skip-gram) son las palabras a vectores