

NLP

Preprocesamiento de texto.
Word Embeddings

Dr. Rodrigo Cardenas Szigety rodrigo.cardenas.sz@gmail.com

Programa de la materia



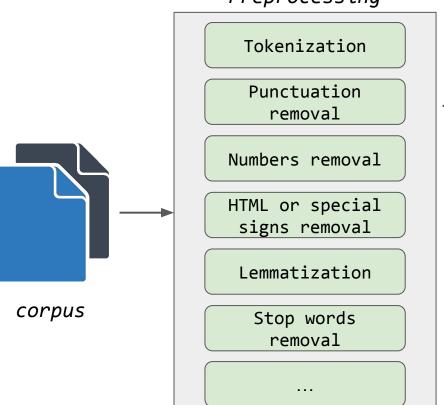
- Clase 1: Introducción a NLP, Vectorización de documentos.
- Clase 2: Pre-procesamiento de texto. Word embeddings.
- Clase 3: Modelos no-BOW I: convolucionales y recurrentes. Generación de secuencias.
- Clase 4: Modelos no-BOW II: mecanismo de atención.
- Clase 5: Modelos Seq2seq.
- Clase 6: Transformers.
- Clase 7: Grandes modelos de lenguaje. RAG.
- Clase 8: Otros temas: Image captioning. ASR y TTS.
- *Unidades con desafíos a presentar al finalizar el curso.
- *Último desafío y cierre del contenido práctico del curso.

Preprocesamiento de texto

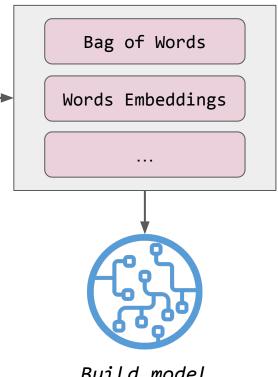
LINK GLOSARIO







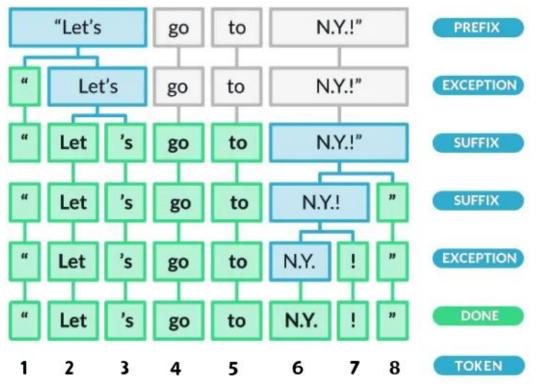
Feature Engineering



Build model

Segmentar y tokenizar

Proceso en el cual una oración o documento es segmentado en términos individuales. Una vez finalizada la segmentación cada término único es referenciado mediante un token.

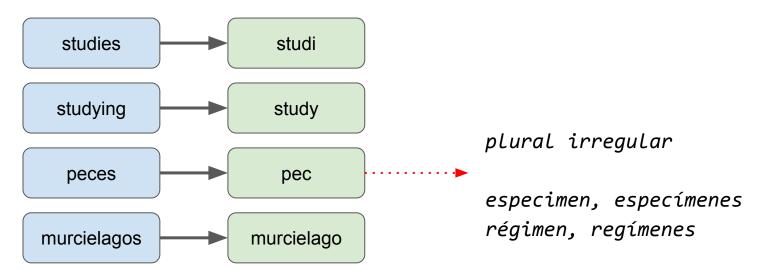


Derivado (steeming)



Aplica reglas de eliminación de patrones recurrentes de la lengua. El resultado es una palabra truncada que no será necesariamente la raíz morfológica de la palabra.

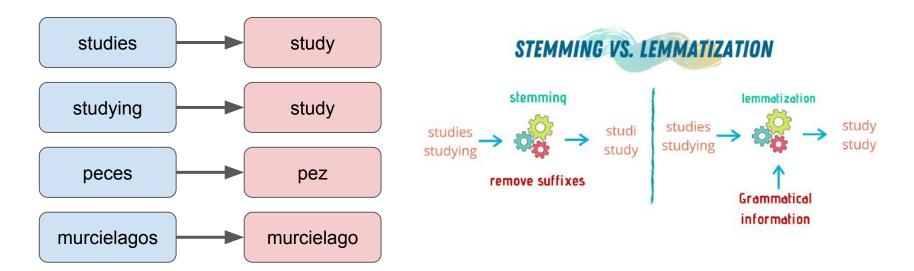
Regla: Eliminar los sufijos



Lematización (lemmatization)



Devuelve la raíz morfológica de una palabra. Para ello se necesita un diccionario del idioma con todas las declinaciones posibles de las palabras raíz.



Parts-of-speech (POS) tagging



POS es el proceso de clasificar cada término en un texto en sus categorías gramaticales, etiquetándolos por ejemplo como **sustantivo** (noun), verbo (verb), adjetivo (adj), etc



Named-entity recognition (NER)

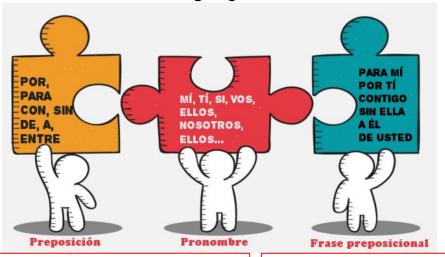


NER es el proceso de clasificar nombres propios de entidades en categorías predefinidas a las cuales pertenecen.

Name Date Designation	Subject	Named Entity Recognition
John McCarthy who was born on	September 4, 1927	was an American computer scientist and
cognitive scientist. He was one of t	he founders of the c	discipline of artificial intelligence. He co-authored
the document that coined the tern	n "Artificial intellige	nce" (AI), developed the programming language
family Lisp, significantly influenced	the design of the lo	anguage ALGOL

Stop words

Palabras que no aportan valor al significado de una oración ya que son muy frecuentes o comunes en el lenguaje



Argentina, oficialmente, República Argentina, es un país soberano de América del Sur, ubicado en el extremo sur y sudeste de dicho subcontinente. Adopta la forma de gobierno republicana, democrática, representativa y federal.

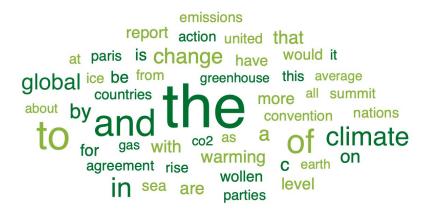
Argentina, oficialmente, República Argentina, es país soberano América Sur, ubicado extremo sur sudeste dicho subcontinente. Adopta forma gobierno republicana, democrática, representativa federal.

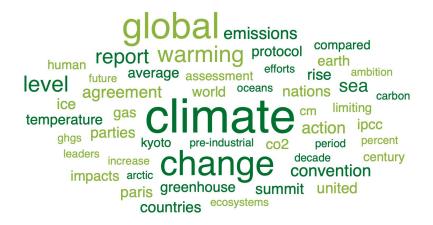
Stop words

Analizar un texto relacionado con calentamiento global (global climate)

Texto con Stop Words

Texto sin Stop Words





Las Stop Words pueden depender del contexto del corpus.

Librerías de NLP





Gran comunidad de desarrollo

No soporta GPU

Más optimizada en CPU

Más lenta en gran volúmenes de datos o operaciones

spaCy

Más moderna e implementa los últimos features

Soporta GPU

Menos optimizada en CPU

Más rápida en gran volúmenes de datos o operaciones (GPU)

LINK PÁGINA





"Librería por excelencia de procesamiento de lenguaje natural para Python" Inicios 2009

Implementa una tool/algoritmo para cada etapa de preprocesamiento de NLP



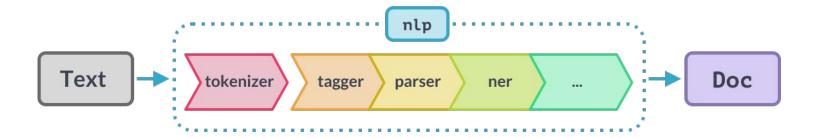
LINK PÁGINA





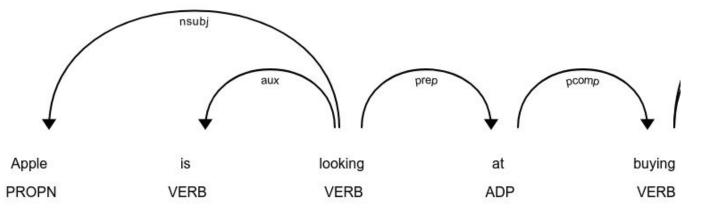
Inicios 2015

- Support for 72+ languages
- 80 trained pipelines for 24 languages
- Multi-task learning with pretrained transformers like BERT
- Pretrained word vectors
- State-of-the-art speed
- Production-ready training system



LINK PÁGINA





Un resumen de todo lo visto

<u>P0S</u>

TAG

DEP

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
Apple	apple	PROPN	NNP	nsubj	Xxxxx	True	False
is	be	AUX	VBZ	aux	xx	True	True
looking	look	VERB	VBG	ROOT	xxxx	True	False
at	at	ADP	IN	prep	xx	True	True
buying	buy	VERB	VBG	pcomp	xxxx	True	False
U.K.	u.k.	PROPN	NNP	compound	x.x.	False	False
startup	startup	NOUN	NN	dobj	xxxx	True	False
for	for	ADP	IN	prep	xxx	True	True
\$	\$	SYM	\$	quantmod	\$	False	False
1	1	NUM	CD	compound	d	False	False
billion	billion	NUM	CD	pobj	xxxx	True	False

Preprocesamiento





Problemas con CountVectorizer/OHE/TF-IDF



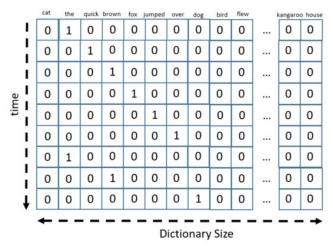
Textos de significado similar pueden ser "ortogonales"

Viajando en colectivo

Voy arriba del bus

La dimensión de los vectores depende del tamaño del vocabulario

No aprovechamos la dimensionalidad

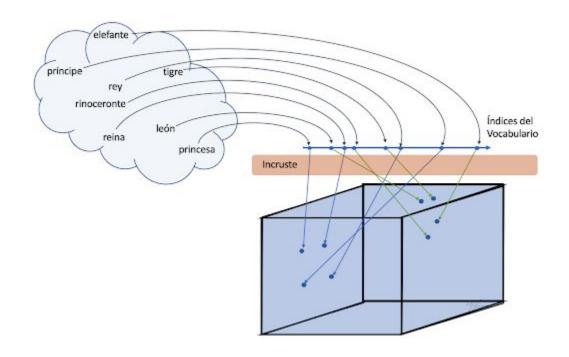


Representaciones



Embeddings

Un embedding es la representación numérica densa de tamaño fijo de un dato estructurado o no estructurado (mapear imagenes, entidades ó palabras a vectores)

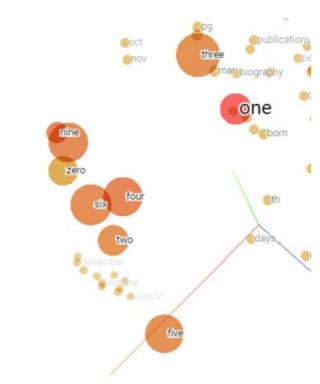


Word Embeddings

Las palabras que tienen un significado similar tendrán una representación similar

como embeddings

http://projector.ten
 sorflow.org/



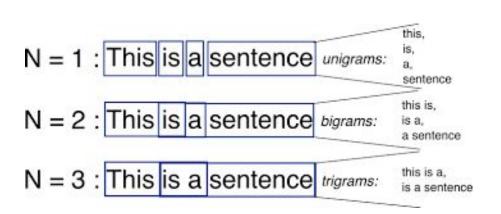
N-GRAM



21

"Subsecuencia de N elementos de una secuencia dada"

Otra forma de agrupar las palabras distinto a word2vec (N=1, unigram) en donde se busca aumentar el poder de generalización y a su vez poder hacer más variado el vocabulario.



Character-level unigranis									
Token Sequence	Token Value								
1	D								
2	0								
3	g								
ogs 4									
Character-level bigrams									
Token Sequence	Token Value								
1	Do								
2	og								
3	gs								
grams									
Token Sequence	Token Value								
1	Dog								
2	ogs								
	Token Sequence 1 2 3 4 grams Token Sequence 1 2 3 grams Token Sequence 1 1								

Character level unigrame

GloVe y fastText



Embeddings pre-entrenados basados en diferentes enfoques:





Tokenización basada en palabras



Entrenado con textos de Wikipedia, Common Crawl y GigaWord 5



Se basa en calcular la matriz de co-ocurrencia de palabras y estimar el cociente de probabilidad de aparición.

fastText



Tokenización basada en N-Grams de caracteres (3 a 6). Mejora la interpretación de sufijos y prefijos



Entrenado con una colección de 8 corpus (portales de noticias, reviews, Wlkipedia)



Basado en word2vec (CBOW/Skip-Gram)



Puede crear un embedding de una palabra que nunca vió

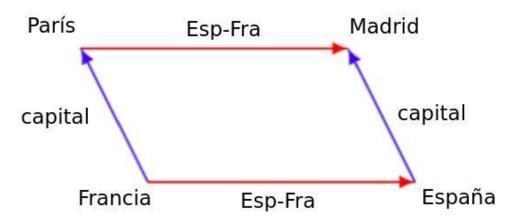
Operaciones con Embeddings: tests de analogías



Una forma de testear la calidad de embeddings es probar su desempeño en tests de analogías:

París es a Francia lo que Madrid es a España. Madrid y París corresponden a España y Francia

$$\overrightarrow{Paris} - \overrightarrow{Francia} \approx \overrightarrow{Madrid} - \overrightarrow{Espana}$$
 $simcos(\overrightarrow{Paris} - \overrightarrow{Francia}, \overrightarrow{Madrid} - \overrightarrow{Espana}) \approx 1$



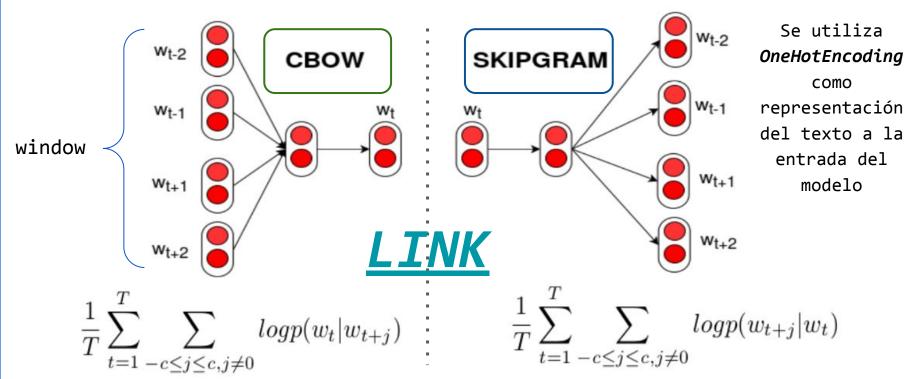
Embeddings Glove y Fasttext





¿Cómo podemos crear nuestros word Embeddings?

Aprendiendo (con redes neuronales) vectores para cada palabra que maximicen la relación entre las palabras de contexto y la palabra objetivo. Esto es lo que se implementó en la librería **word2vec**.

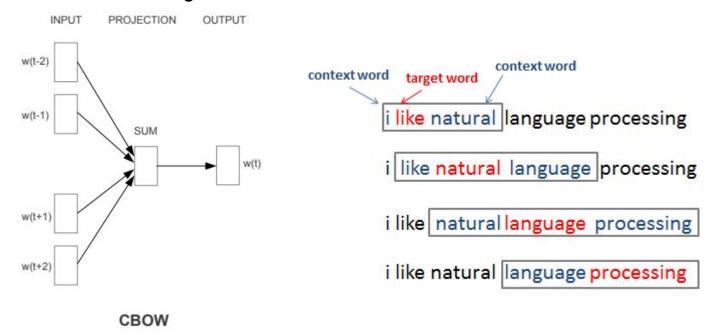


25

Continuous Bag of Words Model (CBOW)



Utiliza como entrada el contexto de la palabra objetivo (palabras a izquierda y derecha de ella). El tamaño de la ventana determina cuántas palabras se tomarán para contextualizar el embedding.

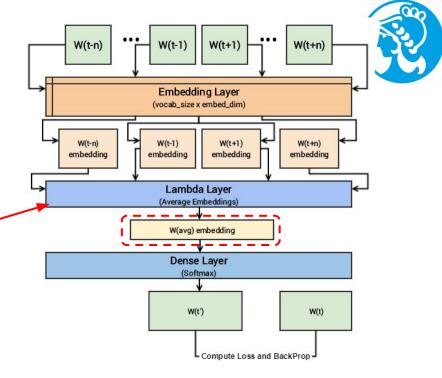


CBOW - Entrenamiento

LINK

Para entrenar necesitamos tener el vocabulario del corpus y las sentencias organizadas por el tamaño de la ventana de entrada.

Los embeddings de cada palabra son el embedding promedio de todas las veces que se utilizó en el corpus.



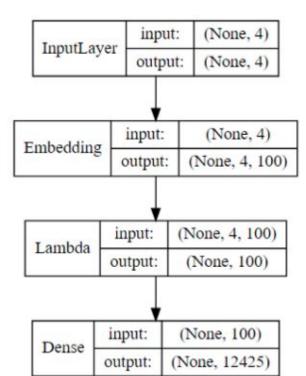
```
cbow = Sequential()
cbow.add(Embedding(input_dim=vocab_size, output_dim=embed_size, input_length=window_size*2))
cbow.add(Lambda(lambda x: K.mean(x, axis=1), output shape=(embed_size,)))
cbow.add(Dense(vocab_size, activation='softmax'))
cbow.compile(loss='categorical_crossentropy', optimizer='rmsprop')
```

CBOW - Entrenamiento



Con tan solo un corpus de 12425 palabras distintas y embedding de 100 dimensiones hay que entrenar **2.5 Millones de parámetros**

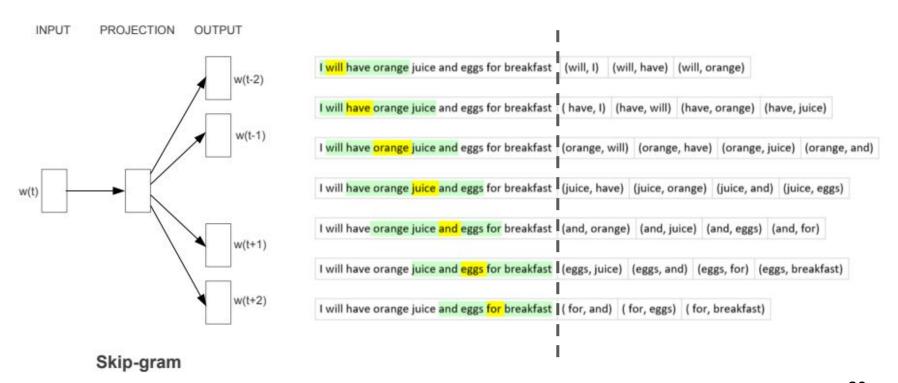
Layer (type)	Output	Shape	Param #
embedding_1 (Embedding)	(None,	4, 100)	1242500
lambda_1 (Lambda)	(None,	100)	0
dense_1 (Dense)	(None,	12425)	1254925
Total params: 2,497,425 Trainable params: 2,497,425 Non-trainable params: 0			



Skip-Gram



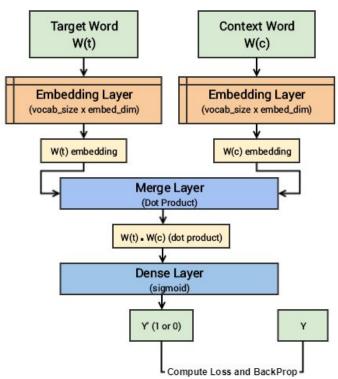
Al contrario de CBOW, este modelo intenta predecir las palabras que rodean (contexto) a una palabra objetivo. Se divide el output como pares [target, context]



Skip-Gram - Entrenamiento LINK

Por cada par [target, context] el sistema determina si las palabras tiene significado en contexto (1) o no lo tiene (0), buscando así acercar las palabras que tienen significado juntas (que se espera que estén juntas en el texto)

```
word model = Sequential()
word_model.add(Embedding(vocab_size, embed_size,
                         embeddings initializer="glorot uniform",
                         input_length=1))
word model.add(Reshape((embed size, )))
context model = Sequential()
context model.add(Embedding(vocab size, embed size,
                  embeddings_initializer="glorot_uniform",
                  input_length=1))
context_model.add(Reshape((embed_size,)))
model = Sequential()
model.add(Merge([word model, context model], mode="dot"))
model.add(Dense(1, kernel initializer="glorot uniform", activation="sigmoid"
```

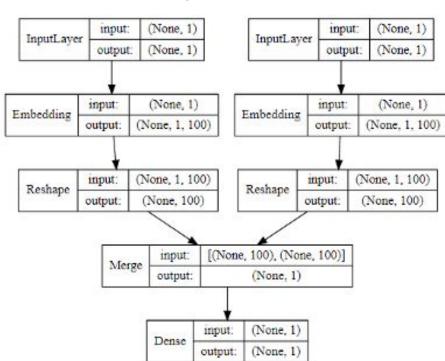


Skip-Gram - Entrenamiento



Skip-Gram requiere más datos para lograr un buen resultado pero obtiene más información sobre el contexto del corpus en sus embeddings.

Layer (type)	Output	Shape	Param #
merge_2 (Merge)	(None,	1)	0
dense_3 (Dense)	(None,	1)	2
Total params: 2,485,002 Trainable params: 2,485,002 Non-trainable params: 0			



Negative sampling

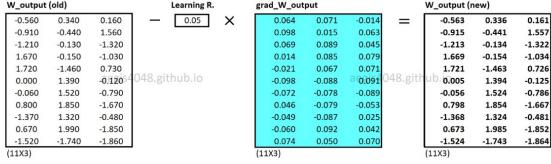




En SkipGram/CBOW la cantidad de parámetros a entrenar en la softmax es enorme:

Parametros = vocab size * embedding size → millones de parámetros





Negative Sampling

(11/3)				(11/3)				(11/13)		
W_output (old)			Learning R.	grad_W_out	put			W_output (r	iew)	5000
-0.560	0.340	0.160	- 0.05 X				_	-0.560	0.340	0.160
-0.910	-0.440	1.560	_ ^				_	-0.910	-0.440	1.560
-1.210	-0.130	-1.320		Note		al I		-1.210	-0.130	-1.320
1.670	-0.150	-1.030		NOT	ompute	a:		1.670	-0.150	-1.030
1.720	-1.460	0.730						1.720	-1.460	0.730
0.000	1.390	-0.12054	048.github.io			aegis4	048.gith	ub. 0.000	1.390	-0.120
-0.060	1.520	-0.790						0.060	1.520	0.790
0.800	1.850	-1.670	Positive sample, w_o	0.031	0.030	0.041		0.798	1.849	-1.672
-1.370	1.320	-0.480	Negative sample, k=1	-0.090	0.031	-0.065		-1.366	1.318	-0.477
0.670	1.990	-1.850	Negative sample, k=2	0.056	0.098	-0.061		0.667	1.985	-1.847
-1.520	-1.740	-1.860	Negative sample, k=3	0.069	0.084	-0.044		-1.523	-1.744	-1.858
(11X3)				(11X3)				(11X3)		

En cada iteración se observa la palabras [target, contexto] y "K" palabras aleatorias del corpus El objetivo es optimizar cómputo. Además funciona como regularización. Para corpus pequeños, el muestreo debe ser

Visualizar embeddings en baja dimensionalidad:





t-SNE (t-distributed stochastic neighbor embedding)



Técnica de reducción de dimensionalidad no-lineal (a diferencia de PCA).



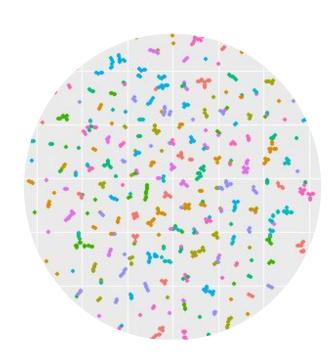
Intenta reproducir en baja dimensionalidad, la localidad de los datos en alta dimensionalidad.



Es estocástica, a priori los resultados no se repiten.



Por su carácter estocástico es sólo recomendable como herramienta de visualización y exploratoria.



Gensim - Doc2Vec paragraph embeddings

LINK



Utilizaremos esta librería que nos facilita generar embeddings tipo Skip-Gram o CBOW de nuestros corpus



- Librería de Python
- Existe desde 2009 y nació originalmente para topic modelling
- Muy popular y muy simple de utilizar

34

Generación de embeddings con Gensim





Desafío



Crear sus propios vectores con Gensim basado en lo visto en clase con otro dataset.

Probar términos de interés y explicar similitudes en el espacio de embeddings. Intentar plantear y probar tests de analogías. Graficar los embeddings

Sacar conclusiones.

resultantes.

Algunos recursos para descargar corpora de texto



Project Gutenberg

Compilación de literatura completa de dominio público principalmente en inglés.

Textos.info

Compilación de literatura completa de dominio público en español.



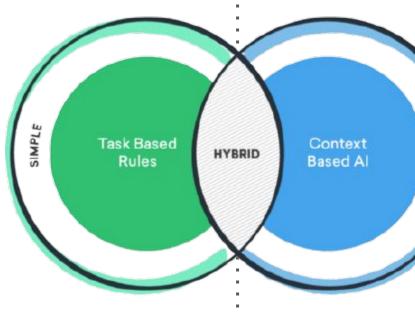
BOTs lingüísticos



Son limitados a una tarea espećifica



Fácil y "barato" de entrenar



Interactúan casi como un humano



Más difíciles de entrenar, requieren más datos y cómputo



Ideal para chats de pedidos

Ideal para
asistentes
virtuales





Sistema de obtención de información

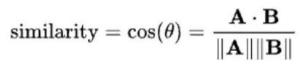




Vectorizaremos texto de un corpus. Por ejemplo: párrafos u oraciones.



Vectorizaremos el texto de entrada y devolveremos la mejor coincidencia del corpus en términos de la similaridad coseno





Probaremos el sistema armando una interfaz con gradio.



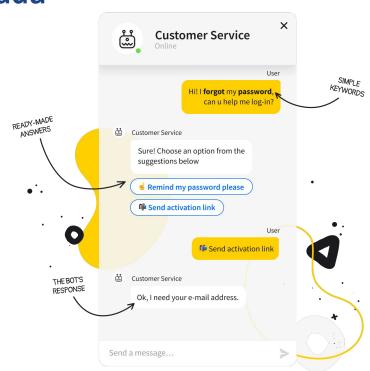
Sistema de obtención de información





BOT de consulta abierta y respuesta predeterminada







Nuestro bot será entrenado con <TAGS> (ej: saludo)



Cada <TAG> será
representado por un
patrón de posibles
preguntas
<patterns> (X)



Cada <TAG> tendrá
uno o varias
posibles respuestas
<classes> (y)

BOT de consulta abierta





Desafio



Tomar uno de los dos ejemplos mostrados y armar una versión propia





Spell checker (algunas librerias, cómo funciona), polyglot, procesar emojis, ner (dar cómo funciona).



Tokenizadores (BPE lo dejamos para después), ver si puedo simplificar la normalización de caracteres raros.

Meter vectorizadores de sklearn, algo de scrapping (no puede ser muy complejo).