Introducción al Procesamiento del Lenguaje Natural con R

Montserrat López Cobo

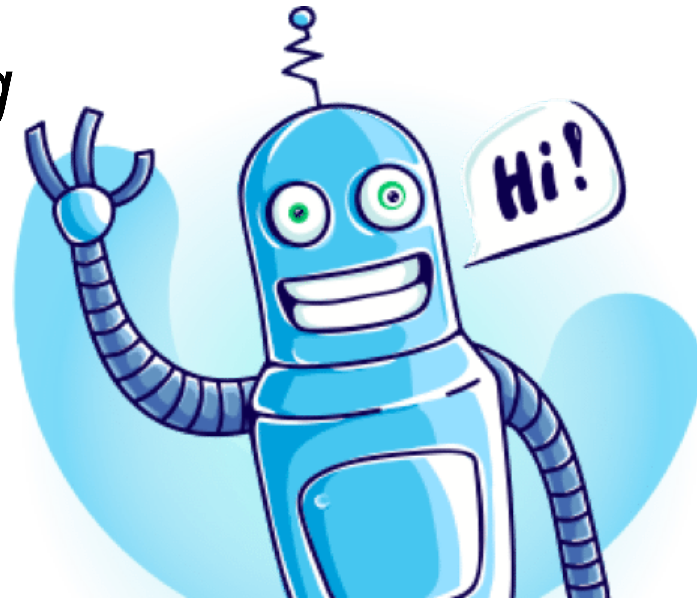
SevillaR, 1 de octubre de 2019

Qué es el PLN (*NLP* en inglés)

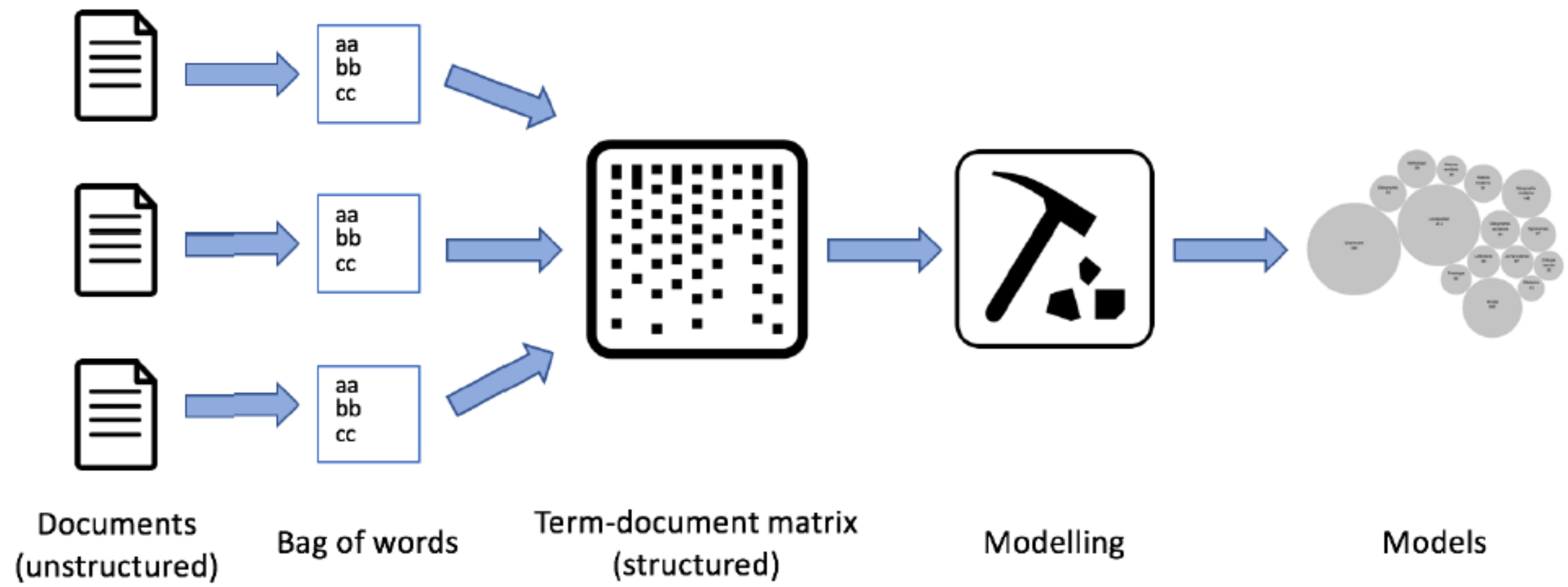
Estudia la interacción entre computadoras y lenguaje humano: cómo hacer a un ordenador comprender, reproducir o generar lenguaje humano, escrito o hablado

Aplicaciones

- traducción automática
- recuperación y extracción de información
- resumen de textos
- análisis de sentimiento
- reconocimiento del habla
- síntesis de voz
- chatbots
- *topic modelling*
- …



PLN para análisis de texto



Fuente: Tutorial on Machine Learning with R. The Melbourne Data Science Week 2017

- **Modelo de espacio vectorial (*vector space model*):**
  - Modelo algebraico
  - Representación del texto como un vector numérico de identificadores (términos)
  - Útil para: selección o filtrado de documentos, ranking de relevancia, clasificación y clustering (similaridad del coseno)

Matriz de término-documento

Binaria

		D1	D2	D3	D4	D5
T1	adversarial	1				
T2	algorithm	1		1		1
T3	computer		1			
T4	convolutional		1		1	
T5	dataset			1		
T6	deep				1	
T7	learning			1		1
T8	machine			1		
T9	network	1	1		1	
T10	neural	1	1		1	
T11	reinforcement				1	
T12	supervised			1		
T13	training			1		
T14	transfer					1
T15	vision		1			

Ponderada: frecuencia de ocurrencia

D1	D2	D3	D4	D5
3				
1		3		2
	1			
	1		1	
		1		
			2	
		2		2
		2		
3	1		3	
2	1		2	
			1	
		1		
		1		
				2
	1			

tf-idf

término **documento**

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

- **tf**: prioriza palabras frecuentes en el documento
- **idf**: prioriza palabras que aparecen en pocos documentos
- **tf-idf**: prioriza palabras que aparecen mucho en el documento y poco en el resto de documentos

tf-idf

		tf(Ti,D1)	tf(Ti,D2)	tf(Ti,D3)	tf(Ti,D4)	tf(Ti,D5)	df(i)	N/df(i)	idf(i)	tf-idf(Ti,D1)	tf-idf(Ti,D2)	tf-idf(Ti,D3)	tf-idf(Ti,D4)	tf-idf(Ti,D5)
T1	adversarial	3					1	5	0.6990	2.0969				
T2	algorithm	1		3		2	3	1.6667	0.2218	0.2218		0.6655		0.4437
T3	computer		1				1	5	0.6990		0.6990			
T4	convolutional		1		1		2	2.5	0.3979		0.3979		0.3979	
T5	dataset			1			1	5	0.6990			0.6990		
T6	deep				2		1	5	0.6990				1.3979	
T7	learning			2		2	2	2.5	0.3979			0.7959		0.7959
T8	machine			2			1	5	0.6990			1.3979		
T9	network	3	1		3		3	1.6667	0.2218	0.6655	0.2218		0.6655	
T10	neural	2	1		2		3	1.6667	0.2218	0.4437	0.2218		0.4437	
T11	reinforcement				1		1	5	0.6990				0.6990	
T12	supervised			1			1	5	0.6990			0.6990		
T13	training			1			1	5	0.6990			0.6990		
T14	transfer					2	1	5	0.6990					1.3979
T15	vision		1				1	5	0.6990		0.6990			

Aplicación

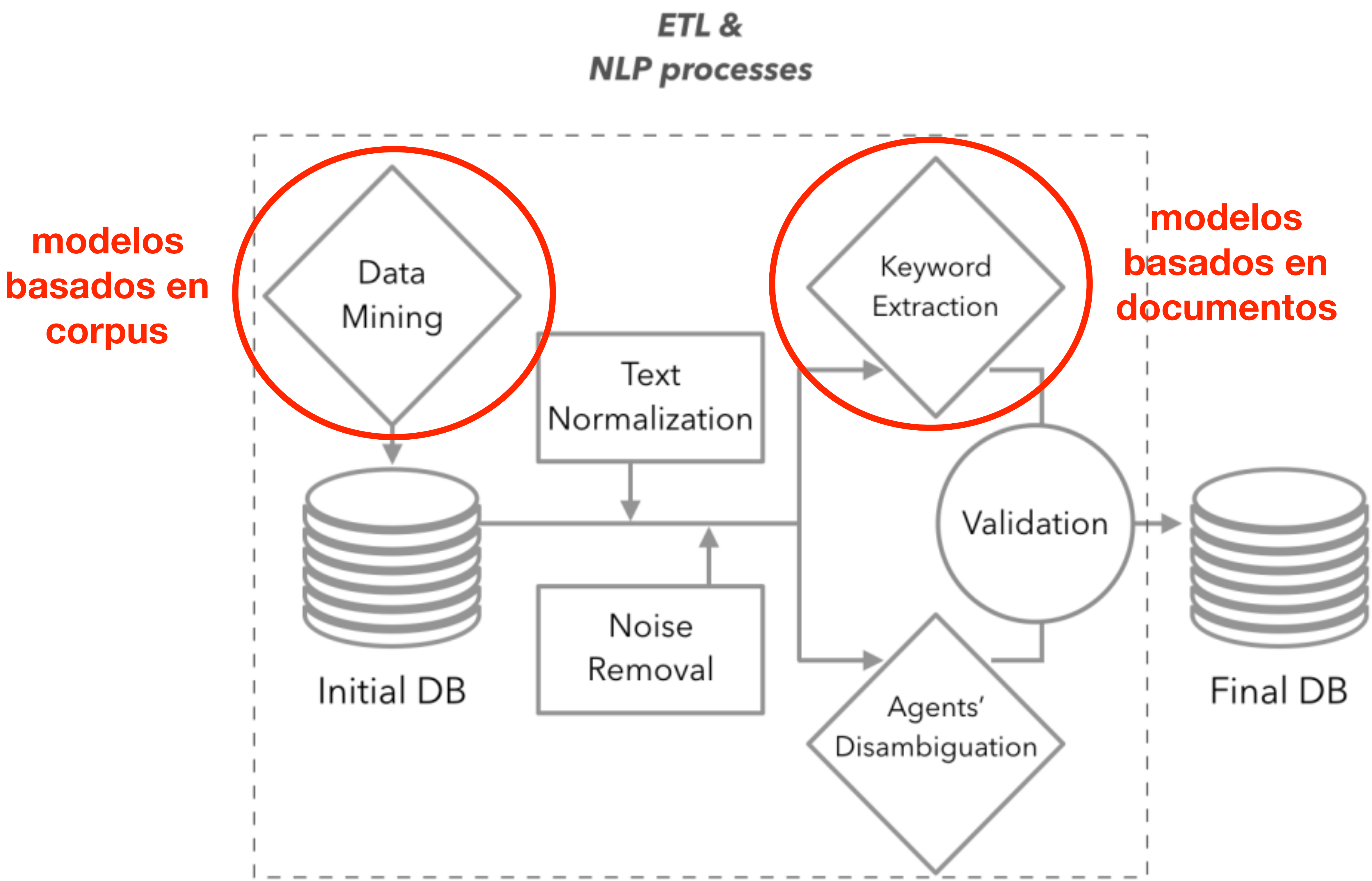
- **Caso de uso**: Analizar contenido de artículos que caracterizan una disciplina

- **Objetivos**:
  - **Identificar términos representativos** -> caracterización de la disciplina
  - **Asociar documentos y términos** -> caracterización del documento

- **Primer ejemplo**: Métodos basados en corpus
  - Comparar artículos de distintas disciplinas - tecnologías

- **Segundo ejemplo**: Métodos basados en documento
  - Resumir artículos
  - Extracción de palabras clave, POS tagging
  - Uso posterior: *Topic modelling* [no en esta sesión]

Aplicación


ETL &
NLP processes
modelos
basados en
corpus
Data
Mining
Text
Normalization
Keyword
Extraction
modelos
basados en
documentos
Initial DB
Noise
Removal
Validation
Agents'
Disambiguation
Final DB

Métodos basados en corpus

Datos de entrada

- 4000 artículos de investigación:
  - 1000 de conferencias de Inteligencia artificial (IA)
  - 1000 de revistas de Observación de la tierra (EO)
  - 1000 de revistas de Sistemas de información geográfica (GIS)
  - 1000 de revistas de Fotónica (PH)
- Fuente: Scopus y arXiv.org

Review

Responsive Photonic Crystals

Dr. Jianping Ge, Prof. Yadong Yin

First published: 20 January 2011 Full publication history

DOI: 10.1002/anie.200907091 View/save citation

Cited by (CrossRef): 432 articles Check for updates Citation tools

Funding Information

View issue TOC
Volume 50, Issue 7
February 11, 2011
Pages 1492–1522

Abstract

This Review summarizes recent developments in the field of responsive photonic crystal structures, including principles for design and fabrication and many strategies for applications, for example as optical switches or chemical and biological sensors. A number of fabrication methods are now available to realize responsive photonic structures, the majority of which rely on self-assembly processes to achieve ordering. Compared with microfabrication techniques, self-assembly approaches have lower processing costs and higher production efficiency, however, major efforts are still needed to further develop such approaches. In fact, some emerging techniques such as spin coating, magnetic assembly, and flow-induced self-assembly have already shown great promise in overcoming current challenges. When designing new systems with improved performance, it is always helpful to bear in mind the lessons learnt from natural photonic structures.

article_id	journal	description	title	tech	year
ai_1	ICML	Orthogonal matching pursuit (OMP) is a widely used a...	Signal and Noise Statistics Oblivious Orthogonal Matc...	ai	2018
ai_2	NIPS	Due to their simplicity and excellent performance, par...	Breaking the Nonsmooth Barrier: A Scalable Parallel M...	ai	2017
ai_3	NIPS	Automaton models are often seen as interpretable m...	Interpreting Finite Automata for Sequential Data	ai	2016
ai_4	CVPR	This paper presents our contribution to the ChaLearn ...	Cultural Event Recognition with Visual ConvNets and ...	ai	2015
ai_5	NIPS	We present a method for explaining the image classifi...	Using KL-divergence to focus Deep Visual Explanation	ai	2017
ai_6	AAAI	We advance the state of the art in biomolecular intera...	Extracting Biomolecular Interactions Using Semantic P...	ai	2015
ai_7	NIPS	We suggest a loss for learning deep embeddings. The...	Learning Deep Embeddings with Histogram Loss	ai	2016
ai_8	AAAI	This paper aims to shed some light on the concept of...	Exploring Text Virality in Social Networks	ai	2012
ai_9	NIPS	Majorization-minimization algorithms consist of itera...	Stochastic Majorization-Minimization Algorithms for ...	ai	2013

Toquenización de textos

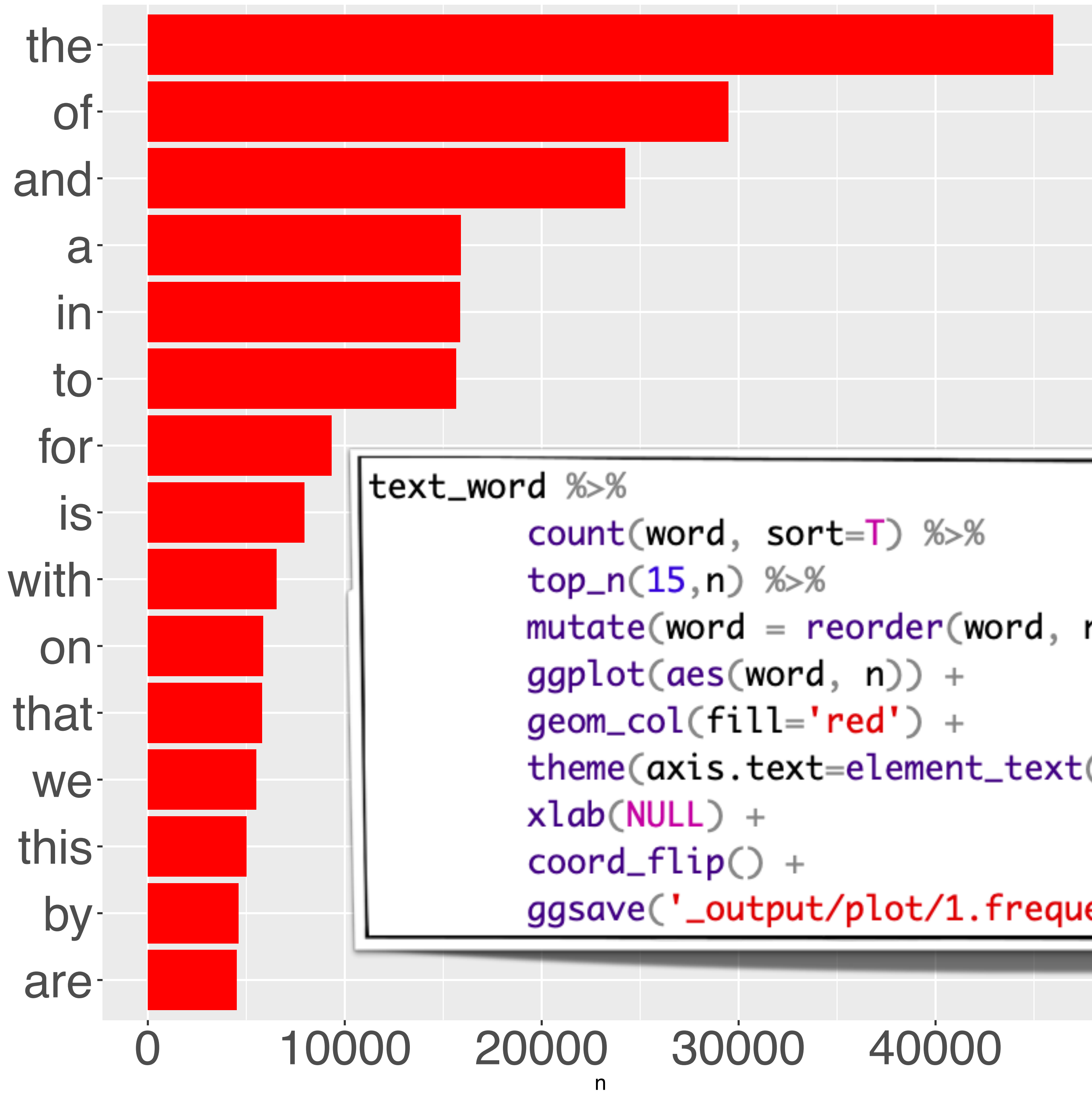
- Tokenizar: segmentar el texto en palabras
- Lematizar: extraer la raíz de la palabra: reduce variantes de una palabra a su raíz

```
text_word <- data0_text %>%
        unnest_tokens(word, text) %>%
        mutate(word = str_extract(word, "[a-zA-Z0-9']+")) %>%
        filter(!is.na(word)) %>%
        mutate(word_stemm = (wordStem(word)))
```

article_id	journal	tech	year	word	word_stemm
ai_1	ICML	ai	2018	orthogonal	orthogon
ai_1	ICML	ai	2018	matching	match
ai_1	ICML	ai	2018	pursuit	pursuit
ai_1	ICML	ai	2018	omp	omp
ai_1	ICML	ai	2018	is	i
ai_1	ICML	ai	2018	a	a
ai_1	ICML	ai	2018	widely	wide
ai_1	ICML	ai	2018	used	us

nb words: 761.719
nb unique words: 27.140

Palabras más frecuentes



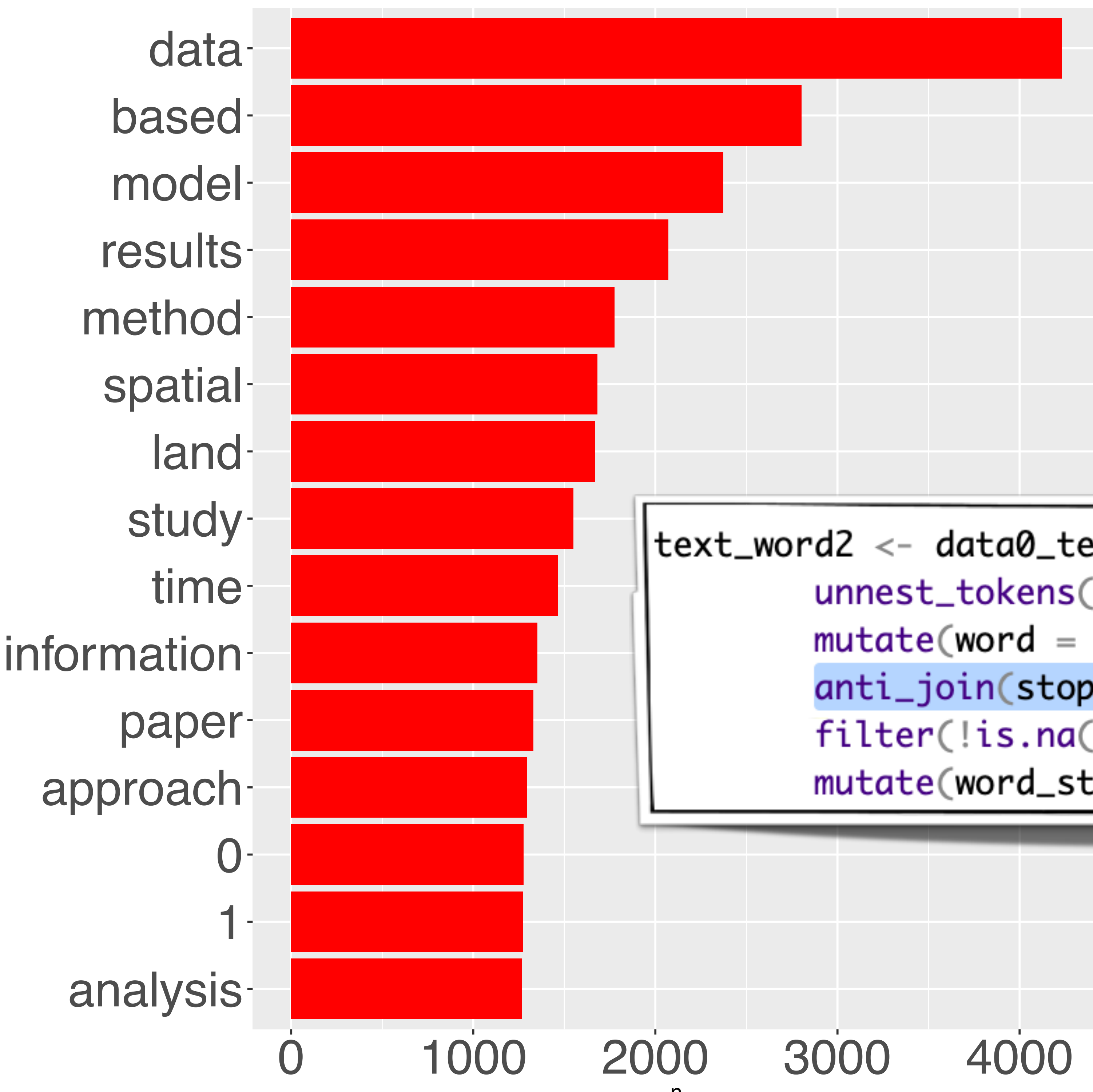

```r
text_word %>%
        count(word, sort=T) %>%
        top_n(15,n) %>%
        mutate(word = reorder(word, n)) %>%
        ggplot(aes(word, n)) +
        geom_col(fill='red') +
        theme(axis.text=element_text(size=25)) +
        xlab(NULL) +
        coord_flip() +
        ggsave('_output/plot/1.frequent_words.pdf', width = 8, height = 8)
```

Stop words

- Palabras que no aportan sentido al texto
- Que se repiten en todos los textos
- No sirven para discriminar, no aportan información
- Suelen ser: preposiciones, conjunciones, verbos…
- **data(“stop_words")** (tidytext):
    - 1149 stop words
    - fuentes: "onix", “SMART", “snowball"
    - Ejemplos:

```
"no"          "turn"      "hello"      "don't"      "doing"
"here"        "he's"      "whole"      "been"       "nine"
"especially"  "whole"     "longer"     "is"         "merely"
"ordered"     "grouping"  "everything" "obviously"  "anyone"
```

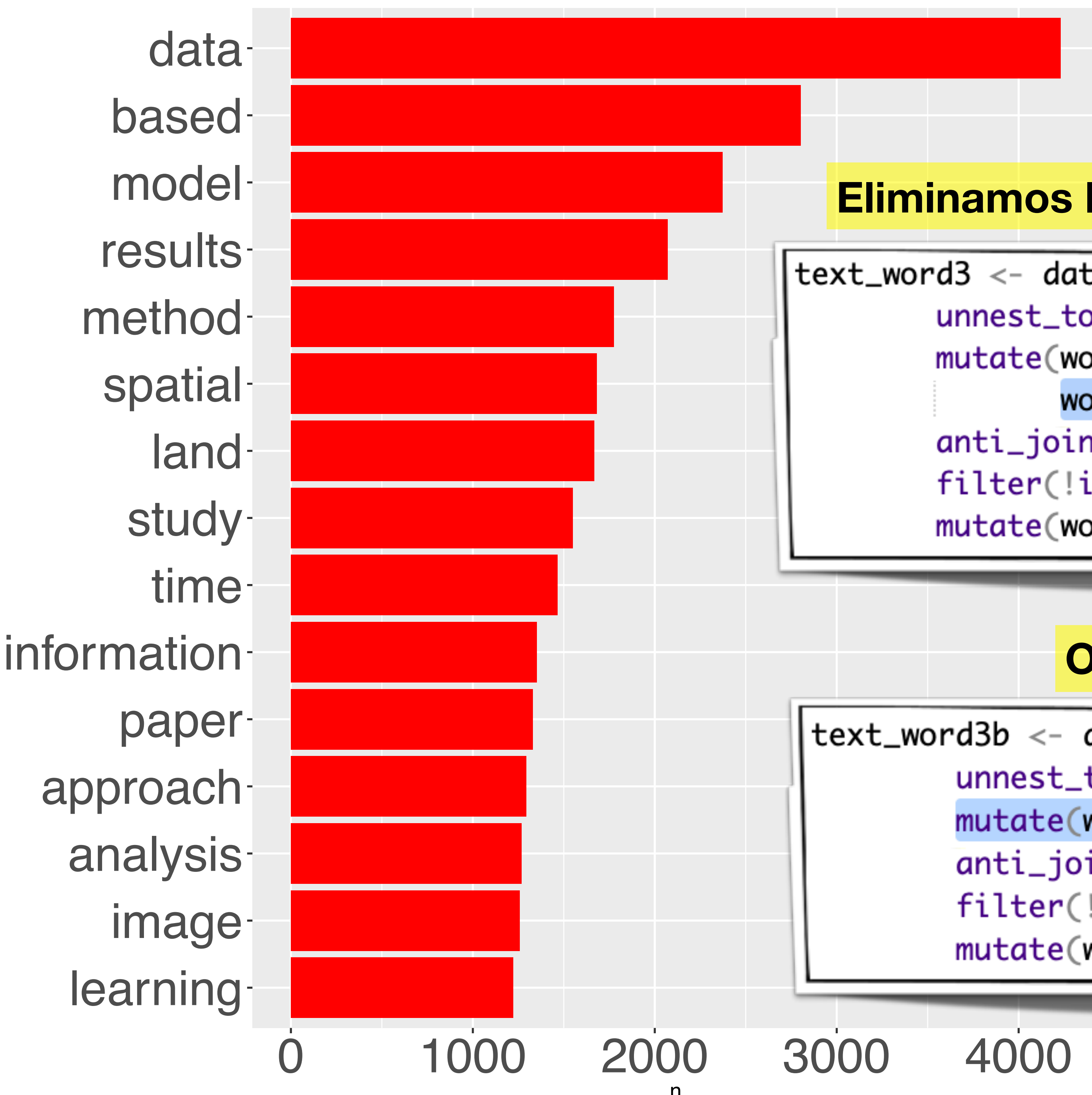
Palabras más frecuentes (sin stop words)



nb words: 431.144
nb unique words: 26.545

```r
text_word2 <- data0_text %>%
        unnest_tokens(word, text) %>%
        mutate(word = str_extract(word, "[a-zA-Z0-9']+")) %>%
        anti_join(stop_words, by='word') %>%
        filter(!is.na(word)) %>%
        mutate(word_stemm = (wordStem(word)))
```

Palabras más frecuentes (sin stop words, ni números)



nb words: 417.430
nb unique words: 25.746

Eliminamos las palabras compuestas solo por números

```
text_word3 <- data0_text %>%
        unnest_tokens(word, text) %>%
        mutate(word = str_extract(word, "[a-zA-Z0-9']+"),
               word = str_replace_all(word, '^[0-9]*$','')) %>%
        anti_join(stop_words, by='word') %>%
        filter(!is.na(word)) %>%
        mutate(word_stemm = (wordStem(word)))
```

O eliminamos todos los números

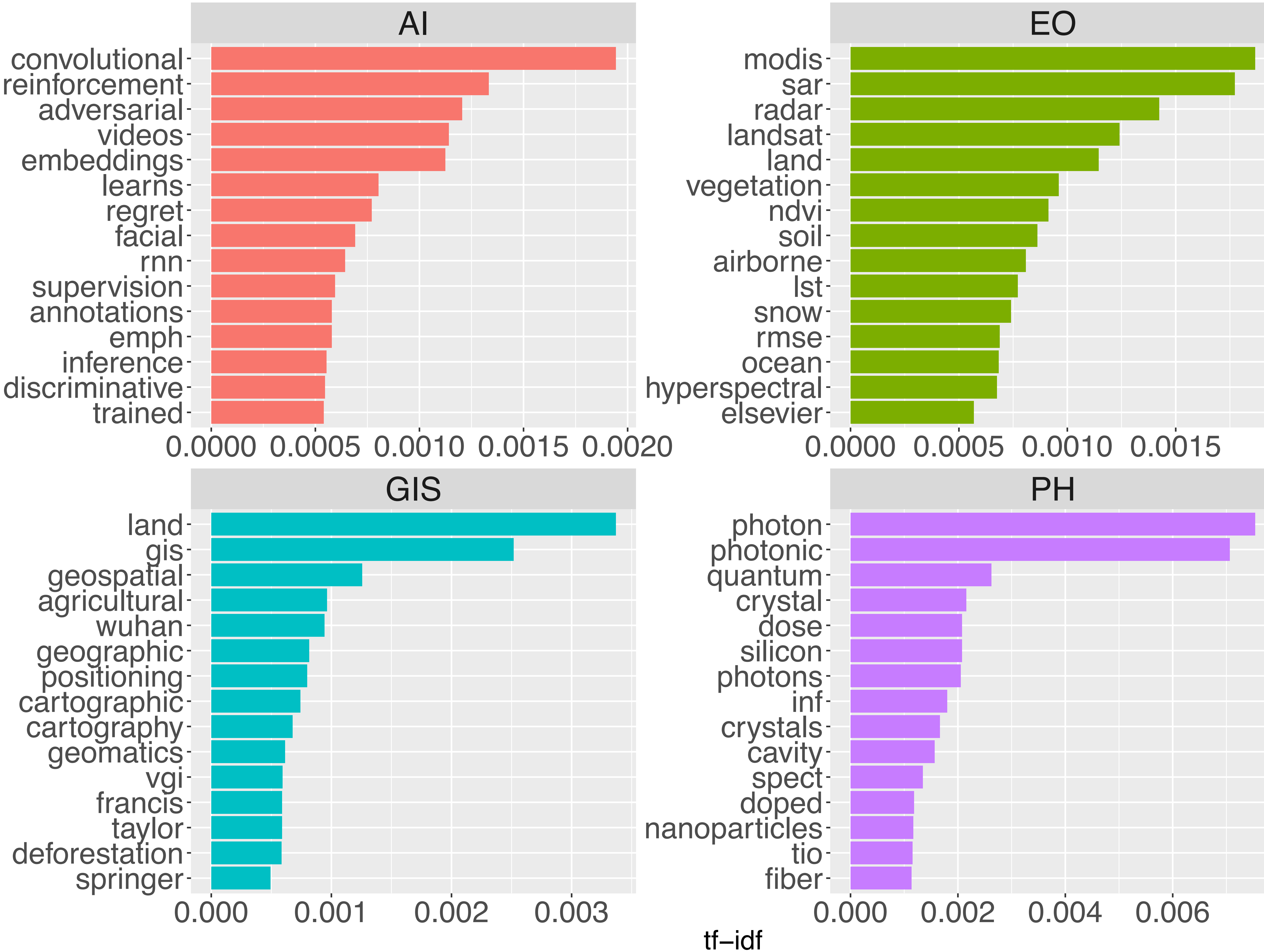
```
text_word3b <- data0_text %>%
        unnest_tokens(word, text) %>%
        mutate(word = str_extract(word, "[a-zA-Z']+")) %>%
        anti_join(stop_words, by='word') %>%
        filter(!is.na(word)) %>%
        mutate(word_stemm = (wordStem(word)))
```

Términos representativos por tecnología

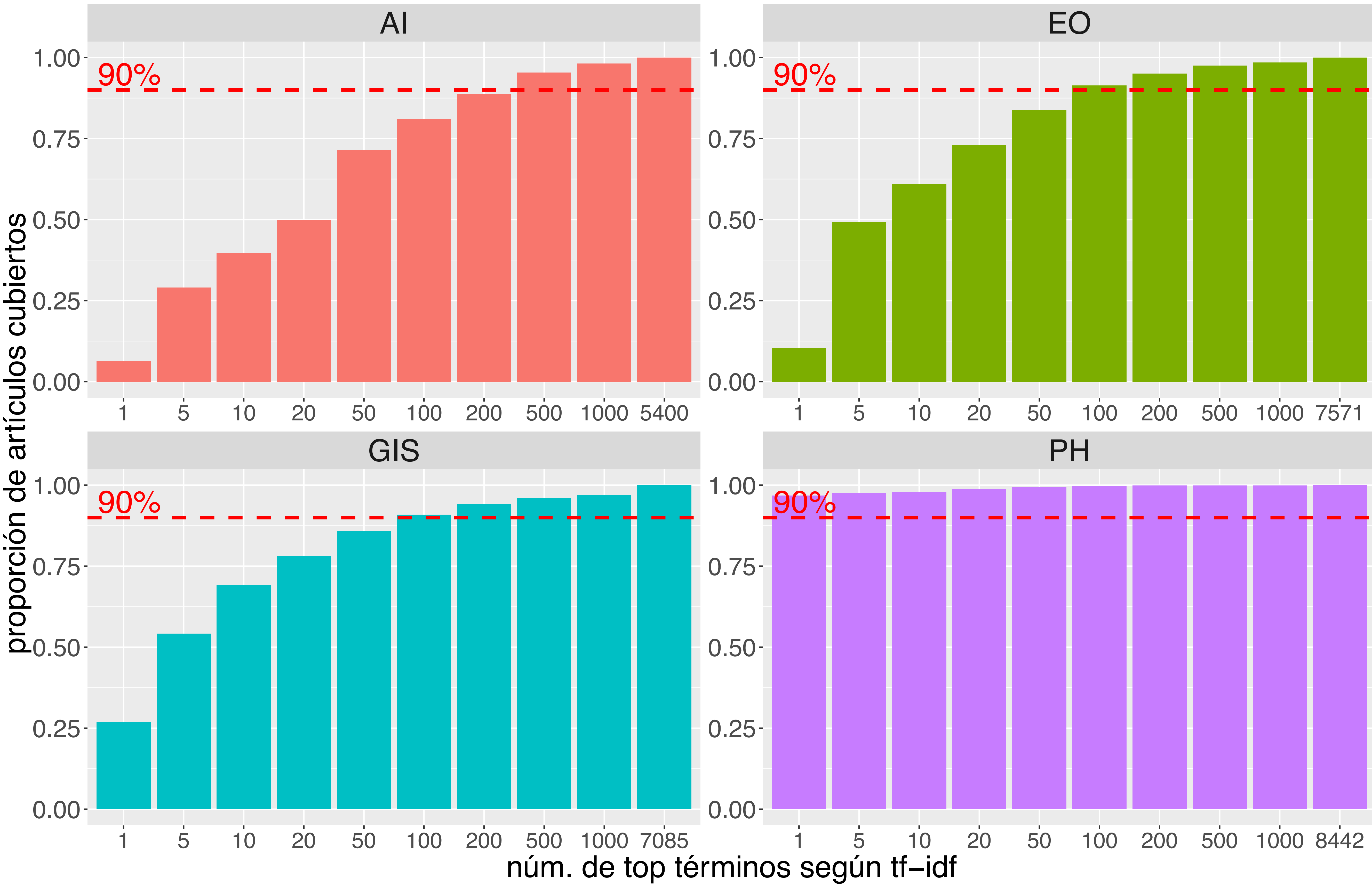
- Usamos TF-IDF para identificar los términos más representativos de cada grupo de artículos

- **tf**: prioriza palabras frecuentes en el documento
- **idf**: prioriza palabras que aparecen en pocos documentos
- **tf-idf**: prioriza palabras que aparecen mucho en el documento y poco en el resto de documentos

```
tfidf_tech_word <- text_word3 %>%
        count(word, tech) %>%
        bind_tf_idf(word, tech, n) %>%
        arrange(desc(tf_idf))
```


AI
convolutional
reinforcement
adversarial
videos
embeddings
learns
regret
facial
rnn
supervision
annotations
emph
inference
discriminative
trained
0.0000 0.0005 0.0010 0.0015 0.0020
EO
modis
sar
radar
landsat
land
vegetation
ndvi
soil
airborne
lst
snow
rmse
ocean
hyperspectral
elsevier
0.0000 0.0005 0.0010 0.0015
GIS
land
gis
geospatial
agricultural
wuhan
geographic
positioning
cartographic
cartography
geomatics
vgi
francis
taylor
deforestation
springer
0.000 0.001 0.002 0.003
PH
photon
photonic
quantum
crystal
dose
silicon
photons
inf
crystals
cavity
spect
doped
nanoparticles
tio
fiber
0.000 0.002 0.004 0.006
tf–idf

Cobertura del corpus



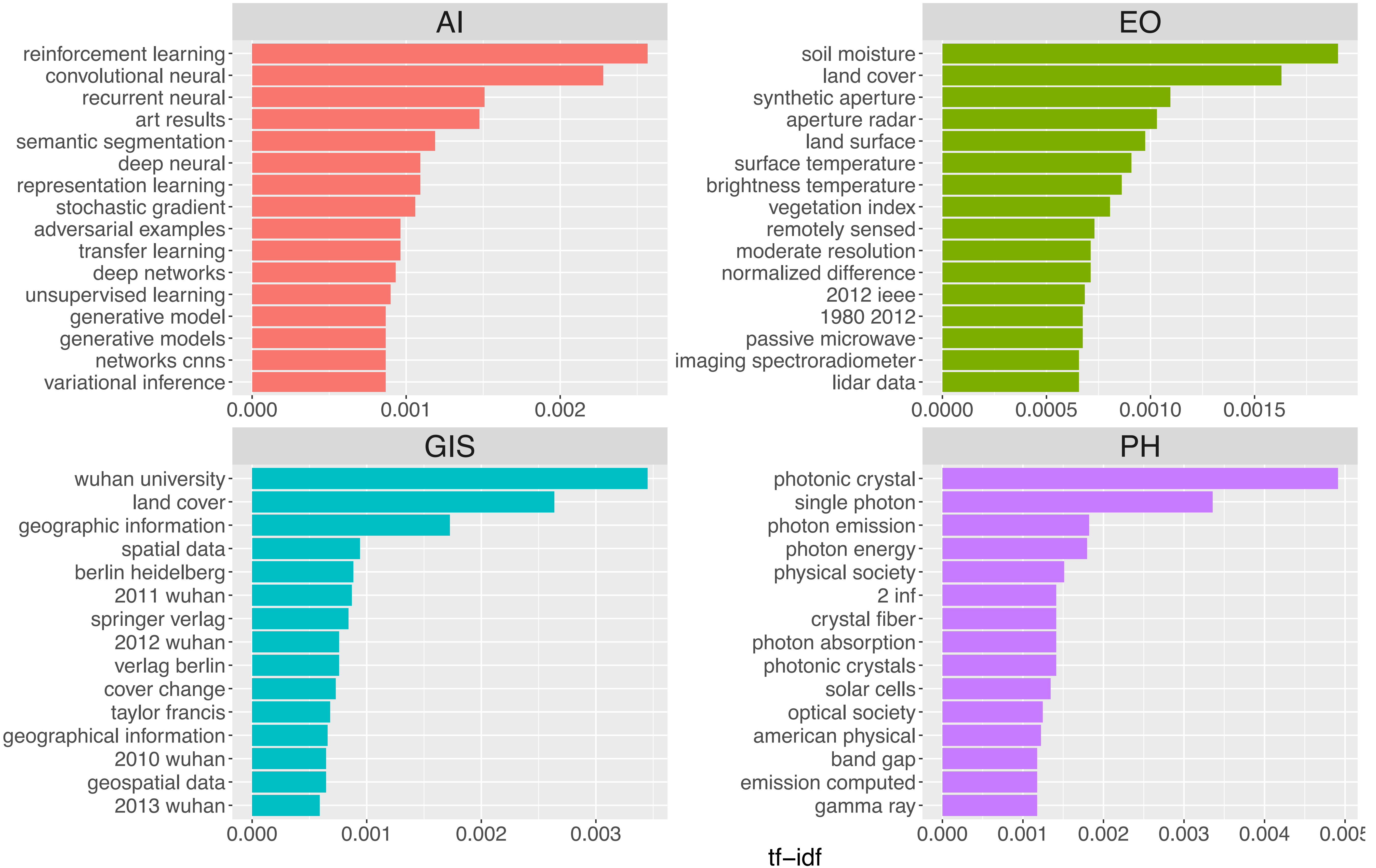
Bigramas

```
text_bigram <- data0_text %>%
        unnest_tokens(bigram, text, token = "ngrams", n=2) %>%
        filter(!is.na(bigram)) %>%
        separate(bigram, c("word1", "word2"), sep = " ") %>%
        mutate(word1 = str_extract(word1, "[a-z0-9']+"),
               word2 = str_extract(word2, "[a-z0-9']+")) %>%
        filter(!is.na(word1)) %>%
        filter(!is.na(word2)) %>%
        filter(!word1 %in% stop_words$word) %>%
        filter(!word2 %in% stop_words$word) %>%
        mutate(word1_stemm = (wordStem(word1)),
               word2_stemm = (wordStem(word2))) %>%
        filter(!is.na(word1_stemm)) %>%
        filter(!is.na(word2_stemm)) %>%
        unite(bigram_stemm, word1_stemm, word2_stemm, sep = " ") %>%
        unite(bigram, word1, word2, sep = ' ')
```

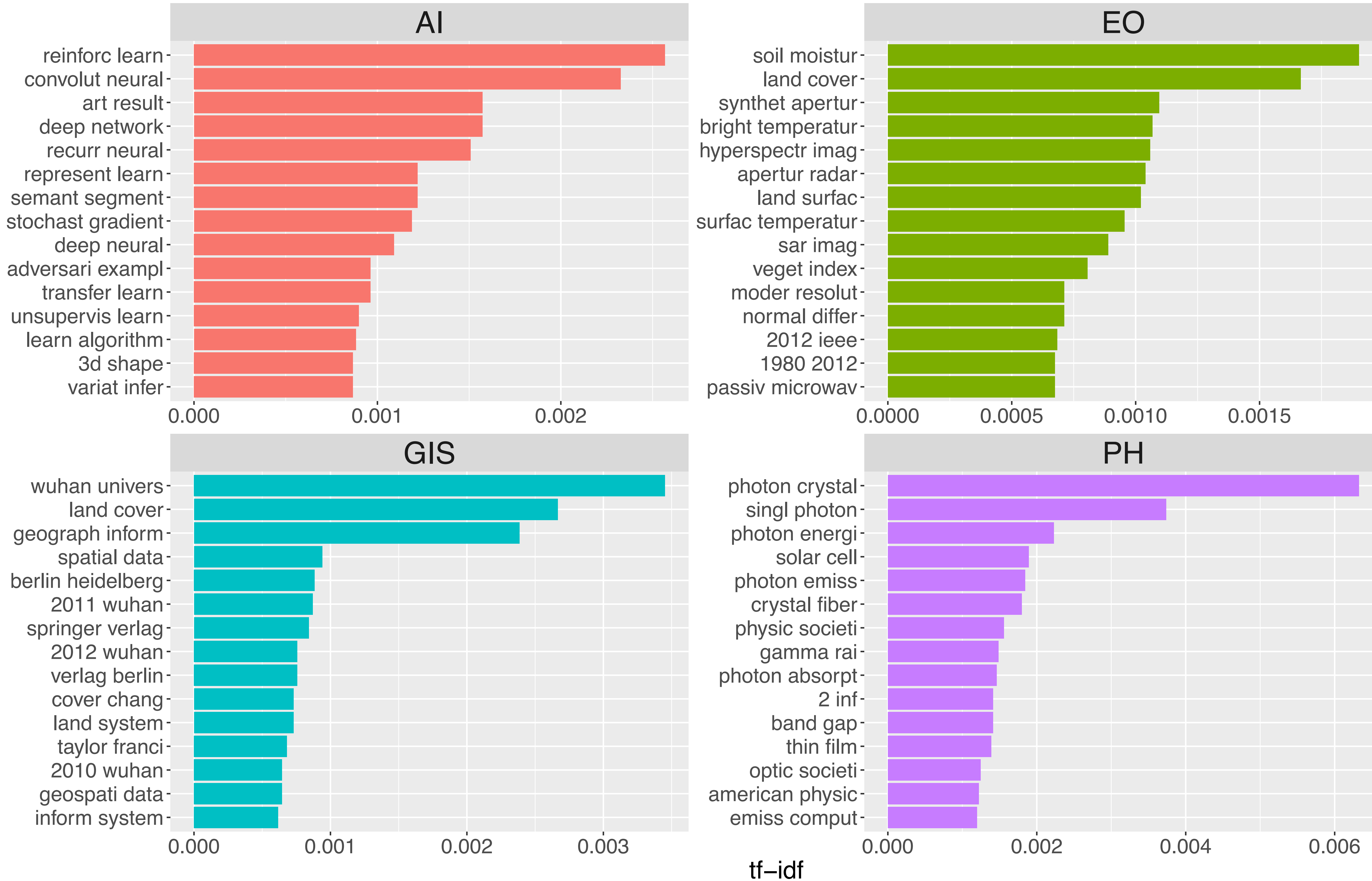
	bigram	n
	<chr>	<int>
1	remote sensing	587
2	land cover	362
3	neural networks	271
4	time series	219
5	experimental results	215
6	taylor francis	209
7	photonic crystal	205
8	soil moisture	205
9	data sets	198
10	neural network	198

… with 138,658 more rows

Bigramas representativos (tf-idf)

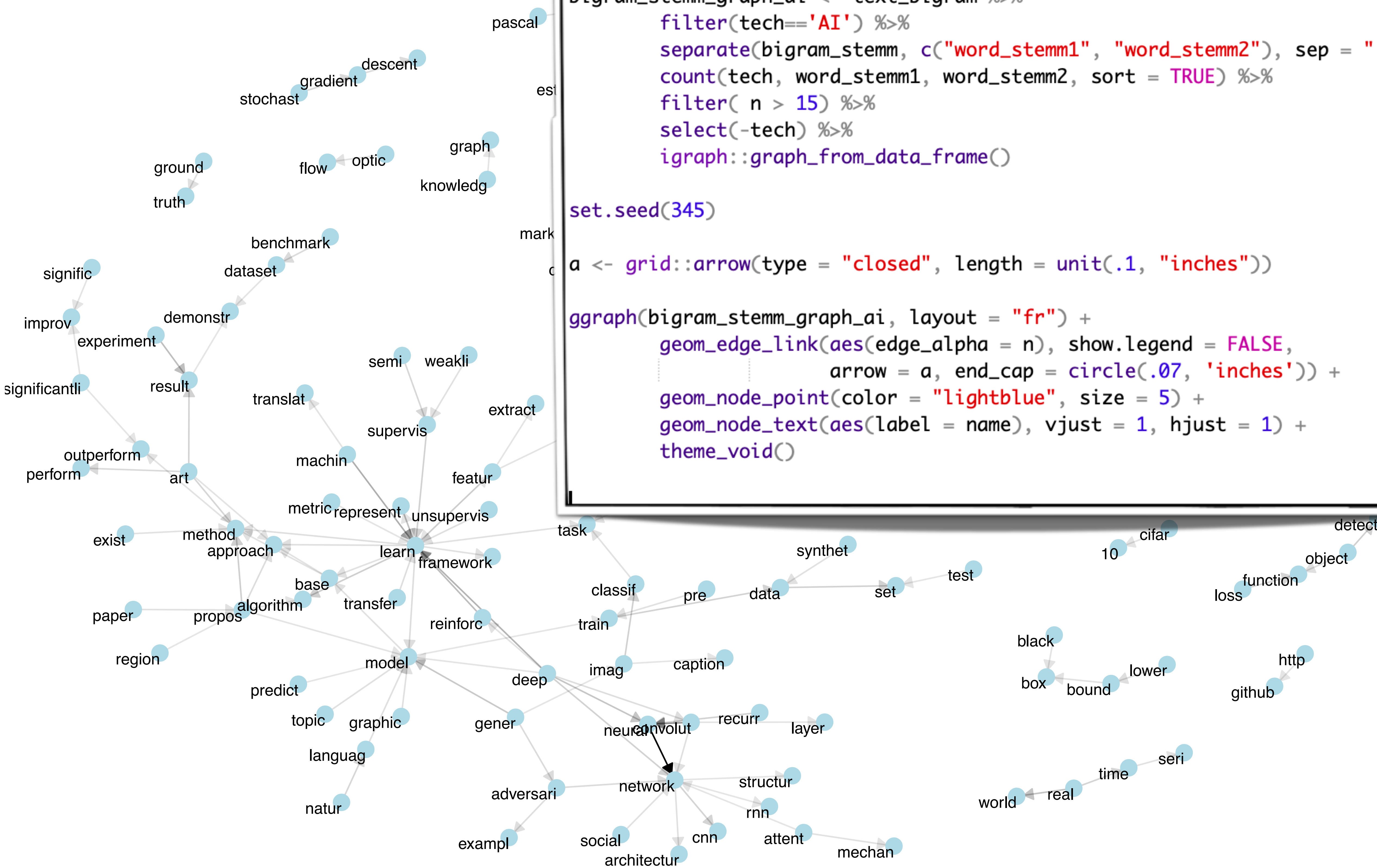


Bigramas repres. stemm (tf-idf)

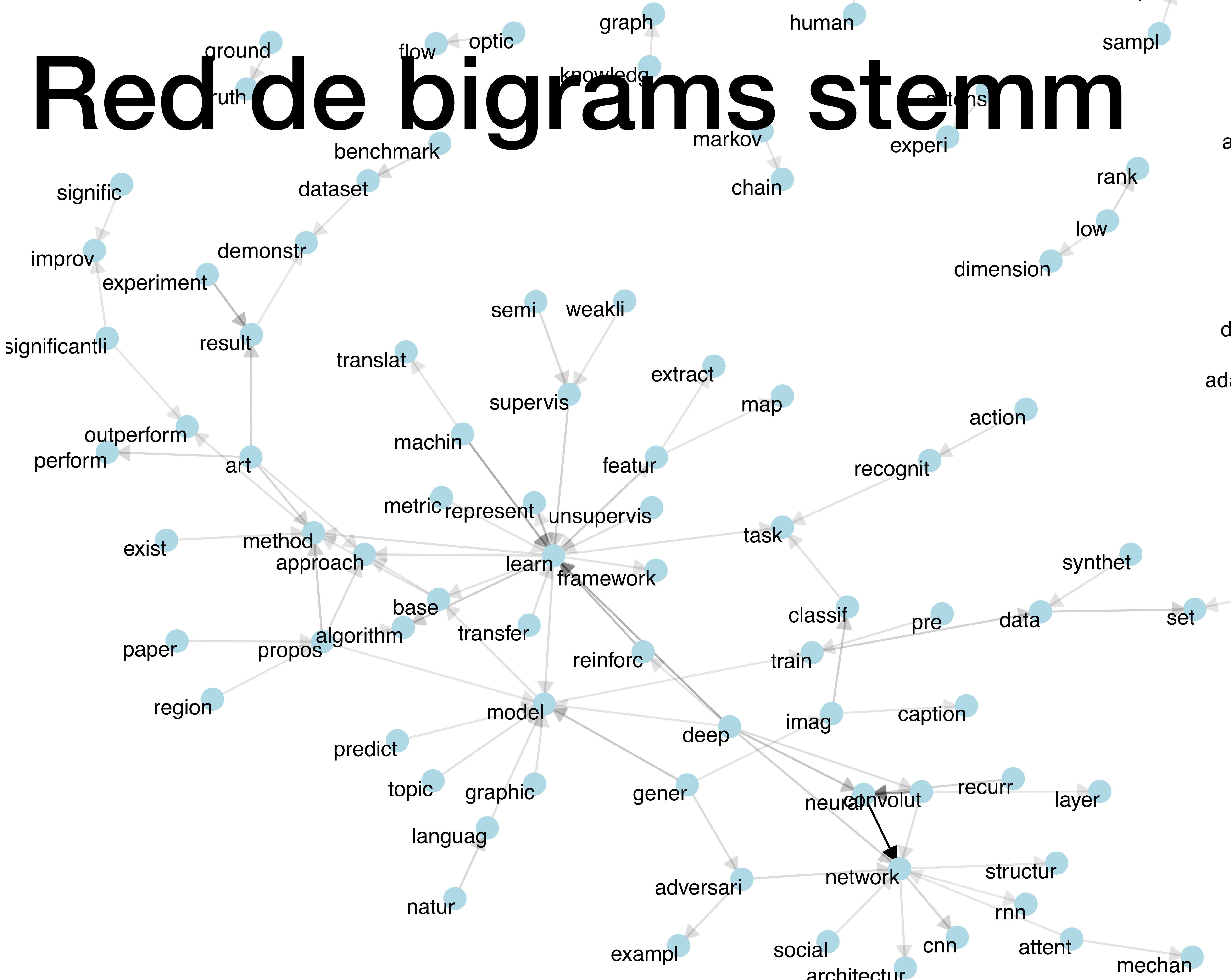


Red de bigrams stemm

```r
bigram_stemm_graph_ai <- text_bigram %>%
        filter(tech=='AI') %>%
        separate(bigram_stemm, c("word_stemm1", "word_stemm2"), sep = " ") %>%
        count(tech, word_stemm1, word_stemm2, sort = TRUE) %>%
        filter( n > 15) %>%
        select(-tech) %>%
        igraph::graph_from_data_frame()

set.seed(345)

a <- grid::arrow(type = "closed", length = unit(.1, "inches"))

ggraph(bigram_stemm_graph_ai, layout = "fr") +
        geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                       arrow = a, end_cap = circle(.07, 'inches')) +
        geom_node_point(color = "lightblue", size = 5) +
        geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
        theme_void()
```



Red de bigrams stemm



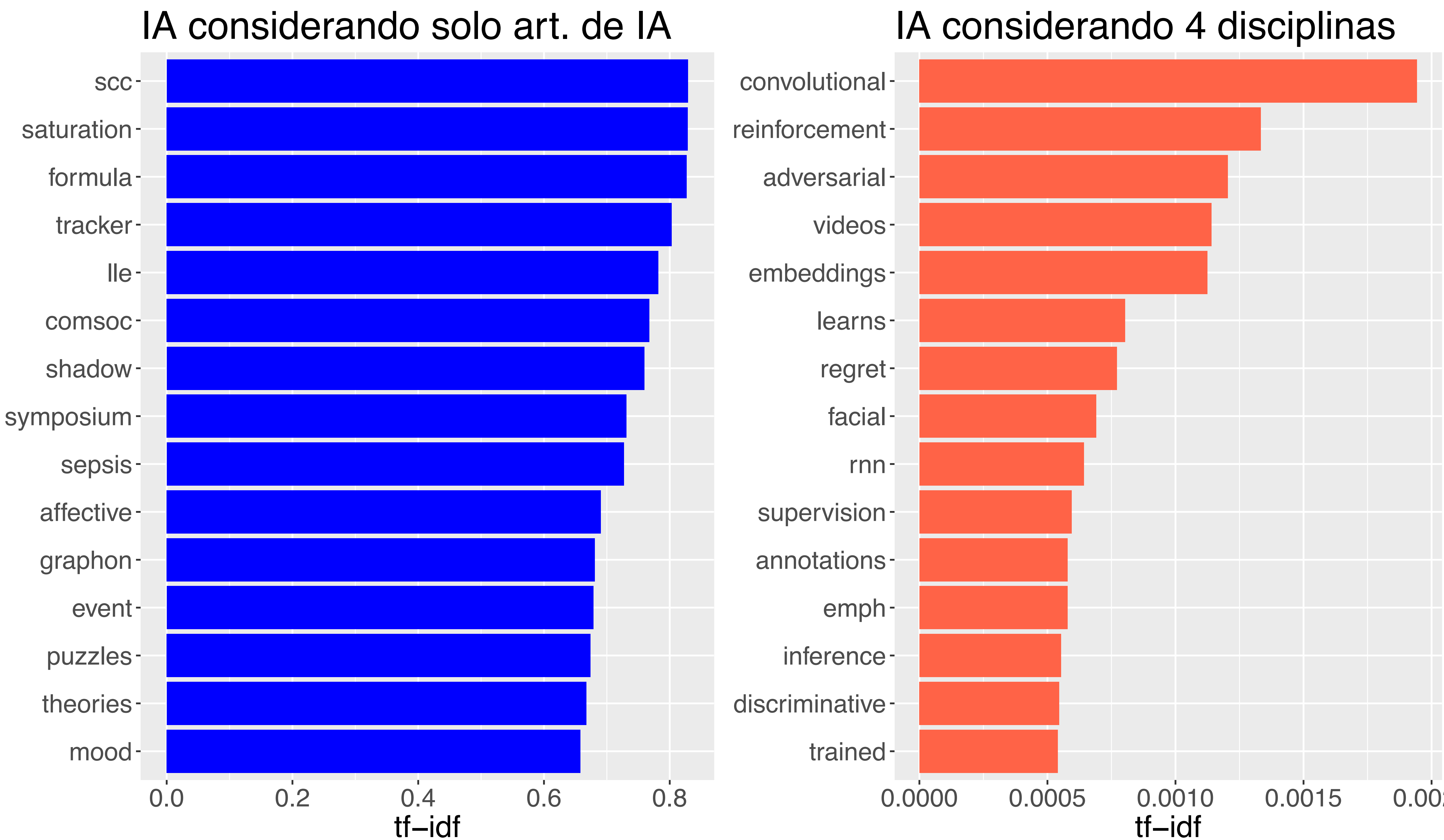
Pero, qué pasa si…?

- … **no** tenemos 4 grupos de artículos para comparar y queremos conocer los términos representativos de uno

- … o necesitamos conocer el **contenido de cada artículo** de manera resumida, para hacer clusters de artículos por tema (por ejemplo con *topic modelling*)?

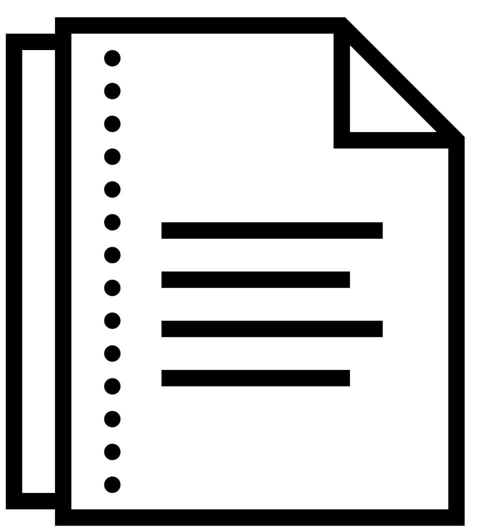
TF-IDF deja de funcionar -> Vamos a verlo con los 1000 artículos de Inteligencia artificial

Top 15 tf-idf en IA



Métodos basados en documento

- RAKE (*rapidraker* y *slowraker*)
- Co-occurencies (*udpipe*)



Extracción de palabras clave con RAKE

RAKE (Rapid Automatic Keyword Extraction):

- Algoritmo no supervisado para extracción de términos clave
- Independiente del dominio
- Analiza frecuencia de aparición de términos en el texto y co-ocurrencia con otros (grado del nodo en la red de términos)
- Identifica frases clave desde ngram =1 hasta 20+
- **Inconveniente**: las frases clave muy largas pierden el objetivo de resumir el texto en lo que entendemos como palabras clave

RAKE

INPUT: *text_clean_text*: vector con texto original con términos alfanum. sin stopwords

```
rake_list <- rapidrake(text_clean_text,
                       stop_pos = c("VB","VBD", "VBG", "VBN", "VBP", "VBZ"),
                       word_min_char = 3, stem = TRUE,
                       phrase_delims = "[,.?:;!]")

rake_df <- rbind_rakelist(rakelist = rake_list, doc_id = text_clean_original$article_id)
```

doc_id	keyword	freq	score	stem
AI_1	performance comparable omp priori knowledge spars...	1	52.849998	perform compar omp priori knowledg sparsiti nois st...
AI_1	analytical results numerical simulations real synthetic...	1	49.000000	analyt result numer simul real synthet data
AI_1	priori knowledge sparsity regression vector noise stat...	1	47.099998	priori knowledg sparsiti regress vector nois statist
AI_1	signal noise statistics oblivious orthogonal matching ...	1	44.099998	signal nois statist oblivi orthogon match pursuit
AI_1	omp priori knowledge sparsity noise statistics	1	39.349998	omp priori knowledg sparsiti nois statist
AI_1	sparse dimensional vectors linear regression models	1	37.000000	spars dimension vector linear regress model
AI_1	finite sample sample support recovery guarantees	1	36.000000	finit sampl sampl support recoveri guarante
AI_1	statistics rarely priori difficult estimate	1	28.100000	statist rare priori difficult estim
AI_1	pursuit omp widely algorithm	1	18.750000	pursuit omp wide algorithm
AI_1	optimal performance omp	1	13.750000	optim perform omp
AI_1	orthogonal	1	4.000000	orthogon
AI_1	residual ratio	1	4.000000	residu ratio
AI_1	paper	1	1.000000	paper
AI_1	rrt	1	1.000000	rrt
AI_1	technique	1	1.000000	techniqu
AI_2	theoretical linear speedup respect sequential version ...	1	159.0000...	theoret linear speedup respect sequenti version assu...
AI_2	parallel asynchronous variants stochastic gradient de...	1	55.799999	parallel asynchron variant stochast gradient descent

Extracción de palabras clave con UDPIPE

udpipe: versión para R de UDPipe de C++

- "UDPipe provides language-agnostic **tokenization**, **tagging**, **lemmatization** and **dependency parsing** of raw text"
- Tiene 3 funciones para detectar palabras clave: RAKE, Point-Wise Mutual Information Collocation, Parts of Speech phrase sequence detection. Pero no funcionan muy bien…
- Otra opción: usar co-occurrence. **Inconveniente**: solo detecta bigramas

UDPIPE

INPUT: *text_clean_original*: data frame original con términos alfanum. sin stopwords

```
# -- Download model (inly the first time) --
ud_model <- udpipe_download_model(language = "english")
ud_model <- udpipe_load_model(ud_model$file_model)

# -- Anotate data --
# pre-Cleaned text
text_clean_udpipe <- udpipe_annotate(ud_model, x = text_clean_original$text,
                                     doc_id = text_clean_original$article_id)
text_udpipe <- as.data.table(text_clean_udpipe)

## bigrams from co-occurrence
text_terms_2gram <- text_udpipe[,cooccurrence(x = lemma,
                                              relevant = upos %in% c("NOUN", "ADJ"),
                                              skipgram = 1),
                                  by=doc_id]
text_terms_2gram[,cooc:=.N,by=list(term1,term2)]
text_terms_2gram <- text_terms_2gram[term1!=term2 & cooc>1,]
text_terms_2gram[,term:= paste(term1, term2, sep = " ")]
text_terms_udpipe <- text_terms_2gram
```

RAKE vs UDPIPE

text	keywords.RAKE	keywords. UDPIPE
Orthogonal matching pursuit (OMP) is a widely used algorithm for recovering **sparse high dimensional vectors** in **linear regression models**. The optimal performance of OMP requires **a priori knowledge** of either the sparsity of regression vector or noise statistics. Both these statistics are rarely known a priori and are very difficult to estimate. In this paper, we present a novel technique called **residual ratio thresholding** (RRT) to operate OMP without any a priori knowledge of **sparsity** and **noise statistics** and establish finite sample and large sample support recovery guarantees for the same. Both analytical results and numerical simulations in real and synthetic data sets indicate that RRT has a performance comparable to OMP with a priori knowledge of sparsity and noise statistics. Signal and Noise Statistics Oblivious Orthogonal Matching Pursuit	**performance comparable omp priori knowledge sparsity noise statistics** -- analytical results numerical simulations real synthetic data -- priori knowledge sparsity regression vector noise statistics -- signal noise statistics oblivious orthogonal matching pursuit -- omp priori knowledge sparsity noise statistics -- sparse dimensional vectors linear regression models -- finite sample sample support recovery guarantees -- statistics rarely priori difficult estimate -- pursuit omp widely algorithm -- optimal performance omp -- orthogonal -- residual ratio -- paper -- rrt -- technique	priori knowledge -- dimensional vector -- real synthetic -- synthetic data -- sparse vector -- result simulation -- real data

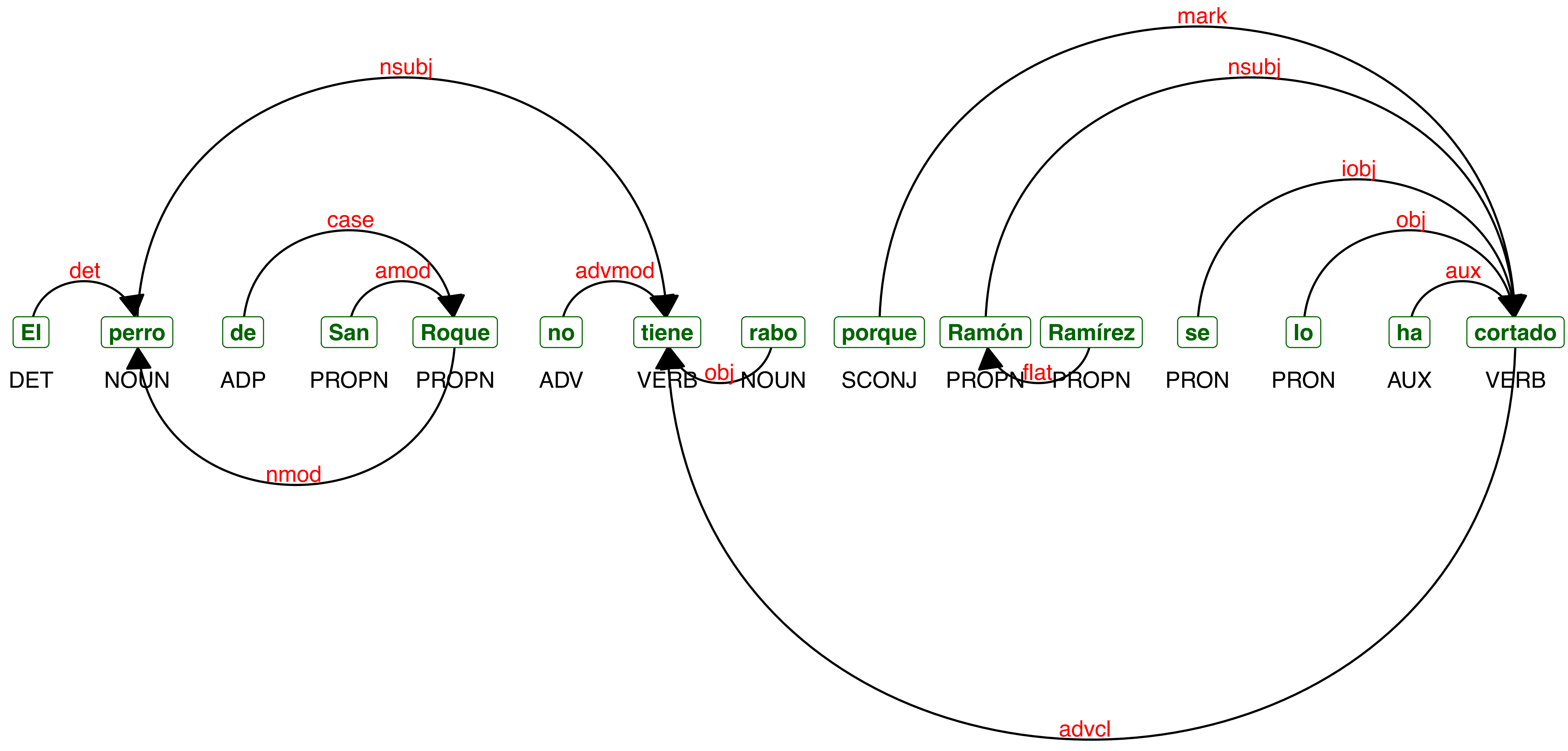
Salida de *udpipe_annotate*

doc_id	paragraph_id	sentence_id	token_id	token	lemma	upos	xpos	feats	head_token_id	dep_rel	deps	misc
AI_1	1	1	1	Orthogonal	Orthogonal	ADJ	JJ	Degree=Pos	3	amod		
AI_1	1	1	2	matching	matching	NOUN	NN	Number=Sing	3	compound		
AI_1	1	1	3	pursuit	pursuit	NOUN	NN	Number=Sing	11	nsubj		
AI_1	1	1	4	(	(	PUNCT	-LRB-		5	punct		SpaceAfter=No
AI_1	1	1	5	OMP	OMP	PROPN	NNP	Number=Sing	3	appos		SpaceAfter=No
AI_1	1	1	6	)	)	PUNCT	-RRB-		5	punct		
AI_1	1	1	7	is	be	AUX	VBZ	Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin	11	cop		
AI_1	1	1	8	a	a	DET	DT	Definite=Ind|PronType=Art	11	det		
AI_1	1	1	9	widely	widely	ADV	RB		10	advmod		
AI_1	1	1	10	used	use	VERB	VBN	Tense=Past|VerbForm=Part				
AI_1	1	1	11	algorithm	algorithm	NOUN	NN	Number=Sing				
AI_1	1	1	12	for	for	SCONJ	IN					
AI_1	1	1	13	recovering	recovere	VERB	VBG	VerbForm=Ger				
AI_1	1	1	14	sparse	sparse	ADV	RB					
AI_1	1	1	15	high	high	ADJ	JJ	Degree=Pos				
AI_1	1	1	16	dimensional	dimensiona	ADJ	JJ	Degree=Pos				
AI_1	1	1	17	vectors	vector	NOUN	NNS	Number=Plur				
AI_1	1	1	18	in	in	ADP	IN					
AI_1	1	1	19	linear	linear	ADJ	JJ	Degree=Pos				
AI_1	1	1	20	regression	regression	NOUN	NN	Number=Sing				
AI_1	1	1	21	models	model	NOUN	NNS	Number=Plur				SpaceAfter=No
AI_1	1	1	22	.	.	PUNCT	.					

- ADJ: adjective
- ADP: adposition
- ADV: adverb
- AUX: auxiliary
- CCONJ: coordinating conjunction
- DET: determiner
- INTJ: interjection
- NOUN: noun
- NUM: numeral
- PART: particle
- PRON: pronoun
- PROPN: proper noun
- PUNCT: punctuation
- SCONJ: subordinating conjunction
- SYM: symbol
- VERB: verb
- X: other

https://universaldependencies.org/format.html

Gráfico de *dependency parsing*



https://www.r-bloggers.com/dependency-parsing-with-udpipe/

Enlaces útiles

- Paquetes y Frameworks de R para PLN: https://cran.r-project.org/web/views/NaturalLanguageProcessing.html

- Tutorial text mining con ***tidytext***: https://www.tidytextmining.com/index.html

- ***udpipe R***: https://bnosac.github.io/udpipe/en/

  UDPipe (C++):
  - Documentación https://ufal.mff.cuni.cz/udpipe
  - Udpipe Demo URL: https://lindat.mff.cuni.cz/services/udpipe/

- **RAKE** (Automatic keyword extraction from individual documents): http://media.wiley.com/product_data/excerpt/22/04707498/0470749822.pdf

- Otros paquetes muy usados: quanteda, tm…

linkedin.com/in/montserrat-lópez-cobo-25069ab1
montse.lopezcobo@gmail.com