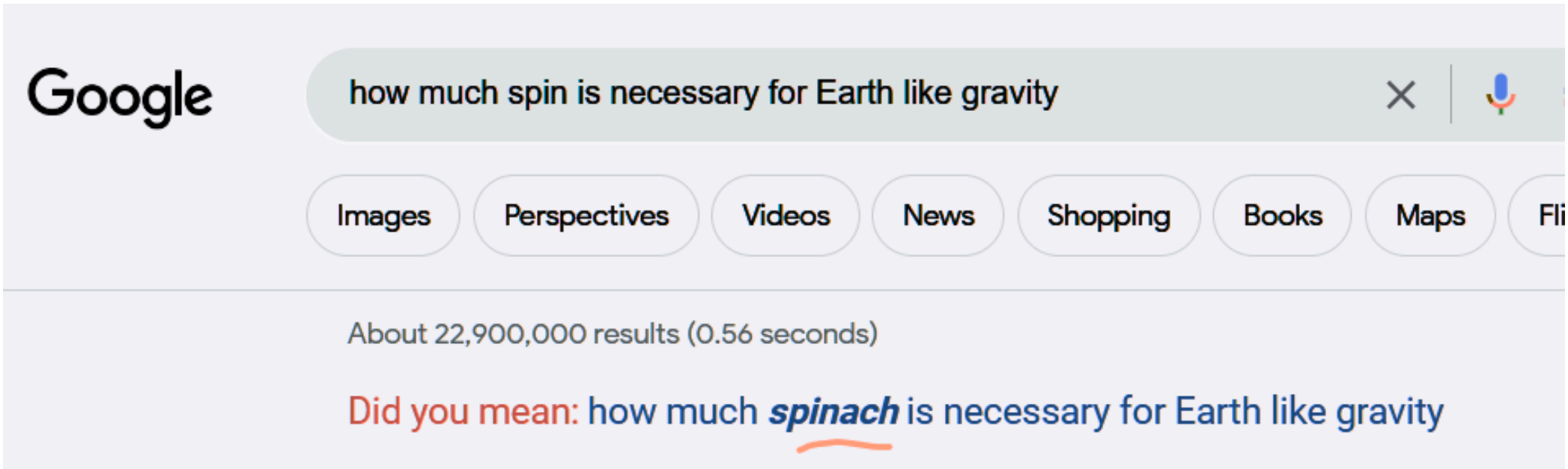
Query spelling correction using pre-trained GloVe word embeddings

CharacterBERT and Self-Teaching for Improving the Robustness of Dense Retrievers on Queries with Typos[1]

Zhuang and Zuccon, 2022

Queries	Method	MS MARCO		DL-typo		
		MRR@10	R@1000	nDCG@10	MRR	MAP
Without Typos	a) pyspellchecker -> StandardBERT-DR	.276	.888	.700	.811	.550
	b) MSspellchecker -> StandardBERT-DR	.324^{ac}	**.951**^{ac}	**.719**	.833	**.563**
	c) pyspellchecker -> CharacterBERT-DR	.279	.887	.703	.824	.527
	d) MSspellchecker -> CharacterBERT-DR	**.326**^{ac}	.948^{ac}	.715	**.855**	.538
	e) CharacterBERT-DR+ST	.325^{ac}	.950^{ac}	.706	.793	.539
With Typos	d) pyspellchecker -> StandardBERT-DR	.231	.819	.475	.562	.340
	f) MSspellchecker -> StandardBERT-DR	.303^{dgi}	.920^{dgi}	**.716**^{dgi}	.833^{dgi}	**.559**^{dgi}
	g) pyspellchecker -> CharacterBERT-DR	.234	.821	.462	.573	.339
	h) MSspellchecker -> CharacterBERT-DR	**.305**^{dgi}	.930^{dgi}	.714^{dgi}	**.855**^{dgi}	.539^{dgi}
	i) CharacterBERT-DR+ST	.263^{dg}	.894^{dg}	.473	.615	.348

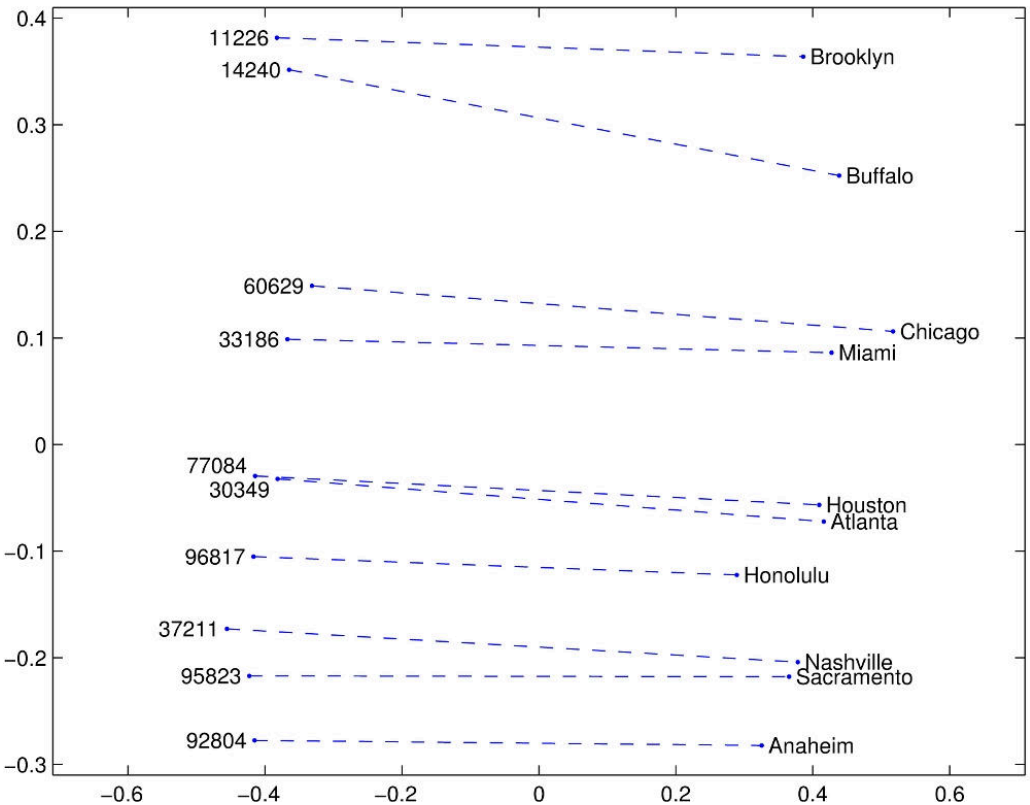
Table 5: Comparison between CharacterBERT-DR+ST and pipelines that involve spell-checkers. Methods statistically significantly better ($p < 0.05$) than others are indicated by superscripts.

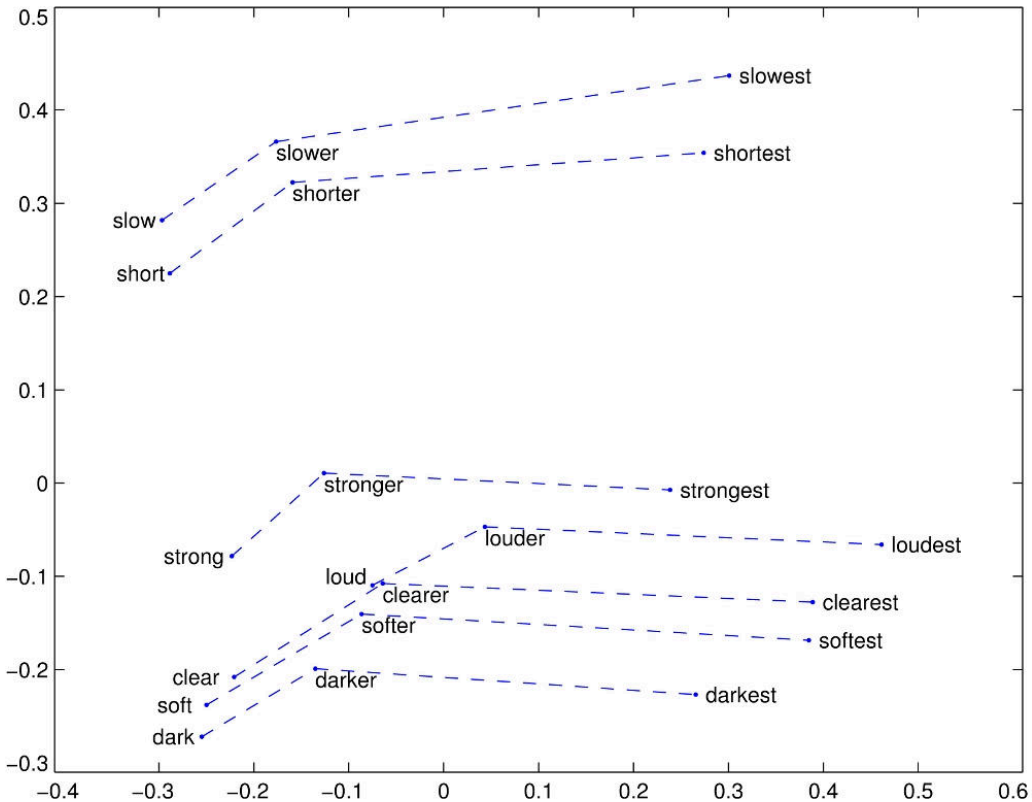


Greg Leuning, https://nerdculture.de/@booksthatgoboom/111710838850653630

GloVe: Global Vectors for Word Representation[2]

Pennington, Socher and Manning, 2014

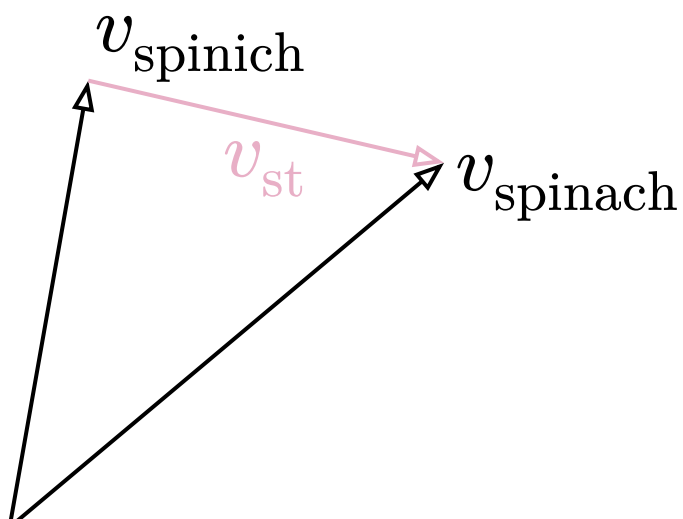


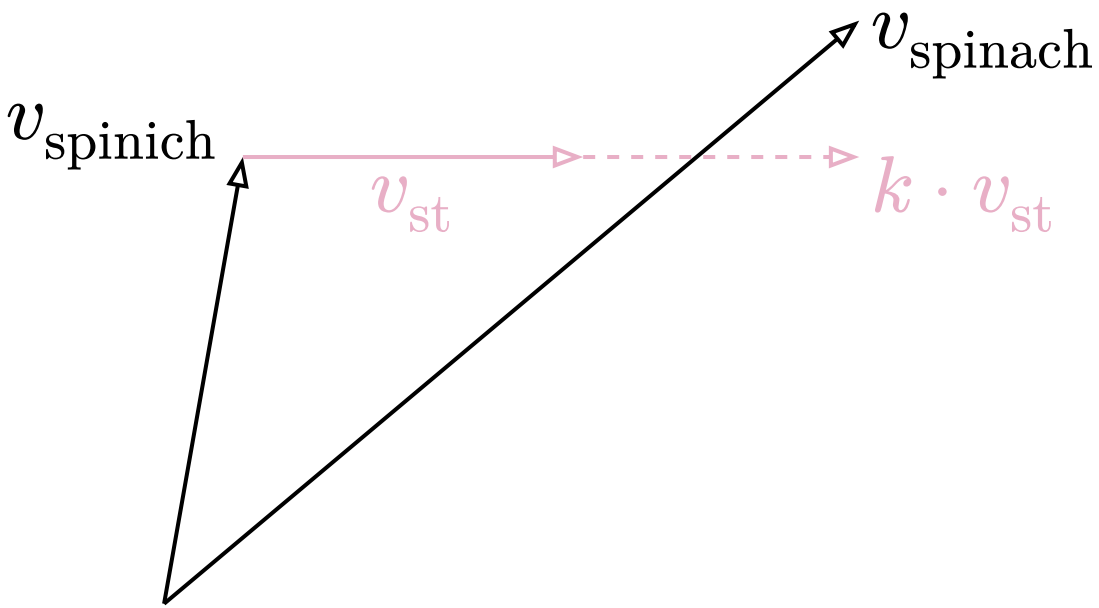


glove.840B.300d.txt: 2,196,017 words, 2.03 GB

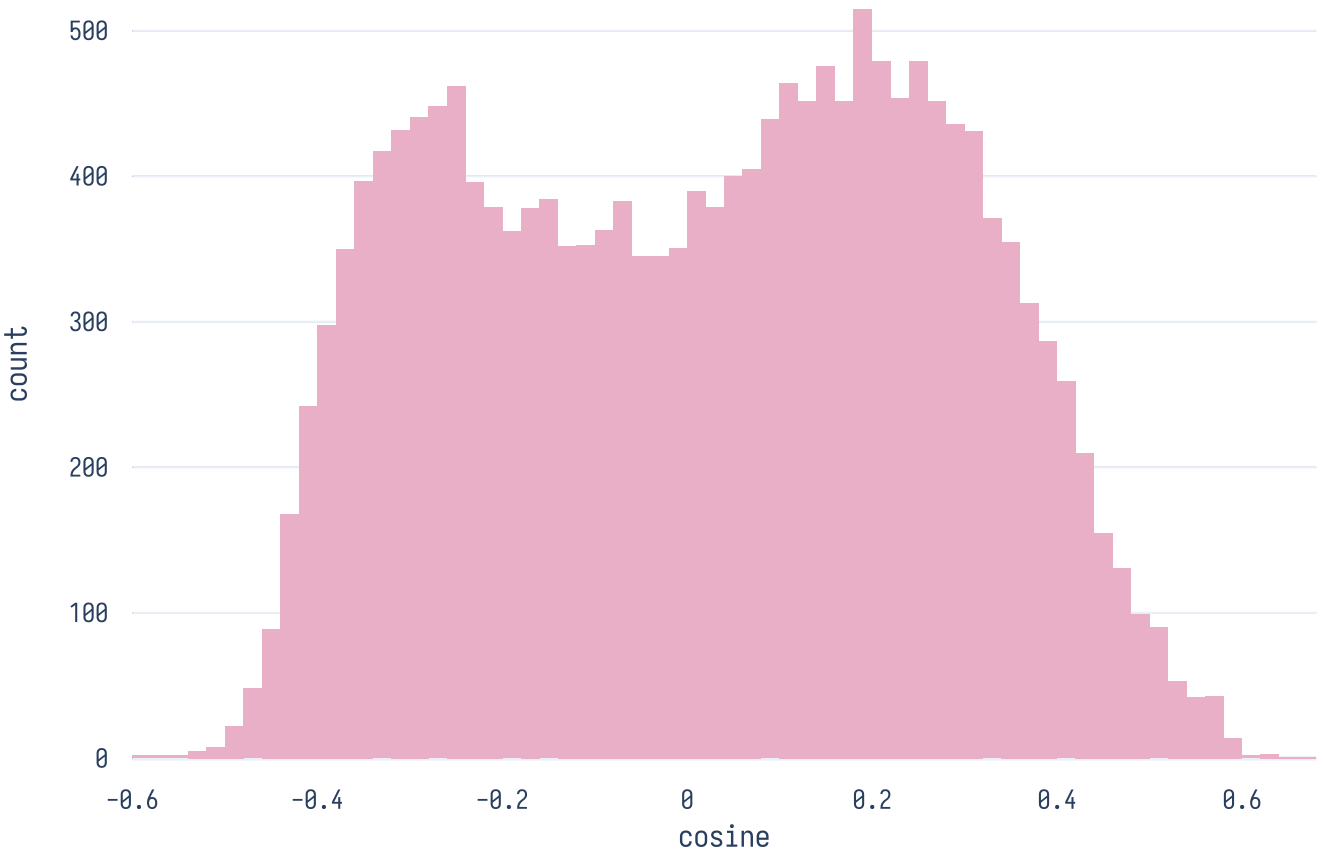
A simple spell checker built from word vectors[3]

Rushton, 2018

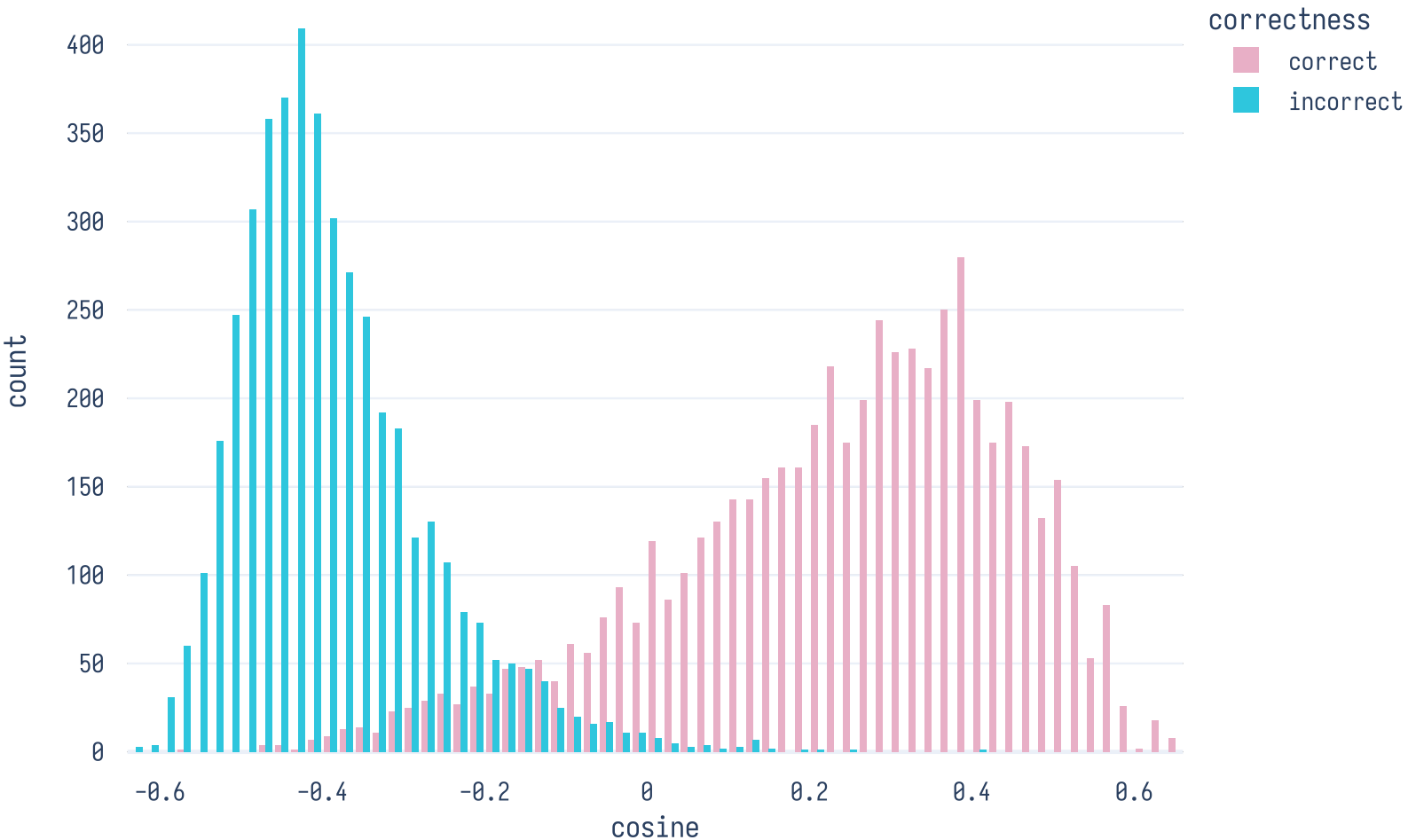


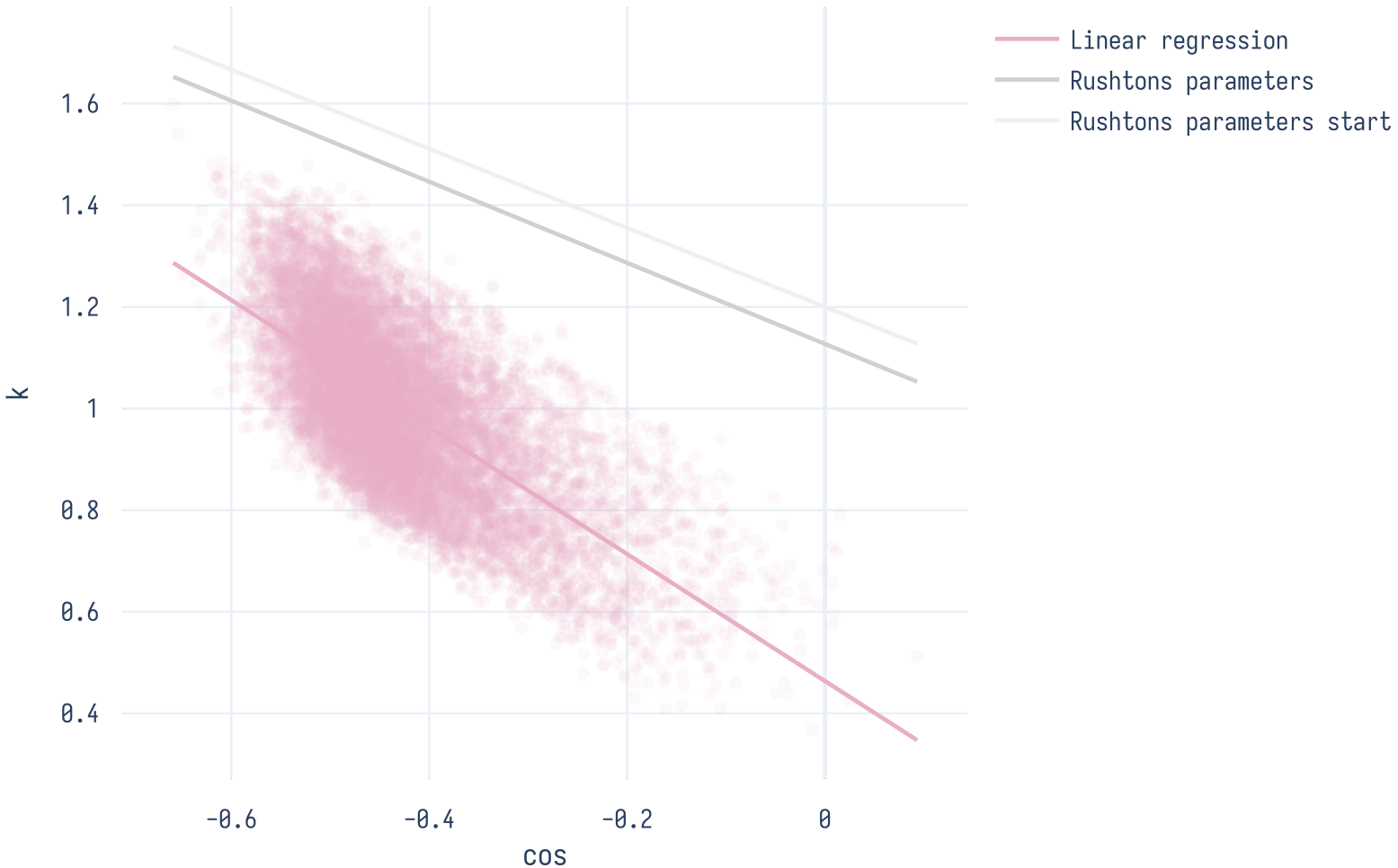


$$k = \frac{\langle v_{\text{spinach}} - v_{\text{spinich}} \mid v_{\text{st}} \rangle}{\langle v_{\text{st}} \mid v_{\text{st}} \rangle}$$



\ neads
\ salado
\ baladi
\ endicott
\ ashby
\ accessoires
\ artista
\ copperfield
\ tradewinds
\ dts
\ sportfishing
| cbr
| honolulu
| gagging
| fireman
| dis
| playlists
| qualifier
| oz
/ mats
/ messenger
/ pine
/ deceased
/ adolescent
/ mounted
/ files


correctness
correct
incorrect
count
400
350
300
250
200
150
100
50
0
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
cosine


Linear regression
Rushtons parameters
Rushtons parameters start
k
1.6
1.4
1.2
1
0.8
0.6
0.4
-0.6
-0.4
-0.2
0
cos

Results

model	p1	num_correct	num_total	c2i	i2c
Bounds replace word	0.9613400885480853	21062	21909	0	1898
Google2017	0.9283399516180565	20339	21909	172	1347
Bounds in glove	0.912136564882012	19984	21909	728	1548
Rushton-re	0.8919622073120635	19542	21909	62	440
Rushton	0.890958053767858	19520	21909	52	408
Wikipedia	0.8789538545803095	19257	21909	29	122
Bing2017	0.8763065406910402	19199	21909	230	265
Simplified	0.8761239673193665	19195	21909	1	32
Baseline	0.874709023688895	19164	21909	0	0
Google2023	0.8718334930850336	19101	21909	322	259
Order neighborhood	0.8625222511296727	18897	21909	874	607
Pyspellchecker	0.5523757359989045	12102	21909	7062	0

Examples

Questions:

- more recent word embeddings with one vector per word
- intuition for vector arithmetic on normalized vectors
- adding a context aware model (BERT?)

https://git.webis.de/code-teaching/theses/thesis-lahmann

Bibliography

[1] S. Zhuang and G. Zuccon, “CharacterBERT and Self-Teaching for Improving the Robustness of Dense Retrievers on Queries with Typos”. 2022. [Online]. Available: http://arxiv.org/abs/2204.00716v2

[2] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation”, [Online]. Available: https://aclanthology.org/D14-1162

[3] Ed Rushton, “A simple spell checker built from word vectors”. May 24, 2018. [Online]. Available: https://edrushton.medium.com/a-simple-spell-checker-built-from-word-vectors-9f28452b6f26