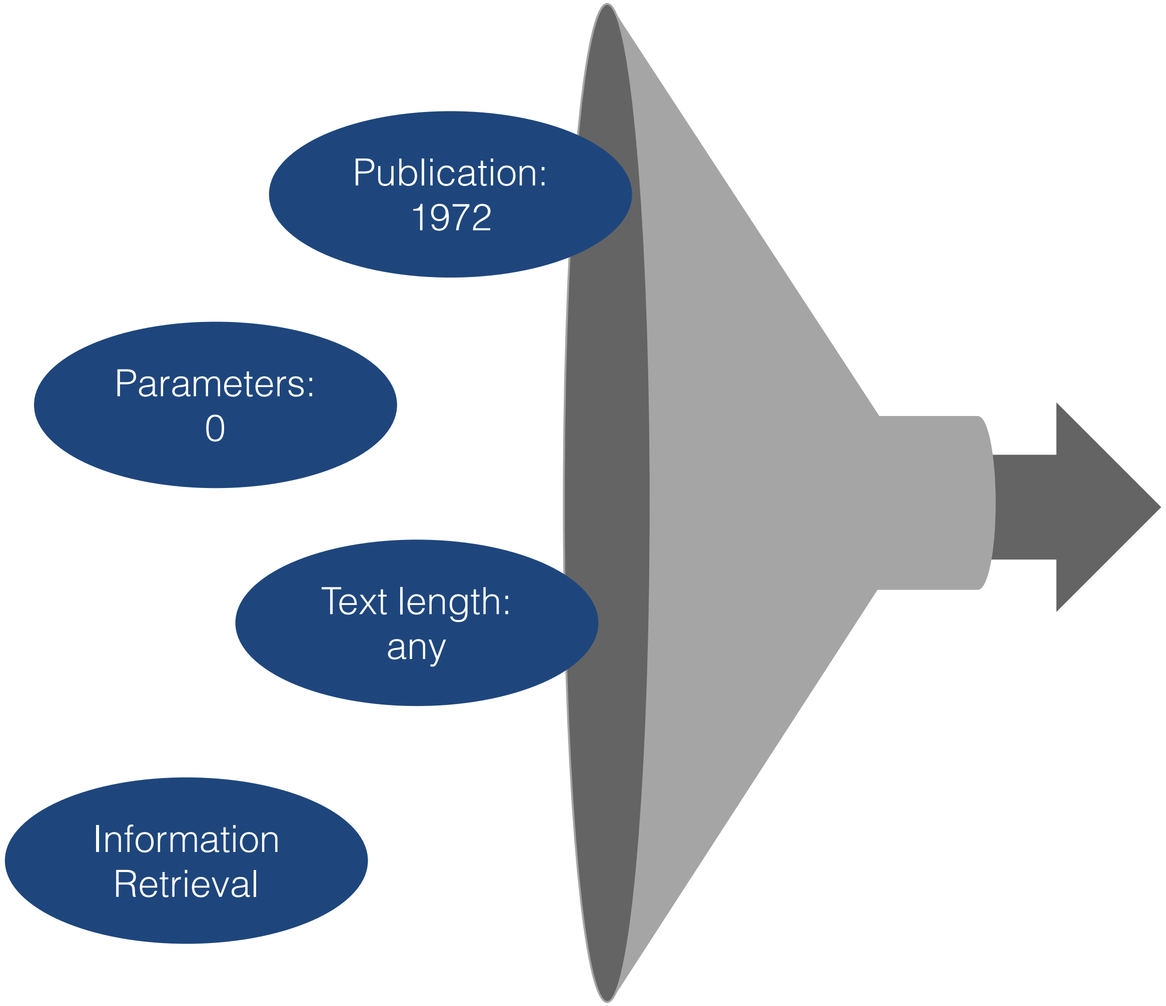
# The state of document embedding research

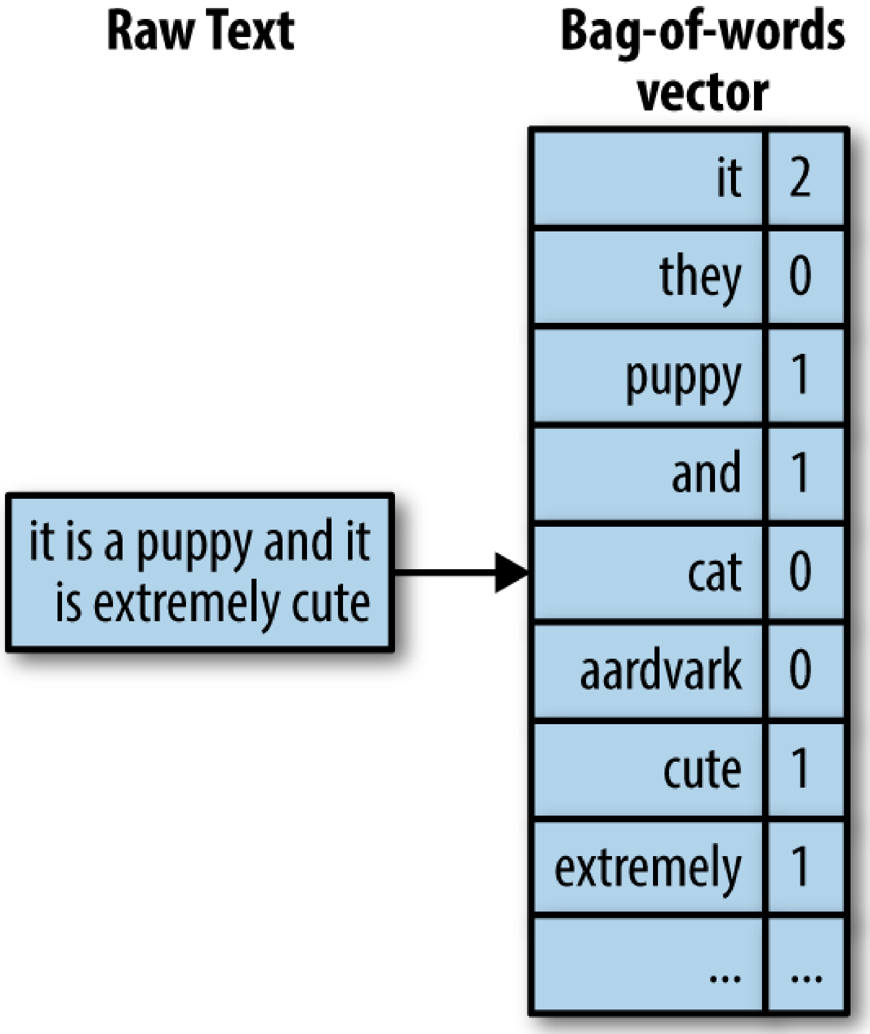
A short story on vectorizing human knowledge

Tim Korjakow

# BOW is the simplest of the presented techniques

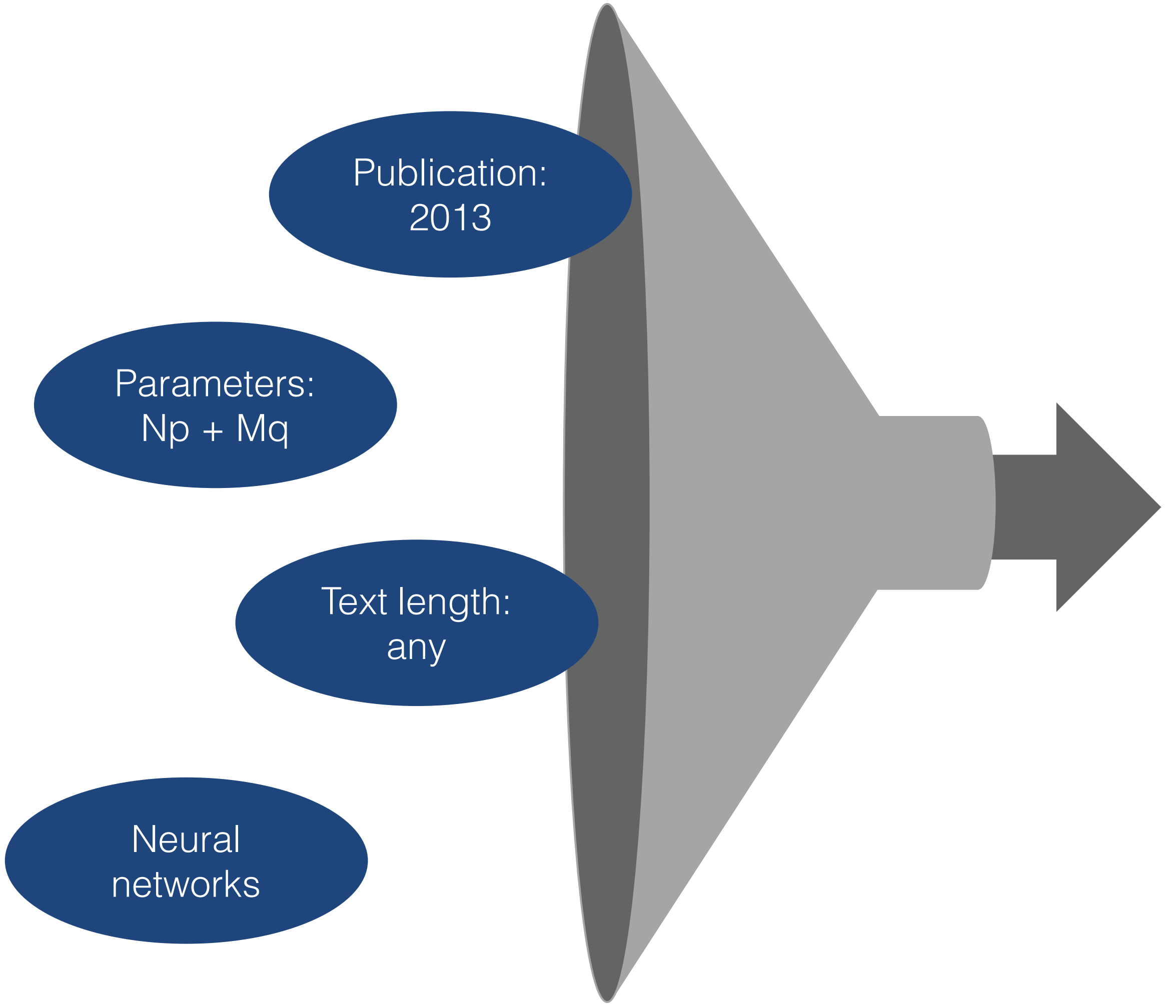
## TF-IDF BAG-OF-WORDS

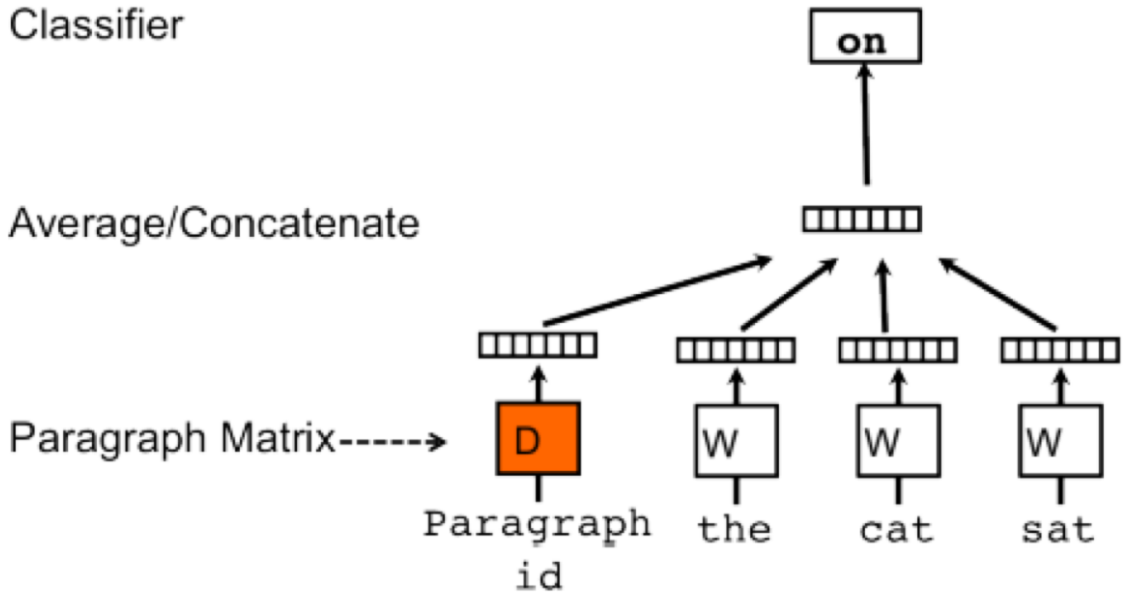


# Doc2Vec is an intellectual successor to the well-known Word2Vec model

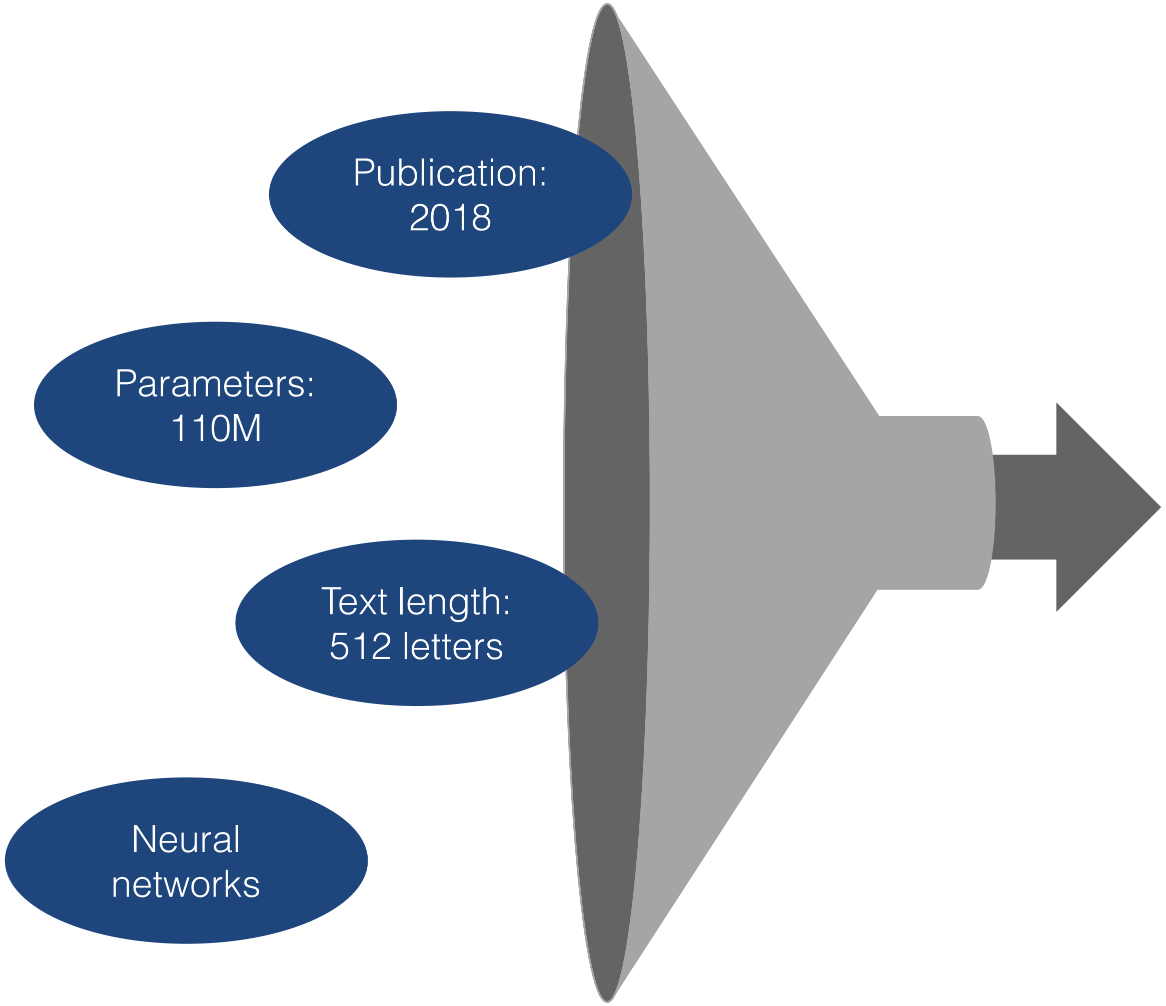
## DOC2VEC

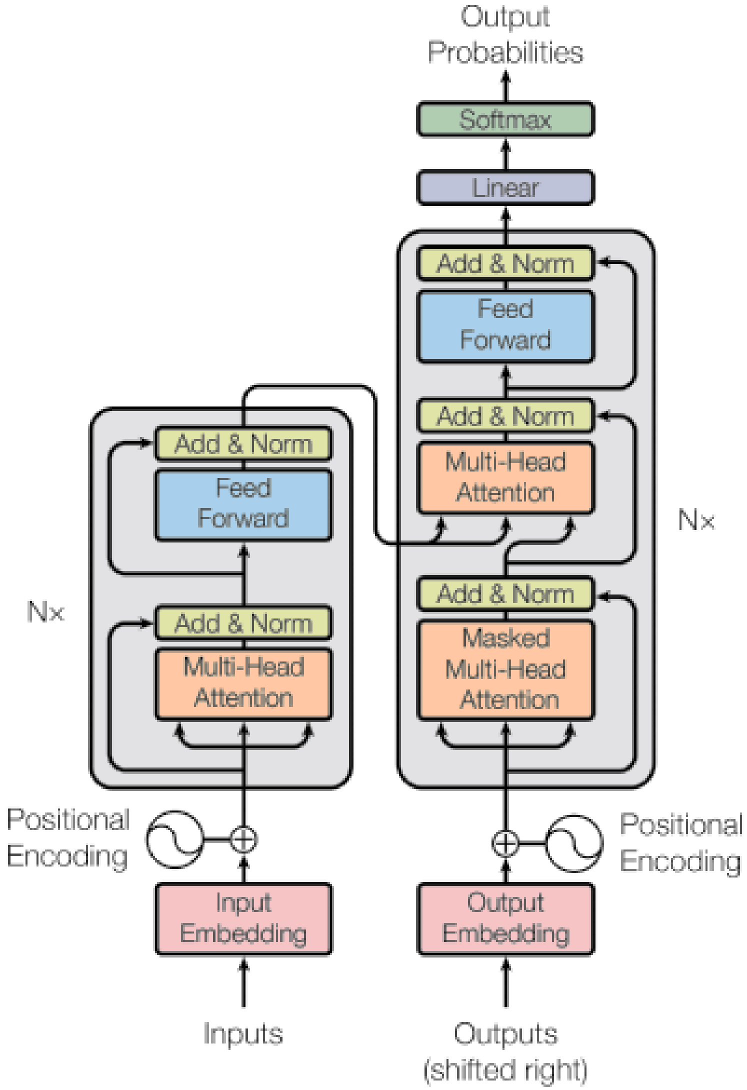


# BERT gains ist predictive power by a huge training corpus

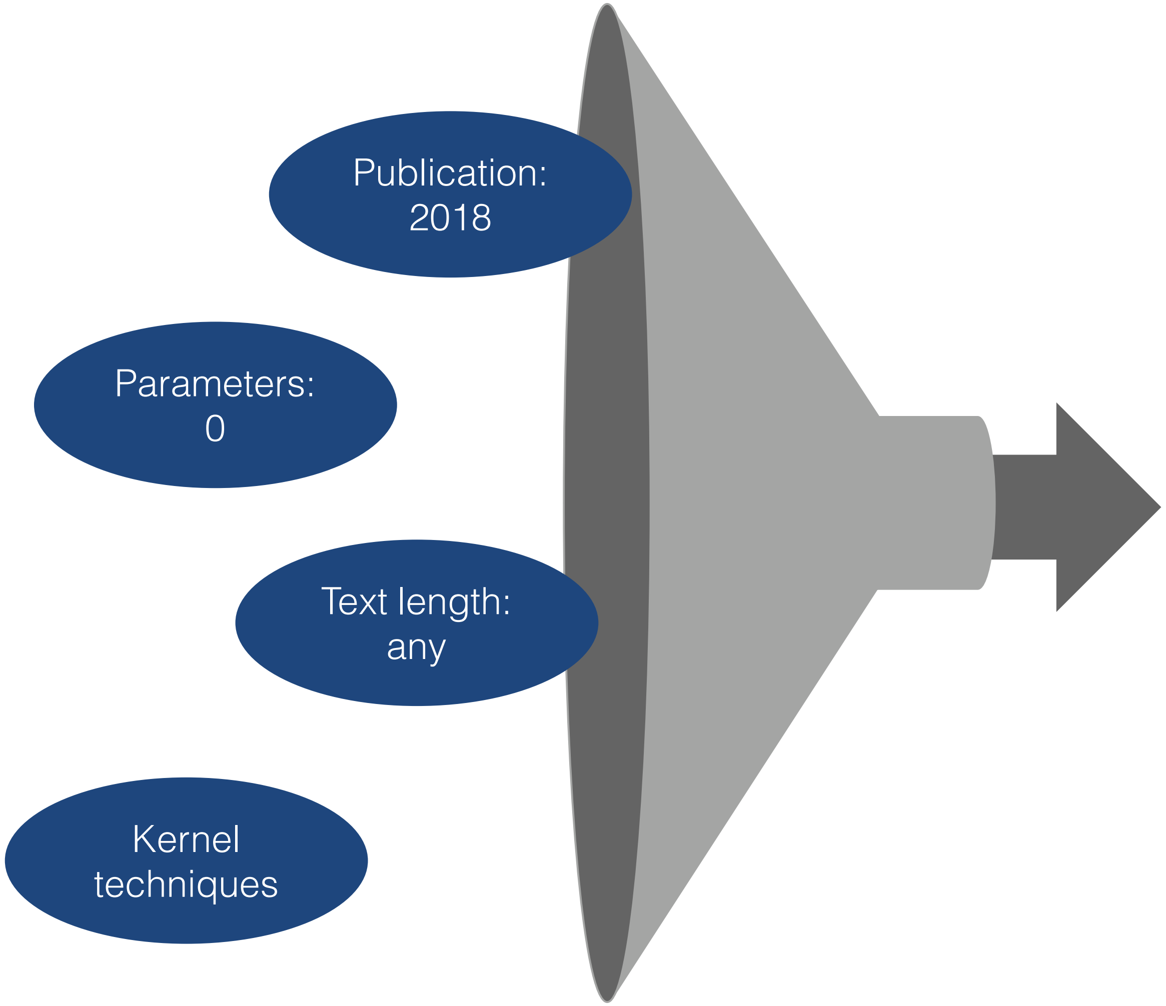
## TRANSFORMER (BERT)

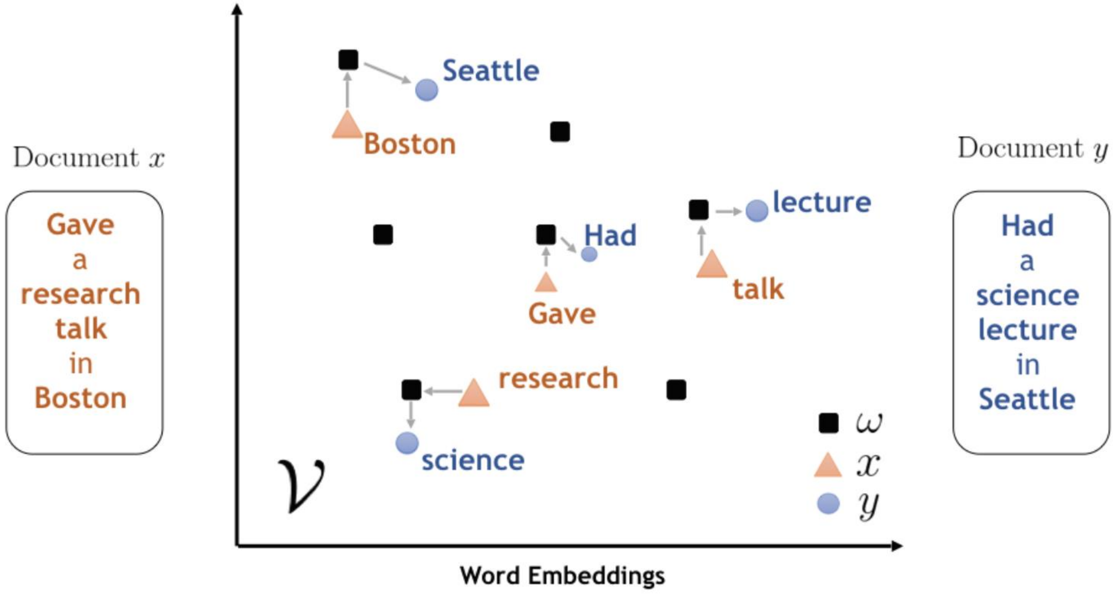


# WME changes the way we think about similarity for documents

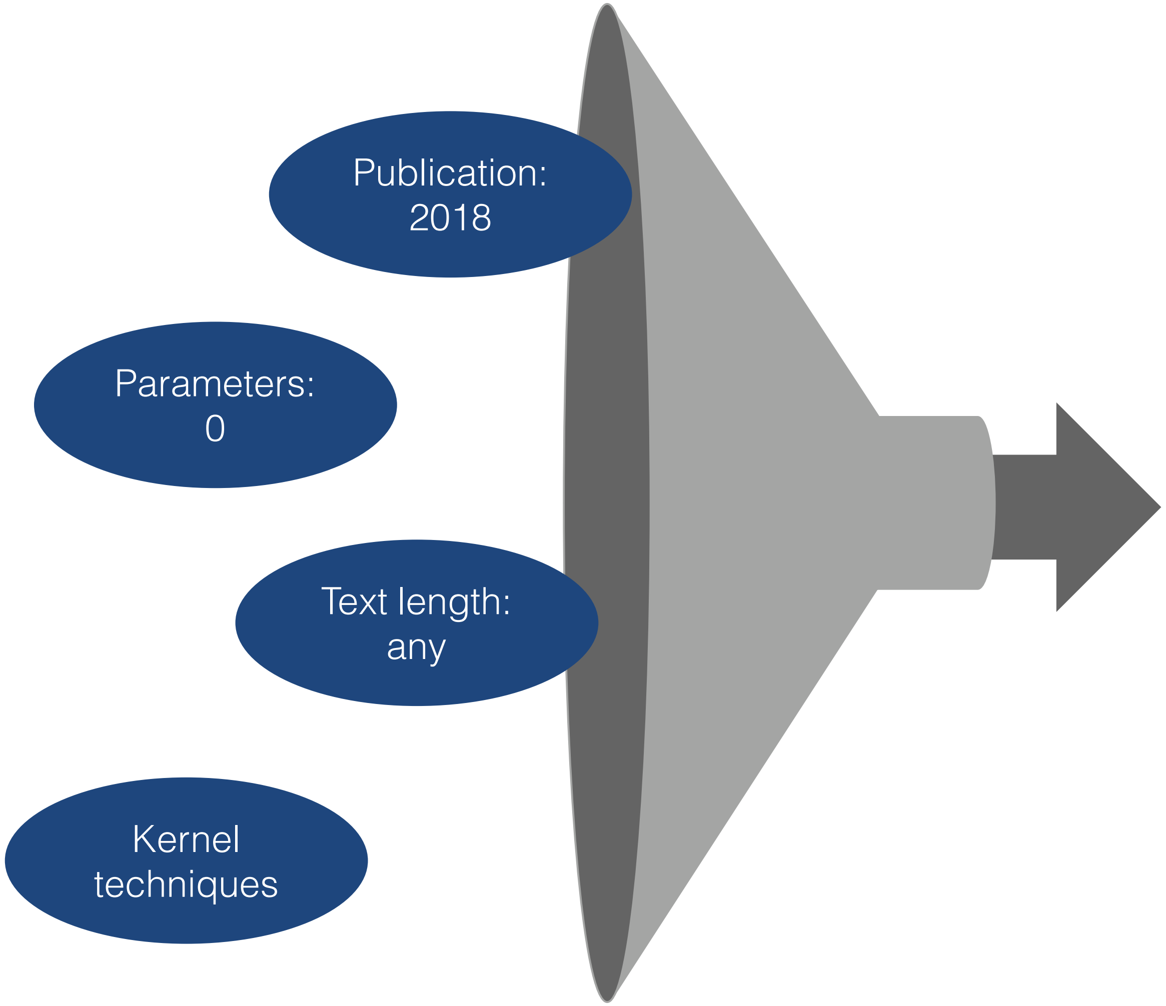
## WORD MOVER‘S EMBEDDING



# WME has a surprisingly simple and efficient algorithm

## WORD MOVER'S EMBEDDING




**Input:** Texts $\{x_i\}_{i=1}^{N}$, $D_{\max}$, $R$.
**Output:** Matrix $Z_{N \times R}$, with rows corresponding to text embeddings.

1: Compute $v_{\max}$ and $v_{\min}$ as the maximum and minimum values, over all coordinates of the word vectors $\boldsymbol{v}$ of $\{x_i\}_{i=1}^{N}$, from any pre-trained word embeddings (e.g. Word2Vec, GloVe or PSL999).
2: **for** $j = 1, \ldots, R$ **do**
3: Draw $D_j \sim \text{Uniform}[1, D_{\max}]$.
4: Generate a random document $\omega_j$ consisting of $D_j$ number of random words drawn as $\omega_{j\ell} \sim \text{Uniform}[v_{\min}, v_{\max}]^d$, $\ell = 1, \ldots, D_j$.
5: Compute $\boldsymbol{f}_{x_i}$ and $\boldsymbol{f}_{\omega_j}$ using a popular weighting scheme (e.g. NBOW or TF-IDF).
6: Compute the WME feature vector $Z_j = \phi_{\omega_j}(\{x_i\}_{i=1}^{N})$ using WMD in Equation (2).
7: **end for**
8: Return $Z(\{x_i\}_{i=1}^{N}) = \frac{1}{\sqrt{R}}[Z_1\ Z_2\ \ldots\ Z_R]$

c/o FU Berlin
Königin-Luise-Straße 24-26
14195 Berlin
www.cct-ev.de

Thank you for your attention!

Tim Korjakow

Student researcher