Information Retrieval SS 2024

Ex3: Revelance Feedback, Semantic Retrieval

Benedikt Ebing & Fabian David Schmidt
partially based on “An Introduction to Information Retrieval” by Manning, Raghavan and Schütze

Revelance Feedback

Task 1

Suppose that a user's initial query is **cheap CDs cheap DVDs extremely cheap CDs**. The user examines two documents, d1 and d2. She judges d1, with the content **CDs cheap software cheap CDs** relevant and d2 with the content **cheap thrills DVDs** nonrelevant. Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Using Rocchio relevance feedback what would the revised query vector be after relevance feedback? Assume α = 1, β = 0.75, γ = 0.25.

word	query q	d1	d2
CDs	2	2	0
cheap	3	2	1
DVDs	1	0	1
extremely	1	0	0
software	0	1	0
thrills	0	0	1

$$q_m = \alpha \cdot q_0 + \left(\beta \cdot \frac{1}{|D_r|} \sum_{d_j \in D_r} d_j \right) - \left(\gamma \cdot \frac{1}{|D_{nr}|} \sum_{d_j \in D_{nr}} d_j \right)$$

Task 2

In Rocchio's algorithm, what weight setting for α/β/γ does a "Find pages like this one" search correspond to?

Task 1

Suppose that a user's initial query is **cheap CDs cheap DVDs extremely cheap CDs**. The user examines two documents, d1 and d2. She judges d1, with the content **CDs cheap software cheap CDs** relevant and d2 with the content **cheap thrills DVDs** nonrelevant. Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Using Rocchio relevance feedback what would the revised query vector be after relevance feedback? Assume α = 1, β = 0.75, γ = 0.25.

word	query q	d1	d2
CDs	2	2	0
cheap	3	2	1
DVDs	1	0	1
extremely	1	0	0
software	0	1	0
thrills	0	0	1

$$\hat{q} = 1 \cdot q + 0.75 \cdot d_1 - 0.25 \cdot d_2 = \begin{bmatrix} 3.5 \\ 4.25 \\ 0.75 \\ 1 \\ 0.75 \\ -0.25 \end{bmatrix} = \begin{bmatrix} 3.5 \\ 4.25 \\ 0.75 \\ 1 \\ 0.75 \\ 0 \end{bmatrix}$$

Task 2

In Rocchio's algorithm, what weight setting for α/β/γ does a "Find pages like this one" search correspond to?

Task 1

Suppose that a user's initial query is **cheap CDs cheap DVDs extremely cheap CDs**. The user examines two documents, d1 and d2. She judges d1, with the content **CDs cheap software cheap CDs** relevant and d2 with the content **cheap thrills DVDs** nonrelevant. Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Using Rocchio relevance feedback what would the revised query vector be after relevance feedback? Assume α = 1, β = 0.75, γ = 0.25.

word	query q	d1	d2
CDs	2	2	0
cheap	3	2	1
DVDs	1	0	1
extremely	1	0	0
software	0	1	0
thrills	0	0	1

$$\hat{q} = 1 \cdot q + 0.75 \cdot d_1 - 0.25 \cdot d_2 = \begin{bmatrix} 3.5 \\ 4.25 \\ 0.75 \\ 1 \\ 0.75 \\ -0.25 \end{bmatrix} = \begin{bmatrix} 3.5 \\ 4.25 \\ 0.75 \\ 1 \\ 0.75 \\ 0 \end{bmatrix}$$

Task 2

In Rocchio's algorithm, what weight setting for α/β/γ does a "Find pages like this one" search correspond to?

"Find pages like this one" ignores the query, also no negative judgements are used here. Hence the values are α = γ = 0, which implies β = 1.

Task 3

Omar has implemented a relevance feedback web search system, where he is going to do relevance feedback based only on words in the title text returned for a page (for efficiency). The user is going to rank 3 results. The first user, Jinxing, queries for:

banana slug

and the top three titles returned are:

banana slug Ariolimax columbianus
Santa Cruz mountains banana slug
Santa Cruz Campus Mascot

Jinxing judes the first two documents Relevant, and the third Not Relevant. Assume that Omar's search engine uses term frequency but no length normalization nor IDF. Assume that he is using the Rocchio relevance feedback mechanism, with $\alpha = \beta = \gamma = 1$. Show the final revised query that would be run. (Please list the vector elements in alphabetical order.)

word	q	d1	d2	d3
Ariolimax	0	1	0	0
banana	1	1	1	0
Campus	0	0	0	1
columbianus	0	1	0	0
Cruz	0	0	1	1
Mascot	0	0	0	1
mountains	0	0	1	0
Santa	0	0	1	1
slug	1	1	1	0

Task 3

Omar has implemented a relevance feedback web search system, where he is going to do relevance feedback based only on words in the title text returned for a page (for efficiency). The user is going to rank 3 results. The first user, Jinxing, queries for:

banana slug

and the top three titles returned are:

banana slug Ariolimax columbianus
Santa Cruz mountains banana slug
Santa Cruz Campus Mascot

Jinxing judes the first two documents Relevant, and the third Not Relevant. Assume that Omar's search engine uses term frequency but no length normalization nor IDF. Assume that he is using the Rocchio relevance feedback mechanism, with α = β = γ = 1. Show the final revised query that would be run. (Please list the vector elements in alphabetical order.)

word	q	d1	d2	d3
Ariolimax	0	1	0	0
banana	1	1	1	0
Campus	0	0	0	1
columbianus	0	1	0	0
Cruz	0	0	1	1
Mascot	0	0	0	1
mountains	0	0	1	0
Santa	0	0	1	1
slug	1	1	1	0

$$q_r = \begin{bmatrix} 1/2 \\ 2 \\ 0 \\ 1/2 \\ 0 \\ 0 \\ 1/2 \\ 0 \\ 2 \end{bmatrix}.$$

Semantic Retrieval
-
Latent Semantic Analysis

Task 1

Consider the following collection of documents:

- **Document 1:** *Frodo and Sam were trembling in the darkness, surrounded in darkness by hundreds of blood-thirsty orcs. Sam was certain these beasts were about to taste the scent of their flesh.*
- **Document 2:** *The faceless black beast then stabbed Frodo. He felt like every nerve in his body was hurting. Suddenly, he thought of Sam and his calming smile. Frodo had betrayed him.*
- **Document 3:** *Frodo's sword was radiating blue, stronger and stronger every second. Orcs were getting closer. And these weren't just regular orcs either, Uruk-Hai were among them. Frodo had killed regular orcs before, but he had never stabbed an Uruk-Hai, not with the blue stick.*
- **Document 4:** *Sam was carrying a small lamp, shedding some blue light. He was afraid that orcs might spot him, but it was the only way to avoid deadly pitfalls of Mordor.*

Your vocabulary consists of the following terms: **Frodo**, **Sam**, **beast**, **orc**, and **blue**. Compute the TF-IDF document-term occurrence matrix for given document collection and vocabulary terms.

$$\text{document-by-term-matrix:} = \begin{array}{c} \begin{array}{ccccc} \text{Frodo} & \text{Sam} & \text{beast} & \text{orc} & \text{blue} \end{array} \\ \begin{pmatrix} 0.1249 & 0.2499 & 0 & 0 & 0 \\ 0.2499 & 0.1249 & 0.6021 & 0 & 0 \\ 0.2499 & 0 & 0 & 0 & 0.6021 \\ 0 & 0.1249 & 0 & 0 & 0.3010 \end{pmatrix} \end{array} \begin{array}{l} \text{doc 1} \\ \text{doc 2} \\ \text{doc 3} \\ \text{doc 4} \end{array}$$

raw tf, idf=log10(N/df_t)

Task 2

Perform the singular value decomposition of the above matrix and write down the obtained factor matrices U, Σ, and V. You can use some existing programming library to perform the SVD (e.g., numpy.linalg.svd in Python).

$$\mathrm{U} = \begin{pmatrix} 0.1422 & 0.0875 & 0.9288 & 0.3309 \\ 0.538 & 0.829 & -0.1349 & -0.0717 \\ 0.7601 & -0.4941 & -0.2019 & 0.3707 \\ 0.3356 & -0.2471 & 0.28 & -0.8648 \end{pmatrix}$$

doc 1
doc 2
doc 3
doc 4

$$\Sigma = \begin{pmatrix} 0.7450 & 0 & 0 & 0 \\ 0 & 0.6366 & 0 & 0 \\ 0 & 0 & 0.2676 & 0 \\ 0 & 0 & 0 & 0.1338 \end{pmatrix}$$

concept 1
concept 2
concept 3
concept 4

Frodo Sam beast orc blue

$$V^T = \begin{pmatrix} 0.4592 & 0.1942 & 0.4348 & 0 & 0.7499 \\ 0.1486 & 0.1485 & 0.7840 & 0 & -0.5841 \\ 0.1192 & 0.9351 & -0.3035 & 0 & -0.1392 \\ 0.8676 & -0.2567 & -0.3228 & 0 & -0.2777 \end{pmatrix}$$

Task 3

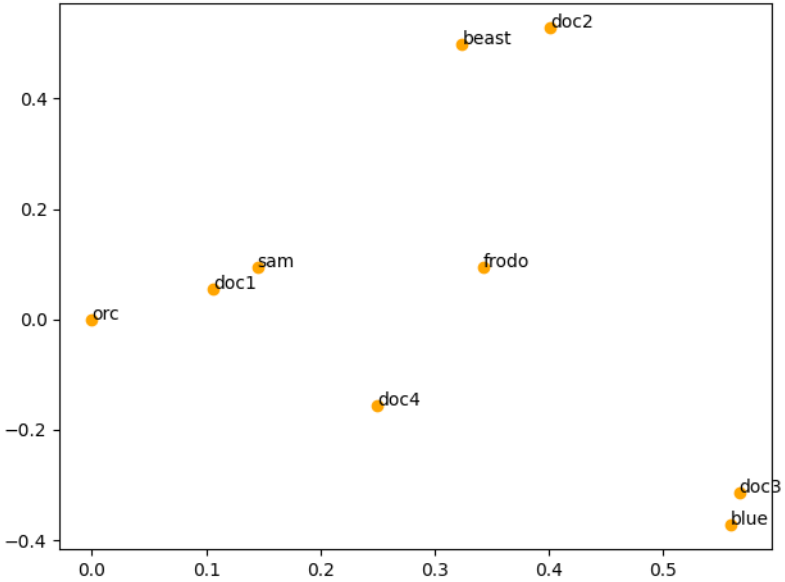
Reduce the rank of the factor matrices to K = 2, i.e., compute the 2-dimensional vectors for vocabulary terms and documents. Show terms and documents as points in a 2-dimensional graph.

Task 3

Reduce the rank of the factor matrices to K = 2, i.e., compute the 2-dimensional vectors for vocabulary terms and documents. Show terms and documents as points in a 2-dimensional graph.

$$\text{projected documents} = \begin{pmatrix} 0.1059 & 0.0557 \\ 0.4008 & 0.5277 \\ 0.5662 & -0.3145 \\ 0.2500 & -0.1573 \end{pmatrix} = \mathrm{U}[:,:2]\,\Sigma[:2,:2]$$

$$\text{projected terms} = \begin{pmatrix} 0.3421 & 0.1447 & 0.3239 & 0.0000 & 0.5586 \\ 0.0946 & 0.0946 & 0.4991 & 0.0000 & -0.3718 \end{pmatrix} = \Sigma[:2,:2]V^T[:2,:]$$



Task 4

You are given the query “Sam blue orc”. Compute the latent vector for the query and rank the documents according to similarity of their latent vectors with the obtained latent vector of the query.

Task 4

You are given the query “Sam blue orc”. Compute the latent vector for the query and rank the documents according to similarity of their latent vectors with the obtained latent vector of the query.

Query- and document-projections:

$$\text{documents in latent space (rows)} = \begin{pmatrix} 0.1059 & 0.0557 & 0.2485 & 0.0443 \\ 0.4008 & 0.5277 & -0.0361 & -0.0096 \\ 0.5662 & -0.3145 & -0.054 & 0.0496 \\ 0.2500 & -0.1573 & 0.0749 & -0.1157 \end{pmatrix} = \mathrm{U}\,\Sigma$$

$$\text{query in latent space} \quad q = \begin{pmatrix} 0.9441 \\ -0.4355 \\ 0.7959 \\ -0.5344 \end{pmatrix}^T = [0 \quad 1 \quad 0 \quad 1 \quad 1]\,V = V^T\,[0 \quad 1 \quad 0 \quad 1 \quad 1]$$

Task 4

You are given the query “Sam blue orc”. Compute the latent vector for the query and rank the documents according to similarity of their latent vectors with the obtained latent vector of the query.

Query- and document-projections:

$$\text{documents in latent space (rows)} = \begin{pmatrix} 0.1059 & 0.0557 & 0.2485 & 0.0443 \\ 0.4008 & 0.5277 & -0.0361 & -0.0096 \\ 0.5662 & -0.3145 & -0.054 & 0.0496 \\ 0.2500 & -0.1573 & 0.0749 & -0.1157 \end{pmatrix} = \mathrm{U}\,\Sigma$$

$$\text{query in latent space} \quad q = \begin{pmatrix} 0.9441 \\ -0.4355 \\ 0.7959 \\ -0.5344 \end{pmatrix}^T = [0 \quad 1 \quad 0 \quad 1 \quad 1]\,V = V^T\,[0 \quad 1 \quad 0 \quad 1 \quad 1]$$

Similarities and Ranking:

$$cos(q, d_1) = 0.6325$$
$$cos(q, d_2) = 0.1331$$
$$cos(q, d_3) = 0.6531$$
$$cos(q, d_4) = 0.9241$$

1. doc 4
2. doc 3
3. doc 1
4. doc 2

Semantic Retrieval
-
Representation Learning

Task 1

For your semantic retrieval system you are training a CBOW model (windows size=2). Your vocabulary consists of the following terms:

[“Frodo”, “followed”, “Sam”, “into”, “the”, “dark”, “Mordor”, “Ring”]

You are currently processing the sentence “Frodo followed Sam into the dark”. Which (positive) training examples are derived from the sentence?

Task 1

For your semantic retrieval system you are training a CBOW model (windows size=2). Your vocabulary consists of the following terms:

[“Frodo”, “followed”, “Sam”, “into”, “the”, “dark”, “Mordor”, “Ring”]

You are currently processing the sentence “Frodo followed Sam into the dark”. Which (positive) training examples are derived from the sentence?

	Input x	**Output y**
Frodo followed Sam into the dark	followed, Sam	Frodo
Frodo followed Sam into the dark	Frodo, Sam, into	followed
Frodo followed Sam into the dark	Frodo, followed, into, the	Sam
Frodo followed Sam into the dark	followed, Sam, the, dark	into
Frodo followed Sam into the dark	Sam, into, dark	the
Frodo followed Sam into the dark	into, the	dark

Task 2

Which (positive) training examples are derived if we would consider the Skip-gram model?

	Input x	Output y
Frodo followed Sam into the dark	Frodo	followed
	Frodo	Sam
Frodo followed Sam into the dark	followed	Frodo
	followed	Sam
	followed	into
Frodo followed Sam into the dark	Sam	Frodo
	Sam	followed
	Sam	into
	Sam	the
Frodo followed Sam into the dark	into	followed
	into	Sam
	into	the
	into	dark
Frodo followed Sam into the dark	the	Sam
	the	into

...

Task 3

Your word vectors, i.e. CBOW model, is parameterized as follows:

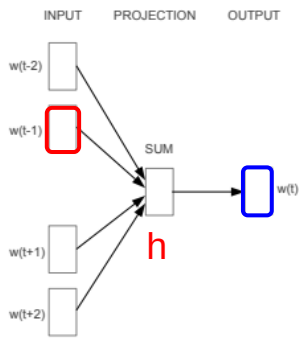
$$W = \begin{bmatrix} -0.01 & -0.88 & -1.74 & -0.4 \\ 0.69 & -0.46 & 0.23 & -1.34 \\ -1.34 & -0.54 & 0.29 & 0.01 \\ 0.17 & -0.36 & -0.06 & 0.93 \\ -0.6 & -0.65 & -0.52 & 1.7 \\ 0.17 & -0.61 & -0.54 & -1.35 \\ -1.32 & -0.89 & -2.4 & 0.09 \\ -2.19 & 1.1 & 0.58 & 0.63 \end{bmatrix}$$

Frodo
followed
Sam
into
the
dark
Mordor
Ring

$$W' = \begin{bmatrix} 0.64 & 0.01 & -0.83 & 0.11 & -0.04 & -0.64 & -1.49 & -0.68 \\ 1.4 & -1.75 & -1.58 & -0.39 & 0.11 & -1.47 & -1.78 & -0.55 \\ -0.83 & -0.6 & 0.67 & -0.71 & 0.22 & 0.45 & -0.52 & -0.91 \\ -0.23 & 1.19 & 0.37 & -0.49 & 0.02 & -0.96 & -2.7 & 0.59 \end{bmatrix}$$

Frodo followed **Sam** into the dark Mordor Ring

Calculate the output of the last layer (softmax layer) for the current sentence in which Sam is the center word.



CBOW *

*Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

Task 3

Your word vectors, i.e. CBOW model, is parameterized as follows:

$$W = \begin{bmatrix} -0.01 & -0.88 & -1.74 & -0.4 \\ 0.69 & -0.46 & 0.23 & -1.34 \\ -1.34 & -0.54 & 0.29 & 0.01 \\ 0.17 & -0.36 & -0.06 & 0.93 \\ -0.6 & -0.65 & -0.52 & 1.7 \\ 0.17 & -0.61 & -0.54 & -1.35 \\ -1.32 & -0.89 & -2.4 & 0.09 \\ -2.19 & 1.1 & 0.58 & 0.63 \end{bmatrix}$$

Frodo
followed
Sam
into
the
dark
Mordor
Ring

$$W' = \begin{array}{c} \begin{matrix} \text{Frodo} & \text{followed} & \textbf{Sam} & \text{into} & \text{the} & \text{dark} & \text{Mordor} & \text{Ring} \end{matrix} \\ \begin{bmatrix} 0.64 & 0.01 & -0.83 & 0.11 & -0.04 & -0.64 & -1.49 & -0.68 \\ 1.4 & -1.75 & -1.58 & -0.39 & 0.11 & -1.47 & -1.78 & -0.55 \\ -0.83 & -0.6 & 0.67 & -0.71 & 0.22 & 0.45 & -0.52 & -0.91 \\ -0.23 & 1.19 & 0.37 & -0.49 & 0.02 & -0.96 & -2.7 & 0.59 \end{bmatrix} \end{array}$$

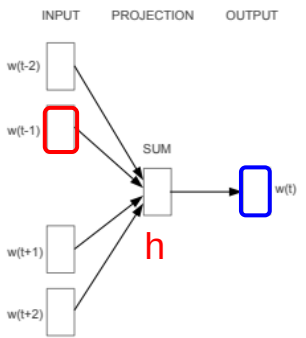
Calculate the output of the last layer (softmax layer) for the current sentence in which Sam is the center word.

(h = avg of context embeddings of Frodo, followed, into, the)

$$h = [0.0625 \quad -0.5875 \quad -0.5225 \quad 0.2225]$$

$$\text{logits} = [-0.4 \quad 1.6070 \quad 0.6086 \quad 0.4980 \quad -0.1776 \quad 0.3749 \quad 0.6236 \quad 0.8874]$$

$$\hat{y} = [0.04 \quad 0.32 \quad 0.12 \quad 0.1 \quad 0.05 \quad 0.09 \quad 0.12 \quad 0.15]$$ (softmax(logits))



CBOW *

*Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

Task 4

What is the final document embedding if we represent it as the average of its constituent word embeddings?

Sentence word embeddings (rows) =

```
[[-0.01, -0.88, -1.74, -0.4 ,  0.64,  1.4 , -0.83, -0.23],
 [ 0.69, -0.46,  0.23, -1.34,  0.01, -1.75, -0.6 ,  1.19],
 [-1.34, -0.54,  0.29,  0.01, -0.83, -1.58,  0.67,  0.37],
 [ 0.17, -0.36, -0.06,  0.93,  0.11, -0.39, -0.71, -0.49],
 [-0.6 , -0.65, -0.52,  1.7 , -0.04,  0.11,  0.22,  0.02],
 [ 0.17, -0.61, -0.54, -1.35, -0.64, -1.47,  0.45, -0.96]]
```

Sentence embedding $s =$ `[-0.15, -0.58, -0.39, -0.08, -0.12, -0.61, -0.13, -0.02]`

Task 5

Name one shortcoming of representing documents and queries as average word embeddings and how to overcome it?

Task 4

What is the final document embedding if we represent it as the average of its constituent word embeddings?

Sentence word embeddings (rows) =

```
[[-0.01, -0.88, -1.74, -0.4 ,  0.64,  1.4 , -0.83, -0.23],
 [ 0.69, -0.46,  0.23, -1.34,  0.01, -1.75, -0.6 ,  1.19],
 [-1.34, -0.54,  0.29,  0.01, -0.83, -1.58,  0.67,  0.37],
 [ 0.17, -0.36, -0.06,  0.93,  0.11, -0.39, -0.71, -0.49],
 [-0.6 , -0.65, -0.52,  1.7 , -0.04,  0.11,  0.22,  0.02],
 [ 0.17, -0.61, -0.54, -1.35, -0.64, -1.47,  0.45, -0.96]]
```

Sentence embedding $s =$ `[-0.15, -0.58, -0.39, -0.08, -0.12, -0.61, -0.13, -0.02]`

Task 5

Name one shortcoming of representing documents and queries as average word embeddings and how to overcome it?

Every word embedding contributes an equal share to the sentence embedding, this can lead to noisy input due to stop words. If we use a parameterized aggregation function (e.g. Deep Learning Model) it can learn to focus on important features (cf. Learning to Rank).

Task 6

Use your computed document embedding as a query vector and rank the four document documents from the previous task by their cosine similarities, use the following document embeddings:

Document 1:	[1.17	0.05	-1.69	0.15	1.87	-0.25	-0.92	0.84]
Document 2:	[-0.88	-0.65	-0.51	-1.08	-0.25	1.01	0.54	-0.7]
Document 3:	[2.93	-2.28	0.01	1.65	1.15	1.24	0.26	0.52]
Document 4:	[1.22	-1.04	0.11	0.97	0.74	0.08	-1.18	-0.11]

Normalized sentence emb $= \hat{q} =$ [-0.16, -0.6, -0.41, -0.08, -0.13, -0.64, -0.14, -0.02]

Normalized embedding for $d_1 = \hat{d_1} =$ [0.38, 0.02, -0.55, 0.05, 0.61, -0.08, -0.3 , 0.27]

$cos(d_1, q) = dot(\hat{d_1}, \hat{q}) = 0.16$

Ranking: d1, d4, d2, d3

Task 7

After training your embedding model you obtain word representations for every word you observed (assuming you derived your vocabulary form the corpus). Why shouldn't we use every available word embedding after training?

When training word embeddings we start with random vectors. Typically word frequencies follow Zipf's Law, hence many words occur only very few times (long tail). For those words we don't have reliable word vectors (infrequent updates during training). Including them would introduce noise. Because of this we limit the vocabulary to the top k most frequently seen words.