Topic Models

Latent Semantic Analysis

Fang Peng

Northwest Evaluation Association

fang.peng@nwea.org

Overview of Latent Semantic Analysis

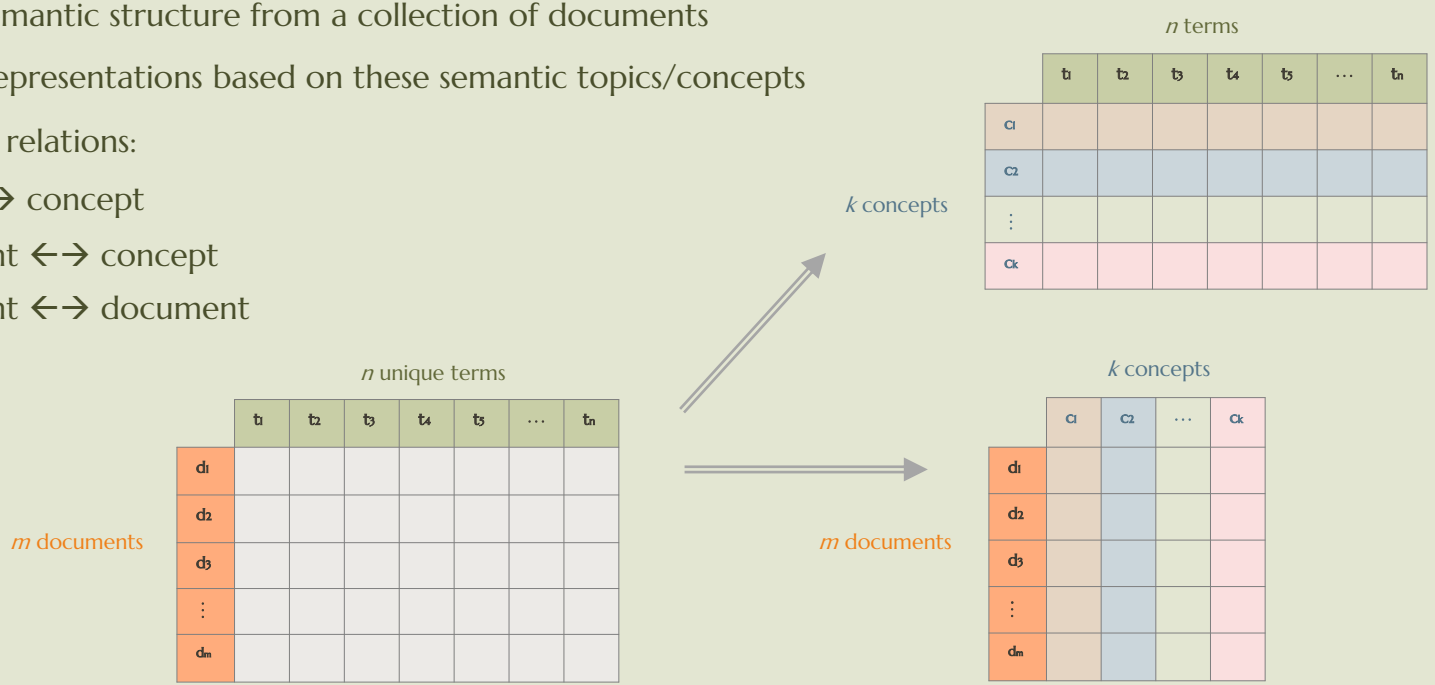
Basic idea behind LSA

- Natural language has a higher order structure (latent semantic structure) which is often obscured by word usage
  - synonyms: edit, revise
  - polysemy: novel, bank
- Words that are closer in semantic meaning will appear in similar pieces of text
  - assess, evaluate, measure
  - bow, arrow, sight
  - election, president, constitution

Overview of Latent Semantic Analysis

Goal of LSA

- Extract latent semantic structure from a collection of documents
- Generate text representations based on these semantic topics/concepts
- Derive semantic relations:
  - word ←→ concept
  - document ←→ concept
  - document ←→ document

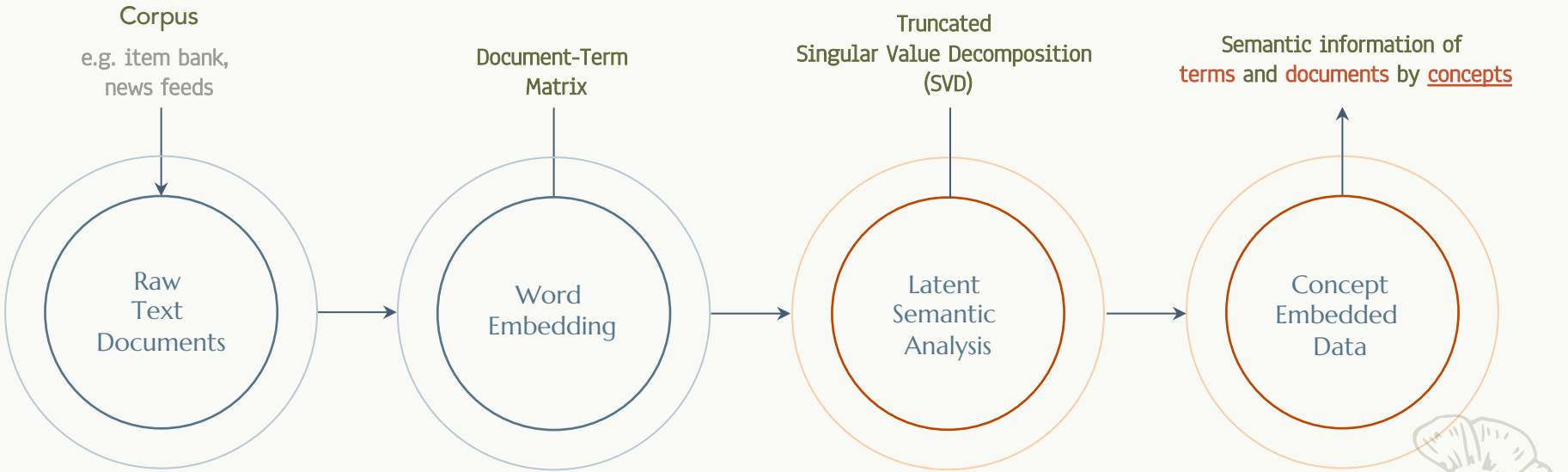


Application of Latent Semantic Analysis

Application of LSA in measurement contexts

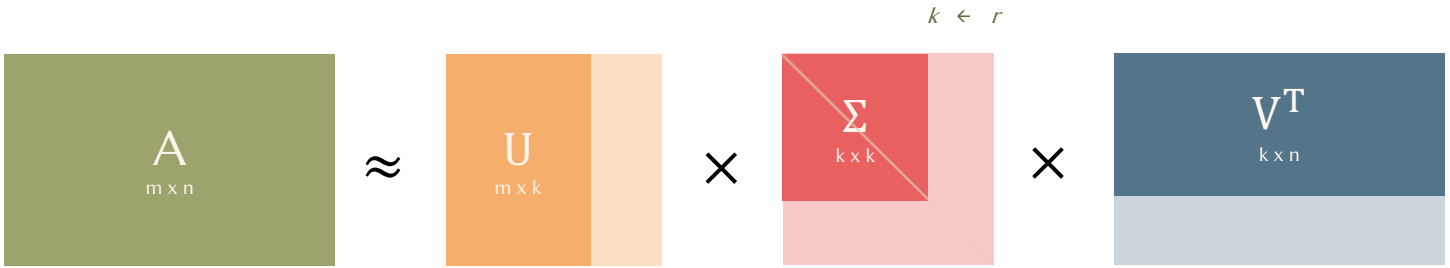
- Automatic scoring
  - Essays: compare student essays with human-scored essays
  - Open-ended questions: compare student responses with correct answers
- Automatic enemy item detection
  - Screen enemy relationships between items in vast item banks
- Automated item generation
  - Select distractors that are homogenous with the correct answer

Procedures of Latent Semantic Analysis



Math behind LSA – Intuition and Graphic Representation

Truncated Singular Value Decomposition (SVD)



Input word embedding matrix:

m documents, *n* terms

Left singular vectors:

m documents, *k* concepts

Singular values:

Strength of each topic in descending order

Right singular vectors:

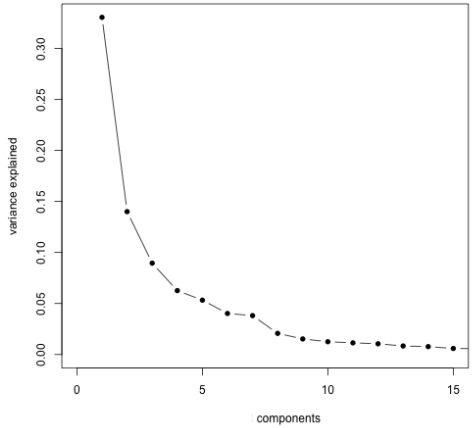
k concepts, *n* terms

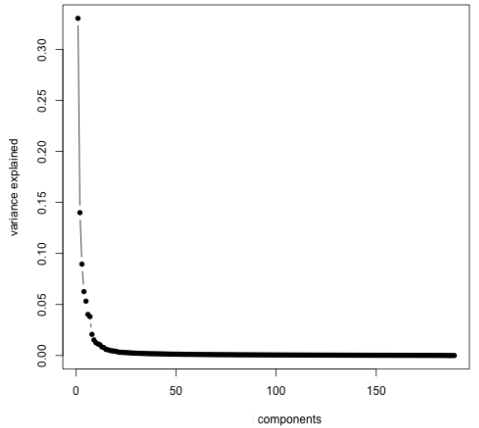
Selection of Optimal Number of Components/Concepts

How to choose *k* ?

- Heuristic approach
  - Elbow / Knee method

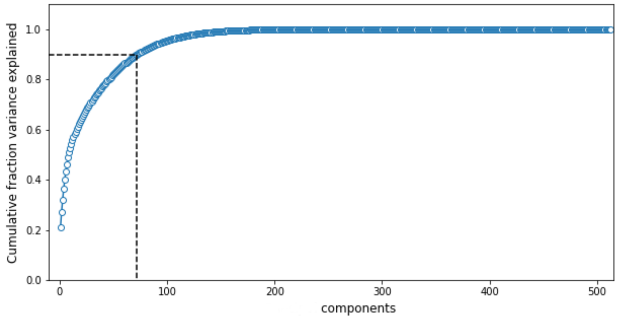
    Square of a singular value (Sigma matrix) divided by sum of squares of singular values, indicates proportion variance explained by the corresponding singular vector





  - Proportion variance explained by singular vectors

    Arbitrary: 80% | 90% | 95%



Selection of Optimal Number of Components/Concepts

How to choose *k* ?

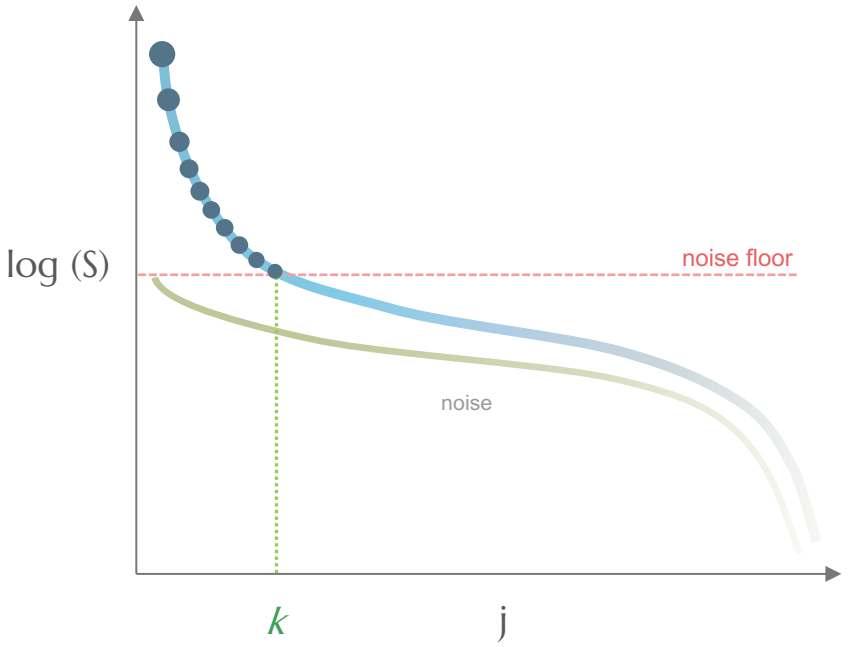
- Gavish & Donoho (2014)

$$X = X_{true} + \sigma Z_{noise}$$

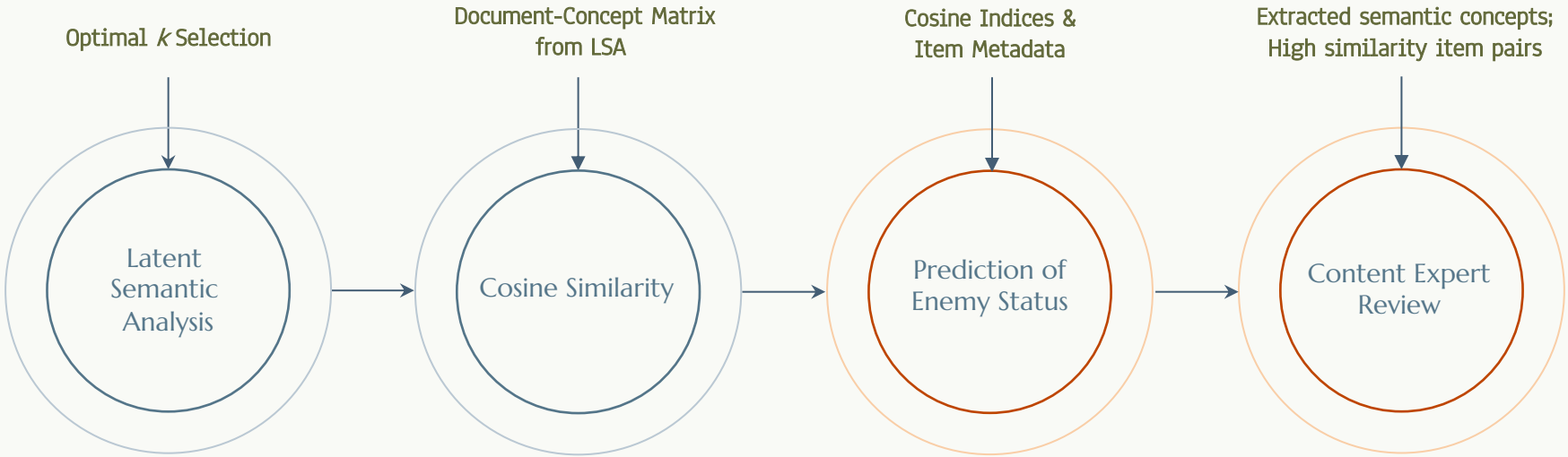
X_{true} True lower rank matrix

Z_{noise} ~ mean = 0; variance = 1

σ Noise level



Procedures of Enemy Item Detection



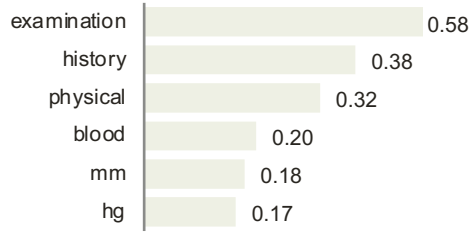
Content Expert Review

Evaluation of concept coherence

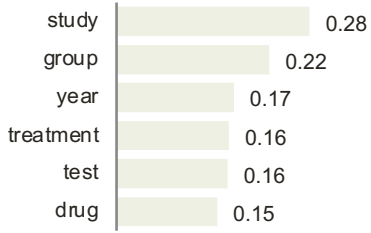
- Present the top concepts extracted to content experts with associated terms
  - Whether the clusters of words translate to meaningful semantic concepts
  - Whether the concepts are too general or too specific
- Review high-similarity item pairs for enemy relation identification
  - Empirical thresholds obtained from content review
  - Review item pairs based on criteria established for similarity indices
  - Train a prediction model on labeled data and predict enemy probability on unlabeled item pairs

Example Concepts Extracted | Item Bank 1

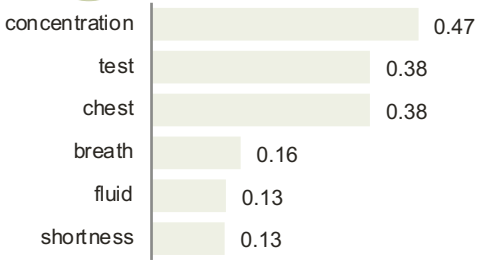
1 Patient Vignette



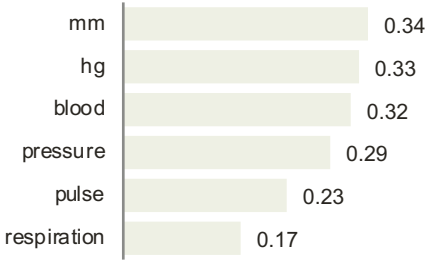
2 Clinical Study



3 Respiratory



4 Blood Pressure



Example Concepts Extracted | Item Bank 2

Concept 1 Routine Work		Concept 2 Report of Abnormality		Concept 3 Emotional Support	
report	0.51	notify	0.48	talk	0.65
charge	0.50	incident	0.42	encourage	0.19
care	0.43	abnormal	0.14	feeling	0.07
provide	0.18	finding	0.06	listen	0.07
plan	0.17	observation	0.04	provide	0.06

Concept 4 Inquiry		Concept 5 Infection Prevention		Concept 6 Standard Precaution	
ask	0.60	hand	0.93	glove	0.40
family	0.29	wash	0.21	wear	0.31
speak	0.12	glove	0.14	standard	0.28
member	0.10	prevent	0.11	precaution	0.19
need	0.09	infection	0.09	universal	0.18

Latent Semantic Analysis on Item Bank 2

Data

1,461 multiple choice items

A total of 1,066,530 item pairs: n(n-1)/2

327 known enemy pairs

Word-Embedding Matrix

2,922 documents (Stems + Keys)

2,147 Unique terms

Latent Semantic Analysis

k= 455

2,922 x 455 Document-Concept matrix

4 cosine similarity indices:

- Stem <--> Stem
- Key <--> Key
- $Stem_1$ <--> Key_2
- $Stem_2$ <-> Key_1

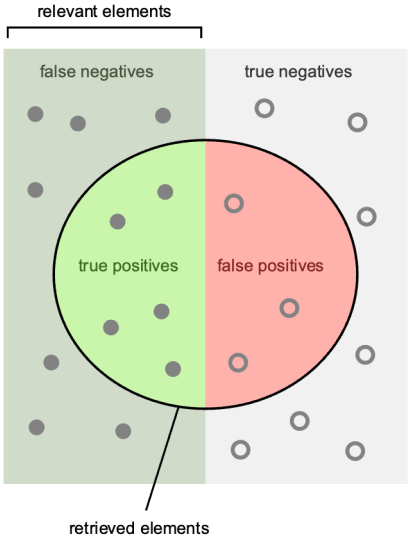
Classification Results (Initial)

Classification Results Before Content Review

Prob. Cutoff	Recall	Precision	F1 Score
0.60	95.2%	1.5%	2.6%
0.90	90.3%	2.5%	4.7%

Initial low precision was expected because true enemy item pairs might occur among the False Positives

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$



Content Review Process

Content Review

- Many unreviewed item pairs with unknown enemy relationship
- True enemy item pairs among the False Positives

Review Rules

- Candidate item pairs: False Positive pairs with predicated probability > .60
- Sorted in descending order of predicted probability and grouped into sets of 20
- Review ordered batches of 20 item pairs and record the number of items confirmed to be enemy
- Stop rule: when less than 10% true enemy item pairs were encountered

Review Results

- 1,040 FP item pairs reviewed
- 469 (45%) confirmed to be true enemies

Classification Results (After Review)

Classification Results After Content Review

New confirmed enemy relationships update:

469 new + 327 existing enemies;

571 non-enemies

Prob. Cutoff	Recall	Precision	F1 Score
0.60	95.2%	1.5%	2.6%
0.90	90.3%	2.5%	4.7%

Prob. Cutoff	Recall	Precision	F1 Score
0.60	97.3%	6.0%	11.3%
0.90	93.3%	11.7%	20.7%

Practical Implications

Content Review

- Many unreviewed item pairs with unknown enemy relationship
- True enemy item pairs among the False Positives

Review Rules

- Candidate item pairs: False Positive pairs with predicated probability > .60
- Sorted in descending order of predicted probability and grouped into sets of 20
- Review ordered batches of 20 item pairs and record the number of items confirmed to be enemy
- Stop rule: when less than 10% true enemy item pairs were encountered

Review Results

- 1,040 FP item pairs reviewed
- 469 (45%) confirmed to be true enemies

Implementation in Python

Python Walkthrough of Latent Semantic Analysis

Quick Setup:

1. Download Anaconda distribution: https://www.anaconda.com

2. Install Jupyter Notebook within Anaconda Navigator

3. Install required packages:

   pandas, numpy

   nltk, sklearn

   seaborn, matplotlib