CS5489 - Machine Learning

Lecture 6b - Linear Dimensionality Reduction for Text

Dr. Antoni B. Chan

Dept. of Computer Science, City University of Hong Kong

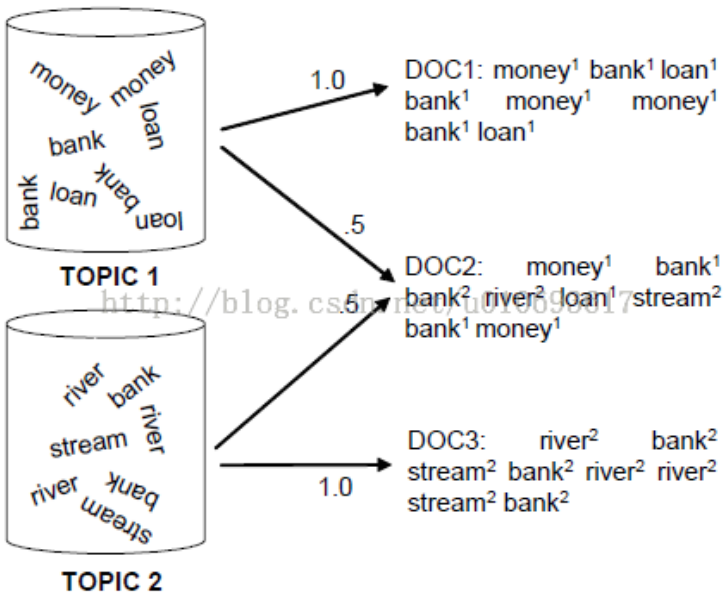
Outline

1. Linear Dimensionality Reduction for Vectors
2. **Linear Dimensionality Reduction for Text**
3. Non-linear Dimensionality Reduction
4. Manifold Embedding

Dimensionality Reduction

- **Goal:** Transform high-dimensional vectors into low-dimensional vectors.
  - Dimensions in the low-dim data represent co-occuring features in high-dim data.
  - Dimensions in the low-dim data may have semantic meaning.

- **For example:** document analysis
  - high-dim: bag-of-word vectors of documents
  - low-dim: each dimension represents similarity to a topic.



Latent Semantic Analysis (LSA)

- Also called *Latent Semantic Indexing*
- Consider a bag-of-word representation (e.g., TF, TF-IDF)
  - document vector $\mathbf{x}_i$
  - $x_{i,j}$ is the frequency of word j in document i

- Approximate each document vector as a weighted sum of topic vectors.
  - $\hat{\mathbf{x}} = \sum_{n=1}^{p} w_p \mathbf{v}_p$
  - Topic vector $\mathbf{v}_p$ contains co-occuring words.
    - corresponds to a particular *topic* or *theme*.
  - Weight w_p represents similarity of the document to the p-th topic.
- Objective:
  - minimize the squared reconstruction error (Similar to PCA):
  - $\min_{\mathbf{v},\mathbf{w}} \sum_i ||\mathbf{x}_i - \hat{\mathbf{x}}_i||^2$
  - Solution computed using SVD.

- Represent each document by its topic weights.
  - Apply other machine learning algorithms...
- **Advantage:**
  - Finds relations between terms (synonymy and polysemy).
  - distances/similarities are now comparing topics rather than words.
    - higher-level semantic representation

Example on Spam Email dataset

- use bag-of-words representation with 50 words
- term-frequency (TF) normalization

In [2]:
```
# Load spam/ham text data from directories
textdata = datasets.load_files("email", encoding="utf8", decode_error="replace")

# convert to bag-of-words representation
cntvect = feature_extraction.text.CountVectorizer(stop_words='english', max_features=50)
X = cntvect.fit_transform(textdata.data)
Y = textdata.target

# TF representation
tf_trans = feature_extraction.text.TfidfTransformer(norm='l1', use_idf=False)
Xtf = tf_trans.fit_transform(X)

# print the vocabulary
print(cntvect.vocabulary_)
```

```
{'online': 39, 'watches': 49, 'brands': 13, 'bags': 11, 'new': 35, 'toda
y': 46, 'hi': 25, 'peter': 42, 'contact': 17, 'mr': 34, 'email': 21, 'co
m': 16, 'number': 37, 'country': 19, 'nigeria': 36, 'answer': 10, 'statu
s': 44, 'good': 23, 'day': 20, 'john': 30, 'united': 47, 'payment': 40, 'b
ank': 12, 'address': 9, 'information': 28, 'inform': 27, '10': 2, 'offic
e': 38, 'kamara': 32, '000': 1, 'codeine': 15, '30mg': 6, '30': 5, '00':
0, '60': 8, 'pills': 43, '120': 3, 'just': 31, 'cost': 18, 'buy': 14, 'm
```

```
g': 33, 'percocet': 41, '50': 7, '15mg': 4, 'visa': 48, 'isidoro': 29, 'st
ore': 45, 'http': 26, 'google': 24, 'files': 22}
```

LSA on Spam data

- Apply LSA with 5 topics
  - implemented as `TruncatedSVD`

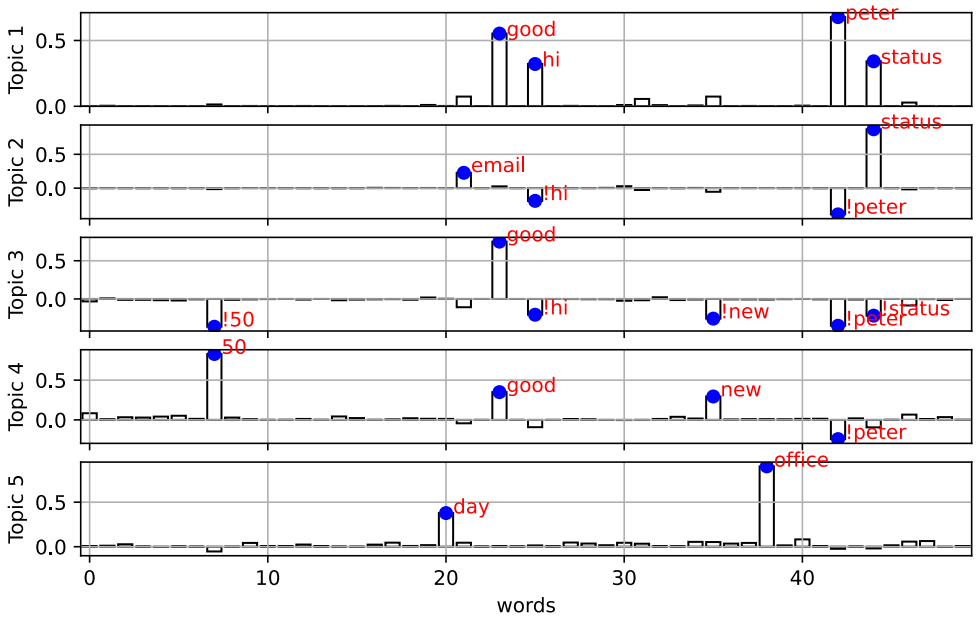
In [3]:
```
lsa = decomposition.TruncatedSVD(n_components=5, random_state=4487)
Wlsa = lsa.fit_transform(Xtf)

# components
V = lsa.components_
```

Topic vectors

- topic vectors contain frequent co-occuring words

In [5]:
```
vocab = asarray(cntvect.get_feature_names())
lsafig = plot_topics(lsa, vocab, figsize=(8,5))
lsafig
```

Out[5]:

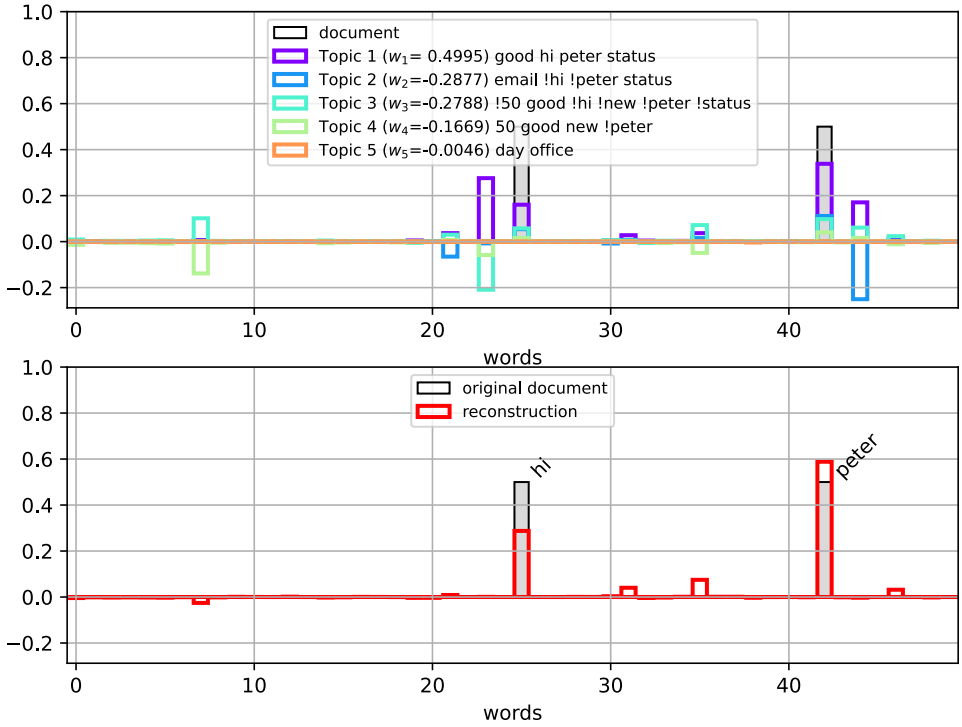


Document representation

- Documents are a combination of topics

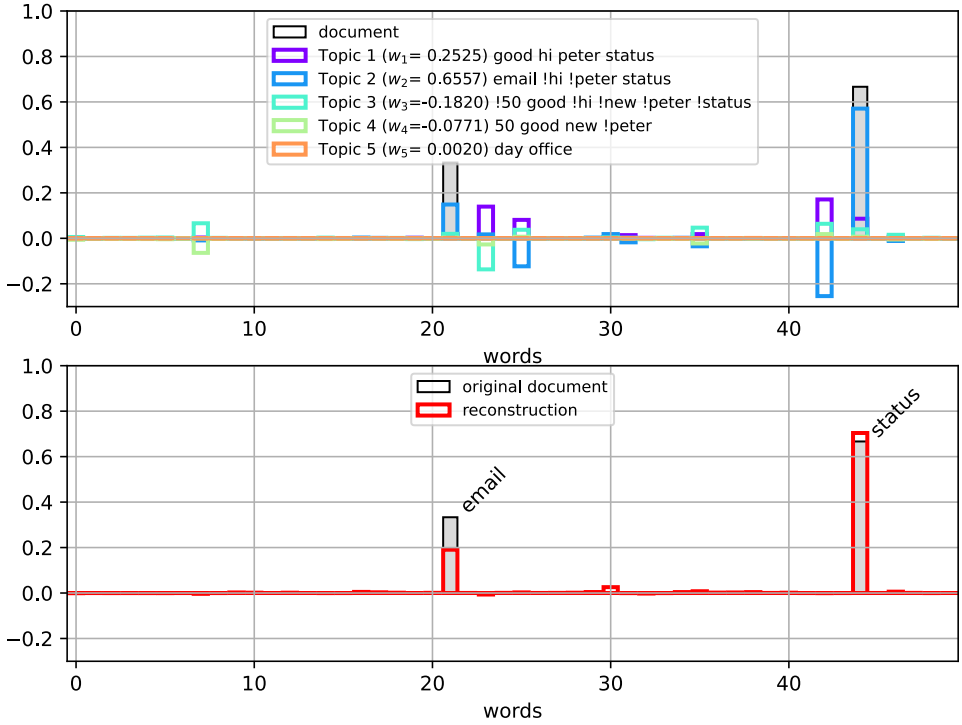
In [7]:
```
plot_doc_topic(Xtf[2,:], Wlsa[2,:], lsa, vocab)
```

Out[7]:



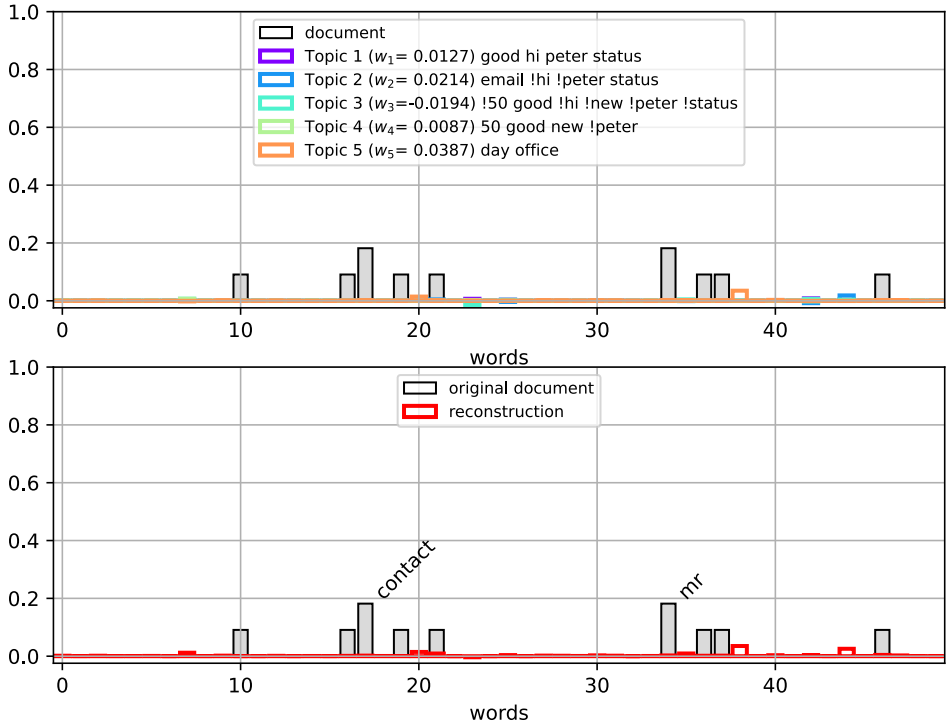
In [8]:
```
plot_doc_topic(Xtf[4,:], Wlsa[4,:], lsa, vocab)
```

Out[8]:



In [9]:
```
plot_doc_topic(Xtf[3,:], Wlsa[3,:], lsa, vocab)
```

Out[9]:

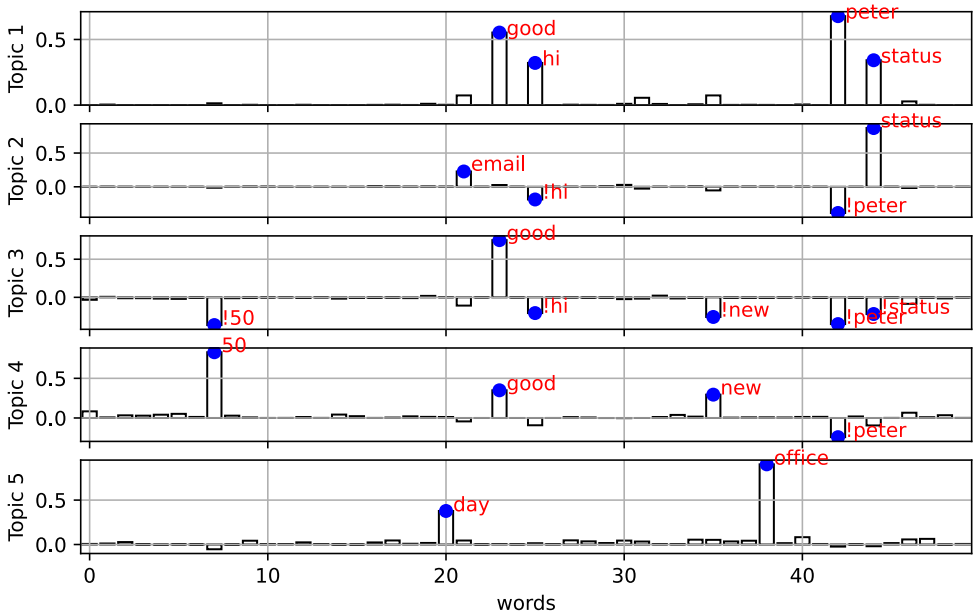


Problem with LSA

- In the topic vector, the "frequency" of a word can be negative!
  - Doesn't really make sense for document bag-of-words model.

In [10]:
```
lsafig
```

Out[10]:



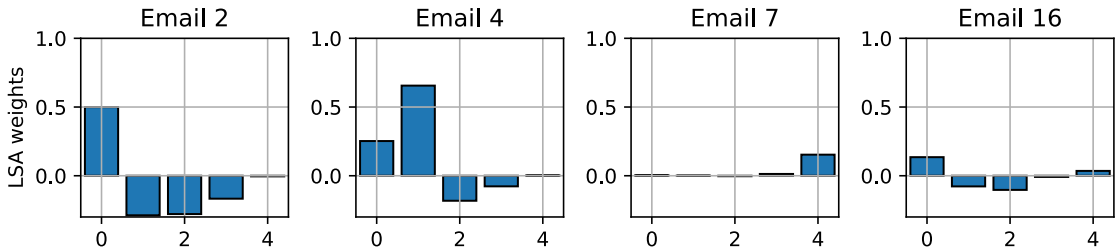
Problems with LSA

- The weights for each topic can be negative!
  - Topics should only be "additive"

- Topics should increase probability of some topic-related words, but not decrease probability of other words.
- It doesn't make sense to "remove" a topic using a negative topic weight.

In [12]:
```
pfig
```

Out[12]:



Non-negative Matrix Factorization (NMF)

- **Solution:** constrain the topic vector and weights to be non-negative.
- Similar to LSA
  - Approximate each document vector as a weighted sum of topic vectors.
    - $\hat{\mathbf{x}}_j = \sum_{n=1}^{p} w_p \mathbf{v}_p$
    - But now, each entry of topic vector $\mathbf{v}_p \geq 0$ and topic weight $w_p \geq 0$
  - Objective: minimize the squared reconstruction error

$$\min_{\mathbf{v} \geq 0, \mathbf{w} \geq 0} \sum_j ||\mathbf{x}_j - \hat{\mathbf{x}}_j||^2$$

    - subject to the non-negative constraints.
    - no closed-form solution, need to use an optimizer.

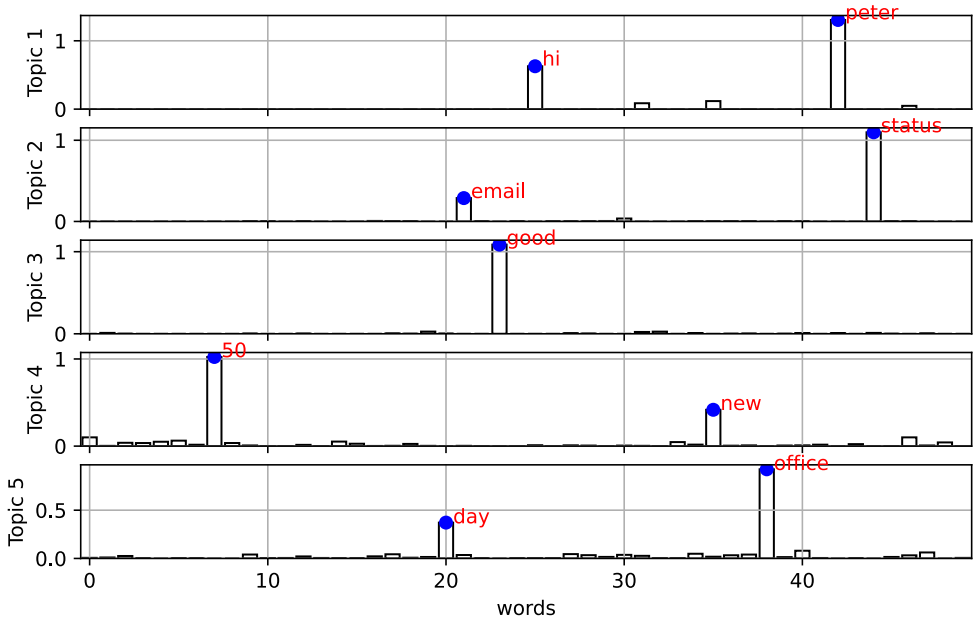
In [13]:
```
# Run NMF
nmf = decomposition.NMF(n_components=5)
Wnmf = nmf.fit_transform(Xtf)

# components
V = nmf.components_
```

Topic vector

- all non-negative entries
- looks much cleaner (less small entries)

In [14]:
```
plot_topics(nmf, vocab, figsize=(8,5))
```

Out[14]:

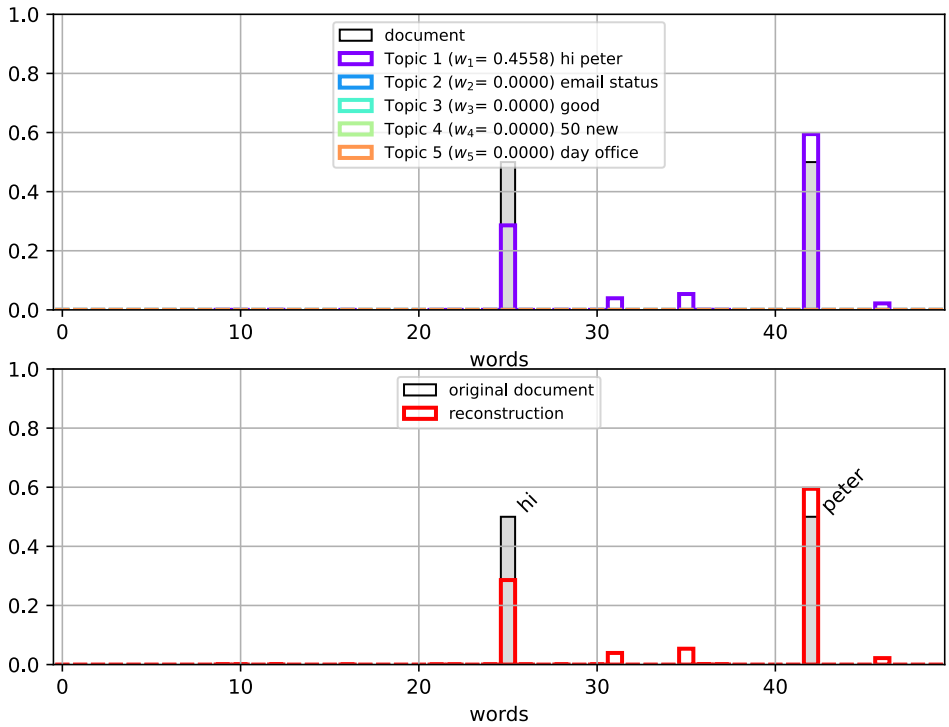


Document vector

- additive combination of topics

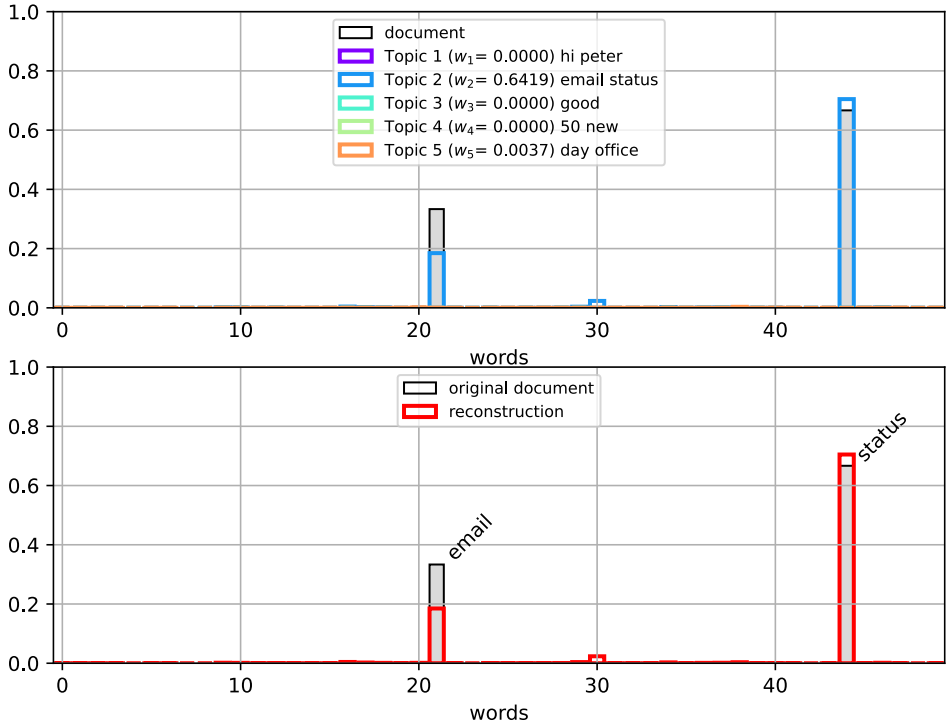
In [15]:
```
plot_doc_topic(Xtf[2,:], Wnmf[2,:], nmf, vocab)
```

Out[15]:



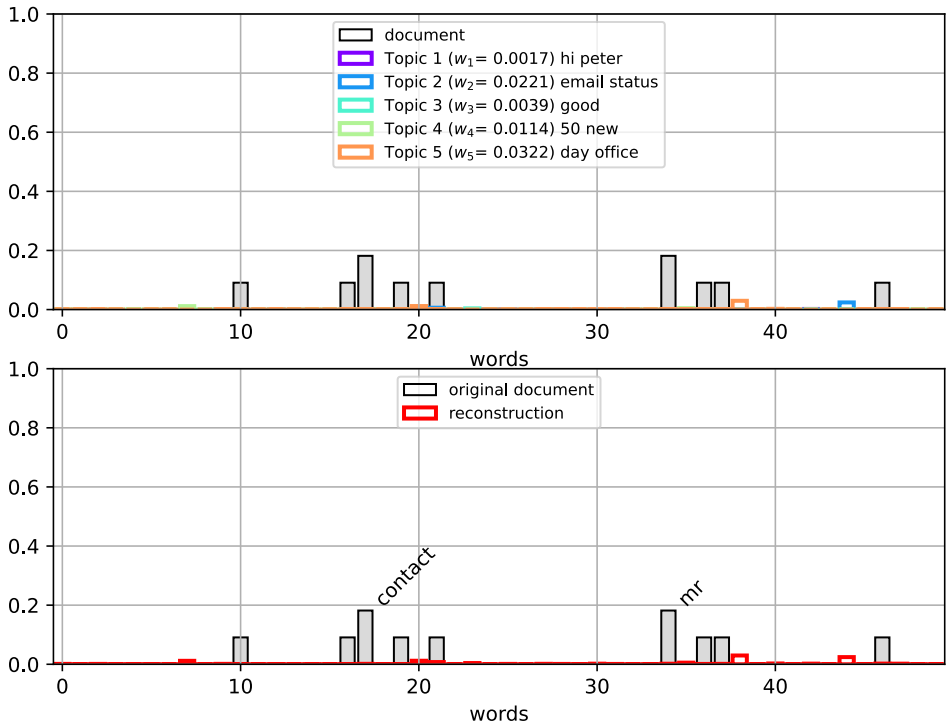
In [16]:
```
plot_doc_topic(Xtf[4,:], Wnmf[4,:], nmf, vocab)
```

Out[16]:



In [17]:
```
plot_doc_topic(Xtf[3,:], Wnmf[3,:], nmf, vocab)
```

Out[17]:

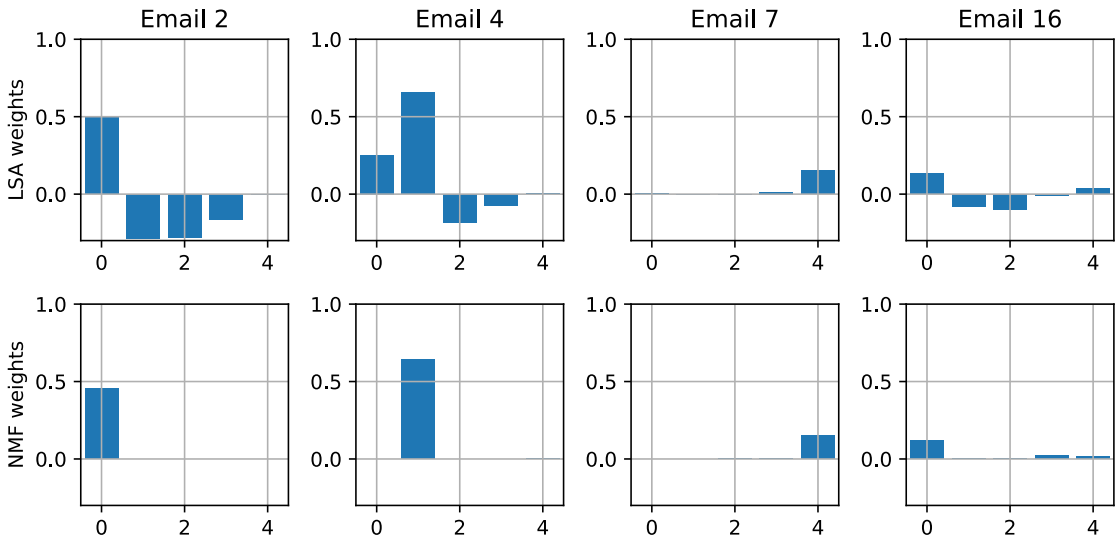


Sparseness

- For NMF representation, most topic weights for a document are zero.
  - this is called a *sparse* representation.
  - each document is only composed of a few topics.

In [19]:
```
spfig
```

Out[19]:

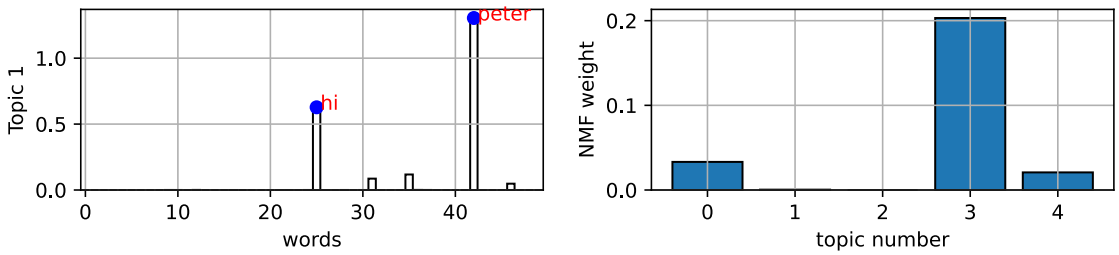


Problem with NMF

- *While the weights and component vectors are non-negative, NMF does not enforce them to be probabilities.*

In [21]:
```
nfig
```

Out[21]:



- TF/TFIDF is a probabilistic model of words in a document
  - the vector of probabilities sums to 1
  - probabilities are between 0 and 1
- The NMF components and weights are difficult to interpret.
- *We will solve this problem later using probabilistic graphical models.*

Linear Dimensionality Reduction - Summary

- **Goal:** given set of input vectors $\{\mathbf{x}_i\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}^d$, represent each input vector as lower-dimensional vector $\mathbf{w}_i \in \mathbb{R}^p$.
  - Approximate $\mathbf{x}$ as a weighted sum of basis vectors $\mathbf{v}_j \in \mathbb{R}^d$
    - $\hat{x} = \sum_{j=1}^p w_j \mathbf{v}_j$
    - minimize the reconstruction error of $\hat{\mathbf{x}}$.
  - enables faster processing, or reduces noise.

Name	Objective	Advantages	Disadvantages
Principal component analysis (PCA)	minimize reconstruction error; preserve the most variance of data	- captures correlated dimensions, removes redundant dimensions, removes noise. - closed-form solution	- does not consider end goal (e.g., classification)

Random Projections	sample random basis vectors.	- fast. - preserves pairwise distances between points (up to accuracy factor).	- adds noise to the pairwise distances.
Fisher's Linear Discriminant (FLD)	maximize class separation	- preserves class separation	- requires class information
Latent Semantic Analysis (LSA)	minimize reconstruction error	- topic vectors have semantic meaning (co-occuring words) - closed-form solution	- topic weights and topic vectors can be negative - does not consider end goal (e.g., classification)
Non-negative Matrix Factorization (NMF)	minimize reconstruction error; non-negative weights and basis vectors.	- "additive" topic/parts model for text or images - sparse topic weights.	- solution requires iterative algorithm. - does not consider end goal (e.g., classification)

Other things

- *Feature Normalization*
  - PCA and LDA are based on the covariance between input dimensions.
  - applying *per-feature* normalization will yield a different PCA result!
    - normalizing each input dimension changes the relative covariances.

In [25]:
```
nfig
```

Out[25]: