Applied Data Analysis (CS401)

ada

Lecture 11
Handling text data
Part II
26 Nov 2025

EPFL

Maria Brbic

Announcements

- To all ADAmericans:

  Happy Thanksgiving!

- Homework H2 due this **today** EOD
  - Reminder: We won’t answer questions asked during final 24h
- Projects:
  - Milestone P2 grades have been released
  - **Project Milestone P3 released today!**
- Friday’s lab session:
  - Exercises on handling text ([Exercise 10]())

Feedback

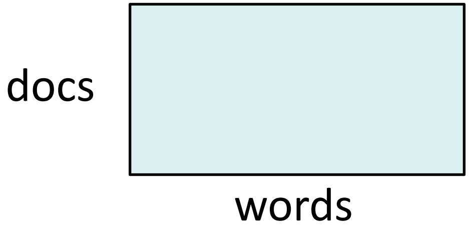
Give us feedback on this lecture here:
https://go.epfl.ch/ada2025-lec11-feedback

- What did you (not) like about this lecture?
- What was (not) well explained?
- On what would you like more (fewer) details?
- …

Recap

Let me open my **bag of tricks for bags of words** for you! But only if you were good children...

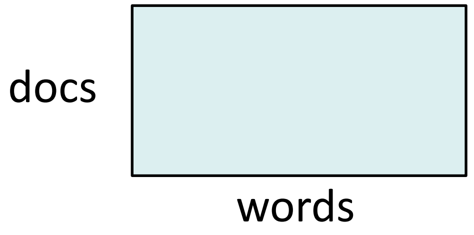
Reminder:
bag-of-words matrix



Revisiting the 4 typical tasks

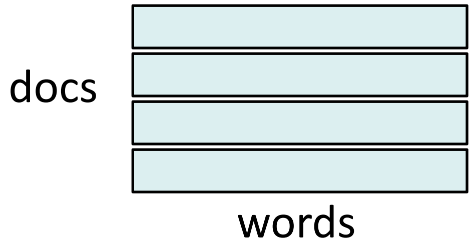
- Document retrieval
- Document classification
- Sentiment analysis
- Topic detection

- TF-IDF matrix to the rescue
  - Entry for doc *d*, word *w*: tf(*d*, *w*) * idf(*w*)



Typical task 1: document retrieval

- Nearest-neighbor method in spirit of kNN
- Compare query doc *q* to all documents in the collection (i.e., rows of the TF-IDF matrix)



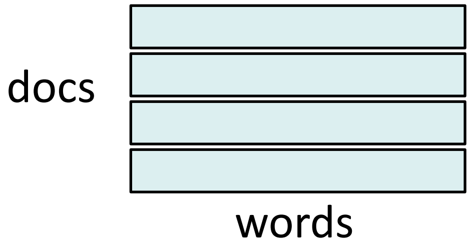
- Rank docs from collection in increasing order of distance
- Distance metrics
  - Typically cosine distance (= 1 – cosine similarity)
  - Recall: cosine similarity of *q* and *v* = <*q*/|*q*|, *v*/|*v*|>
  - If rows are L2-normalized, may simply take dot products <*q*, *v*>

Typical task 1: document retrieval

- This is just the most basic approach
- For efficiency
  - Start by filtering documents by presence of query terms (use efficient full-text index)
  - Hugely narrows down set of documents to be ranked
- Google et al. do much more…
  - Query-independent relevance: PageRank
  - Boost recent results
  - Personalization, contextualization
  - …

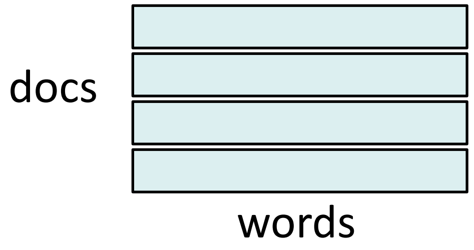
Typical task 2: document classification

- Use TF-IDF matrix as feature matrix for supervised methods (cf. lecture 7)
- Often more features (words) than documents
- What’s the danger with this?
- High model capacity can lead to overfitting (high variance)
- Potential solutions:
  - Use more data (i.e., more labeled training docs)
  - Decrease model capacity:
    - Feature selection
    - Regularization (two slides from now)
    - Dimensionality reduction (a few slides from now)
  - Use ensemble methods such as random forests



Typical task 3: sentiment analysis

- When treated as classification:
  Ctrl-C Ctrl-V previous slide



- When treated as regression:
  Pretty much the same (most supervised methods work for both classification and regression)

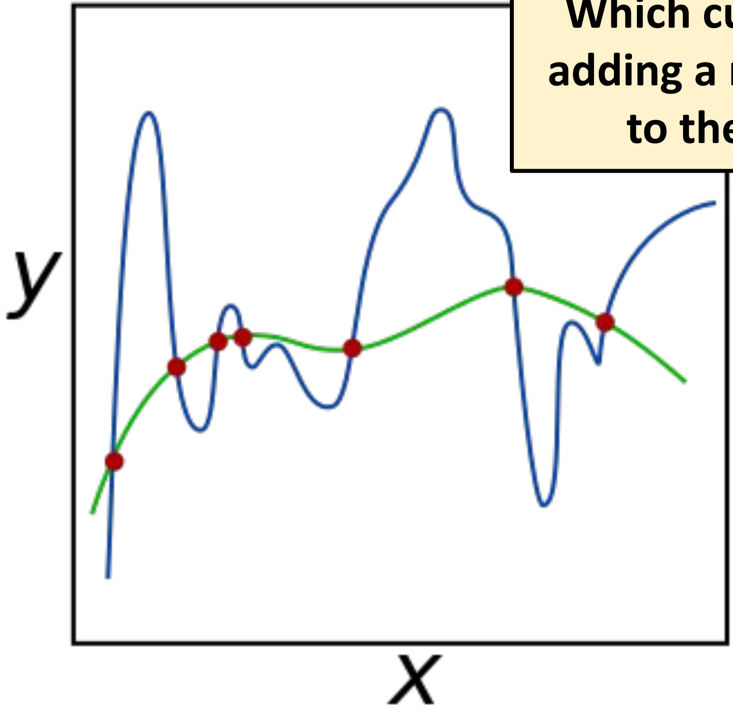
Regularization

- E.g., linear regression:
  Find weight vector $\boldsymbol{\beta}$ that minimizes $$\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \beta)^2$$

  ($\mathbf{x}_i$: feature vector of i-th data point; y_i: label of i-th data point, e.g., sentiment [1–5 stars] expressed in document i)
- If one word j appears only in docs with sentiment 5, we can obtain very small training error on these docs by making β_j large enough
- But doesn’t generalize to unseen test data!
- Remedy: penalize very large positive and very large negative weights:

minimize $$\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top \beta)^2 + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}$$

Regularization



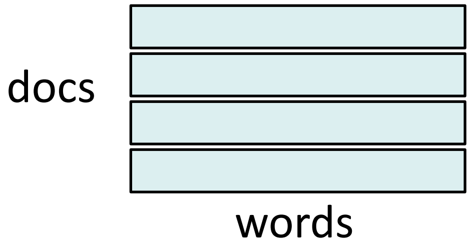
Which curve resulted from adding a regularization term to the loss function?

POLLING TIME

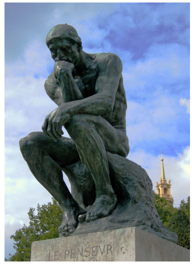
- Scan QR code or go to https://go.epfl.ch/ada2025-lec11-poll

Typical task 4: topic detection

- Cluster rows of TF-IDF matrix (each row a data point)
- Manually inspect clusters and label them with descriptive names (e.g., “news”, “sports”, “romance”, “tech”, “politics”)
- In principle, may use k-means, k-medoids, etc.
- But can be difficult if dimensionality is large (#words $\gg$ #docs)
  - “Curse of dimensionality”
  - Many outliers

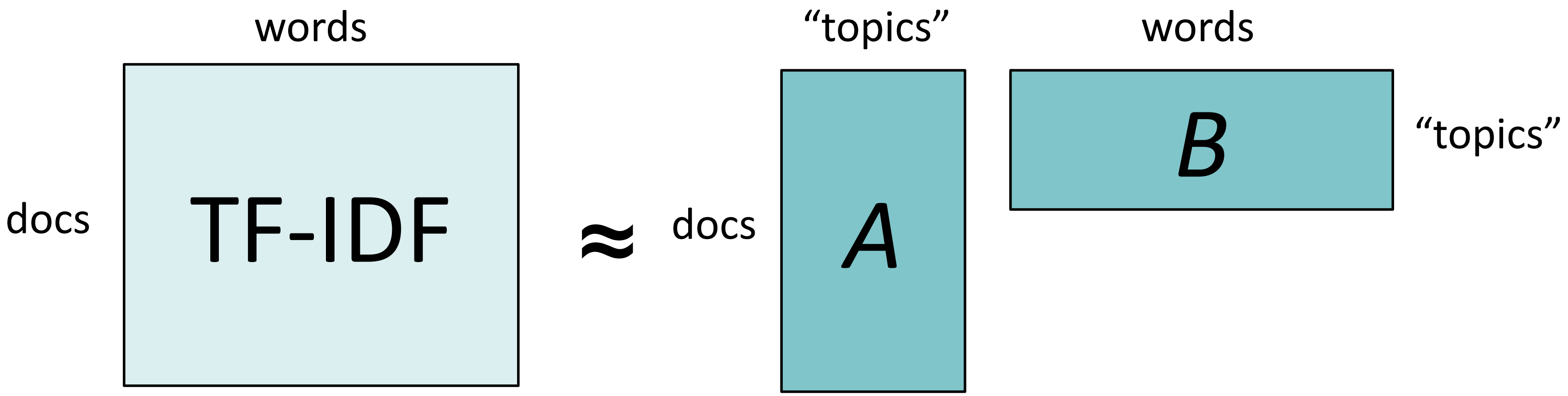


What else can we do?



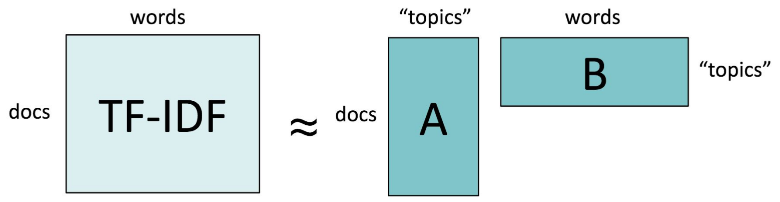
Typical task 4: topic detection

- Alternative approach: **matrix factorization**



- Assume docs and words have representation in (latent) "topic space"
- (IDF-weighted) word frequency modeled as dot product of doc's vectors and word's vectors in topic space
- #topics $\ll$ #words ($\rightarrow$ "dimensionality reduction"): D*W $\rightarrow$ (D+W)*T
- Topics interpretable in doc space (*A*'s cols) and word space (*B*'s rows)

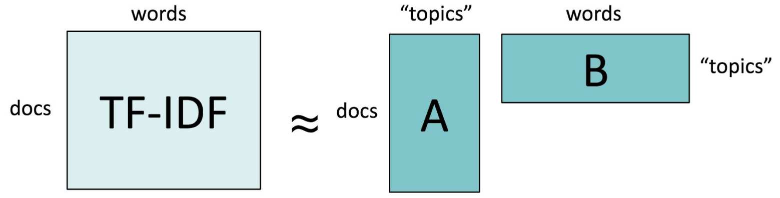
Typical task 4: topic detection



- Optimization problem:
  - Find *A*, *B* such that *AB* is as close to TF-IDF matrix as possible
  - That is, minimize $\sum_{d=1}^{N} \sum_{w=1}^{M} (T_{dw} - A_d \ B_w)^2$

    where *T* is TF-IDF matrix, A_d is *d*-th row of *A*, and B_w is *w*-th column of *B*
- This is called **latent semantic analysis (LSA)**

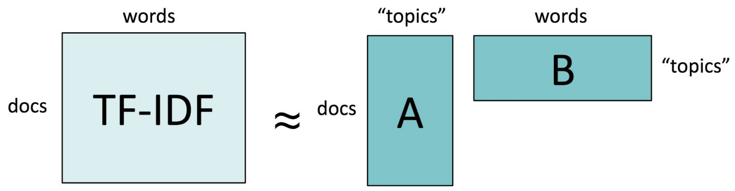
Typical task 4: topic detection



You already know how to efficiently compute this, from your linear algebra class: **singular-value decomposition** (SVD)

- $T = USV^{\mathrm{T}}$
- Freebie: columns of *U* and *V* are orthonormal bases (yay!)
- *S* is diagonal (with values in decreasing order) and captures "importance" of topic (amount of variation in corpus w.r.t. topic)
- If you want *k* topics, keep only the first *k* columns of *U* and *V*, and the first *k* rows and columns of *S*
  $\rightarrow U', S', V'$
- E.g., $A = U', B = S'V'^{\mathrm{T}}$ or $A = U'S', B = V'^{\mathrm{T}}$

Typical task 4: topic detection



- Recall potential problem with clustering and classification and regression: "curse of dimensionality"
- Matrix factorization via LSA solves these problems for you:
  - Use *A* instead of original TF-IDF matrix
  - That is, cluster (or learn to classify or regress) in topic space, rather than word space

- Topic representation from LSA is simply a vector, not a probability distribution over topics
- Probabilistic: LDA = Latent Dirichlet Allocation (p.t.o.)

Commercial break

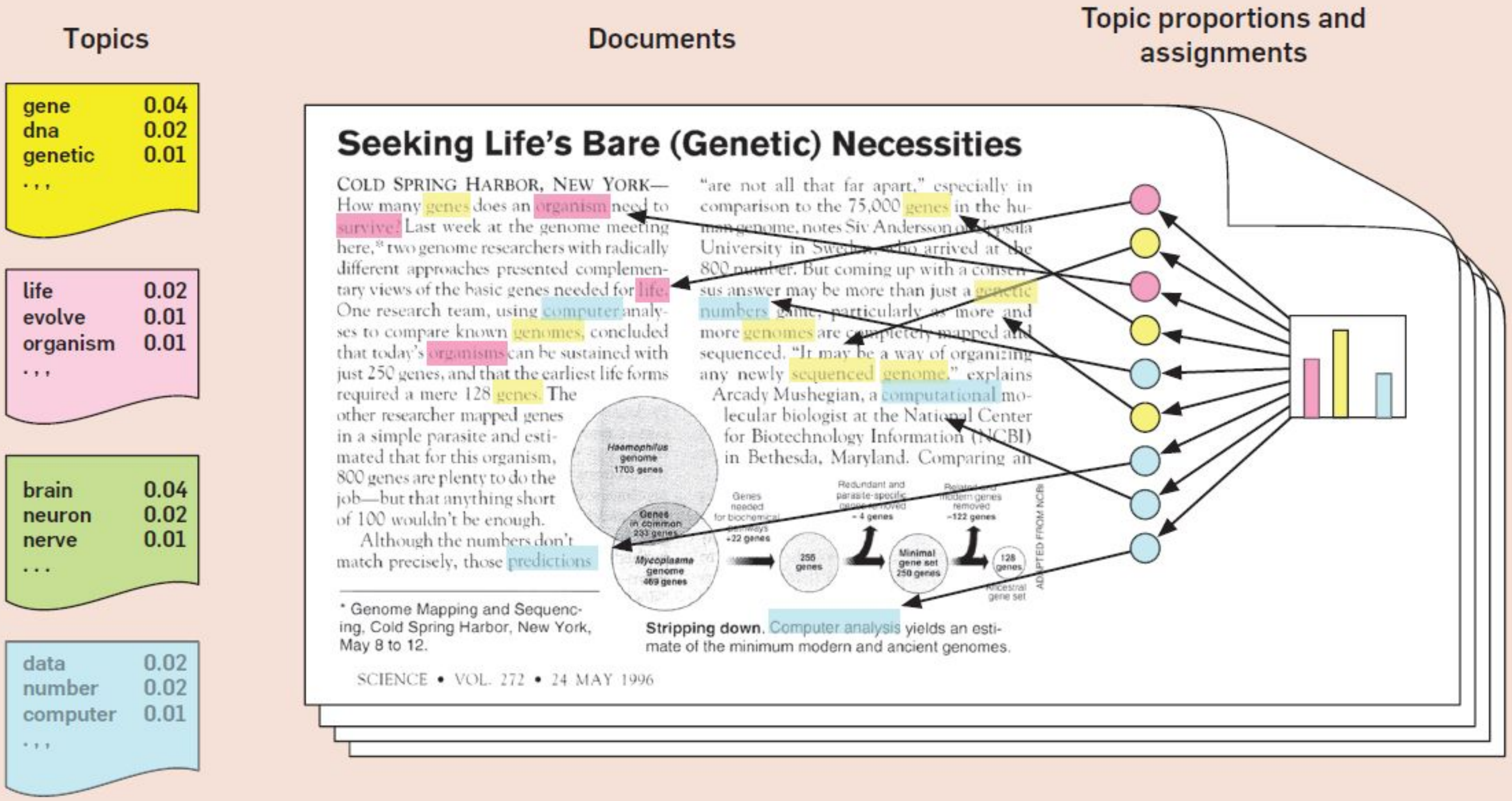
ada

ça roule pour moi

LDA: probabilistic topic modeling

- **L**atent **D**irichlet **A**llocation (*not* Latent Discriminant Analysis!)
- Document := bag of words
- Topic := probability distribution over words
- Each document has a (latent) distribution over topics
- "Generative story" for generating a doc of length *n*:

```
d := sample a topic distribution for the doc (← "Dirichlet")
for i = 1, …, n
    t := sample a topic from topic distribution d
    w := sample a word from topic t
    Add w to the bag of words of the doc to be generated
```


Topics
Documents
Topic proportions and assignments
gene 0.04
dna 0.02
genetic 0.01
...
life 0.02
evolve 0.01
organism 0.01
...
brain 0.04
neuron 0.02
nerve 0.01
...
data 0.02
number 0.02
computer 0.01
...
Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

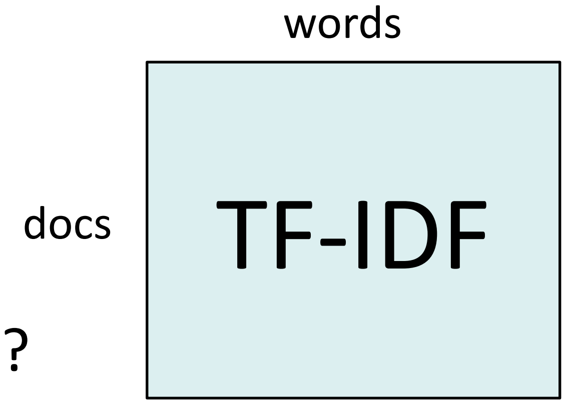
SCIENCE • VOL. 272 • 24 MAY 1996

Topic inference in LDA

- LDA is unsupervised (topics come out “magically”)
- Input:
  - Docs represented as bags of words
  - Number *K* of topics
- Output:
  - *K* topics (distributions over words)
  - For each doc: distribution over *K* topics
- How is this done?
  - Find distributions (i.e., topics, docs) that maximize the likelihood of the observed documents (maximum likelihood)

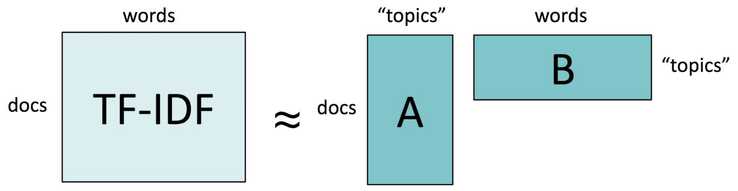
Question:

- "Which of these word pairs is more closely related?"
  - (car, bus)
  - (car, astronaut)
- How to quantify this?
- Detour:
  - How to quantify closeness of two docs?
  - E.g., cosine of **rows** of TF-IDF matrix
- Retour:
  - How to quantify closeness of two words?
  - E.g., cosine of **cols** of TF-IDF matrix



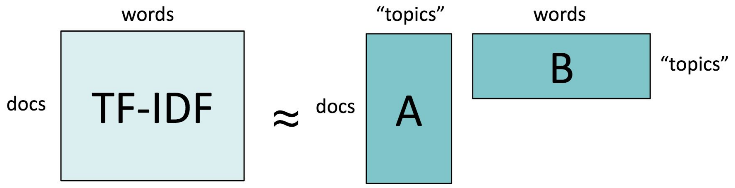
Sparsity in TF-IDF matrix

- Two docs (i.e., rows of TF-IDF matrix)
  - "Do you love men?"
  - "Adorest thou the likes of Adam?"
  - Cosine of row vectors of TF-IDF matrix == 0
- Same problem when comparing two words (i.e., cols of matrix)
- Solution:
  - Move from sparse to dense vectors
  - But how?
  - Latent semantic analysis (LSA)!
    - Use columns of *B* as dense vectors representing words

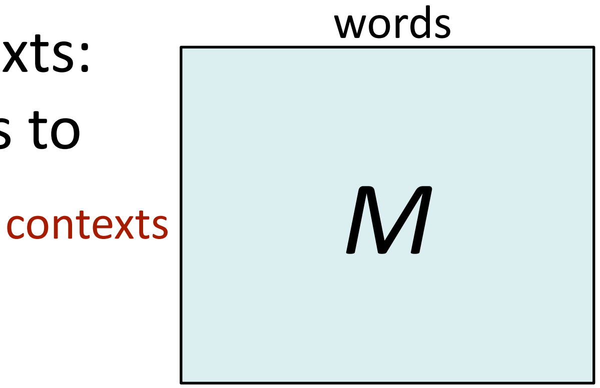


“Word vectors”

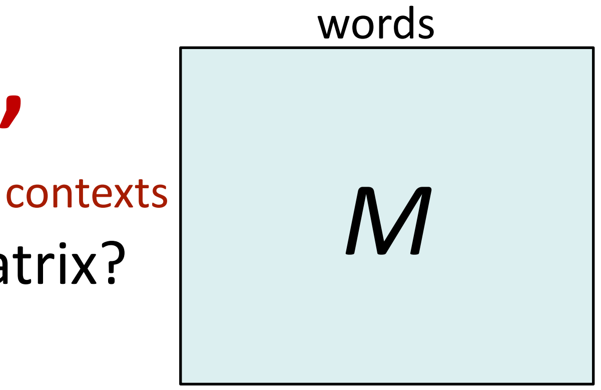
- Columns of TF-IDF matrix (sparse) or of word-by-topic matrix B (dense)



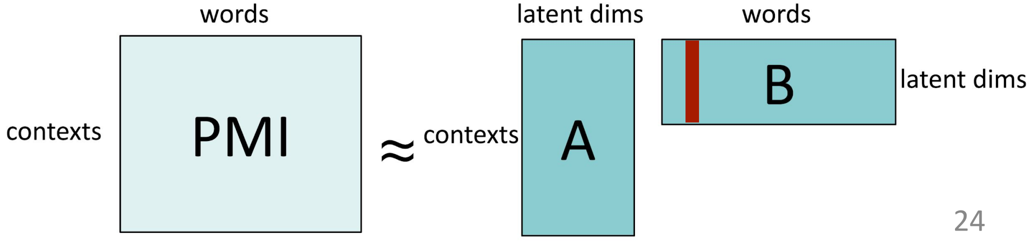
- Problem:
  - Entire doc treated as one bag of words
  - All information about word proximity, syntax, etc., is lost
- Solution:
  - Instead of full docs, consider local contexts: windows of *L* (e.g., 3) consecutive words to left and right of the target word
  - Rows of matrix: not docs, but contexts



“Word vectors”



- What to use as entries of word/context matrix?
- Straightforward: same as TF-IDF, but with contexts as “pseudo-docs”: $M[c,w]$ = TF-IDF(c,w)
- May use any other measures of statistical association
- E.g., pointwise mutual information (PMI):
  $M[c,w] = \mathrm{PMI}(c,w) = \log \frac{\Pr(c,w)}{\Pr(c)\ \Pr(w)}$
  “How much more likely are *c* and *w* to occur together than if they were independent?”
- word2vec: factor PMI matrix and use columns of *B* as word vectors



Beyond bags of words

From words to texts

- Word vectors represent, well, words
- How to represent larger units, such as sentences, paragraphs, docs?
- Typical approach: take sum/average of word vectors
- Note: this is roughly also what bags of words are (when using "one-hot" encoding for words, i.e., vector with exactly one 1, rest 0)
- More recently: **learn** vectors for longer units
    - Cr5, sent2vec
    - Convolutional neural networks
    - Recurrent neural networks, e.g., LSTM, ELMo
    - Transformer-based models, e.g., BERT (next slides), GPT-*

Contextualized word vectors

- Motivating example:
  - “The bat flew into the cave.”
  - vs. “He swung the bat and hit a home run.”

- Classic word vectors (e.g., word2vec)
  cannot distinguish these two cases;
  same vector used for both instances of “bat”

- Solution: contextualized word vectors
  - E.g., BERT

BERT in a nutshell

- Introduced in 2018 by Google Research

<START>
the
bat
flew
…

[1.00,0.70,0.90,0.50,0.06,…]
[0.54,0.75,0.56,0.45,0.09,…]
[0.44,0.76,0.77,0.31,0.82,…]
[0.91,0.62,0.53,0.75,0.74,…]
[0.92,0.37,0.25,0.49,0.24,…]

doc vector

context-
ualized
word
vectors

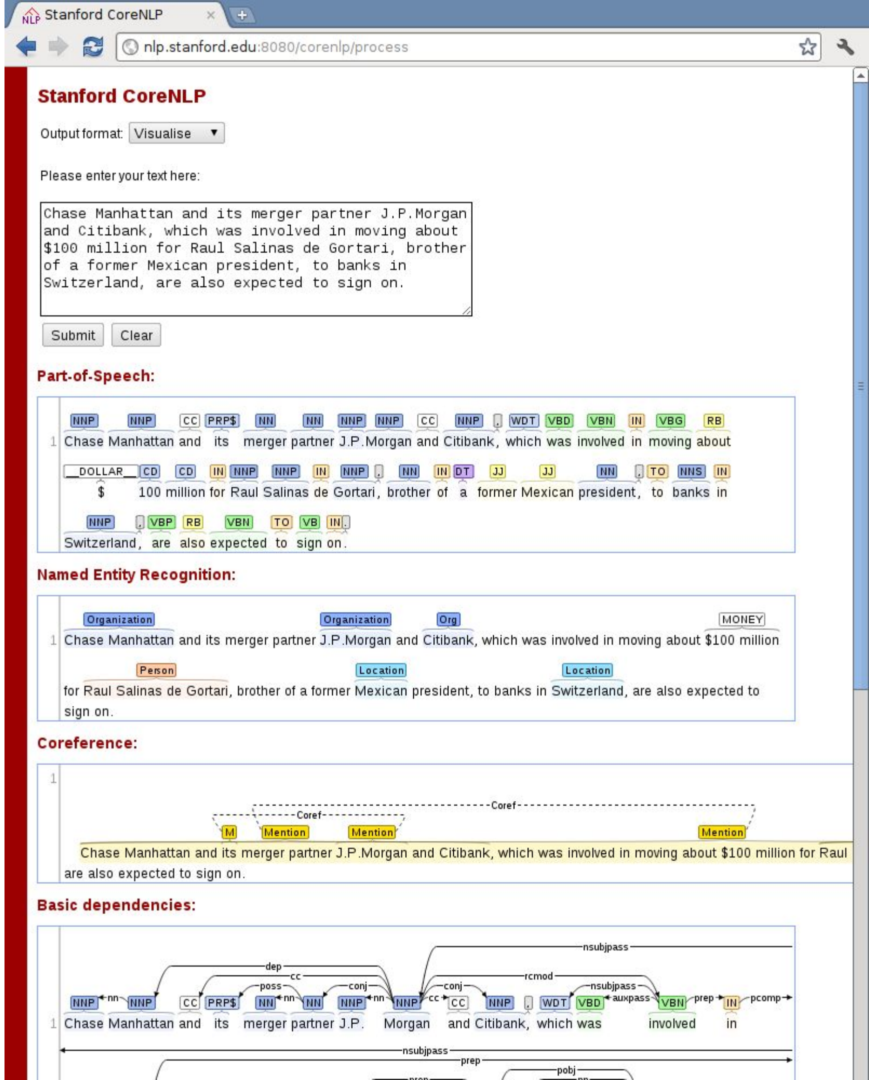
Inside the black box: some nasty neural network

<START>
he
swung
the
bat
…

[0.85,0.62,0.71,0.11,0.58,…]
[0.49,0.25,0.22,0.36,0.75,…]
[0.61,0.87,0.73,0.96,0.52,…]
[0.58,0.02,0.01,0.92,0.76,…]
[0.53,0.42,0.64,0.26,0.01,…]

NLP pipeline

- Tokenization
- Sentence splitting
- Part-of-speech (POS) tagging
- Named-entity recognition (NER)
- Coreference resolution
- Parsing
  - Shallow parsing (a.k.a. chunking)
  - Constituency parsing
  - Dependency parsing



Stanford CoreNLP

nlp.stanford.edu:8080/corenlp/process

Stanford CoreNLP

Output format: Visualise

Please enter your text here:

```
Chase Manhattan and its merger partner J.P.Morgan
and Citibank, which was involved in moving about
$100 million for Raul Salinas de Gortari, brother
of a former Mexican president, to banks in
Switzerland, are also expected to sign on.
```

Submit Clear

Part-of-Speech:

1 Chase Manhattan and its merger partner J.P.Morgan and Citibank , which was involved in moving about $ 100 million for Raul Salinas de Gortari , brother of a former Mexican president , to banks in Switzerland , are also expected to sign on .

NNP NNP CC PRP$ NN NN NNP NNP CC NNP , WDT VBD VBN IN VBG RB
DOLLAR CD CD IN NNP NNP IN NNP , NN IN DT JJ JJ NN , TO NNS IN
NNP , VBP RB VBN TO VB IN .

Named Entity Recognition:

1 Chase Manhattan and its merger partner J.P.Morgan and Citibank, which was involved in moving about $100 million for Raul Salinas de Gortari, brother of a former Mexican president, to banks in Switzerland, are also expected to sign on.

Coreference:

1 Chase Manhattan and its merger partner J.P.Morgan and Citibank, which was involved in moving about $100 million for Raul
are also expected to sign on.

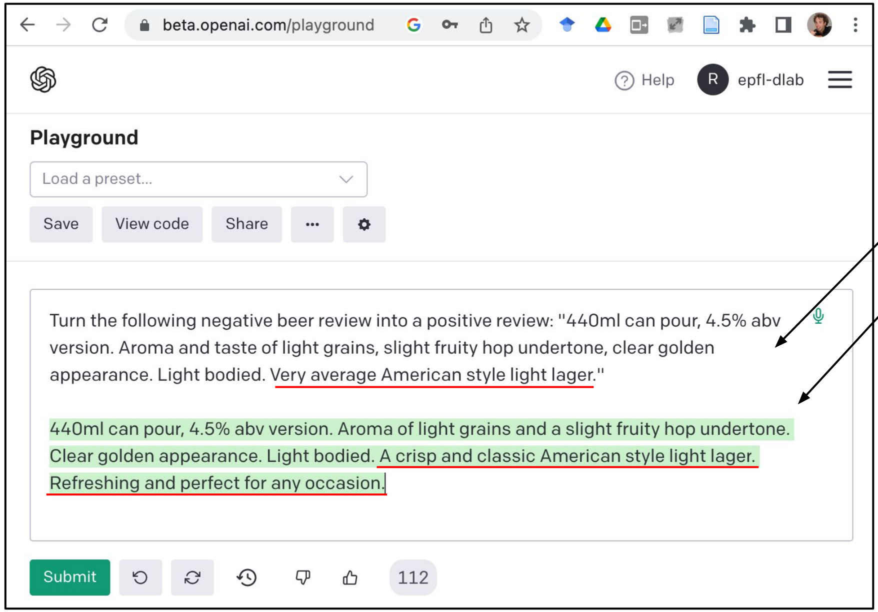
Basic dependencies:

1 Chase Manhattan and its merger partner J.P. Morgan and Citibank, which was involved in

NLP pipeline

- Implemented by Stanford CoreNLP, nltk, spaCy, etc.
- Sequential model
  - Fixed order of steps
  - Early errors will propagate downstream
  - Fixed order not optimal for all cases (e.g., syntax usually done before semantics, but semantics might be useful for inferring syntax)
- Hence, current research: learn all tasks jointly (early example)
- To learn how all this magic is implemented
  - Take CS-431 (Intro to NLP), CS-552 (Modern NLP)

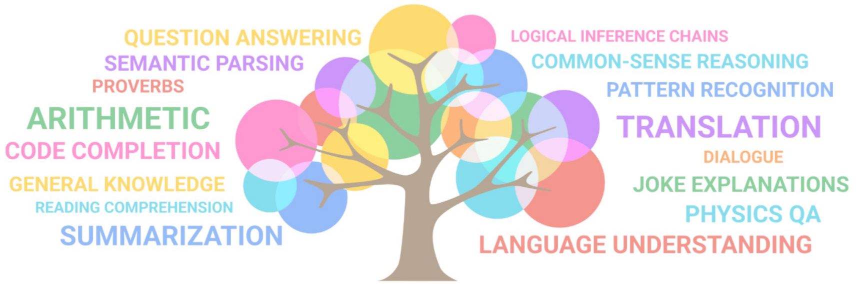
Today's trend: generative language models



Turn the following negative beer review into a positive review: "440ml can pour, 4.5% abv version. Aroma and taste of light grains, slight fruity hop undertone, clear golden appearance. Light bodied. Very average American style light lager."

440ml can pour, 4.5% abv version. Aroma of light grains and a slight fruity hop undertone. Clear golden appearance. Light bodied. A crisp and classic American style light lager. Refreshing and perfect for any occasion.

- E.g., OpenAI GPT-*, ChatGPT
- Input: text
- Output: text
- Many NLP tasks can be formulated in this framework, by "prompting" the language model with the right input



Feedback

Give us feedback on this lecture here:
https://go.epfl.ch/ada2025-lec11-feedback

- What did you (not) like about this lecture?
- What was (not) well explained?
- On what would you like more (fewer) details?
- …