Maschinelle Sprachverarbeitung

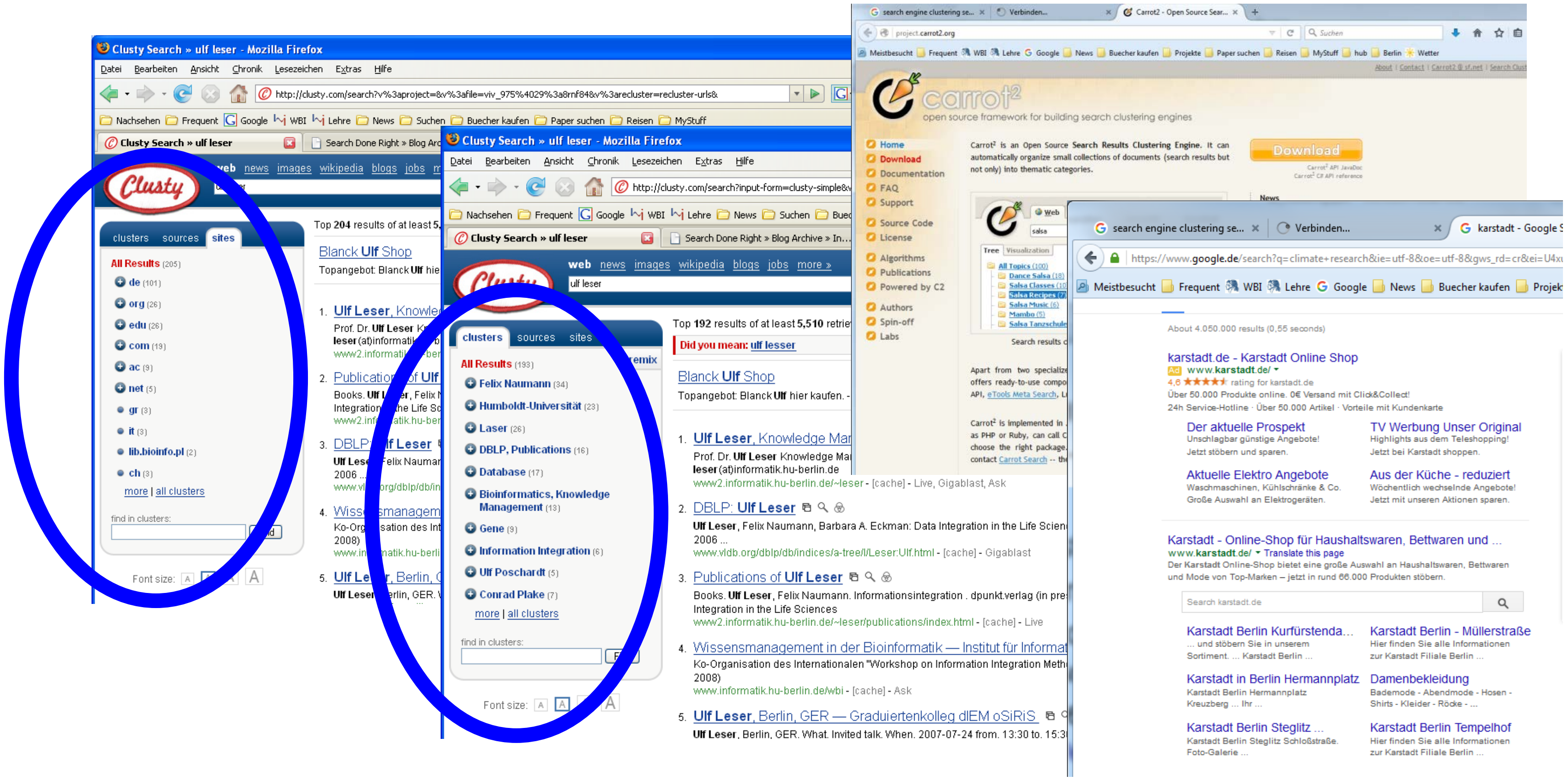
Text Clustering

Ulf Leser

Content of this Lecture

- (Text) clustering
- Cluster quality
- Clustering algorithms
- Application

Processing Search Results


Clusty Search » ulf leser - Mozilla Firefox

Top 204 results of at least 5,...

clusters sources sites

All Results (205)
- de (101)
- org (26)
- edu (26)
- com (19)
- ac (9)
- net (5)
- gr (3)
- it (3)
- lib.bioinfo.pl (2)
- ch (3)

more | all clusters

Clusty Search » ulf leser - Mozilla Firefox

Top 192 results of at least 5,510 retrie...

Did you mean: ulf lesser

clusters sources sites

All Results (193)
- Felix Naumann (34)
- Humboldt-Universität (23)
- Laser (26)
- DBLP, Publications (16)
- Database (17)
- Bioinformatics, Knowledge Management (13)
- Gene (9)
- Information Integration (6)
- Ulf Poschardt (5)
- Conrad Plake (7)

more | all clusters

Blanck Ulf Shop
Topangebot: Blanck Ulf hier kaufen.

1. Ulf Leser, Knowledge Mar...
Prof. Dr. Ulf Leser Knowledge Ma... leser (at)informatik.hu-berlin.de
www2.informatik.hu-berlin.de/~leser - [cache] - Live, Gigablast, Ask

2. DBLP: Ulf Leser
Ulf Leser, Felix Naumann, Barbara A. Eckman: Data Integration in the Life Scien... 2006 ...
www.vldb.org/dblp/db/indices/a-tree/l/Leser:Ulf.html - [cache] - Gigablast

3. Publications of Ulf Leser
Books. Ulf Leser, Felix Naumann. Informationsintegration . dpunkt.verlag (in pre... Integration in the Life Sciences
www2.informatik.hu-berlin.de/~leser/publications/index.html - [cache] - Live

4. Wissensmanagement in der Bioinformatik — Institut für Informat...
Ko-Organisation des Internationalen "Workshop on Information Integration Meth... 2008)
www.informatik.hu-berlin.de/wbi - [cache] - Ask

5. Ulf Leser, Berlin, GER — Graduiertenkolleg dIEM oSiRiS
Ulf Leser, Berlin, GER. What. Invited talk. When. 2007-07-24 from. 13:30 to. 15:3...

Carrot2 - Open Source Sear...

carrot2 open source framework for building search clustering engines

Carrot² is an Open Source Search Results Clustering Engine. It can automatically organize small collections of documents (search results but not only) into thematic categories.

Download

karstadt - Google S...

About 4.050.000 results (0,55 seconds)

karstadt.de - Karstadt Online Shop
www.karstadt.de/
Über 50.000 Produkte online. 0€ Versand mit Click&Collect!
24h Service-Hotline · Über 50.000 Artikel · Vorteile mit Kundenkarte

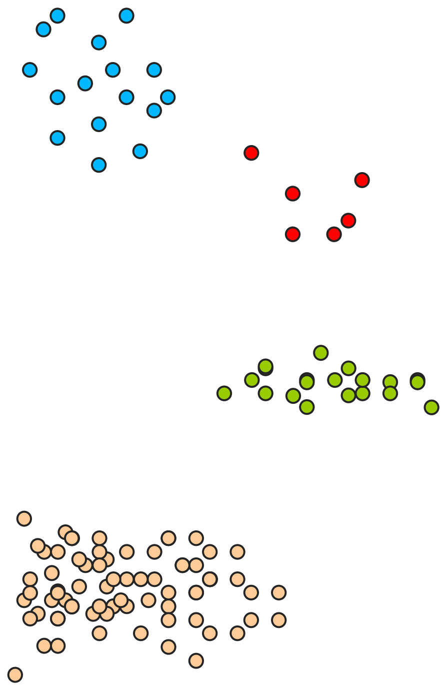
Karstadt - Online-Shop für Haushaltswaren, Bettwaren und ...
www.karstadt.de/ Translate this page
Der Karstadt Online-Shop bietet eine große Auswahl an Haushaltswaren, Bettwaren und Mode von Top-Marken – jetzt in rund 66.000 Produkten stöbern.

- “… The research breakthrough was labeling the clusters, i.e., grouping search results into folder topics …”
  – [Clusty.com blog]

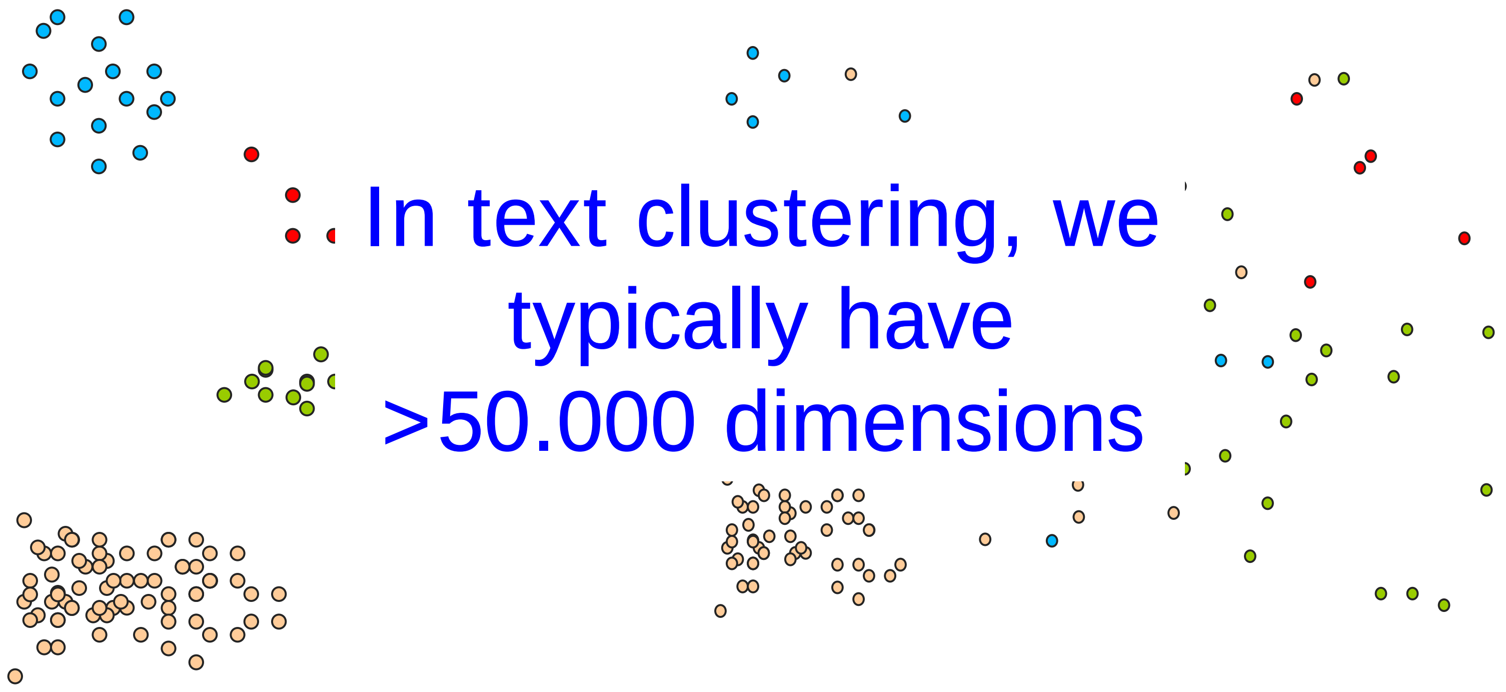
Clustering

- Clustering groups objects (docs) into (usually disjoint) sets
- Intuitively, each set should contain objects that are similar to each other and dissimilar to objects in any other set
  - We need a similarity or distance function
  - That is the only text-specific bit – the rest is “just” clustering
- Often called “unsupervised learning”
  - We don’t know how many sets/classes there are (if there are any)
  - We don’t know how those sets should look like

Nice



Nice – Not Nice



In text clustering, we typically have >50.000 dimensions

Text Clustering Applications

- Explorative data analysis
  - Learn about the structure within your document collection
- Corpus preprocessing
  - Clustering provides a “semantic index” to a corpus
  - Group docs into clusters to ease navigation
  - Retrieval speed: Index only one representative per cluster
- Processing of search results
  - Cluster all hits into groups of similar hits (in particular: duplicates)
- Improving recall
  - Return doc and all members of its cluster
  - Has similarity to automatic relevance feedback using top-k docs
- Word sense disambiguation
  - The different senses of a word should appear as clusters
- ...

Similarity between Documents

- Clustering requires a distance function
  - Should always be a metric
  - $d(x,x)=0$, $d(x,y)=d(y,x)$, $d(x,y) \leq d(x,z)+d(z,y)$
- In contrast to search, we compare two docs
  - And not a document and a query
- Nevertheless, often the same methods are used
  - Vector space , TF*IDF, cosine distance

$$sim(d_1, d_2) = \cos(d_1, d_2) = \frac{d_1 \circ d_2}{|d_1| * |d_2|} = \frac{\sum \left(d_1[i] * d_2[i]\right)}{\sqrt{\sum d_1[i]^2} * \sqrt{\sum d_2[i]^2}}$$

Clustering Speed

- In Information Retrieval
  - We compare a vector of 100K dimensions with ~3 non-null values (query) with one with ~500 non-null values
  - Use inverted files to pick docs that have an overlap with the query
- In clustering
  - We compare a vector with ~500 nnv with one with ~500 nnv
  - We need to compare many (all) docs with many (all) docs
    - Depends on the clustering algorithm
  - Inverted files offer much less speed-up
- Feature selection or dimensionality reduction is essential
  - E.g., use the 1.000 “most descriptive” terms
  - E.g., perform Latent Semantic Indexing (LSI) before clustering

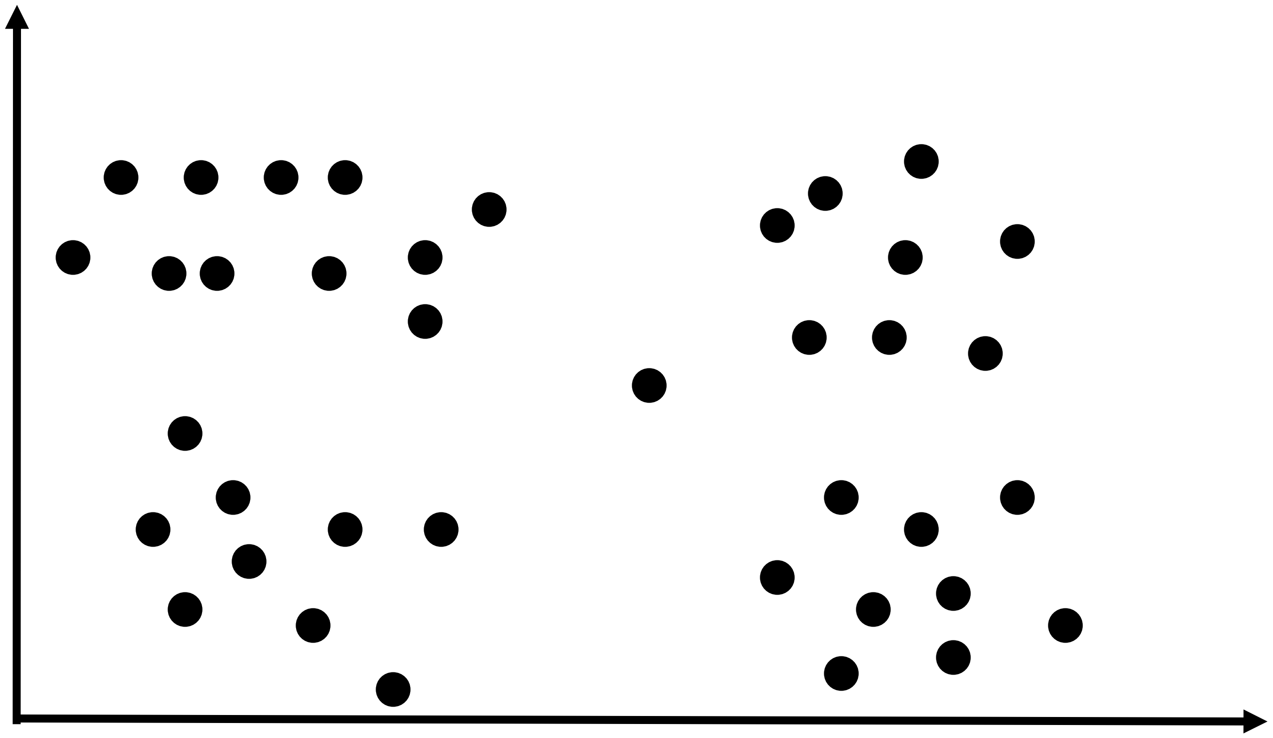
Cluster Labels

- For user interaction, clusters need to have a name
- Names should capture the topic (semantic) of the cluster
- Some possibilities
  - Chose term with highest TF*IDF value in cluster
    - E.g. TF computed as average or considering all docs in cluster as one
  - Chose term with highest TF*IDF value in cluster centre
  - Apply statistical test to find terms whose TF*IDF distribution deviates the most between clusters
    - E.g: t-Test (assuming normal distribution), Kullback–Leibler divergence
    - Requires comparison of each cluster with each cluster for each term
    - Only possible when strict pre-filtering was applied
  - Report top-K token or top-K terms (by whatever method)
  - ...

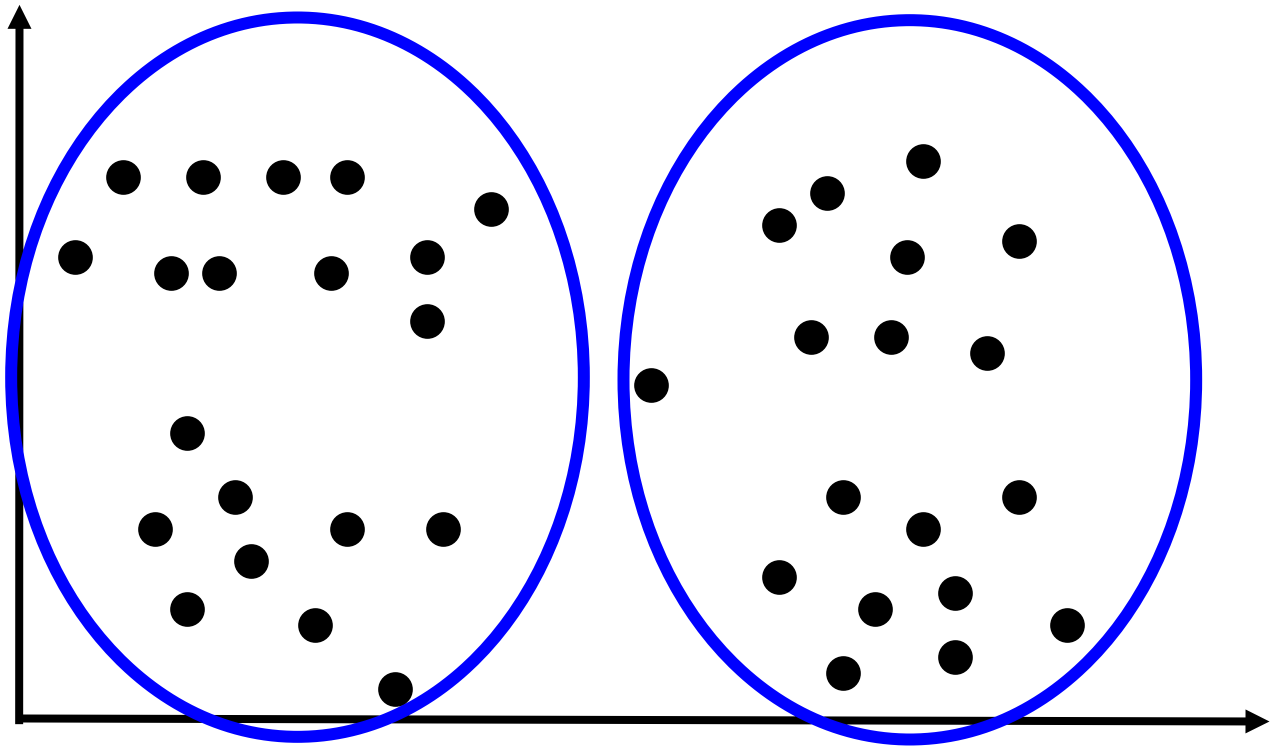
Content of this Lecture

- (Text) clustering
- Cluster quality
- Clustering algorithms
- Application

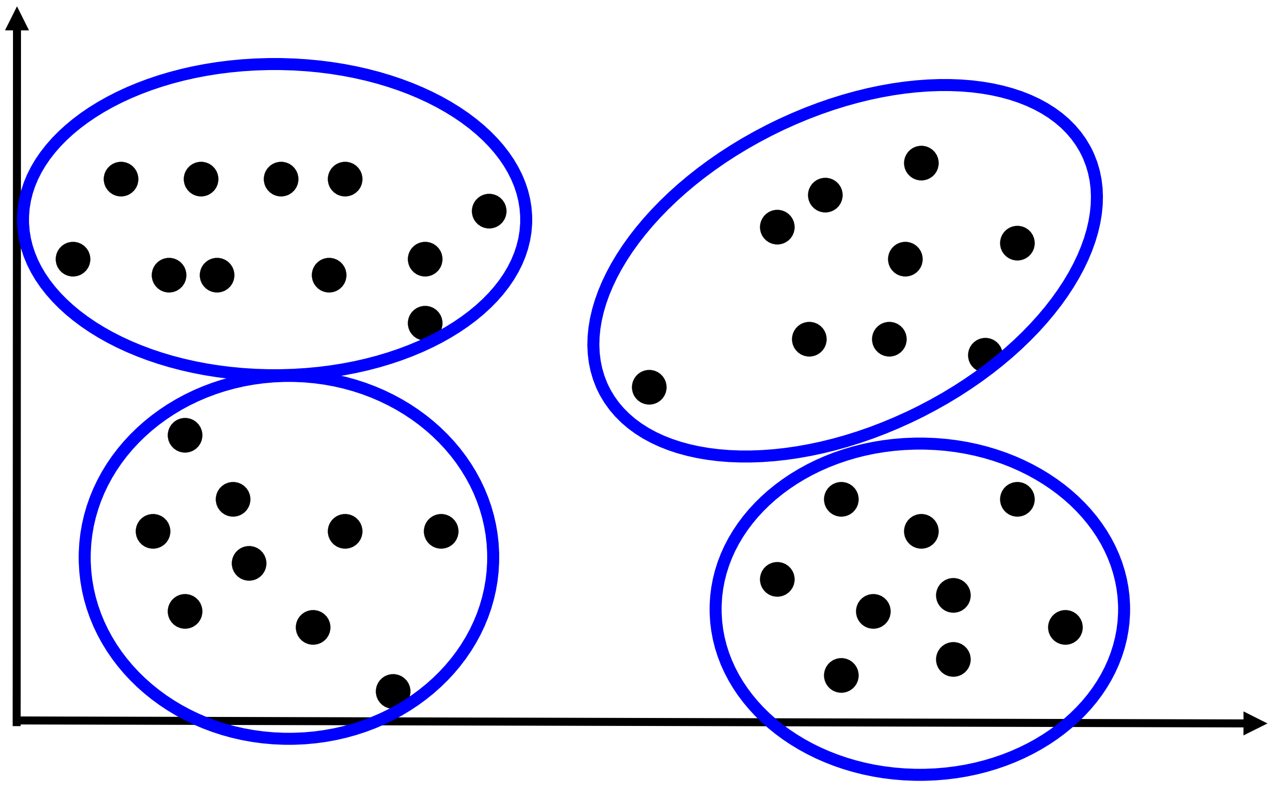
How many Clusters?



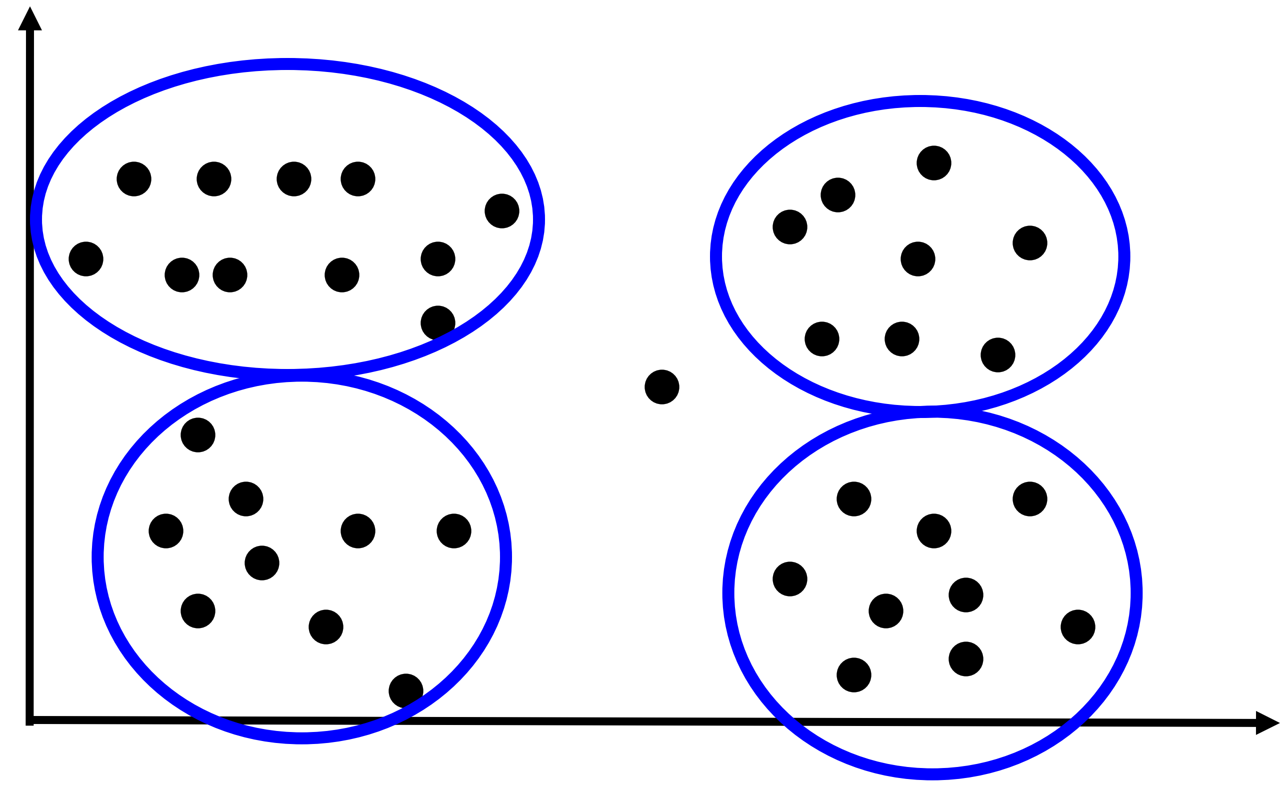
Maybe 2?



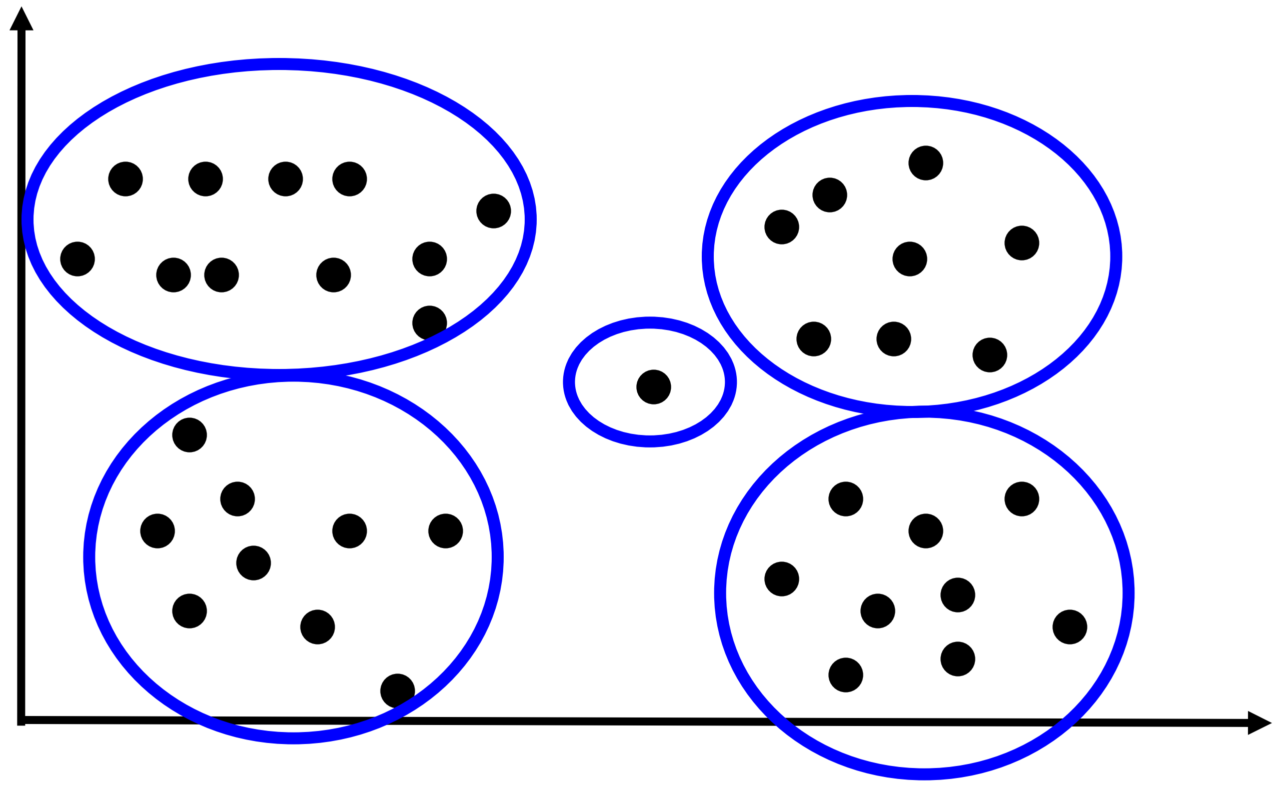
Maybe 4?



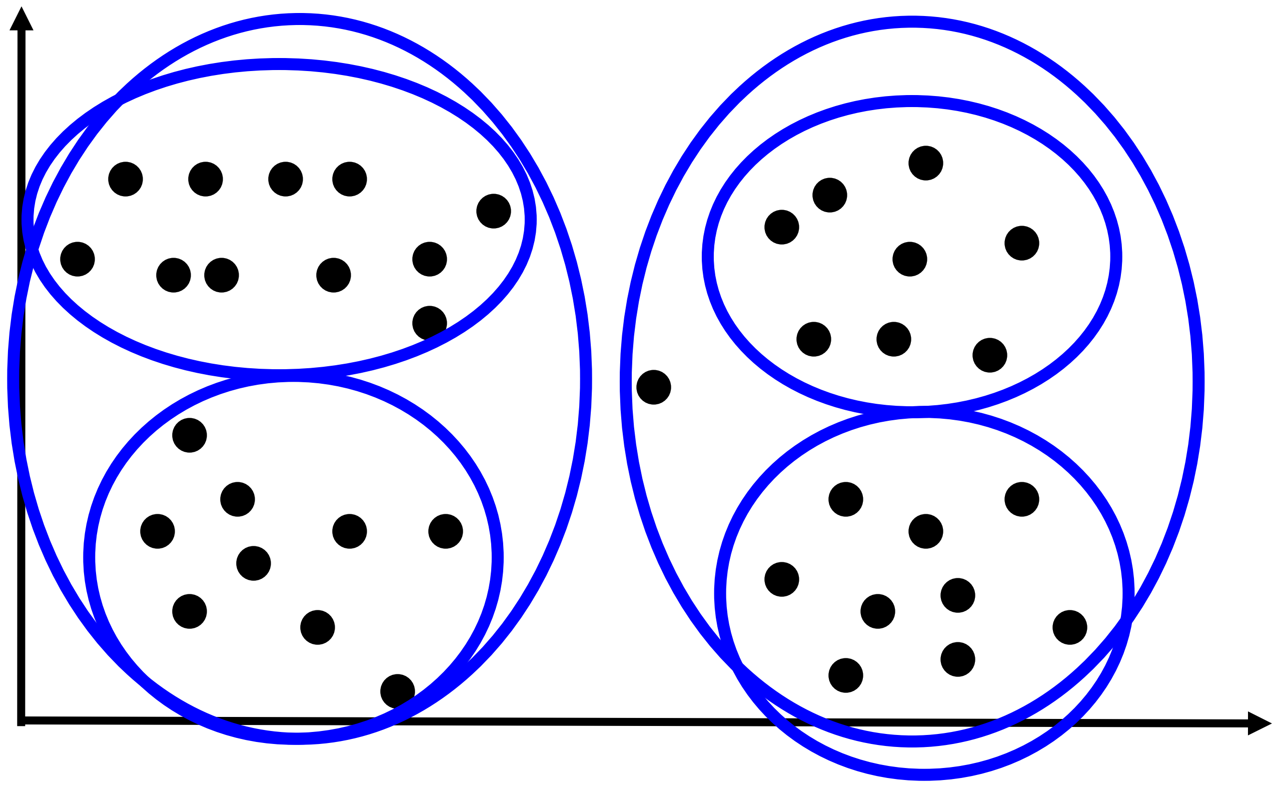
Maybe 4 and One Outlier?



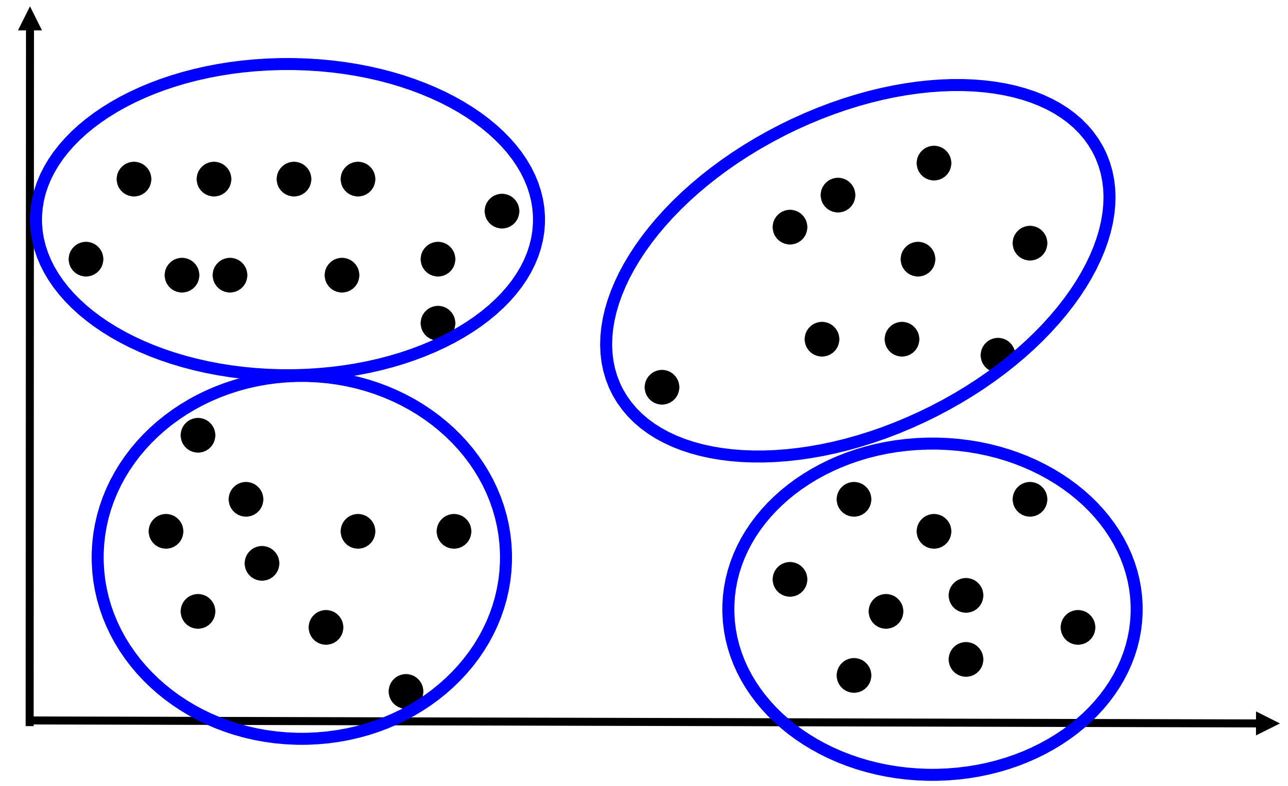
Maybe 5?



Maybe 4 and 2 – at Different Levels?



Which Distance Measure did you Use?



Quality of a Clustering

- There is no “true” number of clusters
- In real data sets, one cannot determine the number of clusters by “looking at the data”
  - Too many dimensions
  - Clustering should help you in looking at the data
- We need to define the quality of a clustering
- Ideally, this quality score peaks at the intuitively best number of clusters

Distance to a Cluster

- We frequently will have to compute the distance between a point o and a cluster c, d(o,c)
  - And sometimes distances between clusters – see hier. clustering
- Various methods
  - Distance to numerical center of a cluster
  - Distance to the most central point of a cluster
  - Average distance to all points in cluster

$$d(o,c) = d(o, c_{mean})$$

$$d(o,c) = d(o, c_{median})$$

$$d(o,c) = \sum_{p \in c} d(o,p) * {}^{1}\!/_{|c|}$$

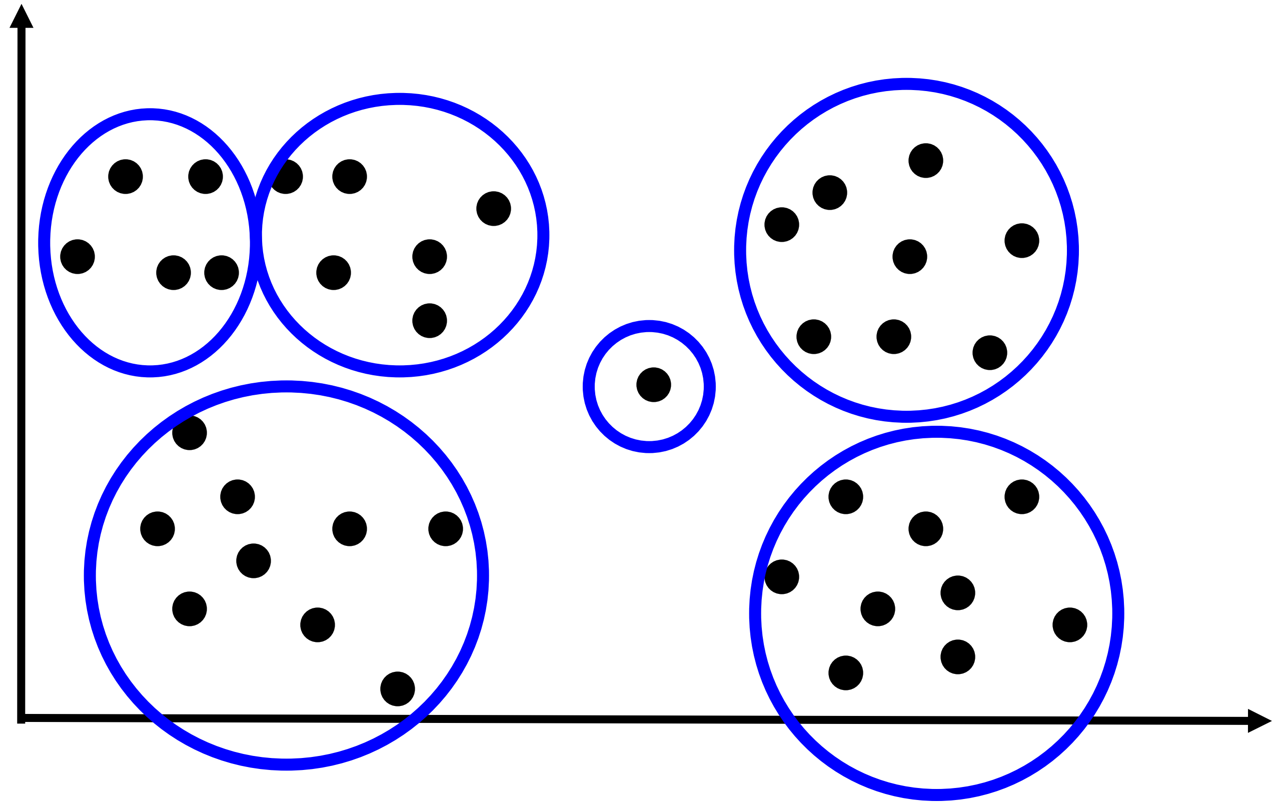
Quality of a Clustering – First Approach

- Compute average distance between its objects
- Definition
  Let f be a clustering of a set of objects O into a set C of classes with |C|=k. The k-score q_k of f is

$$q_k(f) = \sum_{i=1..k} \sum_{f(o)=c_i} d(o, c_i)$$

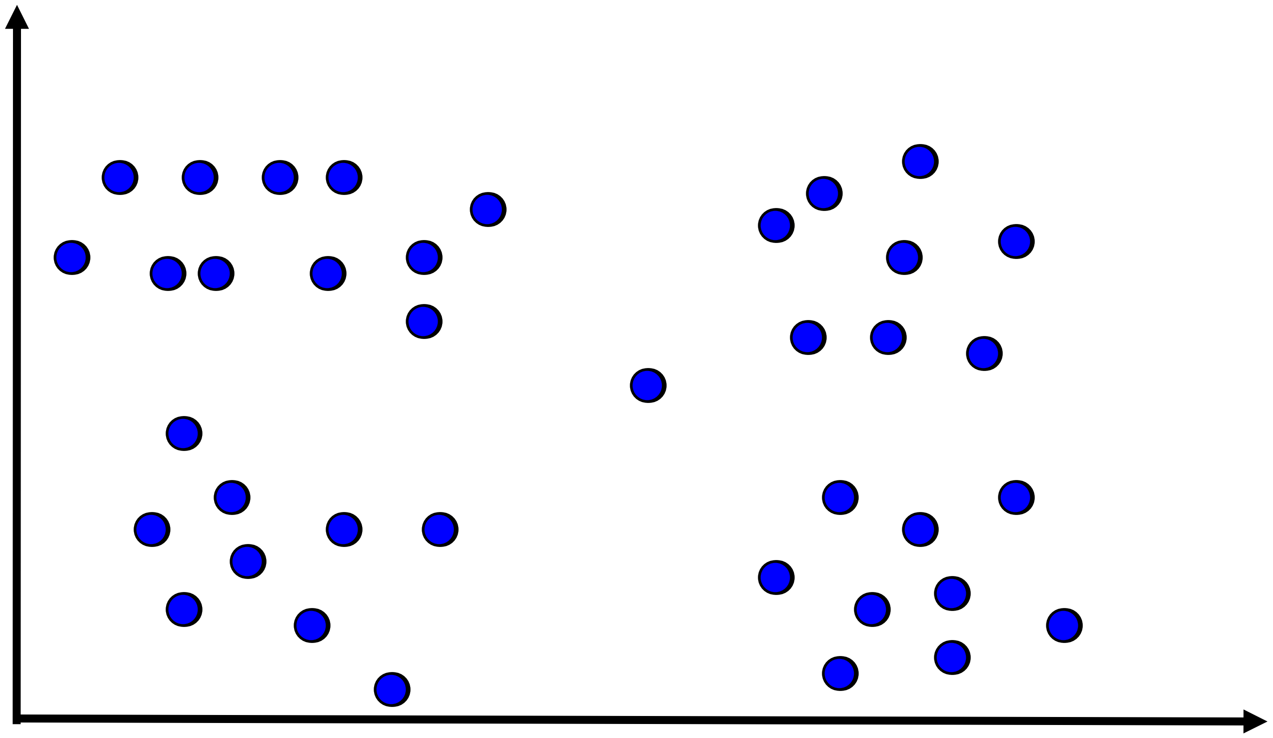
- Any measure for point-to-cluster distance may be used

6-Score



- Certainly better than the 2/4/5-score we have seen
- Thus: Chose the k with the best k-score?

Disadvantage



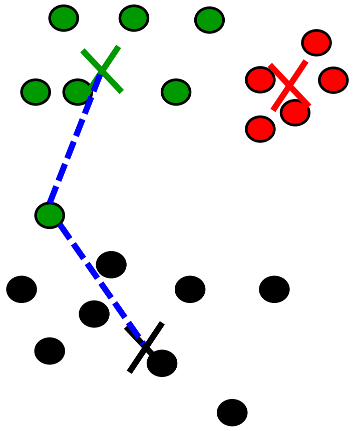
- Always has a trivially optimal solution: k=|O|
- Points in a cluster should be close to each other but also far away from points in other clusters
- Still useful to compare different clusterings for the same k

Silhouette

- Alternative: Silhouette of a clustering
  - Punish points that are not “uniquely” assigned to one cluster
  - Captures how clearly points are part of their cluster
- Definition
  Let $f: O \rightarrow C$ with $|C|$ arbitrary. We define
  - *Inner score: $a(o) = d(o, f(o))$*
  - *Outer score: $b(o) = \min(d(o,c_i))$ with $c_i \neq f(o)$*
  - *The silhouette of o, s(o), is defined as* $$s(o) = \frac{b(o) - a(o)}{\max(a(o), b(o))}$$
  - *The silhouette of f, s(f), is defined as* $$s(f) = \sum s(o)$$

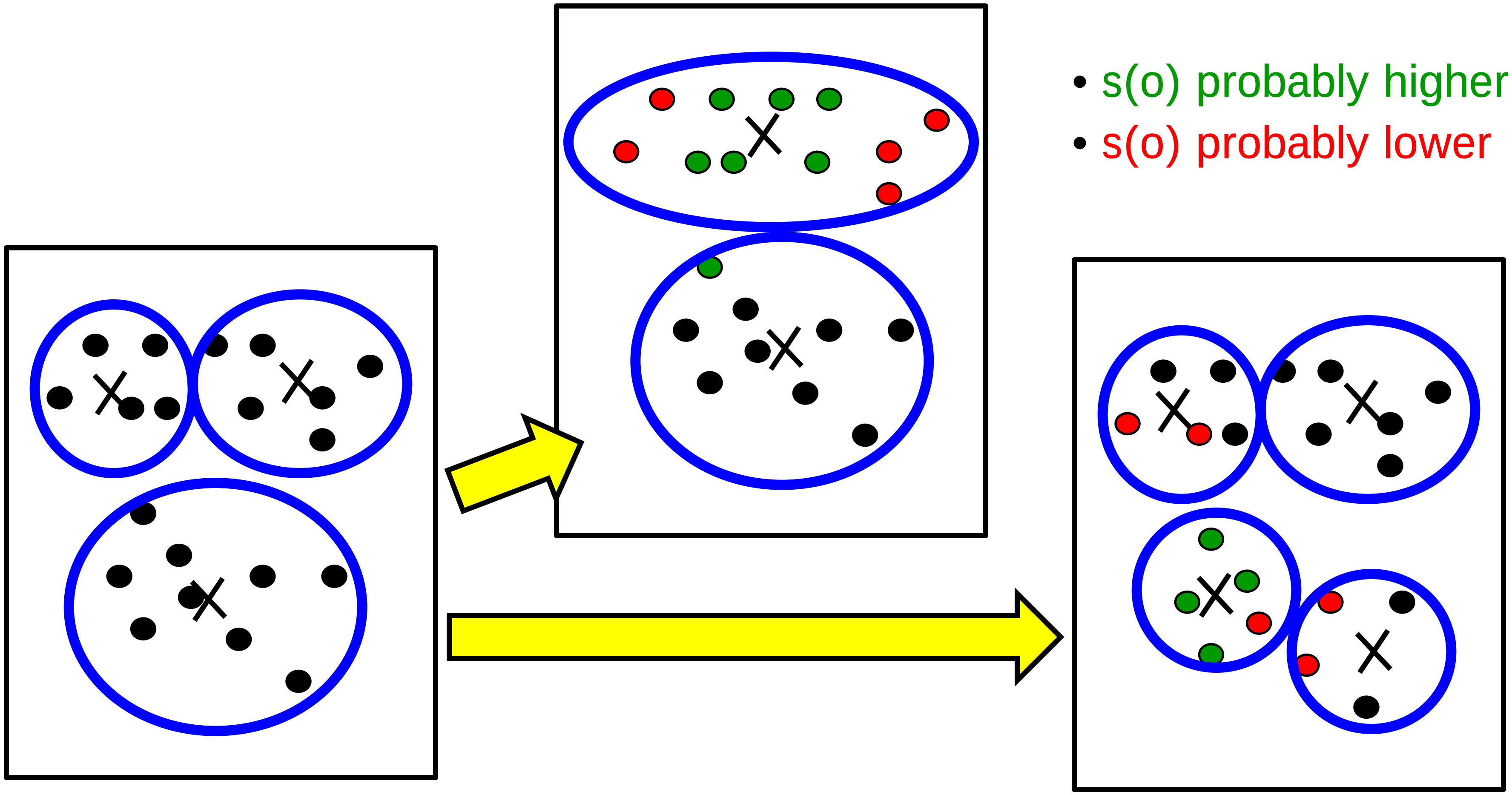
Intuition

- It holds: -1 ≤ s(o) ≤1
  - s(o) ≈ 0: Point right between two cluster
  - s(o) ~ 1: Point very close to only one (its own) cluster
  - s(o) ~ -1: Point far away from its own cluster
- Caution: Computing the silhouette is in O(nm+km)
  - If clusters are represented by centroids



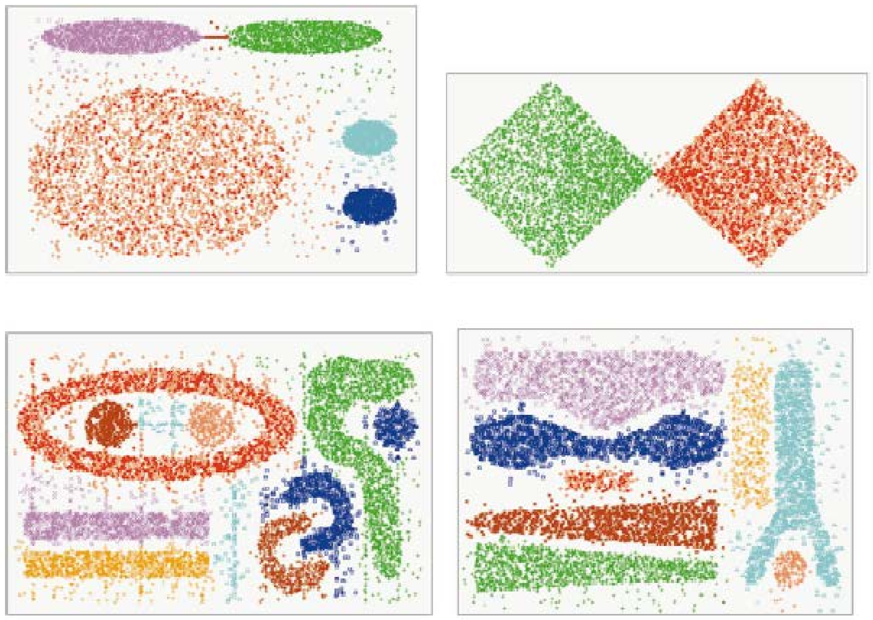
Behavior

- Silhouette is not always better / worse for more clusters



Not the End

- In general, clusters need not be hyper-spheres
- Clusters need not even have convex shapes
- Cluster centre need not be part of a cluster
- Requires completely different quality metrics
- Definition must fit to the data/application
- Not used in text clustering
  - To my knowledge



Source: [FPPS96]

Content of this Lecture

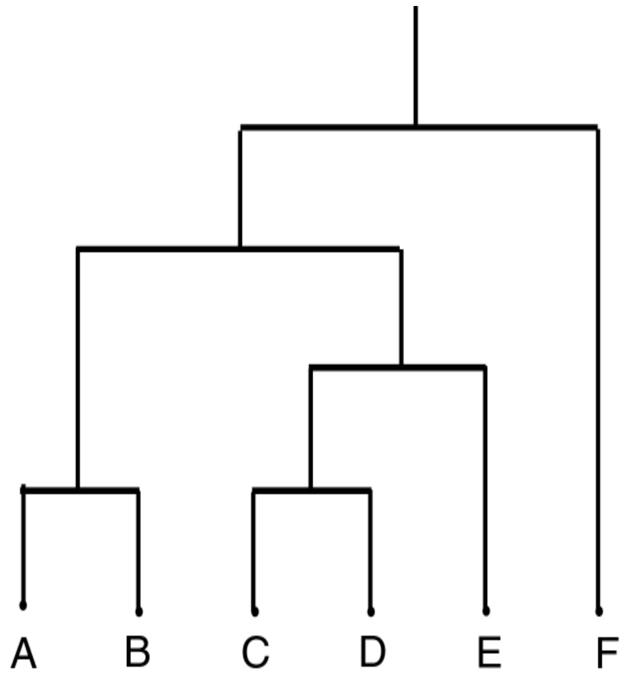
- Text clustering
- Cluster quality
- Clustering algorithms
  - Hierarchical clustering
  - K-means
  - Soft clustering: EM algorithm
- Application

Classes of Cluster Algorithms

- Hierarchical clustering
  - Iteratively creates a hierarchy of clusters
  - Bottom-Up: Start from |O| cluster and merge until only 1 remains
  - Top-Down: Start from one cluster and split
  - (... or until some stop criterion is met)
- Partitioning
  - Heuristically partition all objects in k clusters
  - Guess a first partitioning and improve iteratively
  - k is a parameter of the method, not a result
- Other
  - Graph-Theoretic: Min-Cut (partitioning) etc.
  - Density-base clustering
  - ...

Hierarchical Clustering

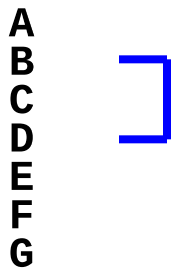
- Also called UPGMA: Unweighted Pair-group method with arithmetic mean
- Computes a binary tree (dendrogram)
- Simple algorithm
  - Compute distance matrix M
  - Choose pair d_1, d_2 with smallest distance
  - Define x as centre point of d_1 and d_2
    - Coordinates need not be computed
  - Remove d_1, d_2 from M
  - Insert x into M
    - Distance between x and any d in M: Average distance between d_1 and d and d_2 and d
  - Loop until M has size 2x2



Example

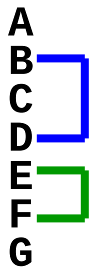
```
 ABCDEFG
A
B.
C..
D...
E....
F.....
G......
```

$\boxed{(B,D) \rightarrow a}$



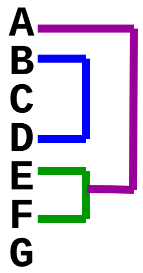
```
 ACEFGa
A
C.
E..
F...
G....
a.....
```

$\boxed{(E,F) \rightarrow b}$



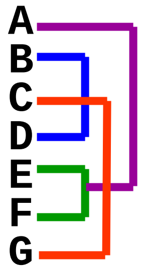
```
 ACGab
A
C.
G..
a...
b....
```

$\boxed{(A,b) \rightarrow c}$



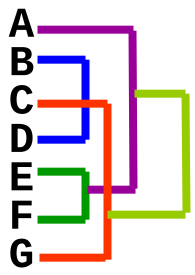
```
 CGac
C
G.
a..
c...
```

$\boxed{(C,G) \rightarrow d}$



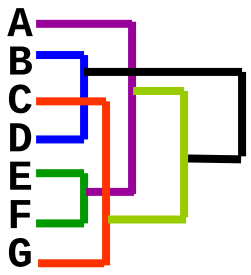
```
 acd
a
c.
d..
```

$\boxed{(d,c) \rightarrow e}$

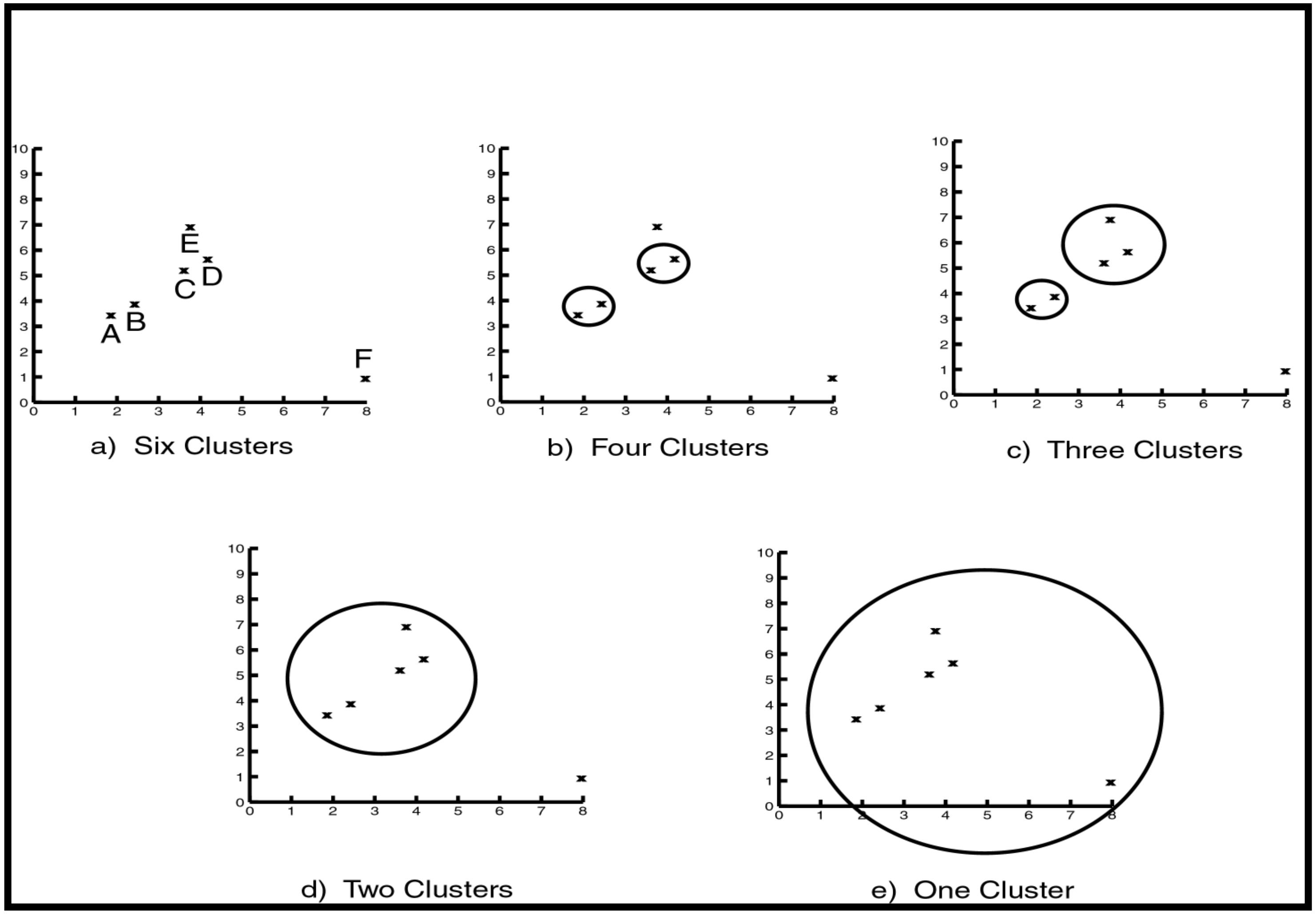


```
 ae
a
e.
```

$\boxed{(a,e) \rightarrow f}$

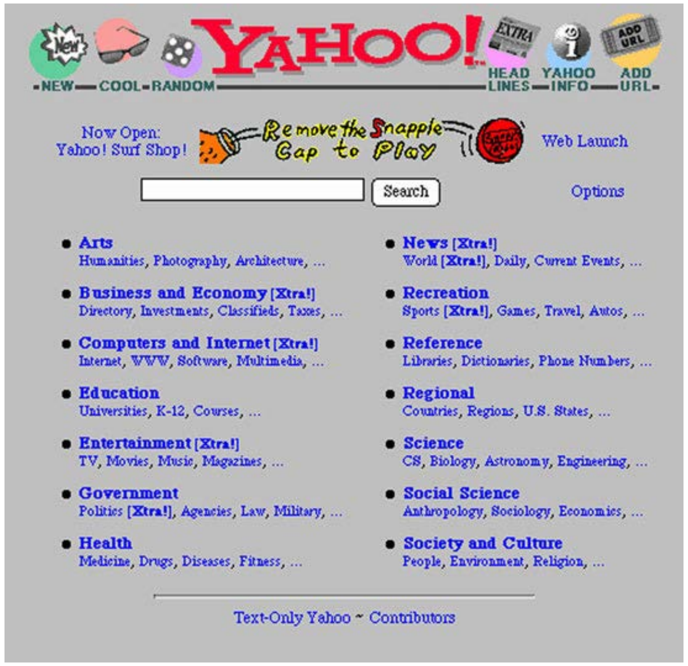


Visual


A
B
C
D
E
F
a) Six Clusters
b) Four Clusters
c) Three Clusters
d) Two Clusters
e) One Cluster

Intuition

- Hierarchical clustering organizes a doc collection
- Ideally, hierarchical clustering directly creates a hierarchical and intuitive directory of the corpus
- Not easy
  - Many, many ways to group objects – hierarchical clustering will choose just one
  - No guarantee that clusters make sense semantically

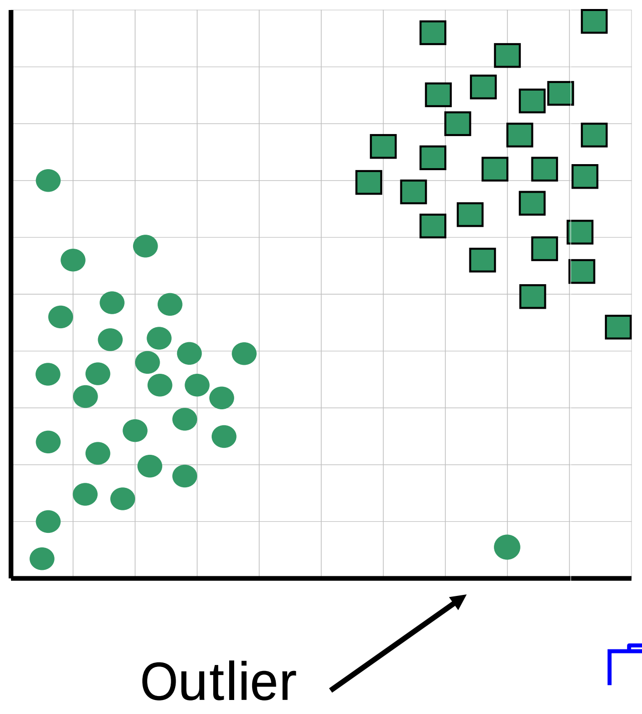


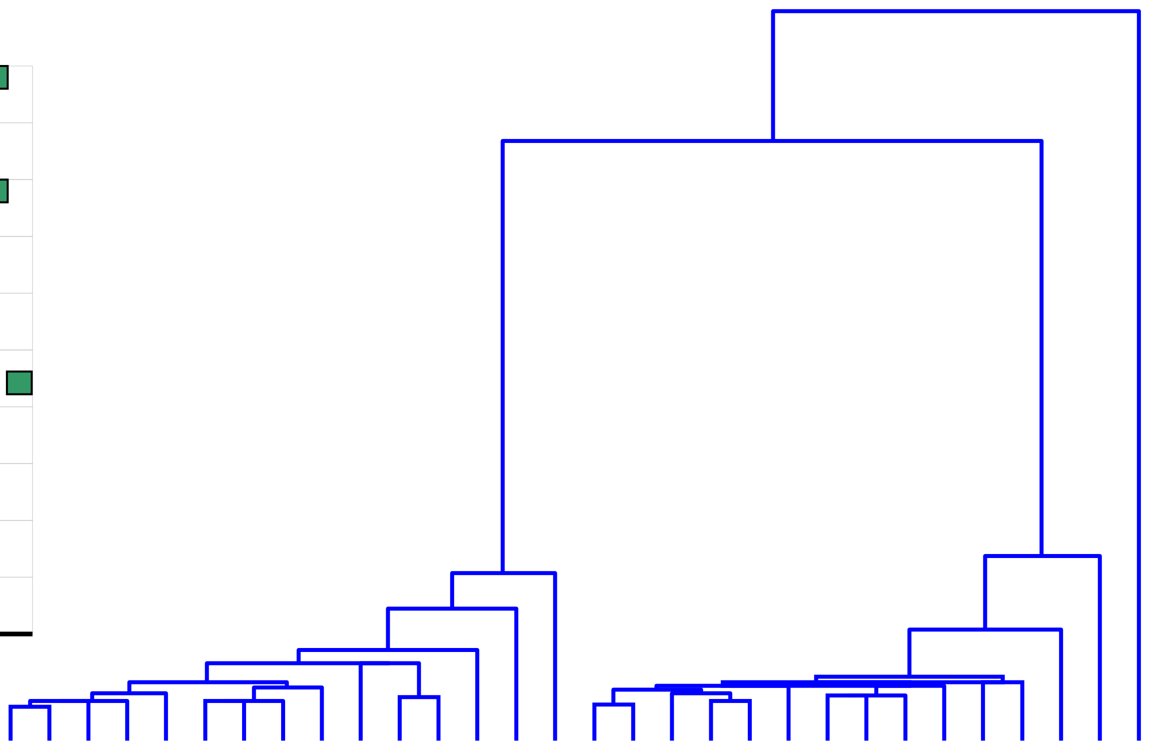
- **Arts**
  Humanities, Photography, Architecture, ...
- **Business and Economy** [Xtra!]
  Directory, Investments, Classifieds, Taxes, ...
- **Computers and Internet** [Xtra!]
  Internet, WWW, Software, Multimedia, ...
- **Education**
  Universities, K-12, Courses, ...
- **Entertainment** [Xtra!]
  TV, Movies, Music, Magazines, ...
- **Government**
  Politics [Xtra!], Agencies, Law, Military, ...
- **Health**
  Medicine, Drugs, Diseases, Fitness, ...
- **News** [Xtra!]
  World [Xtra!], Daily, Current Events, ...
- **Recreation**
  Sports [Xtra!], Games, Travel, Autos, ...
- **Reference**
  Libraries, Dictionaries, Phone Numbers, ...
- **Regional**
  Countries, Regions, U.S. States, ...
- **Science**
  CS, Biology, Astronomy, Engineering, ...
- **Social Science**
  Anthropology, Sociology, Economics, ...
- **Society and Culture**
  People, Environment, Religion, ...

Text-Only Yahoo - Contributors

Branch Length

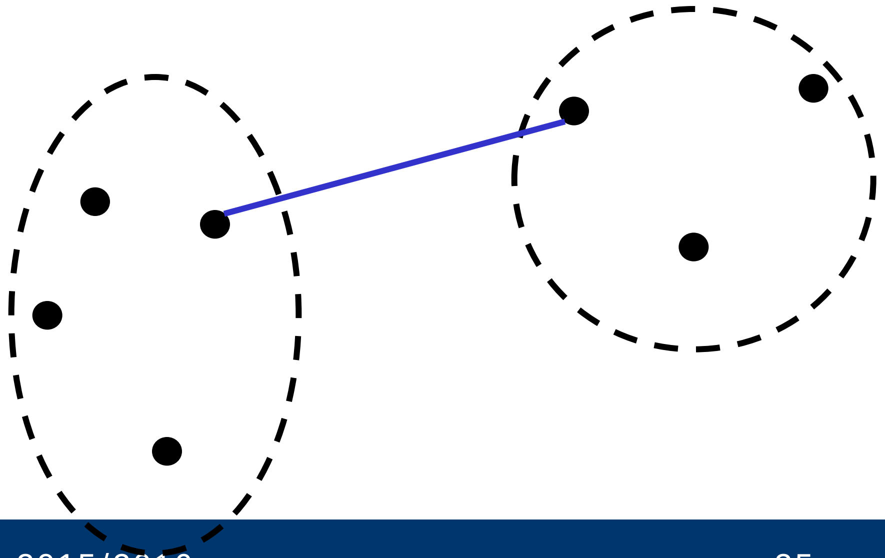
- Use branch length to symbolize distance
- Outlier detection





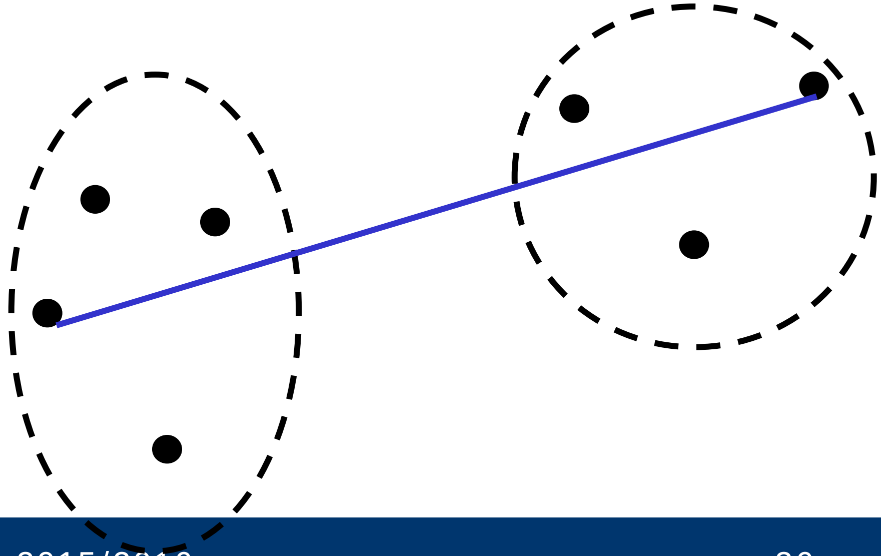
Variations

- We used the distance between the centers of two clusters to decide about distance between clusters
- Other alternatives (incurring different complexities)
  - Single Link: Distance of the two closest docs in both clusters
  - Complete Link: Distance of the two furthest docs
  - Average Link: Average distance between pairs of docs from both clusters
  - Centroid: Distance between centre points



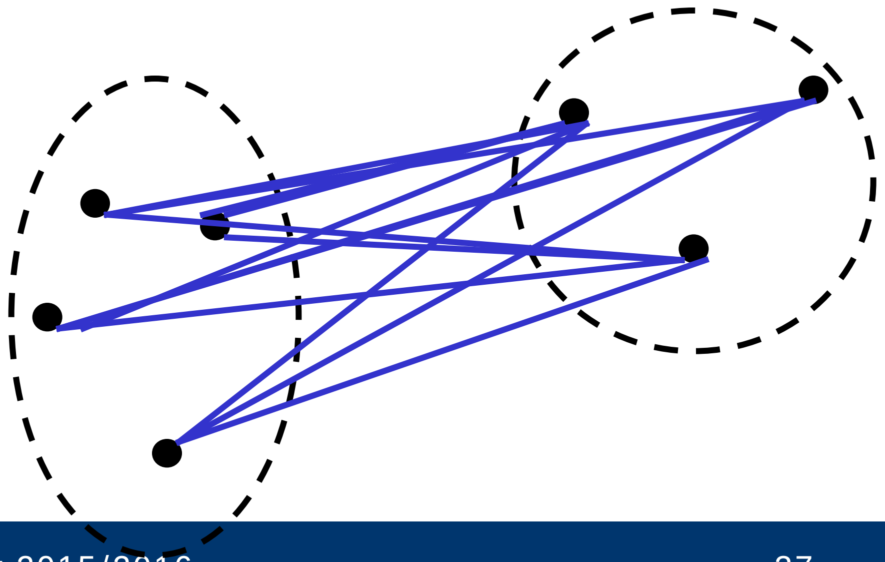
Variations

- We used the distance between the centers of two clusters to decide about distance between clusters
- Other alternatives (incurring different complexities)
  - Single Link: Distance of the two closest docs in both clusters
  - Complete Link: Distance of the two furthest docs
  - Average Link: Average distance between pairs of docs from both clusters
  - Centroid: Distance between centre points



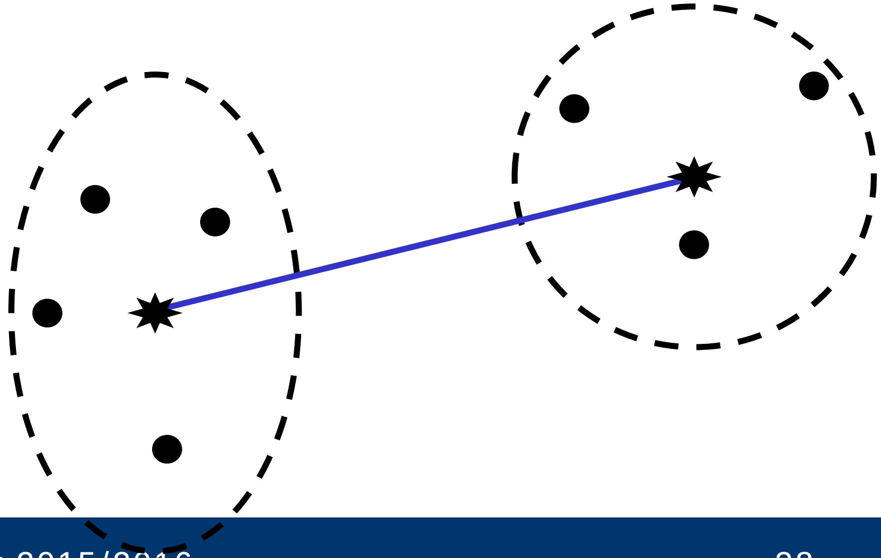
Variations

- We used the distance between the centers of two clusters to decide about distance between clusters
- Other alternatives (incurring different complexities)
  - Single Link: Distance of the two closest docs in both clusters
  - Complete Link: Distance of the two furthest docs
  - Average Link: Average distance between pairs of docs from both clusters
  - Centroid: Distance between centre points



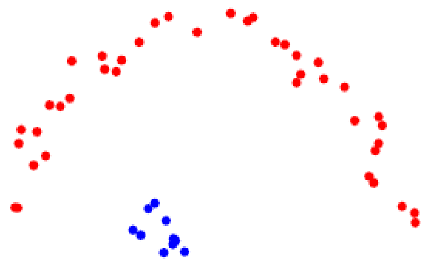
Variations

- We used the distance between the centers of two clusters to decide about distance between clusters
- Other alternatives (incurring different complexities)
  - Single Link: Distance of the two closest docs in both clusters
  - Complete Link: Distance of the two furthest docs
  - Average Link: Average distance between pairs of docs from both clusters
  - Centroid: Distance between centre points

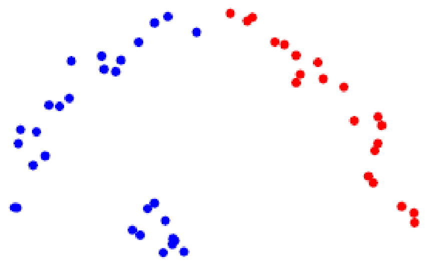


Comparison

- Single-link
  - Optimizes a local criterion (only look at the closest pair)
  - Similar to computing a minimal spanning tree
    - With cuts at most expensive branches as going down the hierarchy
  - Creates elongated clusters (chaining effect)
- Complete-link
  - Optimizes a global criterion (look at the worst pair)
  - Creates more compact, “more” convex, spherical clusters

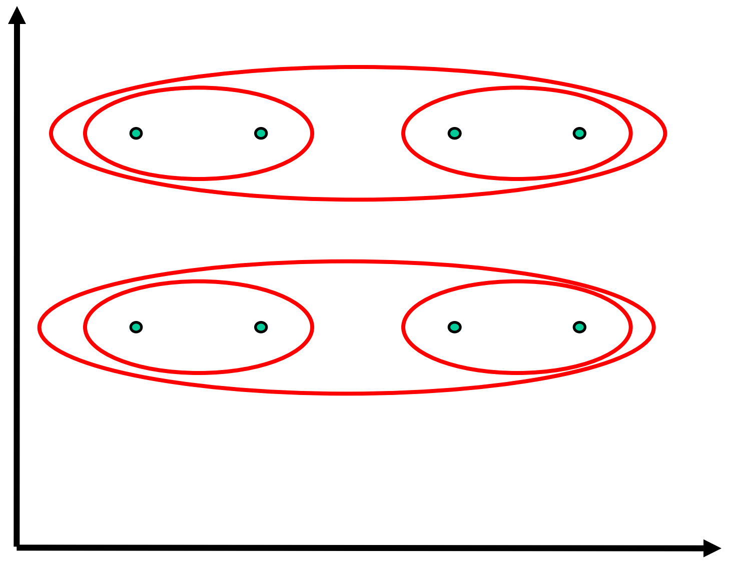


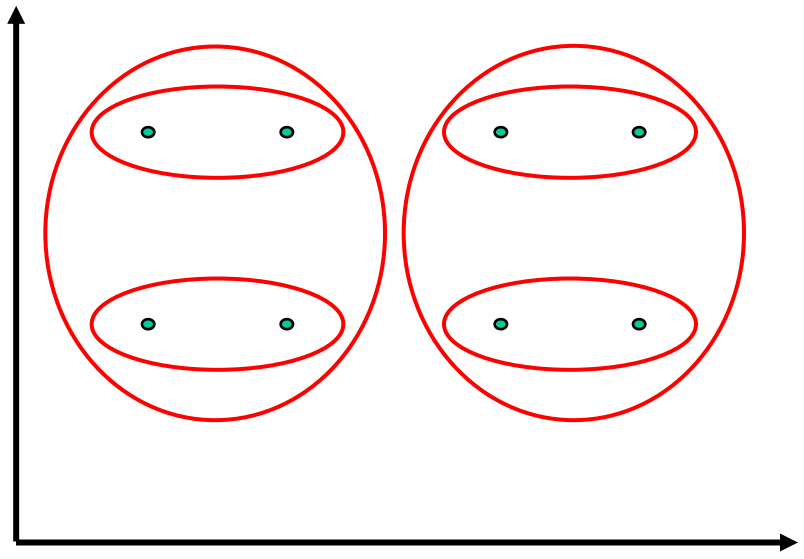
Single Linkage



Complete Linkage

Single-link versus Complete-link





Properties of Hierarchical Clustering

- Advantages
  - Simple and intuitive
  - Number of clusters is not an input of the method
  - Usually good quality clusters (which clusters?)
- Disadvantage
  - Does not really generate clusters
  - Very expensive; let n=|O|, m=|K|
    - Computing M requires $O(n^2)$ space and $O(mn^2)$ time
    - Naïve implementation requires $O(m*n^2*\log(n))$
    - Can be achieved in $O(m*n^2)$ (SLINK, CLINK)
  - Not applicable as such to large doc sets

Content of this Lecture

- Text clustering
- Cluster quality
- Clustering algorithms
  - Hierarchical clustering
  - K-means
  - Soft clustering: EM algorithm
- Application

Min-k-Cut Clustering

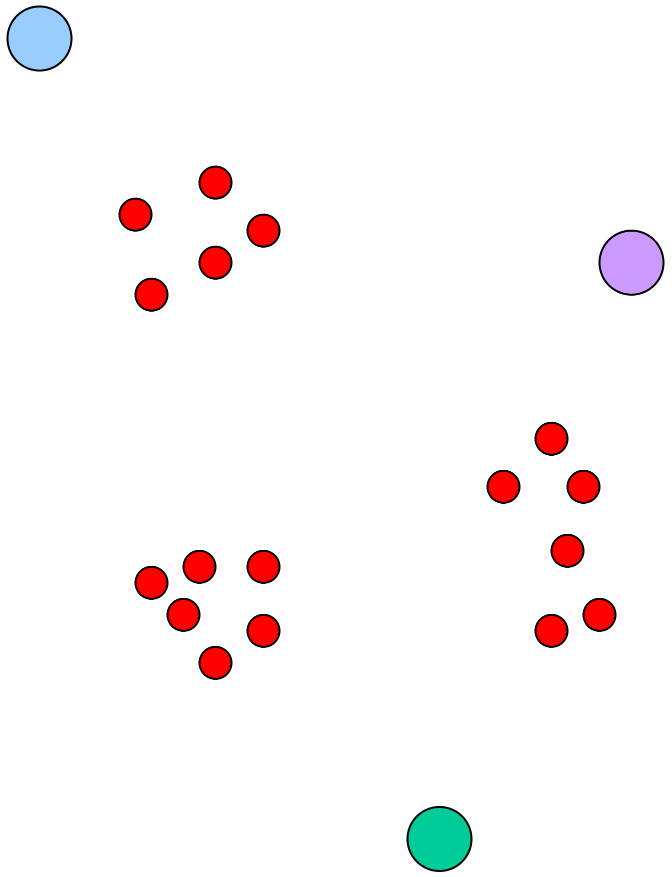
- Clustering in graph-theoretic concepts
- Definition
  Let G=(V,E) be a complete, weighted, undirected graph with V=O and $w(o_1,o_2)) = sim(o_1, o_2)$.
  - *A k-cut of G is a set S of edges such G‘=(V,E\S) has k connected components.*
  - *A min-k-cut of G is a k-cut of G such that w(S) is minimal*
- Notes
  - Every k-cut is a clustering of G into k clusters
  - Finding a min-k-cut is in O(|V|^k^2)
  - Not feasible in practice

Partitioning: K-Means

- Probably the most popular clustering algorithm
- Heuristic for solving the min-k-cut problem
- Requires the number k of clusters to be predefined
- Algorithm
  - Fix k
  - Guess k cluster centers
    - Can use k randomly chosen docs or k random points in feature-space
  - Loop forever
    - Assign all docs to their closest cluster center
    - If no doc has changed its assignment, stop
      - Or if sufficiently few docs have changed their assignment
    - Otherwise, compute new cluster centers

Example 1

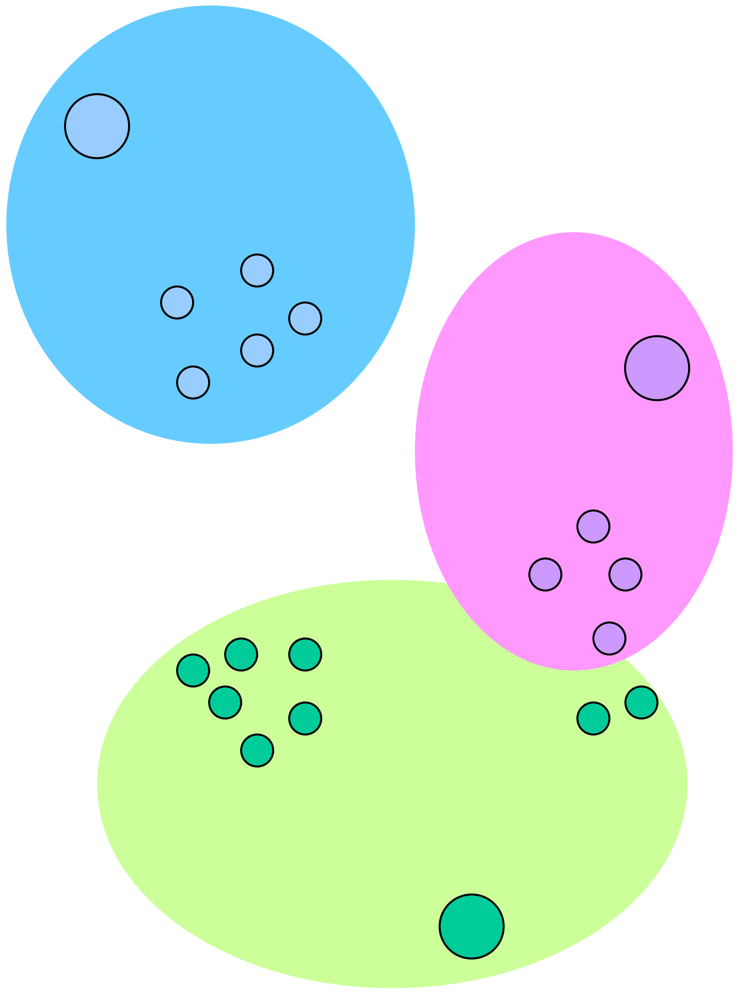
- k=3
- Choose random start points



Quelle: Stanford, CS 262 Computational Genomics

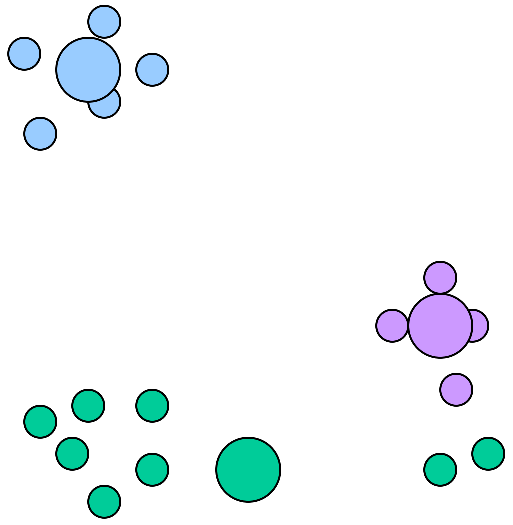
Example 2

- Assign docs to closest cluster centre

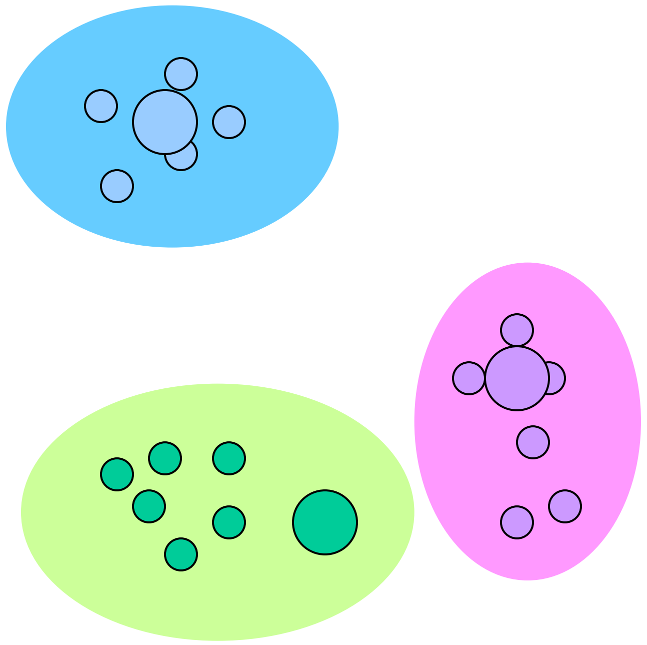


Example 3

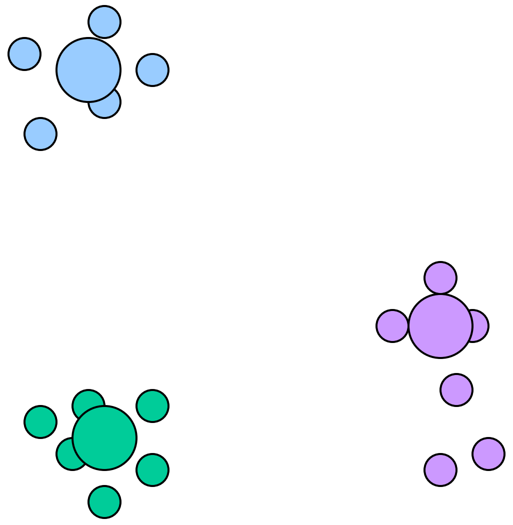
- Compute new cluster centre



Example 4

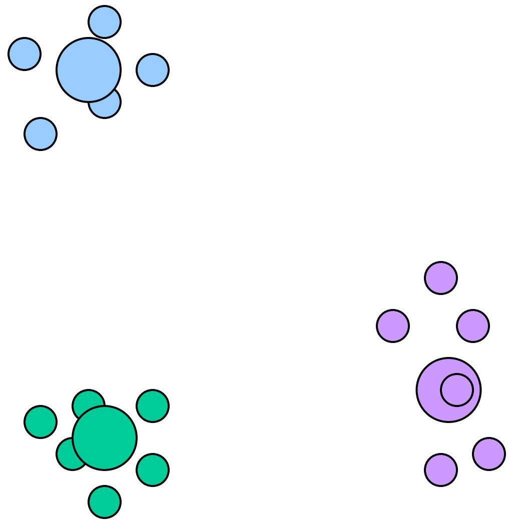


Example 5



Example 6

- Converged

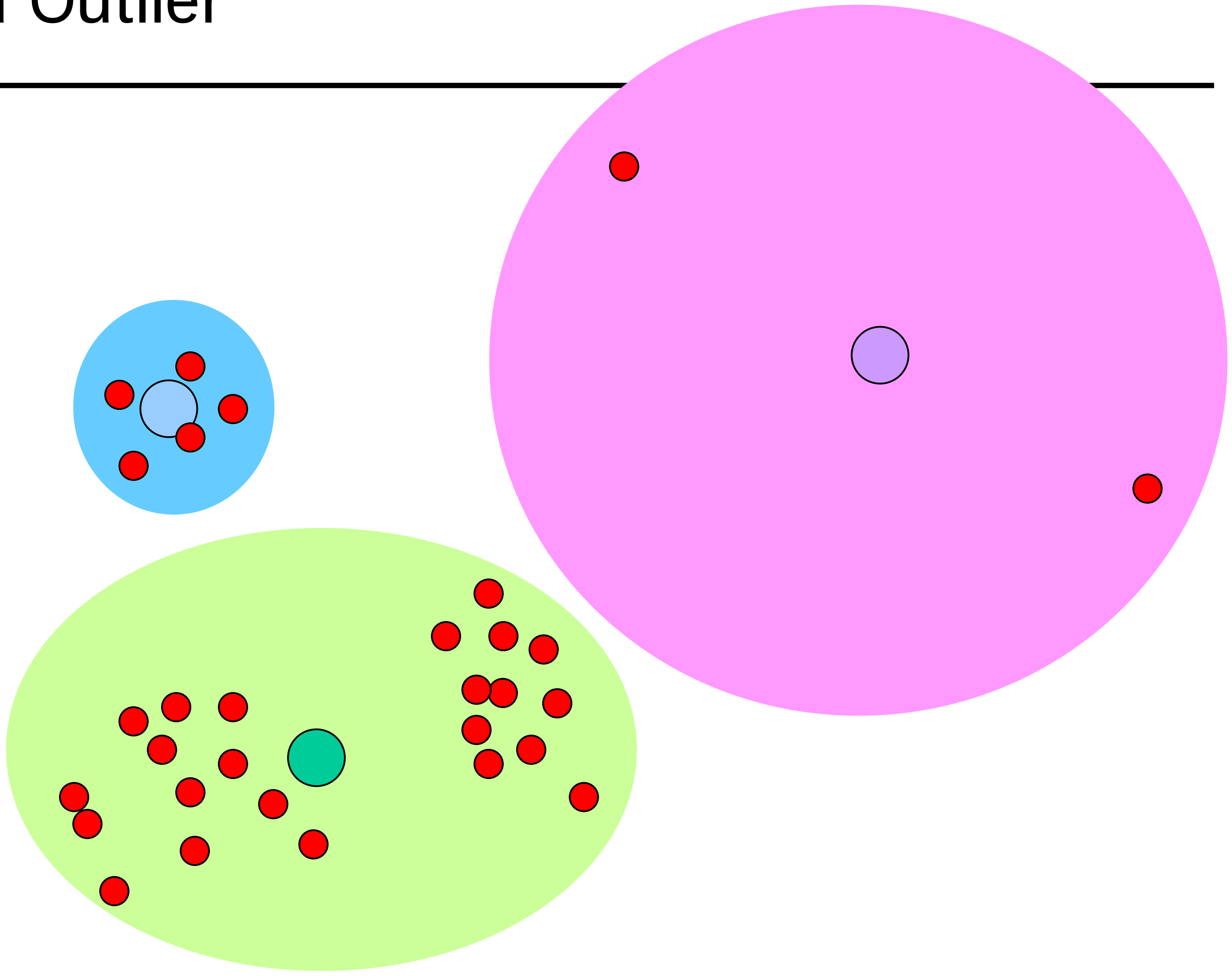


Properties

- Usually, k-Means converges quite fast
- Reasonable complexity: O(l*k*n*m)
  - Let l be the number of iterations
  - Assignment: n*k distance computations with O(m) each
  - New centers: Summing up n vectors of size m in k partitions
- Choosing the “right” start points is important
  - k-Means is a greedy heuristic and only finds local optima
  - Option 1: Start several times with different start points
  - Option 2: Compute hierarchical clustering on small random sample and choose cluster centers as start points (“Buckshot” algorithm)
- How to choose k?
  - Try for different k and compare quality score(s)

k-Means and Outlier

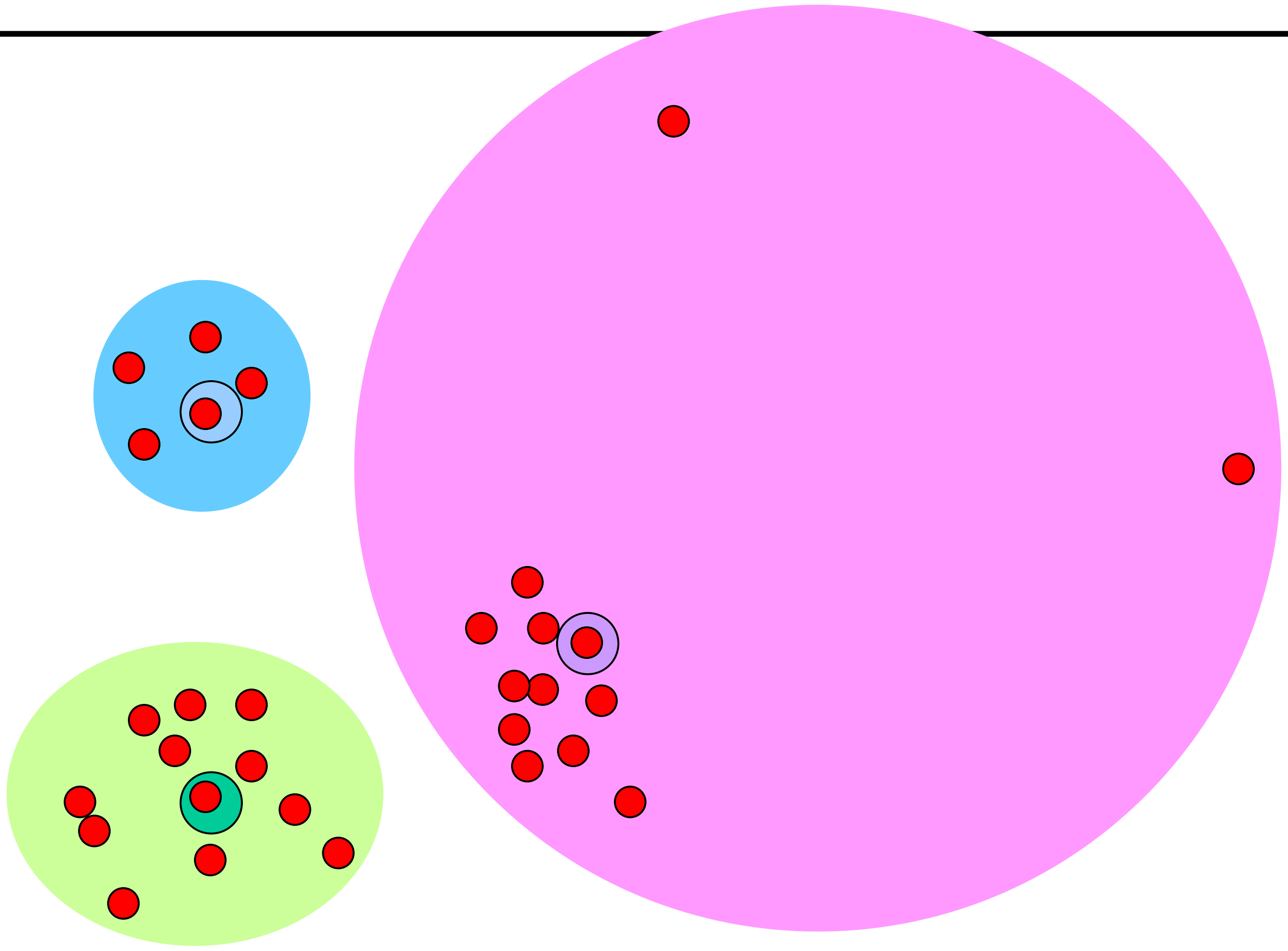
Assume k=3



Help: K-Medoid

- Chose the doc in the middle of a cluster as representative
  - Kaufman, Rousseeuw (1990): "Partitioning around medoids (pam)." in *Finding groups in data: an introduction to cluster analysis*
- Advantage
  - Less sensitive to outliers
  - Also works for non-metric spaces as no “new” center point needs to be computed
- Disadvantage: Increased complexity
  - Finding the median doc requires computing all pair-wise distances in each cluster in each round
  - Overall complexity is $O(n^3)$ in each step
    - We can save re-computations at the expense of more space

k-Medoid and Outlier



Content of this Lecture

- Text clustering
- Cluster quality
- Clustering algorithms
  - Hierarchical clustering
  - K-means
  - Soft clustering: EM algorithm
- Application

Soft Clustering

- We assumed docs are assigned to exactly one cluster
- Probabilistic interpretation: All docs pertain to all clusters with a certain probability
- Generative model
  - Assume we have k “doc-producing” devices
    - Such as authors, topics, ...
  - Each device produces docs that are normally distributed in feature space with device-specific mean and variance
  - Assume that k devices produced |D| documents
  - Clustering: Re-discovery of mean and variance of each device
- Solution: Expectation Maximization Algorithm (EM)

Expectation Maximization (rough sketch, no math)

- EM optimizes set of parameters P of a multivariate normal distribution (mean and variance, k clusters) given the data
- Iterative process with two phases
  - Guess an initial P
  - Expectation: Assign all docs its most likely generator based on P
  - Maximization: Compute new optimal P based on assignment
    - Using MLE or other estimation techniques
  - Iterate through both steps until convergence
- Finds a local optimum, convergence guaranteed
- K-Means: Special case of EM
  - Clusters with different means but equal variance
  - K-Means assumes all clusters have the same error model

Content of this Lecture

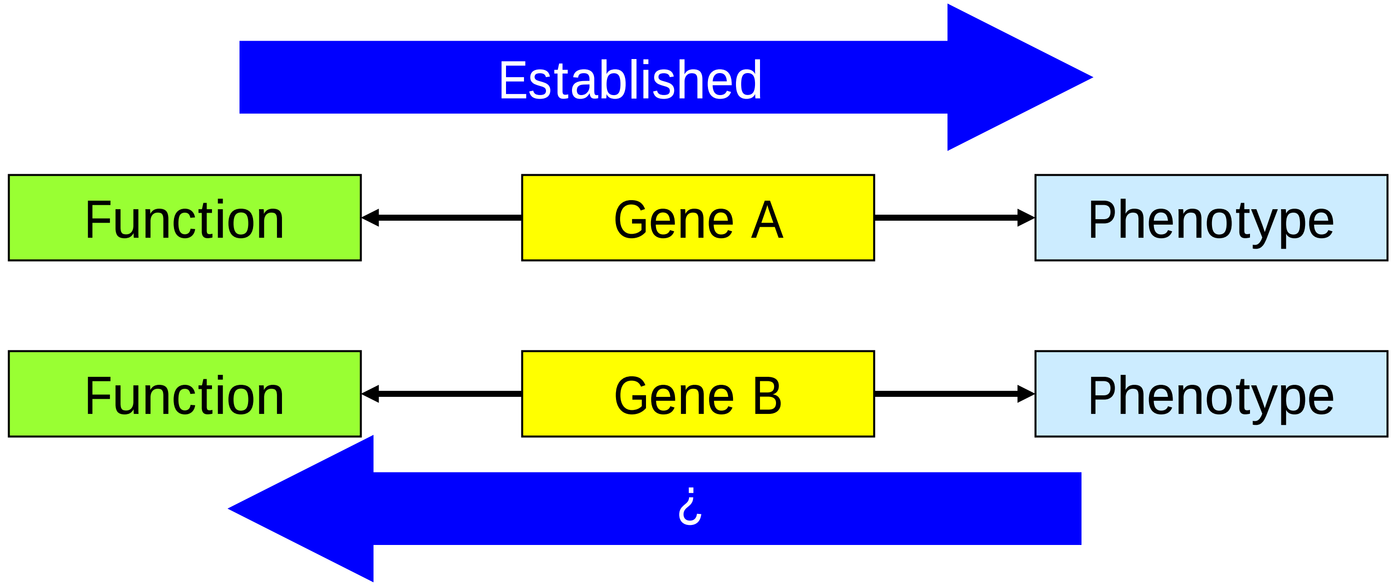
- Text clustering
- Cluster quality
- Clustering algorithms
- Application
  - Clustering Phenotypes

Mining Phenotypes for Function Prediction

Or ...

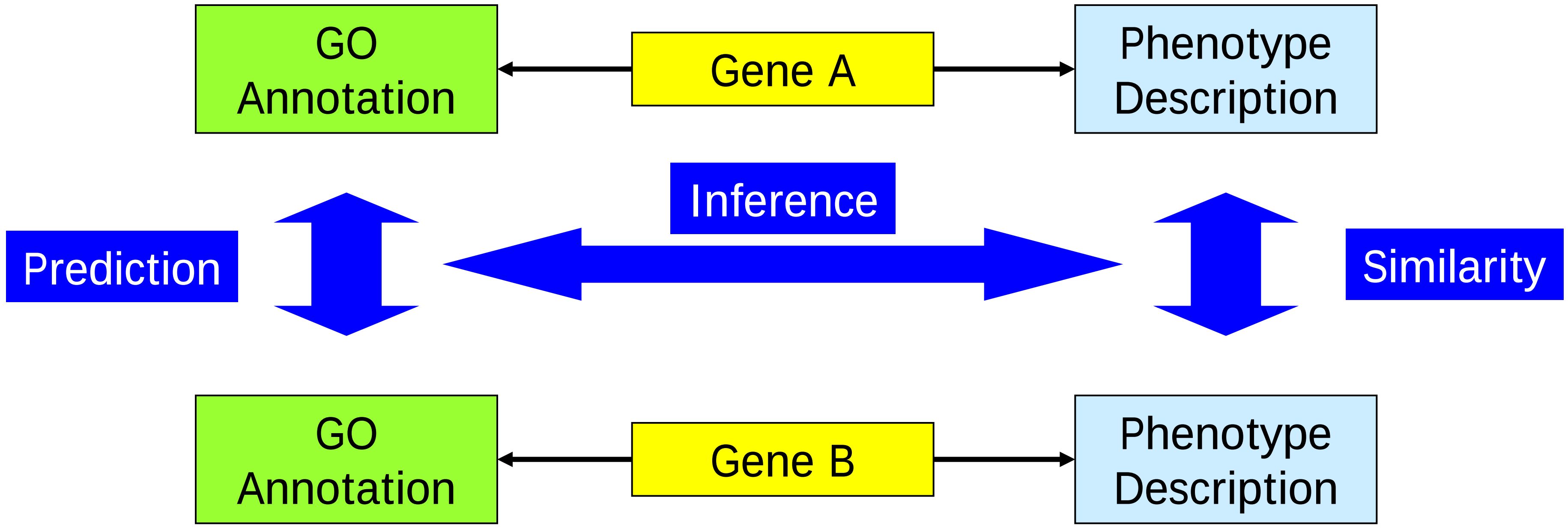
Source: http://www.guy-sports.com/humor/videos/powerpoint_presentation_dogs.htm

Mining Phenotypes: General Idea



- Known: Genes with sim. functions have sim. phenotypes
- Question: If genes generate very similar phenotypes – do they have the same functions?
  – Groth et al. (2008). "Mining phenotypes for gene function prediction." BMC Bioinformatics 9: 136.

Approach


GO Annotation
Gene A
Phenotype Description
Inference
Prediction
Similarity
GO Annotation
Gene B
Phenotype Description

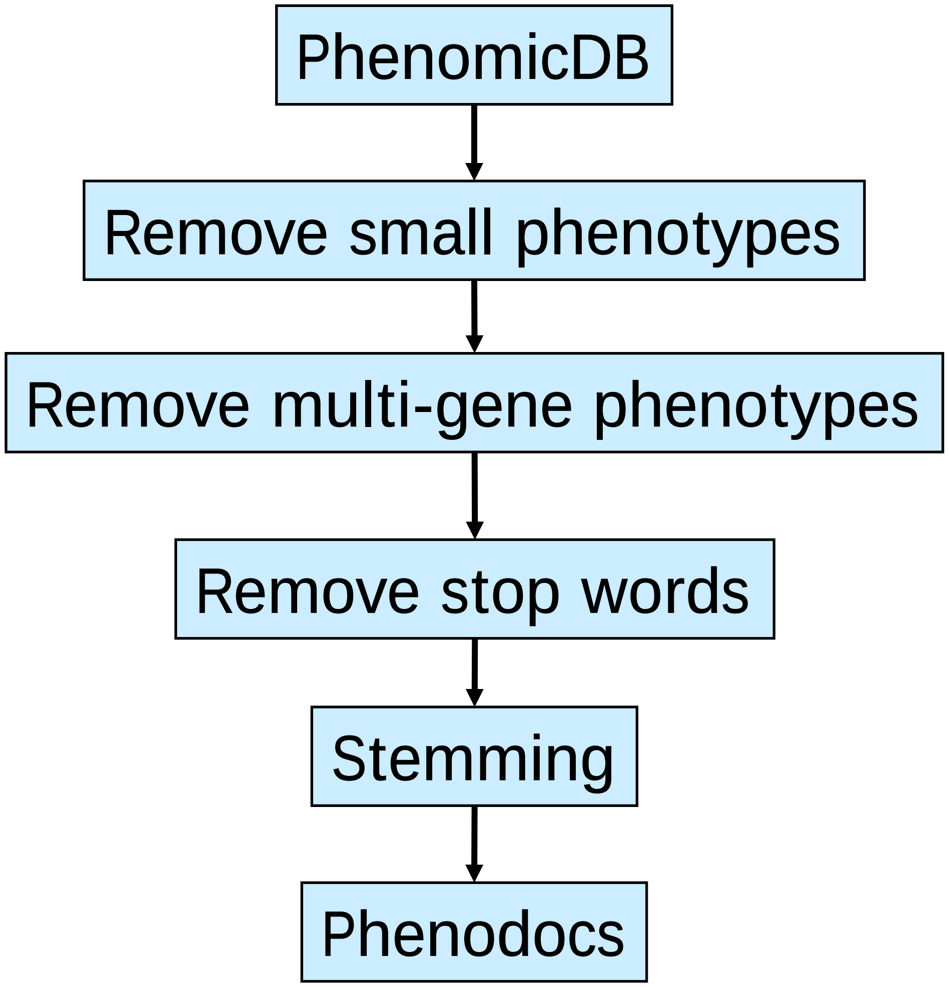
Phenodocs

411,102 phenotype texts

Short: <250 words

Remove all phenotypes associated to more than one gene (~500)

39,610 ‘phenodocs’ for 15,426 genes

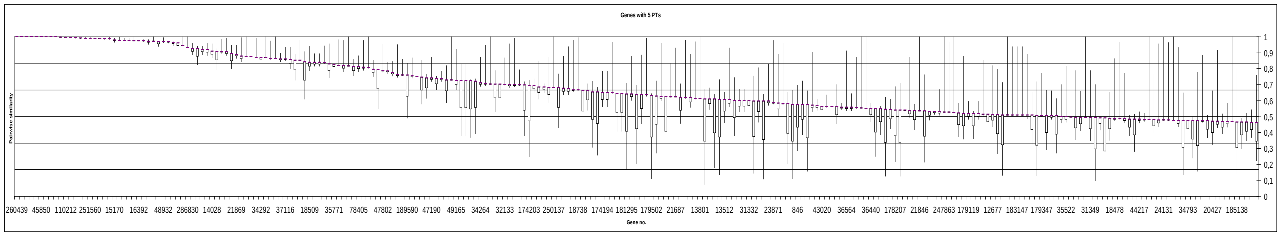


K-Means Clustering

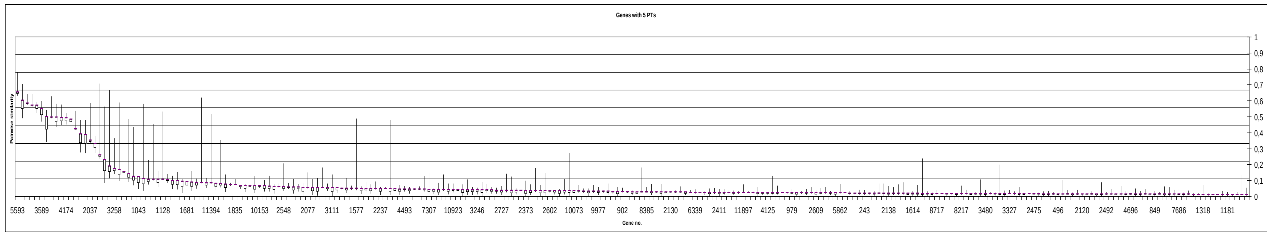
- Hierarchical clustering would require ~ 40.000*40.000 = 1.600.000.000 comparisons
- K-Means: Simple, iterative algorithm
- Number of clusters must be predefined
  - We experimented with 250 ... 3000 clusters

Properties: Phenodoc Similarity of Genes

Genes in phenoclusters



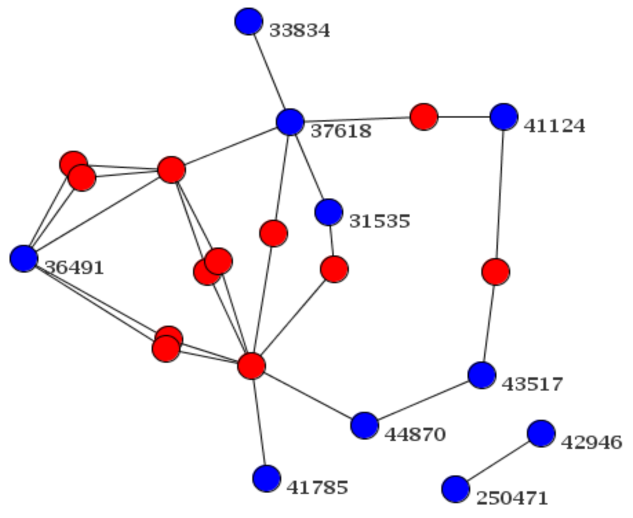
Control (Random selection)



- Pair-wise similarity scores of phenodocs of genes in the same cluster, sorted by score
- Result: Phenodocs of genes in phenoclusters are highly similar to each other

PPI: Inter-Connectedness

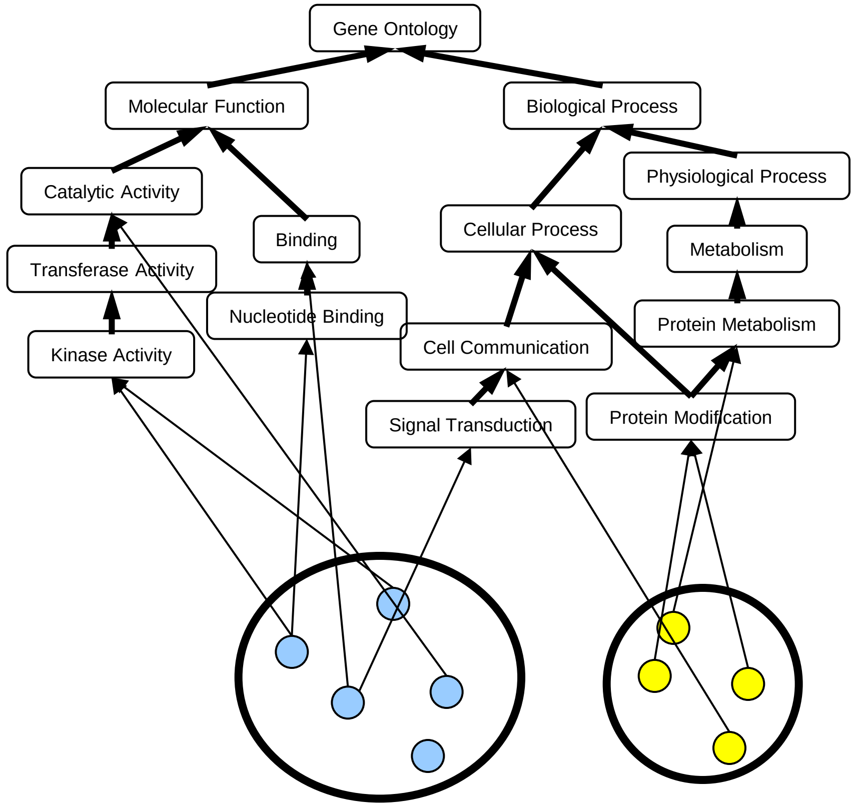
- Interacting proteins often share function
- PPI from BIOGRID database
  - Not at all a complete dataset
- In >200 clusters, >30% of genes interact with each other
- Control (random groups): 3 clusters
- Result: Genes in phenoclusters interact with each other much more often than expected by chance



Proteins and interactions from BioGrid. Red proteins have no phenotypes in PhenomicDB

Coherence of Functional Annotation

- Comparison of GO annotation of genes in phenoclusters
  - Data from Entrez Gene
  - Similarity of two GO terms: Normalized number of shared ancestors
  - Similarity of two genes: Average of the top-k GO pairs
- >200 clusters with score >0.4
  - Control: 2 clusters
- Results: Genes in phenoclusters have a much higher coherence in functional annotation than expected by chance

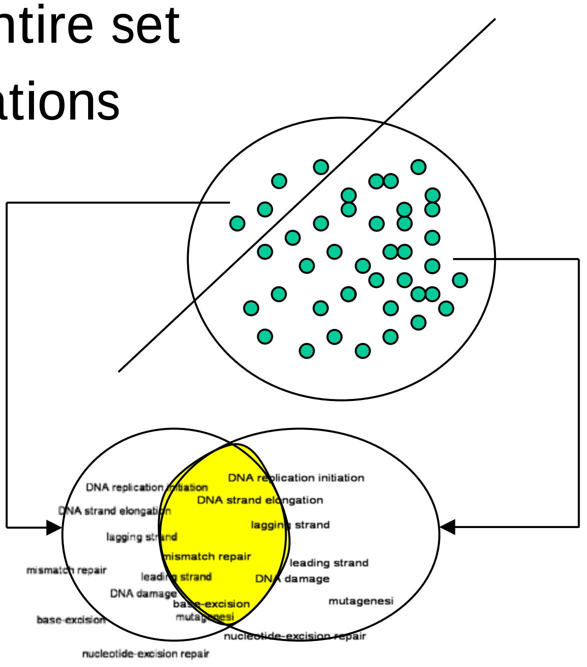


Function Prediction

- Can increased functional coherence of clusters be exploited for function prediction?
- Approach
  - Compute phenoclusters
  - For each cluster, compute set of associated genes (gene cluster)
  - In each gene cluster, predict frequent GO terms to all genes
    - Frequent: annotated to >50% of genes in the cluster
- Some filtering of clusters required / useful
  - Filter 1: Only clusters with >2 members and at least one common GO term
  - Filter 2: Only clusters with GO coherence>0.4
  - Filter 3: Only clusters with PPI-connectedness >33%
  - ...

Evaluation

- How can we know how good we are?
- Cross-validation
  - Separate genes in training (90%) and test (10%)
  - Remove annotation from genes in test set
  - Build clusters and predict functions on entire set
  - Compare predicted with removed annotations
    - Precision and recall
  - Repeat and average results
    - Macro-average
- Note: This punishes new and potentially valid annotations



Results for Different Filters

	(Filter 1)	(Filter 1 & Filter 2)	(Filter 1 & Filter 3)
# of clusters	196	74	53
# of terms	345	159	102
# of genes	3213	711	409
Precision	67.91%	62.52%	60.52%
Recall	22.98%	26.16%	19.78%

- What if we consider predicted terms to be correct that are a little more general than the removed terms (filter 1)?
  - One step more general: 75.6% precision, 28.7% recall
  - Two steps: 76.3% precision, 30.7% recall
- The less stringent "GO equality", the better the results
  - This is a common "trick" in studies using GO

Results for Different Cluster Sizes

K	250	500	750	1,000		2,750	3,000
Cluster w/ GO-Sim ≥ 1	14 (5.6%)	26 (5.2%)	44 (5.9%)	71 (7.1%)		273 (9.9%)	309 (10.3%)
# Genes	561	781	943	1155		2094	2221
Cluster w/ PPi ≥ 75%	12 (4.8%)	34 (6.8%)	65 (8.7%)	88 (8.8%)		314 (11.4%)	353 (11.8%)
# Genes	785	988	1166	1263		1810	1914
Cluster w/ PPi ≥ 33%	49 (19.6%)	119 (23.8%)	193 (25.7%)	252 (25.2%)		662 (24.1%)	717 (23.9%)
# Genes	3362	4044	4296	4417		4811	4833
Cluster for GO-Pred.	73 (29.2%)	153 (30.6%)	230 (30.7%)	295 (29.5%)	…	748 (27.2%)	816 (27.2%)
# Genes	3465	4139	4344	4438		5016	5115
# Terms	123	247	383	489		1436	1557
Precision	81.53%	77.16%	74.26%	71.73%		63.92%	62.89%
Recall	16.90%	20.22%	24.45%	26.36%		34.64%	34.61%
Avg. Genes/Cluster	52	26	17	13		4	4

- With increasing k
  - Clusters are smaller
  - Number of predicted terms increases
    - Clusters are more homogeneous
  - Number of genes which receive annotations increases
  - Precision decreases slowly, recall increases
    - Effect of the rapid increase in number of predictions