Semi-supervised learning in natural language processing

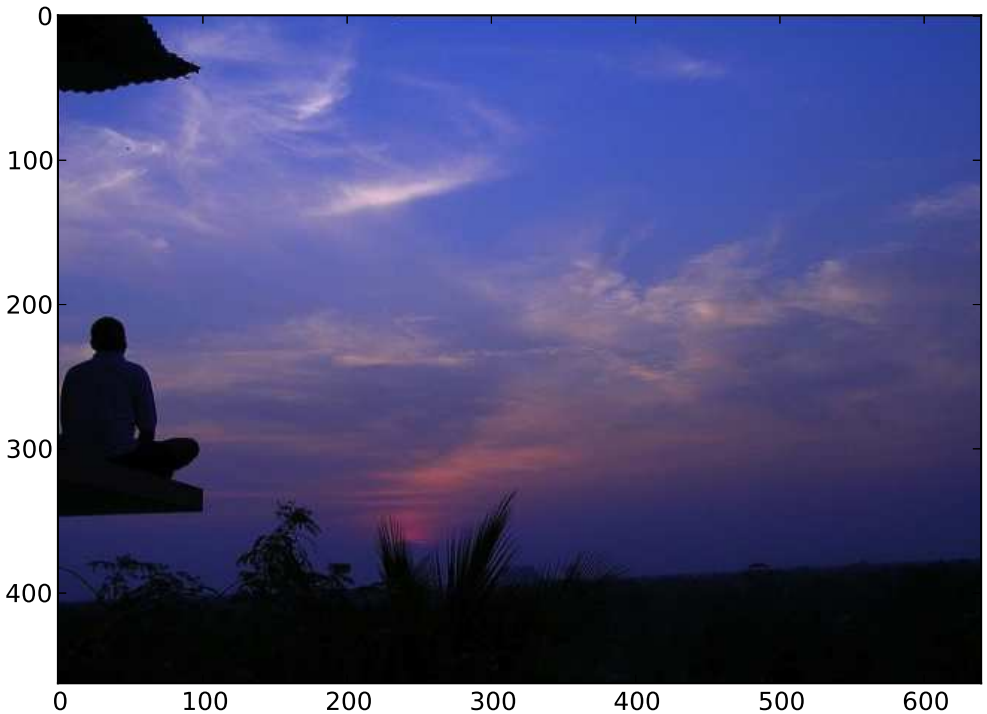
Anders Søgaard

Unsupervised learning

Semi-supervised learning

Course outline

1. Supervised learning
2. **Unsupervised learning and semi-supervised learning**
3. Learning from weighted data and transfer learning
4. Applications to dependency parsing
5. Transfer learning in the blind



From `meditationworkshop.org`: "...breathe in as slowly and quietly as possible, so that if somebody placed a tiny thread in front of your nose it would not move at all. And when you breathe out, try to breathe out even more slowly than you breathed in. If possible, leave a short pause between the end of your exhalation and the beginning of your inhalation."

Exercise

y	x_1	x_2
1	0.1	0.8
1	0.2	0.9
0	0.5	0.2
0	0.55	0.4

- What is the nearest neighbor prediction on $\langle .5, .5 \rangle$?
- Find a linear classifier $\mathbf{w}$ that separates positives from negatives.

Exercise

y	x_1	x_2
1	0.1	0.8
1	0.2	0.9
0	0.5	0.2
0	0.55	0.4

- What is the nearest neighbor prediction on $\langle .5, .5 \rangle$?
- 0
- Find a linear classifier $\mathbf{w}$ that separates positives from negatives.

Exercise

y	x_1	x_2
1	0.1	0.8
1	0.2	0.9
0	0.5	0.2
0	0.55	0.4

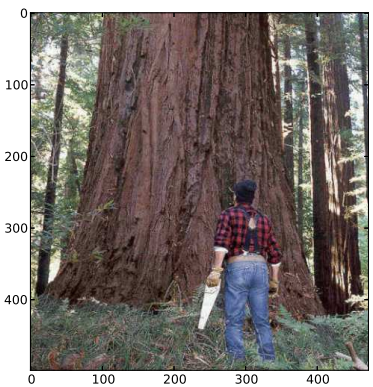
- What is the nearest neighbor prediction on $\langle .5, .5 \rangle$?
- 0
- Find a linear classifier $\mathbf{w}$ that separates positives from negatives.
- e.g. $\langle -1, 0, -.4 \rangle$

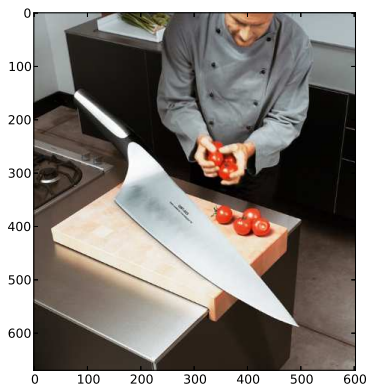
Main lessons from yesterday

1. In NLP, labeled samples are always too small.

One day there was a fire in a wastebasket in the Dean's office and in rushed a physicist, a chemist, and a statistician. The physicist immediately started to work on how much energy would have to be removed from the fire to stop the combustion. The chemist worked on which reagent would have to be added to the fire to prevent oxidation. While they did this, the statistician was setting fire to all the other wastebaskets in the office. "What are you doing?" the Dean asked.
Exercise: What did the statistician answer?

2. In NLP, labeled samples are always biased (measured by e.g. *KL*-divergence).





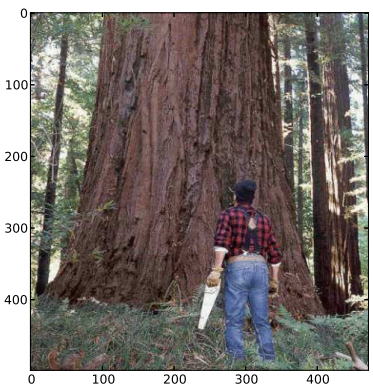
Main lessons from yesterday

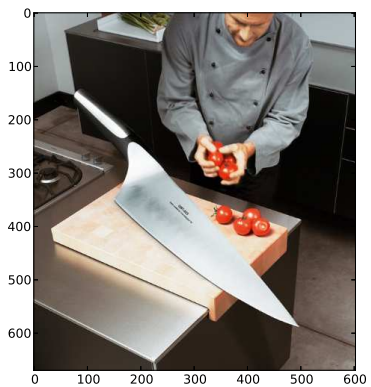
1. In NLP, labeled samples are always too small.

One day there was a fire in a wastebasket in the Dean's office and in rushed a physicist, a chemist, and a statistician. The physicist immediately started to work on how much energy would have to be removed from the fire to stop the combustion. The chemist worked on which reagent would have to be added to the fire to prevent oxidation. While they did this, the statistician was setting fire to all the other wastebaskets in the office. "What are you doing?" the Dean asked.
Exercise: What did the statistician answer?

- "To solve the problem, obviously you need a larger sample size."
- (From Hugh Foley.)

2. In NLP, labeled samples are always biased (measured by e.g. *KL*-divergence).





How can we learn from unlabeled data?

Unsupervised learning

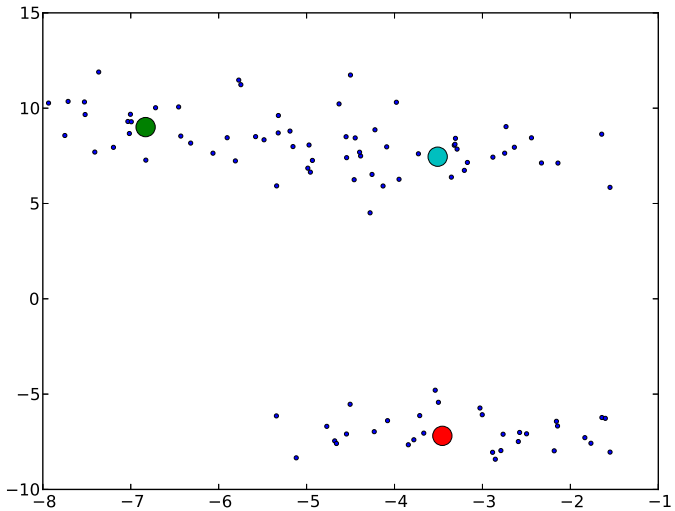


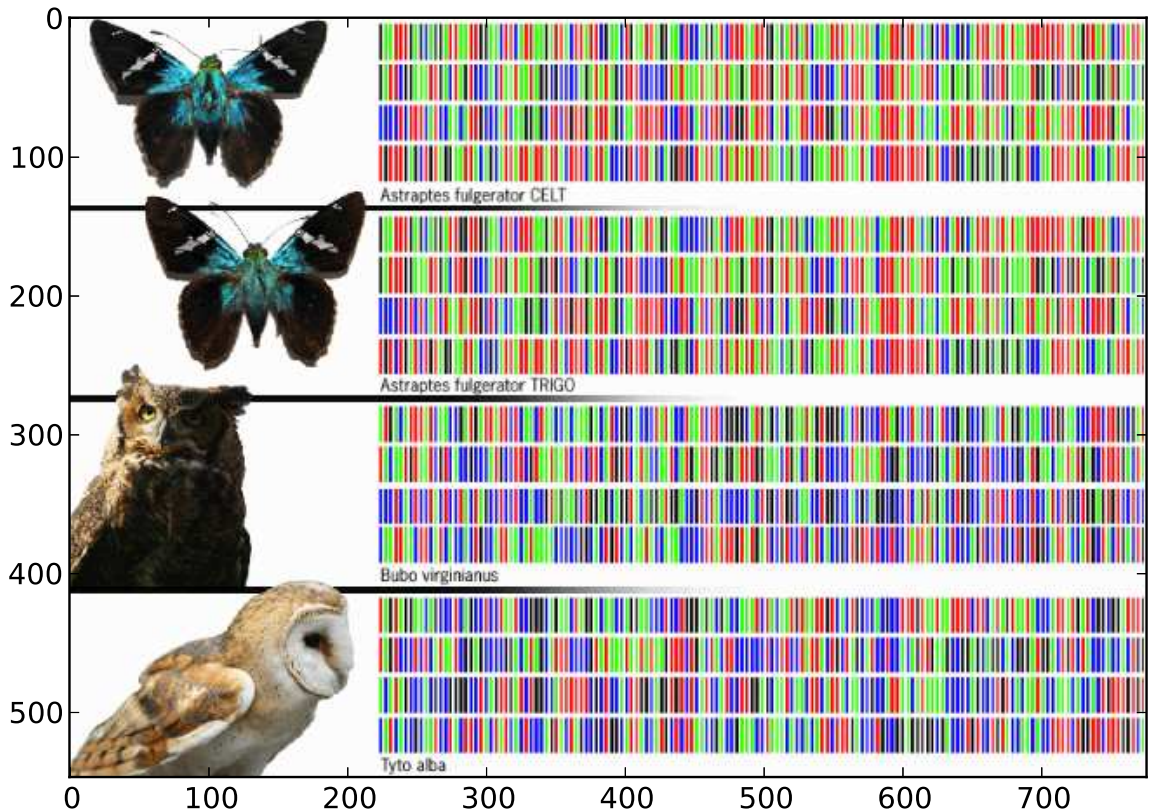
Figure: Randomly generated clusters (three isotopic Gaussian blobs)

Clusters are used for summarization, compression and finding nearest neighbors, but:

- How many clusters?
  - What is the appropriate level of granularity?
- Do the clusters correspond to concepts?

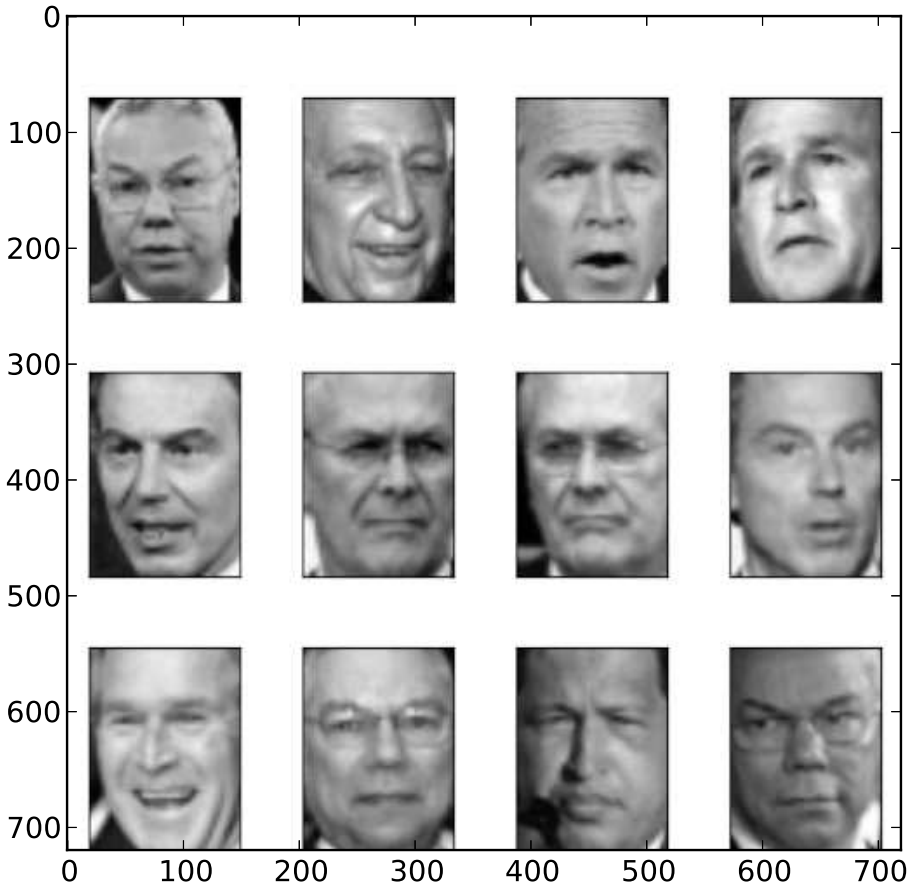
Exercise

Exercise: How many clusters (species) do you see here?



Exercise

Exercise: How many clusters do you see here?

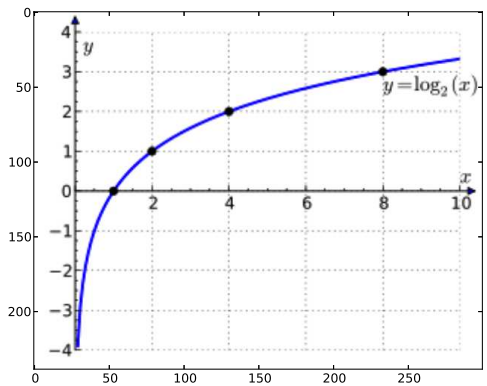


Evaluation of clustering methods

- Homogeneity: All members of a cluster should be in the same class.

$$1 - \frac{-\sum_{k \in K} \sum_{c \in C} \frac{|c \cup k|}{N} \log \frac{|c \cup k|}{|k|}}{-\sum_{c \in C} \frac{|c|}{N} \log \frac{|c|}{N}}$$

- Completeness: All members of a class should be in the same cluster.
- V-measure: Harmonic mean of Homogeneity and Completeness.



- Alternatives: F-measure, down-stream evaluation, etc.

Hierarchical agglomerative clustering

1: dataset $X = \{\mathbf{x}_i\}_{i=1}^N$, a distance function d
2: $C = \{\{\mathbf{x}_i\} \mid \mathbf{x}_i \in X\}$ # each datapoint a cluster
3: **while** $\{\mathbf{x}_i\}_{i=1}^N \notin C$ **do**
4: $\quad c_j, c_k \leftarrow \arg\min_{c_{j'}, c_{k'} \in C, j' \neq k'} d(c_{j'}, c_{k'})$
5: $\quad C \leftarrow C \cup \{(c_j \cup c_k)\}$
6: **end while**
7: **return** C

Figure: Agglomerative clustering

with

single linkage	$d(c_j, c_k) = \min_{\mathbf{x} \in c_j, \mathbf{x}' \in c_k} d(\mathbf{x}, \mathbf{x}')$
all linkage	$d(c_j, c_k) = \max_{\mathbf{x} \in c_j, \mathbf{x}' \in c_k} d(\mathbf{x}, \mathbf{x}')$
average linkage	$d(c_j, c_k) = \frac{\sum_{\mathbf{x} \in c_j, \mathbf{x}' \in c_k} d(\mathbf{x}, \mathbf{x}')}{\lvert C \rvert}$

Note: Agglomerative clustering does not provide a decision boundary.

Ward's method

Ward's method is very similar to average-link agglomerative clustering. It formalizes the notion of merging cost:

$$\Delta(c_j, c_k) = \sum_{i \in c_j \cup c_k} d(\mathbf{x}_i - m_{c_j \cup c_k})^2 - \sum_{i \in c_j} d(\mathbf{x}_i - m_{c_j})^2 - \sum_{i \in c_k} d(\mathbf{x}_i - m_{c_k})^2 \quad (1)$$

It then replaces $d(c_{j'}, c_{k'})$ in line 6 in agglomerative clustering with the left-hand side of Equation 1. The algorithm is greedy and constrained by previous choices, and the sum-of-squares for a given number k of clusters is usually larger than the minimum, and even larger than what k-means (below) will achieve. Ward's algorithm can be used to initialize k-means, however, leading k-means into better local maxima.
In `scikits`:

- from sklearn.cluster import Ward
- clu=Ward(n_clusters=2)
- clu.fit(X_train)
- print sklearn.metrics.v_measure_score(y_train,clu.labels_)

k-means and EM

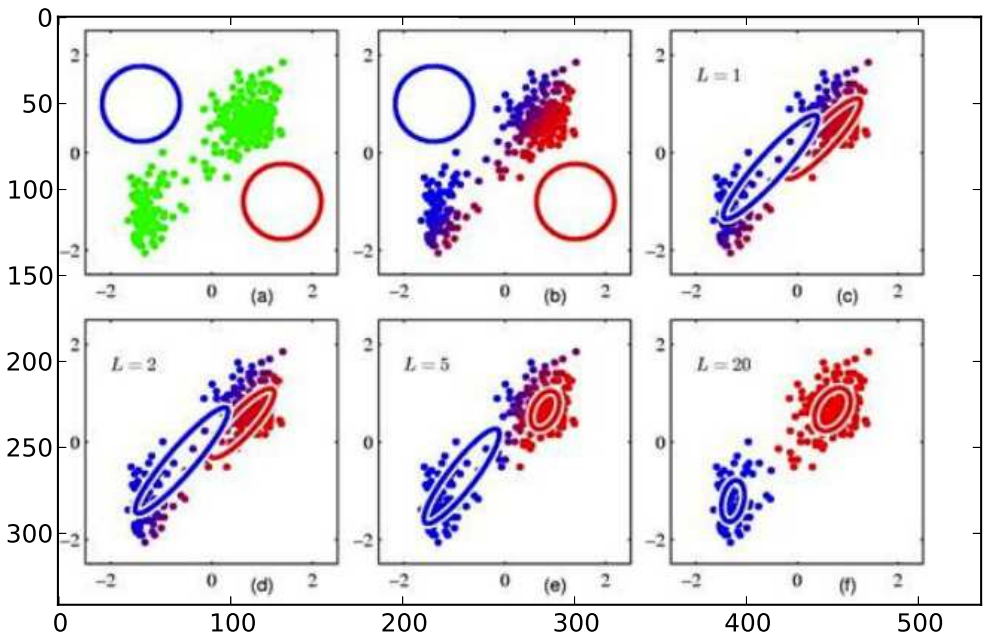
1: $\{m_1, \ldots, m_k\} \leftarrow$ **random.choice**$(\{\mathbf{x}\}_{i=1}^{N}, k)$
2: $C_j = \{m_j\}$ for $j \leq k$
3: **for** $p \in P$ **do**
4: **for** $n \in N$ **do**
5: $C_i \leftarrow C_i \cup \{\mathbf{x}_n\}$ s.t. $i = \arg\min_{j \leq k} d(m_j, \mathbf{x}_n)$
6: **end for**
7: $\{m_j\} \leftarrow$ **centroid**(C_j) for $j \leq k$
8: **end for**

Figure: k-means with P passes

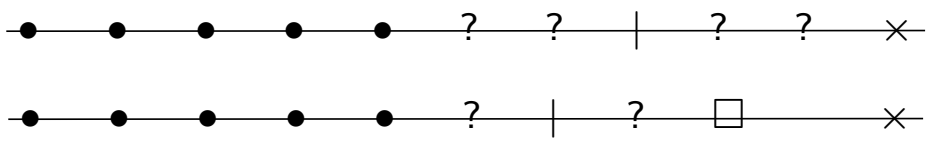
In `scikits`:

- from sklearn.cluster import KMeans
- clu=KMeans(k=2)
- clu.fit(X_train)
- sklearn.metrics.v_measure_score(y_train,clu.labels_)

EM (Gaussian mixture model)



Semi-supervised learning



Semi-supervised learning

T	U	$T + U$
NN	Hier.Cl.	Graph-based†
NB	EM	EM
Perceptron	k-means	aux.obj. (e.g. S3VM)

†: Special interest in condensation.

- Cluster-then-label
- Clusters-as-features
- **Wrapper methods**

Semi-supervised learning

T	U	$T + U$
NN	Hier.Cl.	Graph-based†
NB	EM	EM
Perceptron	k-means	aux.obj. (e.g. S3VM)

†: Special interest in condensation.

- Cluster-then-label
- Clusters-as-features
- **Wrapper methods**
- **Semi-supervised condensation**

Self-training

1: $L = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^{N}, U = \{\mathbf{x}_i\}_{i=1}^{M}$
2: $c \leftarrow \texttt{train}(L)$
3: **while** stopping criterion is not met **do**
4: $\quad L \leftarrow L + \texttt{select}(\texttt{label}(U, c))$
5: $\quad c \leftarrow \texttt{train}(L)$
6: **end while**
7: **return** c

Figure: Self-training

Exercise: What are the hyper-parameters?
Exercise: What is the caveat of confidence-based selection, even with perfect confidence estimation?

Self-training (delible)

1: $L = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^{N}, U = \{\mathbf{x}_i\}_{i=1}^{M}$
2: $c \leftarrow \texttt{train}(L)$
3: **while** stopping criterion is not met **do**
4: $\quad L_0 \leftarrow L + \texttt{select}(\texttt{label}(U, c))$
5: $\quad c \leftarrow \texttt{train}(L_0)$
6: **end while**
7: **return** c

Figure: Self-training

Self-training (delible) in scikits

Parameters: 20 iterations, no confidence-based selection, no pooling, no throttling, no balancing.

```
weights=[1]*(X_labeled.shape[0])+[0]*U_size*2
y_train=list(y_labeled)+[1]*U_size+[0]*U_size
y_train=np.array(y_train)
scores=[clf.score(X_test,y_test)]
for _ in range(20): do
    for i in range(X_labeled.shape[0],X_train.shape[0]-U_size):  do
        if float(clf.predict(X_train[i,:]))==1:  then
            weights[i]=1
        else
            weights[i+U_size]=1
        end if
    end for
    clf.fit(X_train,y_train,sample_weight=weights)
    scores.append(clf.score(X_test,y_test))
end for
```

EM in scikits

```
weights=[1]*(X_labeled.shape[0])+[0]*U_size*2
y_train=list(y_labeled)+[1]*U_size+[0]*U_size
y_train=np.array(y_train)
scores=[clf.score(X_test,y_test)]
for _ in range(20): do
    for i in range(X_labeled.shape[0],X_train.shape[0]-U_size):  do
        if float(clf.predict(X_train[i,:]))==1:  then
            weights[i]=myMax(clf.predict_proba(X_train[i,:]))
            weights[i+U_size]=myMin(clf.predict_proba(X_train[i,:]))
        else
            weights[i]=myMin(clf.predict_proba(X_train[i,:]))
            weights[i+U_size]=myMax(clf.predict_proba(X_train[i,:]))
        end if
    end for
    clf.fit(X_train,y_train,sample_weight=weights)
    scores.append(clf.score(X_test,y_test))
end for
```

EM (clf=nb())

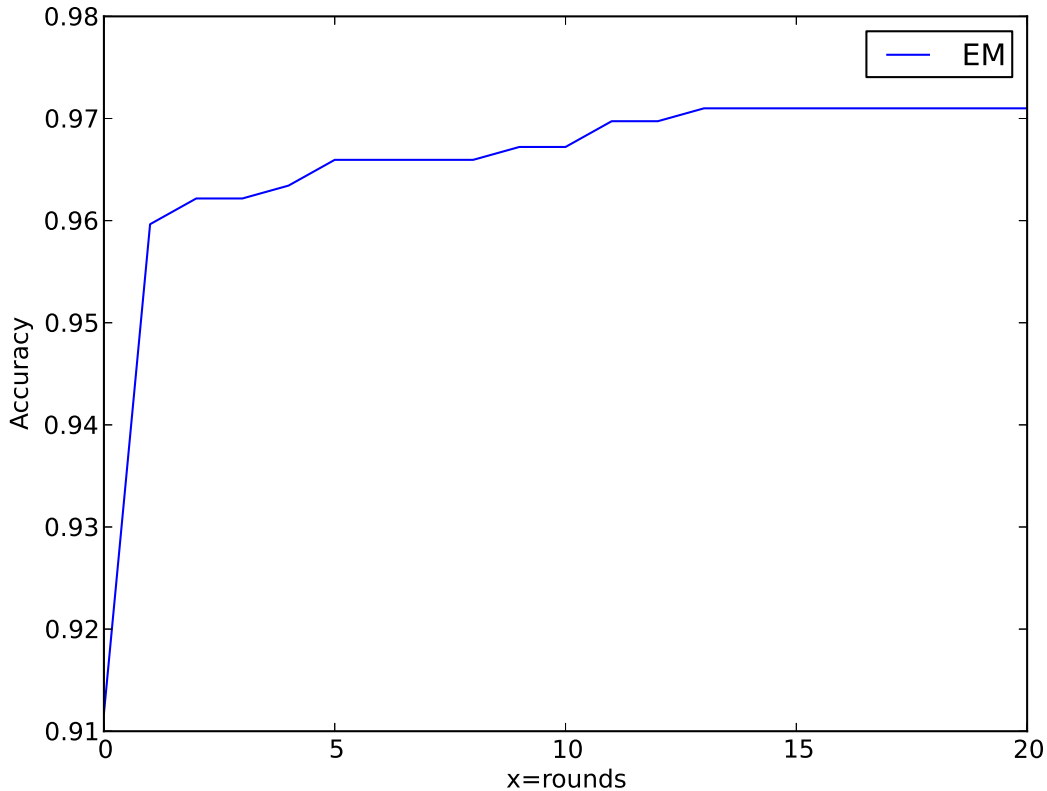


Figure: Delible soft self-training (EM) on Hockey-Motorcycles to Baseball-Autos

Co-training

1: $L = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^{N}, U = \{\mathbf{x}_i\}_{i=1}^{M}$
2: $c \leftarrow \texttt{train}(L)$
3: **while** stopping criterion is not met **do**
4: $\quad c_1 \leftarrow \texttt{train}(\texttt{view}_1(L))$
5: $\quad c_2 \leftarrow \texttt{train}(\texttt{view}_2(L))$
6: $\quad L \leftarrow L + \texttt{select}(\texttt{label}(U, c_1)) + \texttt{select}(\texttt{label}(U, c_2))$
7: **end while**
8: $c \leftarrow \texttt{train}(L)$
9: **return** c

Figure: Co-training

Co-training

$L = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^{N}, U = \{\mathbf{x}_i\}_{i=1}^{M}$
$c \leftarrow \texttt{train}(L)$
$L_1 \leftarrow L$, $L_2 \leftarrow L$
while stopping criterion is not met **do**
  $c_1 \leftarrow \texttt{train}(\texttt{view}_1(L))$
  $c_2 \leftarrow \texttt{train}(\texttt{view}_2(L))$
  $L_2 \leftarrow L_2 + \texttt{select}(\texttt{label}(U, c_1))$
  $L_1 \leftarrow L_1 + \texttt{select}(\texttt{label}(U, c_2))$
end while
$c \leftarrow \texttt{train}(L)$
return c

Figure: Co-training

Note: EM and co-training are easily combined (Co-EM; Nigam and Ghani, 2000).

Comparison (clf=nb())

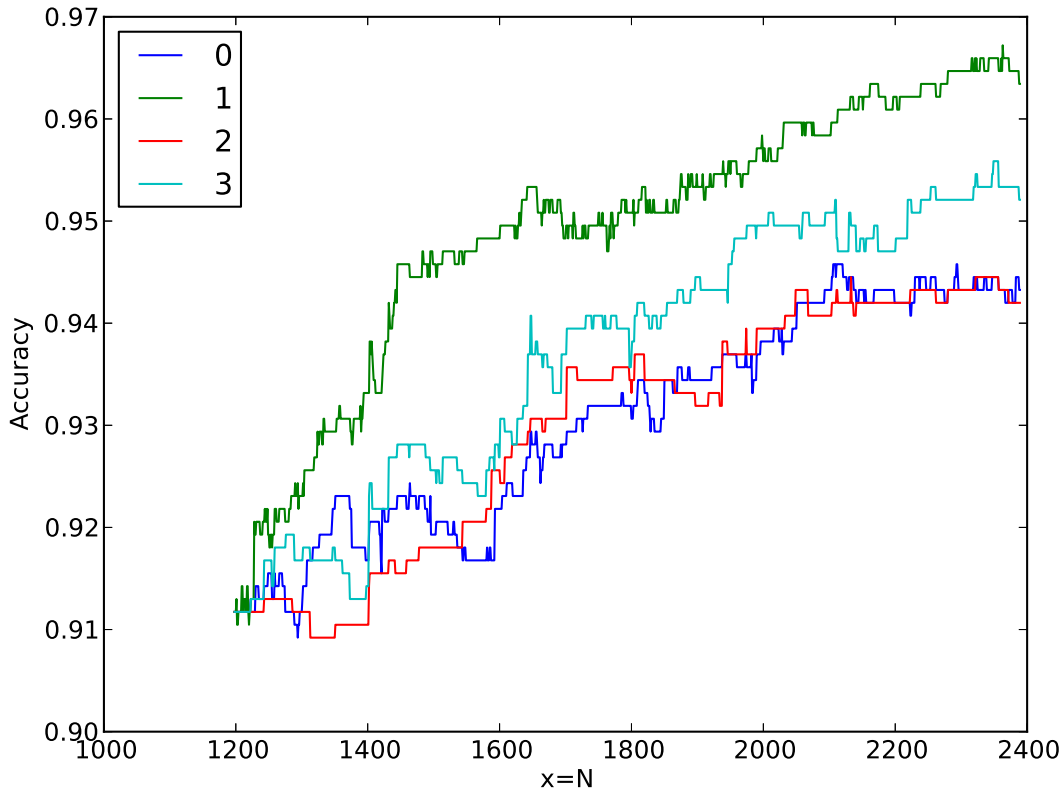


Figure: Comparison of (0) self-training, (1) soft self-training, (2) random co-training and (3) regular co-training on HOCKEY-MOTORCYCLES to BASEBALL-AUTOS (pool size=1)

Tri-training

for $i \in \{1..3\}$ **do**
  $c_i \leftarrow train\_classifier(l_i, L)$
end for
repeat
  for $i \in \{1..3\}$ **do**
    for $x \in U$ **do**
      $L_i \leftarrow \emptyset$
      if $c_j(x) = c_k(x)(j, k \neq i)$ **then**
        $L_i \leftarrow L_i \cup \{(x, c_j(x)\}$
      end if
    end for
    $c_i \leftarrow train\_classifier(L \cup L_i)$
  end for
until stopping criterion is met
apply c_i

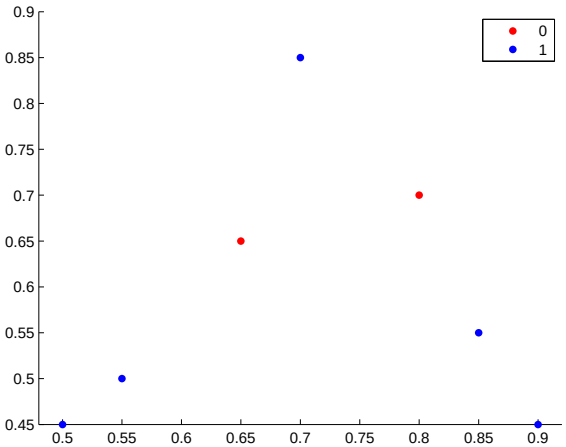
Figure: Generalized tri-training

Li and Zhou (2005) use bootstrap sampling to create a diverse sample, but . . .
Exercise: What other options are there?

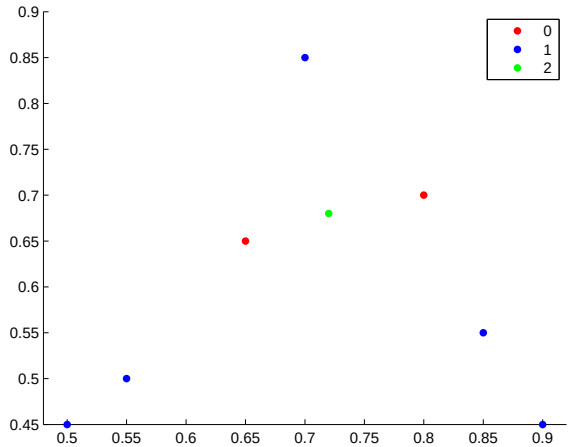
Semi-supervised dataset condensation

- Nearest neighbor methods are linear in the size of T.
- Typically, only a small subset of data points are important; cf. support vectors in SVMs.
- While T may be small, $T + U$ is usually *as big as we can handle*.
- Unlabeled data can improve condensation (Duang et al., 2008; Søgaard, 2011).
- Semi-supervised condensation can improve performance (op.cit.).

Intuition



a)



b)

Figure: Unlabeled data may help find better representatives in condensed training sets.

Semi-supervised nearest neighbor editing

$\{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^N$, $T_e = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^N$
for n to N **do**
  if $c(\mathbf{x}_n) = y_n$ **then**
    $T_e = T_e \setminus \{\langle y_n, \mathbf{x}_n \rangle\}$
  end if
end for
return T_e

Figure: Nearest neighbor editing

Guan et al. (2009) showed that nearest neighbor editing takes advantage of unlabeled data, labeled by a variant of tri-training where only unlabeled data points that three different classifiers agree on are added to the labeled data. They do two rounds of tri-training to obtain T_U, a pseudo-labeled portion of the unlabeled data U.

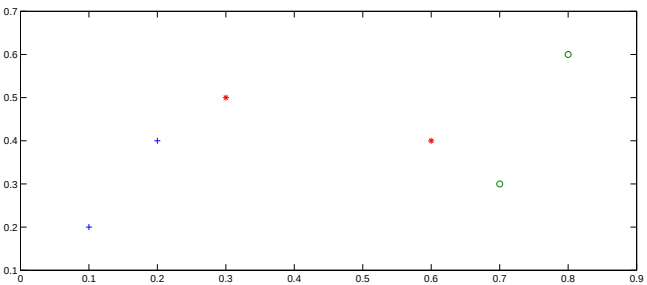
Semi-supervised condensed nearest neighbor

$T = \{\langle \mathbf{x}_1, y_1 \rangle, \ldots, \langle \mathbf{x}_n, y_n \rangle\}$, $C = \emptyset$, $C' = \emptyset$
$U = \{\langle \mathbf{x'}_1 \rangle, \ldots, \langle \mathbf{x'}_m \rangle\}$ # unlabeled data
for $\langle \mathbf{x}_i, y_i \rangle \in T$ **do**
  if $C(\mathbf{x}_i) \neq y_i$ or $P_C(\langle \mathbf{x}_i, y_i \rangle | \mathbf{x}_i) < 0.55$ **then**
    $C = C \cup \{\langle \mathbf{x}_i, y_i \rangle\}$
  end if
end for
for $\langle \mathbf{x'}_i \rangle \in U$ **do**
  if $P_T(\langle \mathbf{x'}_i, T(\mathbf{x'}_i) \rangle | \mathbf{w}_i) > 0.90$ **then**
    $C = C \cup \{\langle \mathbf{x'}_i, T(\mathbf{x'}_i) \rangle\}$
  end if
end for
for $\langle \mathbf{x}_i, y_i \rangle \in C$ **do**
  if $C'(\mathbf{x}_i) \neq y_i$ **then**
    $C' = C' \cup \{\langle \mathbf{x}_i, y_i \rangle\}$
  end if
end for
return C'

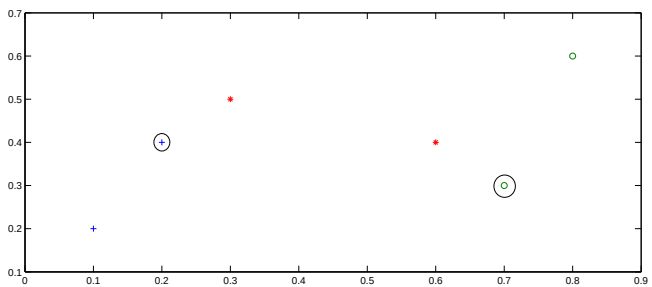
Figure: Semi-supervised condensed nearest neighbor.

Semi-supervised condensed nearest neighbor

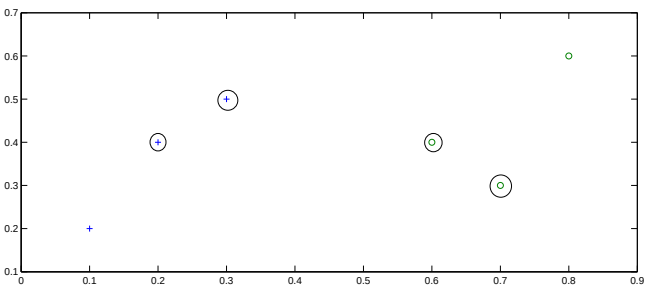
a)



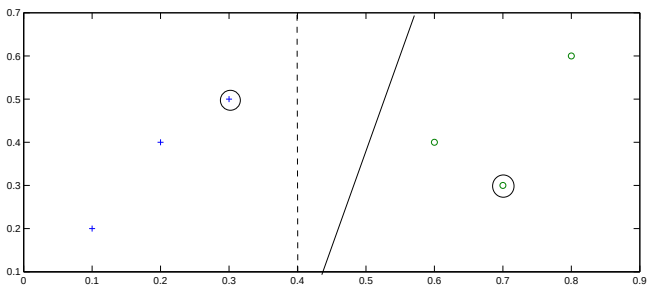
b)



c)



d)



Comparison

dataset	knn	s-knn	lbl	nne	s-nne	cnn	s-cnn
1	81.46	85.17	86.06	78.26(0.96)	80.82(1.86)	81.84(0.10)	86.31
2	62.58	**60.43**	52.47	59.92(0.94)	59.04(1.81)	65.11(0.13)	59.42
3	51.68	49.22	**62.44**	51.55(0.98)	48.96(1.88)	61.53(0.13)	48.96
4	65.48	50.46	69.04	77.71(0.96)	84.52(1.73)	81.11(0.12)	57.30

Figure: $k = 9$

Classification

1/10	cnn		scnn		
	acc	dps	acc	dps	err.red
anneal	81.11	31	**83.33**	40	11.75%
balance-scale	69.84	28	**82.54**	49	42.11%
brown-selected	78.95	5	**89.47**	7	49.98%
bupa	57.14	23	**62.86**	17	13.35%
car	78.61	45	**87.28**	81	40.53%
crx	81.16	15	**84.06**	34	15.39%
ionosphere	**88.89**	11	72.22	13	-
lung	52.38	8	**90.48**	10	80.01%
monks-2	80.33	29	**81.97**	38	8.34%
mushroom	77.61	284	**82.17**	318	20.37%
primary-tumor	52.94	23	**58.82**	25	12.49%
shuttle-landing-control	80.77	8	**96.15**	12	79.98%
tic-tac-toe	34.38	3	**76.04**	24	63.49%
titanic	70.59	18	**76.92**	21	21.52%
wdbc	98.25	10	98.25	13	0
yeast	65.10	78	**69.13**	90	11.55%

Figure: Comparison of CNN and SCNN on 16 classification data sets.

Classification

1/20	cnn		scnn		
	acc	dps	acc	dps	err.red
anneal	66.67	11	**76.19**	23	28.56%
balance-scale	75.56	20	**81.11**	32	22.71%
brown-selected	78.95	5	78.95	5	0
bupa	48.57	11	**54.29**	10	11.12%
car	73.99	27	**77.46**	55	13.34%
crx	72.46	7	**88.41**	22	57.92%
ionosphere	69.44	9	**72.22**	11	9.10%
lung	52.34	5	**71.43**	6	40.05%
monks-2	68.85	18	68.85	23	0
mushroom	**73.31**	160	72.20	184	-
primary-tumor	47.06	12	47.06	12	0
shuttle-landing-control	88.46	8	88.46	8	0
tic-tac-toe	34.38	3	**68.75**	24	52.38%
titanic	82.35	16	82.35	16	0
wdbc	94.74	4	**96.49**	9	36.53%
yeast	58.39	52	**60.40**	57	4.83%

Figure: Comparison of CNN and SCNN on 16 classification data sets.

POS tagging

	English SVMTool			MaxEnt		
	TA	err.red.	*p*-value	TA	err.red.	*p*-value
BL	97.15			96.31		
SVM-stacking	97.19			-		
self-tr	97.26	3.86	<0.0001	96.36	1.36	<0.0001
tri-tr	97.27	4.21	<0.0001	96.36	1.36	<0.0001
tri-disagr	97.27	4.21	<0.0001	96.36	1.36	<0.0001
gen-tri	**97.31**	5.61	<0.0001	**96.57**	7.05	<0.0001

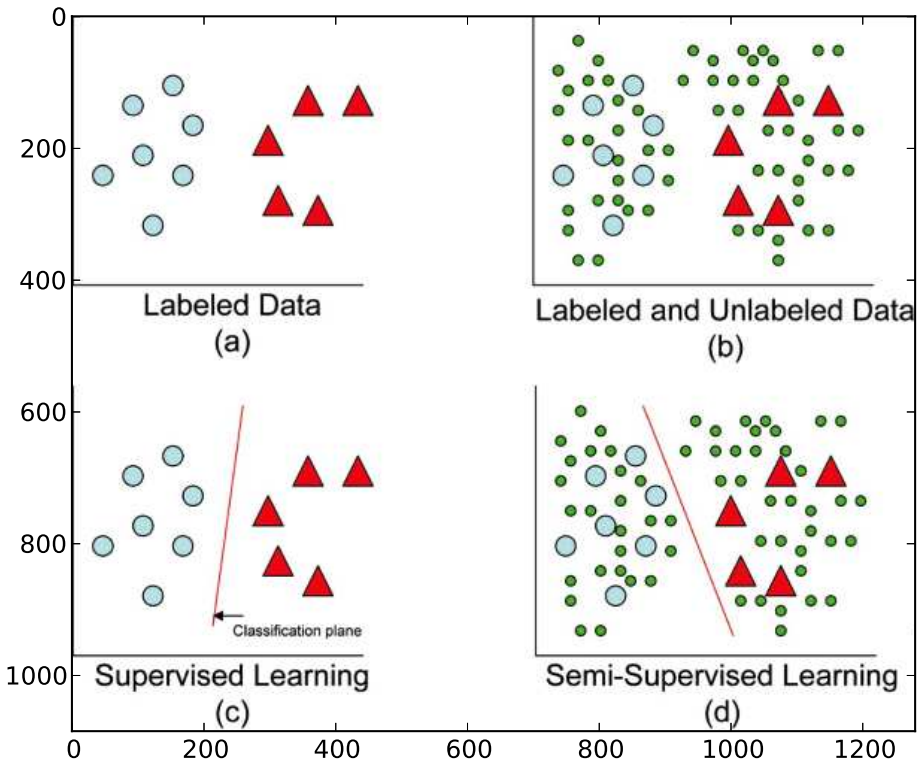
	Dutch			Swedish		
	TA	err.red.	*p*-value	TA	err.red.	*p*-value
BL	94.74			93.44		
stacking	95.09			93.52		
self-tr	95.09	6.65	<0.01	93.56	1.83	-
tri-tr	94.80			91.93		
tri-disagr	94.80			91.85		
gen-tri	**95.72**	18.63	<0.0001	**94.52**	16.46	<0.0001

Dutch Alpino (train-test) and 2,336k Europarl (unl).

Swedish Talbanken05 (train-test) and 1,727k Leipzig Corpora Collection (unl).

Note: Søgaard (2011) reports 97.5 on PTB-23-24 using semi-supervised condensed nearest neighbor; the best reported result so far.

Auxiliary objectives




0
50
100
150
200
250
0
50
100
150
200
250
300
Polydor
Bis
morgen
Ganz
genau
wie
du
RENATE
KERN