Evaluation scripts and the meaning of life

Anders Søgaard

NLP 101

NLP 102

NLP 201

NLP 202

NLP 301

NLP 302

NLP 401

NLP 402

My initial brainstorm

OUTLINE:

1. Why NLP is not just ML literature look-up.
2. Why evaluating NLP is tricky.
3. What I'll advocate:
   a) Significance across datasets.
   b) Meta-analysis.
   c) Correlating performance with data characteristics.
   d) Down-stream performance.

ASSUMPTIONS:

1. Labeled data is scarce.
2. Labeled data is biased.

E.g.

HIT-Baseline	LAS	POS
Wall Street Journal	91.88	97.76
Yahoo Answers!	80.75	90.99
BBC Newsgroups	85.26	92.32
Amazon (reviews)	81.60	90.46

Philosophy jam session

"All science is either physics or stamp collection." (Richard Feynman)

- What's the difference between meteorology and weather reports?
- The difference between scientific computational linguistics and what Hal recently referred to as *data porn*.

"Predictions can be very difficult – especially about the future." (Niels Bohr)

- We want to be able to predict NLP performance on future data. Just like meteorologists want to predict global warming in 2030.
- Significance across datasets, meta-analysis and correlating performance with data characteristics.

Eight courses in four hours

	Stamp collection	**Physics**
Document classification	NLP 101	NLP 102
Domain adaptation	NLP 201	NLP 202
Robust learning	NLP 301	NLP 302
Syntactic parsing	NLP 401	NLP 402

Language as data points

In NLP, we represent language (documents, sentences, words) by arrays of numbers, most often 0s and 1s:

$$\langle 0, 1, 0, 0, 1 \rangle$$

could for example represent the text:

McCain just gave a cheap plug to Ed Kennedy.

The array may be a series of values for the attributes ⟨Obama, McCain, Malcolm X, Mary Poppins, Ed Kennedy⟩, where 1 means that a feature (word) is present in the text, and 0 means it isn't. It is often convenient to store data points as sparse matrices representing only non-zero values:

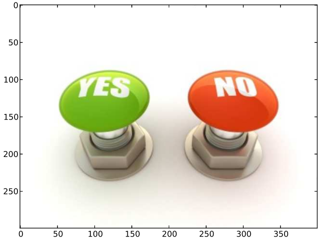
$$\langle 1 : 1, 4 : 1 \rangle$$

Labeled datapoints

Say we are interested in subsets (or classes) of documents, sentences, or words, e.g. positive user reviews, spam emails, sentences with relative clauses, or metaphors. The class of each datapoint is encoded by its associated class label.

$$\langle 1, \langle 0, 1, 0, 0, 1 \rangle\rangle \text{ or, in sparse format: } \langle 1, \langle 1:1, 4:1 \rangle\rangle$$

We write $\mathbf{x}$ for data points and y for class labels. For now class labels are assumed to be binary, i.e. $y \in \{0, 1\}$, but the observed variables can be both discrete (with values 0 and 1) and continuous (real-valued).



20 Newsgroups

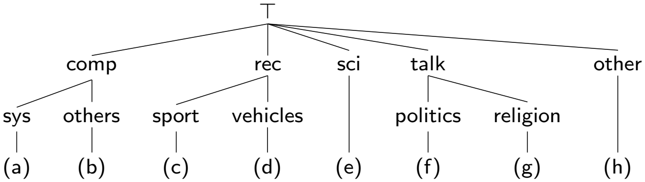


Figure: Hierarchical structure of 20 Newsgroups. (a) IBM, Mac, (b) Graphics, MS-Windows, X-Windows, (c) Baseball, Hockey, (d) Autos, Motorcycles, (e) Cryptography, Electronics, Medicine, Space, (f) Guns, Mideast, Miscellaneous, (g) Atheism, Christianity, Miscellaneous, (h) Forsale

Classification:

$$\arg\max_{y \in \mathcal{Y}} P(y|\mathbf{x})$$

Note: Instead of 20-way classification we will use 20 Newsgroups to perform domain adaptation experiments.

Three basic algorithms

y	zebra	viagra	venus
spam	0	1	0
non-spam	1	0	0
non-spam	1	0	1
?	0	1	1

Figure: Toy example

- Nearest neighbor solves $\arg\min_{\langle y,\mathbf{x}\rangle \in T} D(\mathbf{x}', \mathbf{x})$
- Naive Bayes solves $\arg\min_{y \in T} P(y) \prod_{x_i} P(x_i \mid y)$
- Perceptron solves $\text{sign}(\mathbf{w} \cdot \mathbf{x})$ after maintaining $\mathbf{w}$ in several passes over T modifying $\mathbf{w}$ at a fixed rate (α)

Three basic algorithms

y	zebra	viagra	venus
spam	0	1	0
non-spam	1	0	0
non-spam	1	0	1
?	0	1	1

Figure: Toy example

Nearest neighbor:

	Manhattan	Euclidean
$D(\mathbf{x}_1, \mathbf{x}')$	1	1
$D(\mathbf{x}_2, \mathbf{x}')$	3	$\sqrt{3}$
$D(\mathbf{x}_3, \mathbf{x}')$	2	$\sqrt{2}$
Prediction	spam	spam

Three basic algorithms

y	zebra	viagra	venus
spam	0	1	0
non-spam	1	0	0
non-spam	1	0	1
?	0	1	1

Figure: Toy example

Naive Bayes:

	Unsmoothed	Laplace
P(spam,$\mathbf{x}'$)	$\frac{1}{3}\frac{1}{1}\frac{1}{1}\frac{0}{1}=0$	$\frac{2}{5}\frac{2}{3}\frac{2}{3}\frac{1}{3}\sim 6\%$
P(non-spam,$\mathbf{x}'$)	$\frac{2}{3}\frac{0}{2}\frac{0}{2}\frac{1}{2}=0$	$\frac{4}{5}\frac{1}{4}\frac{1}{4}\frac{2}{4}\sim 3\%$
Prediction	?	spam

Three basic algorithms

y	zebra	viagra	venus
spam	0	1	0
non-spam	1	0	0
non-spam	1	0	1
?	0	1	1

Figure: Toy example

Perceptron: (iters=1,α=0.1)

	w	$\hat{y}$
1	$\langle 0, 0, 0, 0 \rangle$	non-spam (False)
2	$\langle 0, .1, 0, -.1 \rangle$	spam (False)
3	$\langle -.1, .1, 0, 0 \rangle$	non-spam (True)
Prediction	$\langle -.1, .1, 0, 0 \rangle$	spam

Note: The **averaged** perceptron model would be: $\langle -.07, .1, 0, -.3 \rangle$ ...

Comparison 101

On BASEBALL-WINDOWSX:

learner	acc	nn	nb	perc
nn	0.951	-	**	*
nb	0.992	**	-	
perc	0.984	*		-

*: $p < 0.05$, **: $p < 0.01$.

- but what does that *really* tell us?

Student's *t*-test for dependent means (W. Gosset, 1876-1937):

$$t = \frac{M - \mu}{\sqrt{\frac{\frac{SS}{df}}{N}}}$$

Assumes:

(a) data is sampled i.i.d.

(b) data is normally distributed.

Standard assumptions in supervised learning

- Smoothness assumption
- Independently and **identically distributed** (i.i.d.)
- Single cluster assumption
- Low-density separation

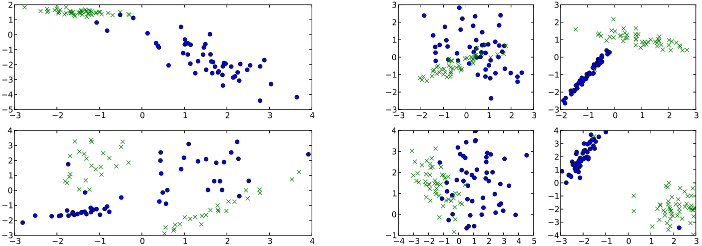


Figure: Binary classification problems where each class consists of one or two clusters (left) and binary classification problems with varying degrees of separability (right)

How to check whether the assumptions hold?

Identically distributed:

- *KL* divergence: cross-entropy $(-\sum_{\mathbf{x}} P(\mathbf{x}) \log Q(\mathbf{x}))$ minus entropy $(-\sum_{\mathbf{x}} P(\mathbf{x}) \log P(\mathbf{x}))$:

$$\sum_{\mathbf{x}} P(\mathbf{x}) \log \frac{P(\mathbf{x})}{Q(\mathbf{x})}$$

- Jensen-Shannon divergence

$$D_{\mathrm{JS}}(P, Q) = \frac{1}{2} D_{\mathrm{KL}}(P, M) + \frac{1}{2} D_{\mathrm{KL}}(Q, M)$$

- Rényi divergence:

$$\frac{1}{\alpha - 1} \log(\sum_{i=1}^{n} \frac{p_i^{\alpha}}{q_i^{\alpha-1}})$$

Coherence and separability:

- within-class scatter

$$\sum_{c} \sum_{i \in c} (\mathbf{x}_i - m_c)(\mathbf{x}_i - m_c)^T$$

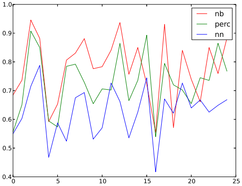
- between-class scatter

$$\sum_{c} (m_c - \bar{\mathbf{x}})(m_c - \bar{\mathbf{x}})^T$$

Comparison 102

Evaluation over 25 randomly selected cross-domain datasets from 20 Newsgroups:

learner	acc	$\rho(KL)$
nb	0.778	-0.36
perc	0.727	-0.21
nn	0.627	-0.39



Question: Why might perceptron be less affected by KL-divergence?

Pearson's ρ (K. Pearson, 1857-1936):

$$\rho = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

Spearman's ρ (C. Spearman, 1863-1945): Use ranks rather than raw scores.

Weighted learning

β	y	zebra	viagra	venus
0.7	spam	0	1	0
0.3	non-spam	1	0	0
0.9	non-spam	1	0	1
	?	0	1	1

Figure: Toy example

Naive Bayes:

	Unsmoothed	Laplace
P(spam,$\mathbf{x}'$)	$\frac{0.7}{1.9}\frac{0.7}{0.7}\frac{0.7}{0.7}\frac{0}{0.7} = 0$	$\frac{1.7}{3.9}\frac{1.7}{2.7}\frac{1.7}{2.7}\frac{1}{2.7} \sim 7\%$
P(non-spam,$\mathbf{x}'$)	$\frac{1.2}{1.9}\frac{0}{1.2}\frac{0}{1.2}\frac{0.9}{1.2} = 0$	$\frac{2.2}{3.9}\frac{1}{3.2}\frac{1}{3.2}\frac{1.9}{3.2} \sim 3\%$
Prediction	?	spam

Question: What is *weighted* nearest neighbor and perceptron learning?

Weighted PA, MIRA and SVM

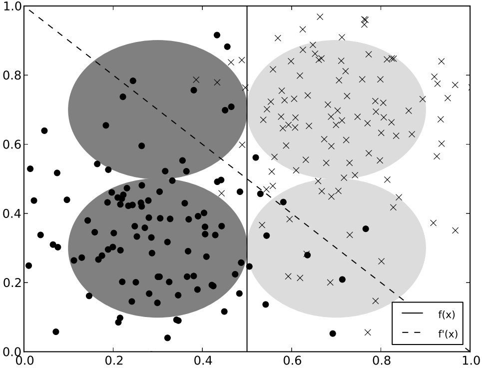
- The passive-agressive algorithm can be weighted by updating by a stepsize $\beta_n\alpha$ where β_n is the instance weight assigned to $\langle y_n, \mathbf{x}_n\rangle$.
- Søgaard and Haulrich (2011) present an instance-weighted version of the MIRA algorithm and apply it to dependency parsing.
- Huang et al. (2007) present an instance-weighted learning algorithm for support vector machines. Here's the SVM objective with a capacity constant C to weight in-sample classification error:

$$\min_{\mathbf{w},b,\xi} C\sum_{i=1}^{N}\xi_i + \lambda||\mathbf{w}||^2 \quad (1)$$
$$\text{s.t. } y_i(\mathbf{w}^\top\mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i \in \{1,\ldots,N\}$$

The weighted objective:

$$\min_{\mathbf{w},b,\xi} C\sum_{i=1}^{N}\beta_i\xi_i + \lambda||\mathbf{w}||^2 \quad (2)$$
$$\text{s.t. } y_i(\mathbf{w}^\top\mathbf{x}_i + b) \geq 1 - \xi_i \text{ for } i \in \{1,\ldots,N\}$$

Motivation for importance weighting



Importance weighting

- What weight function should we use in transfer learning?

Shimodaira (2001) shows that the optimal weight function with sufficiently large samples is

$$\frac{P_T(\mathbf{x})}{P_S(\mathbf{x})}$$

where $P_T(\mathbf{x})$ is the density function in the target domain, and $P_S(\mathbf{x})$ is the density function in the source domain.

- ... but we can't compute density functions.

Importance weighting

You can obtain an estimated importance weight function by:

- domain classification (Zadrozny et al., 2004; Bickel and Scheffer, 2007; Søgaard and Haulrich, 2011),
- perplexity in target domain language model (Søgaard, 2011),
- compute reduced density functions (Søgaard and Plank, 2011),
- kernel mean matching (Huang et al., 2007), or
- minimizing *KL*-divergence (Sugiyama et al., 2007).

Comparison 201

Source	Target	NB	IW-NB	Perc.	IW-Perc.
Hockey-IBM	Baseball-Mac	94.76	95.14	86.32	90.28
Hockey-Crypt	Baseball-Electronics	88.99	90.63	76.58	77.22
Guns-Electronics	MidEast-Medicine	72.93	*65.16*	69.69	71.24
Graphics-Misc(Politics)	Windows-Misc(Religion)	94.58	95.36	89.16	89.94

- Jiang and Zhai (2007) report a 6.6% error reduction for POS tagging.
- Søgaard and Haulrich (2011) report a 3.2% error reduction for dependency parsing.

Comparison 202

	BL	$\rho(Sw)$	$\rho(KL)$	IW	$\rho(Sw)$	$\rho(KL)$
NB	0.720	0.01	-0.02	0.745	-0.12	-0.16
P	0.528	-0.22	-0.27	0.705	-0.01	-0.05

Figure: Results on 25 randomly selected cross-domain datasets from 20 Newsgroups

When the target is unknown

- Importance weighting assumes we can pool unlabeled data from the target distribution,
- but sometimes we can't . . .
- This is the ROBUST LEARNING scenario, but can we make any assumptions about the likely source-target differences?

Out-of-vocabulary effects

- One of the main reasons for performance drops when evaluating supervised NLP models on out-of-domain data is out-of-vocabulary (OOV) effects (Blitzer et al., 2007; Daume and Jagarlamudi, 2011).
- Spelling expansion, morphological expansion, dictionary term expansion, proper name transliteration, correlation analysis, and word clustering still leave us with "removed" dimensions.
- This is a potential source of error.

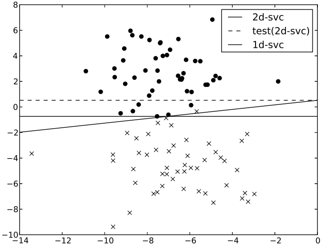


Figure: Optimal decision boundary is not optimal when one dimension is removed

Robust optimization

In robust optimization (Ben-Tal and Nemirovski, 1998) we aim to find a solution $\mathbf{w}$ that minimizes a (parameterized) cost function $f(\mathbf{w}, \xi)$, where the true parameter ξ may differ from the observed $\hat{\xi}$. The task is to solve

$$\min_{\mathbf{w}} \max_{\hat{\xi} \in \Delta} f(\mathbf{w}, \hat{\xi}) \tag{3}$$

with Δ all possible realizations of ξ. An alternative to minimizing loss in the worst case is minimizing loss in the average case, or the sum of losses:

$$\min_{\mathbf{w}} \sum_{\hat{\xi} \in \Delta} f(\mathbf{w}, \hat{\xi}) \tag{4}$$

Robust learning in random subspaces

Let ξ be a binary vector of length M. If $\xi = \langle 1, \ldots, 1 \rangle$ we can write learning linear models such as perceptron as a problem of minimizing expected loss:

$$\min_{\mathbf{w}} \mathbb{E}_{\langle y, \mathbf{x} \rangle \sim \rho} L(y, \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} \circ \xi)) \tag{5}$$

but if we have a set Δ of binary vectors ξ, we can instead minimize average expected loss *under different subspaces*:

$$\min_{\mathbf{w}} \sum_{\hat{\xi} \in \Delta} \mathbb{E}_{\langle y, \mathbf{x} \rangle \sim \rho} L(y, \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} \circ \hat{\xi})) \tag{6}$$

We refer to this idea as robust learning in random subspaces (RLRS).

Robust learning in random subspaces

$X = \{\langle y_i, \mathbf{x}_i \rangle\}_{i=1}^{N}$
for $r \in R$ **do**
  $\mathbf{w}^0 = 0, \mathbf{v} = 0, i = 0$
  $\xi \leftarrow$ **random.bytes**(M)
  for $k \in K$ **do**
    for $n \in N$ **do**
      if $\text{sign}(\mathbf{w} \cdot \mathbf{x} \circ \xi) \neq y_n$ **then**
        $\mathbf{w}^{i+1} \leftarrow$ **update**$(\mathbf{w}^i)$
        $i \leftarrow i + 1$
      end if
      $\mathbf{v} \leftarrow \mathbf{v} + \mathbf{w}^i$
    end for
  end for
end for
return $\mathbf{w} = \mathbf{v}/(N \times K \times R)$

Figure: Robust learning in random subspaces

Robust learning in random subspaces

	P	P-RLRS	err.red	p-value	SGD	SGD-RLRS	err.red	p-value
25	67.2	70.1	0.09	< 0.01	75.2	75.7	0.02	~ 0.17
50	63.8	66.2	0.07	< 0.01	68.6	70.9	0.07	~ 0.02
75	73.2	75.3	0.08	< 0.01	76.3	78.9	0.11	< 0.01
100	72.0	73.3	0.05	~ 0.06	73.6	77.1	0.15	< 0.01
150	72.3	76.2	0.14	< 0.01	74.6	79.2	0.18	< 0.01
250	70.4	72.6	0.07	~ 0.02	75.0	78.7	0.15	< 0.01

Figure: Results on 25 randomly selected cross-domain datasets from 20 Newsgroups

Robust learning in random subspaces

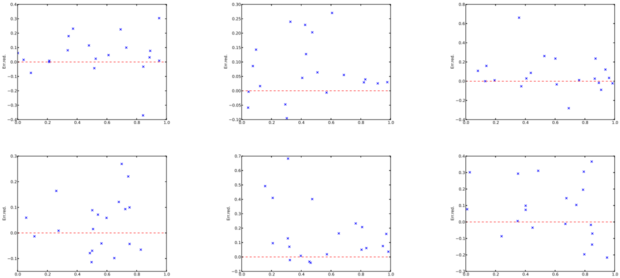


Figure: Plots of P-RLRS error reductions on 20 Newsgroups with $R = 25$ (upper left), $R = 50$ (upper right), $R = 75$ (lower left), $R = 100$ (lower mid), $R = 150$ (lower mid) and $R = 250$ (lower right).

Comparison 302

	P	AP	PA	CW	DS08	L2-SVM	L1-SVM	L2-LR	L1-LR
tw-av	0.834	0.822	0.820	**0.865**	0.789	0.819	0.831	0.836	0.845
tw-sd	0.088	0.120	0.108	0.092	0.140	0.101	0.116	0.100	0.122
dom-av	0.742	0.752	0.773	**0.815**	0.802	0.768	0.735	0.787	0.731
dom-sd	0.131	0.127	0.126	0.129	0.129	0.126	0.125	0.132	0.129
tw+dom-av	0.605	0.578	0.603	**0.647**	0.530	0.588	0.545	0.602	0.538
tw+dom-sd	0.124	0.084	0.104	0.136	0.145	0.122	0.105	0.114	0.116

Table: Comparison of learning algorithms on 30 randomly extracted classification datasets.

	P	AP	PA	CW	DS08	L2-SVM	L1-SVM	L2-LR	L1-LR
tw-av	0.01	0.09	-0.16	-0.13	-0.22	-0.27	0.02	-0.14	0.09
dom-av	**-0.49	**-0.52	*-0.43	*-0.40	**-0.48	**-0.46	**-0.48	*-0.44	-0.45
tw+dom-av	**-0.48	**-0.46	**-0.60	*-0.41	**-0.51	**-0.46	**-0.60	**0.50	**-0.49

Table: Correlation with *KL*-divergence of learning algorithms on 30 randomly extracted classification datasets.

Sequential labeling

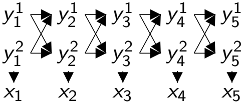
(1) Time flies like arrows.

Figure: Naive Bayes and hidden Markov models as Bayesian networks

$$P(\mathbf{x}) = \prod_{v \in \mathcal{V}} P(x_v \mid x_{\mathrm{pa}(v)})$$

1. Consecutive classification, e.g. transition-based parsing
2. Structured prediction, e.g. MST: score all edges and search for sequence/tree/dag. . .

HMM $(\mathcal{Y}+1 \times N)$:



MST $(N+1 \times N(\times \mathcal{Y}))$

POS and grammatical functions

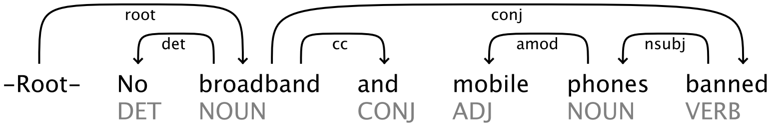


Figure: Syntactic analysis: POS and grammatical functions (Web 2.0 Collection)

Why semi-supervised and cross-domain parsing?

- Annotation rate $\sim$ 40 words per hour. Entire treebank: 5 years.
- How many treebanks?
- Well ...

languages $\times$ domains $\times$ language change $\times$ within-population variation

- Supervised parsing, total cost: $6,909 \times \infty \times \kappa \times \nu_L \times 5$ years $\sim$ Expensive

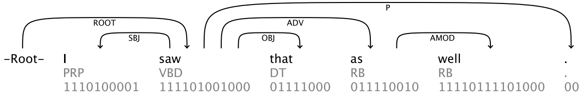
Why semi-supervised and cross-domain parsing?

HIT-Baseline	LAS	POS
Wall Street Journal	91.88	97.76
Yahoo Answers!	80.75	90.99
BBC Newsgroups	85.26	92.32
Amazon (reviews)	81.60	90.46

Charniak (from McClosky et al., 2010)	F-score
Wall Street Journal	89.7
Brown (WSJ)	84.1
Genia (med.)	76.2
Switchboard (spoken)	76.7

What's in our toolbox?

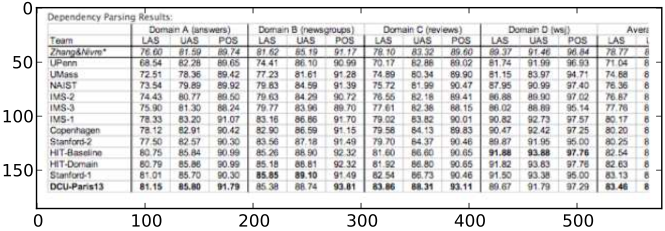
- Co-training (Sagae and Tsujii, 2007).
- Clusters-as-features (Koo et al., 2008, Turian et al., 2010, Rishøj and Søgaard, 2010).
- Stacking on unsupervised models (Suzuki et al., 2008; Suzuki et al., 2009).
- Tri-training (Søgaard and Rishøj, 2010).



UAS	malt-mst2	S3VMs	self-training	orig-tri-training	co-forests	tri-training	tri-training[full]
Danish	90.50	90.47	89.68	89.66	88.79	**90.60**	92.21
Dutch	84.58	85.34	84.06	83.83	83.97	**86.07**	88.06
German	90.53	90.15	89.83	89.92	88.47	**90.81**	93.20
Portuguese	88.80	65.64	87.60	87.62	87.06	**89.16**	91.87
Swedish	89.83	81.46	89.09	89.20	88.65	**90.22**	92.24
AV	88.80	82.61	88.05	88.05	87.44	**89.37**	91.52

Figure: From Søgaard and Rishøj (2010): Comparison of different semi-supervised learning algorithms (10% of unlabeled data) using 2-fold CV and no reparsing, UAS **including punctuation**.

SANCL 2012 shared task (Parsing the Web)



Dependency Parsing Results:

	Domain A (answers)			Domain B (newsgroups)			Domain C (reviews)			Domain D (wsj)			Avera	
Team	LAS	UAS	POS	LAS	UAS	POS	LAS	UAS	POS	LAS	UAS	POS	LAS	
*Zhang&Nivre**	*76.60*	*81.59*	*89.74*	*81.62*	*85.19*	*91.17*	*78.10*	*83.32*	*89.60*	*89.37*	*91.46*	*96.84*	*78.77*	*8*
UPenn	68.54	82.28	89.65	74.41	86.10	90.99	70.17	82.88	89.02	81.74	91.99	96.93	71.04	8
UMass	72.51	78.36	89.42	77.23	81.61	91.28	74.89	80.34	89.90	81.15	83.97	94.71	74.68	8
NAIST	73.54	79.89	89.92	79.83	84.59	91.39	75.72	81.99	90.47	87.95	90.99	97.40	76.36	8
IMS-2	74.43	80.77	89.50	79.63	84.29	90.72	76.55	82.18	89.41	86.88	89.90	97.02	76.87	8
IMS-3	75.90	81.30	88.24	79.77	83.96	89.70	77.61	82.38	88.15	86.02	88.89	95.14	77.76	8
IMS-1	78.33	83.20	91.07	83.16	86.86	91.70	79.02	83.82	90.01	90.82	92.73	97.57	80.17	8
Copenhagen	78.12	82.91	90.42	82.90	86.59	91.15	79.58	84.13	89.83	90.47	92.42	97.25	80.20	8
Stanford-2	77.50	82.57	90.30	83.56	87.18	91.49	79.70	84.37	90.46	89.87	91.95	95.00	80.25	8
HIT-Baseline	80.75	85.84	90.99	85.26	88.90	92.32	81.60	86.60	90.65	**91.88**	**93.88**	**97.76**	82.54	8
HIT-Domain	80.79	85.86	90.99	85.18	88.81	92.32	81.92	86.80	90.65	91.82	93.83	97.76	82.63	8
Stanford-1	81.01	85.70	90.30	**85.85**	**89.10**	91.49	82.54	86.73	90.46	91.50	93.38	95.00	83.13	8
DCU-Paris13	**81.15**	**85.80**	**91.79**	85.38	88.74	**93.81**	**83.86**	**88.31**	**93.11**	89.67	91.79	97.29	**83.46**	**8**

Our results

	emails LAS	emails UAS	blogs LAS	blogs UAS
Baseline	75.28	79.73	82.66	85.98
Non-weighted	76.04	80.26	83.79	86.92
Bagging	76.80	81.00	84.28	87.15
wh-words	76.64	80.95	84.12	87.19
Commas	76.20	80.64	83.63	86.60
LSI ($k = 100$)	76.55	76.18	84.02	87.08
LSI ($k = 200$)	76.18	80.57	83.99	86.98
ppl	76.41	80.89	83.91	86.93
wh-words ($Q = 100$)	76.37	80.60	83.73	86.73
LSI ($k = 100$, $Q = 100$)	76.40	80.89	84.12	87.17
System	**77.07**	**81.27**	**84.42**	**87.35**

Table: Parsing results on development data **incl. punctuation**.

	answers	emails	newsgroups	reviews	blogs
Non-weighted	*83.04*	81.26	85.79	*84.95*	*86.98*
SH11	81.70	80.76	84.92	83.73	85.52
Commas	82.46	81.14	85.48	84.17	86.02
wh-words	82.98	81.36	85.90	84.84	86.38
LSI ($k = 100$)	82.84	81.29	85.80	84.83	86.32
LSI ($k = 200$)	82.96	*81.46*	85.82	84.83	86.41
ppl	82.94	81.38	*86.10*	84.73	86.62
System:	**83.57**	**81.92**	**86.50**	**85.46**	**87.03**

Table: Parsing results (LAS) on test data **incl. punctuation**.

More work on importance weighting

- Tan and Cheng (2009) show how to combine SCL and importance weighting for sentiment analysis.
- Foster et al. (2010) use importance weighting in the context of SMT with 3+ BLEU absolute improvements.
- Søgaard (2011) apply importance weighting to cross-language dependency parsing with up to 60% error reductions.
- Søgaard and Wulff (COLING 2012) apply importance weighting to cross-language dependency parsing with a 3% average error reduction.

Structured perceptron – POS tagging

	AP	AP-RLRS$_{K=25}$	AP-RLRS$_{K=50}$	AP-RLRS$_{K=100}$
EWT-Answers	85.22	85.63	**85.69**	85.68
EWT-Newsgroups	86.82	87.26	**87.36**	87.26
EWT-Reviews	84.92	85.32	85.31	**85.35**
EWT-Weblogs	87.00	87.54	87.52	**87.61**

Table: Results on the EWT (Søgaard and Johannsen, COLING 2012)

Structured perceptron – dependency parsing

	AP	AP-RLRS$_{K=5}$	AP-RLRS$_{K=10}$	AP-RLRS$_{K=15}$
EWT-Email(dev)	81.52	81.68	81.84	81.49
EWT-Answers	81.45	**81.90**	81.23	81.39
EWT-Newsgroups	83.88	**84.14**	83.80	83.85
EWT-Reviews	82.97	**83.03**	82.64	82.85
EWT-Weblogs	**84.88**	84.83	84.71	84.74

Figure: Results (UAS) on the EWT transition-based dependency parsing

	MIRA	M-RLRS$_{0.01}$	M-RLRS$_{0.03}$	M-RLRS$_{0.05}$
EWT-Answers	78.63	**78.81**	78.80	78.75
EWT-Emails	78.09	77.94	**78.19**	77.47
EWT-Newsgroups	82.21	**82.25**	82.07	81.81
EWT-Reviews	80.47	80.68	**80.67**	80.37
EWT-Weblogs	82.36	82.43	**82.57**	82.02

Figure: Results (LAS) on the EWT graph-based dependency parsing doing two passes over data (deleting positive and negative updates)

Comparison 402: Down-stream evaluation

	bl	C07	LTH
DEPRELS	-	21	41
PTB-23 (LAS)	-	**88.79**	87.64
PTB-23 (UAS)	-	90.26	**90.28**
NegSco-all	-	89.71	**90.09**
NegSco-neg	-	43.75	**45.83**
SentComp	35.64	33.68	**38.71**
SMT-dev-Meteor	35.83	36.12	**36.21**
SMT-test-Meteor	37.26	37.62	**37.68**
SMT-dev-BLEU	13.74	13.94	**14.10**
SMT-test-BLEU	14.80	15.09	**15.12**
SRL-22-gold	-	85.29	**86.26**
SRL-23-gold	-	88.33	**89.11**
SRL-22-pred	-	**75.78**	65.29
SRL-23-gold	-	**69.13**	59.01
Sum-R1	-	0.333	**0.358**
bitterlemons.org	96.02	95.52	**96.52**

Table: Comparison of tree-to-dependency conversions

Significance testing

Three steps to enlightenment:

1. Significance across datasets, not across data points.

2. Using a paired t-test for comparing two systems on m datasets has three weaknesses:
   (a) Scores need to be commensurable for averaging to make sense.
   (b) Unless we have about thirty or more datasets, the performances across datasets need to be normally distributed, which they probably are not.
   (c) The t-test is very sensitive to outliers.

- Demsar (2006) proposes to use Wilcoxon signed rank test.
- NEW THING: meta-analysis on error reductions.

3. *CORRELATIONS, NOT SCORES.

*Does not rely on datasets being commensurable.

Wilcoxon's signed-rank test:

1. Order (Y_i, X_i) by abs$(\cdot)$, excl. ties.
2. Compute

$$z = \frac{(\sum_{i=1}^{n}[\text{sign}(Y_i > X_i) * \text{rank}_i]) - .5}{\sqrt{\frac{m(m+1)(2m+1)}{6}}}$$

Appendix: Crash-course in meta-analysis

The **fixed effects** model:

$$w_i = \frac{1}{v_i}$$

$$\hat{T} = \frac{\sum_{i \geq 1}^{M} w_i T_i}{\sum_{i \geq 1}^{M} w_i}$$

$$v = \frac{1}{\sum_{i \geq 1}^{M} w_i}$$

The 95% confidence interval is:

$$\hat{T} \pm 1.96\sqrt{v}$$

The **random effects** model:

$$w_i = \frac{1}{v_i + \tau^2}$$

with

$$\tau^2 = \frac{\sum_{i\geq 1}^{k} w_i T_i^2 - \frac{(\sum_{i\geq 1}^{k} w_i T_i)^2}{\sum_{i\geq 1}^{k} w_i} - df}{\sum_{i\geq 1}^{k} w_i - \frac{\sum_{i\geq 1}^{k} w_i^2}{\sum_{i\geq 1}^{k} w_i}}$$

$$\hat{T} = \frac{\Sigma_{i\geq 1}^{M} w_i T_i}{\sum_{i\geq 1}^{M} w_i}$$

$$v = \frac{1}{\sum_{i\geq 1}^{M} w_i}$$

The 95% confidence interval is:

$$\hat{T} \pm 1.96\sqrt{v}$$

Meta-analysis of error reduction

- Error reductions are more commensurable than accuracies.
- Question: Can we assume a true effect size?

Using **fixed effects** model:

```
20 Newsgroups:  160 ['comp.sys.ibm.pc.hardware', 'talk.politics.misc'] -> ['comp.sys.mac.hardware',
'talk.religion.misc']
20 Newsgroups:  81 ['comp.os.ms-windows.misc', 'rec.motorcycles'] -> ['comp.sys.ibm.pc.hardware',
'rec.sport.baseball']
20 Newsgroups:  225 ['rec.motorcycles', 'sci.electronics'] -> ['rec.sport.hockey', 'sci.space']
20 Newsgroups:  49 ['comp.graphics', 'talk.politics.guns'] -> ['comp.os.ms-windows.misc', 'talk.politics.misc']
20 Newsgroups:  144 ['comp.sys.ibm.pc.hardware', 'sci.electronics'] -> ['comp.sys.mac.hardware', 'sci.med']
20 Newsgroups:  125 ['comp.os.ms-windows.misc', 'talk.politics.misc'] -> ['comp.windows.x', 'talk.religion.misc']
20 Newsgroups:  251 ['rec.sport.baseball', 'talk.politics.misc'] -> ['rec.sport.hockey', 'talk.religion.misc']
20 Newsgroups:  31 ['comp.graphics', 'sci.crypt'] -> ['comp.sys.mac.hardware', 'sci.med']
20 Newsgroups:  169 ['comp.sys.mac.hardware', 'sci.crypt'] -> ['comp.windows.x', 'sci.med']
20 Newsgroups:  14 ['comp.graphics', 'rec.motorcycles'] -> ['comp.sys.ibm.pc.hardware', 'rec.sport.baseball']
macro-av.:  0.179257496159
weighted mean:  0.272335199447

95% conf.int.:  0.23473701143 <--> 0.309933387465
```

The **random effects** model:

```
20 Newsgroups:  160 ['comp.sys.ibm.pc.hardware', 'talk.politics.misc'] -> ['comp.sys.mac.hardware',
'talk.religion.misc']
20 Newsgroups:  81 ['comp.os.ms-windows.misc', 'rec.motorcycles'] -> ['comp.sys.ibm.pc.hardware',
'rec.sport.baseball']
20 Newsgroups:  225 ['rec.motorcycles', 'sci.electronics'] -> ['rec.sport.hockey', 'sci.space']
20 Newsgroups:  49 ['comp.graphics', 'talk.politics.guns'] -> ['comp.os.ms-windows.misc', 'talk.politics.misc']
20 Newsgroups:  144 ['comp.sys.ibm.pc.hardware', 'sci.electronics'] -> ['comp.sys.mac.hardware', 'sci.med']
20 Newsgroups:  125 ['comp.os.ms-windows.misc', 'talk.politics.misc'] -> ['comp.windows.x', 'talk.religion.misc']
20 Newsgroups:  251 ['rec.sport.baseball', 'talk.politics.misc'] -> ['rec.sport.hockey', 'talk.religion.misc']
20 Newsgroups:  31 ['comp.graphics', 'sci.crypt'] -> ['comp.sys.mac.hardware', 'sci.med']
20 Newsgroups:  169 ['comp.sys.mac.hardware', 'sci.crypt'] -> ['comp.windows.x', 'sci.med']
20 Newsgroups:  14 ['comp.graphics', 'rec.motorcycles'] -> ['comp.sys.ibm.pc.hardware', 'rec.sport.baseball']
macro-av.:  0.179257496159
weighted mean:  0.262148982429

95% conf.int.:  -2.61333033369 <--> 3.13762829855
```

Caveat – Anderson-Darling test results:

```
norm :  (1.7242089421545437, array([ 0.514, 0.586, 0.703, 0.82 , 0.975]), array([
15.  , 10.  , 5.  , 2.5, 1.  ]))
logistic :  (1.0895338931557852, array([ 0.422, 0.557, 0.653, 0.761, 0.897, 1.
]), array([ 25.  , 10.  , 5.  , 2.5, 1.  , 0.5]))
gumbel :  (0.70544064245152072, array([ 0.456, 0.612, 0.728, 0.843, 0.998]),
array([ 25.  , 10.  , 5.  , 2.5, 1.  ]))
```

Gumbel distribution (E. Gumbel, 1891-1966):

$$\frac{1}{\beta}e^{z-e^{-z}}$$

where:

$$z = \frac{x-\mu}{\beta}$$

```
In[5]:  random.gumbel(0,2,size=10)
Out[5]:  array([ 0.75703198, 2.84769697, -0.13214758, 2.41533214, -1.34462822,
2.14806616, 5.02817841, 13.78349016, 1.06106743, -1.46889266])
```