Automatic categorical coding of transcribed talk: a system that forecasts its own future performance

Richard Forsyth, David Clarke, & Shaaron Ainsworth

School of Psychology 0115-951 5281
rsf@psychology.nottingham.ac.uk

E·S·R·C ECONOMIC & SOCIAL RESEARCH COUNCIL

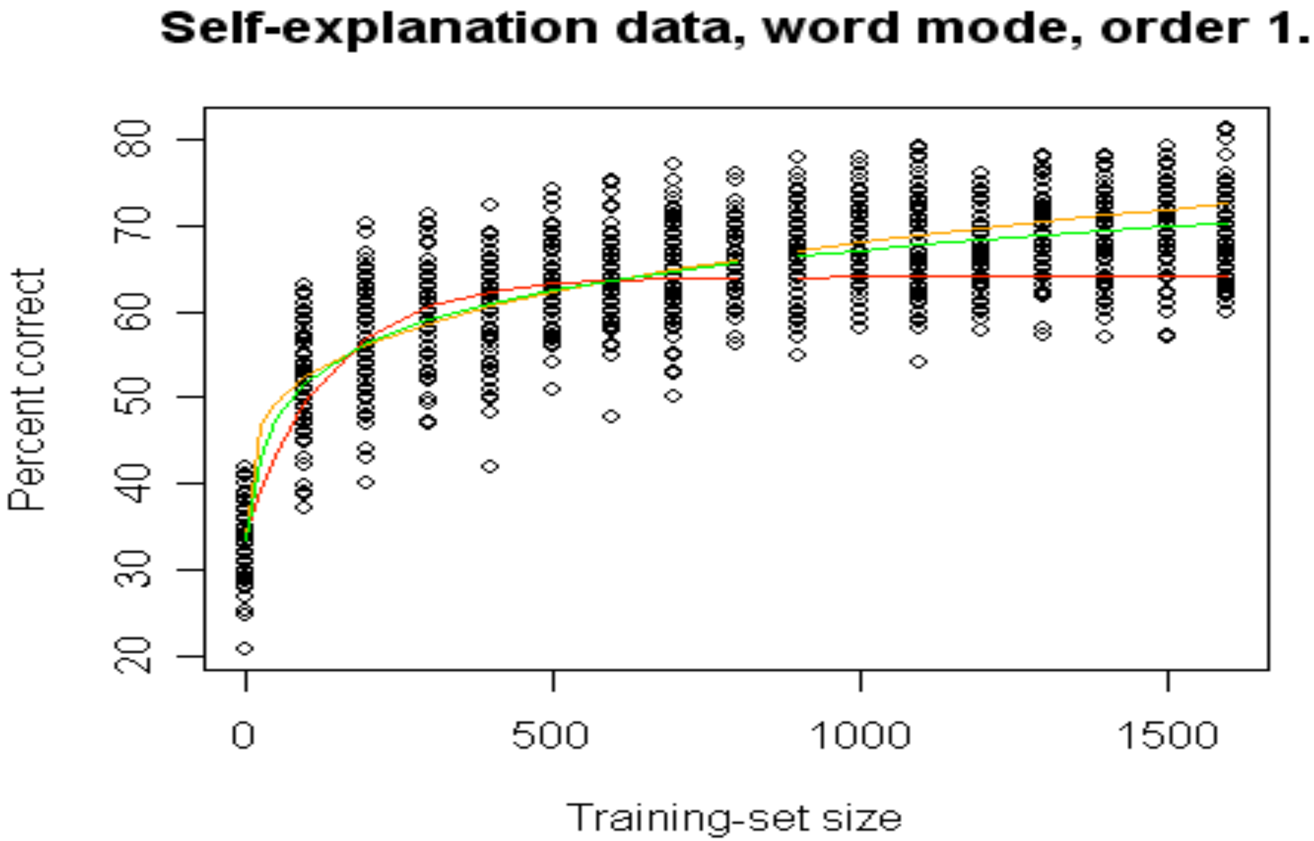
CAQDAS07 / 19 April 2007

NCeSS National Centre for e-Social Science

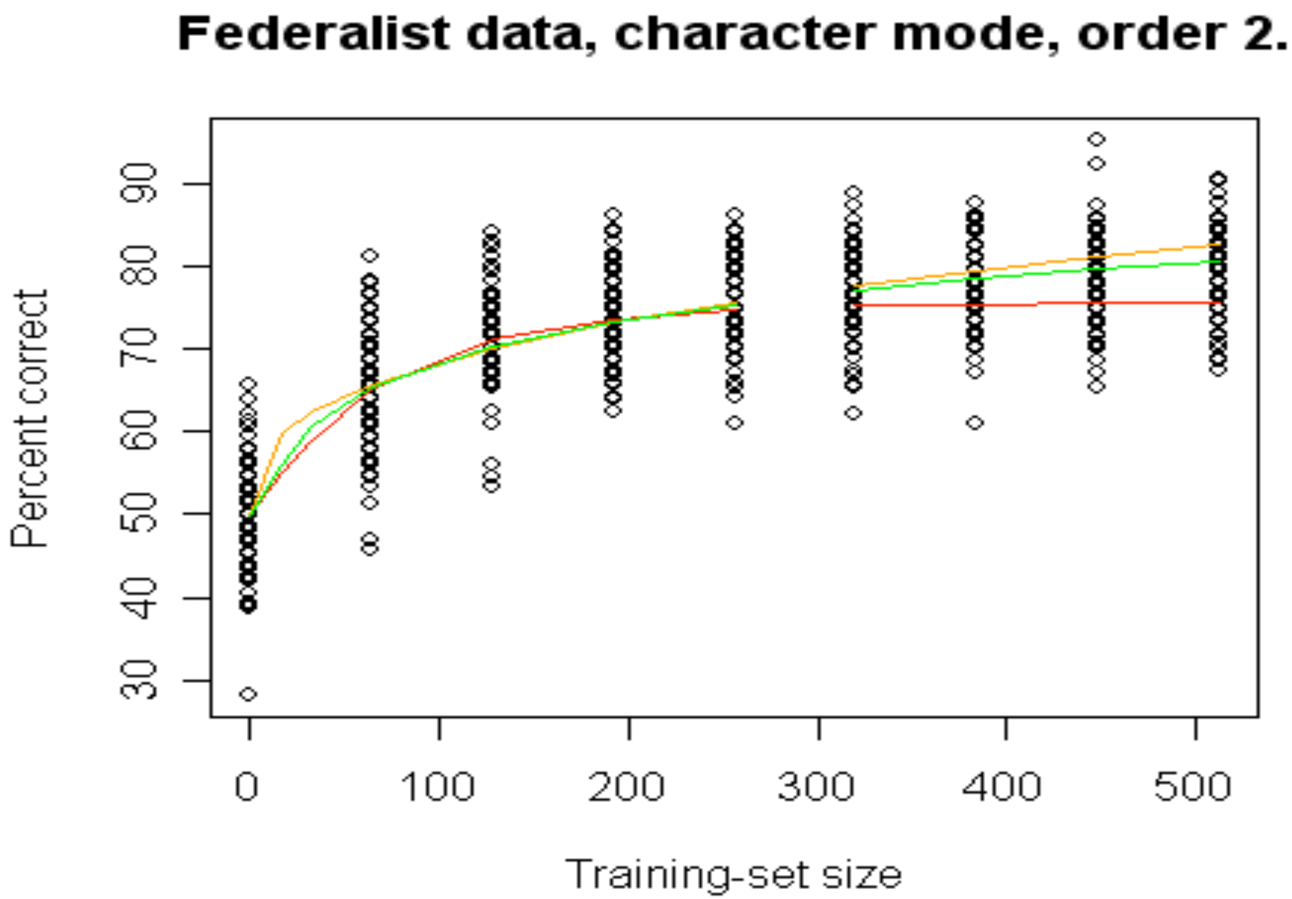
Outline (back to front)

- Some “learning curves”
  - Data and 3 fitted lines
- A solution to a problem you don’t think you have
  - Not “the” solution
  - Why show graphs & formulae to a qualitative audience?
- Where did these learning curves come from?
  - Data Sets
  - CODELEARNER project
  - Learning algorithm
  - Models of the learning curve
- Why this may be of interest
  - Automated coding must be adaptive, should be iterative
  - Self-prediction will save wasted effort
- Discussion

A learning curve that fills me with glee!



And another ("Goldilocks effect")



Dataset details, Self-Explanation data:

- Learning topic = cardiovascular system

Dataset	Self-explanation transcripts (Ainsworth et al., 2007)
Participants	24 (13 female, 11 male)
Size	23330 words
Segments	1784 (mean length = 13.08 words)
Categories	3: monitoring = 63, paraphrase = 1022, self-explanation = 699

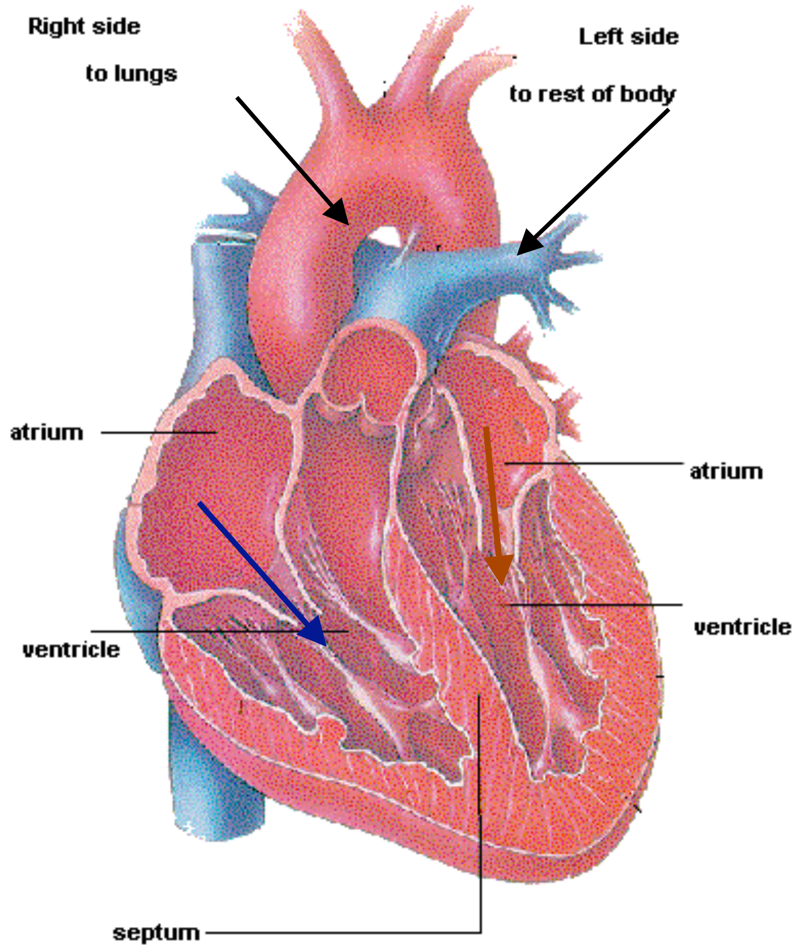
Self-explanation data, 3 main categories:

Textual Material

"The septum divides the heart lengthwise into two sides"

3 codes:

- Paraphrase,
  - The septum is what goes down the middle of the heart
- Self-explanation,
  - Septum is what separates the two ... some sort of control
- Monitoring-statement,
  - I'm not sure why



Dataset details: Federalist papers

- Classic authorship problem

Dataset	Federalist Essays (+2), 17 by Hamilton, 16 by Madison http://www.yale.edu/lawweb/avalon/federal/fed83.htm
Participants	2 (2 male)
Size	84594 words
Segments	583 (mean length = 145.1 words)
Categories	2: Hamilton = 259, Madison = 324

CodeLearner: Main aim & context

- Objective:
  - (semi-)automatic classifier to assist categorical coding
- Context:
  - Most coding schemes novel
  - ==> **trainable** classifier essential (machine learning)
  - Human effort (to be economized) expended **iteratively**:
    - Code another block of text segments by expert
    - Test learning system on cases so far (x-validated)
    - Decide whether to continue:
      - Stop, accuracy good enough
      - Abandon, accuracy will never be good enough
      - Code more cases (accuracy level will be ok with reasonable effort)
  - ==> system must **self-predict** its future performance

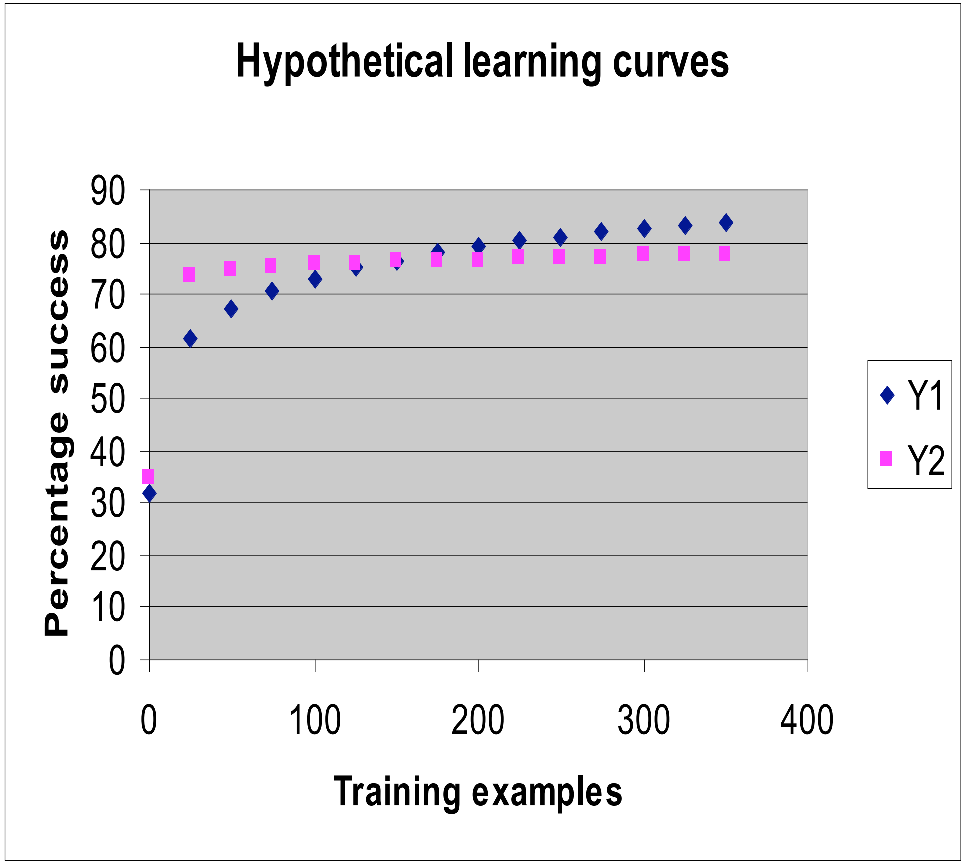
Learning Algorithm

- Hybrid algorithm: “Naïve Markov Classifier”
  - N-gram **Markovian model** at character or word level
  - Naïve **Bayesian inference** for probabilistic classification
  - (“m-estimate” for attenuating probabilities)
  - (Naïve Bayes used in many spam-detection systems)

- Embedded within iterative test harness
  - Allows analysis of “learning curves”
  - Also allows testing of self-predictions
  - N.B. testing always on **unseen** examples

Key desiderata for a Code-learner

- Accuracy
  - Learns well (final height)
- Economy
  - Learns fast (initial gradient)
- Self-Prediction
  - Forecasts its own future performance



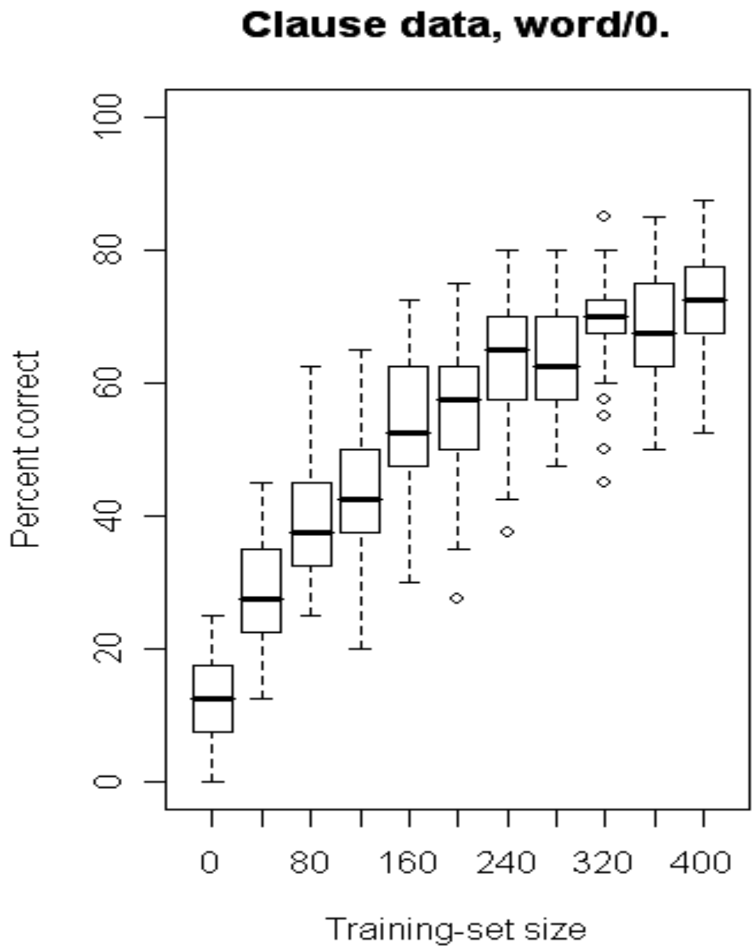
What do we mean by Self-prediction?

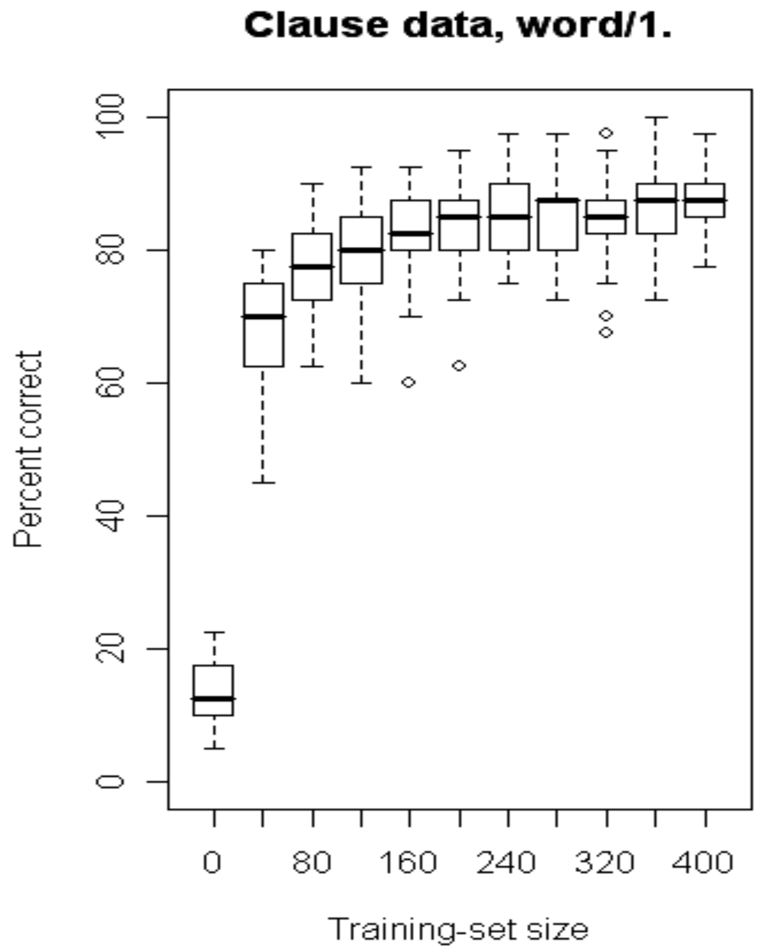
- System trained on small amount of examples, accurately forecasts its performance on large number of examples

- Ideally:
  - The further ahead the better
  - The fewer examples the better
  - The more accurate the better

- 80/20 bad; 20/80 good!

Specimen learning curves (slow & steady; fast but flat)





Fitting the “learning curve” / “experience curve”

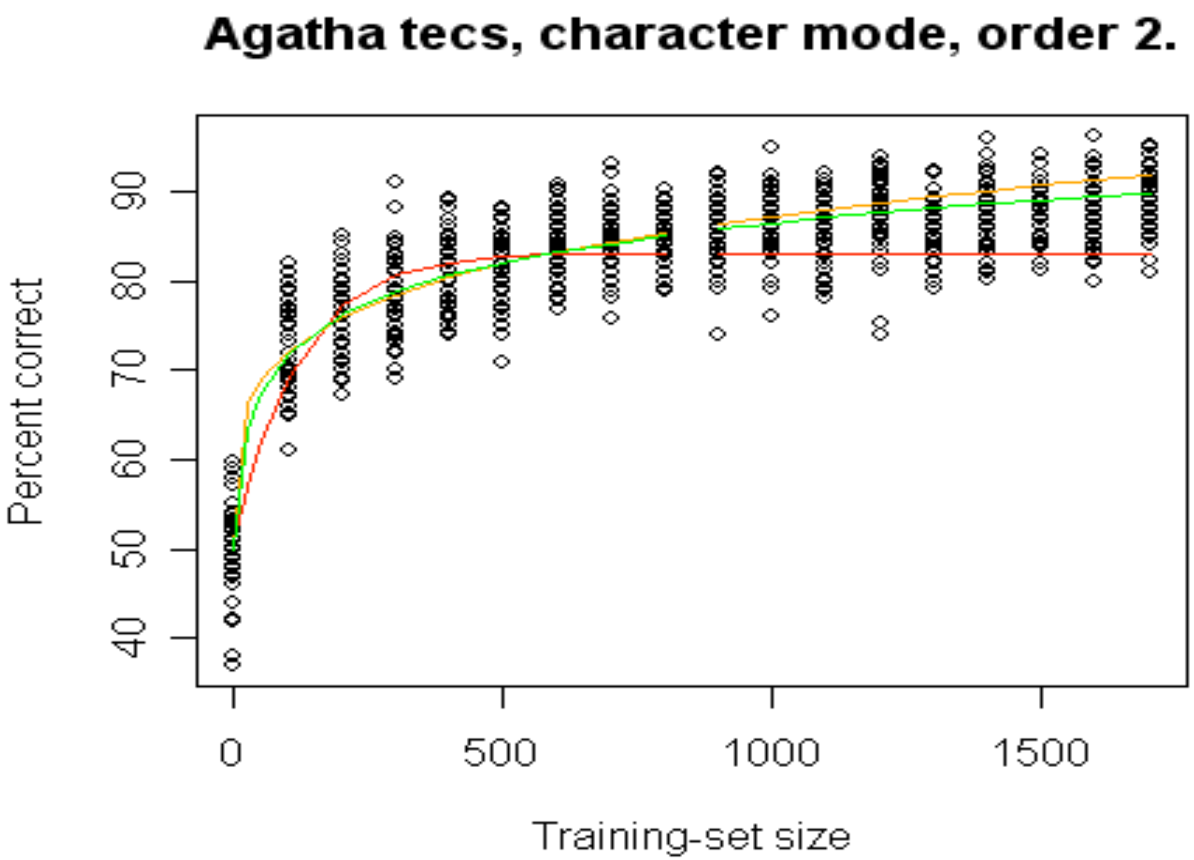
- 3 formulae tried
  - Power, Exponential, Log-reciprocal

- Y = a + b * x ^ c
  - Wright (1936), management science
  - e.g. cost per unit declines as production continues
- Y = a + b * (1 – 10 ^ (c * x))
  - Hull (1943), psychology
  - e.g. time for rat to find food decreases with repeated trials
- Y = a + b * ln(x+1) + c * 1/(x+1)
  - Forsyth (2006), machine learning (ad hoc curve-fitting)
  - e.g. error rate goes down as size of training data goes up

Curve-fitting:

- Interpolation:
  - Estimating within data range used to optimize coefficients of model
- Extrapolation:
  - Predicting outside data range used to optimize coefficients of model
- Quality score:
  - Usually mean squared deviation between real and fitted data values

The Goldilocks effect, again

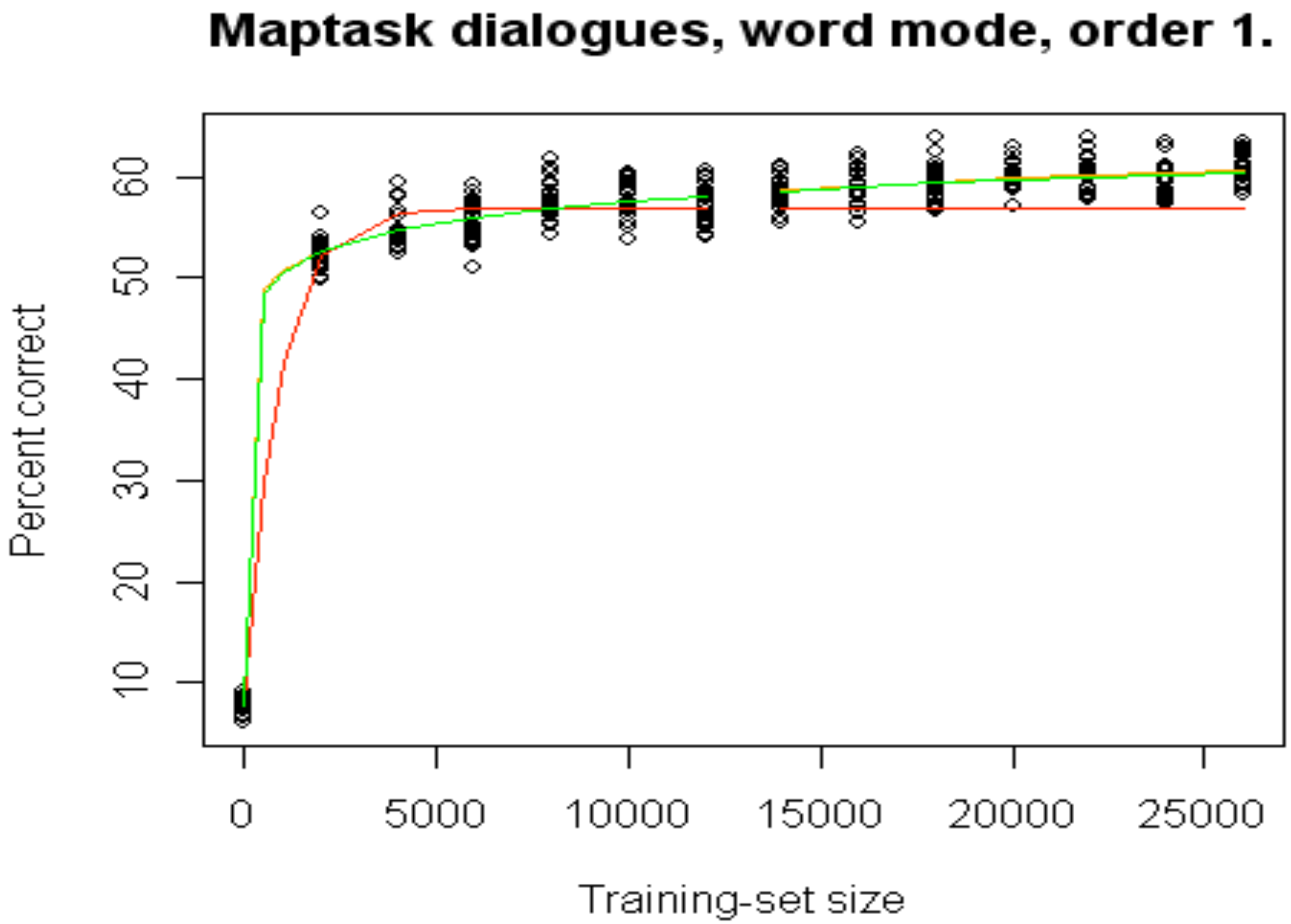


Dataset details, Tectalk data:

"Spoken" dialogue by Marple or Poiroit

Dataset	"talk" from 16 detective novels by Agatha Christie (8 Jane Marple, 8 Hercule Poirot)
Participants	2 fictional! (1 female, 1 male)
Size	50754 words
Segments	1810 (mean length = 28.04 words)
Categories	2: Marple = 1065, Poirot = 745

Silverlocks??



Dataset details, Maptask dialogues

- Dialogue-act classification

Dataset	MapTask dialogues (n=128) http://www.hcrc.ed.ac.uk/maptask
Participants	64 (32 female, 32 male)
Size	156310 words
Segments	27084 (mean length = 5.77 words)
Categories	13: acknowledge = 5605, align = 1778, check = 3137, clarify = 1193, explain = 2160, instruct = 4267, query-w = 772, query-yn = 1758, ready = 2062, reply-n = 884, reply-w = 916, reply-y = 3230, uncodable = 322

Log-reciprocal is the winner

- 8 trials: 4 datasets, 2 unit modes

- Log-reciprocal always best (8/8)
  - In terms of mean squared deviation on **extrapolations**
  - Exponential rubbish
  - Power law versus Log-reciprocal:
    - Student's $t = 3.4$, $df = 7$, $p = 0.01144$

- Extrapolation MORE accurate than interpolation!
  - Exponential E/I = 166% (66% worse)
  - Power-law E/I = 97%
  - Log-reciprocal E/I = 85% (15% better!)

So what?

- If you're like most social scientists, you'll have plenty of short text segments to code
- If you're like most social scientists, you'll have a non-standard coding scheme
- If you have plenty of short text segments to code with a non-standard coding scheme, you'll want a trainable system to do most of the work
- If you want a trainable system to do most of the work, you'll need to know when to stop training it
- If you need to know when to stop training it, it will need to predict its future performance
- Q.E.D.

E·S·R·C ECONOMIC & SOCIAL RESEARCH COUNCIL

NCeSS National Centre for e-Social Science

That's all folks

- Thank you for your attention

☺

[Thanks to CODELEARNER team:
  Shaaron Ainsworth
  David Clarke
  Richard Forsyth
  Claire O'Malley,
and our Sponsors (below).]

E·S·R·C ECONOMIC & SOCIAL RESEARCH COUNCIL

NCeSS National Centre for e-Social Science

X. Discussion points

- NMC gives respectable performance
  - Other algorithms to be tried
- Log-reciprocal formula best for self-prediction (so far)
  - Beats power law (Management Science tradition)
  - Beats exponential law (Learning Theory tradition)

- Key point:
  - Iterative expert coding till automatic system takes over
    - Therefore system must **self-predict**
  - Standard machine-learning systems don’t do this
    - Therefore our simple model is probably best around

X. Text categorization

- Disciplinary differences in approach

- Linguistics:
  - Tagging (PoS, semantic)
  - Mostly at word level
- Computing:
  - Classifying (authorship, content)
  - Mostly at document level
- Social Sciences:
  - Categorical coding
  - Mostly at “segment” level (phrase, utterance)

X. Naïve Markov classifier

- Why I like this algorithm:

- Is fast & not very memory-hungry
- Has no pre-processing phase
- Needs no lexicons or external support s/w
- Has no variable-selection phase
  - (therefore less danger of overfitting)
- Uses **all** the data of a given type
- Has a Bayesian underpinning
- Is highly generic
- Can work in almost any language
  - (in principle could handle DNA sequences etc.)