Document Recognition Without Strong Models

Henry S. Baird

Computer Science & Engineering

Lehigh University

(Based on work by & with T. Pavlidis, T. K. Ho, D. Ittner, K. Thompson, G. Nagy, R. Haralick, T. Hong, T. Kanungo, P. Chou, D. Lopresti, G. Kopec, D. Bloomberg, A. Popat, T. Breuel, E. Barney Smith, P. Sarkar, H. Veeramachaneni, J. Nonnemaker, and P. Xiu.)

How to Find Good Problems?

When I was finishing my Ph.D. dissertation,
my advisor Ken Steiglitz said to me:

> **"There are a lot of smart people out there who,**
> **if you hand them a hard problem,**
> **they can solve it.**
>
> **But, picking good problems is a rarer skill."**

At Bell Labs in 1984, I was free to choose any problem I liked…

Document Image Recognition?

I had been interested for years in Computer Vision.
I asked myself: what seems to be missing ….?

Strategic problem:

Vision systems were brittle:
overspecialized & hard to engineer.

Theo Pavlidis & I debated, & decided:
We'd try to invent highly versatile CV systems.
Tactical goal: **Read *any* page of printed text.**
Open, hard, potentially useful…

But, could this help solve the strategic problem? (DARPA had doubts…)

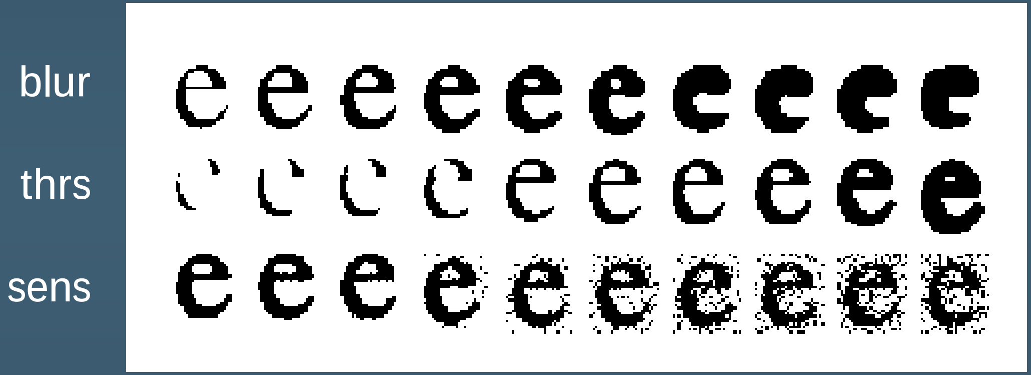
Versatility Goals

- Try to guarantee high accuracy across any given set of:
  - symbols
  - typefaces
  - type sizes
  - image degradations
  - layout geometries
  - languages & writing systems
- First step: a 100-typeface, full-ASCII classifier
- Automate everything possible:
  - emphasize machine learning (avoid hand-crafted rules)
  - identify good features semi-automatically
  - train classifiers fully automatically
  - model image quality, then generate synthetic training data

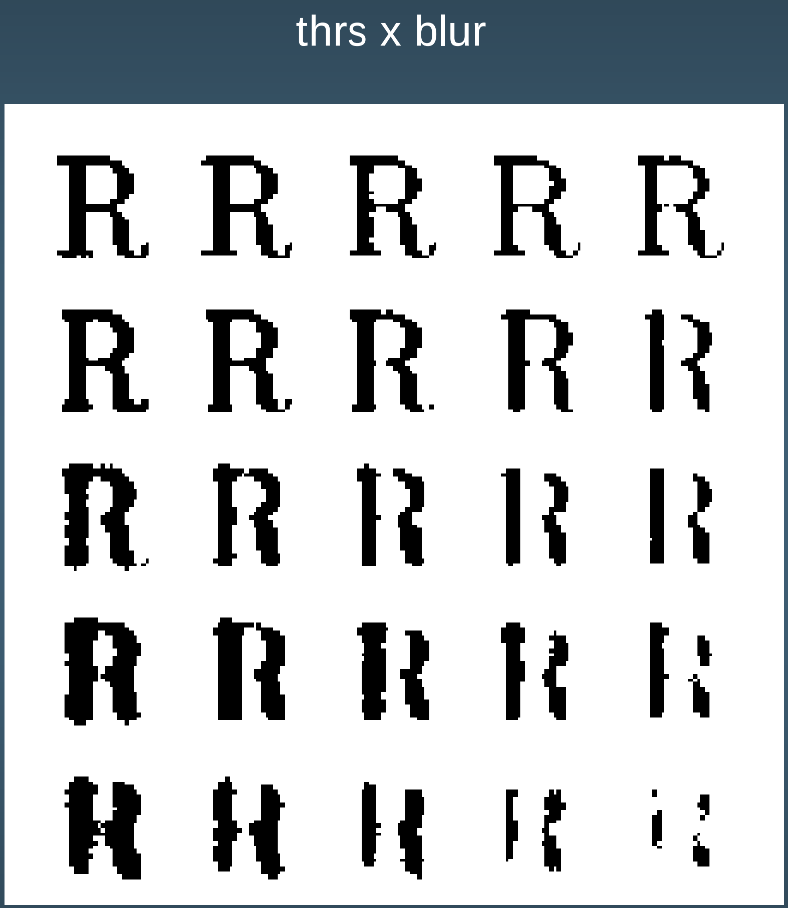
Pavlidis, Baird, Kahan, & Fossey (1985-1992)

Image Quality Modeling

Effects of printing & imaging:



Also, 8 other parameters



Baird & Pavlidis (1985-1992)

Image Quality Models: Fitting to Real Data & Using Safely

- Testing dissimilarity of two sets of images

  a sensitive bootstrap statistic: indirectly infer parameters

  (Kanungo Ph.D., 1996 ff)
- Estimating parameters directly from sample images

  a few character images are sufficient

  (Barney Smith Ph.D., 1998 ff)
- Ensuring the safety of training on synthetic data

  by interpolation in generator parameter space

  (Nonnemaker Ph.D., 2008)

Many open questions remain (several Ph.D.s' worth?)

Model-Driven Architecture

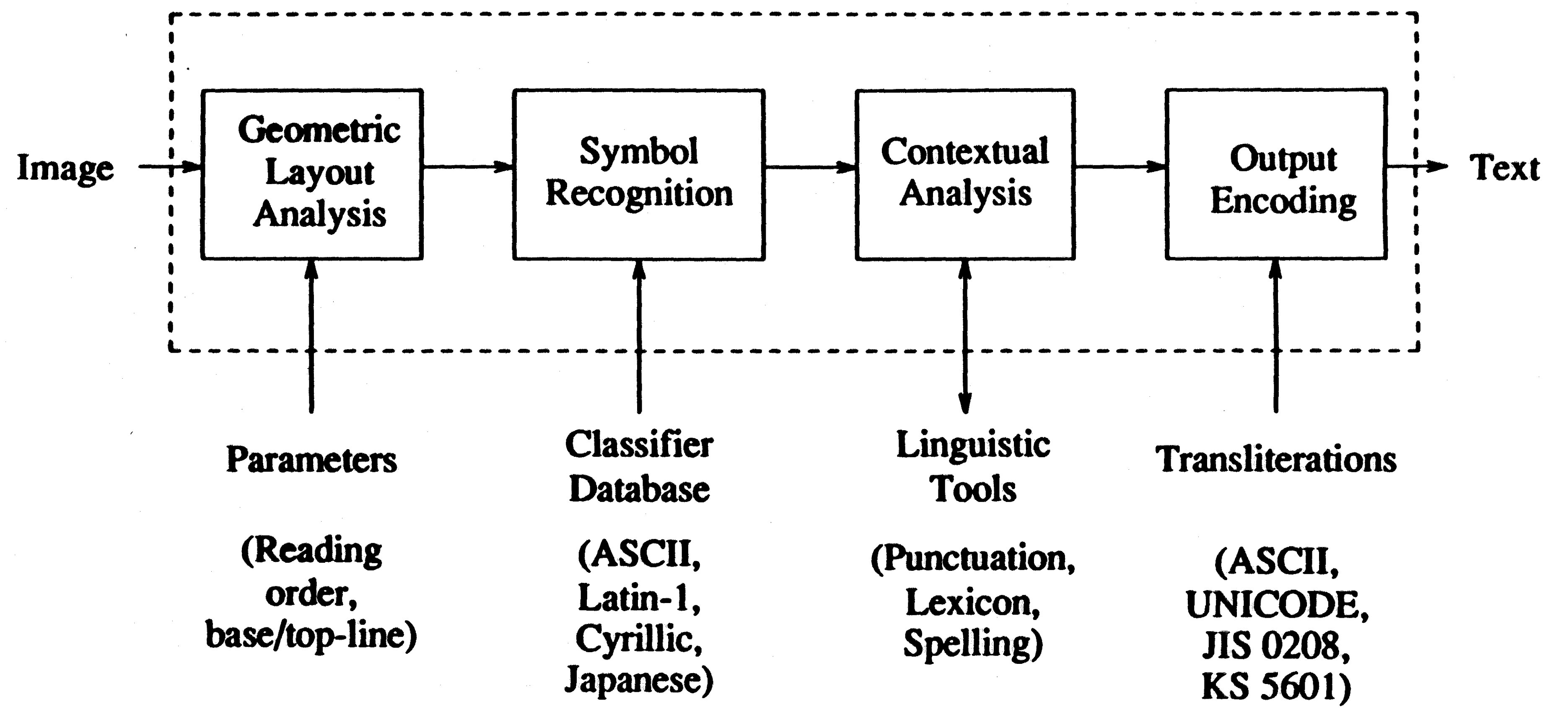


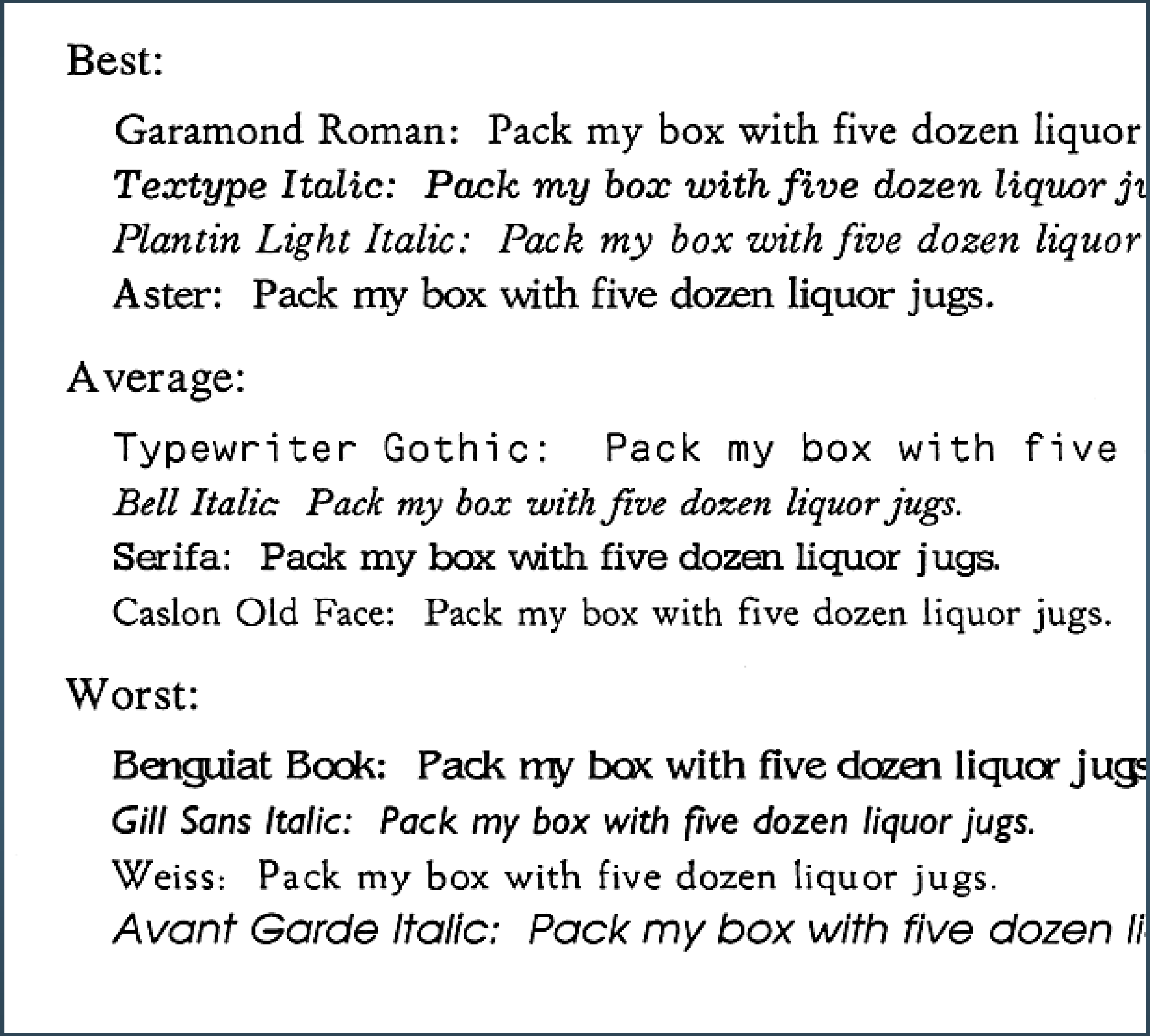
Image
Geometric Layout Analysis
Symbol Recognition
Contextual Analysis
Output Encoding
Text
Parameters
(Reading order, base/top-line)
Classifier Database
(ASCII, Latin-1, Cyrillic, Japanese)
Linguistic Tools
(Punctuation, Lexicon, Spelling)
Transliterations
(ASCII, UNICODE, JIS 0208, KS 5601)

Several application-specific models of knowledge:

most can be acquired (trained, hand-crafted, bought) off-line.

Baird & Ittner (1988-1994)

Accuracy was High, but <u>Not Uniform</u>



Best:

Garamond Roman: Pack my box with five dozen liquor
Textype Italic: Pack my box with five dozen liquor ju
Plantin Light Italic: Pack my box with five dozen liquor
Aster: Pack my box with five dozen liquor jugs.

Average:

Typewriter Gothic: Pack my box with five
Bell Italic: Pack my box with five dozen liquor jugs.
Serifa: Pack my box with five dozen liquor jugs.
Caslon Old Face: Pack my box with five dozen liquor jugs.

Worst:

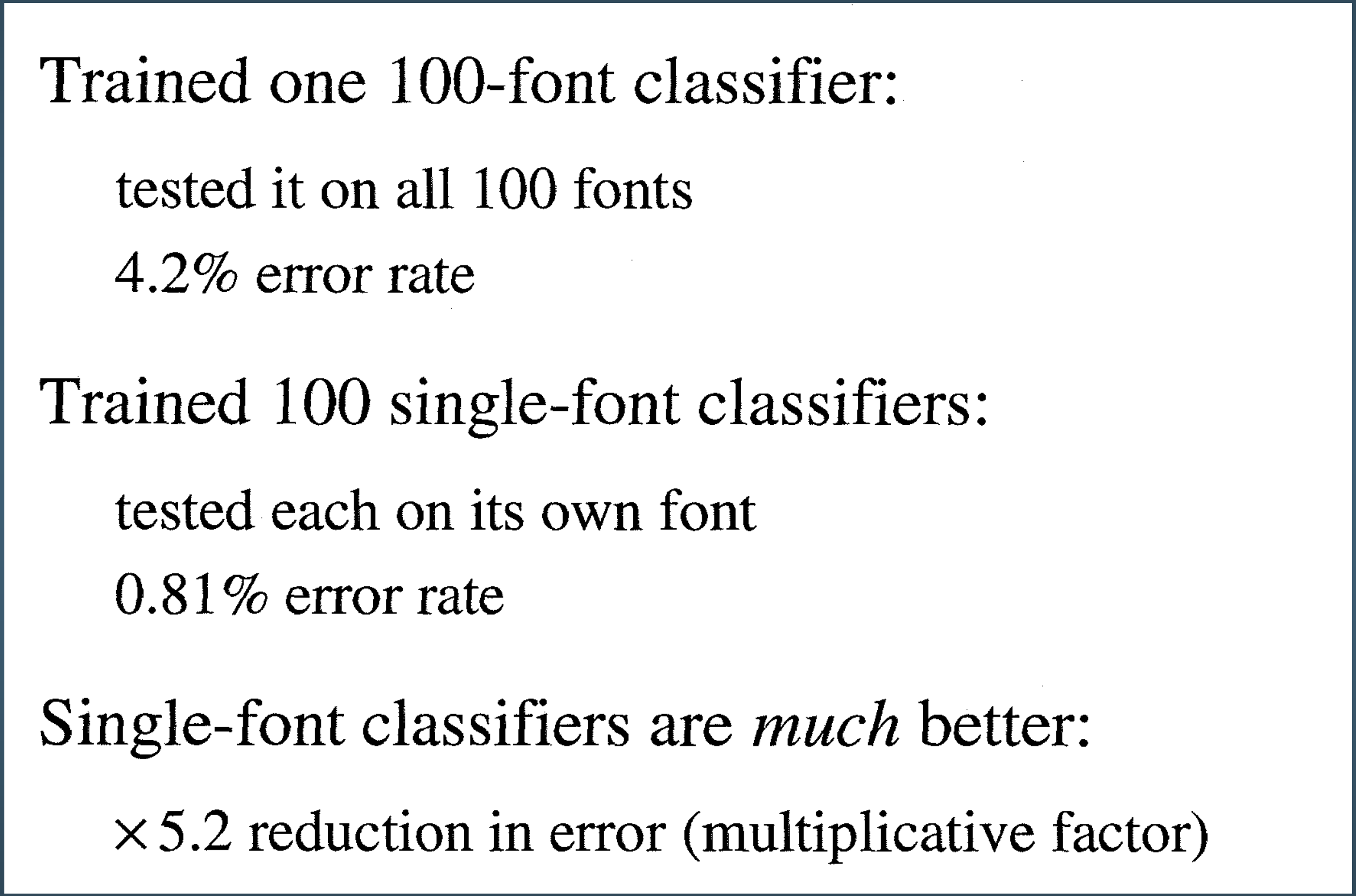
Benguiat Book: Pack my box with five dozen liquor jugs
Gill Sans Italic: Pack my box with five dozen liquor jugs.
Weiss: Pack my box with five dozen liquor jugs.
Avant Garde Italic: Pack my box with five dozen li

> 99.97%

~ 99.7%

< 99%

Single-font Classifiers are far more accurate on their font



Trained one 100-font classifier:

tested it on all 100 fonts

4.2% error rate

Trained 100 single-font classifiers:

tested each on its own font

0.81% error rate

Single-font classifiers are *much* better:

× 5.2 reduction in error (multiplicative factor)

Generic:

versatile, but…

not a best fit to many fonts

Specific:

brittle, but…

the best fit to some font

If can recognize the input font, can benefit a lot

(but, hard to do)

“Strong” versus “Weak” Models

I sometimes find it helpful to distinguish between them.

- **Strong** models:
  - application-specific,
  - a close fit to the input,
  - often detailed and formal.

Expensive to acquire

More accurate

(on the right input)

We often feel forced to choose one over the other

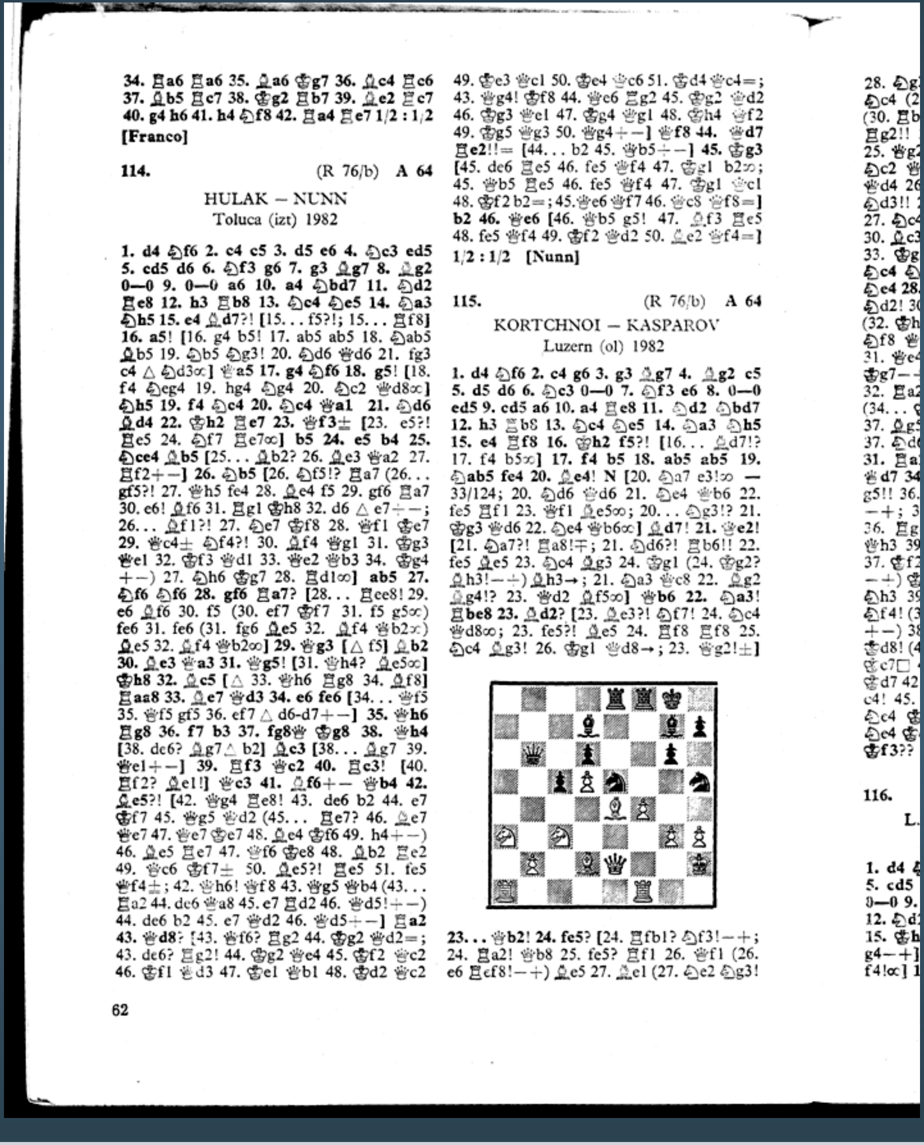
- **Weak** models:
  - generic,
  - applicable to other inputs too,
  - often informal or imprecise.

Cheap to acquire

Less accurate

(on average)

Strong Models: *e.g.* *Sahovsky Informator*



34. ♖a6 ♖a6 35. ♗a6 ♔g7 36. ♗c4 ♖c6 37. ♗b5 ♖c7 38. ♔g2 ♖b7 39. ♗e2 ♖c7 40. g4 h6 41. h4 ♘f8 42. ♖a4 ♖e7 1/2 : 1/2 [Franco]

114. (R 76/b) A 64

HULAK – NUNN
Toluca (izt) 1982

1. d4 ♘f6 2. c4 c5 3. d5 e6 4. ♘c3 ed5 5. cd5 d6 6. ♘f3 g6 7. g3 ♗g7 8. ♗g2 0—0 9. 0—0 a6 10. a4 ♘bd7 11. ♘d2 ♖e8 12. h3 ♖b8 13. ♘c4 ♘e5 14. ♘a3 ♘h5 15. e4 ♗d7?! [15. . . f5?!; 15. . . ♖f8] 16. a5! [16. g4 b5! 17. ab5 ab5 18. ♘ab5 ♗b5 19. ♘b5 ♘g3! 20. ♘d6 ♕d6 21. fg3 c4 △ ♘d3∞] ♕a5 17. g4 ♘f6 18. g5! [18. f4 ♘eg4 19. hg4 ♘g4 20. ♘c2 ♕d8∞] ♘h5 19. f4 ♘c4 20. ♘c4 ♕a1 21. ♘d6 ♗d4 22. ♔h2 ♖e7 23. ♕f3± [23. e5?! ♖e5 24. ♘f7 ♖e7∞] b5 24. e5 b4 25. ♘ce4 ♗b5 [25. . . ♗b2? 26. ♗e3 ♕a2 27. ♖f2+−] 26. ♘b5 [26. ♘f5!? ♖a7 (26. . . gf5?! 27. ♕h5 fe4 28. ♗e4 f5 29. gf6 ♖a7 30. e6! ♗f6 31. ♖g1 ♔h8 32. d6 △ e7+−; 26. . . ♗f1?! 27. ♘e7 ♔f8 28. ♕f1 ♔e7 29. ♕c4± ♘f4?! 30. ♗f4 ♕g1 31. ♔g3 ♕e1 32. ♔f3 ♕d1 33. ♕e2 ♕b3 34. ♔g4 +−) 27. ♘h6 ♔g7 28. ♖d1∞] ab5 27. ♘f6 ♘f6 28. gf6 ♖a7? [28. . . ♖ee8! 29. e6 ♗f6 30. f5 (30. ef7 ♔f7 31. f5 g5∞) fe6 31. fe6 (31. fg6 ♗e5 32. ♗f4 ♕b2∞) ♗e5 32. ♗f4 ♕b2∞] 29. ♕g3 [△ f5] ♗b2 30. ♗e3 ♕a3 31. ♕g5! [31. ♕h4? ♗e5∞] ♔h8 32. ♗c5 [△ 33. ♕h6 ♖g8 34. ♗f8] ♖aa8 33. ♗e7 ♕d3 34. e6 fe6 [34. . . ♕f5 35. ♕f5 gf5 36. ef7 △ d6-d7+−] 35. ♕h6 ♖g8 36. f7 b3 37. fg8♕ ♔g8 38. ♕h4 [38. de6? ♗g7 △ b2] ♗c3 [38. . . ♗g7 39. ♕e1+−] 39. ♖f3 ♕c2 40. ♖c3! [40. ♖f2? ♗e1!] ♕c3 41. ♗f6+− ♕b4 42. ♗e5?! [42. ♕g4 ♖e8! 43. de6 b2 44. e7 ♔f7 45. ♕g5 ♕d2 (45. . . ♖e7? 46. ♗e7 ♕e7 47. ♕e7 ♔e7 48. ♗e4 ♔f6 49. h4+−) 46. ♗e5 ♖e7 47. ♕f6 ♔e8 48. ♗b2 ♖e2 49. ♕c6 ♔f7± 50. ♗e5?! ♖e5 51. fe5 ♕f4±; 42. ♕h6! ♕f8 43. ♕g5 ♕b4 (43. . . ♖a2 44. de6 ♕a8 45. e7 ♖d2 46. ♕d5!+−) 44. de6 b2 45. e7 ♕d2 46. ♕d5+−] ♖a2 43. ♕d8? [43. ♕f6? ♖g2 44. ♔g2 ♕d2=; 43. de6? ♖g2! 44. ♔g2 ♕e4 45. ♔f2 ♕c2 46. ♔f1 ♕d3 47. ♔e1 ♕b1 48. ♔d2 ♕c2 49. ♔e3 ♕c1 50. ♔e4 ♕c6 51. ♔d4 ♕c4=; 43. ♕g4! ♔f8 44. ♕e6 ♖g2 45. ♔g2 ♕d2 46. ♔g3 ♕e1 47. ♔g4 ♕g1 48. ♔h4 ♕f2 49. ♔g5 ♕g3 50. ♕g4+−] ♕f8 44. ♕d7 ♖e2!!= [44. . . b2 45. ♕b5+−] 45. ♔g3 [45. de6 ♖e5 46. fe5 ♕f4 47. ♔g1 b2∞; 45. ♕b5 ♖e5 46. fe5 ♕f4 47. ♔g1 ♕c1 48. ♔f2 b2=; 45. ♕e6 ♕f7 46. ♕c8 ♕f8=] b2 46. ♕e6 [46. ♕b5 g5! 47. ♗f3 ♖e5 48. fe5 ♕f4 49. ♔f2 ♕d2 50. ♗e2 ♕f4=] 1/2 : 1/2 [Nunn]

62

115. (R 76/b) A 64

KORTCHNOI – KASPAROV
Luzern (ol) 1982

1. d4 ♘f6 2. c4 g6 3. g3 ♗g7 4. ♗g2 c5 5. d5 d6 6. ♘c3 0—0 7. ♘f3 e6 8. 0—0 ed5 9. cd5 a6 10. a4 ♖e8 11. ♘d2 ♘bd7 12. h3 ♖b8 13. ♘c4 ♘e5 14. ♘a3 ♘h5 15. e4 ♖f8 16. ♔h2 f5?! [16. . . ♗d7!? 17. f4 b5∞] 17. f4 b5 18. ab5 ab5 19. ♘ab5 fe4 20. ♗e4! N [20. ♘a7 e3!∞ — 33/124; 20. ♘d6 ♕d6 21. ♘e4 ♕b6 22. fe5 ♖f1 23. ♕f1 ♗e5∞; 20. . . ♘g3!? 21. ♔g3 ♕d6 22. ♘e4 ♕b6∞] ♗d7! 21. ♕e2! [21. ♘a7?! ♖a8!∓; 21. ♘d6?! ♖b6!! 22. fe5 ♗e5 23. ♘c4 ♗g3 24. ♔g1 (24. ♔g2? ♗h3!−+) ♗h3→; 21. ♘a3 ♕c8 22. ♗g2 ♗g4!? 23. ♕d2 ♗f5∞] ♕b6 22. ♘a3! ♖be8 23. ♗d2? [23. ♗e3?! ♘f7! 24. ♘c4 ♕d8∞; 23. fe5?! ♗e5 24. ♖f8 ♖f8 25. ♘c4 ♗g3! 26. ♔g1 ♕d8→; 23. ♕g2!±]

23. . . ♕b2! 24. fe5? [24. ♖fb1? ♘f3!−+; 24. ♖a2! ♕b8 25. fe5? ♖f1 26. ♕f1 (26. e6 ♖ef8!−+) ♗e5 27. ♗e1 (27. ♘e2 ♘g3!

28. ♘g
♘c4 (2
(30. ♖b
♖g2!!
25. ♕g2
♘c2 ♕
♕d4 26
♘d3!!
27. ♘c4
30. ♗c3
33. ♔g
♘c4 ♘
♘e4 28.
♘d2! 30
(32. ♔h
♘f8 ♕
31. ♕e4
♔g7−
32. ♖a2
(34. . .
37. ♗g5
37. ♘d6
31. ♖a
♕d7 34
g5!! 36.
−+; 3
36. ♖g
♕h3 39
37. ♔f2
−+) ♔
♘h3 39
♘f4! (3
+−) 38
♔d8! (4
♔c7□
♔d7 42
c4! 45.
♘c4 ♔
♘e4 ♔
♔f3??

116.

L.

1. d4 ♘
5. cd5
0—0 9.
12. ♘d
15. ♔h
g4−+]
f4!∞] 1

Chess encyclopaedia in 20+ volumes

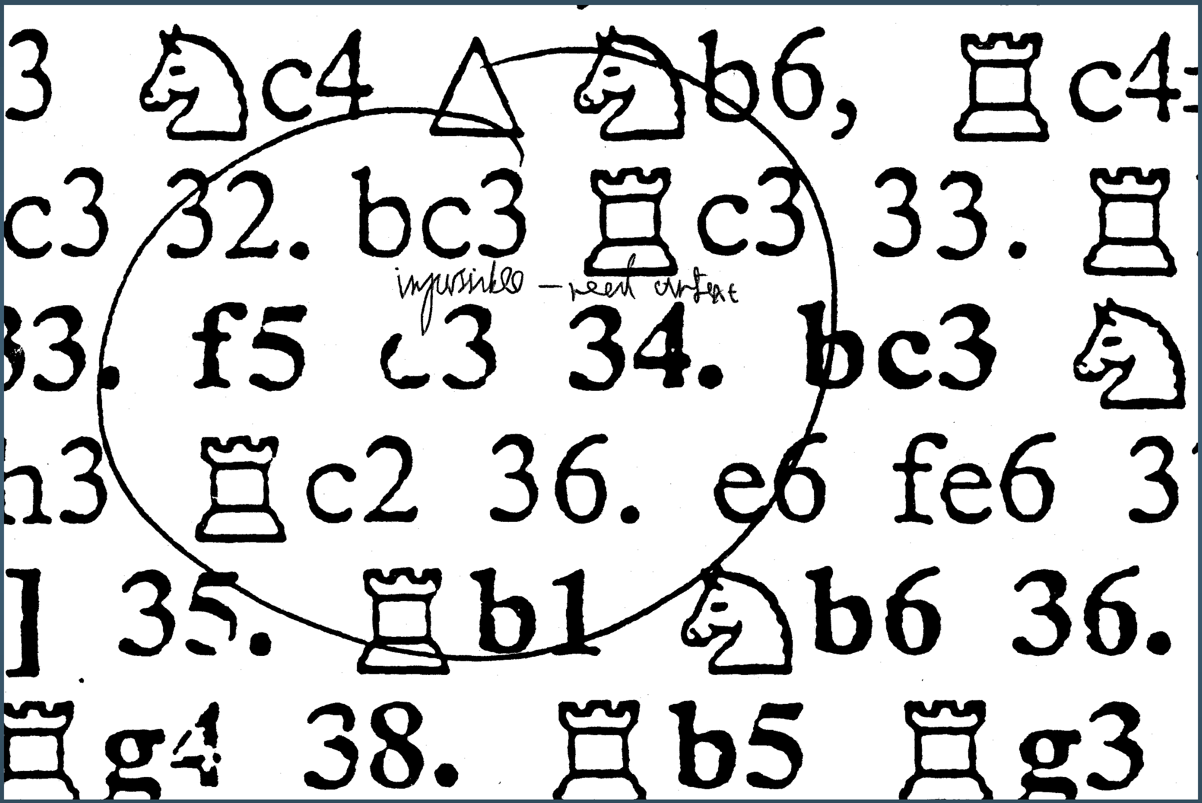
Games of theoretical interest

Ken Thompson wanted to teach these games to *Belle*, his chess machine

Ken coded-up syntax & semantic models

Baird & Thompson (1990)

Challenging Print Quality



3 ♘c4 △ ♘b6, ♖c4

c3 32. bc3 ♖c3 33. ♖

impossible – need context

3. f5 c3 34. bc3 ♘

n3 ♖c2 36. e6 fe6 3

] 35. ♖b1 ♘b6 36.

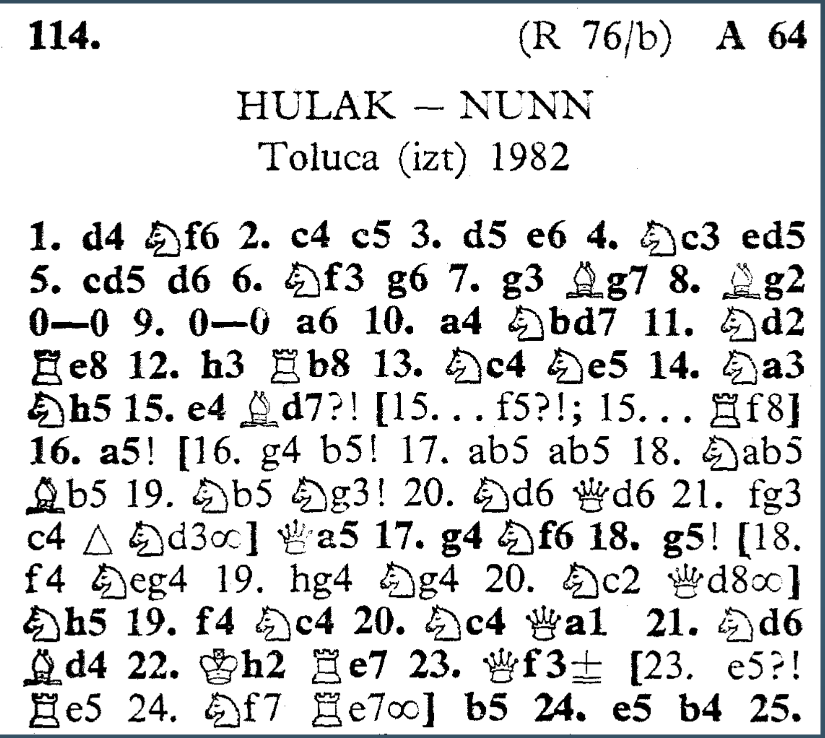
♖g4 38. ♖b5 ♖g3

I trained on this special chess font

Near-perfect OCR is impossible on such poor quality

But, unless *entire games* are correctly read, then it’s not worth doing…!

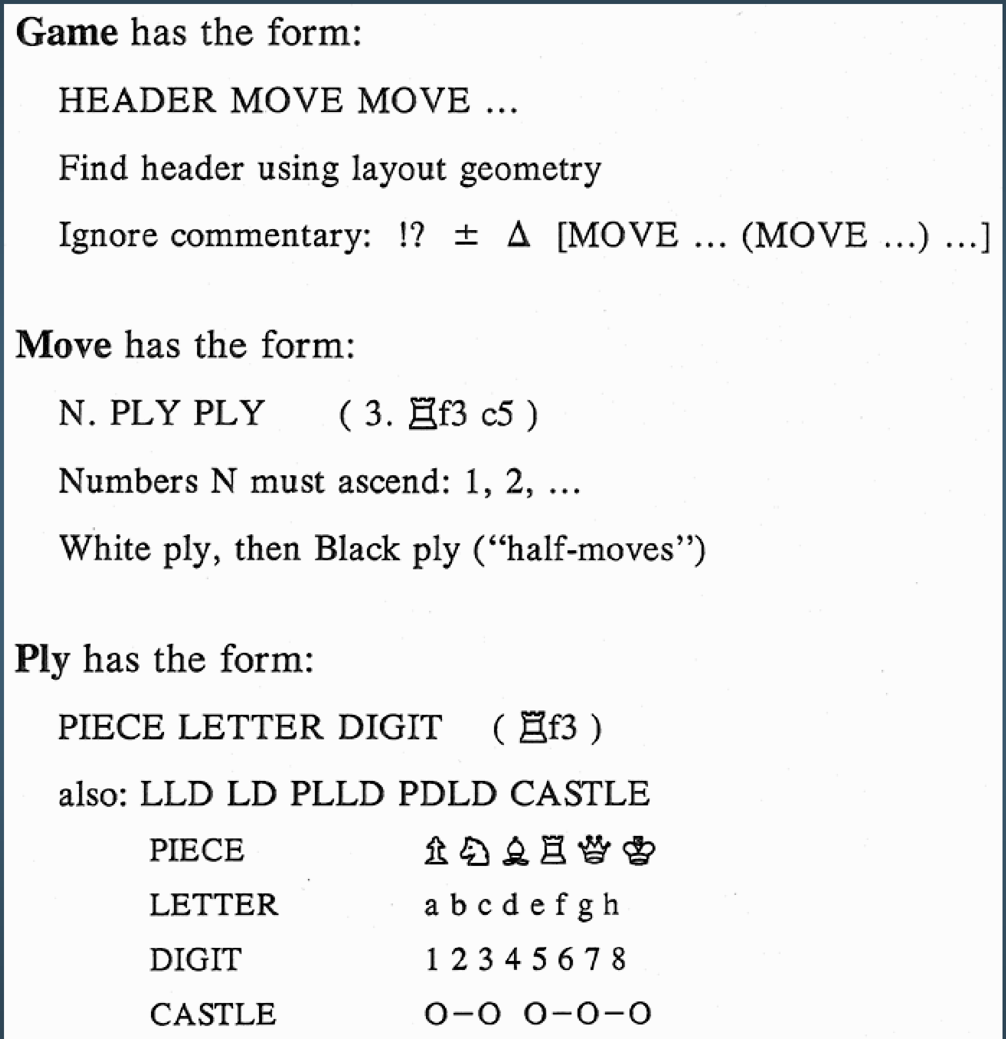
Informator Syntax is Computable



114. (R 76/b) **A 64**

HULAK – NUNN
Toluca (izt) 1982

1. d4 ♘f6 2. c4 c5 3. d5 e6 4. ♘c3 ed5 5. cd5 d6 6. ♘f3 g6 7. g3 ♗g7 8. ♗g2 0—0 9. 0—0 a6 10. a4 ♘bd7 11. ♘d2 ♖e8 12. h3 ♖b8 13. ♘c4 ♘e5 14. ♘a3 ♘h5 15. e4 ♗d7?! [15. . . f5?!; 15. . . ♖f8] **16. a5!** [16. g4 b5! 17. ab5 ab5 18. ♘ab5 ♗b5 19. ♘b5 ♘g3! 20. ♘d6 ♕d6 21. fg3 c4 △ ♘d3∞] **♕a5 17. g4 ♘f6 18. g5!** [18. f4 ♘eg4 19. hg4 ♘g4 20. ♘c2 ♕d8∞] **♘h5 19. f4 ♘c4 20. ♘c4 ♕a1 21. ♘d6 ♗d4 22. ♔h2 ♖e7 23. ♕f3±** [23. e5?! ♖e5 24. ♘f7 ♖e7∞] **b5 24. e5 b4 25.**



Game has the form:

HEADER MOVE MOVE …

Find header using layout geometry

Ignore commentary: !? ± Δ [MOVE … (MOVE …) …]

Move has the form:

N. PLY PLY (3. ♖f3 c5)

Numbers N must ascend: 1, 2, …

White ply, then Black ply (“half-moves”)

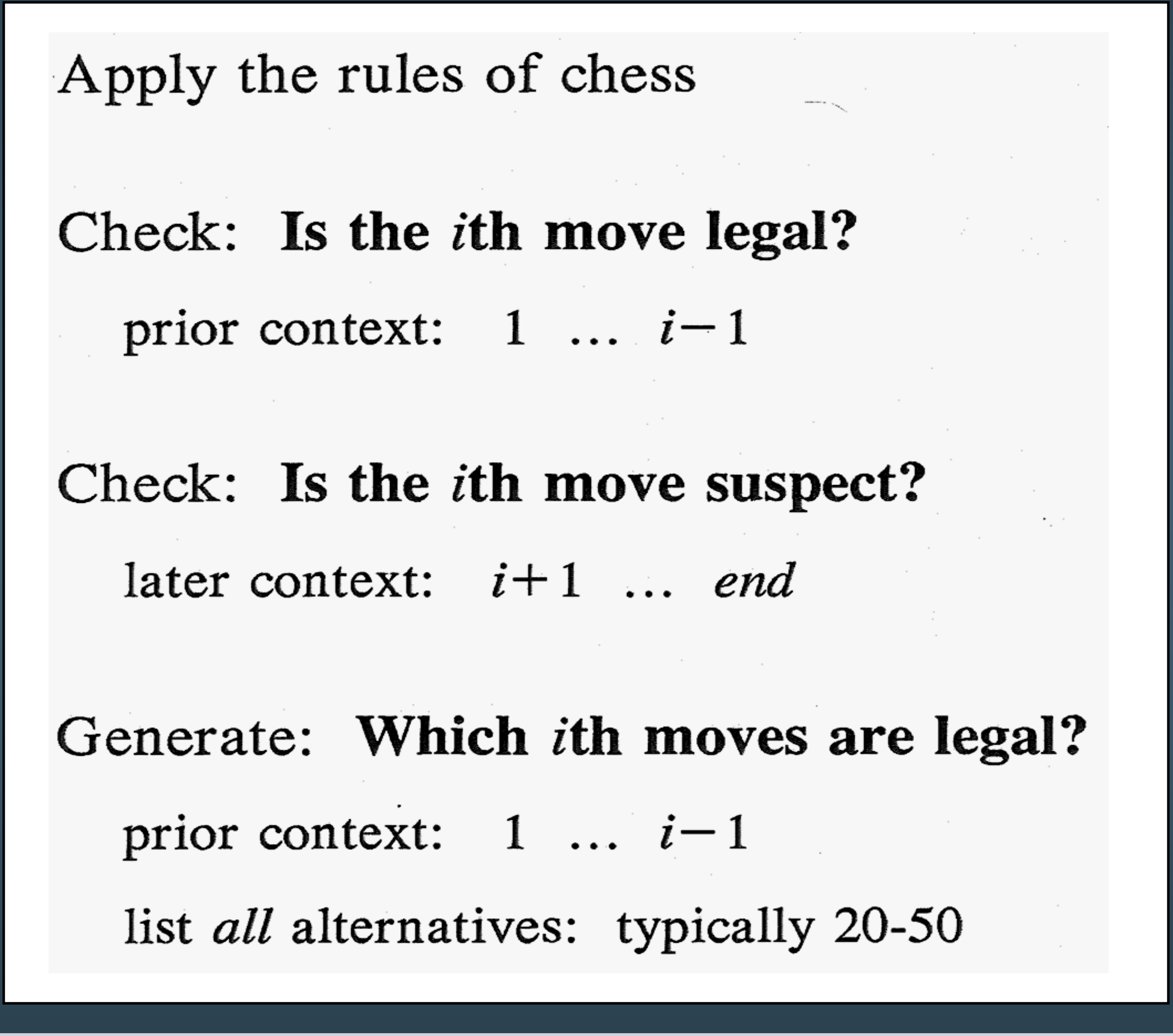
Ply has the form:

PIECE LETTER DIGIT (♖f3)

also: LLD LD PLLD PDLD CASTLE

PIECE	♙ ♘ ♗ ♖ ♕ ♔
LETTER	a b c d e f g h
DIGIT	1 2 3 4 5 6 7 8
CASTLE	O–O O–O–O

Chess Semantics is Computable



Apply the rules of chess

Check: **Is the *i*th move legal?**

prior context: 1 … $i-1$

Check: **Is the *i*th move suspect?**

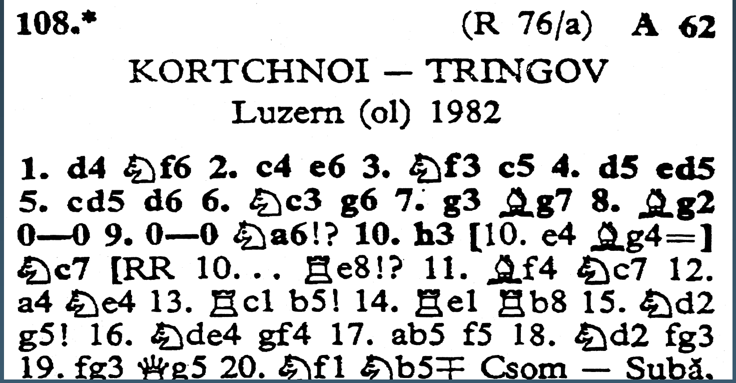
later context: $i+1$ … *end*

Generate: **Which *i*th moves are legal?**

prior context: 1 … $i-1$

list *all* alternatives: typically 20-50

Fully Automatic Extraction of Games

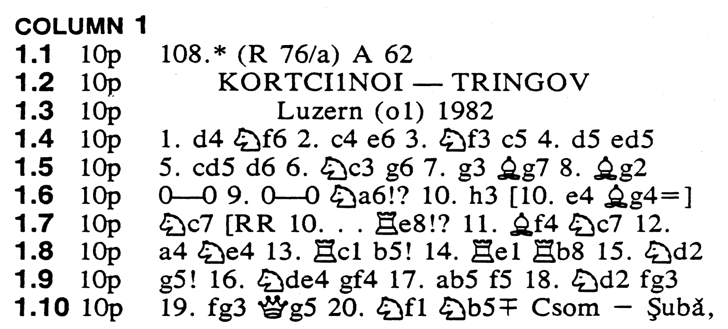


108.* (R 76/a) A 62

KORTCHNOI – TRINGOV
Luzern (ol) 1982

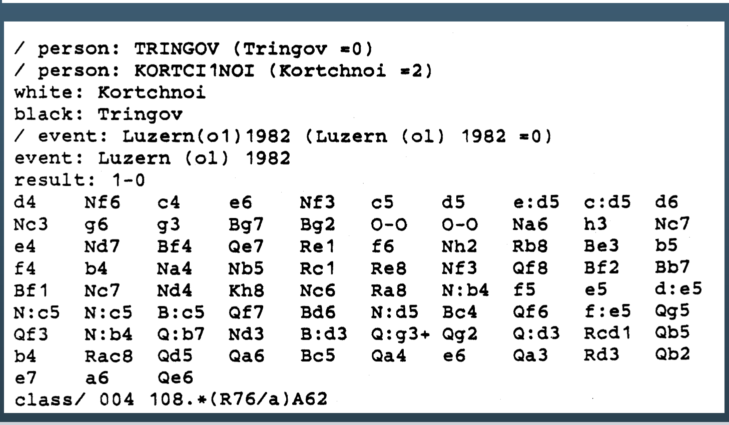
1. d4 ♘f6 2. c4 e6 3. ♘f3 c5 4. d5 ed5
5. cd5 d6 6. ♘c3 g6 7. g3 ♗g7 8. ♗g2
0—0 9. 0—0 ♘a6!? 10. h3 [10. e4 ♗g4=]
♘c7 [RR 10. . . ♖e8!? 11. ♗f4 ♘c7 12.
a4 ♘e4 13. ♖c1 b5! 14. ♖e1 ♖b8 15. ♘d2
g5! 16. ♘de4 gf4 17. ab5 f5 18. ♘d2 fg3
19. fg3 ♕g5 20. ♘f1 ♘b5∓ Csom — Subă,

Image of page



```
COLUMN 1
1.1  10p   108.* (R 76/a) A 62
1.2  10p          KORTCI1NOI — TRINGOV
1.3  10p               Luzern (o1) 1982
1.4  10p   1. d4 ♘f6 2. c4 e6 3. ♘f3 c5 4. d5 ed5
1.5  10p   5. cd5 d6 6. ♘c3 g6 7. g3 ♗g7 8. ♗g2
1.6  10p   0—0 9. 0—0 ♘a6!? 10. h3 [10. e4 ♗g4=]
1.7  10p   ♘c7 [RR 10. . . ♖e8!? 11. ♗f4 ♘c7 12.
1.8  10p   a4 ♘e4 13. ♖c1 b5! 14. ♖e1 ♖b8 15. ♘d2
1.9  10p   g5! 16. ♘de4 gf4 17. ab5 f5 18. ♘d2 fg3
1.10 10p   19. fg3 ♕g5 20. ♘f1 ♘b5∓ Csom – Şubă,
```

‘Galley-proof’ format output from the OCR

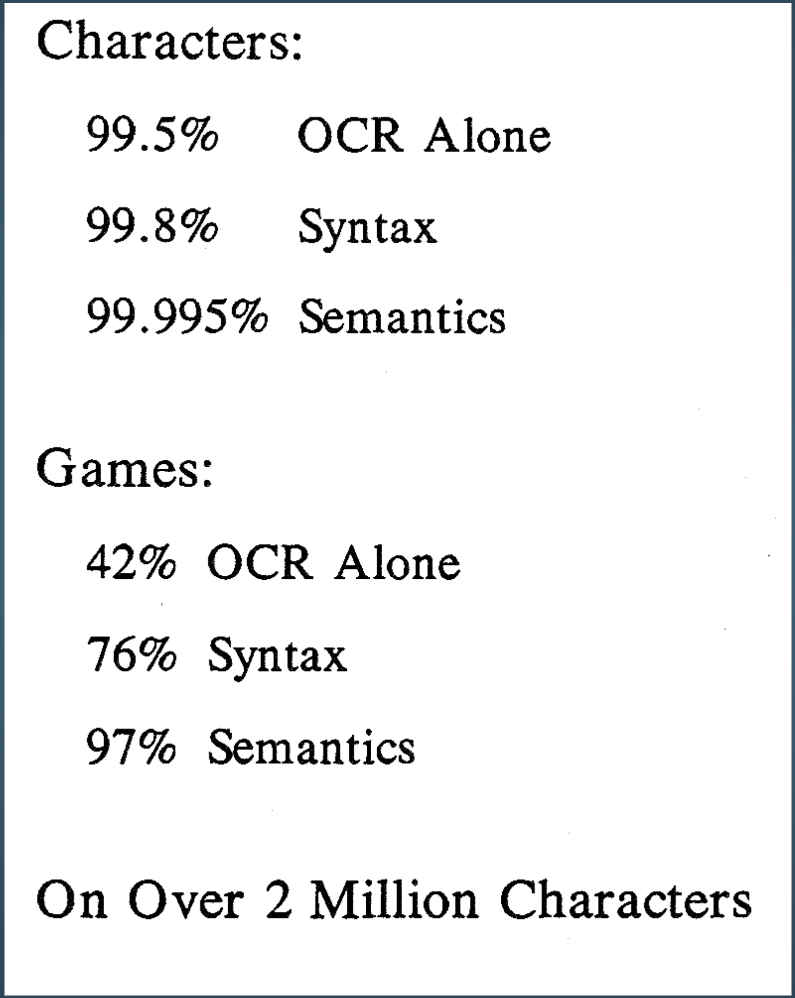


```
/ person: TRINGOV (Tringov =0)
/ person: KORTCI1NOI (Kortchnoi =2)
white: Kortchnoi
black: Tringov
/ event: Luzern(o1)1982 (Luzern (ol) 1982 =0)
event: Luzern (ol) 1982
result: 1-0
d4     Nf6    c4     e6     Nf3    c5     d5     e:d5   c:d5   d6
Nc3    g6     g3     Bg7    Bg2    O-O    O-O    Na6    h3     Nc7
e4     Nd7    Bf4    Qe7    Re1    f6     Nh2    Rb8    Be3    b5
f4     b4     Na4    Nb5    Rc1    Re8    Nf3    Qf8    Bf2    Bb7
Bf1    Nc7    Nd4    Kh8    Nc6    Ra8    N:b4   f5     e5     d:e5
N:c5   N:c5   B:c5   Qf7    Bd6    N:d5   Bc4    Qf6    f:e5   Qg5
Qf3    N:b4   Q:b7   Nd3    B:d3   Q:g3+  Qg2    Q:d3   Rcd1   Qb5
b4     Rac8   Qd5    Qa6    Bc5    Qa4    e6     Qa3    Rd3    Qb2
e7     a6     Qe6
class/ 004 108.*(R76/a)A62
```

Database of games, moves

this game = 83 half-moves

Semantic Model Astonishingly Helpful



Characters:

99.5%	OCR Alone
99.8%	Syntax
99.995%	Semantics

Games:

42%	OCR Alone
76%	Syntax
97%	Semantics

On Over 2 Million Characters

Syntactic model cuts errors in half

Semantics cuts errors by another <u>factor of 40!</u>

99.5% OCR accuracy implies that
game accuracy is only 40%

After semantic analysis,
almost all games are completely correct

Lessons from Reading Chess

An extreme illustration of strong modeling:

- Syntax & semantics fitted precisely to these books
- Remarkably high performance: 50 errors per million chars

But: wasn't this a unique event?

- Can we model syntax and semantics of other books?
- Will our users be domain experts w/ software skills?

Note the size of the context is many dozens of moves, all operated on by the semantic analysis.

- Perhaps we can operate on long passages in other ways....
- Would that help…?   (Open question, for years.)

Beyond Versatility: George Nagy's Adapting Recognizers

Can a recognition system adapt to its input?

Can weak models "self-correct," and so strengthen themselves fully automatically?

When a 100-font system reads a document in a single font, can it specialize to it *without:*

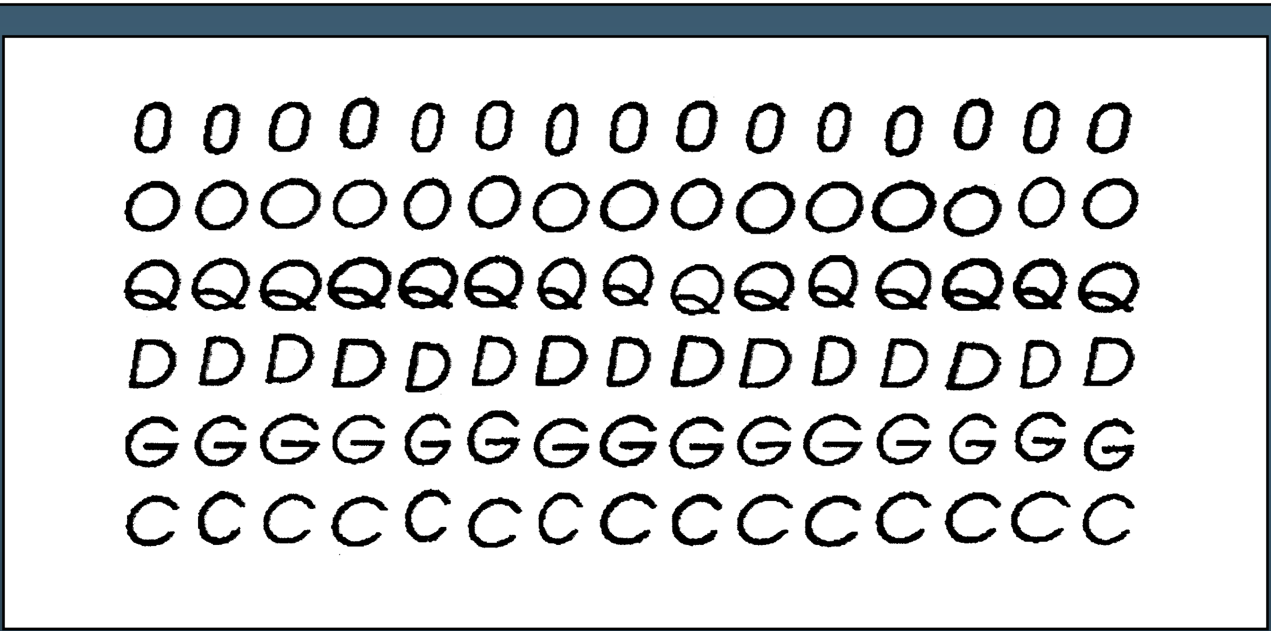
- knowing which font it is,
- recognizing the font, or
- using a library of pre-trained single-font classifiers ?

Nagy, Shelton, & Baird (1966 & 1994)

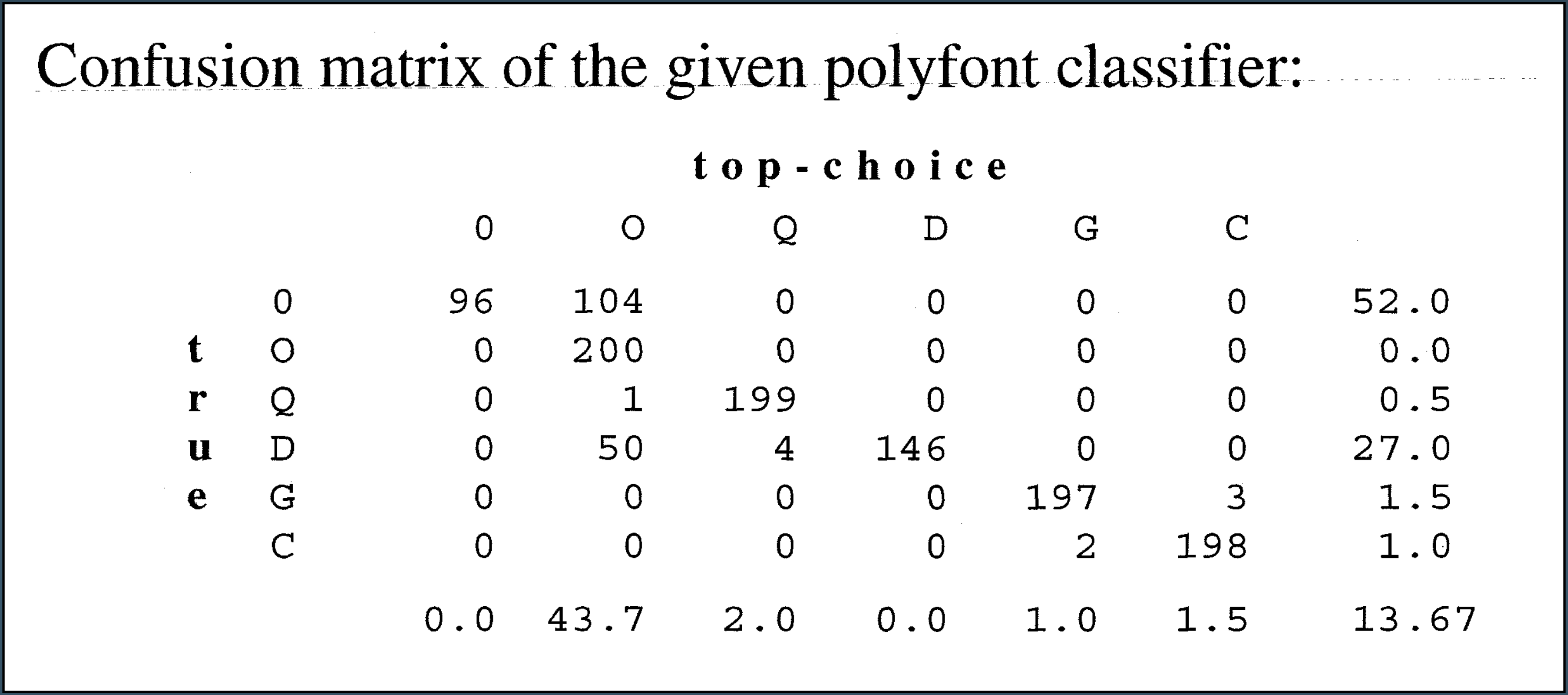
Toy Example: a Single-Font Test

{ *0, O, Q, D, G, C* } in *Avant Garde Book Oblique*

10 point size, 300 pixels/inch resolution, 200 sample images each



The weak (100-font) classifier performs poorly on this….



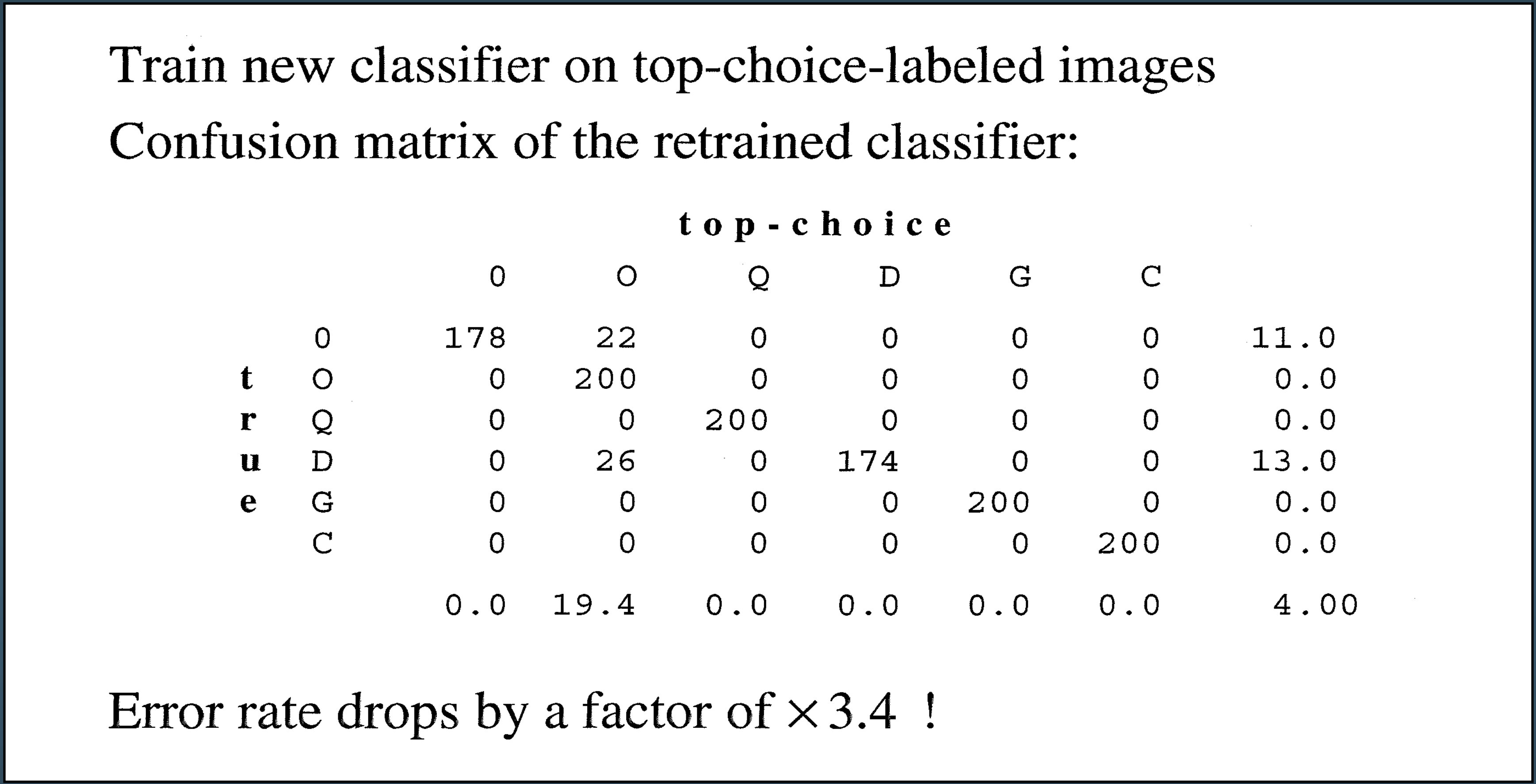
Confusion matrix of the given polyfont classifier:

		top-choice						
		0	O	Q	D	G	C	
	0	96	104	0	0	0	0	52.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	1	199	0	0	0	0.5
u	D	0	50	4	146	0	0	27.0
e	G	0	0	0	0	197	3	1.5
	C	0	0	0	0	2	198	1.0
		0.0	43.7	2.0	0.0	1.0	1.5	13.67

Far from perfect: 14% error rate

Especially: 0/O and D/O confusions

Now, pretending that we believe this classifier, we boldly *retrain*….



Train new classifier on top-choice-labeled images

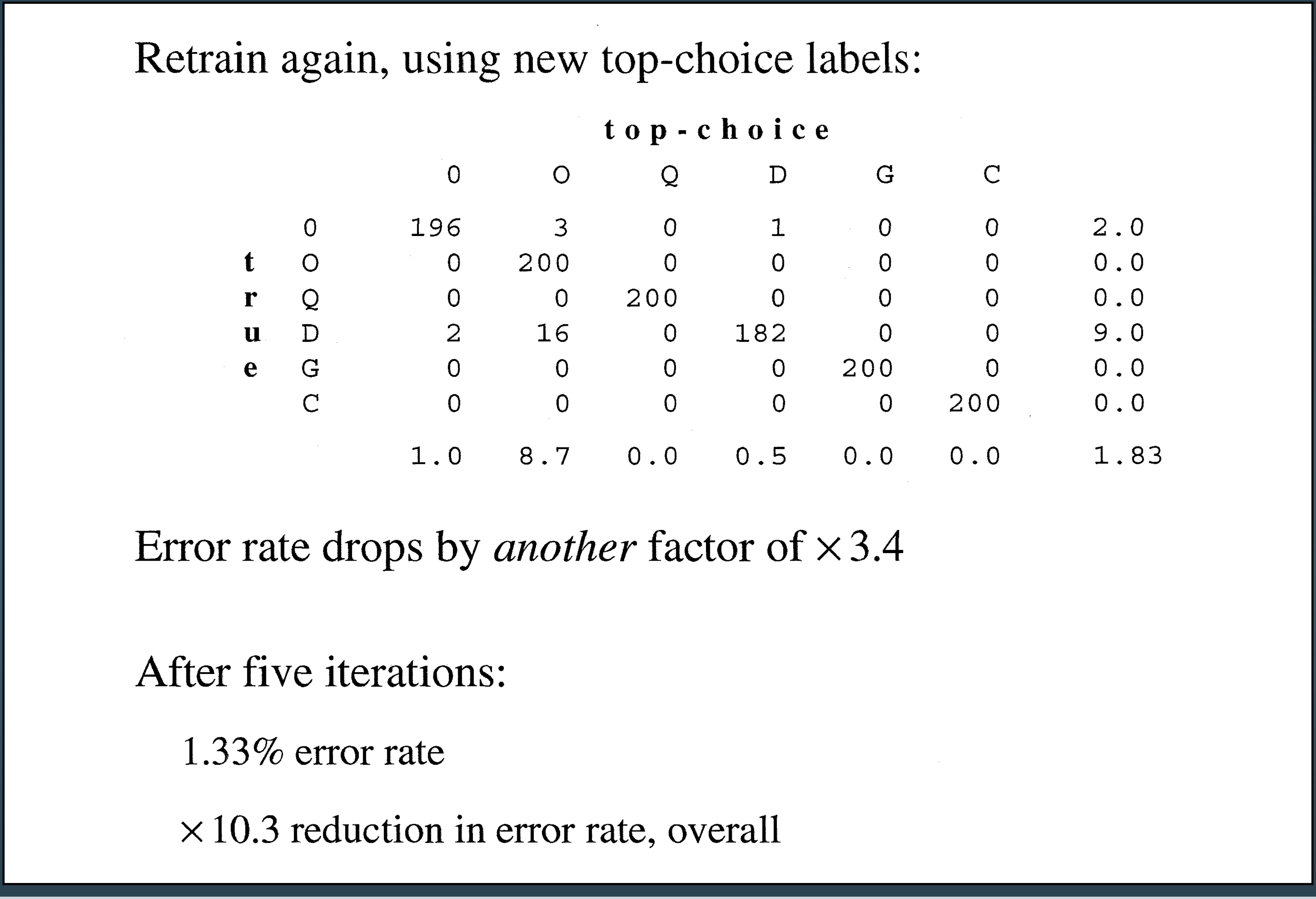
Confusion matrix of the retrained classifier:

		top-choice						
		0	O	Q	D	G	C	
	0	178	22	0	0	0	0	11.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	0	200	0	0	0	0.0
u	D	0	26	0	174	0	0	13.0
e	G	0	0	0	0	200	0	0.0
	C	0	0	0	0	0	200	0.0
		0.0	19.4	0.0	0.0	0.0	0.0	4.00

Error rate drops by a factor of × 3.4 !

The risk of training on (some) mislabeled test data didn’t hurt us!
Lucky!! … or is it reliable?

How lucky can we get…?



Retrain again, using new top-choice labels:

		top-choice						
		0	O	Q	D	G	C	
	0	196	3	0	1	0	0	2.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	0	200	0	0	0	0.0
u	D	2	16	0	182	0	0	9.0
e	G	0	0	0	0	200	0	0.0
	C	0	0	0	0	0	200	0.0
		1.0	8.7	0.0	0.5	0.0	0.0	1.83

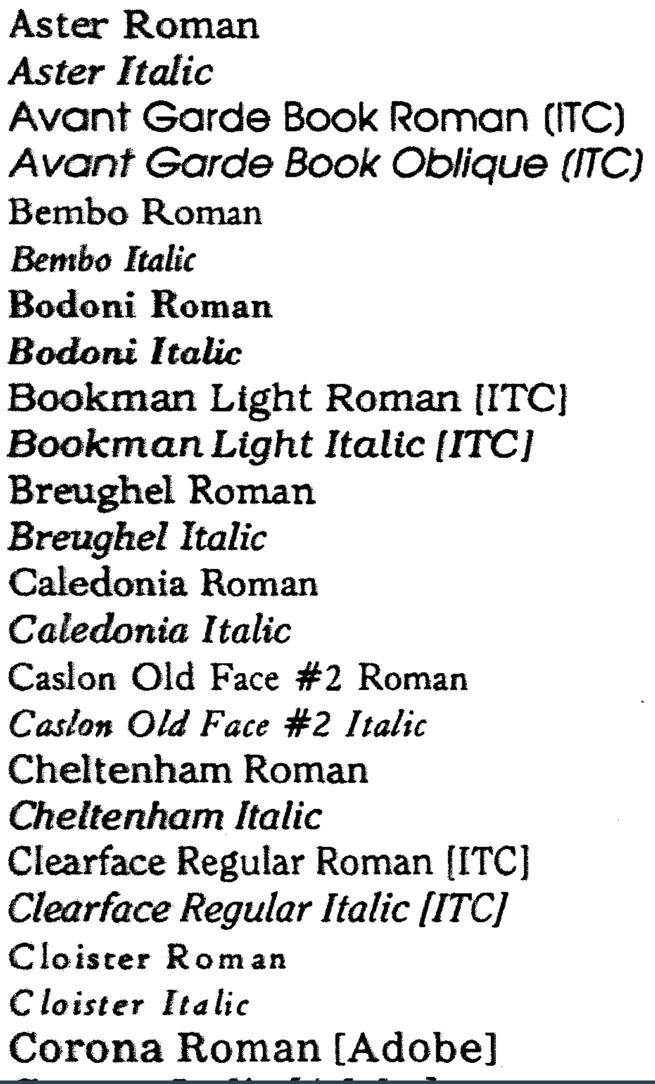
Error rate drops by *another* factor of × 3.4

After five iterations:

1.33% error rate

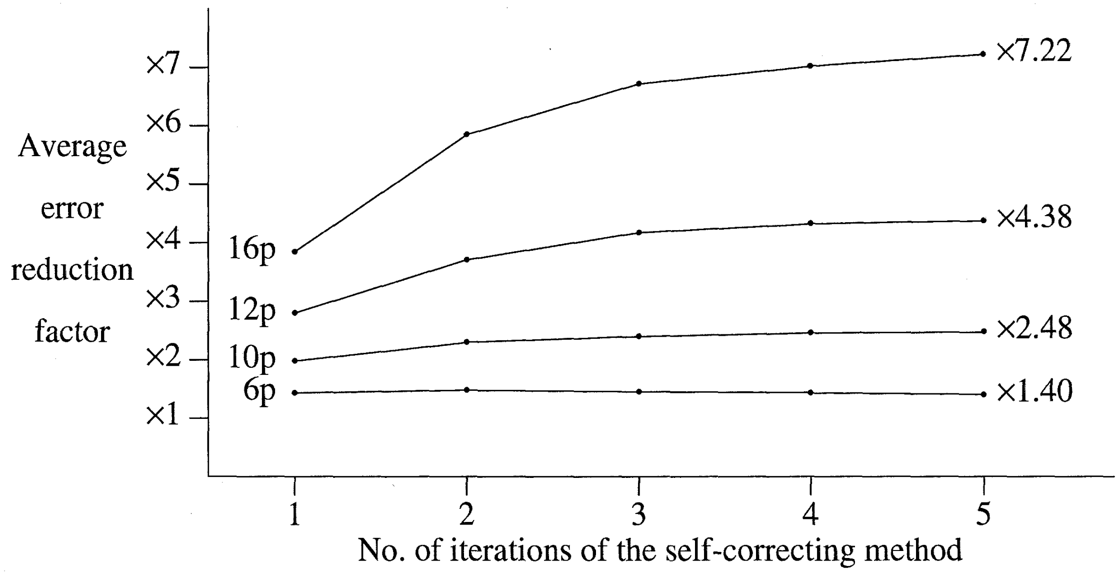
× 10.3 reduction in error rate, overall

In fact this works reliably *(…but why??)*



Aster Roman
Aster Italic
Avant Garde Book Roman (ITC)
Avant Garde Book Oblique (ITC)
Bembo Roman
Bembo Italic
Bodoni Roman
Bodoni Italic
Bookman Light Roman [ITC]
Bookman Light Italic [ITC]
Breughel Roman
Breughel Italic
Caledonia Roman
Caledonia Italic
Caslon Old Face #2 Roman
Caslon Old Face #2 Italic
Cheltenham Roman
Cheltenham Italic
Clearface Regular Roman [ITC]
Clearface Regular Italic [ITC]
Cloister Roman
Cloister Italic
Corona Roman [Adobe]

Average error-reduction factors over 100 fonts

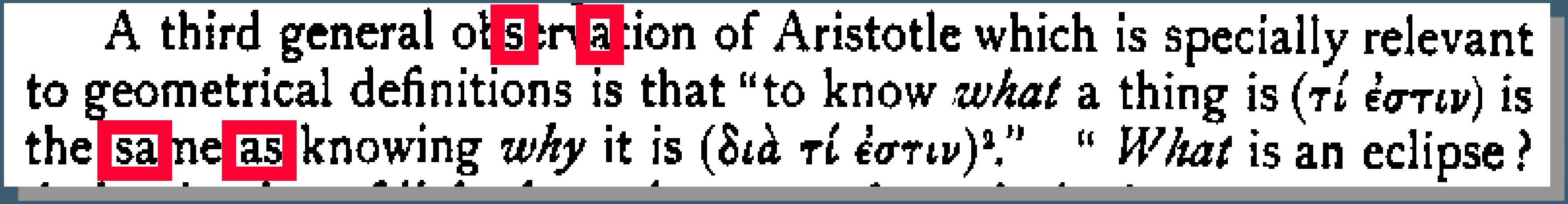


Some improvement at all four sizes

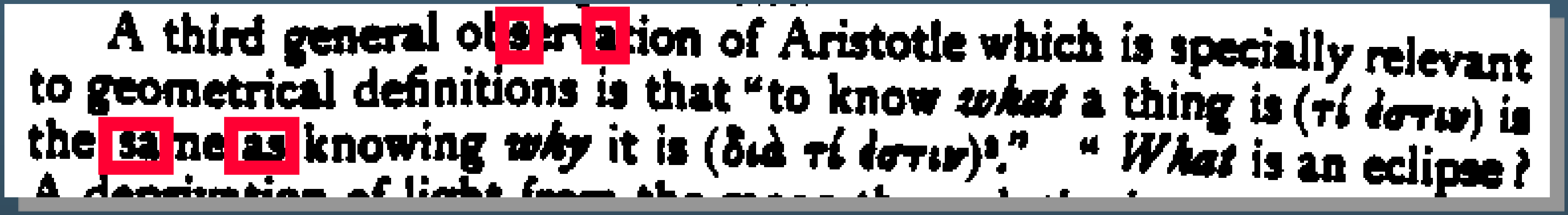
Three iterations are enough

Image Quality also is often Constant throughout a Document

Rather like typefaces, a “style” determined by image degradations due to printing, scanning, etc.



A third general observation of Aristotle which is specially relevant to geometrical definitions is that “to know *what* a thing is (τί ἐστιν) is the same as knowing *why* it is (διὰ τί ἐστιν)[2].” “*What* is an eclipse?



A third general observation of Aristotle which is specially relevant to geometrical definitions is that “to know *what* a thing is (τί ἐστιν) is the same as knowing *why* it is (διὰ τί ἐστιν)[2].” “*What* is an eclipse?

Sarkar (2000)

A Theory of Adaptation: Prateek Sarkar's Style-Conscious Recognition

- Many documents possess a consistent *style:*
  - *e.g.* printed in one (or only a few) typefaces
  - or, handwritten by one person
  - or, noisy in a particular way
  - or, using a fixed page layout
  - ….(many examples)

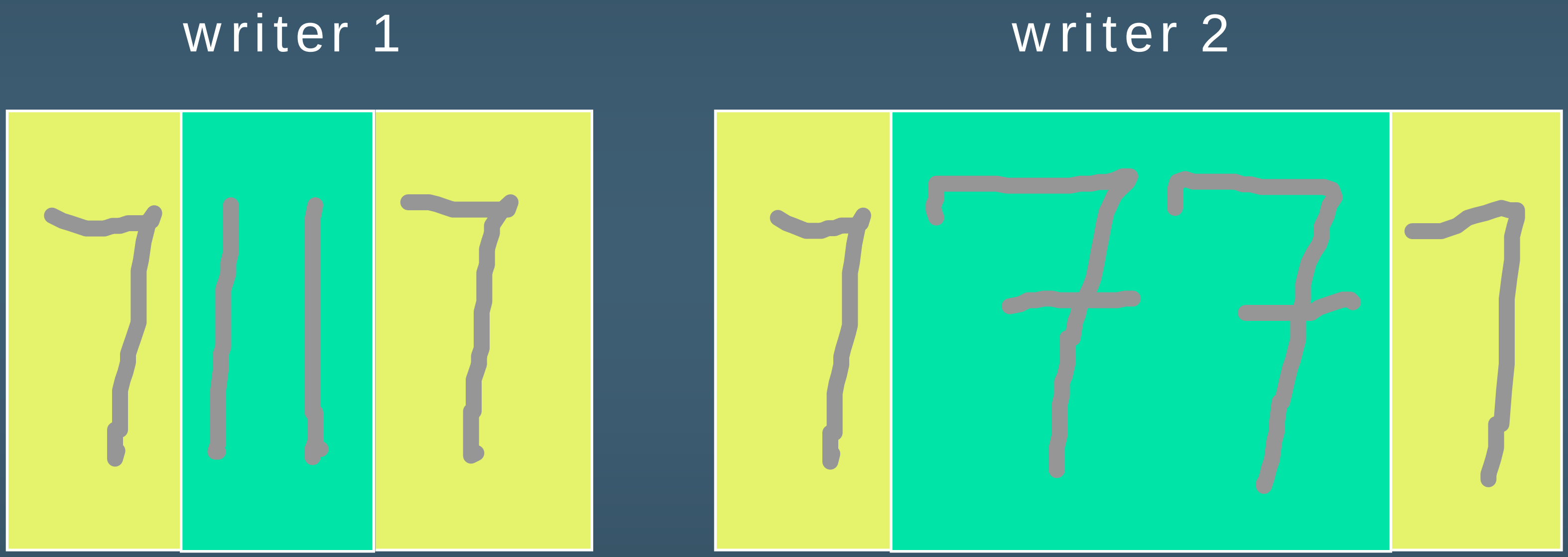
- Broadly applicable idea: a style is a manner of rendering (or, generating) patterns.

- Isogenous— *i.e*. 'generated from the same source' —documents possess a uniform style

Sarkar, Nagy, Veeramachaneni (2000-2005)

Style-Consistent Recognition

Sevens, or ones…? Ambiguous!

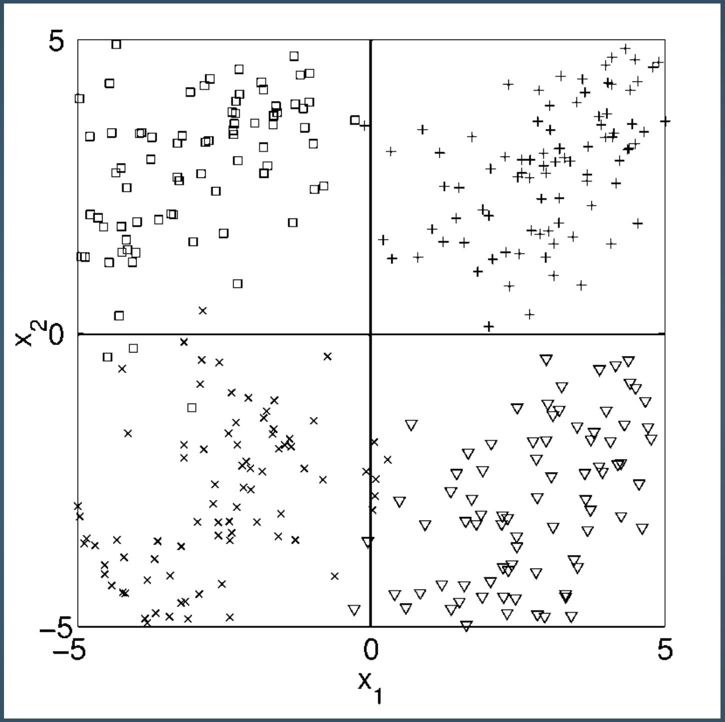


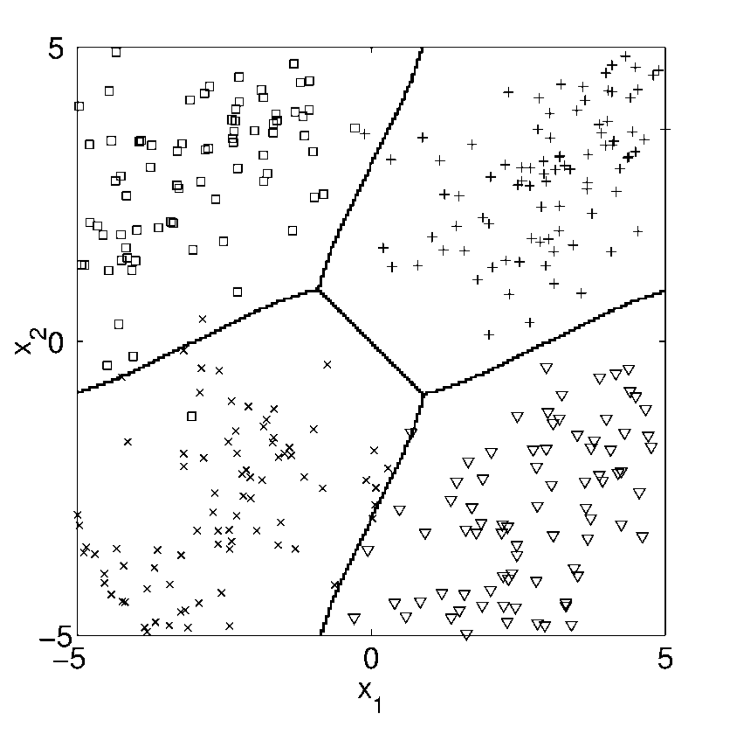
Ambiguity is resolved by style-consistency.

Style-Conscious Methodology

- Modeling style-consistency improves classification on isogenous input
- Improvement is higher on longer input passages
- Styles and style parameters can be estimated without style labels
- Style models complement, and do not impede, other recognition models (*e.g.* linguistic)
- ***Lesson:* *weak models can become stronger* *when operating on long isogenous passages***

Refined Classifier Decision Regions
(for a passage with two symbols)



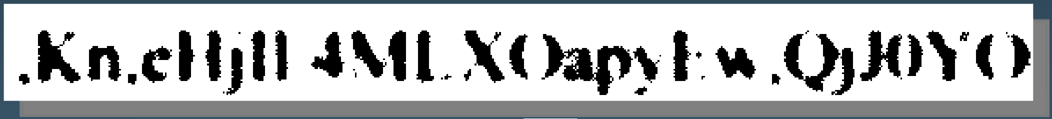


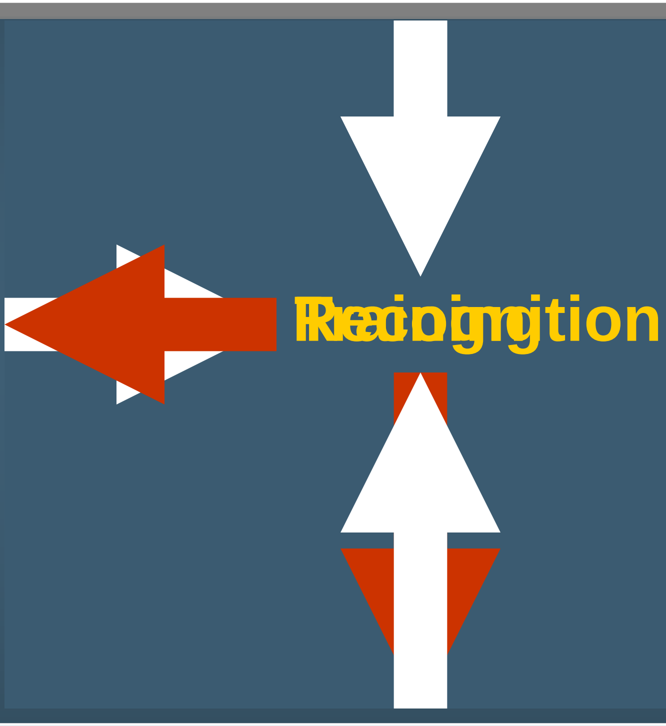
Style-unconscious: suboptimal

Style-conscious: optimal

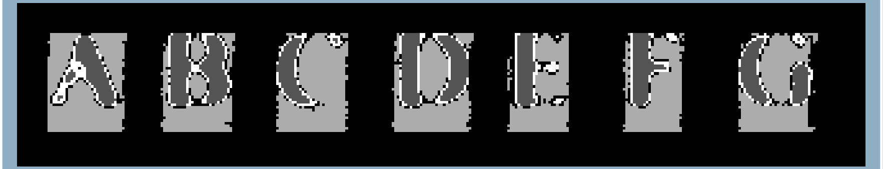
PARC's Document Image Decoding

Text image

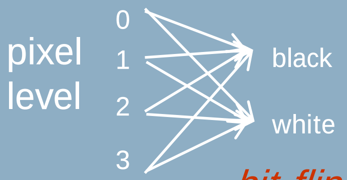




Multi-level image degradation model



Character templates



Side-bearing model

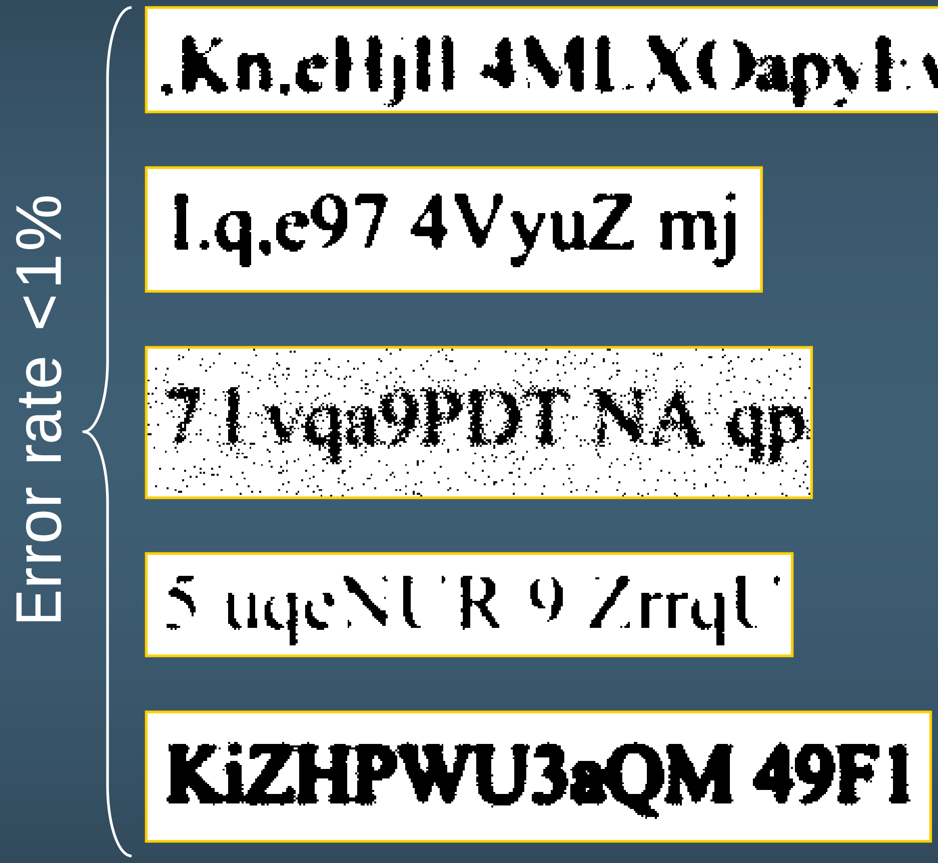
bit-flip probabilities

Text content

```
.Kn,eHjl1 4MLXOapykw,QjJ0YO
```

Kopec, Chou, Kam, & Lomelin (1994-1997)

DID can learn Strong Models of even extreme image degradations



- Works over a wide range of image qualities
- A system can adapt to any of a large set of pre-trained qualities

Sarkar, Baird, & Zhang (2003)

DID is Model-Intensive

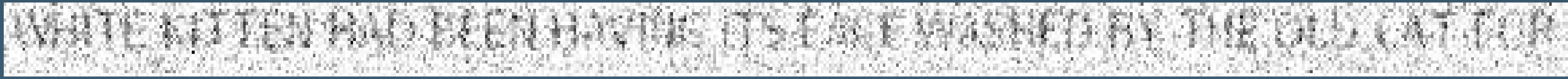
- Explicit formal stochastic models of
  - **text generation**: language, format
  - **image rendering**: typefaces, layout
  - **image quality**: asymmetric bit-flip

  (combined in a single finite-state Markov network)
- Search algorithms find best ‘decoding’
  - provably optimal (under MAP criterion)
  - Viterbi and Iterated Complete Path: often fast
- Joint over many models, some weak:
  - linguistic: char *N*-gram & imperfect lexica
  - quality: simplistic bit-flip model

Kopec, Chou, Minka, Popat, Bloomberg (1994-2001)

Weak Language Models Can Help Overcome Severe Image Noise

Degraded, subsampled, greyscale image



DID recognition without a language model

WHITR.KITTIVI HAO BEEN HAVING IT.,.RACE,WASHEI4.BX THB.UI,D CAT FOR

DID w/ n-gram char model, Iterated Complete Path search algorithm

WHITE KITTEN HAD BEEN HAVING ITS FACE WASHED BY THE OLD CAT FOR

Kopec, Popat, Bloomberg, Greene (2000-2002)

K. Popat, "Decoding of Text Tines in Grayscale Document Images," *Proc., ICASSP*, Salt Lake City, Utah, May 2001.

Lessons from DID

- Combining several models, even if some are weak, can yield high accuracy
- Joint recognition over many models---iconic, linguistic, quality, layout---can be performed provably optimally, and fast
- Recognizing entire text-lines at a time helps
- ***Weak models can provide the basis for high performance recognition systems***

Extremely Long Passages:
“Whole-Book” Recognition

Operate on the complete set of a book's page images, using automatic unsupervised adaptation to improve accuracy.

Given: *(1) images of an entire book,*

(2) an initial transcription (generally erroneous), &

(3) a dictionary (generally imperfect),

Try to: *improve recognition accuracy fully automatically, guided only by evidence within the images.*

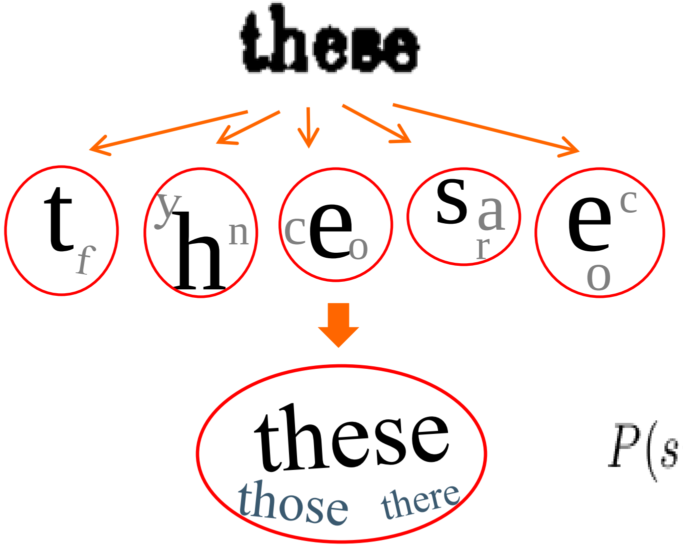
Xiu & Baird (2008-2011)

Start with Two Weak Models

- **Iconic model:**
  - *Describes image formation and determines the behaviour of a character-image classifier*
  - *For example, the prototypes in a template-matching character classifier.*
  - *Weak: inferred from buggy OCR transcription*
- **Linguistic model:**
  - *Describes word-occurrence probabilities*
  - *For example, a dictionary*
  - *Weak: not a perfect lexicon: too small (or too large)*

Word recognition, driven by (1) iconic model alone, and (2) both iconic and linguistic models (jointly), may get different results, indicating "disagreements" between the models.

Disagreements can be Detected Statistically



Char recognition (apply iconic model alone):

$$P(s_1|x_1)\cdots P(s_T|x_T)$$

Word recognition (iconic & linguistic jointly):

$$P(s_1\cdots s_T|x_1\cdots x_T) = \frac{1}{\alpha}P(s_1|x_1)\cdots P(s_T|x_T)C(s_1\cdots s_T)$$

***Character** disagreement:* $\epsilon(i|X) \equiv -\sum_{s\in\Sigma} P(s_i = s|X)\cdot\log P(s|x_i)$ (**cross entropy** on a char)

***Word** disagreement:* $\epsilon(X) = \sum_{i=1}^{T}\epsilon(i|X)$ (…w/in a word)

***Passage** disagreement:* $\epsilon(\mathcal{P}) = \sum_{X\in\mathcal{P}}\epsilon(X)$ (…w/in the whole book)

Disagreement-Driven Model Adaptation Algorithm

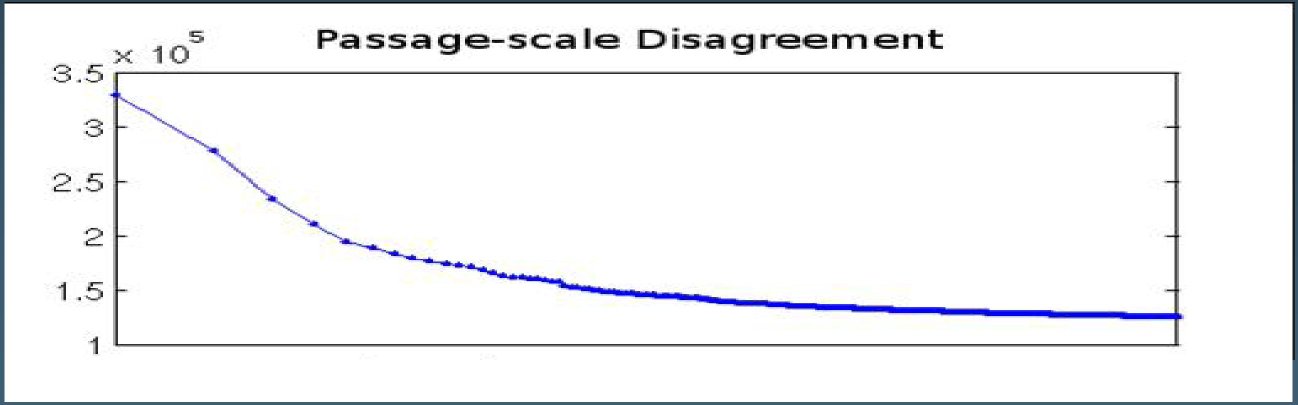
Iterate many times….

- Compute all character, word & passage disagreements
- Identify words and characters where the two models most disagree.
- Propose adaptations to the models to reconcile them.
- Check that each proposed adaptation reduces passage disagreement: if so, accept the adaptation.

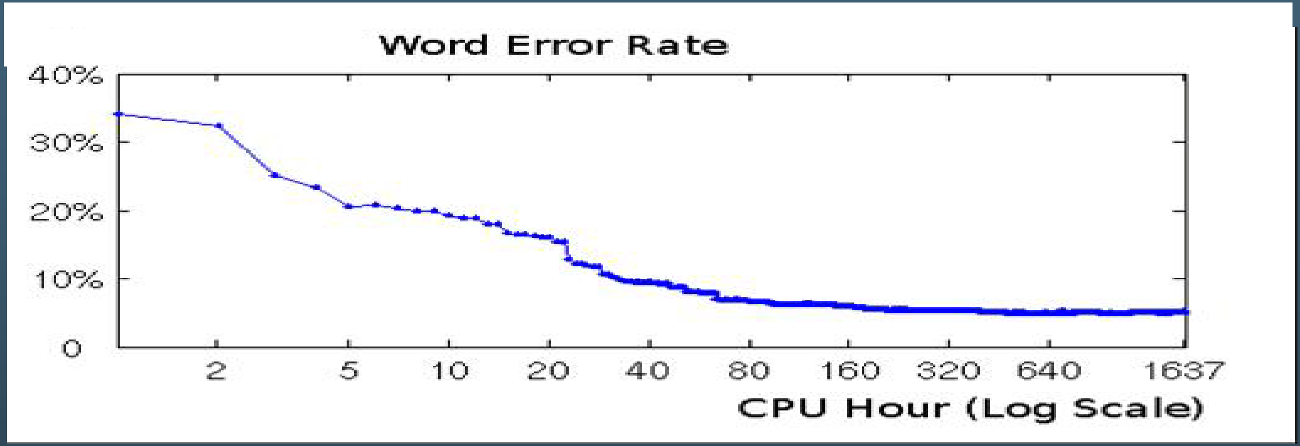
The two models are "criticizing" one another, & correcting one another—although both are imperfect!

Disagreements Identify Errors

The algorithm drives disagreements down.....



...and, disagreements are correlated with errors....

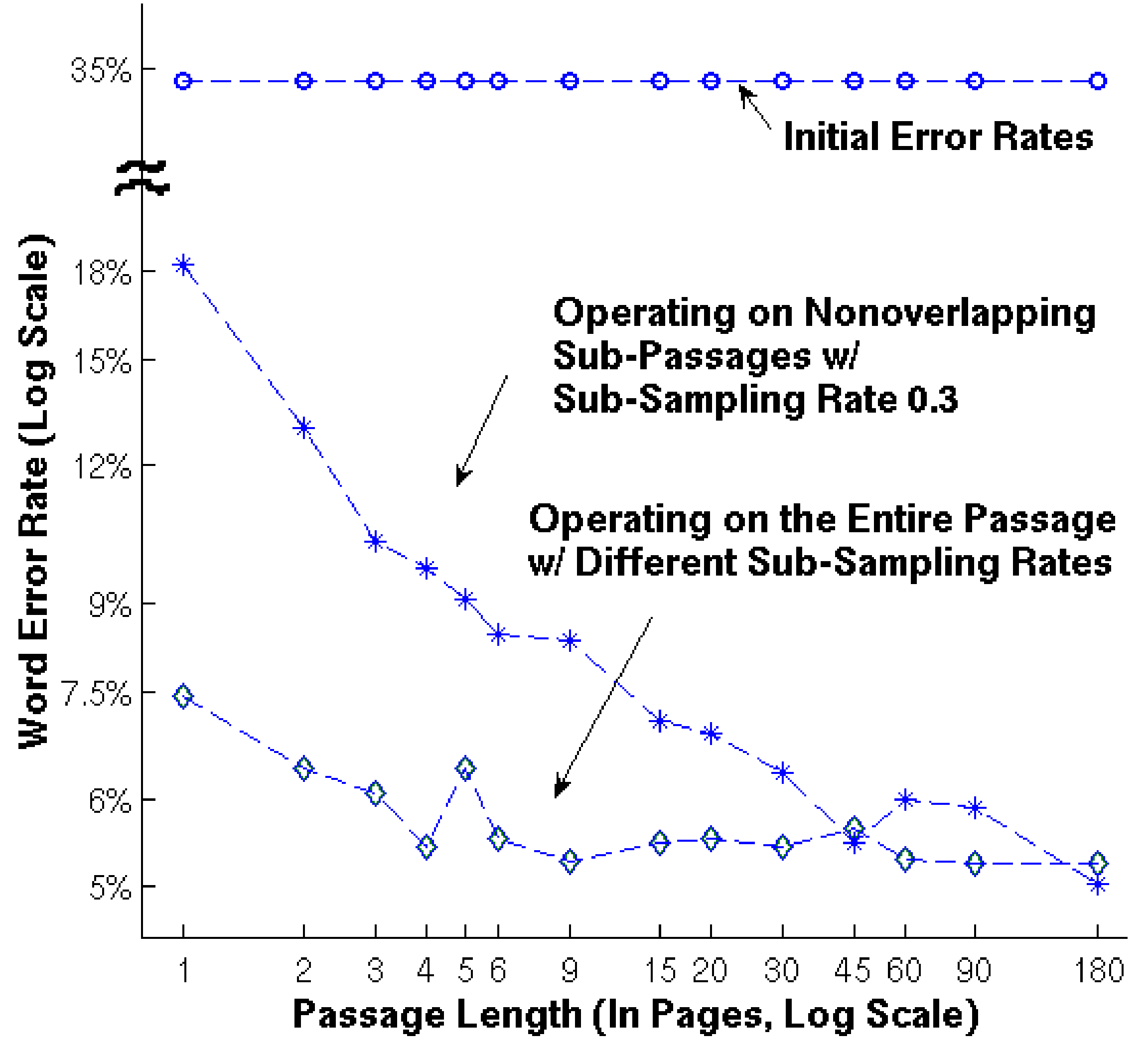


...so, the algorithm drives down errors.

Longer Passages Improve More

Benefits of isogeny:

Longer passages are driven to lower error rates ultimately.



Pingping Xiu's Whole-Book Recognition

- The larger the input passage is, the better the algorithm performs: the lower the final error rate.
- The algorithm can be sped up by two orders of magnitude using randomization and caching.
- Rigorous sufficient conditions for the algorithm to succeed have been proven.
- Two weak models, although both are imperfect, can criticize and correct one another, both becoming stronger.

Enables ‘Anytime’ Recognition

- Recognizers which run ‘forever’
  - safe, since accuracy improves nearly monotonically
  - trade runtime for (eventual) accuracy
- Can be interrupted at any time to see the best interpretation found so far
  - system is always operating on the *entire* document
- A good fit to ‘personal recognition’ needs
  - users are unskilled: can’t engineer; won’t correct
  - no tight deadline: soak up idle cycles

Twenty-five Years of DAR Research:
Model-Intensive Recognition

- **Specify the domain precisely:**

  define quantitative generative models
  of document images to be recognized

- **Learn models from examples:**

  synthetic training data can be safe;
  affordable weak models may be good enough

- **Strive for provable performance guarantees:**

  invent joint recognition algorithms which are
  formally optimal w.r.t. to the models

- **Adapt weak models, strengthen automatically:**

  on short passages, apply style-conscious adaptation;
  on long passages, mutual criticism and correction

Advantages of Model-Intensive Recognition

- When the models are strong (closely fit the input),
  results are the best possible
- When models can be trained nearly automatically,
  effort required for best results is minimized
- When training is known to work across a wide range,
  confidence in high performance is high
- If the system isn't yet good enough:
  improve the models: adaptively perhaps
  ---but not the recognition algorithms!

Focusing on a peculiar distinction: ‘Strong’ *versus* ‘Weak’ Models

Shifts our attention away from end-results:
accuracy, speed, and costs of engineering
—and towards this question:

How well do our models fit the particular input which our system is trying to recognize?

The answer to this can determine accuracy, engineering costs, even speed….

By working this way, we may enjoy the best of both: affordable engineering costs, plus high accuracy!

Thanks!

And thanks to all those who inspired me, especially:

Theo Pavlidis, Tin Kam Ho, David Ittner, Ken Thompson, George Nagy, Robert Haralick, Tao Hong, Tapas Kanungo, Phil Chou, Dan Lopresti, Gary Kopec, Dan Bloomberg, Ashok Popat, Tom Breuel, Elisa Barney Smith, Prateek Sarkar, Harsha Veeramachaneni, Jean Nonnemaker, and Pingping Xiu.