Lecture 5: Designing smart systems

Using statistical methods to anticipate user needs and actions with Bayesian strategies

Overview of the course

- Theory driven approaches to HCI
- Design of visual displays
- Goal-oriented interaction
- Designing efficient systems
- **Designing smart systems**
- Designing meaningful systems (guest lecturer)
- Evaluating interactive system designs
- Designing complex systems

Uniform text entry

esc F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12
~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 — - + = delete
tab Q W E R T Y U I O P { [}] | \
caps lock A S D F G H J K L : ; " ' return
shift Z X C V B N M < , > . ? / shift
fn control option command command option

Apple UK

Information gain per key press

$$h(x_i) = \log \frac{1}{p(x_i)}$$

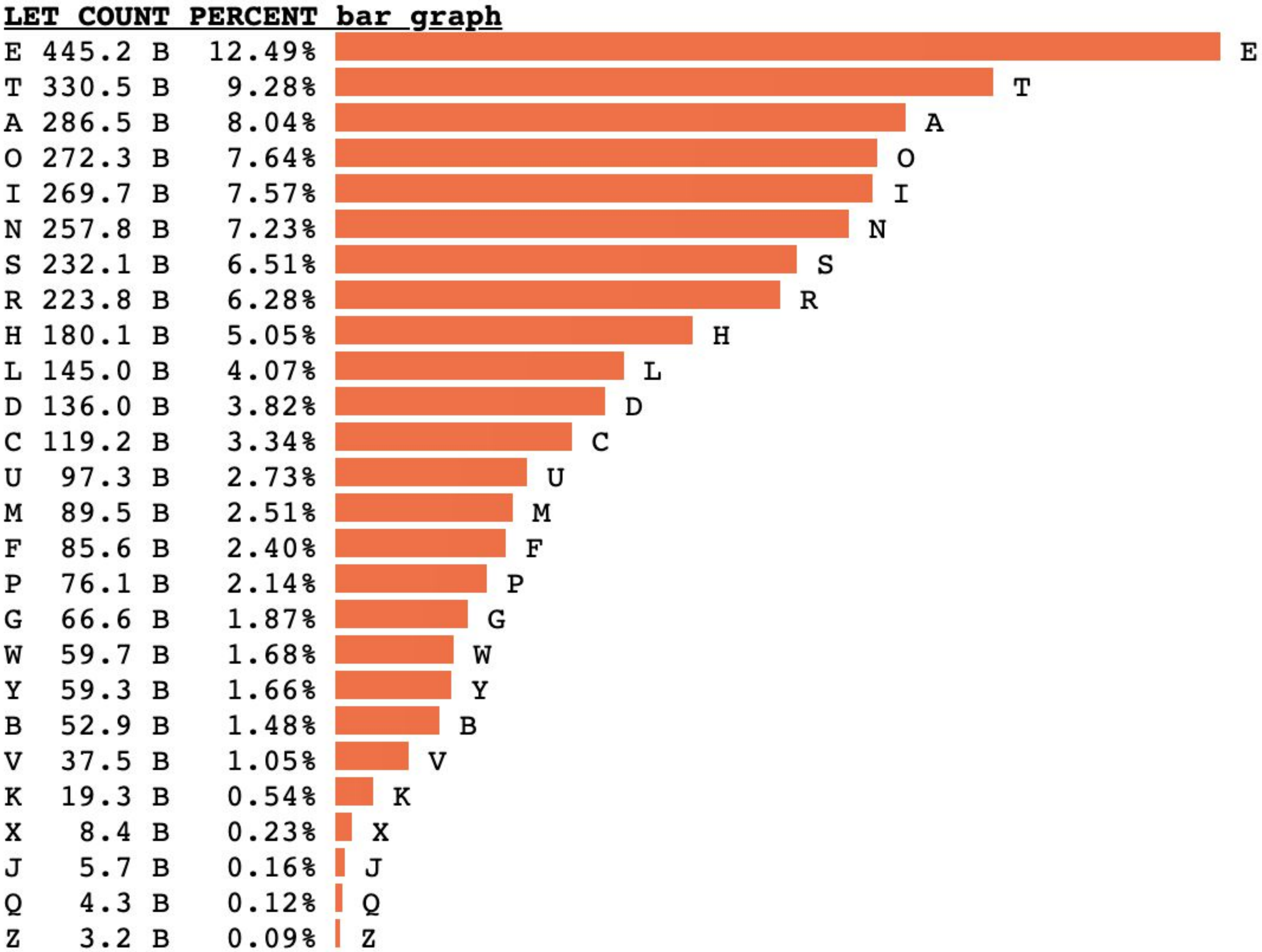
The q?

Information gain per key press

"As you are aware, E is the most common letter in the English alphabet, and it predominates to so marked an extent that even in a short sentence one would expect to find it most often"

The Adventure of the Dancing Men,

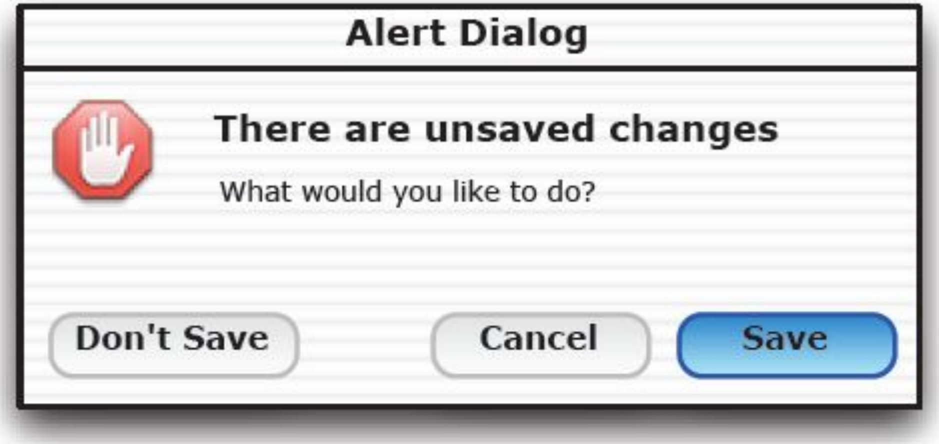
Sir Arthur Conan Doyle



LET	COUNT	PERCENT	bar graph
E	445.2 B	12.49%	E
T	330.5 B	9.28%	T
A	286.5 B	8.04%	A
O	272.3 B	7.64%	O
I	269.7 B	7.57%	I
N	257.8 B	7.23%	N
S	232.1 B	6.51%	S
R	223.8 B	6.28%	R
H	180.1 B	5.05%	H
L	145.0 B	4.07%	L
D	136.0 B	3.82%	D
C	119.2 B	3.34%	C
U	97.3 B	2.73%	U
M	89.5 B	2.51%	M
F	85.6 B	2.40%	F
P	76.1 B	2.14%	P
G	66.6 B	1.87%	G
W	59.7 B	1.68%	W
Y	59.3 B	1.66%	Y
B	52.9 B	1.48%	B
V	37.5 B	1.05%	V
K	19.3 B	0.54%	K
X	8.4 B	0.23%	X
J	5.7 B	0.16%	J
Q	4.3 B	0.12%	Q
Z	3.2 B	0.09%	Z

https://norvig.com/mayzner.html

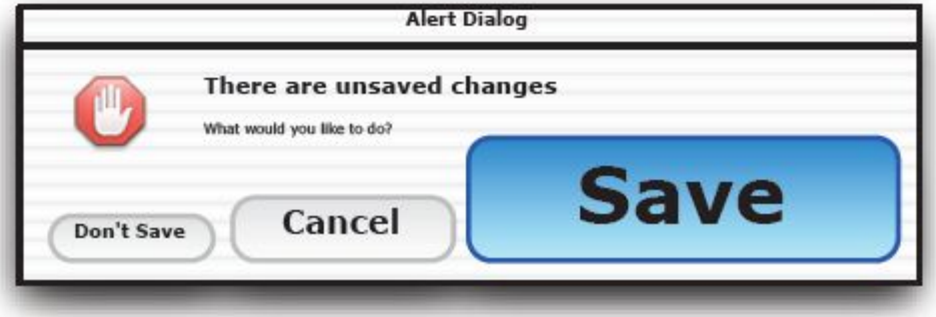
Hacking Fitt’s Law: “semantic pointing”



Alert Dialog

There are unsaved changes

What would you like to do?



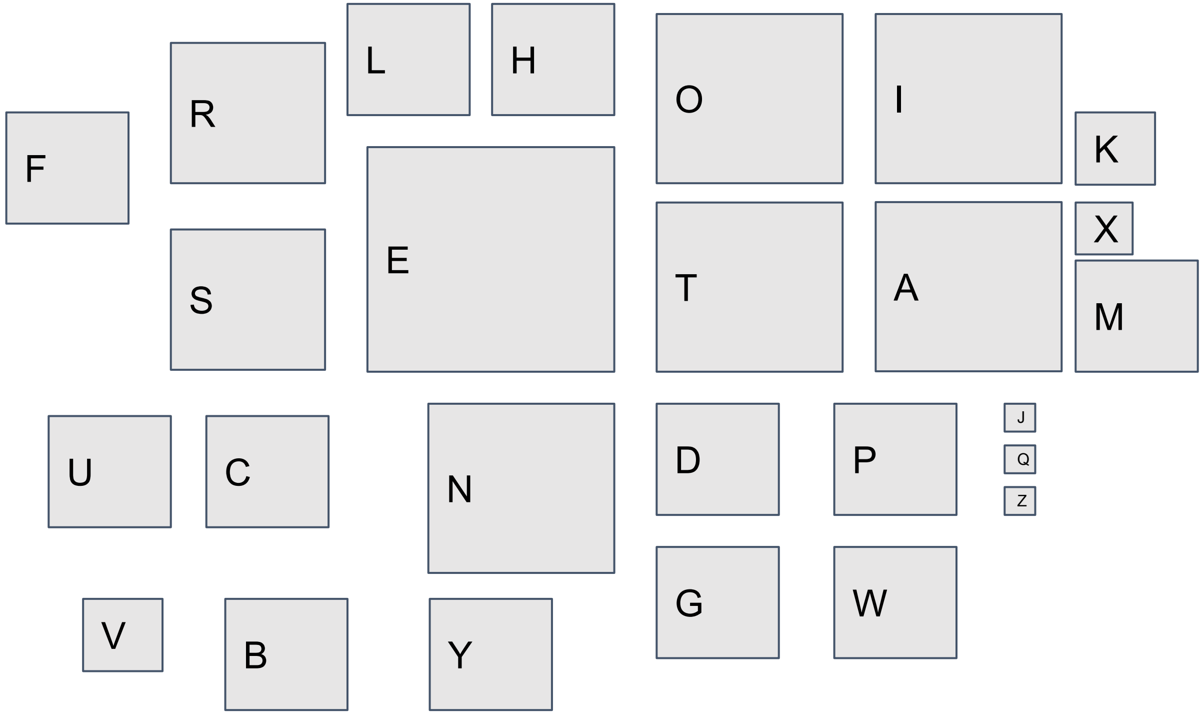
Alert Dialog

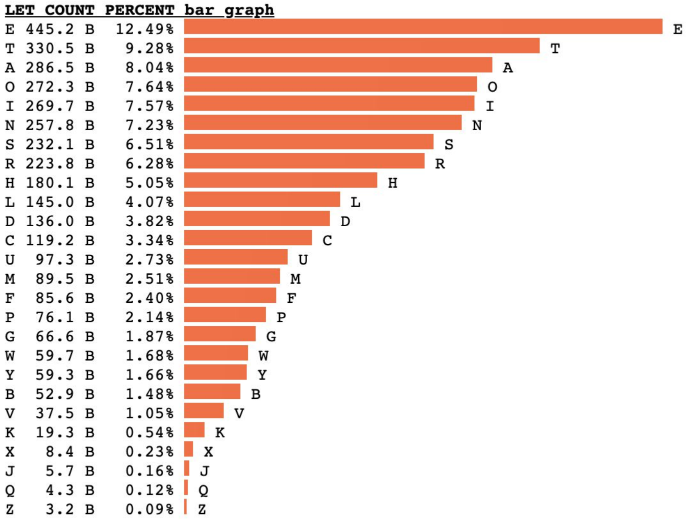
There are unsaved changes

What would you like to do?

Renaud Blanch, Yves Guiard and Michel Beaudouin-Lafon. **Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation.** In *Proceedings of CHI 2004*, pages 519-526, Vienna - Austria, April 2004.

Simple application of Fitts Law

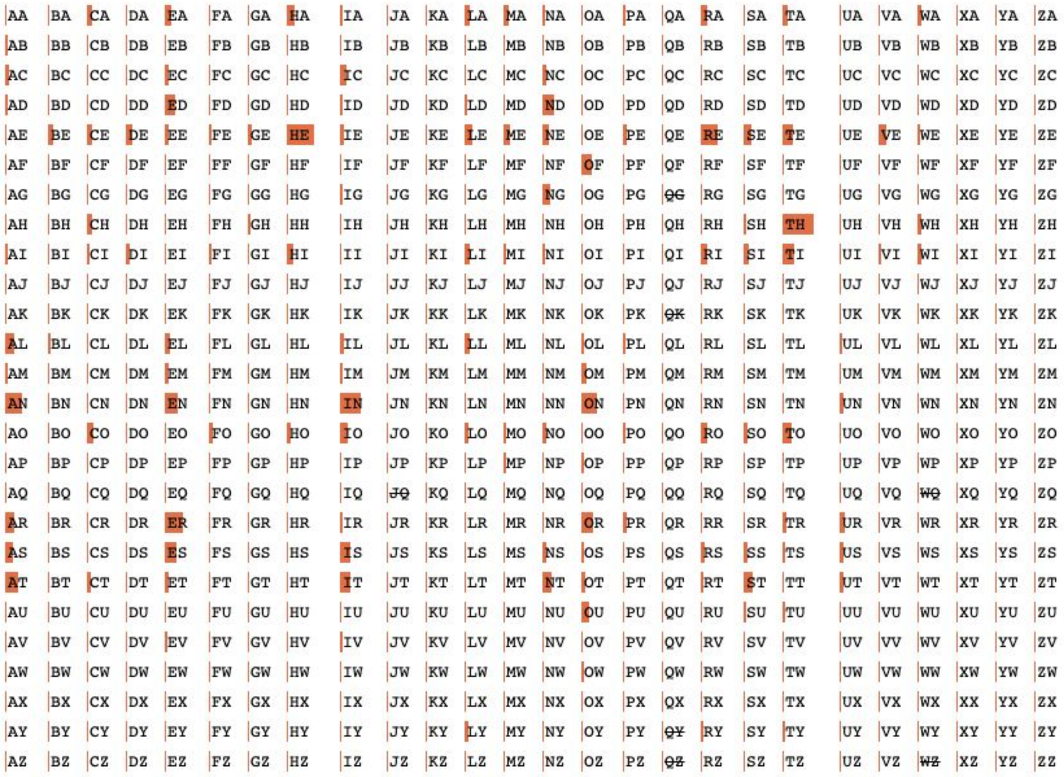




```
LET COUNT PERCENT bar graph
E 445.2 B  12.49%
T 330.5 B   9.28%
A 286.5 B   8.04%
O 272.3 B   7.64%
I 269.7 B   7.57%
N 257.8 B   7.23%
S 232.1 B   6.51%
R 223.8 B   6.28%
H 180.1 B   5.05%
L 145.0 B   4.07%
D 136.0 B   3.82%
C 119.2 B   3.34%
U  97.3 B   2.73%
M  89.5 B   2.51%
F  85.6 B   2.40%
P  76.1 B   2.14%
G  66.6 B   1.87%
W  59.7 B   1.68%
Y  59.3 B   1.66%
B  52.9 B   1.48%
V  37.5 B   1.05%
K  19.3 B   0.54%
X   8.4 B   0.23%
J   5.7 B   0.16%
Q   4.3 B   0.12%
Z   3.2 B   0.09%
```

What’s wrong with this?

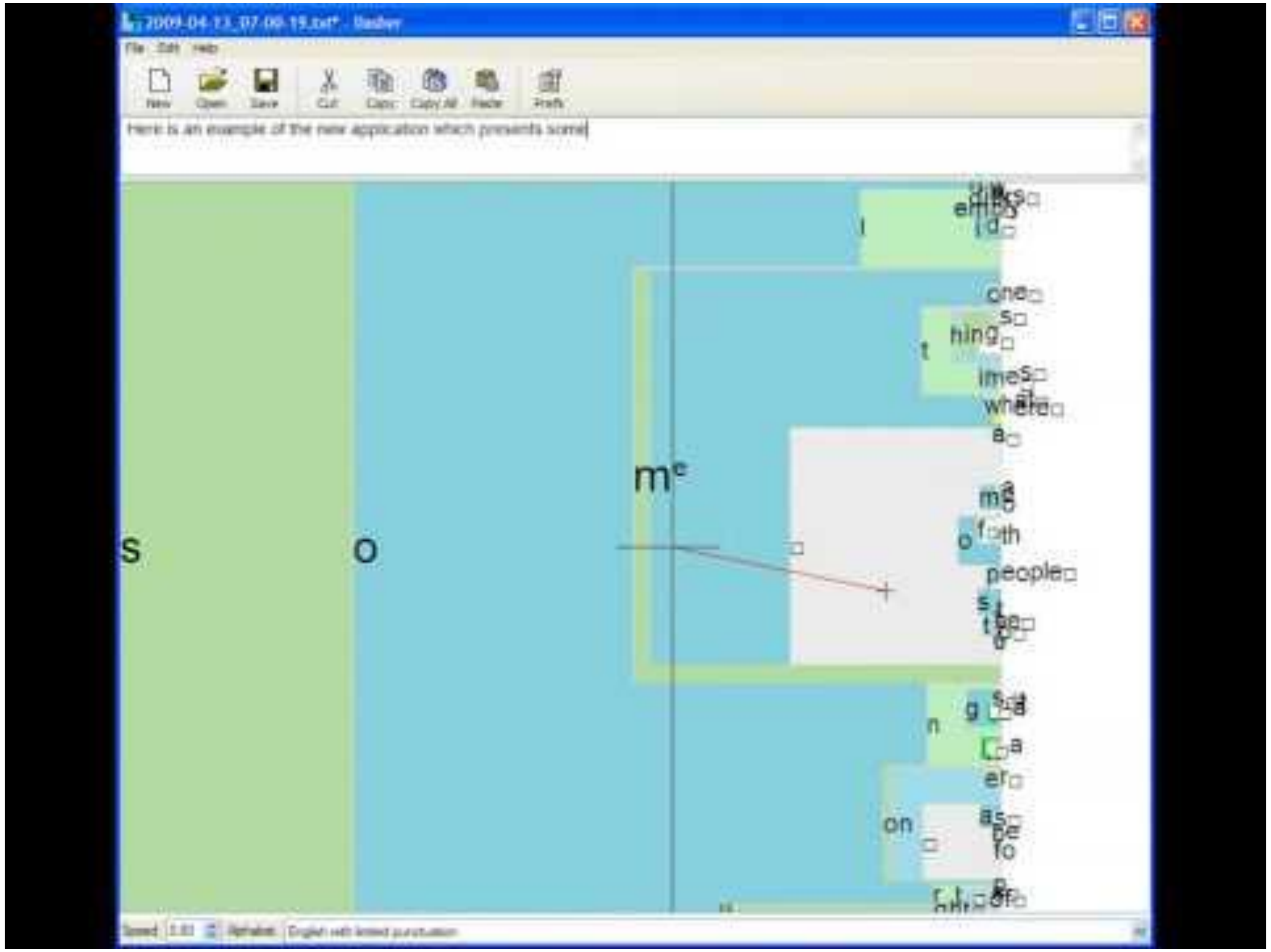
Bigrams



AA	BA	CA	DA	EA	FA	GA	HA	IA	JA	KA	LA	MA	NA	OA	PA	QA	RA	SA	TA	UA	VA	WA	XA	YA	ZA
AB	BB	CB	DB	EB	FB	GB	HB	IB	JB	KB	LB	MB	NB	OB	PB	QB	RB	SB	TB	UB	VB	WB	XB	YB	ZB
AC	BC	CC	DC	EC	FC	GC	HC	IC	JC	KC	LC	MC	NC	OC	PC	QC	RC	SC	TC	UC	VC	WC	XC	YC	ZC
AD	BD	CD	DD	ED	FD	GD	HD	ID	JD	KD	LD	MD	ND	OD	PD	QD	RD	SD	TD	UD	VD	WD	XD	YD	ZD
AE	BE	CE	DE	EE	FE	GE	HE	IE	JE	KE	LE	ME	NE	OE	PE	QE	RE	SE	TE	UE	VE	WE	XE	YE	ZE
AF	BF	CF	DF	EF	FF	GF	HF	IF	JF	KF	LF	MF	NF	OF	PF	QF	RF	SF	TF	UF	VF	WF	XF	YF	ZF
AG	BG	CG	DG	EG	FG	GG	HG	IG	JG	KG	LG	MG	NG	OG	PG	~~QG~~	RG	SG	TG	UG	VG	WG	XG	YG	ZG
AH	BH	CH	DH	EH	FH	GH	HH	IH	JH	KH	LH	MH	NH	OH	PH	QH	RH	SH	TH	UH	VH	WH	XH	YH	ZH
AI	BI	CI	DI	EI	FI	GI	HI	II	JI	KI	LI	MI	NI	OI	PI	QI	RI	SI	TI	UI	VI	WI	XI	YI	ZI
AJ	BJ	CJ	DJ	EJ	FJ	GJ	HJ	IJ	JJ	KJ	LJ	MJ	NJ	OJ	PJ	QJ	RJ	SJ	TJ	UJ	VJ	WJ	XJ	YJ	ZJ
AK	BK	CK	DK	EK	FK	GK	HK	IK	JK	KK	LK	MK	NK	OK	PK	~~QK~~	RK	SK	TK	UK	VK	WK	XK	YK	ZK
AL	BL	CL	DL	EL	FL	GL	HL	IL	JL	KL	LL	ML	NL	OL	PL	QL	RL	SL	TL	UL	VL	WL	XL	YL	ZL
AM	BM	CM	DM	EM	FM	GM	HM	IM	JM	KM	LM	MM	NM	OM	PM	QM	RM	SM	TM	UM	VM	WM	XM	YM	ZM
AN	BN	CN	DN	EN	FN	GN	HN	IN	JN	KN	LN	MN	NN	ON	PN	QN	RN	SN	TN	UN	VN	WN	XN	YN	ZN
AO	BO	CO	DO	EO	FO	GO	HO	IO	JO	KO	LO	MO	NO	OO	PO	QO	RO	SO	TO	UO	VO	WO	XO	YO	ZO
AP	BP	CP	DP	EP	FP	GP	HP	IP	JP	KP	LP	MP	NP	OP	PP	QP	RP	SP	TP	UP	VP	WP	XP	YP	ZP
AQ	BQ	CQ	DQ	EQ	FQ	GQ	HQ	IQ	~~JQ~~	KQ	LQ	MQ	NQ	OQ	PQ	QQ	RQ	SQ	TQ	UQ	VQ	~~WQ~~	XQ	YQ	ZQ
AR	BR	CR	DR	ER	FR	GR	HR	IR	JR	KR	LR	MR	NR	OR	PR	QR	RR	SR	TR	UR	VR	WR	XR	YR	ZR
AS	BS	CS	DS	ES	FS	GS	HS	IS	JS	KS	LS	MS	NS	OS	PS	QS	RS	SS	TS	US	VS	WS	XS	YS	ZS
AT	BT	CT	DT	ET	FT	GT	HT	IT	JT	KT	LT	MT	NT	OT	PT	QT	RT	ST	TT	UT	VT	WT	XT	YT	ZT
AU	BU	CU	DU	EU	FU	GU	HU	IU	JU	KU	LU	MU	NU	OU	PU	QU	RU	SU	TU	UU	VU	WU	XU	YU	ZU
AV	BV	CV	DV	EV	FV	GV	HV	IV	JV	KV	LV	MV	NV	OV	PV	QV	RV	SV	TV	UV	VV	WV	XV	YV	ZV
AW	BW	CW	DW	EW	FW	GW	HW	IW	JW	KW	LW	MW	NW	OW	PW	QW	RW	SW	TW	UW	VW	WW	XW	YW	ZW
AX	BX	CX	DX	EX	FX	GX	HX	IX	JX	KX	LX	MX	NX	OX	PX	QX	RX	SX	TX	UX	VX	WX	XX	YX	ZX
AY	BY	CY	DY	EY	FY	GY	HY	IY	JY	KY	LY	MY	NY	OY	PY	~~QY~~	RY	SY	TY	UY	VY	WY	XY	YY	ZY
AZ	BZ	CZ	DZ	EZ	FZ	GZ	HZ	IZ	JZ	KZ	LZ	MZ	NZ	OZ	PZ	~~QZ~~	RZ	SZ	TZ	UZ	VZ	~~WZ~~	XZ	YZ	ZZ

Increasing the depth of the language allows for a further separation...

Building a system based on relative frequencies



2009-04-13_07-00-19.txt* - Dasher

Here is an example of the new application which presents some

Speed: 1.81 Alphabet: English with limited punctuation

Dasher
(https://www.youtube.com/watch?v=FLalNywdHxU)

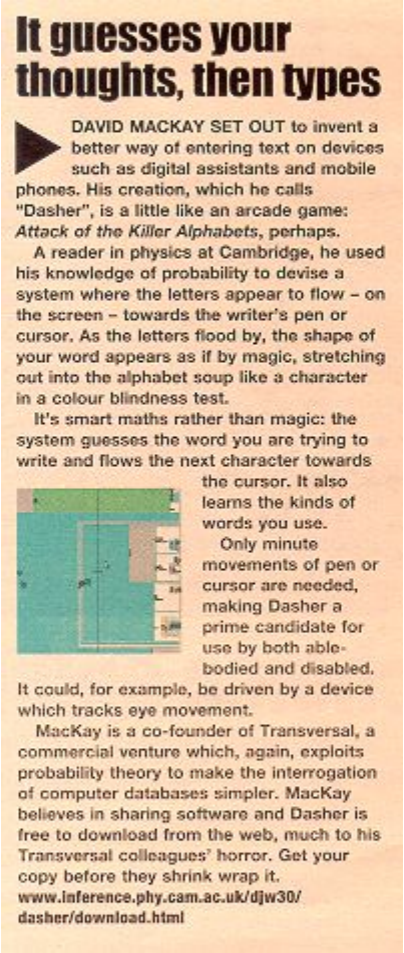
Some lessons from Dasher

- Turning an information theoretic model into a user interface requires a lot of creativity

  => Interaction with Machine learning course

- In many cases simple models (nGrams + smoothing) are as - or more - effective than complex ones (neural nets)

- Supporting even famous software, useful for marginalised groups is hard



It guesses your thoughts, then types

DAVID MACKAY SET OUT to invent a better way of entering text on devices such as digital assistants and mobile phones. His creation, which he calls "Dasher", is a little like an arcade game: *Attack of the Killer Alphabets*, perhaps.

A reader in physics at Cambridge, he used his knowledge of probability to devise a system where the letters appear to flow – on the screen – towards the writer's pen or cursor. As the letters flood by, the shape of your word appears as if by magic, stretching out into the alphabet soup like a character in a colour blindness test.

It's smart maths rather than magic: the system guesses the word you are trying to write and flows the next character towards the cursor. It also learns the kinds of words you use.

Only minute movements of pen or cursor are needed, making Dasher a prime candidate for use by both able-bodied and disabled. It could, for example, be driven by a device which tracks eye movement.

MacKay is a co-founder of Transversal, a commercial venture which, again, exploits probability theory to make the interrogation of computer databases simpler. MacKay believes in sharing software and Dasher is free to download from the web, much to his Transversal colleagues' horror. Get your copy before they shrink wrap it.

www.inference.phy.cam.ac.uk/djw30/dasher/download.html

(The Financial Times, February 2002)

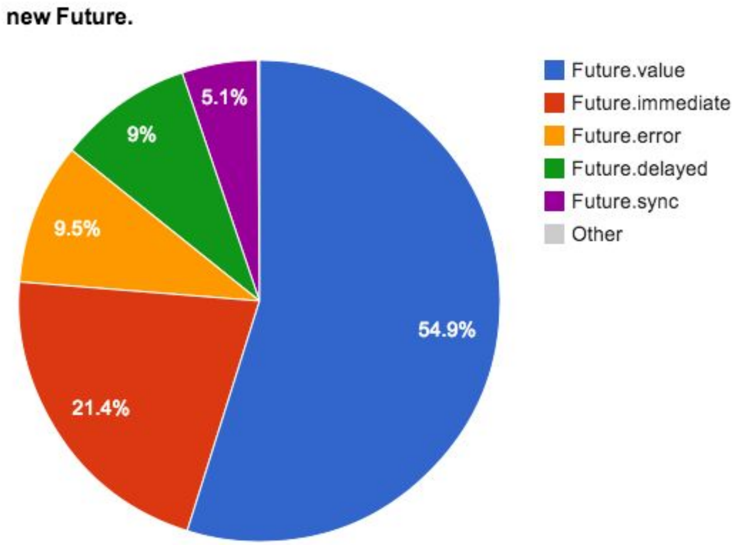
Artificial languages

```
new Future.?
```

```
New Future.aaaaa()
New Future.aaaab()
New Future.aaaac()
New Future.error()
…
New Future.aaaad()
New Future.aaaae()
New Future.value()
```

Artificial languages

`new Future.?`



Liveness becomes Entelechy, Church et al

Ordering code completion suggestions

A simple scheme for predicting code completions:

```
void             main()          {
    Stopwatch sw = new Stopwatch();
     sw. // <--- What goes here?
}
```

elapsed
elapsedMicroseconds
elapsedMilliseconds
elapsedTicks
Frequency
hashCode
isRunning
noSuchMethod
Reset
runtimeType
Start
Stop
toString

Ordering code completion suggestions

We calculate:

```
P(completion = “reset”    | context = “void main() { Stopwatch sw = new Stopwatch(); sw.”)
P(completion = “start”    | context = “void main() { Stopwatch sw = new Stopwatch(); sw.”)

…
```

And the usual:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)},$$

Ordering code completion suggestions

$$P(completion = ? \mid context = \text{“...”}) \propto P(context = \text{“...”} \mid completion = ?)\ P(completion = ?)$$

Feature vector

Completion c	Count of seen completions	P(completion)
start	10	0.5
reset	5	0.25
elapsed	5	0.25

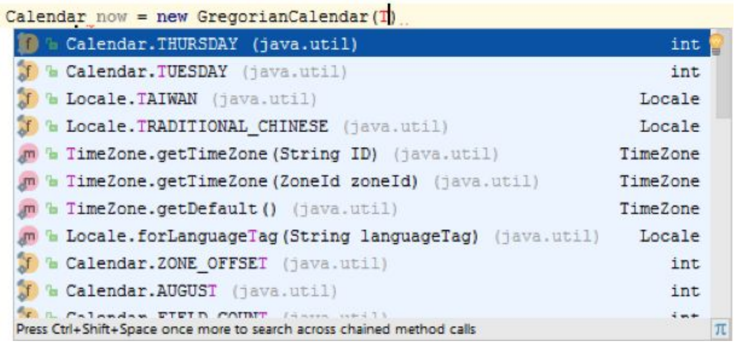
Ordering code completion suggestions

$$P(\mathit{completion} = c \mid \mathit{context} = \text{“...”}) \propto P(\mathit{context} = \text{“...”} \mid \mathit{completion} = c)\ P(\mathit{completion} = c)$$

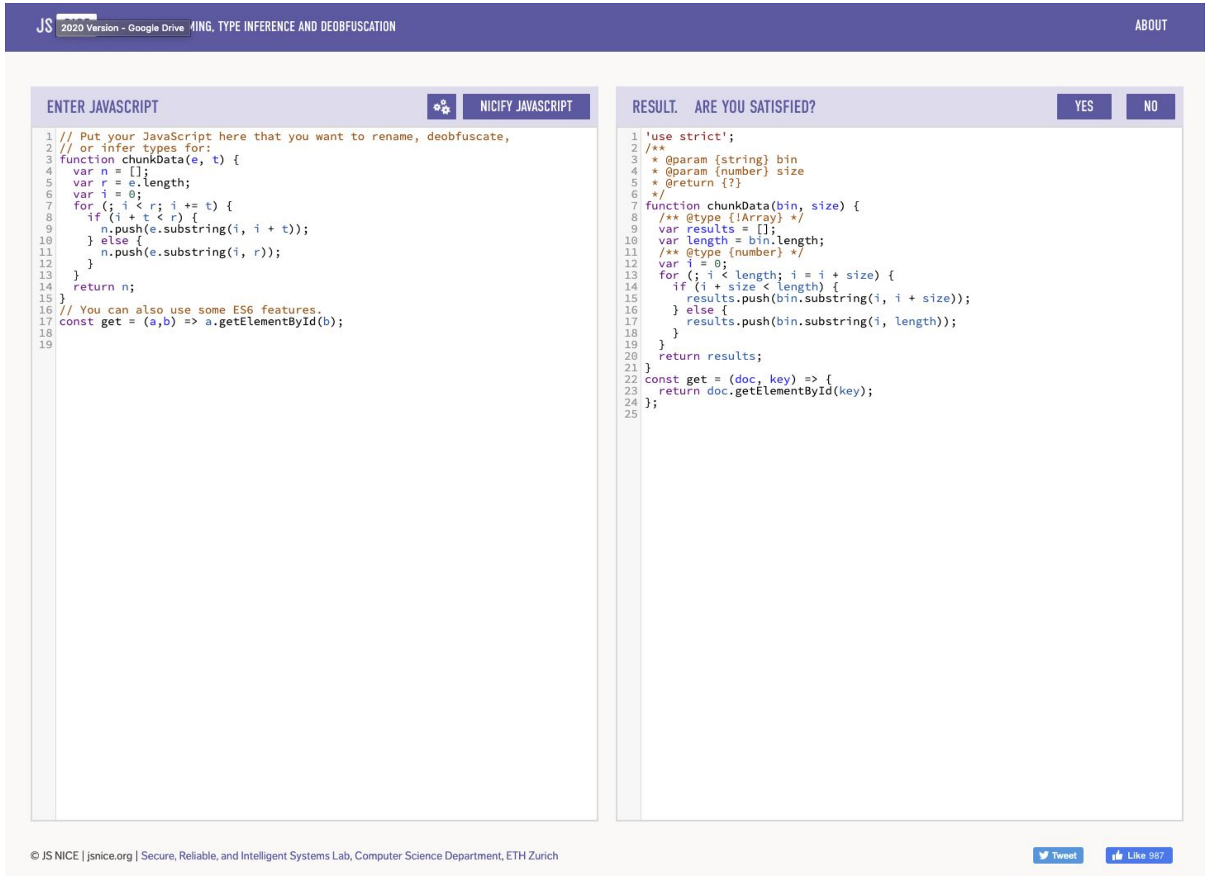
Completion c	P(completion==c | context) ∝	Order
start	0.9 * 0.5 = 0.45	0
reset	0.4 * 0.25 = 0.1	1
elapsed	0.2 * 0.25 = 0.06	2

Completion c	Feature	Feature value	Count
start	“First-Use”	true	9
		false	1
reset	“First-Use”	true	2
		false	3
elapsed	“First-Use”	true	1
		false	4

Some progress in information efficient IDEs



```
Calendar now = new GregorianCalendar(T)
Calendar.THURSDAY (java.util)                        int
Calendar.TUESDAY (java.util)                         int
Locale.TAIWAN (java.util)                         Locale
Locale.TRADITIONAL_CHINESE (java.util)            Locale
TimeZone.getTimeZone(String ID) (java.util)     TimeZone
TimeZone.getTimeZone(ZoneId zoneId) (java.util) TimeZone
TimeZone.getDefault() (java.util)               TimeZone
Locale.forLanguageTag(String languageTag) (java.util) Locale
Calendar.ZONE_OFFSET (java.util)                     int
Calendar.AUGUST (java.util)                          int
Calendar.FIELD_COUNT (java.util)                     int
Press Ctrl+Shift+Space once more to search across chained method calls
```



JS
2020 Version - Google Drive
MING, TYPE INFERENCE AND DEOBFUSCATION
ABOUT
ENTER JAVASCRIPT
NICIFY JAVASCRIPT
RESULT. ARE YOU SATISFIED?
YES
NO

```javascript
// Put your JavaScript here that you want to rename, deobfuscate,
// or infer types for:
function chunkData(e, t) {
  var n = [];
  var r = e.length;
  var i = 0;
  for (; i < r; i += t) {
    if (i + t < r) {
      n.push(e.substring(i, i + t));
    } else {
      n.push(e.substring(i, r));
    }
  }
  return n;
}
// You can also use some ES6 features.
const get = (a,b) => a.getElementById(b);
```

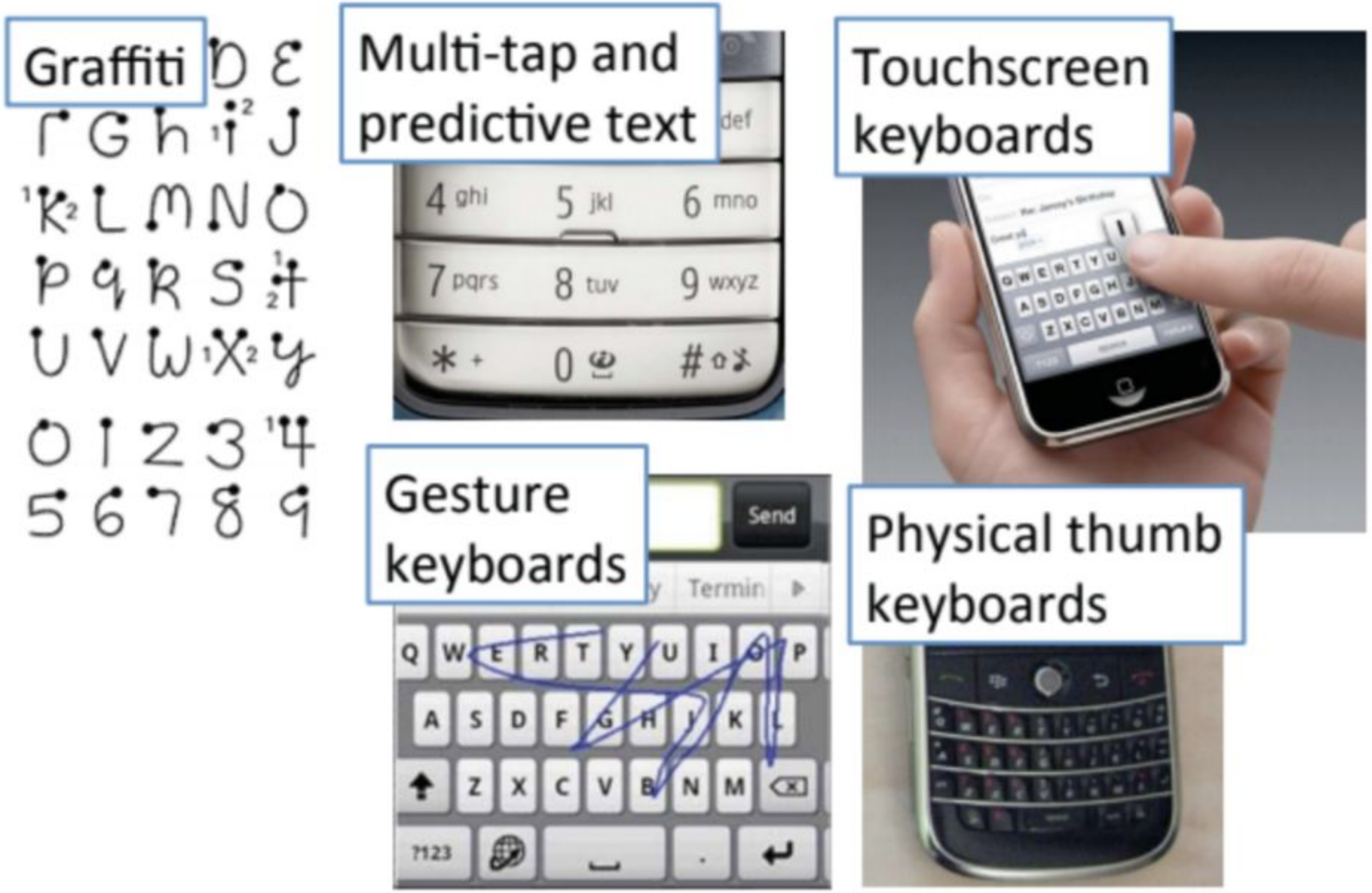
```javascript
'use strict';
/**
 * @param {string} bin
 * @param {number} size
 * @return {?}
 */
function chunkData(bin, size) {
  /** @type {!Array} */
  var results = [];
  var length = bin.length;
  /** @type {number} */
  var i = 0;
  for (; i < length; i = i + size) {
    if (i + size < length) {
      results.push(bin.substring(i, i + size));
    } else {
      results.push(bin.substring(i, length));
    }
  }
  return results;
}
const get = (doc, key) => {
  return doc.getElementById(key);
};
```

© JS NICE | jsnice.org | Secure, Reliable, and Intelligent Systems Lab, Computer Science Department, ETH Zurich

IntelliJ (Jetbrains)
JSNICE (ETH Zurich)

Building user interfaces

(from Per Ola's material)



Graffiti

Multi-tap and predictive text

Touchscreen keyboards

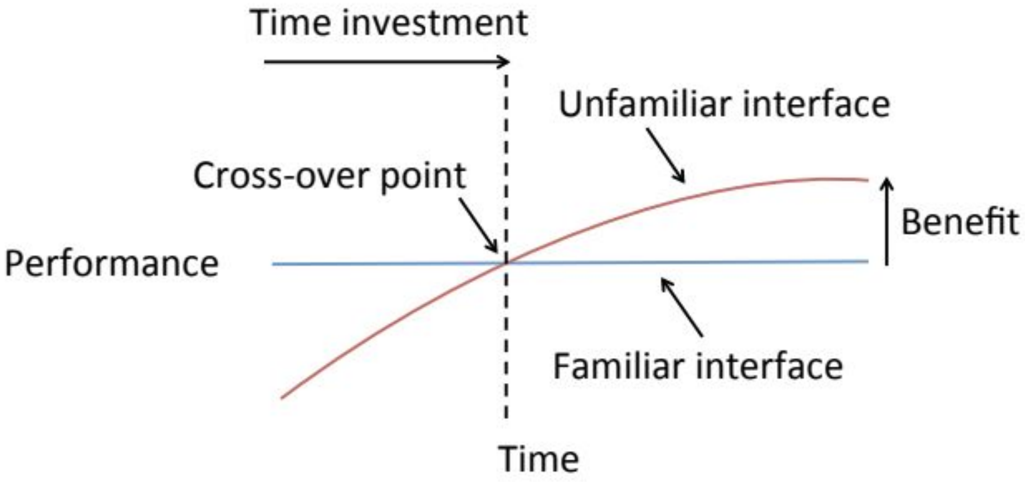
Gesture keyboards

Physical thumb keyboards

Building user interfaces
(from Per Ola’s material)

- Entry and error rate
- Learning curve, familiarity and immediate efficacy
- Form factor, presentation, time and comfort
- User engagement
- Visual attention and cognitive resources
- Privacy
- Single vs Multi-character entry
- Specification vs Navigation
- One/Two handed
- Task integration
- Robustness
- Device independence
- Computational demands
- Manufacturing and support cost
- Localisation
- Market acceptance

The cross-over point



Building user interfaces: Solution principles

(from Per Ola’s material)

- From closed to open-loop
    - Avoid the need for a visual feedback loop
- Continuous novice-to-expert transition
    - Avoid explicit learning
- Path dependency
    - Avoid redesign the interaction layer
- Flexibility
    - Enable users to compose and edit in a variety of styles without explicit mode switching
- Efficiency
    - Let users’ creativity by the bottle-neck