

Symbolic Indirect Correlation: A New Paradigm for Pattern Recognition

Daniel Lopresti¹, George Nagy², and Sharad Seth³

November 2006

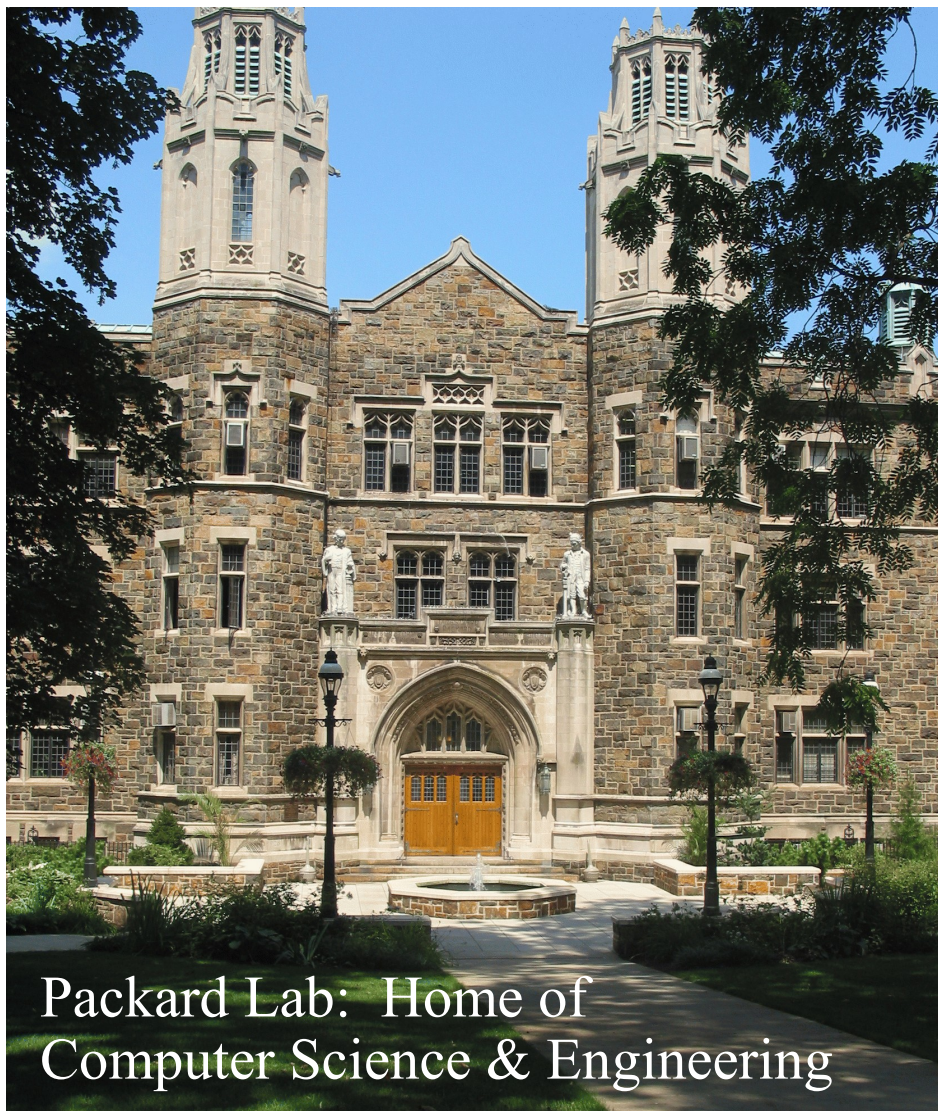
¹ Lehigh University
Bethlehem, PA 18015, USA
lopresti@cse.lehigh.edu

² Rensselaer Polytechnic Institute
Troy, NY 12180, USA
nagy@ecse.rpi.edu

³ University of Nebraska
Lincoln, NE 68588, USA
seth@cse.unl.edu

* Also draws heavily on the work of former RPI students Adnan El-Nasan, Sriharsha Veeramachaneni, and Ashutosh Joshi.

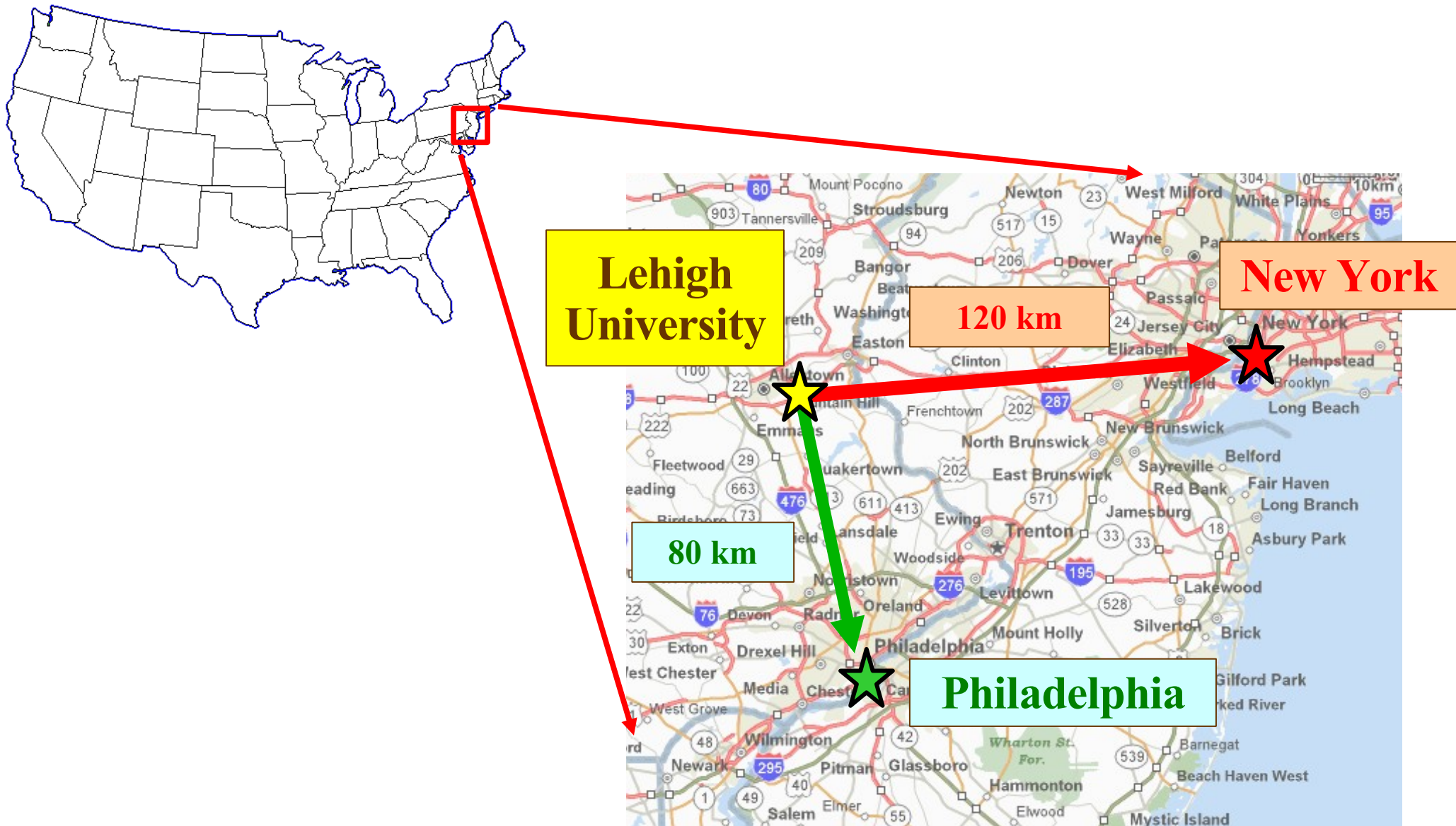
Lehigh University



Key facts about Lehigh:

- A research university founded in 1865.
- Four colleges: Engineering, Arts & Sciences, Business, Education.
- Faculty = 441 full-time.
- Graduate students = 2,064.
- Undergraduates = 4,577.
- Three campuses spread over 1,600 acres (mountain side, wooded).
- Located in northeastern U.S. (about 1.5 hours from New York and Philadelphia, 3 hours from Washington, DC).
- Engineering College ranked in top 20% of Ph.D.-granting schools in U.S.
- University ranked in top 15% of U.S. national universities.

Lehigh University



Talk Outline

- Motivation: segmentation is a hard task in many pattern recognition problems.
- Solution: *Symbolic Indirect Correlation (SIC)*, a new paradigm that exploits 1-D basis of language.
- Simulation results demonstrating efficacy of SIC.
- Detailed error analysis for first stage of computation.
- Maximum likelihood variant for online handwriting.
- Conclusions and future work.

** Important note: SIC has an intuitive appeal. Still, experimental results thus far are only promising, but not conclusive.*

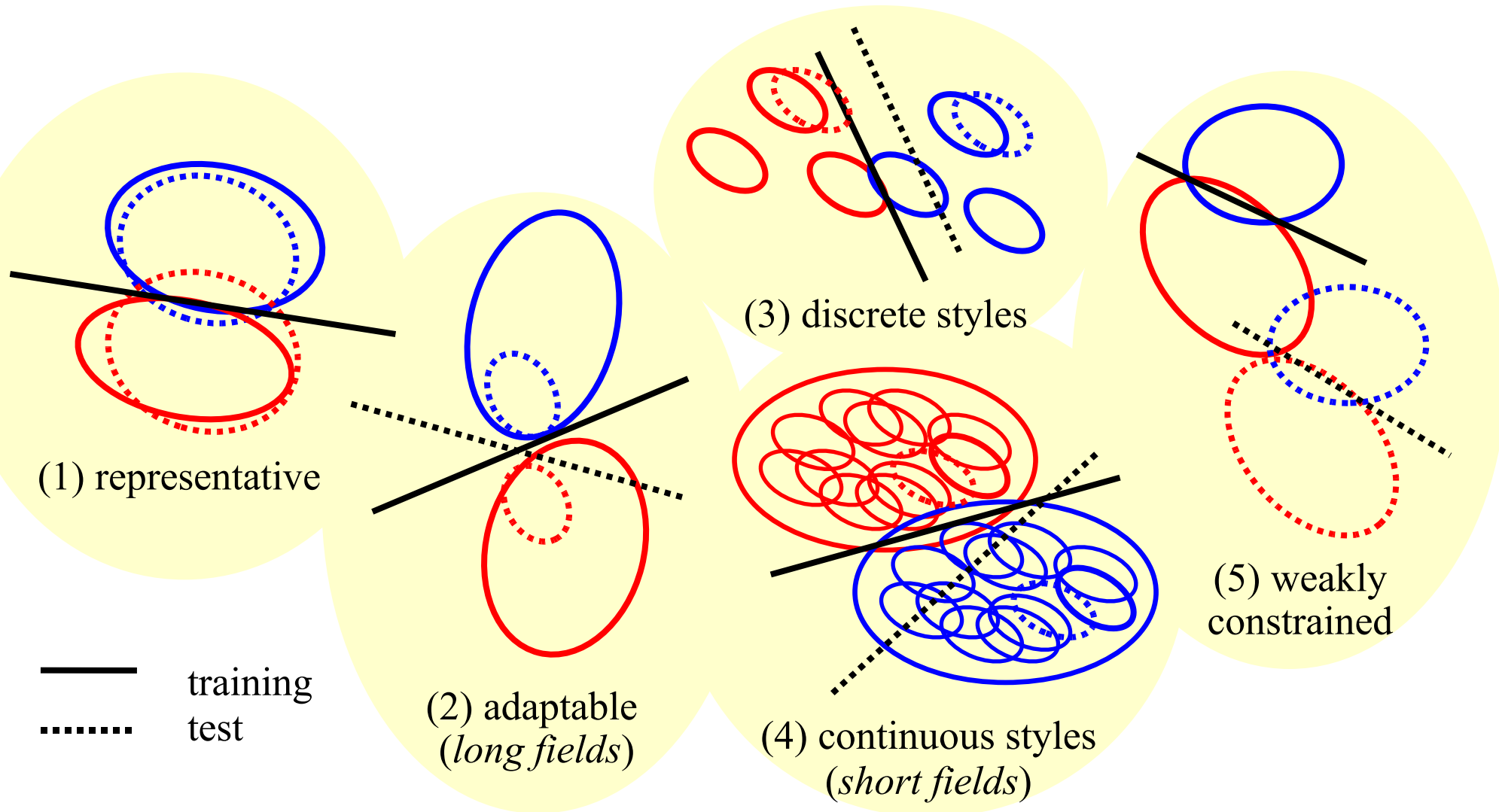
Philosophy

In-house training sets are *never large enough* and *never representative enough*.

- We must augment with samples from actual (real-time, real-world) operations on documents of interest.
- SIC is designed with this goal in mind.
- Initial phase employs supervised learning that places minimal (we hope) demands on the user. Later, the scheme can exploit unsupervised adaptation.

Fundamental Question

How representative is the training set?



Symbolic Indirect Correlation (SIC)

SIC is a new pattern recognition paradigm.

<i>Symbolic</i>	because it exploits the ordering of matches in lexical (symbolic) strings.
<i>Indirect</i>	because it is based on two levels of comparisons.
<i>Correlation</i>	because it can be viewed as making use of sliding windows.

SIC is still a relatively new idea and largely untested.

Key Aspects of SIC

- Matches based on signal subsequences which are typically longer than single characters or phonemes.
- Common distortions in handwriting, camera- and tablet-based OCR (stretching, contraction), as well as speech (time-warping) can be accommodated.
- Independent of medium, feature set, and vocabulary.
- Minimal training – only a reference set as in Nearest Neighbor – thus allowing unsupervised adaptation.
- Lexicon created “on-the-fly” (domain-dependant).

Example Match Graph

Lexicon word



s p l a s h i n e s s

Reference string



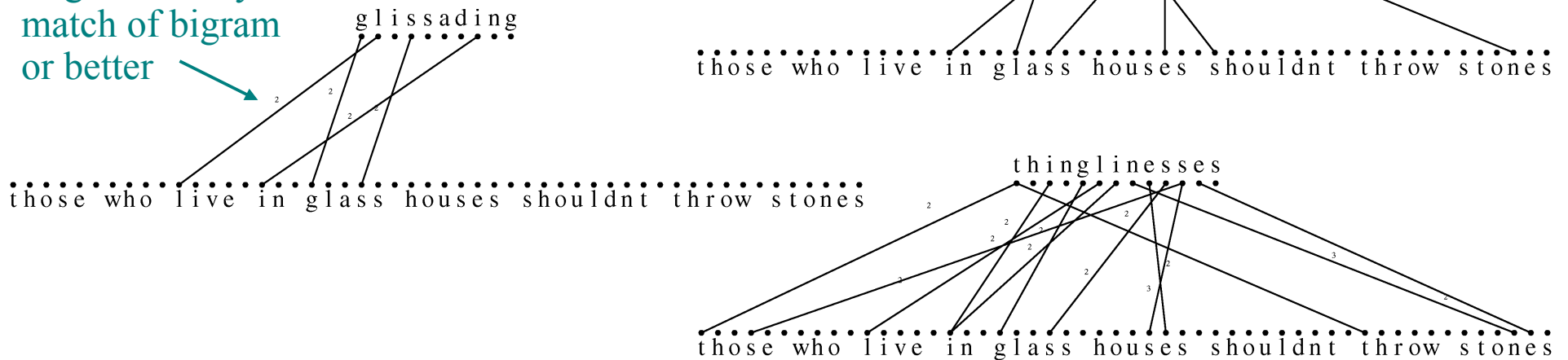
t h o s e w h o l i v e i n g l a s s h o u s e s s h o u l d n t t h r o w s t o n e s

- Bipartite graph: word vs. reference string.
- Note edge for every match of bigram or longer.

SIC Example

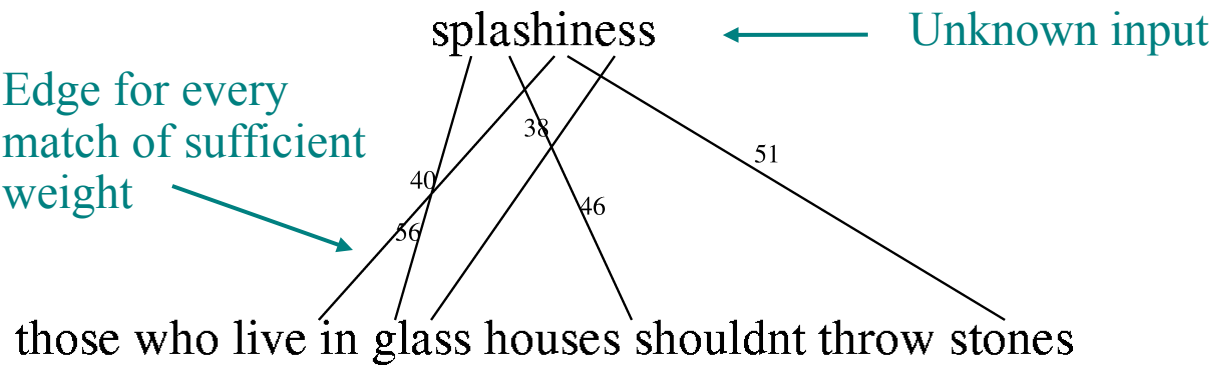
Lexical domain

Edge for every match of bigram or better



Signal domain

Edge for every match of sufficient weight

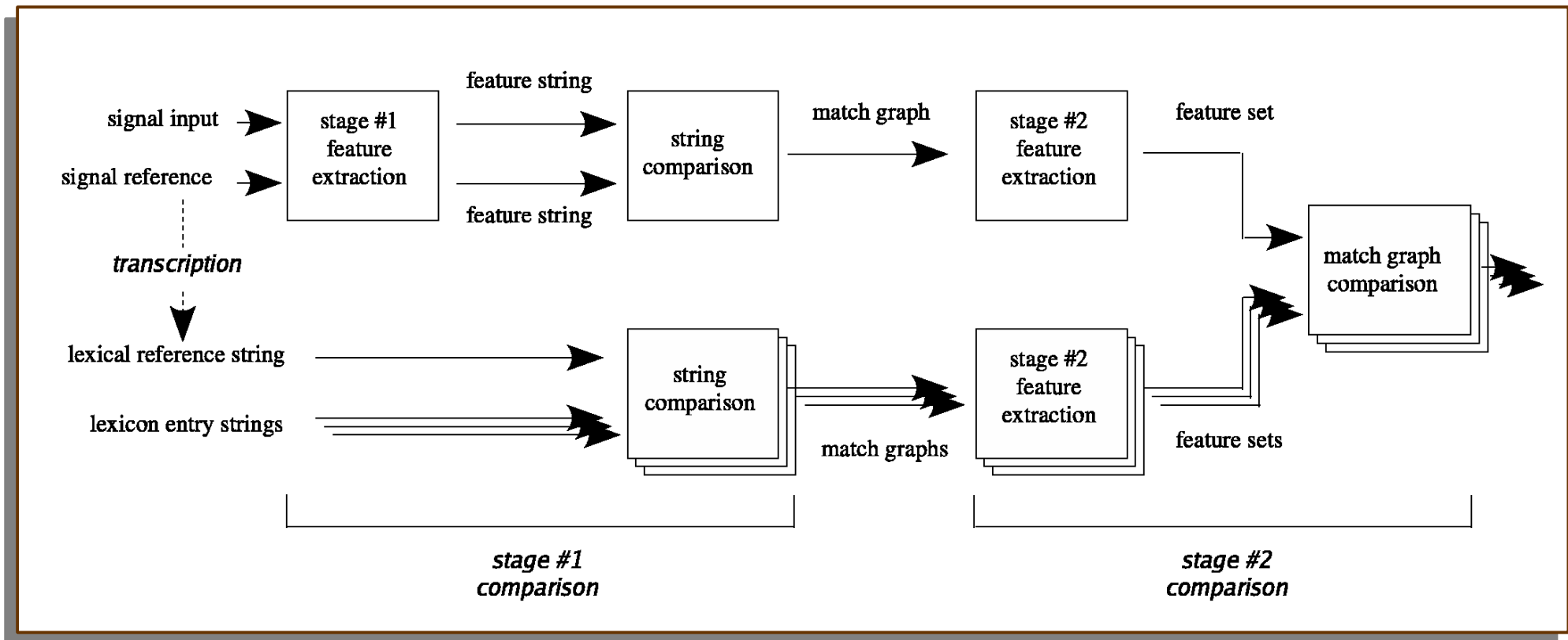


Outline of SIC Approach

- (1) *Lexical Matching.* Match polygrams in every lexicon word against transcription of reference signal (off-line pre-processing).
- (2) *Feature Matching.* Match feature strings derived from query and reference signals.
- (3) *Graph Matching.* Match feature graph (2) against lexical graphs (1) for each word in lexicon.
- (4) *Result.* Output best-matching lexicon word from Step (3) as result.

SIC Overview

SIC is a two-stage matching process:



First stage is efficient, second is more demanding.

Approximate String Matching

SIC uses Smith-Waterman string matching algorithm*:

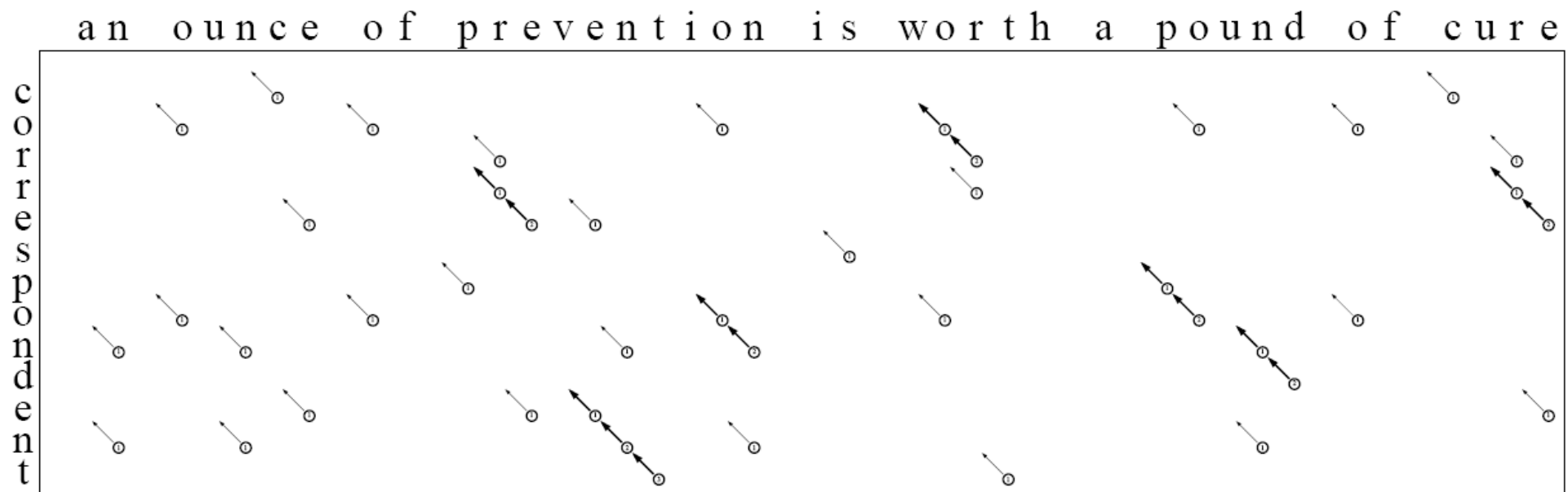
$$\begin{aligned} dist_{0,0} &= 0 \\ dist_{i,0} &= 0 \\ dist_{0,j} &= 0 \end{aligned} \quad dist_{i,j} = \min \begin{cases} 0 \\ dist_{i-1,j} + c_{del}(s_i) \\ dist_{i,j-1} + c_{ins}(t_j) \\ dist_{i-1,j-1} + c_{sub}(s_i, t_j) \end{cases}$$
$$1 \leq i \leq m, 1 \leq j \leq n$$

Note this differs from more widely-known Wagner-Fischer (Needleman-Wunsch) version as it allows for multiple matches that can start and end anywhere.

* “Identification of common molecular sequences,” T. F. Smith and M. S. Waterman, *Journal of Molecular Biology*, vol. 147, pp. 195-197, 1981.

Lexical Distance Matrix Example

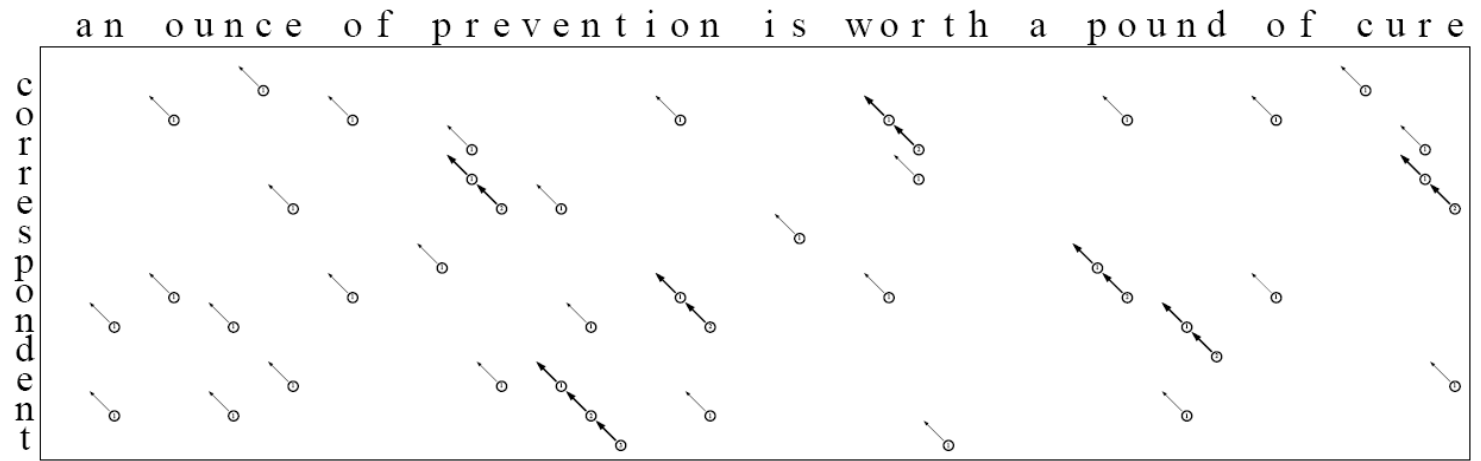
We have developed a series of visualizations for reviewing results of intermediate steps in computation:



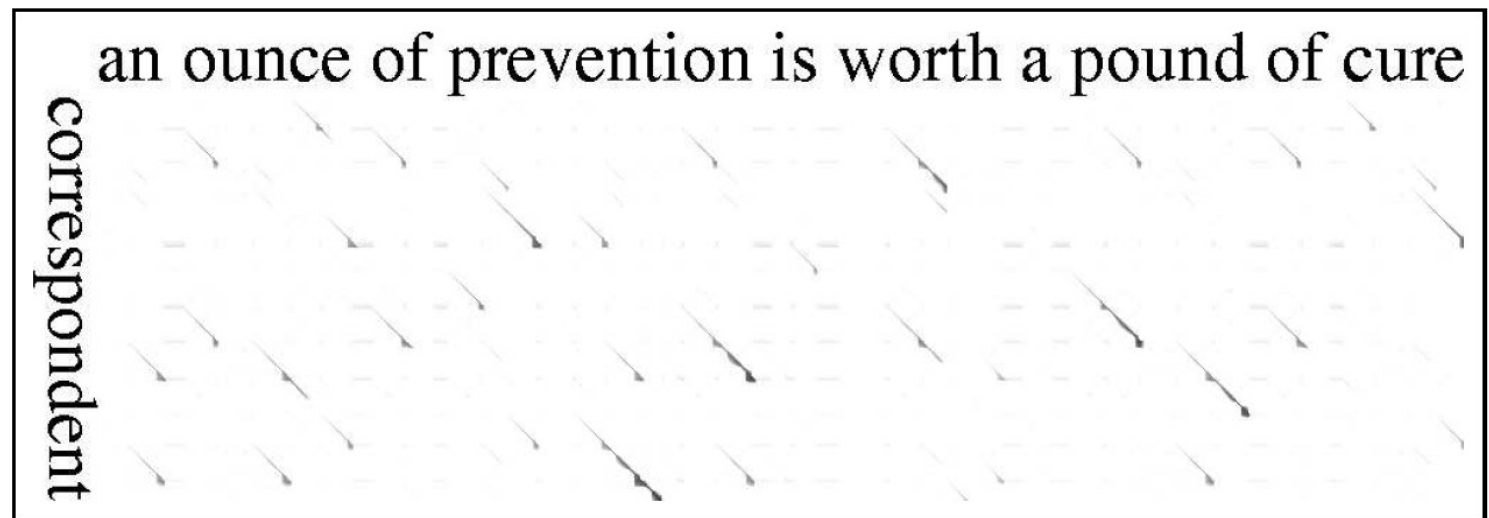
* Note that this graph, like most others in this talk, was generated automatically by running the algorithm in question.

Visualization of Distance Matrices

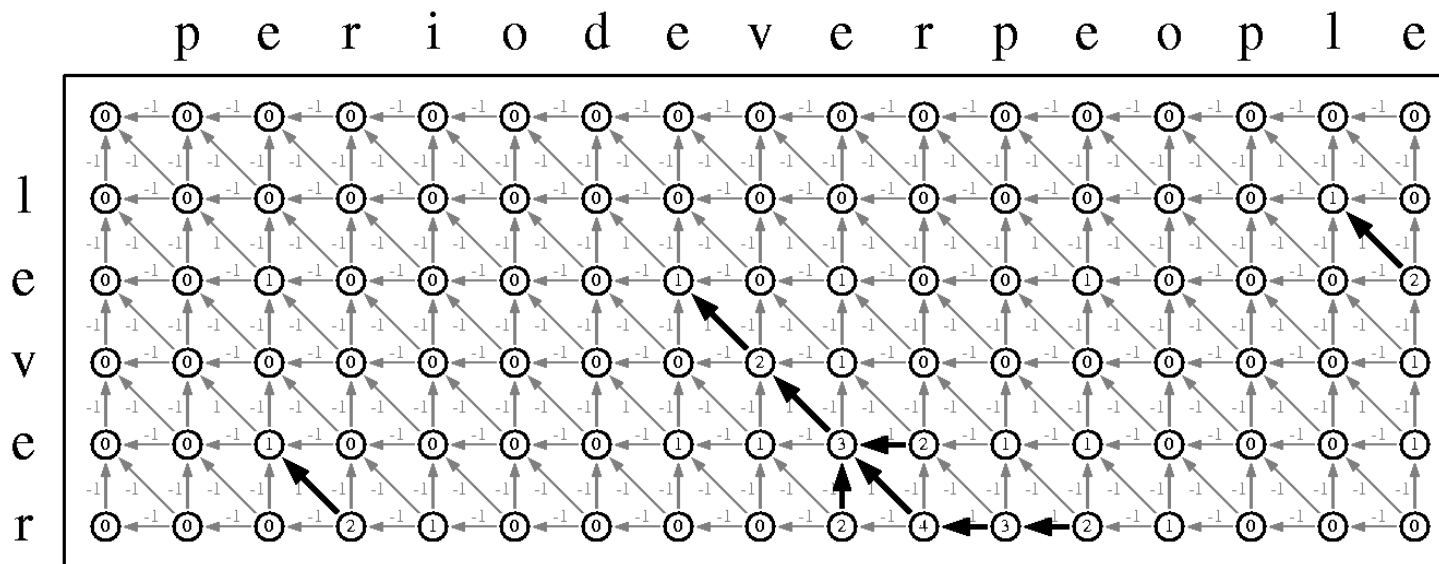
Result of
lexical
comparison:



Result of
signal
comparison:

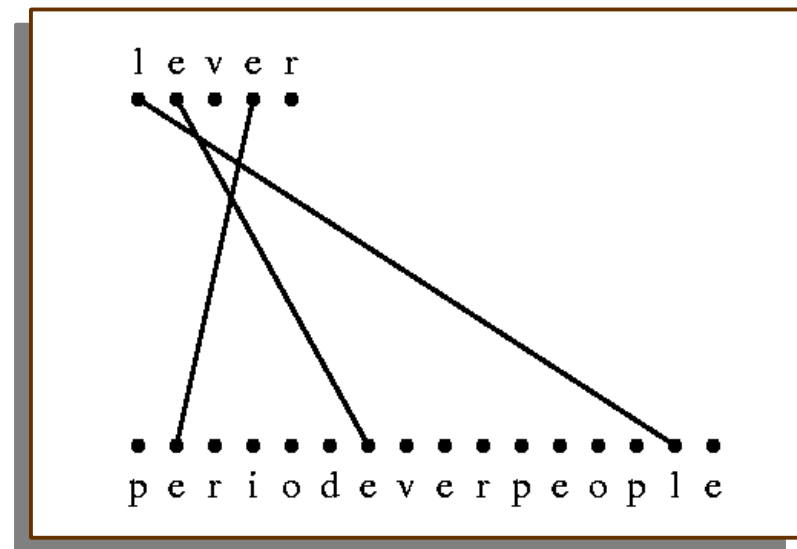


String Matching \Rightarrow Match Graph



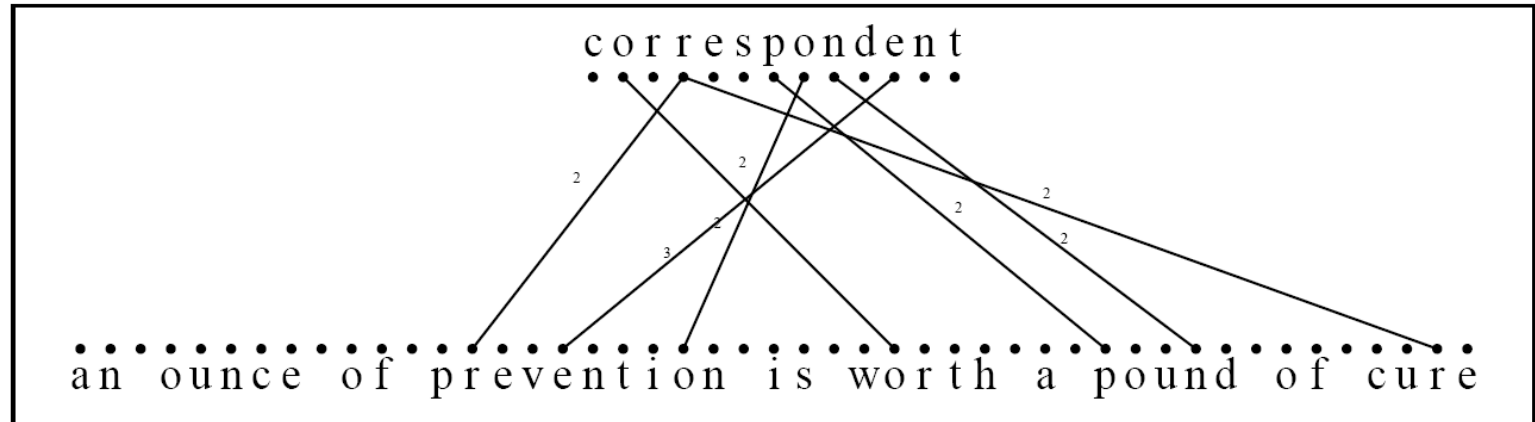
Smith-Waterman
approximate
string matching

Match
graph

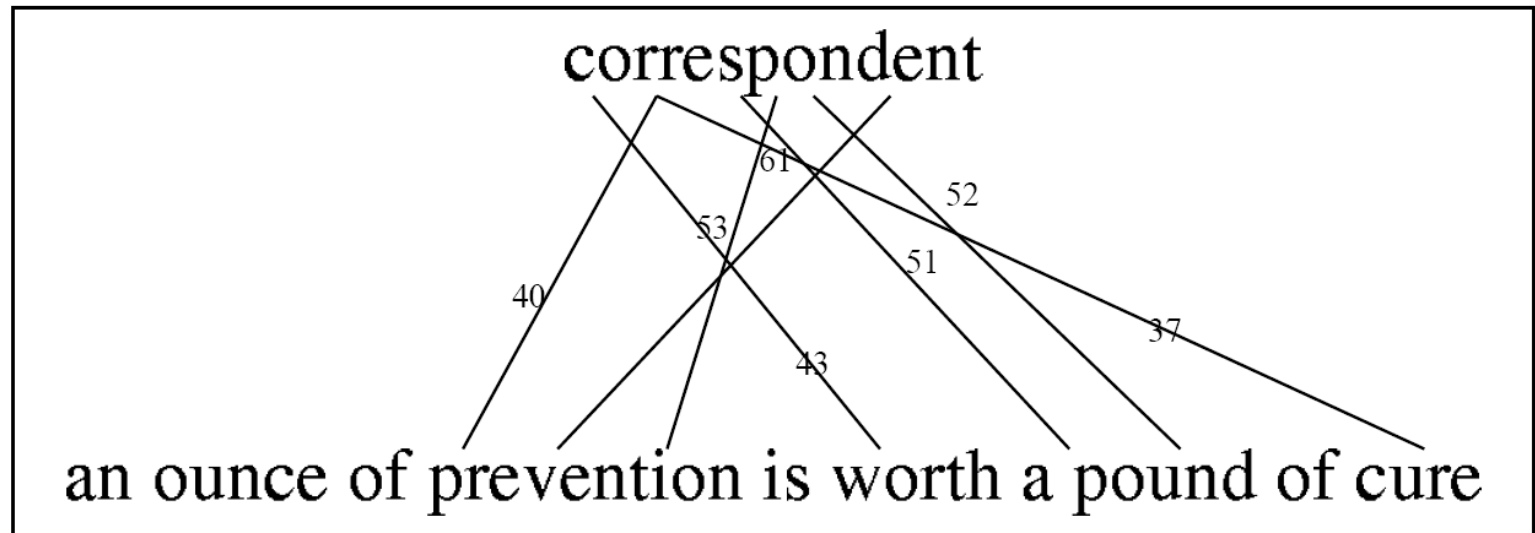


Resulting Match Graphs

Lexical
domain:



Signal
domain:

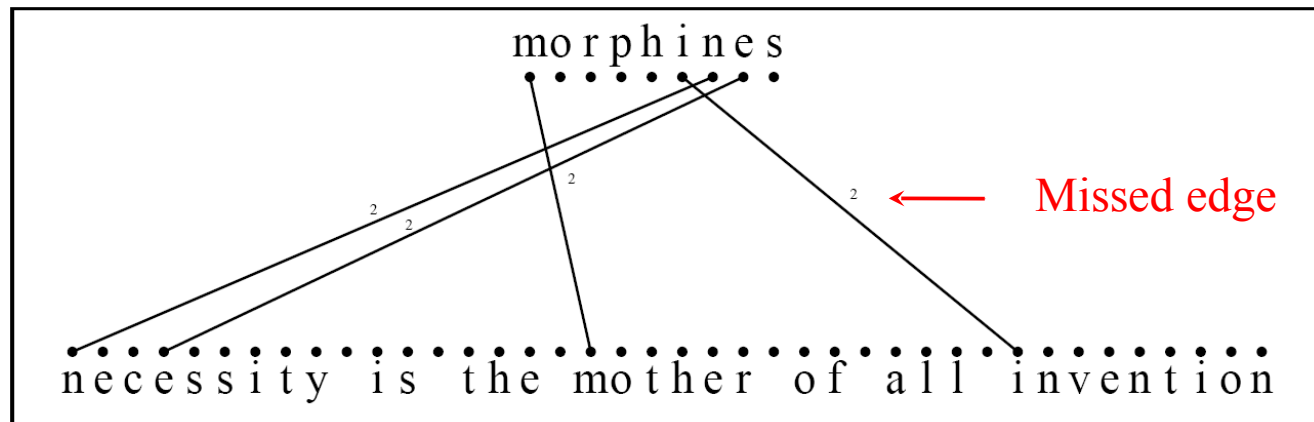


Note that these match graphs correspond perfectly.

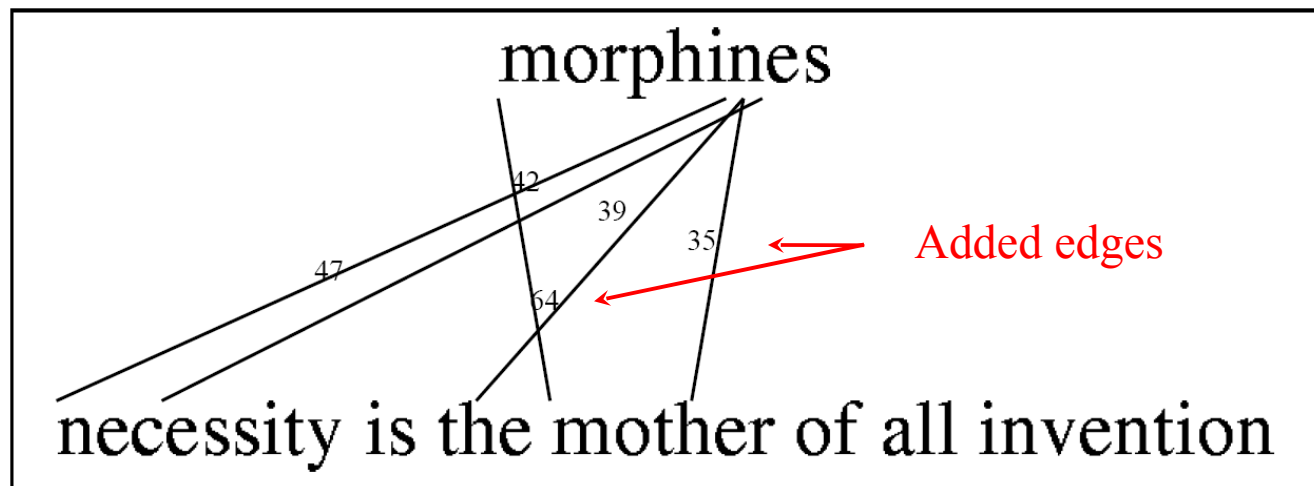
Match Graph Errors

The real world is rarely so cooperative, however.

Lexical
domain:



Signal
domain:



Formal Problem Statement

$$L^* = \operatorname{argmax}_i \{ C_M(M_V(V_x, V_{Ref}), M_L(L_i, L_{Ref})) \}$$

V = feature string

L = lexical string (i ranges over all entries).

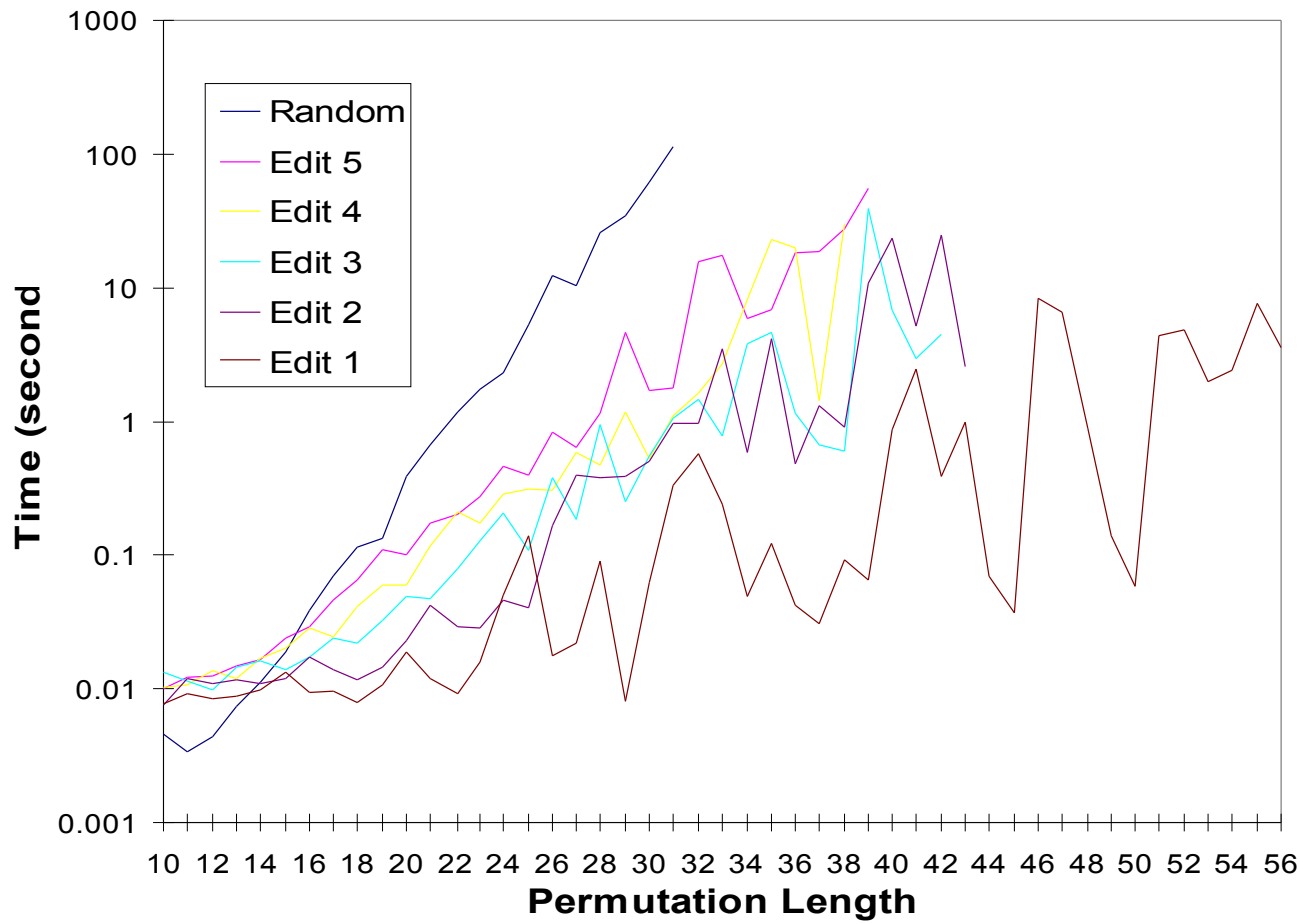
M = string comparison via Smith-Waterman

C_M = meta-comparison on two bipartite graphs

C_M could be a Branch & Bound algorithm that finds longest common subpermutation, for example.

Growth in Runtime of B&B

Branch & Bound time as function of graph similarity:



Bad news ...
more on this
later.

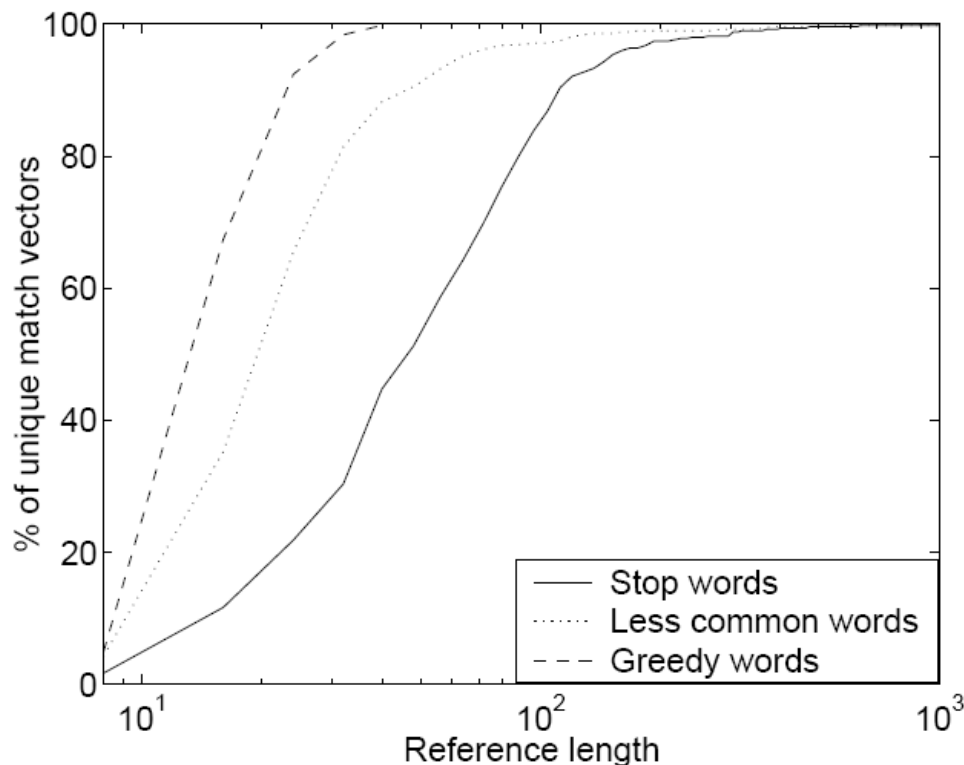
Simulation Experiments

- Select (1) lexicon, and (2) reference string.
- Determine lower bound on error rate: number of unique matches for given-length reference string.
- Choose noise models to reflect likely errors in match graphs (missed edges and/or spurious edges).
- Measure SIC accuracy as function of noise level.

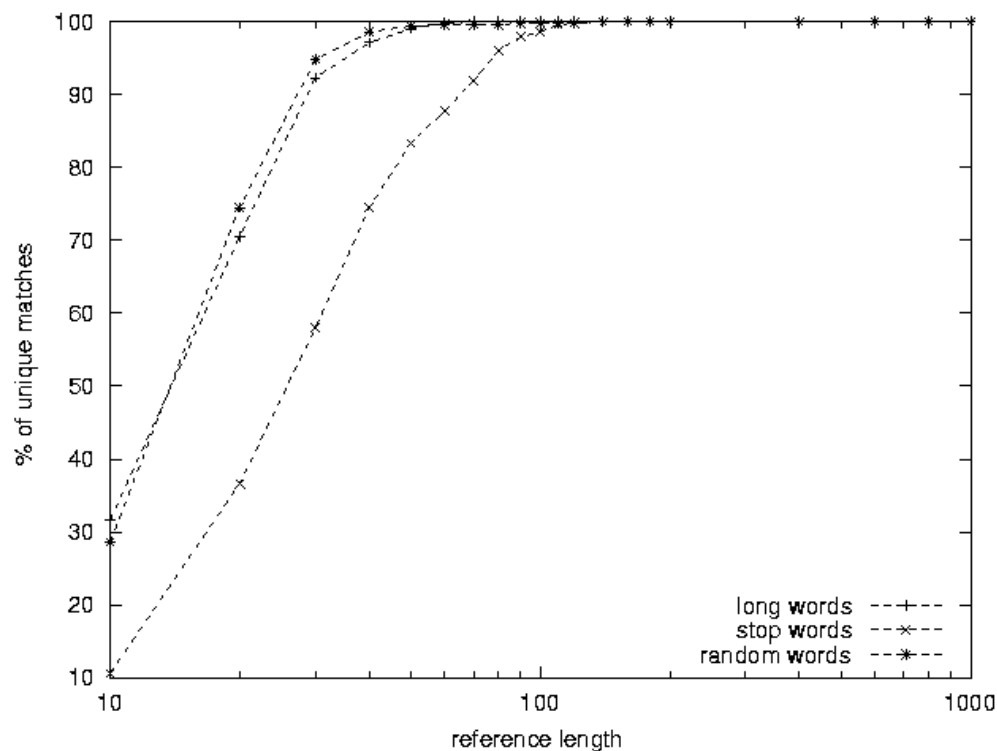
Reference set: select 1,000 words from Brown Corpus in three different ways: short common words (stop words), long uncommon words, random words.

Lower Bound on Error Rate (No Noise)

Bi-gram matching:



Graph matching:



Conclusion: choice of reference set matters, perfect accuracy achievable with ~ 100 reference characters.

Noise Model

- Noise level normalized parameter Q in $[0,1]$ range.
- Symmetric Noise Model:

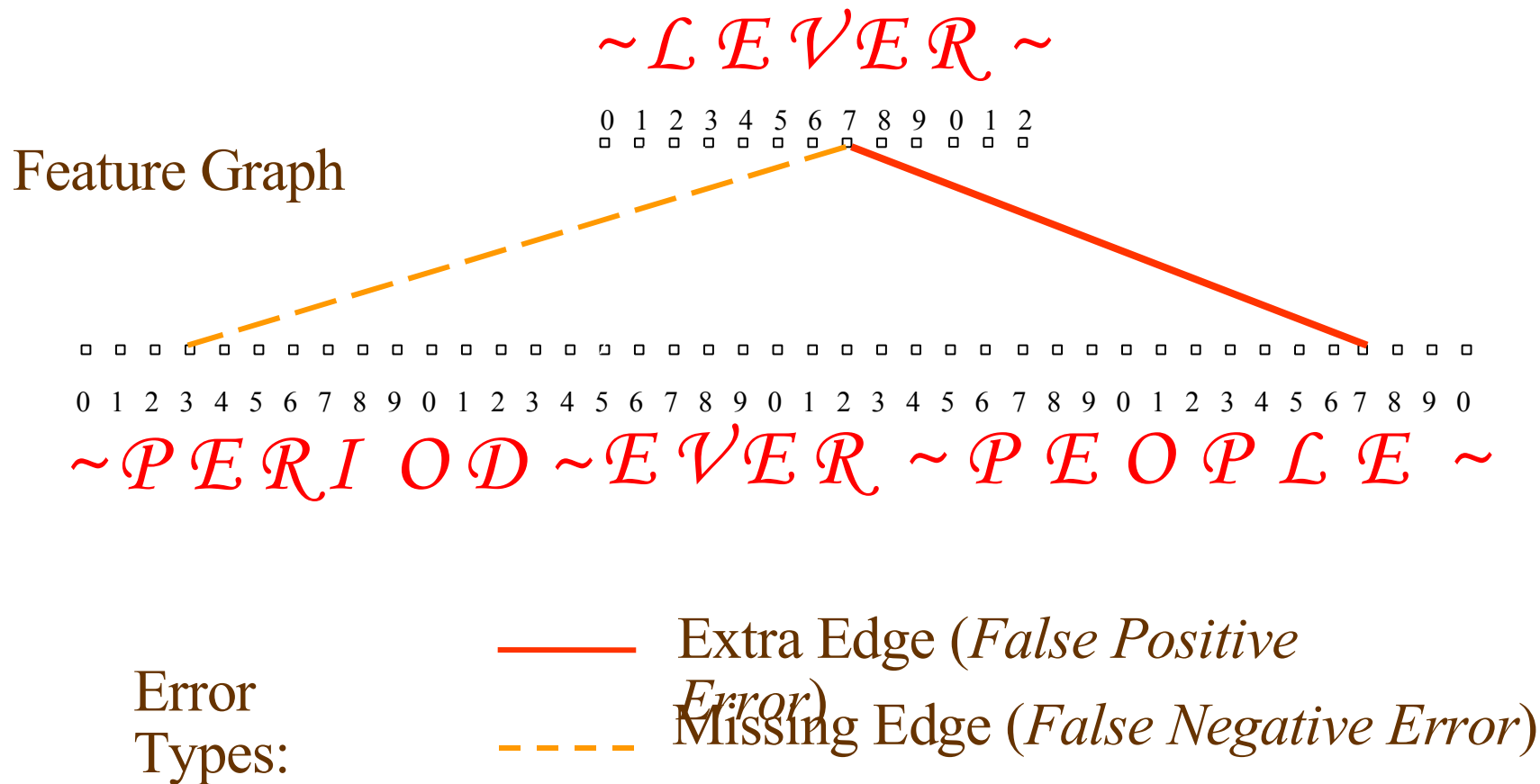
$$Q = p(e|0) + p(e|1) \quad \text{and} \quad p(e|0) = p(e|1)$$

- Weighted Noise Model:

$$Q = w_1 \cdot p(e|1) + (1-w_1) \cdot p(e|0)$$

where w_1 = size of query graph normalized w.r.t. the size of the complete bipartite graph.

Feature Matching Errors



Simulation Details

- (1) Graph size check: eliminate lexical graphs that differ substantially in size from query graph. (Only surviving entries used in later steps.)
- (2) Reference partitioning: 1,000-word reference divided into 100 substrings of 10 words each.
- (3) Matching: query word matched with lexical words for each substring and match scores accumulated.
- (4) Recognition result: top-scoring lexical word.

Simulation Results

Symmetric Noise Model

Q (noise):	0.1	0.2	0.3	0.4	0.6	0.8	1.0
% Rec. Rate:	99.2	99.1	98.7	98.3	96.0	85.5	66.6

Weighted Noise Model

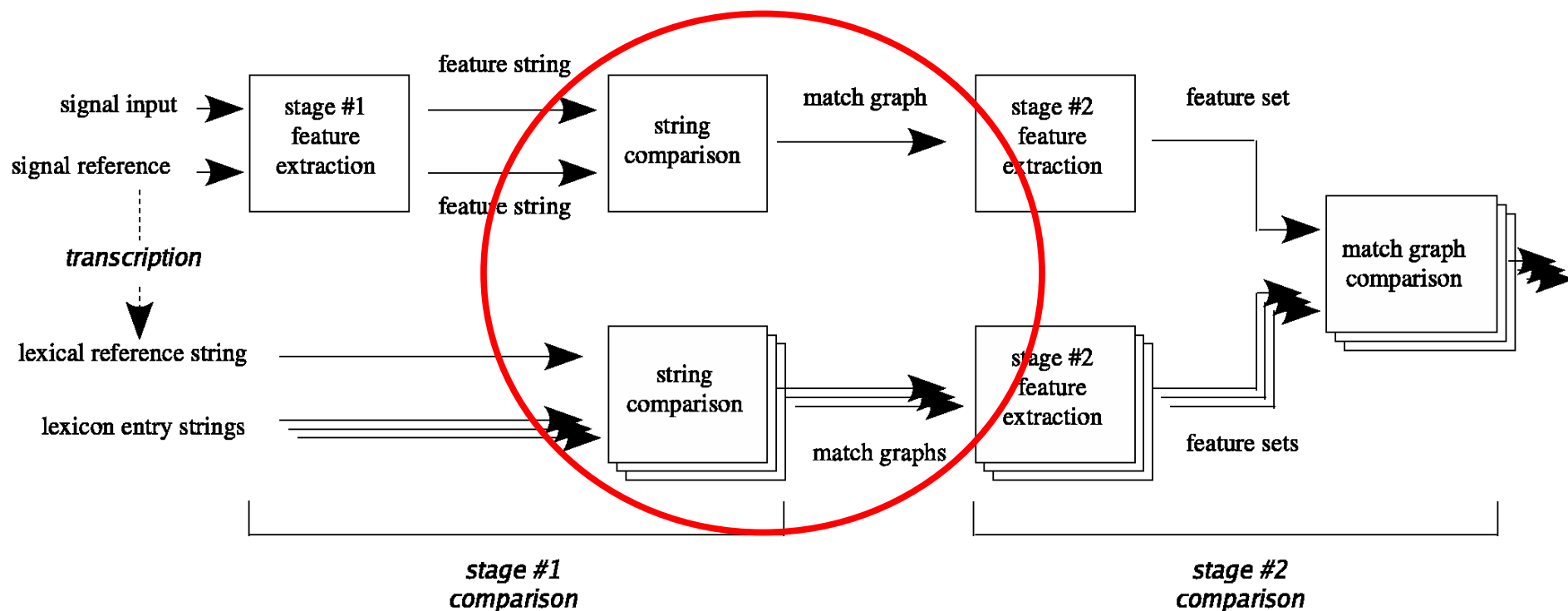
Q:	0.4	0.6	0.8	1.0
% Rec. Rate:	100	100	100	96.0
Reference String Size:	390	500	600	750
$Mean\left(\frac{Top_1 - Top_2}{Top_1}\right)$	0.20	0.16	0.14	0.13

Obvious Research Questions

- What is best strategy for selecting reference set?
- How should costs be set during feature matching?
- Can Branch & Bound algorithm be made efficient enough, or must we turn to other approaches for second stage comparison (e.g., clique algorithms)?

Another Study: Focus on First Stage

SIC performance is impacted by errors at any point:

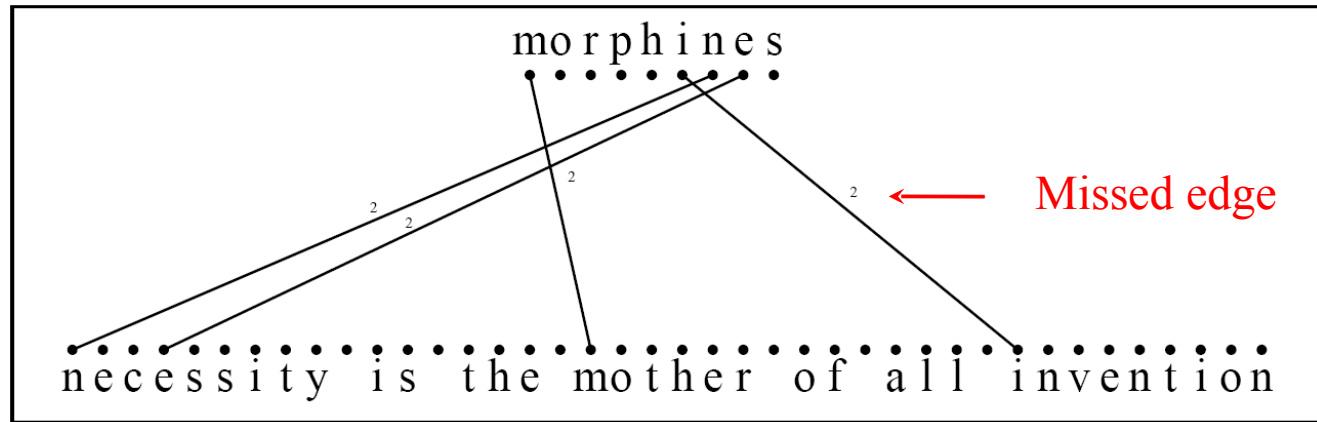


For this study, we bypass final stages of SIC and compare results of match graph generation directly.

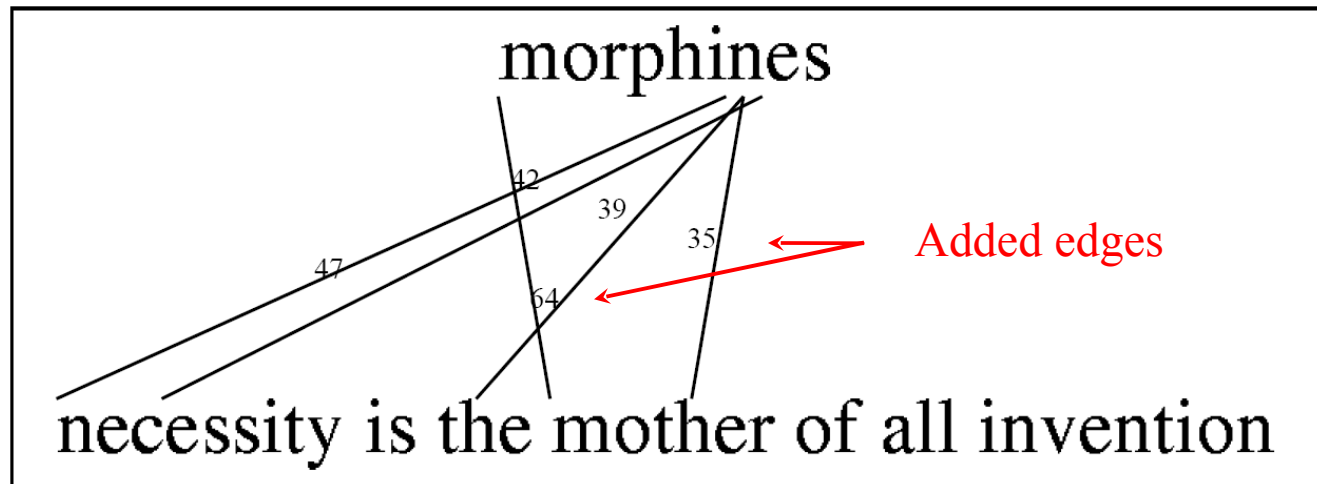
Recall Match Graph Errors

Measure errors in terms of missed and added edges:

Lexical
domain:



Signal
domain:



SIC Evaluation

- Employ synthesized TIF bitmaps of known strings.
- Reference strings = 100 random proverbs.
- Query strings = 100 random words from YAWL*.
- Compare match graphs, count missing/added edges.
- Recall = percentage of lexical match graph edges correctly represented in signal match graph.
- Precision = percentage of signal match graph edges truly present in lexical match graph.
- Total match graphs tested = 10,000 (= 100 × 100).

* “Yet Another Word List,” <http://www.ibiblio.org/pub/linux/libs/>.

Signal Features

To evaluate match graph generation, we performed a pilot study using synthesized images of text strings.

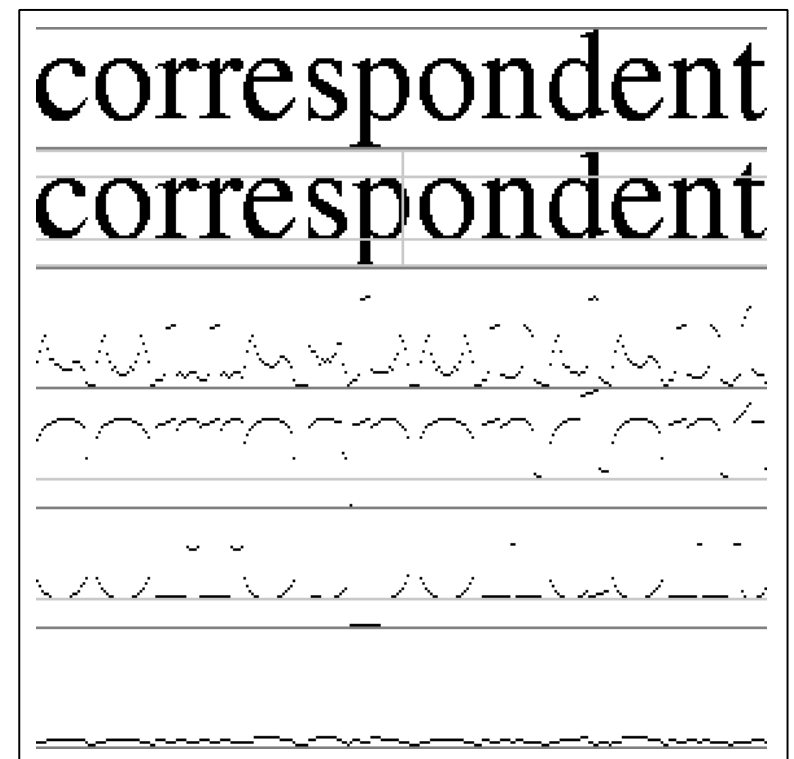
Features are adapted from set used by Manmatha and Rath for offline handwriting.*

Black pixel density →

Upper text contour →

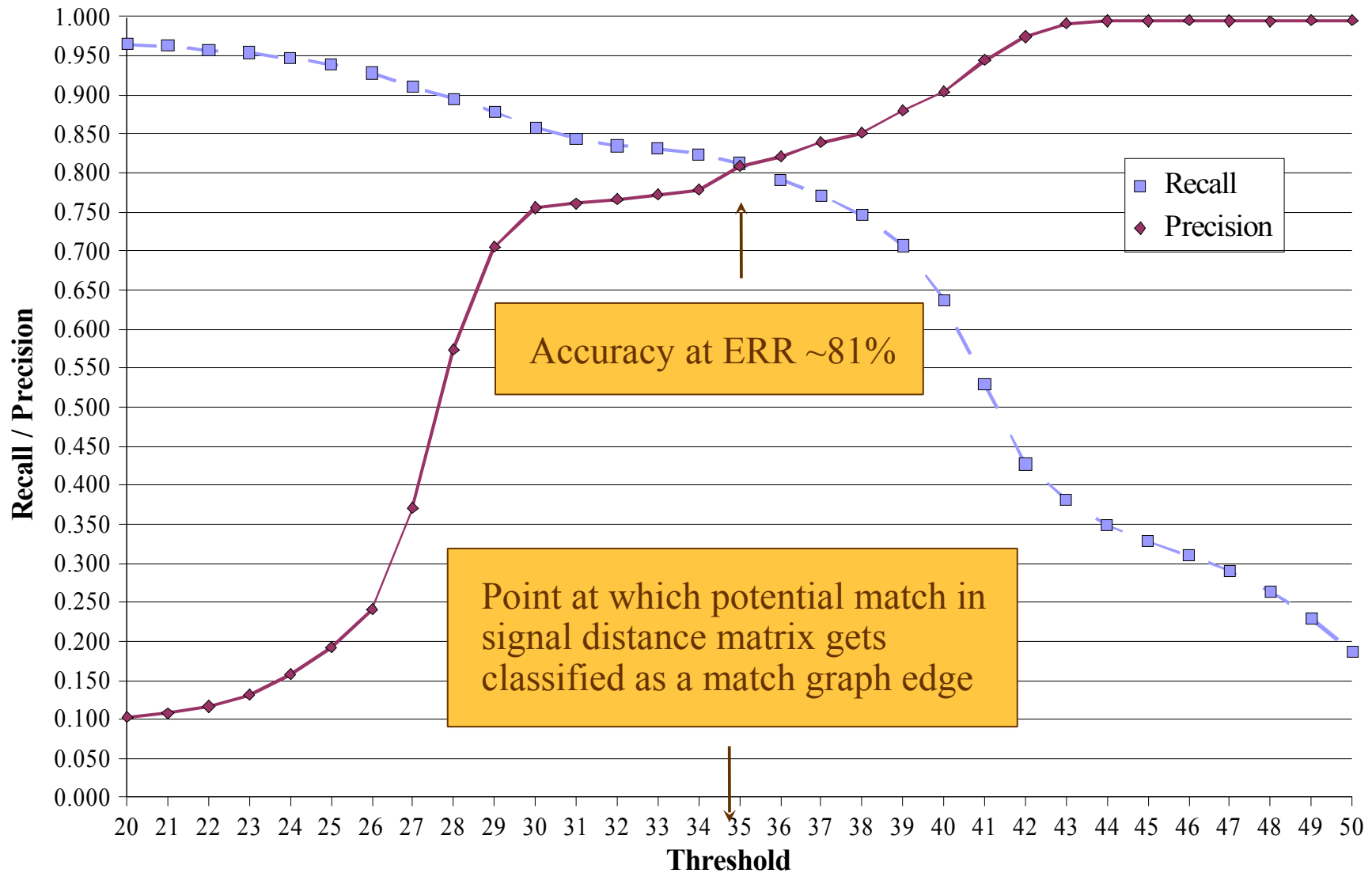
Lower text contour →

0-1 transitions →



* "Indexing Handwritten Historical Documents – Recent Progress," R. Manmatha and T. Rath, *Proceedings of the Symposium on Document Image Understanding*, pp. 195-197, 2003.

SIC Results



Most Frequent Edge Effects

Tabulate various effects we saw at optimal threshold:

Rank	Correct Edges		Missed Edges		Spurious Edges	
	Count	Pattern(s)	Count	Pattern(s)	Count	Pattern(s)
1	1,056	“es”	331	“it”, “es”	37	“nes” ↔ “her”
2	971	“th”	219	“st”	34	“me” ↔ “ma”
3	957	“in”	209	“ti”	32	“nes” ↔ “he m”
4	827	“er”	199	“li”	30	“mo” ↔ “ma”
5	537	“re”	178	“is”	25	“mo” ↔ “me”
6	471	“at”	141	“re”	24	“me” ↔ “mo”
7	426	“on”	124	“ll”	22	“nes” ↔ “he b”
8	394	“an”	95	“te”	21	“nes” ↔ “he h”
9	391	“he”	82	“er”, “at”	19	“nes” ↔ “he d”, “owe” ↔ “wi”, “mo” ↔ “mi”
10	380	“nt”	57	“en”	18	“me” ↔ “mi”

- Missed edges due largely to thin characters (e.g., i).
- Spurious edges due to feature similarity, including character prefixes and suffixes (e.g., h ↔ n).

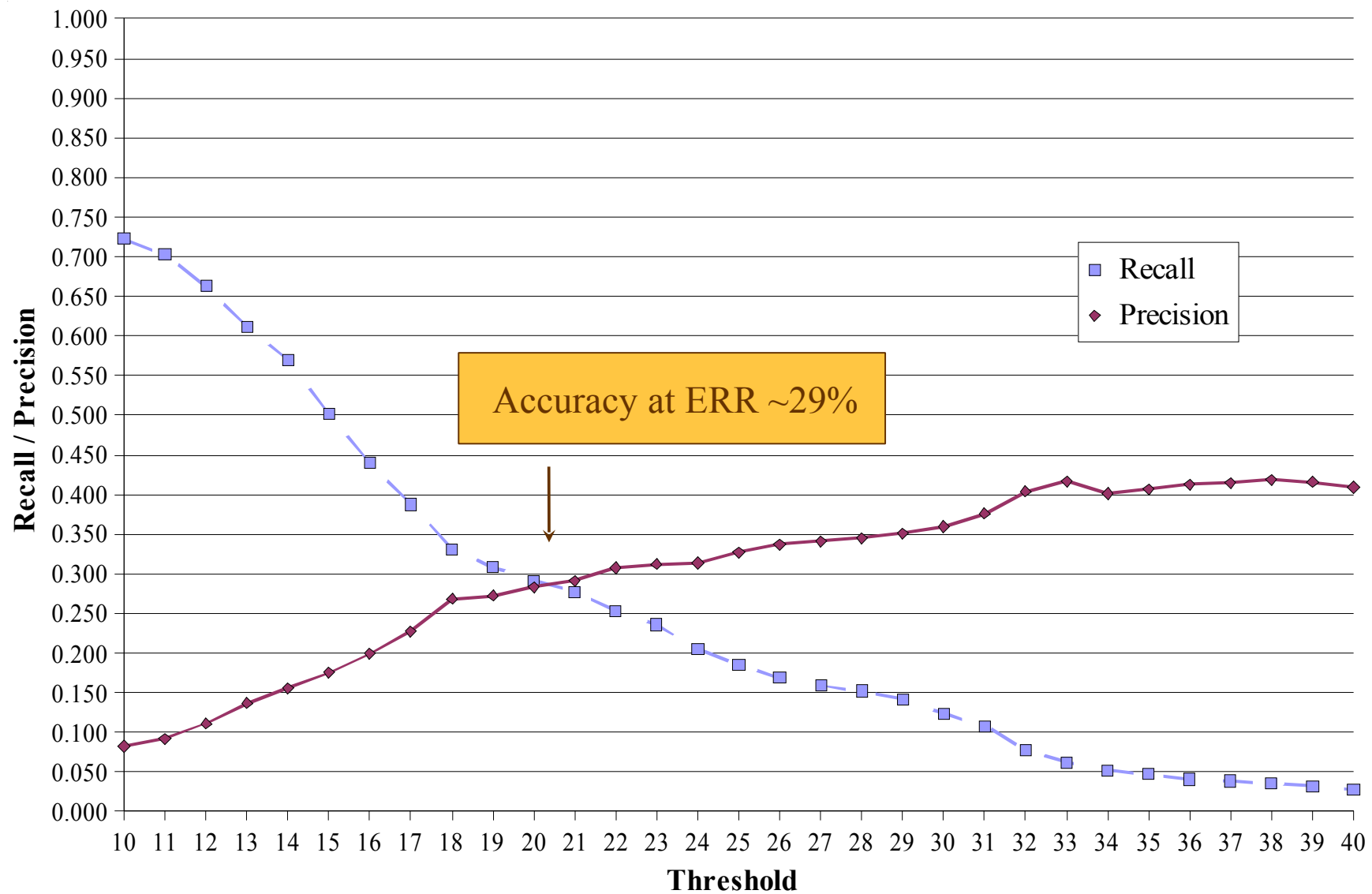
More Challenging Evaluation

- SIC proposed for handling hard-to-segment inputs.
- Repeat exact same experiment, only this time using highly condensed text strings.

correspondent

correspondent

SIC Results (Condensed Text)



Conclusions For This Study

- Smith-Waterman approach appears to be right model for building match graphs.
- Current problems lie with feature representation. Some issues may be challenging to surmount (e.g., suffix of “h” will always resemble suffix of “n”).
- On the other hand, final stage of SIC has ability to overcome a certain number of errors.
- Future work includes exploring connection between match graph errors and overall SIC error rate, as well as extending evaluation to real handwriting and scanned text inputs (appropriately ground-truthed).

Maximum-Likelihood Approach to SIC

Second stage of SIC is computationally demanding, so we explored an alternative formulation:

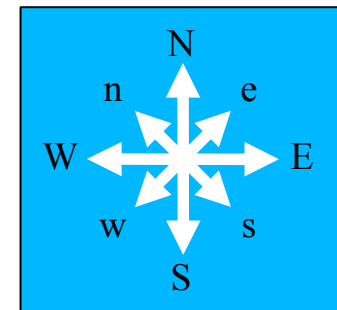
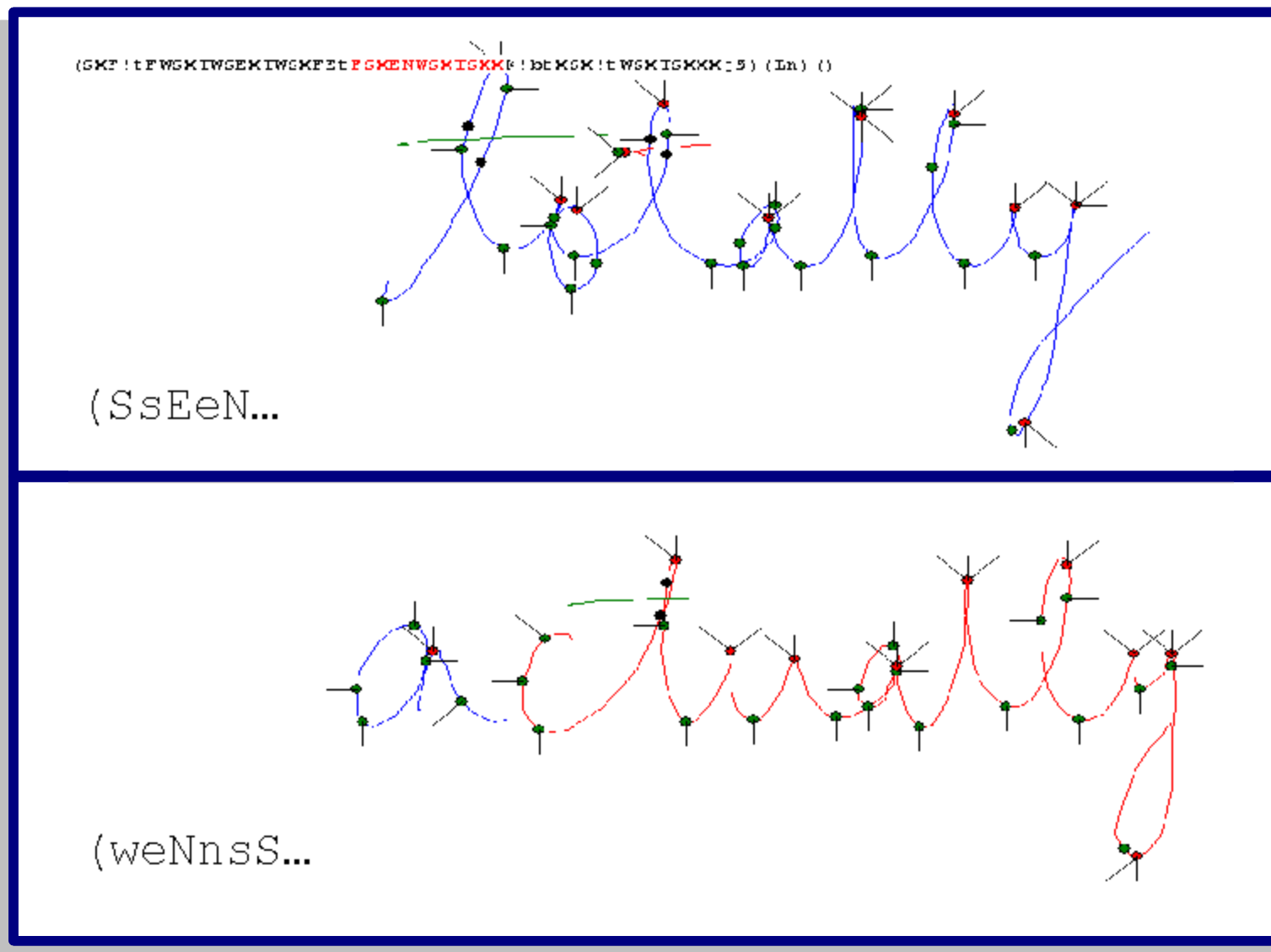
1. *Lexical Matching*. Match polygrams as before.
2. *Feature Matching*. Match feature strings as before
3. *Maximum Likelihood*. Under a class hypothesis, find best assignment of polygrams to feature segments such that temporal ordering between polygrams is same as between feature segments.
4. *Result*. Query is assigned to class whose assignment has maximum likelihood.

Online Handwriting Recognition

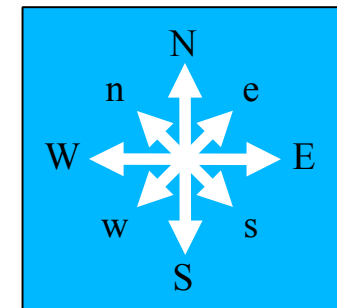
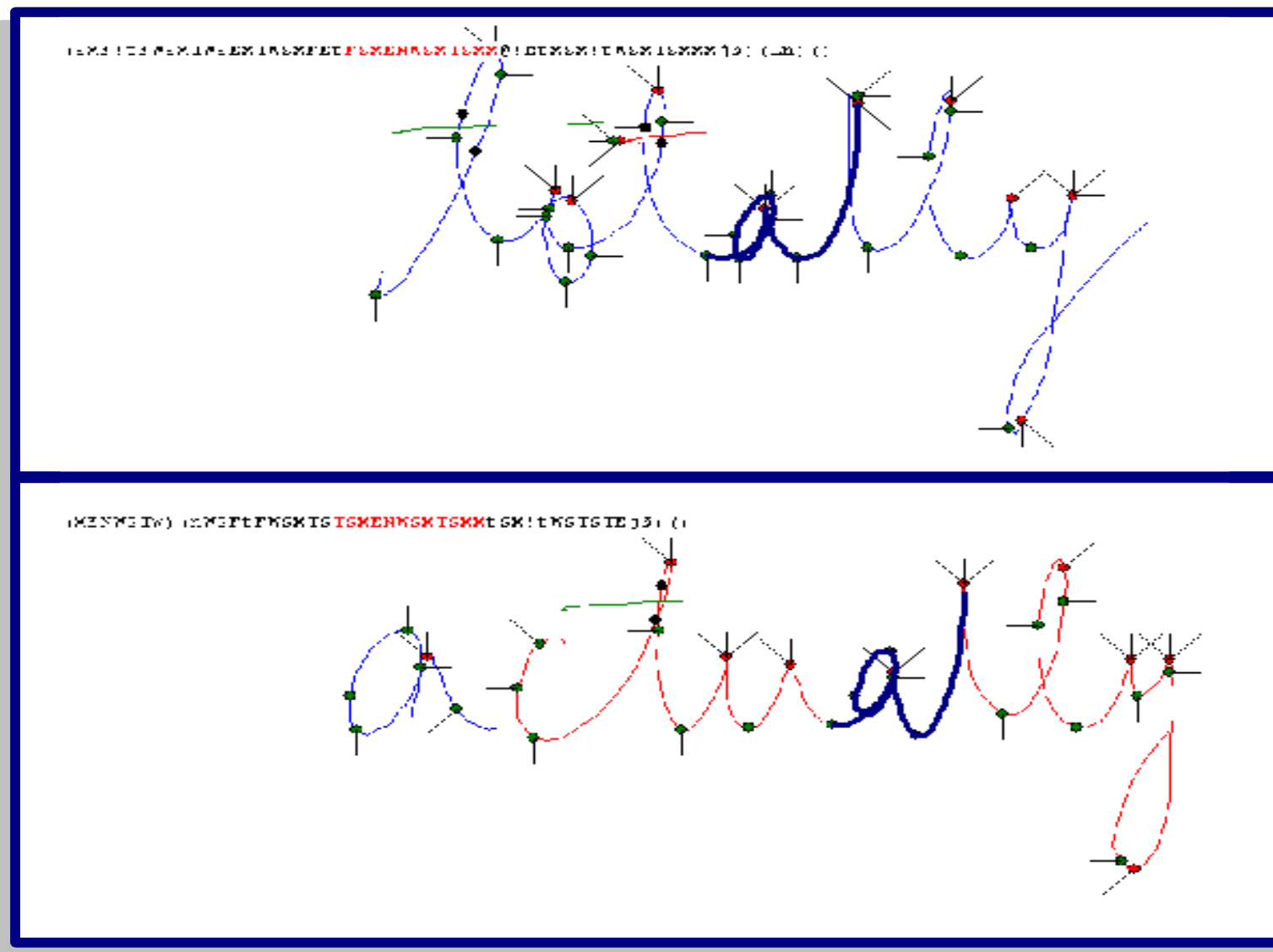
Examine SIC in context of handwriting recognition:

- Ink sampling rate of 133Hz.
- Handwritten words represented as time-ordered sequence of local maxima in 8 directions.
- Features also include zoning information: ascender, body, descender.
- Ink is dehooked & corrected for baseline shift.
- Words normalized for equal-height body zone.

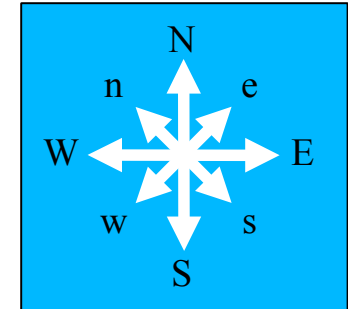
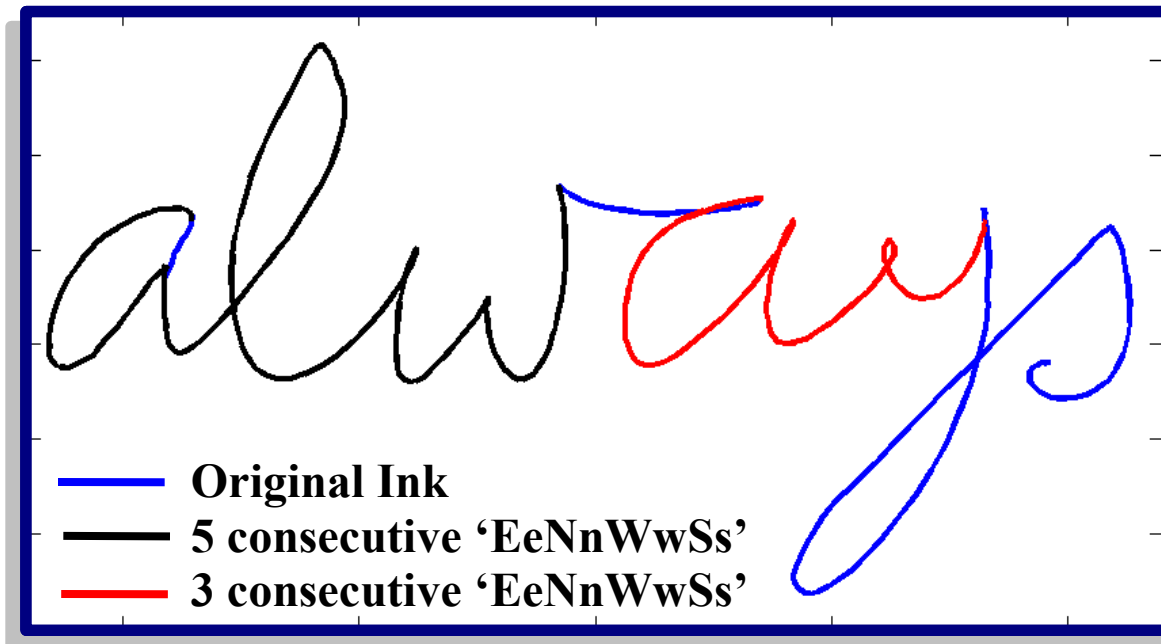
Handwriting Features



Handwriting Features

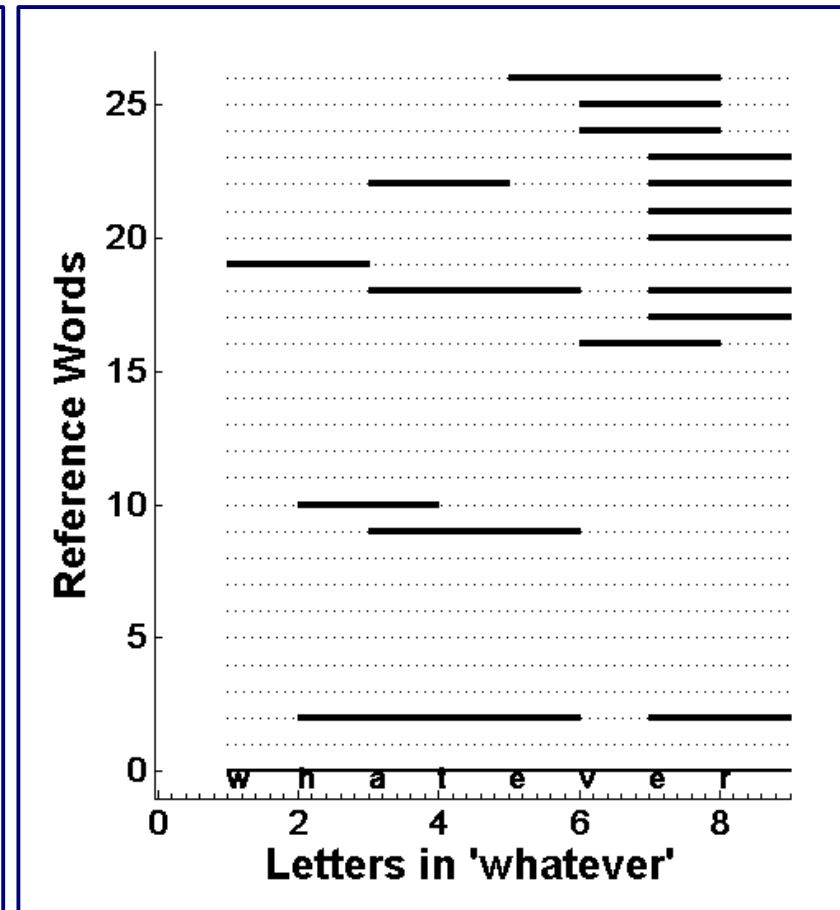
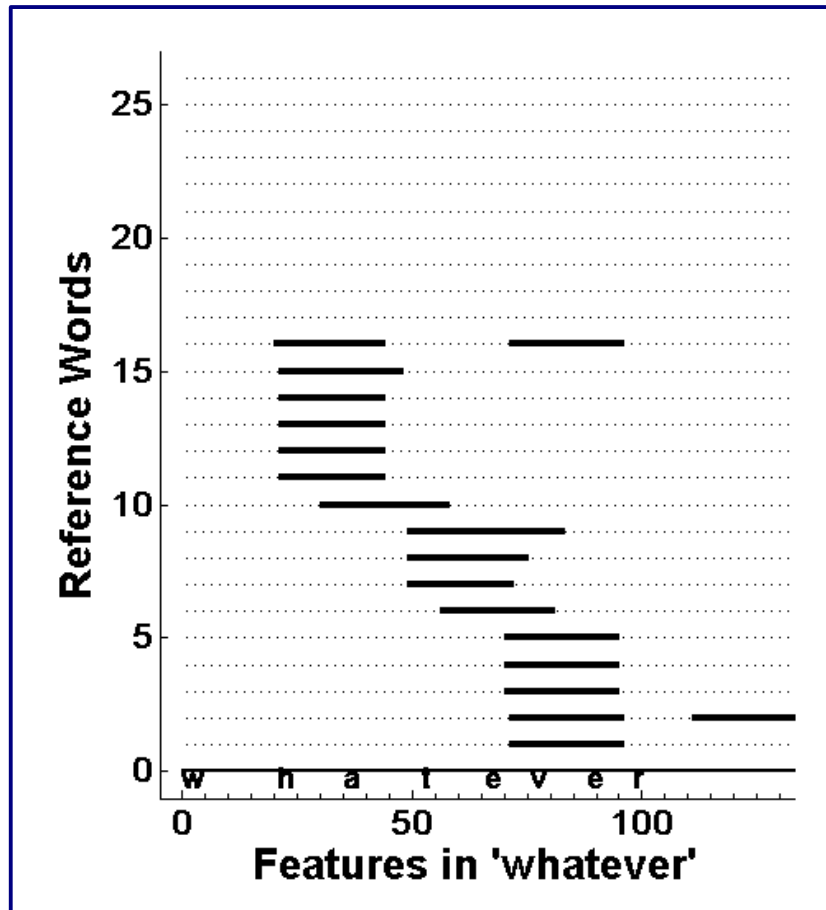


Discrimination Capabilities



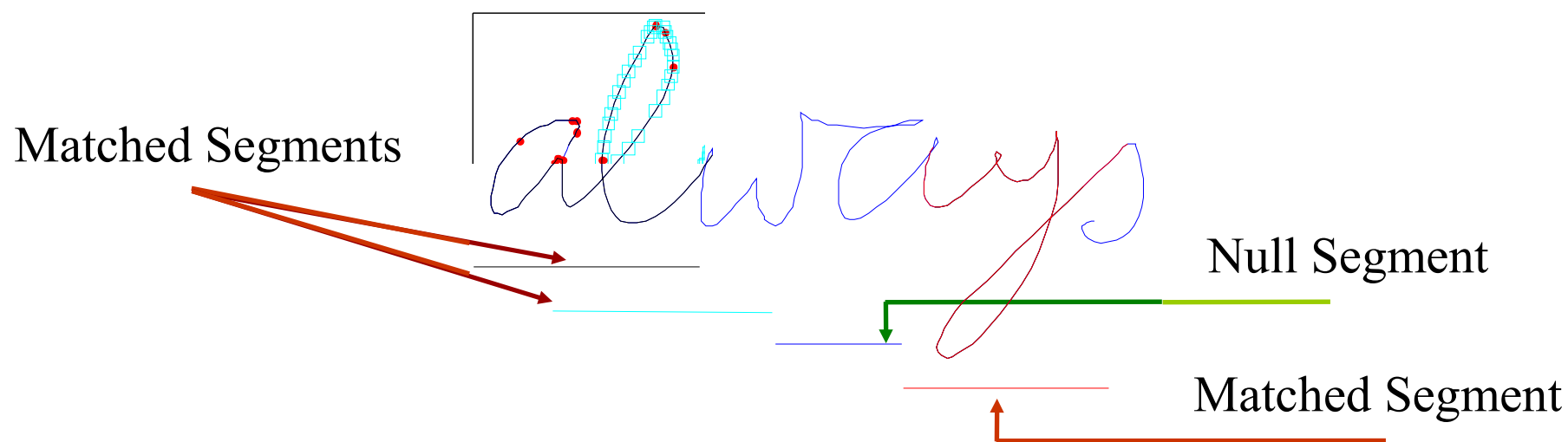
- Note 8 total instances of pattern 'EeNnWwSs' in above figure: 5 consecutive (in black) and another 3 consecutive (in red).
- Average of 9.2 loops per word and 1.4 loops per letter.
- Average of 5.2 instances of 'EeNnWwSs' per word.
- Average of 87.8 features per word (ignoring pen-ups and -downs).

Clustered Feature Matches



Correlation Coefficient between lexical and feature matches is only ~ 0.11 .

Partial Feature Matches: Segments

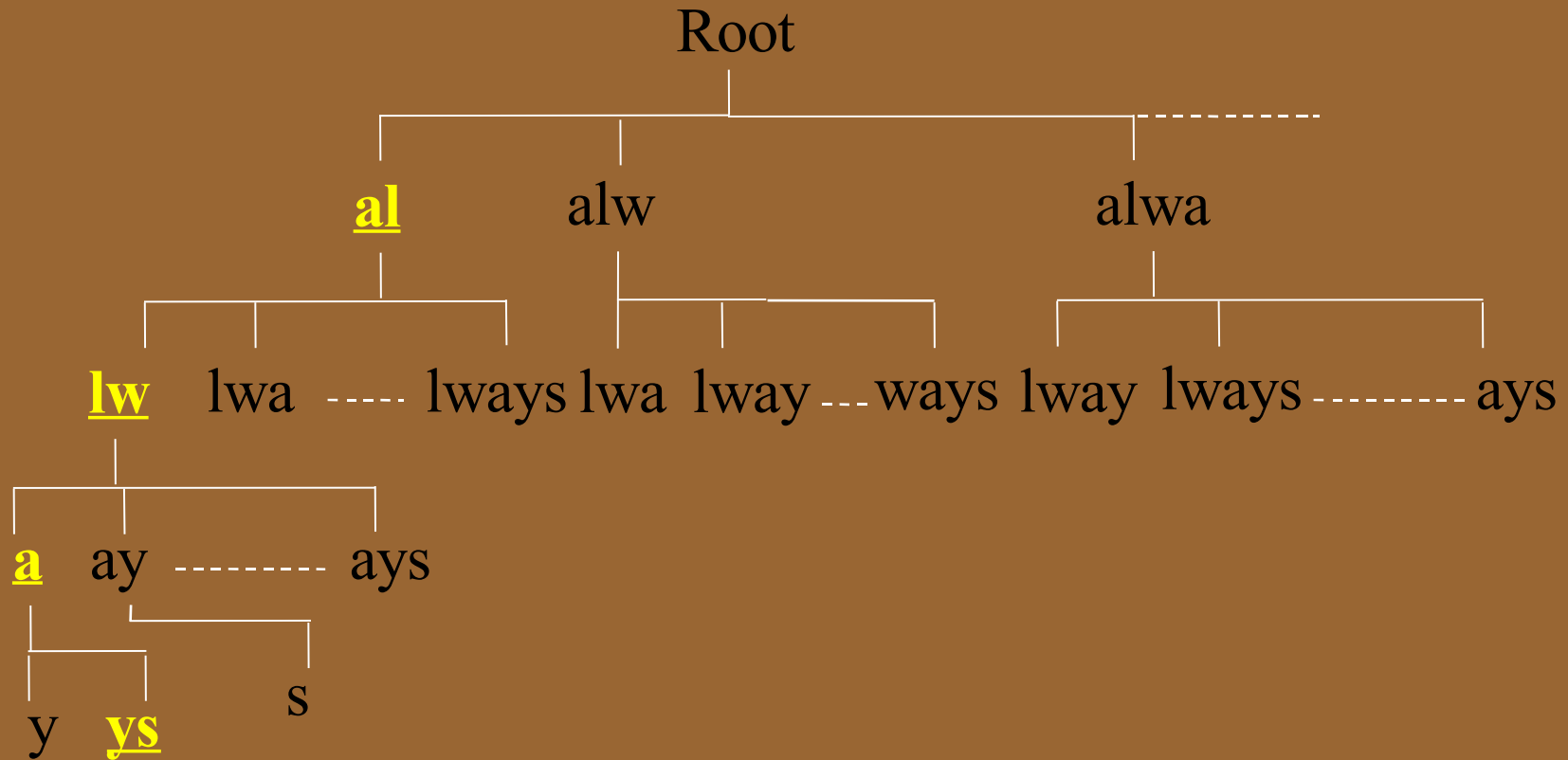
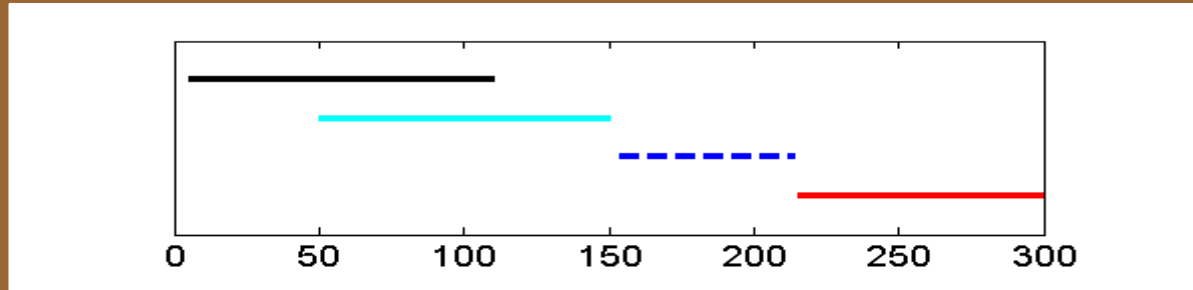


walked plays anyway

Temporal Relationships

Relation	Represent	Complement	Pictorial
X simultaneous Y	$X = Y$	$X = Y$	XXX YYY
X before Y	$X < Y$	$X > Y$	XXX---YYY
X meets Y	$X m Y$	$X mi Y$	XXXYYY
X during Y	$X d Y$	$X di Y$	--XXX-- YYYYYYY
X overlaps Y	$X o Y$	$X oi Y$	XXX-- -YYY
X starts Y	$X s Y$	$X si Y$	XXX-- YYYYYY
X finishes Y	$X f Y$	$X fi Y$	---XXX YYYYYYY

Assignments



Segment Likelihood

- Relation between segment and reference word:
 - Valid match ($p_{1|1}$)
 - Spurious match ($p_{1|0}$)
 - Missed match ($p_{0|1}$)
 - Correct rejection ($p_{0|0}$)
- Conditioned on polygram assignment to that segment.
- Reference words are independent.
- Segment likelihood:

$$P[Seg | Pgm] = \prod_{m=1}^{NR} p_{i|j}$$

$i, j \in \{0,1\}$; $NR =$ Number of reference words

Joint Likelihood

- Assume matched segments are independent.
- Null segments are dependent on matched segments, but are very rare for long reference sets.
- Assignment Score = Joint Likelihood:

$$P[AllSeg | Assignment] = \prod_{i=1}^{NS} P [Seg_i | Pgm_i]$$

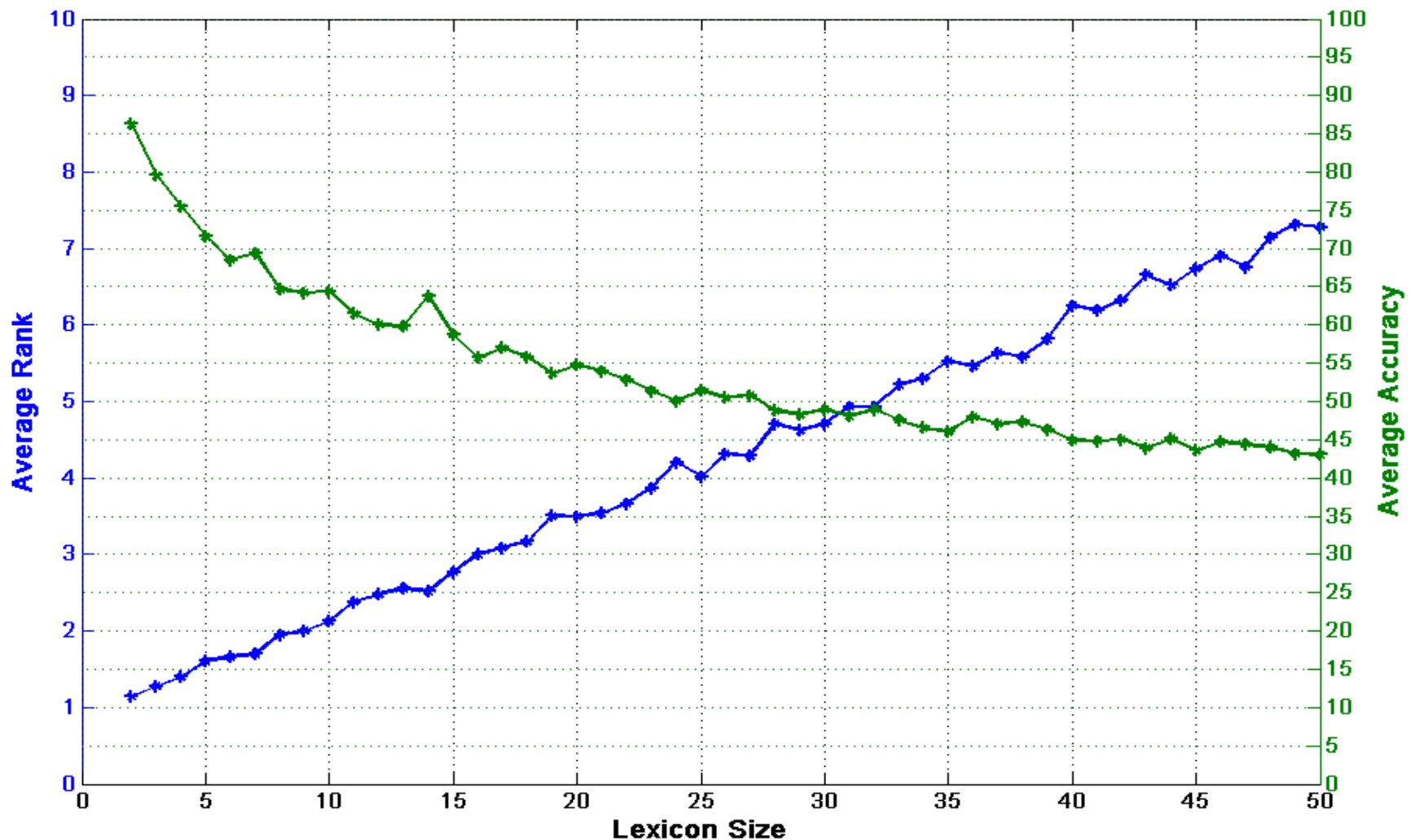
$NS =$ Number of segments

- Set probabilities by comparing reference set to itself.

Experiment: Online English HWX

- From Brown Corpus: use 1,000 most-frequent words that are at least 5 characters long.
- A single handwritten sample per class.
- Query chosen randomly, with remaining 999 words serving as reference set.
- Select a random lexicon of indicated size which includes query. Repeat 50x for each size.
- Note: query does not appear in training set!


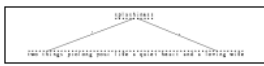
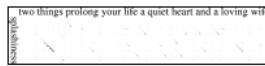







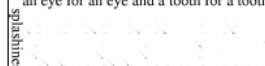



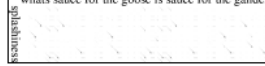





Experimental Results



Visualization Techniques for SIC

- Debugging a radical new paradigm like SIC is hard.
- One difficulty concerns seeing what is happening internally. What is the nature of the problem at each stage? Why do errors arise? How can they be fixed?
- We have developed several tools and visualization techniques to help us with our debugging. These make use of Tcl/Tk, a popular language for prototyping graphical user interfaces, as well as intermediate output in PostScript and bitmap formats.
- Most graphics in this talk were generated this way.

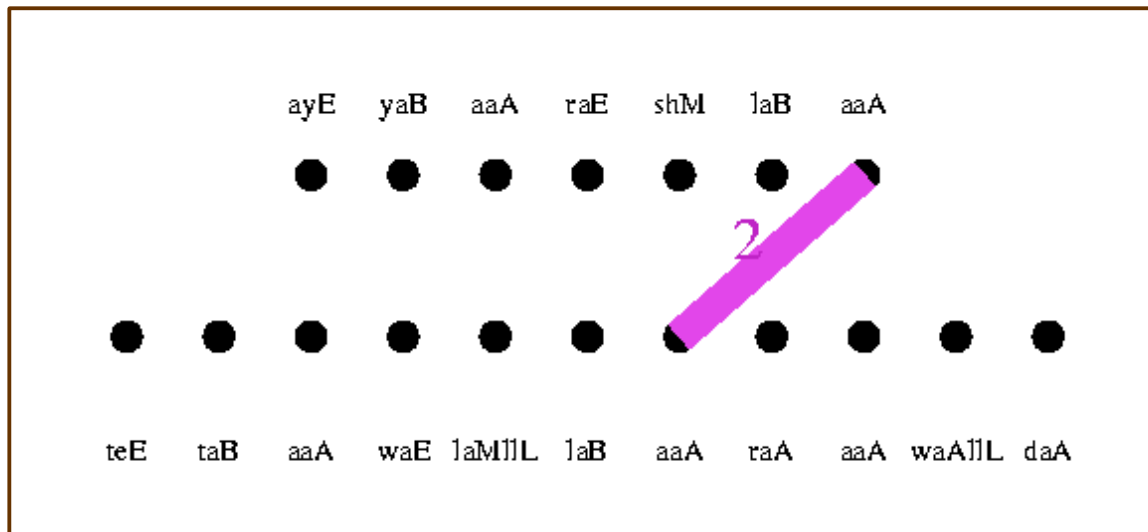
Web Browser Interface to SIC Results

Query String	Reference String	Lexical Distance Matrix	Lexical Match Graph	Signal Distance Spectrum	Signal Match Graph
<i>splashiness</i> features: GIF char positions: TXT	<i>two things prolong your life a quiet heart and a loving wife</i> features: GIF char positions: TXT	 GIF PS	 GIF PS	 GIF	 GIF PS
<i>splashiness</i> features: GIF char positions: TXT	<i>those who live in glass houses shouldnt throw stones</i> features: GIF char positions: TXT	 GIF PS	 GIF PS	 GIF	 GIF PS
<i>splashiness</i> features: GIF char positions: TXT	<i>an eye for an eye and a tooth for a tooth</i> features: GIF char positions: TXT	 GIF PS	 GIF PS	 GIF	 GIF PS
<i>splashiness</i> features: GIF char positions: TXT	<i>whats sauce for the goose is sauce for the gander</i> features: GIF char positions: TXT	 GIF PS	 GIF PS	 GIF	 GIF PS
<i>people who live in glass houses</i>	<i>people who live in glass houses shouldnt throw stones</i>	 GIF PS	 GIF PS	 GIF	 GIF PS

Each table row corresponds to one query-reference comparison

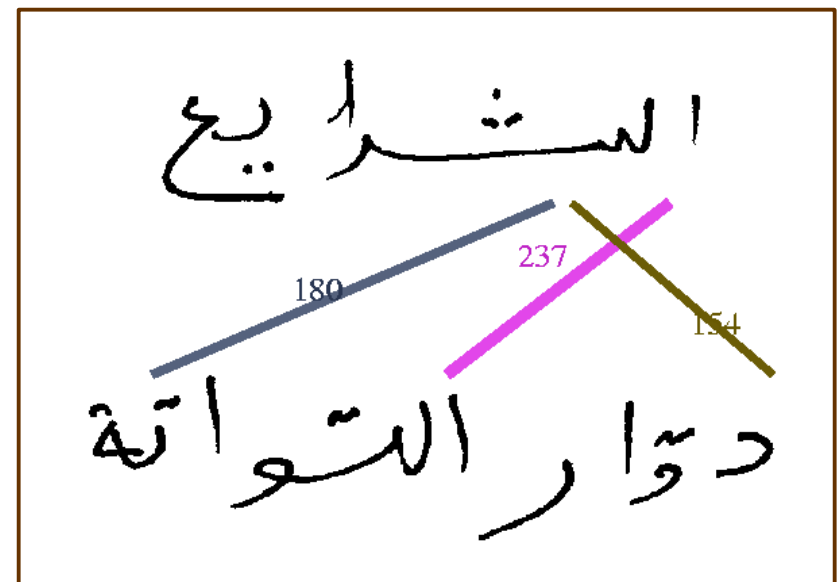
Thumbnail images are clickable to hires versions

SIC Example: Offline Arabic HWX

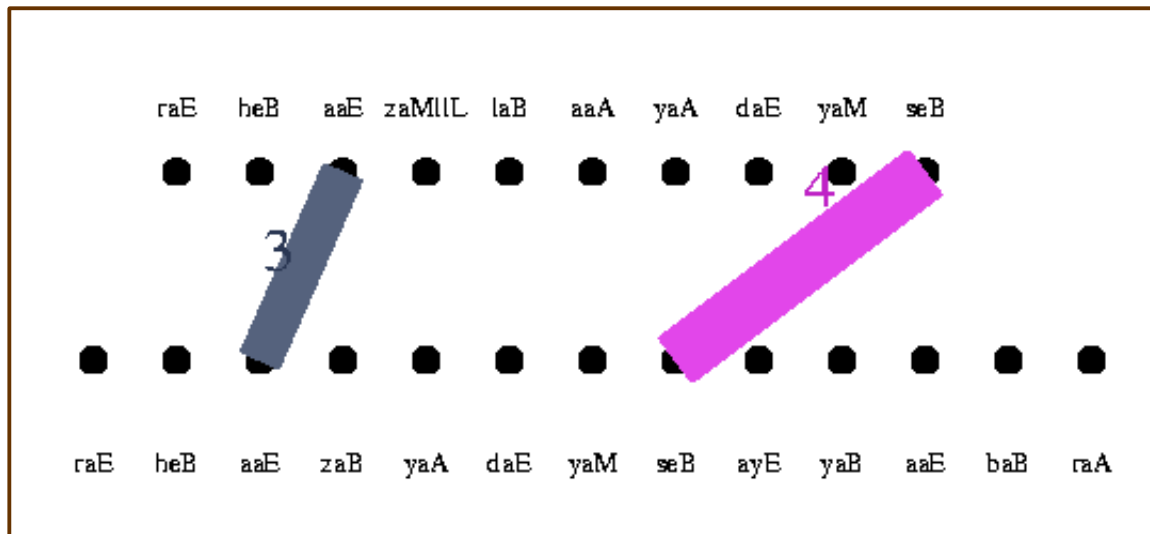


Lexicon match graph (bigrams or longer)

Corresponding signal match graph (showing top 3 weighted edges)

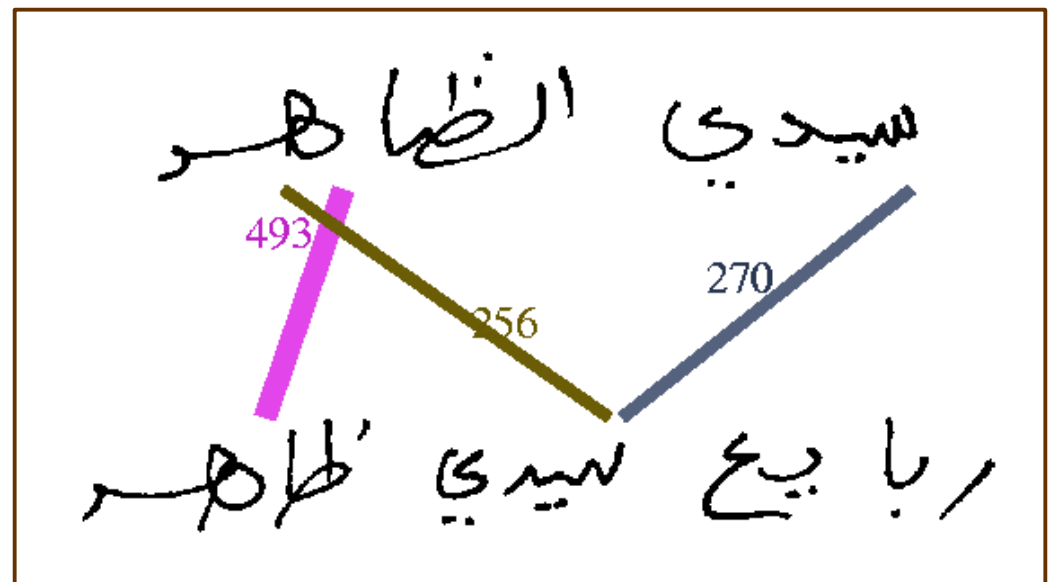


SIC Example: Offline Arabic HWX



Lexicon match graph
(bigrams or longer)

Corresponding signal
match graph (showing
top 3 weighted edges)

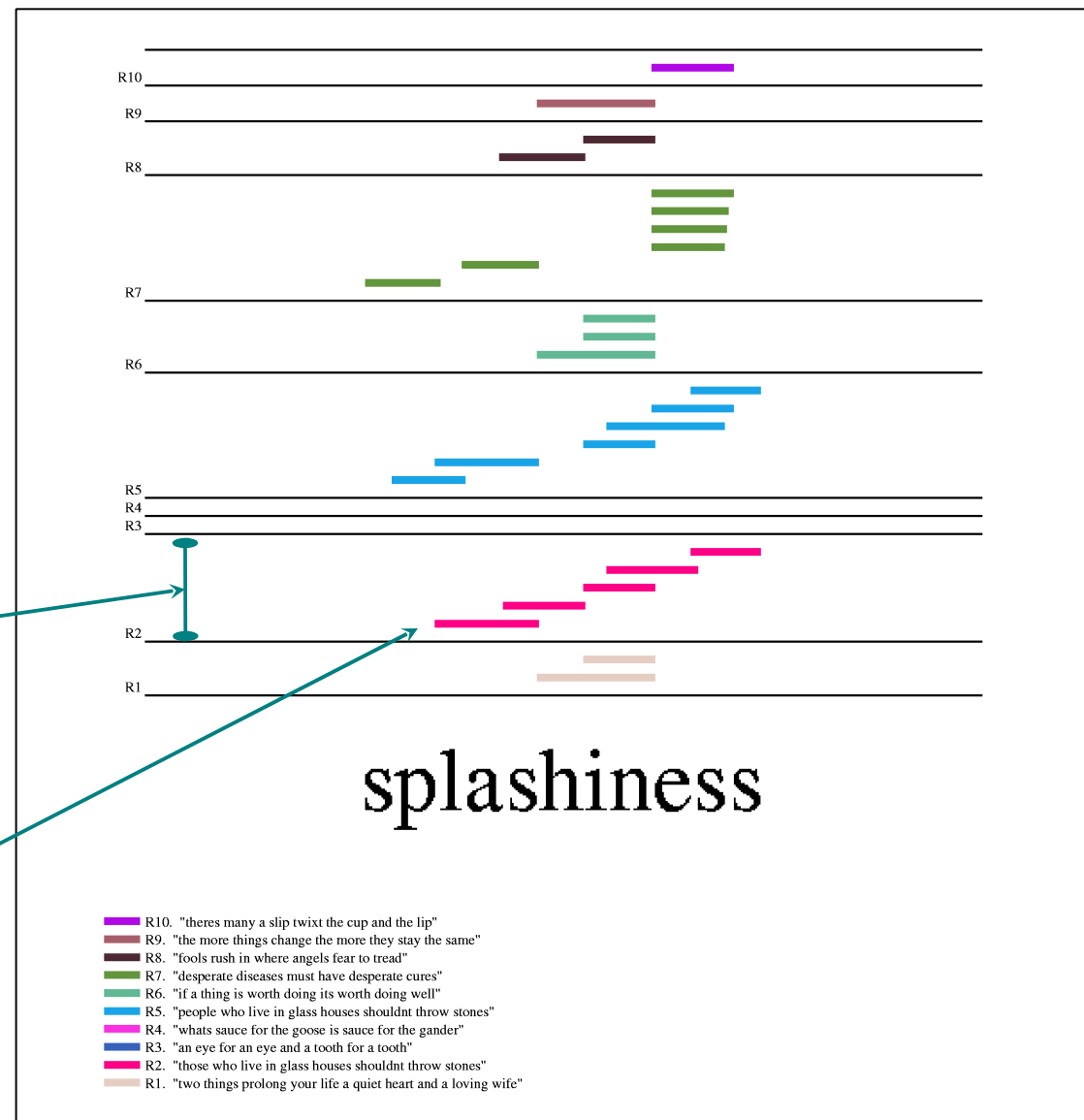


Visualizing Multiple Matching Results

Results of comparing signal input “splashiness” to 10 different reference strings:

Each reference string corresponds to a set of colored bars

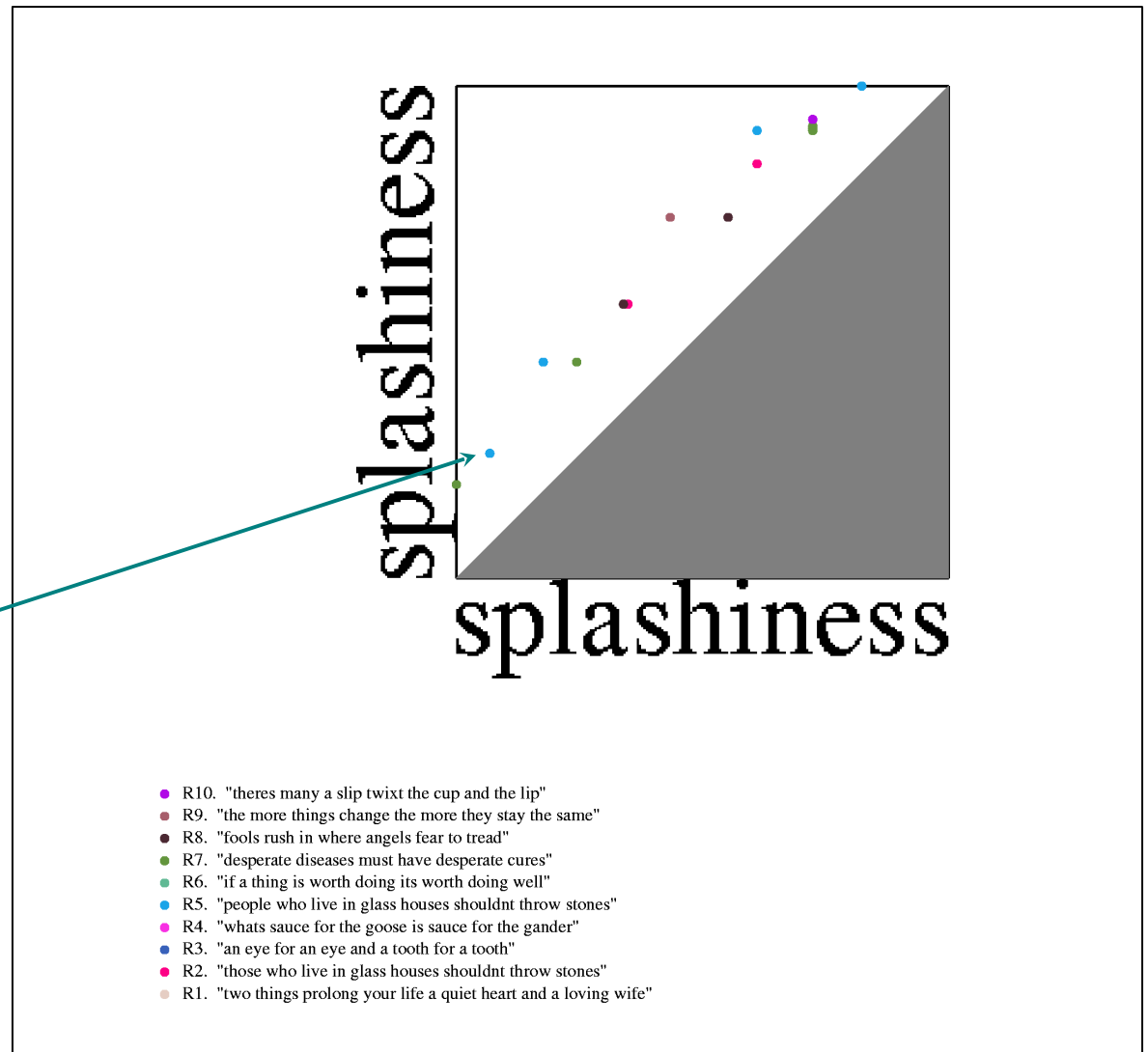
Each colored bar records starting and ending positions of one match along signal input



Visualizing Multiple Matching Results

Results of comparing signal input “splashiness” to 10 different reference strings:

Each match corresponds to a single datapoint. x-coordinate records starting position, y-coordinate records ending position.



Application Scenario

- You uncover a collection of high-value documents authored by a small number of writers.
- Existing techniques are confounded.
- You can't afford to spend time manually labelling large quantities of training data.
- Segmentation looks like it will be hard (Arabic?).
- SIC seems ideally suited to such situations.

~~Conclusions~~ Claims

- No parameter estimation required \Rightarrow easy to update.
Recognition should improve with use.
Humans adapt faster than machines, but static systems can't exploit this.
- Intrinsically more accurate than character-based schemes.
Lexical context is essential for speech and handwriting recognition.
- New words can be added without changing reference set.
Many applications require a large, easily extensible vocabulary.
Current training methods (HMMs) are unfriendly and unrepresentative.
- Source-specific recognition is easier.
In most human-computer interactions, the machine knows the user.
- Suitable for keyword-based IR.
- Can be extended from word-matching to phrase-matching.
- Moore's law favors non-parametric recognizers!

Final Words ...

- SIC is a promising new paradigm applicable to handwriting recognition and other hard problems.
- Groundwork has been laid.
- Future work to address feature choice for offline handwriting, adaptation strategies, computational efficiency, user interaction and interface.
- Basic needs that must be addressed: better-quality first-level matching, faster second-level matching.

SIC References

- “Indirect Symbolic Correlation Approach to Unsegmented Text Recognition,” G. Nagy, S. C. Seth, S. K. Mehta, and Y. Lin, *Computer Vision and Pattern Recognition, Workshop on Document Image Analysis and Retrieval*, Madison, WI, 2003.
- “A Nonparametric Classifier for Unsegmented Text,” G. Nagy, A. Joshi, M. Krishnamoorthy, Y. Lin, D. Lopresti, S. Mehta, and S. Seth, *Document Recognition and Retrieval XI (IS&T/SPIE International Symposium on Electronic Imaging)*, January 2004, Santa Jose, CA, pp. 102-108.
- “Match Graph Generation for Symbolic Indirect Correlation,” D. Lopresti, G. Nagy, and A. Joshi, *Document Recognition and Retrieval XIII (IS&T/SPIE International Symposium on Electronic Imaging)*, January 2006, San Jose, CA.
- Symbolic Indirect Correlation Classifier, A. Joshi, *PhD Dissertation*, Rensselaer Polytechnic Institute, Troy, NY, 2006.
- “A Maximum-Likelihood Approach to Symbolic Indirect Correlation,” A. Joshi, G. Nagy, D. Lopresti, and S. Seth, *Eighteenth International Conference on Pattern Recognition*, August 2006, Hong Kong.

The End

Thank you! Questions?