Document and Content Analysis

Summer 2009

Lecture 9
OCR Training and HMMs

Thomas Breuel
Faisal Shafait

Text Line Recognition

text line recognition

- oversegmentation
- grouping
- character recognition
- language modeling

statistical model

$$W = arg\,max_W\, P(W|x)$$

Consider segmentations S from some space of segmentations.

$$P(W|x) = \Sigma_S P(W, S|x) P(S)$$

$$P(W, S|x) = \frac{p(x|W, S) P(W, S)}{p(x)}$$

segmentation

Per character likelihoods are independent given the segmentation.

$$p(x|W, S) = \Pi_i \, p(x_i|W_i)$$

Assumption: joint density between string and segmentation depends only on length.

$$P(W, S) \approx P(W) \cdot P(S) \cdot [|W| = |S|]$$

$$P(W,S|x)=\frac{\Pi_i p(x_i|w_i)P(W)}{p(x)}P(S,W)$$

$$=\frac{\Pi_i p(x_i|w_i)P(W)}{\Pi_i p(x_i)}\cdot\frac{\Pi_i p(x_i)}{p(x)}\cdot P(S)\cdot[|W|=|S|]$$

Apply Bayes rule.

$$=\Pi_i\frac{P(w_i|x_i)}{p(w_i)}P(W)\cdot\frac{\Pi_i p(x_i)}{p(x)}\cdot P(S)\cdot[|W|=|S|]$$

For classification, p(x) doesn't matter.

$$\propto\prod_i\frac{P(w_i|x_i)}{p(w_i)}P(W)\cdot\prod_i p(x_i)\cdot P(S)\cdot[|W|=|S|]$$

Assume a uniform prior for the segmentations.

$$\approx\prod_i\frac{P(w_i|x_i)}{p(w_i)}P(W)\cdot\prod_i p(x_i)\cdot[|W|=|S|]$$

Viterbi approximation

$$P(W|x) = \Sigma_S P(W, S|x)\, P(S)$$

Viterbi approximation

$$\approx max_S P(W, S|x)\, P(S)$$

Uniform prior on segmentations.

$$\approx max_S P(W, S|x)$$

components of model

$$arg\,max_W\, P(W|x)$$

$$\approx arg\,max_{W,S,|W|=|S|} \prod_i \frac{P(w_i|x_i)}{p(w_i)} P(W) \cdot \prod_i p(x_i)$$

isolated
character
classification

Per character prior.

Language model (prior
over strings).

Density models of
characters in correct
segmentations.

components of model (log)

$$arg\,max_W \log P(W|x)$$

$$\approx arg\,max_{W,S,|W|=|S|} \underbrace{\log P(W)}_{\text{language model costs}} + \sum_i \underbrace{\log P(w_i|x_i) - \log p(w_i) + \log p(x_i)}_{\text{per-character costs}}$$

summary

- **arg $\max_W$ P(W|x) - Bayes-optimal recognition**
- **$\log P(W|x) \approx \log P(W) + \sum \log f(x_i, w_i)$**
  - cost of path through recognition lattice
  - cost of path through language model

Finite State Transducers

weighted finite state transducers

- a set of states Q
- an input alphabet Σ
- an output alphabet Γ
- initial states $I \subseteq Q$
- final states $F \subseteq Q$
- semi-ring K of weights
- a transition relation $Q \times \Sigma \times \Gamma \times Q \times K$

weighted finite state transducers

- **weighted finite state transducers represent string replacements with associated costs**

abotu → about / 1.0
actualyl → actually / 1.0
aboto → about / 2.0
i snot → is not / 1.0
...

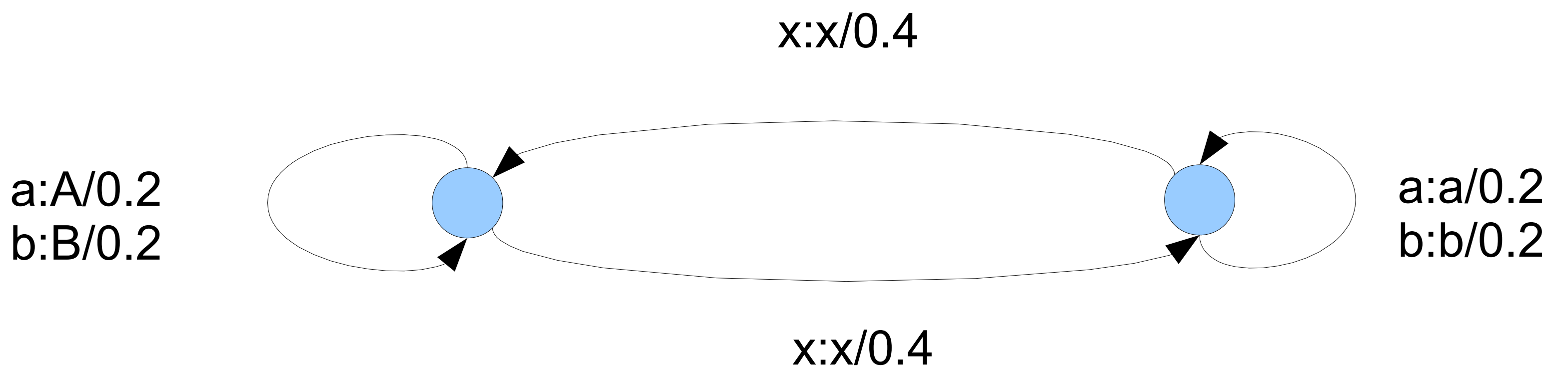
weighted finite state acceptor

- **special case:**
  input symbols = output symbols
- **weighted finite state acceptors represent sets of strings and associated weights**

special cases

- input / output symbols can be “empty” ε
- transducers can be non-deterministic

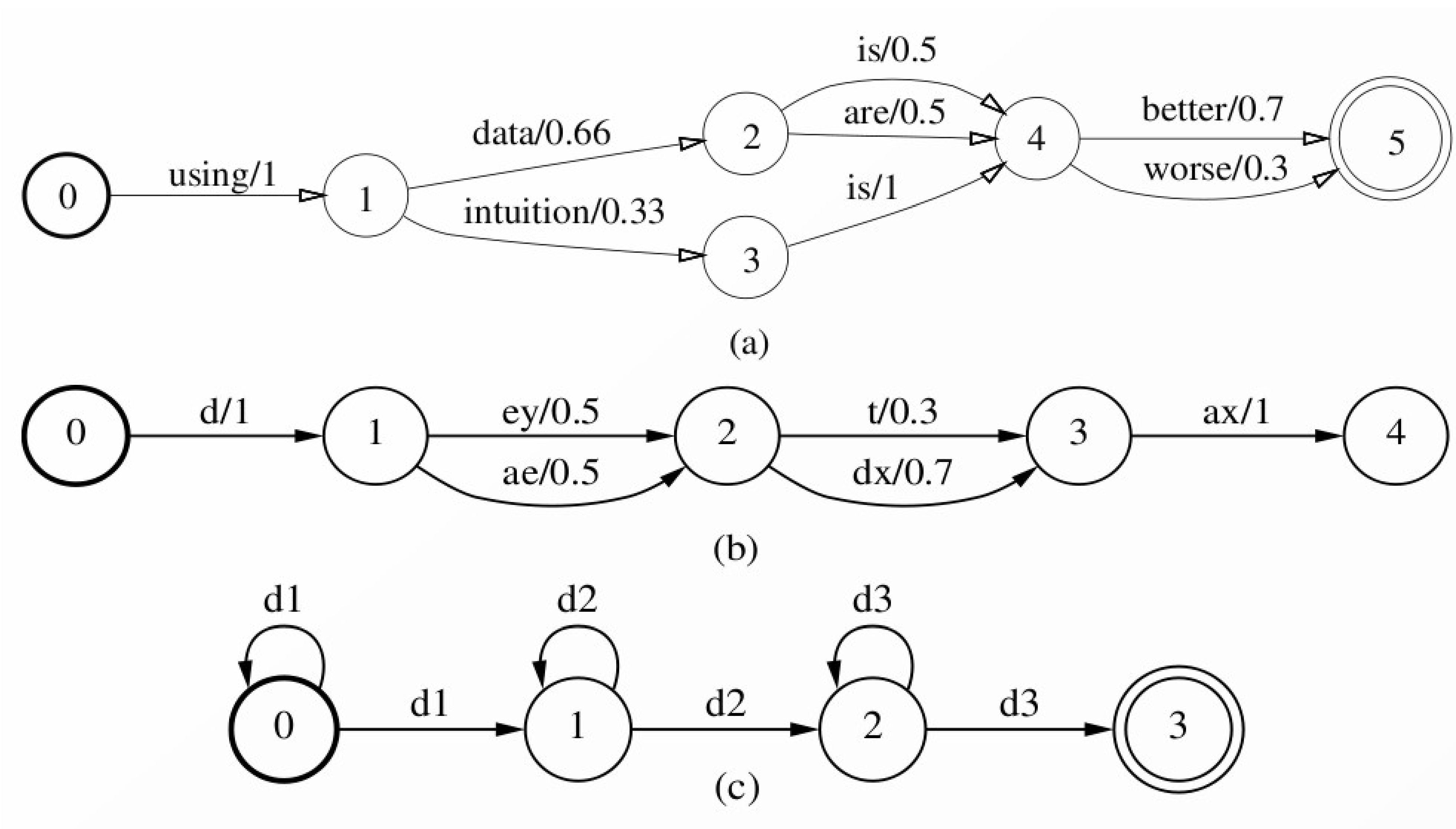
simple example



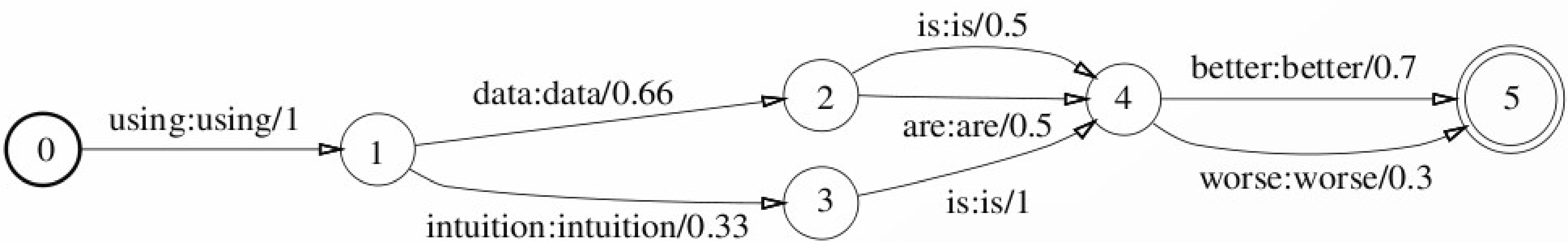
abbaxabbaxabba → ABBAxabbaxABBA

aaaaaaaxxbbaxbbbbb → AAAAAAAxBBAxbbbbb

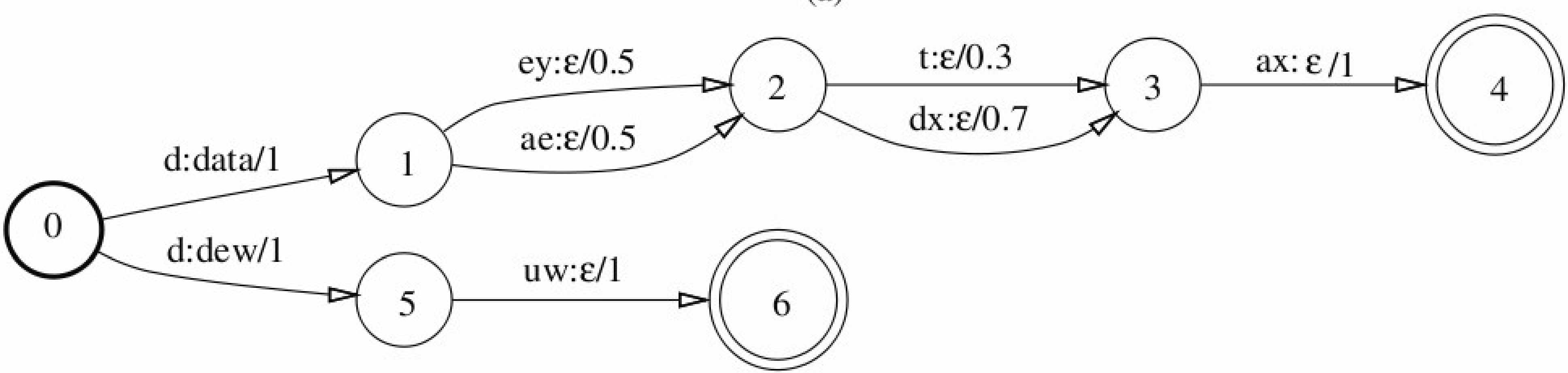
acceptor examples



transducer examples



(a)



(b)

semiring

- **different operations for combining weights**
  - along a path — addition
  - when paths meet — multiplication
- **common semirings**
  - (R,+,x,0,1) — probability semiring
  - ($R_+ \cup \{\infty\}$,min,+,∞,0) — tropical semiring (Viterbi approx.)

basic operations on FSTs

- **union**
  - $x[T \cup S]y$ iff $x[T]y$ or $x[S]y$
- **intersection**
  - $x[T \cap S]y$ iff $x[T]y$ and $x[S]y$
- **Kleene closure**
  - $\varepsilon[T^*]\varepsilon$, $a[T^*]b$ and $x[T]y \rightarrow ax[T]by$
- **concatenation**
  - $uv[T \cdot S]wx$ iff $u[T]w$ and $v[S]x$
- **composition**
  - $x[T \circ S]z$ iff $x[T]y$ and $y[S]z$

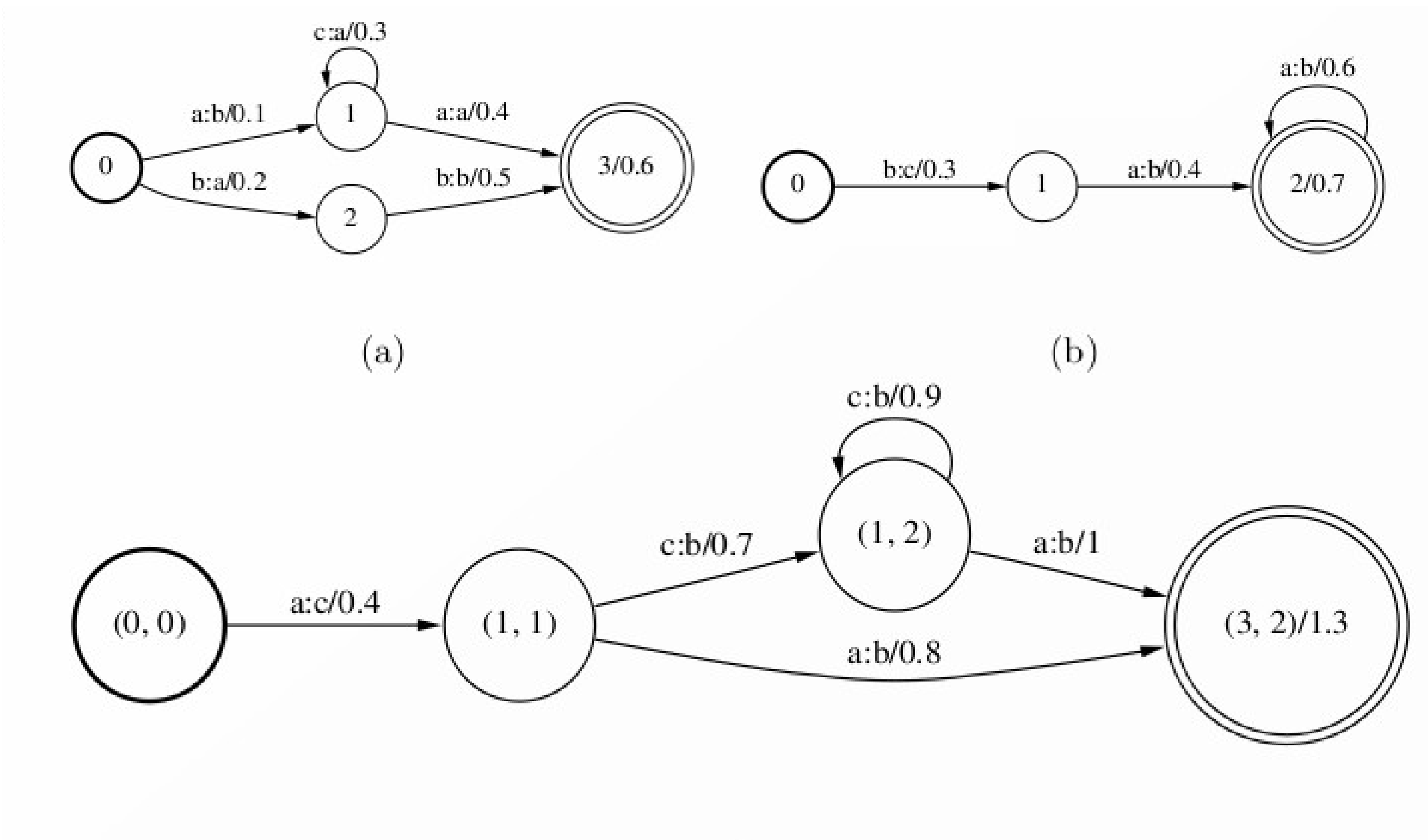
WFST algorithms

- **ε-removal**
  - eliminate empty symbols
- **determinization**
  - eliminate non-deterministic transitions
- **minimization**
  - minimum number of states/transitions
- **garbage collection**
  - remove inaccessible states

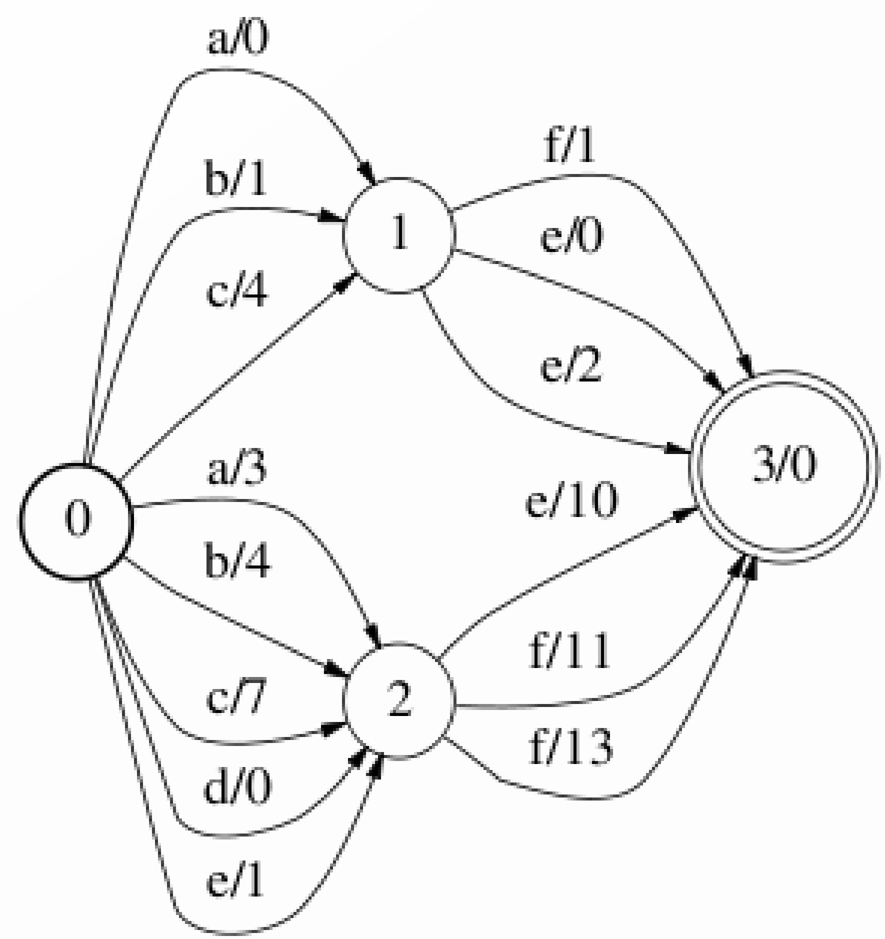
WFST algorithms

- **bestpath**
  - find the best path through a given WFST
- **bestpath through composition**
  - find the best path through a composition of WFSTs (without explicitly computing the composed WFST)

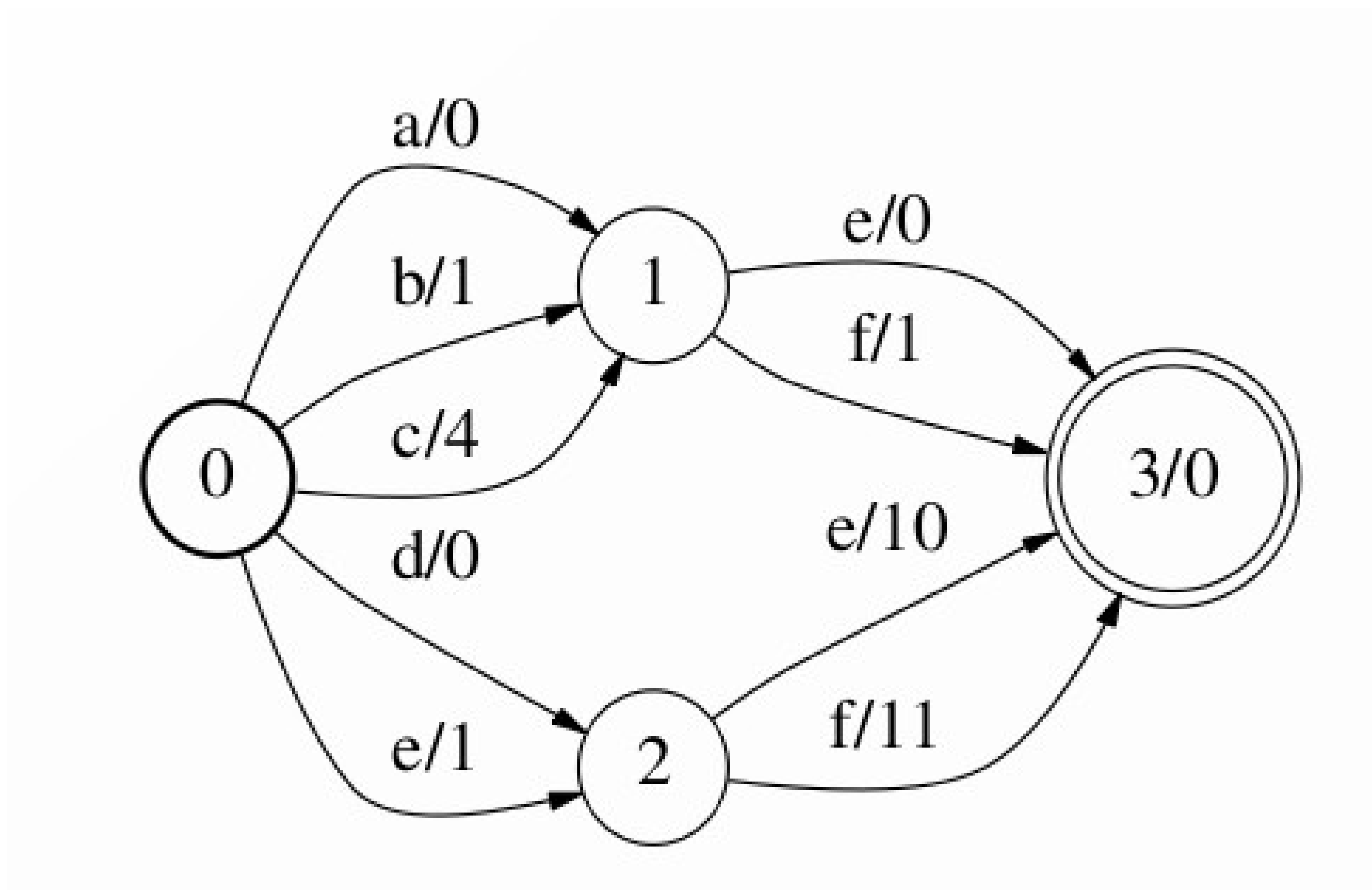
composition



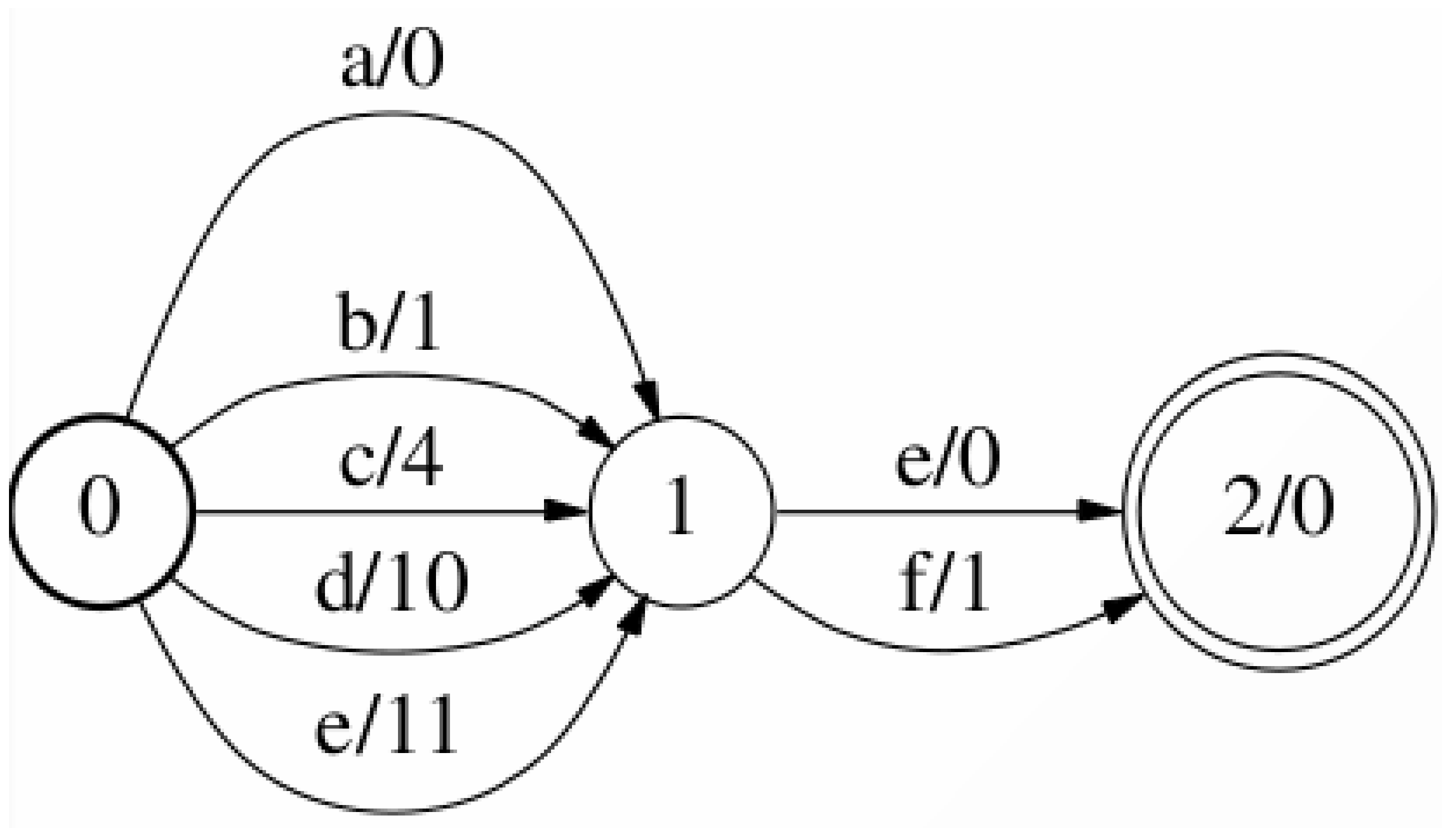
non-deterministic acceptor



deterministic equivalent



minimized equivalent



Uses of Weighted Finite State Transducers

recall...

- OCR produces recognition alternatives
- the best alternative may not be the right one
- language models re-weight alternatives

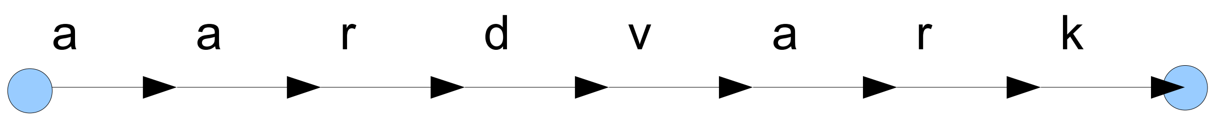
lang model for text lines

- **language model should contain "Department of Geological Sciences"**
- **allow any repetition of words in the dictionary, separated by spaces or punctuation**

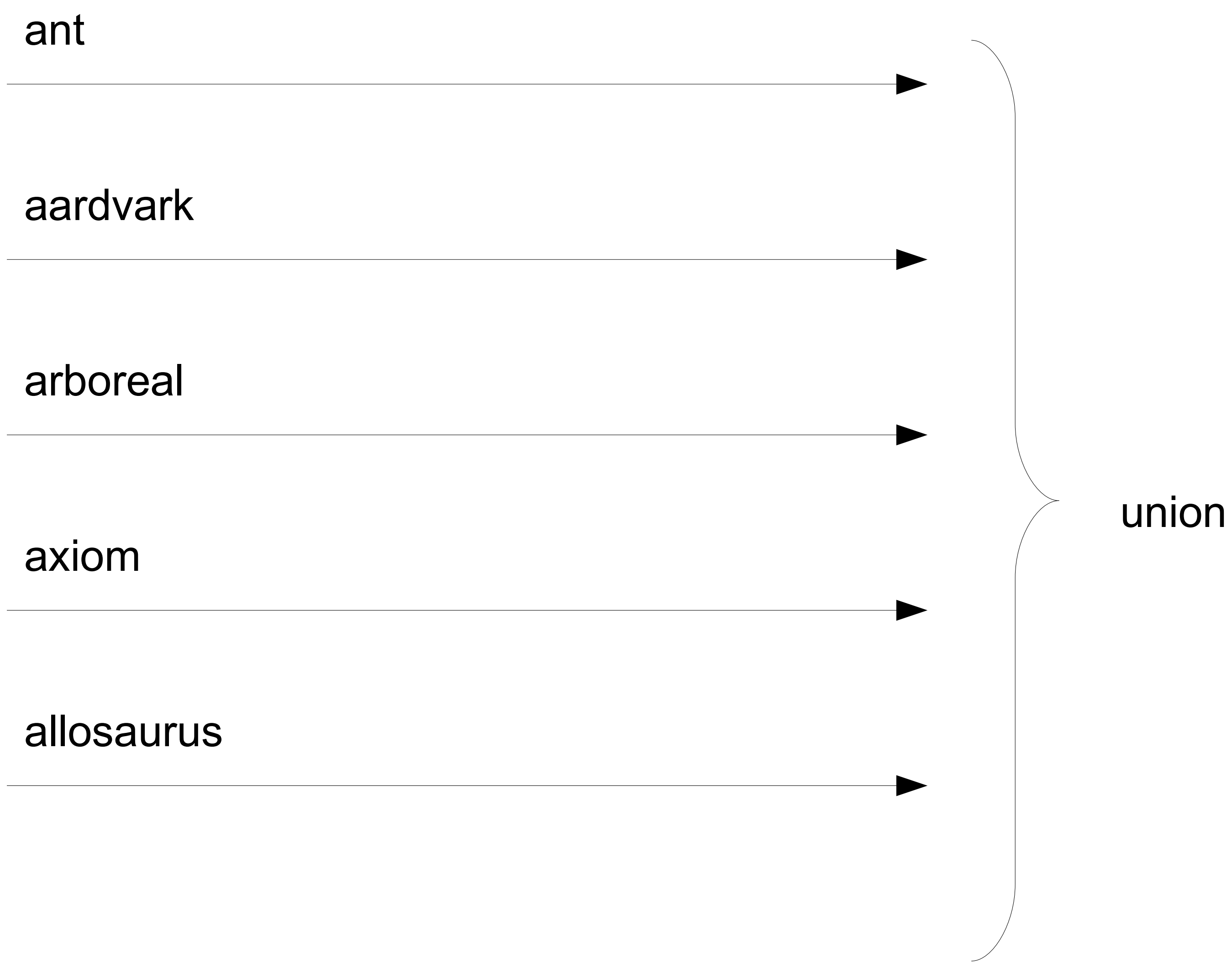
constructing a language model

Construct a simple language model from individual words/strings.

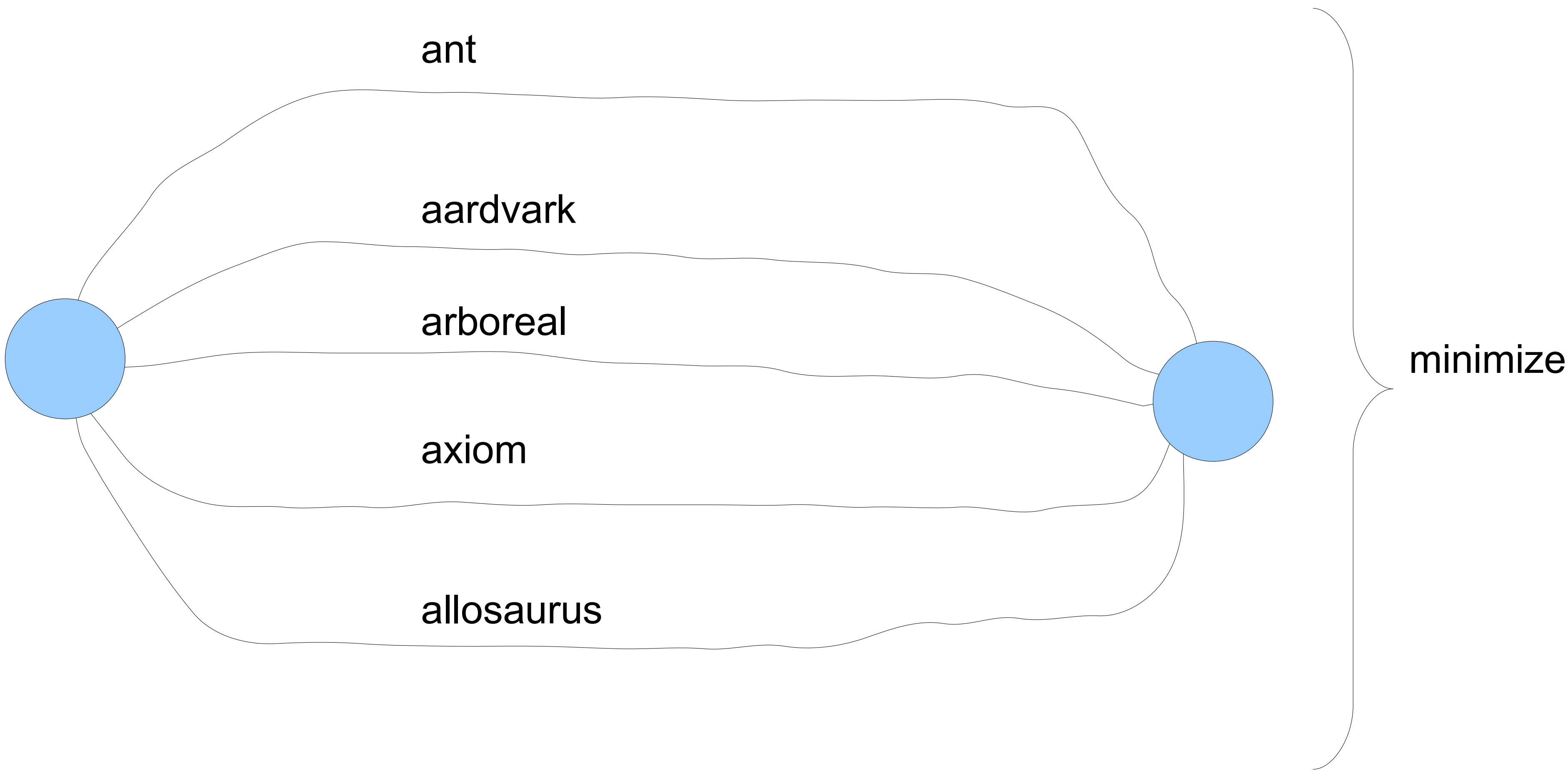
“aardvark”



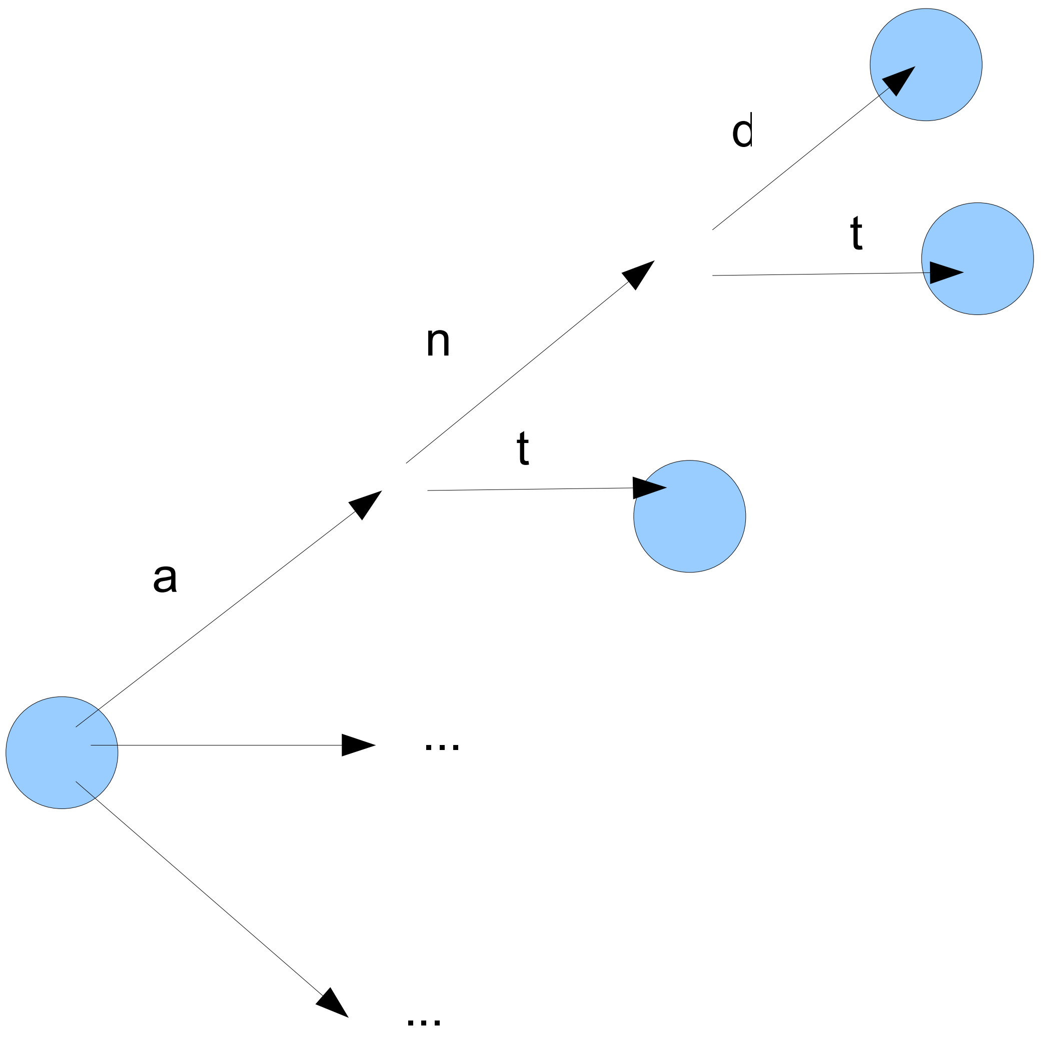
constructing a language model



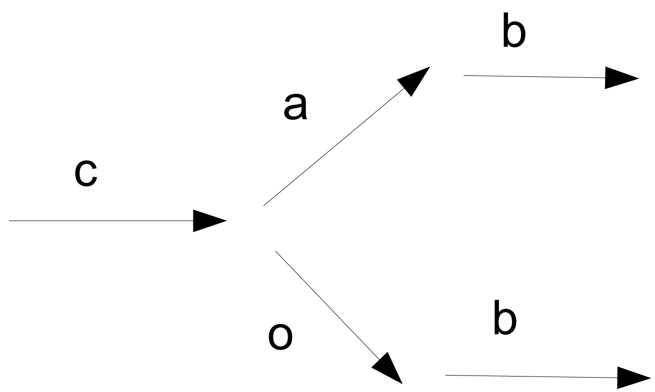
constructing a language model



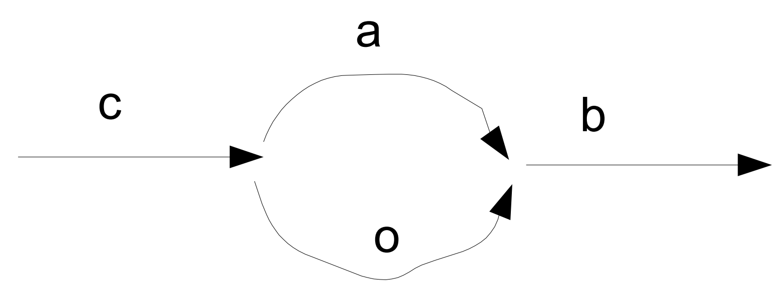
trie models



trie vs minimized

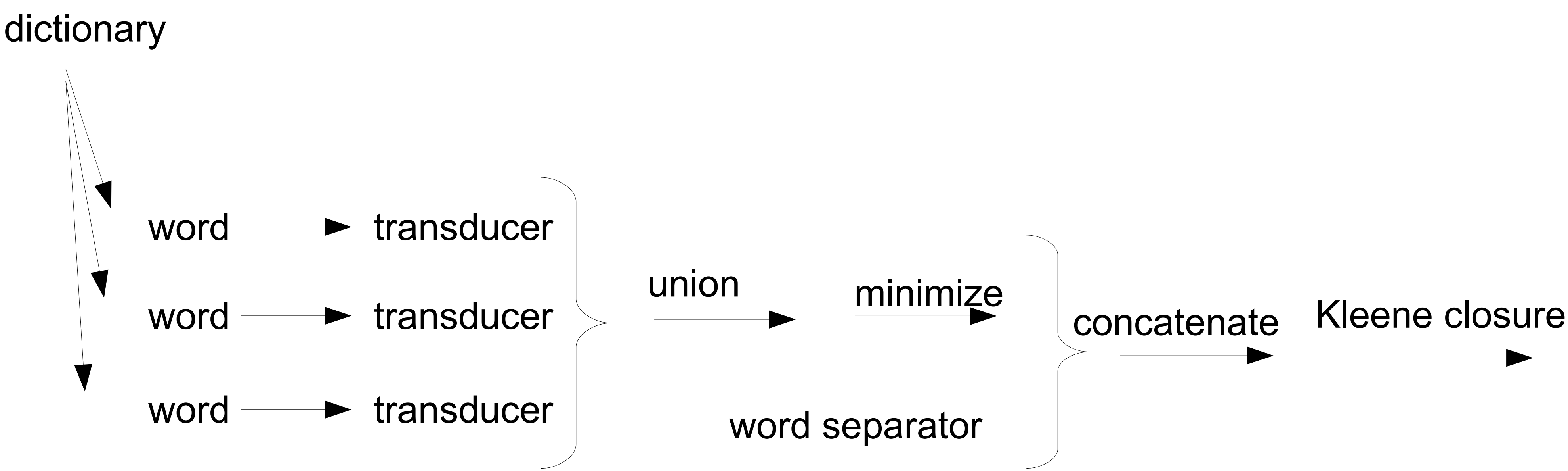


trie

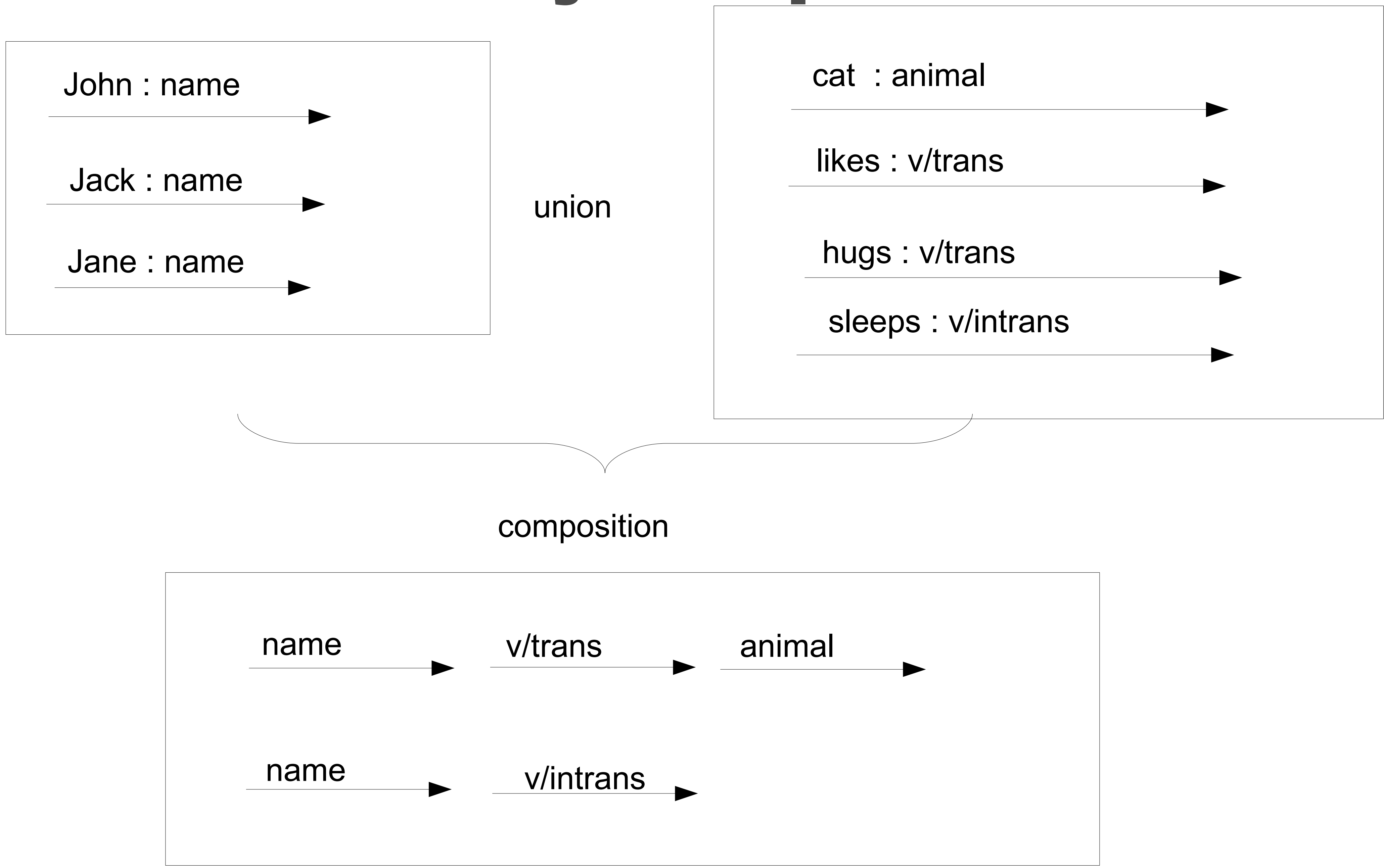


minimized
transducer

constructing a language model



construction by composition



n-gram probability models

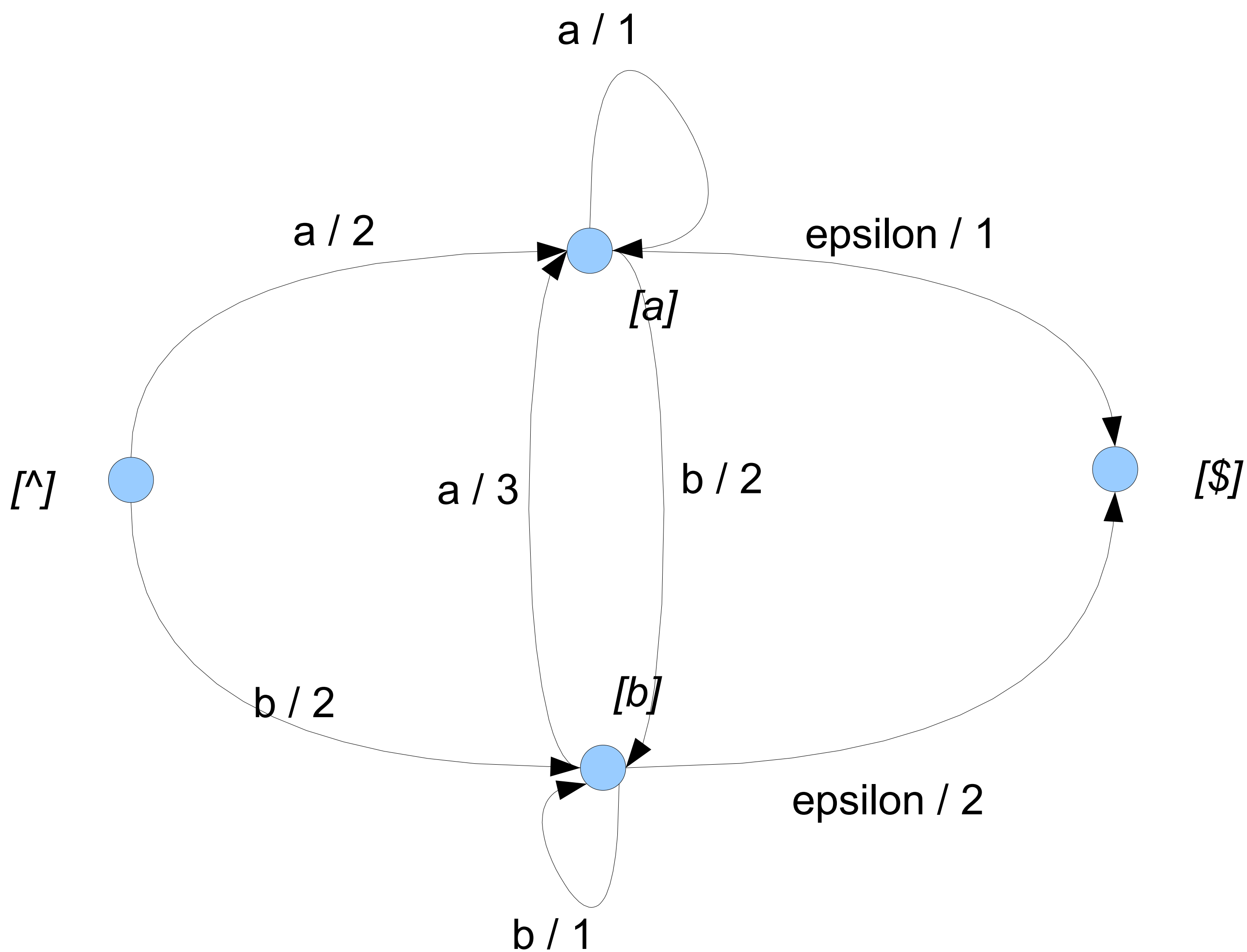
- **recall n-gram probability models**
- **we're given**

$$P(c_l \mid c_{l-n} \ldots c_{l-(n-1)})$$

- **transduction**
  - string to itself (i.e., weighted acceptor)
  - cost is the product of all the conditional probabilities

constructing an n-gram transducer

^a / 2
^b / 2
aa / 1
bb / 1
ab / 2
ba / 3
a$ / 1
b$ / 2



Key idea: each state represents the history;
transitions on the last symbol in the string.

modular language model construction

- **construct language models from...**
  - strings
  - union
  - minimization
  - Kleene closure
  - concatenation
- **special cases**
  - trie model
  - n-gram model

Training

how do we train an OCR system?

$$\log P(W) + \sum_i \log P(w_i|x_i) - \log p(w_i) + \log p(x_i)$$

language model

classifier

character frequencies

density model of
character shapes

(character /
non-character
classification)

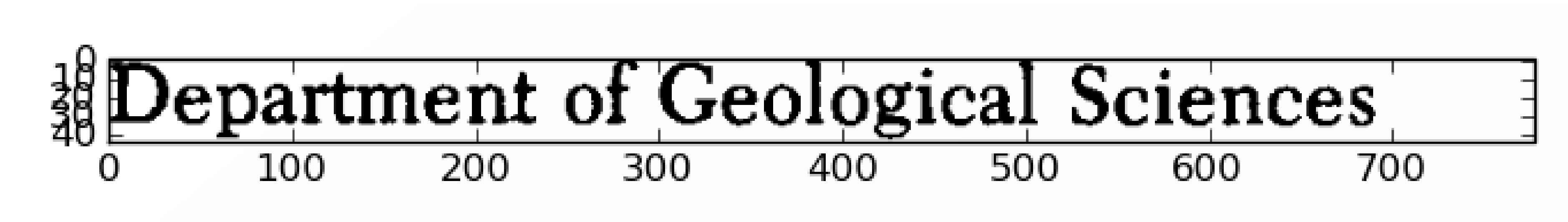
x_i requires knowledge of the correct segmentation S

S is a missing variable

- $P(c|x_i)$ and $P(x_i)$ require isolated characters for training
- knowledge of isolated characters requires knowledge of segmentation S
- the correct segmentation S isn't usually given or known

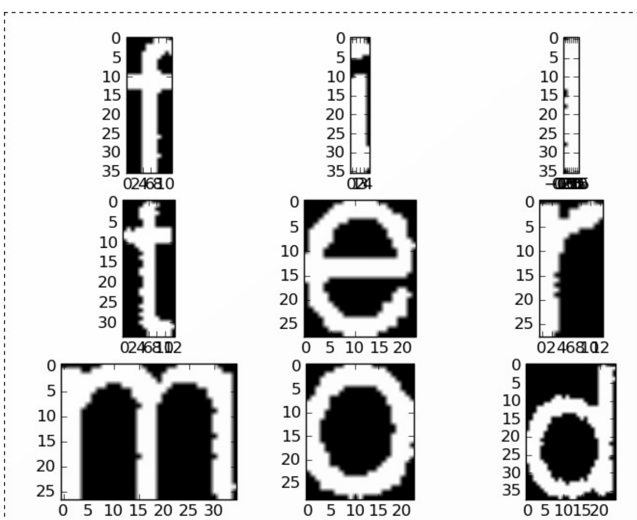
OCR training

GIVEN



“Department of Geological Sciences”

NEEDED

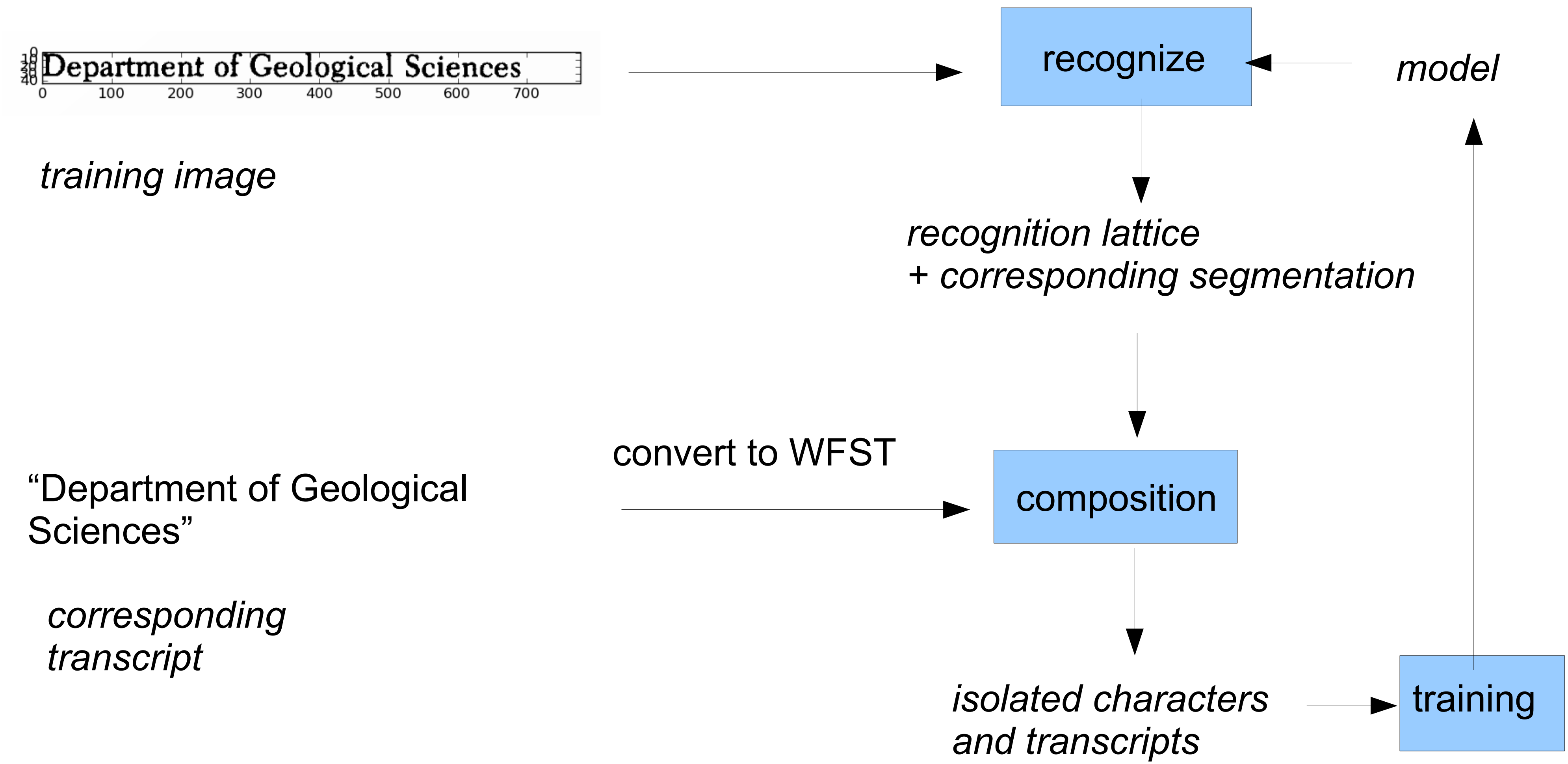


“f” “i” “l”

“t” “e” “r”

“m” “o” “d”

OCR training



OCR training

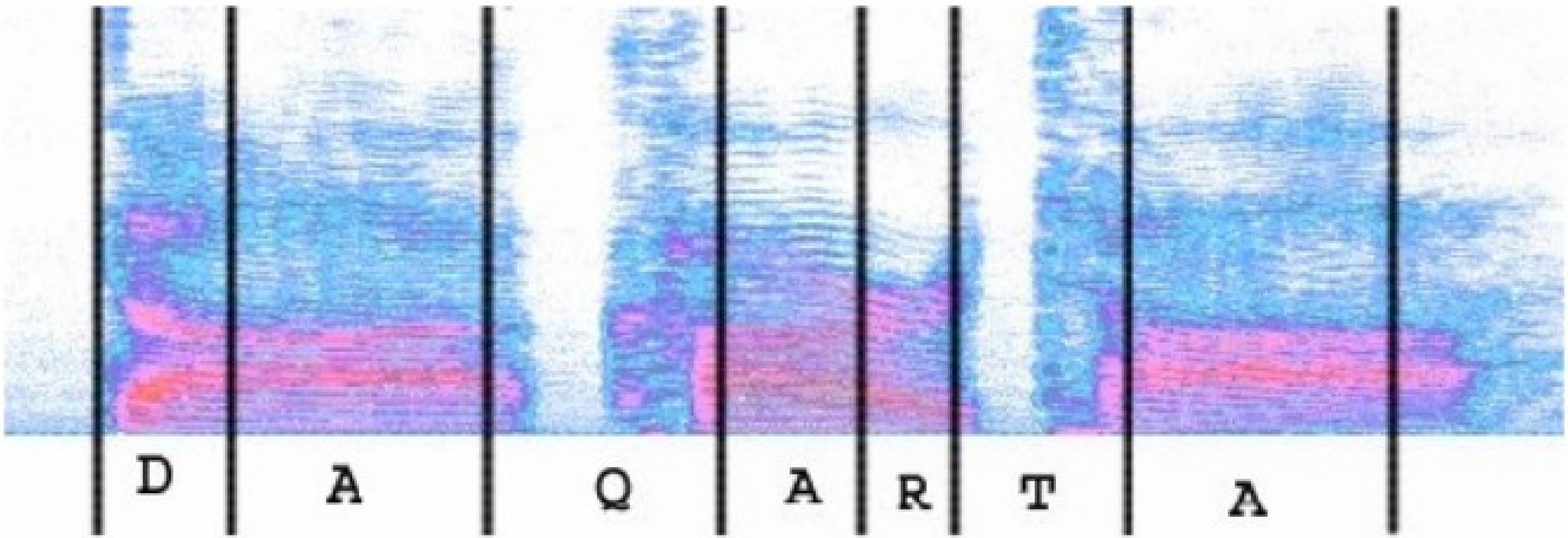
- character recognizers need to be trained on isolated characters
- training data is usually given on a per-line basis
- therefore, training requires computing segmentation S

OCR training

- **EM-algorithm**
  - given an initial classifier, compute W and S for each text line, using the transcription as the language model
    - throw away W (it is just the transcription)
    - S gives the segmentation into characters
    - $|S| = |W|$
  - use S to divide the training image into isolated characters x_i
  - look up the corresponding w_i in W
  - use the (x_i,w_i) to train a better model
  - repeat until convergence

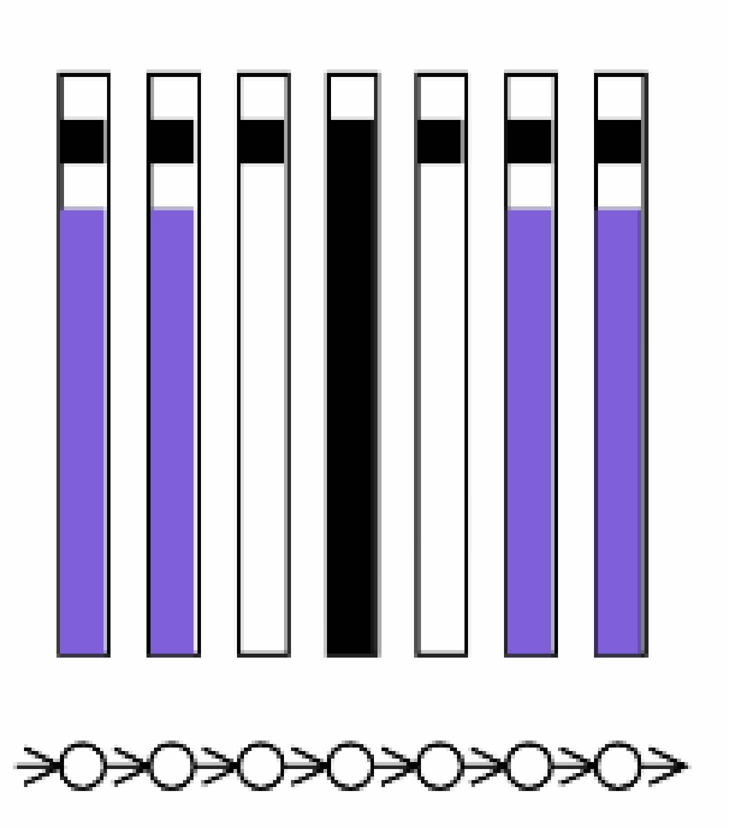
Relationship to HMMs

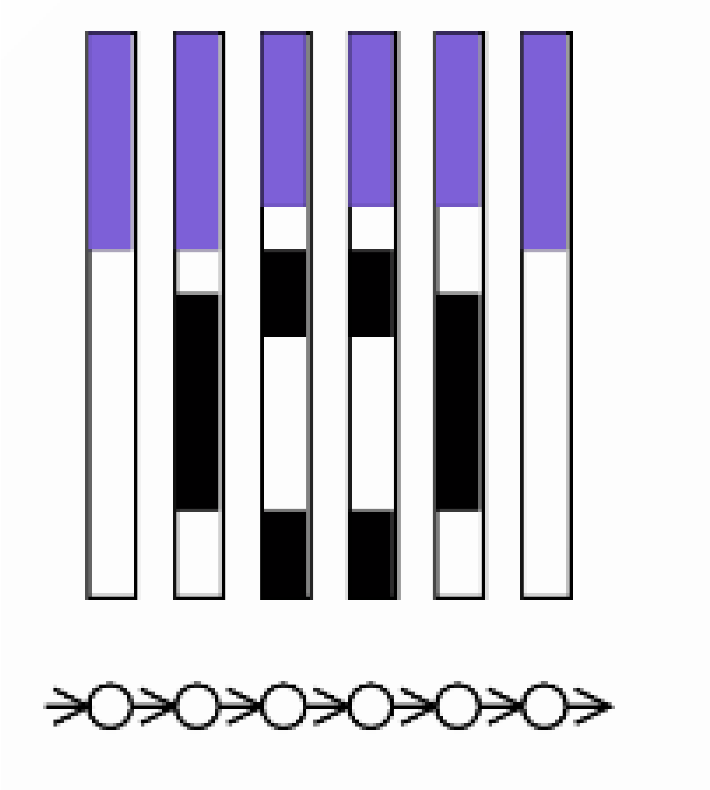
generalization of HMM



- “dense” segmentation—cuts can be anywhere
- restricted shape of cuts

OCR as HMM





OCR as discrete HMM

- **consider images 8 pixels high**
- **each vertical slice is one of 256 symbols**
- **the input becomes a symbol sequence**
  - "_ _ _ _ ' ' | ' ' _ _ |: :| _ _"
- **OCR becomes translating symbol sequences**
  - "_ _ _ _ ' ' | ' ' _ _ |: :| _ _" → " TO "

continuous HMMs

- **discrete HMM**
  - each input is a discrete symbol c
  - each state i is associated with a cost M_{ci}
- **continuous HMM**
  - each input is a vector x in R^n
  - each state i is associated with a density $p(x|i)$ (often a normal density)
- **decoding**
  - analogous in discrete and continuous case
  - usually need different algorithm, though

OCRopus

OCRopus representation

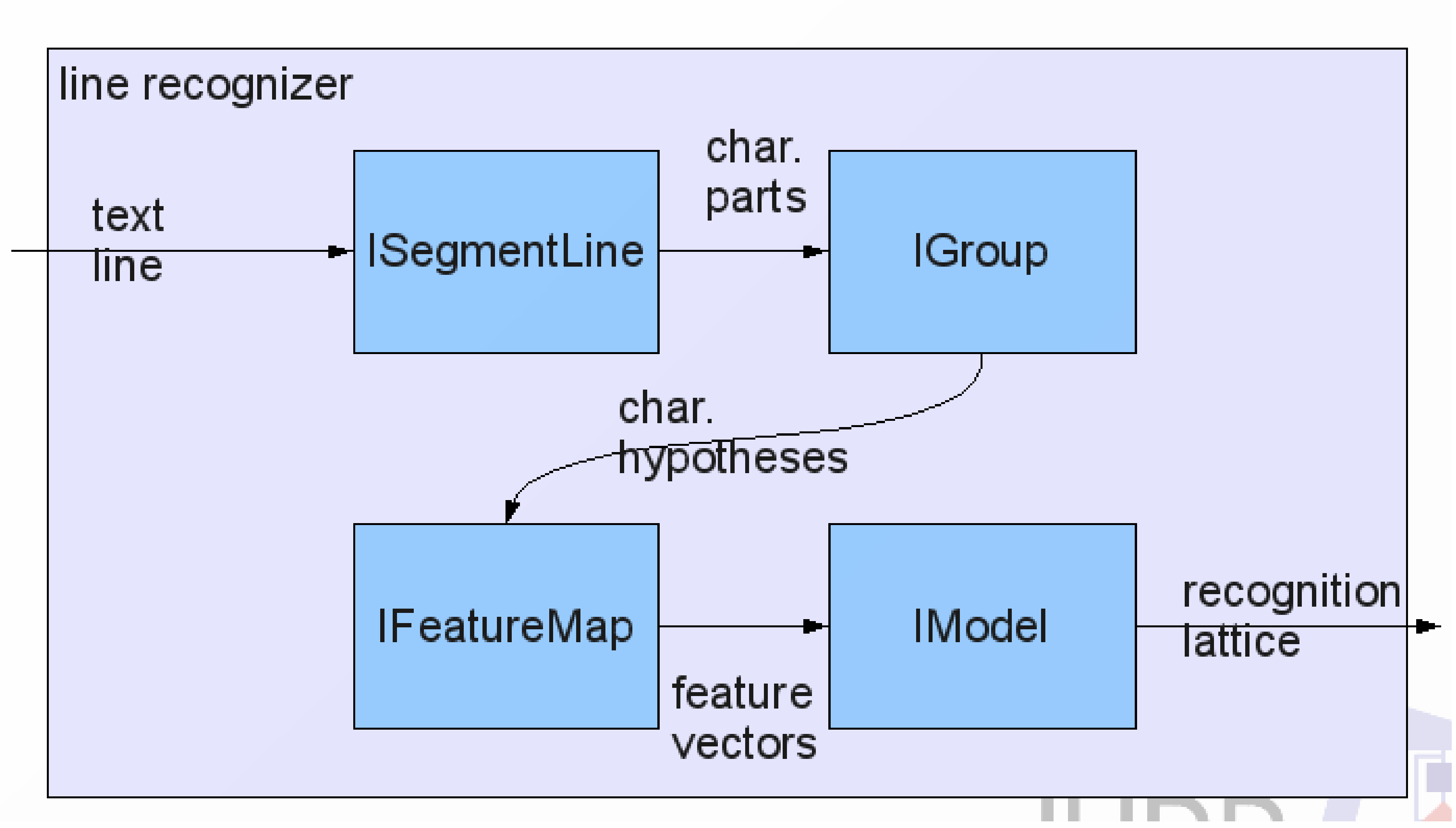
- **book directory**
  - 0000.png – physical page 0
  - 0000/
    - 0001 – physical line 1 (in reading order)
    - 0001.png – line image
    - 0001.cseg.png – character segmentation
    - 0001.fst – recognition lattice
    - 0001.txt – text output
  - 0001/
    - ...

OCRopus commands

- **book2pages**
  - generates book/0000.png …
- **pages2lines**
  - generates book/0000/0001.png …
- **lines2fsts**
  - generates book/0000/0001.fst
- **fsts2text**
  - generates book/0000/0001.txt
- **buildhtml**
  - generates HTML/hOCR output from book

OCRopus commands

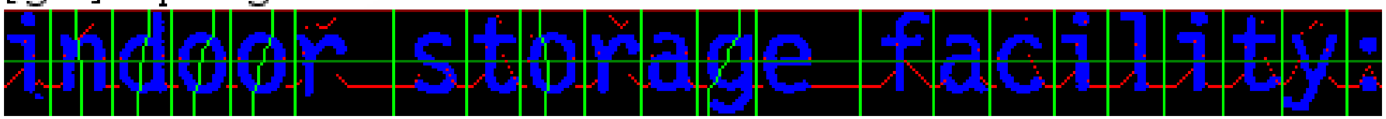
- **align**
  - generates 0000/0001.cseg.png from...
    - 0000/0001.png
    - 0000/0001.rseg.png
    - 0000/0001.txt
- **trainseg**
  - generates a model from...
    - 0000/0001.png
    - 0000/0001.cseg.png
    - 0000/0001.txt


line recognizer
text
line
ISegmentLine
char.
parts
IGroup
char.
hypotheses
IFeatureMap
IModel
recognition
lattice
feature
vectors

[glr] Recognizing book-10k8/0001/0005.png

[glr] input :

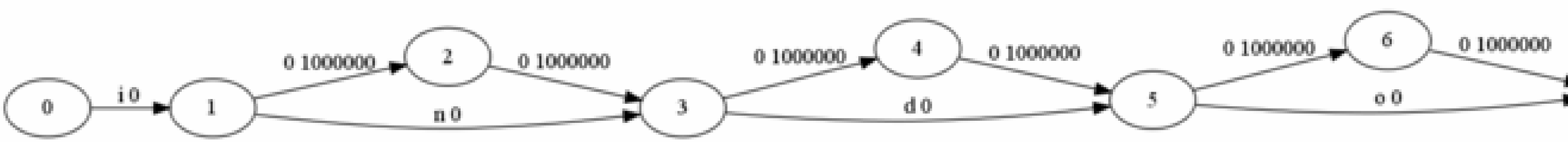
[glr] DpSegmenter:



[glr] segmentation:

[glr] baseline :

[glr] fst:



indoor storage facility:

summary

- statistical foundations
- weighted finite state transducers
- building language models
- OCR training and EM training
- relation to HMMs
- OCRopus commands