Overview

Schone and Jurafsky (2000) “Knowledge-Free Induction of Morphology Using Latent Semantic Analysis”

- Presents an unsupervised (“knowledge-free”) algorithm for morphological induction
- “...with the exception of word segmentation, we provide it no human information...”
- The input is a space-separated, unlabeled text corpus (they use an 8 million word English corpus)
- Algorithm output “conflation sets” of morphologically related words

Their algorithm is divided into four parts:

1. Hypothesize candidate affixes
2. Identify pairs of candidate affixes which may be morphological variants, e.g. (*ed*, *ing*) or (*s*, NULL).
3. Collect contextual information about all word pairs which share these morphologically variant affixes, e.g. (*walked*, *walking*) or (*walks*, *walk*).
4. Determine "morphologically relatedness" for those word pairs with similar semantics (as defined by their ± 50 word context).

Hypothesize candidate affixes

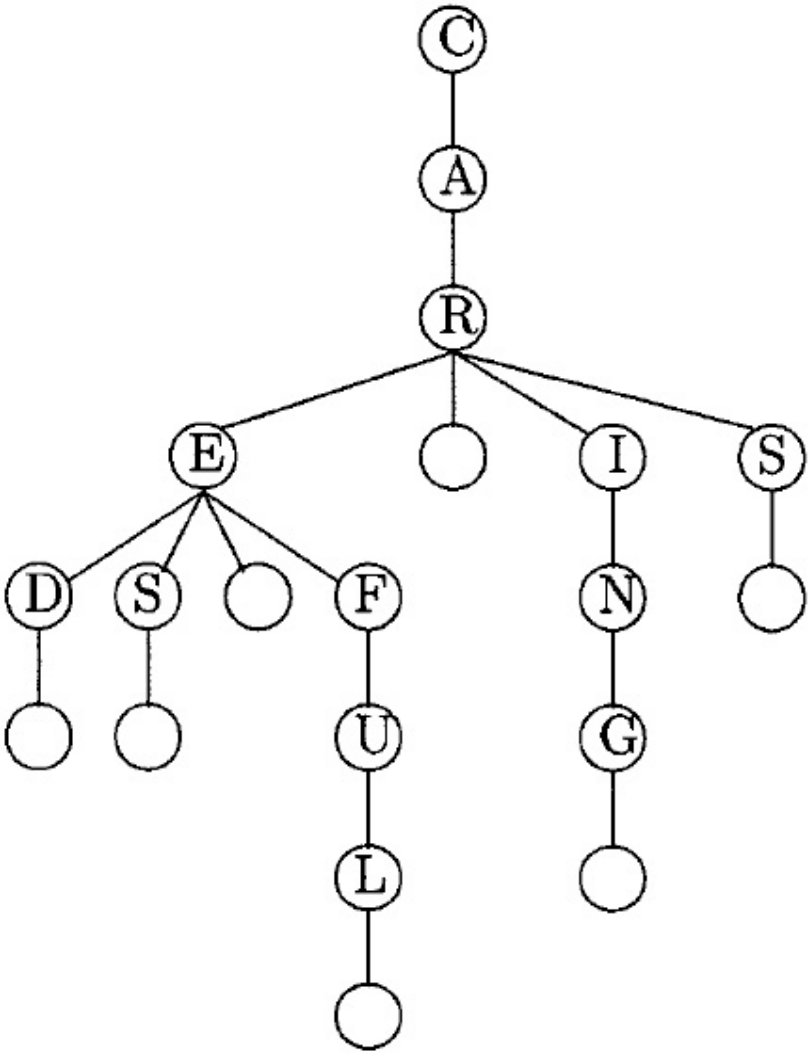
- Identify p-similar words (from Gaussier 1999):

  Def'n: Two words w_1 and w_2 are said to be ***p*-similar** if and only if:

  a. the first p characters of w_1 are the same as the first p characters of w_2
  b. the $p+1$ characters of w_1 and w_2 are not the same

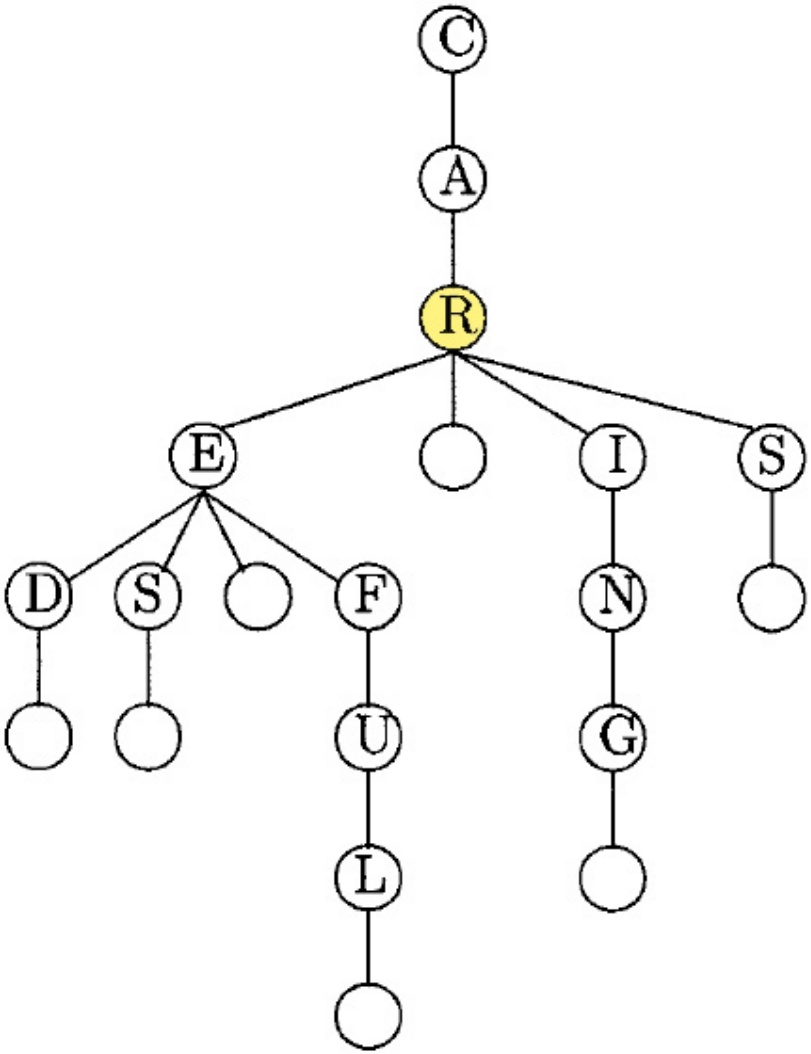
- Ex. *walks* and *walking* are 4-similar, as are *walk* and *walks*.
- These pairs are not 5-similar, by rule (a), and not 3-similar, by rule(b).

Identifying affixes



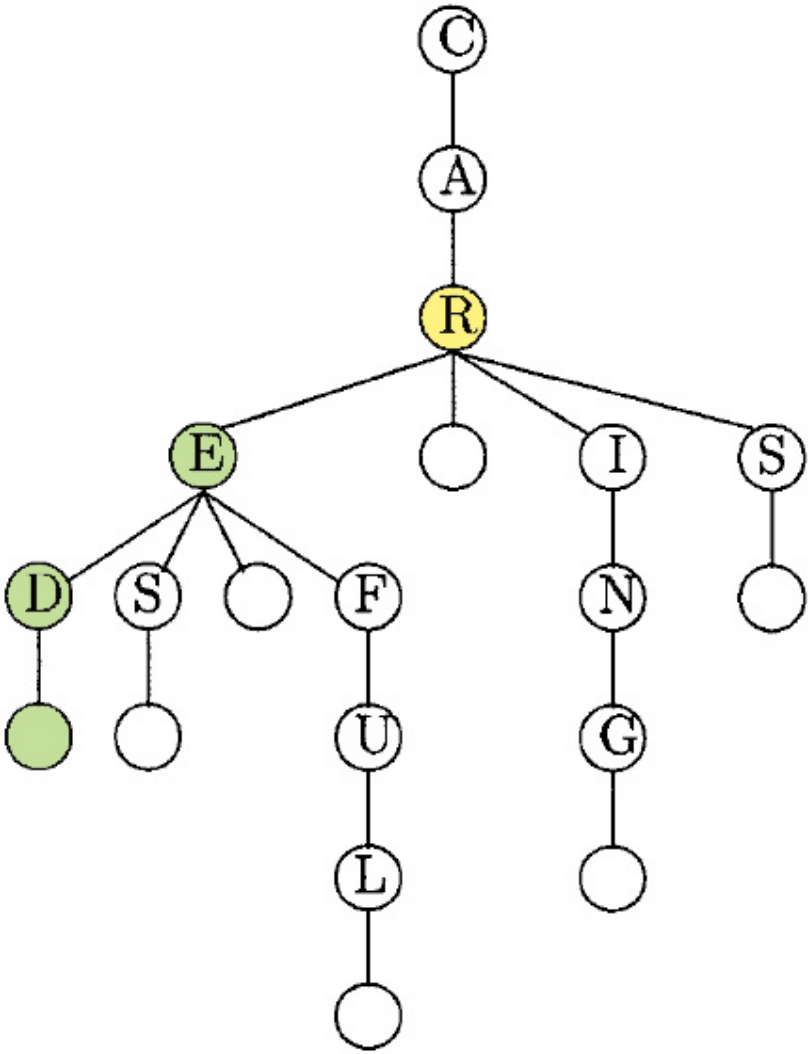
The trie on left is the result of inserting the words CAR, CARE, CARED, CARES, CAREFUL, CARING, CARS

Identifying affixes



First we find a branching point at the `R` in `CAR`

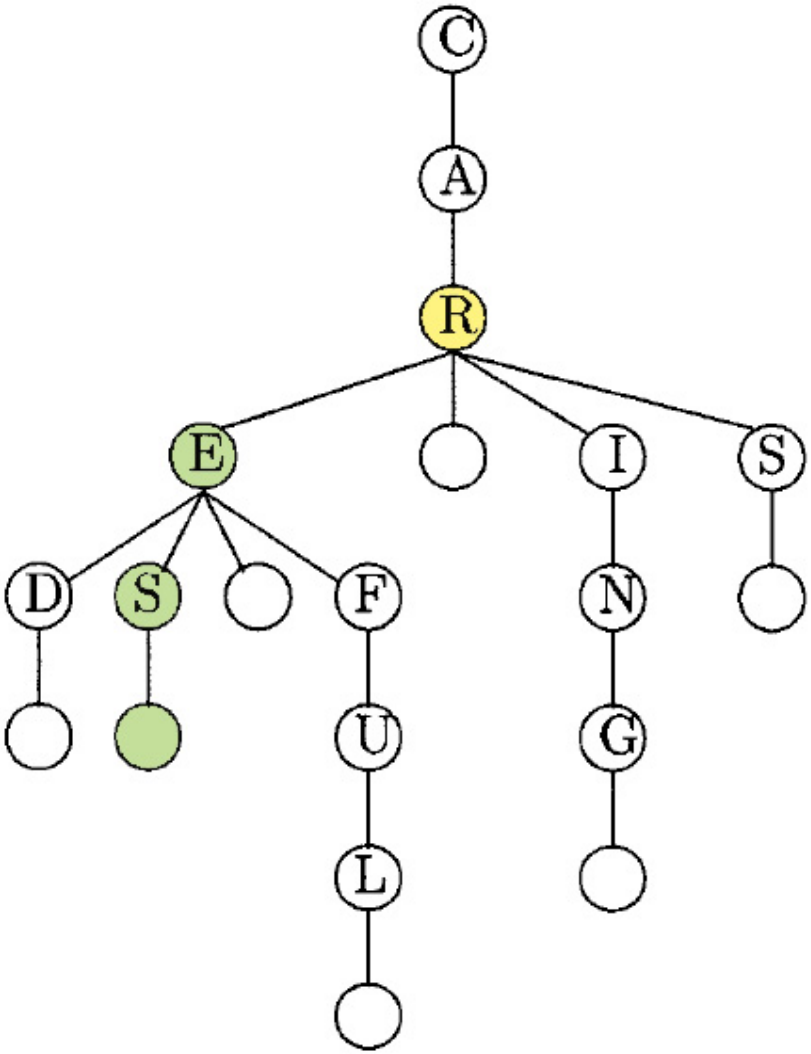
Identifying affixes



Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory	
ED	1

Identifying affixes

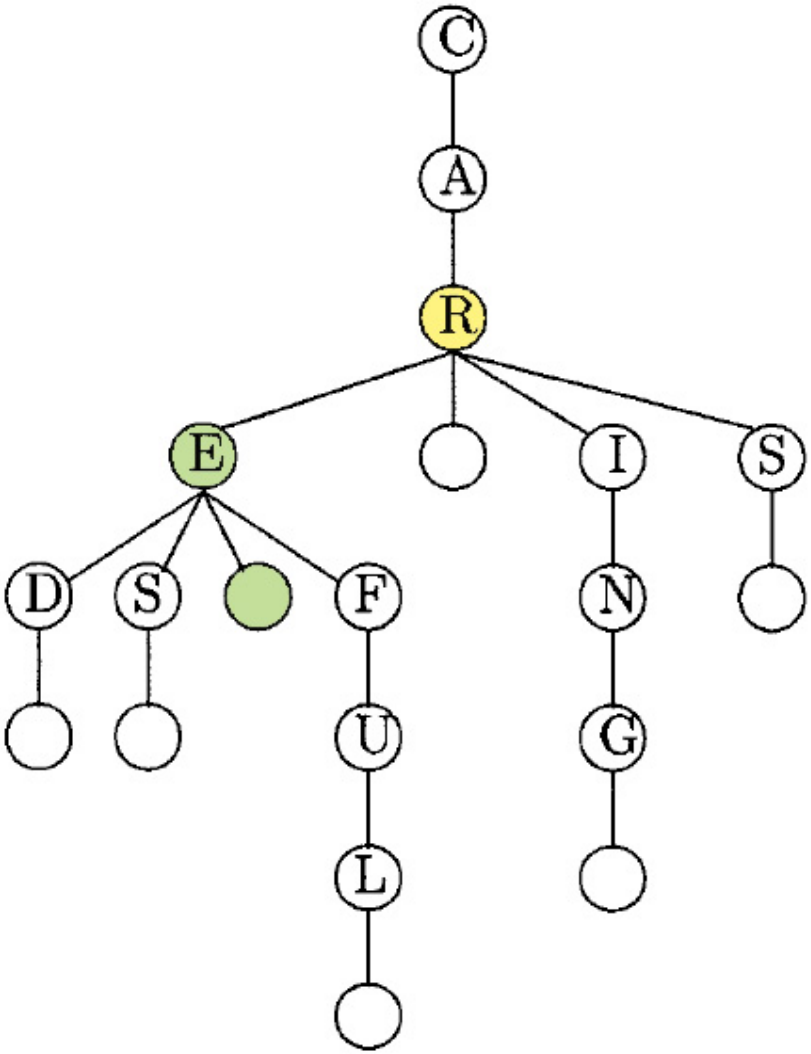


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1

Identifying affixes

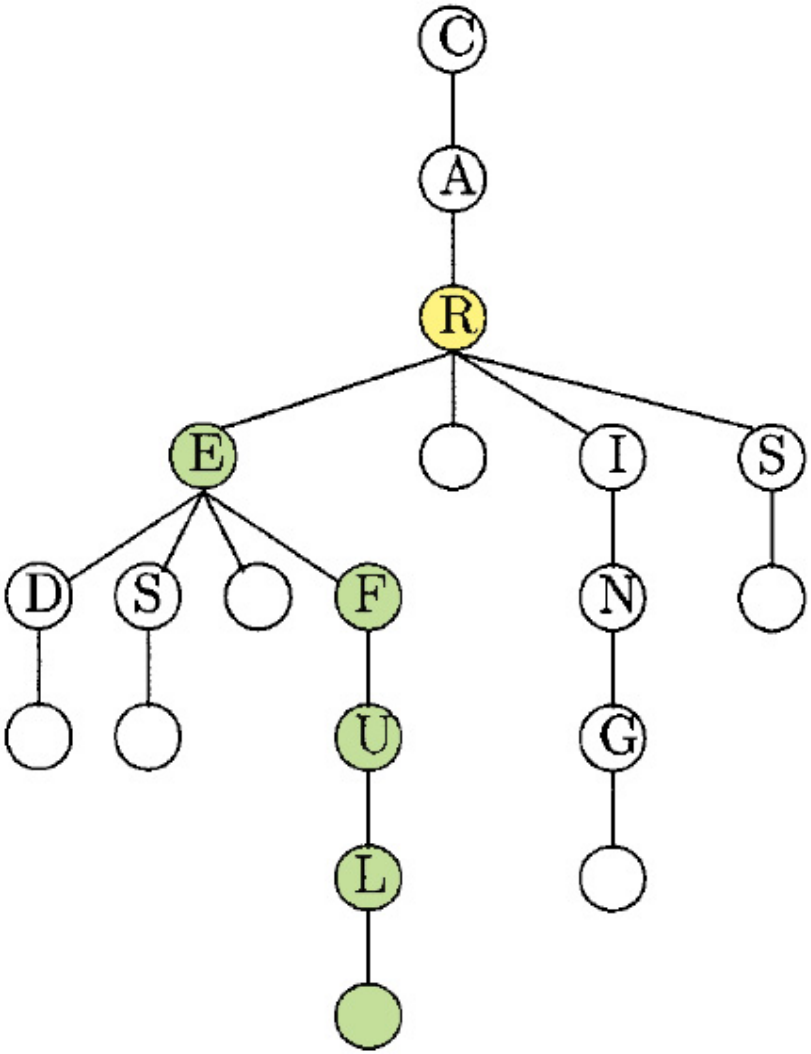


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1		

Identifying affixes

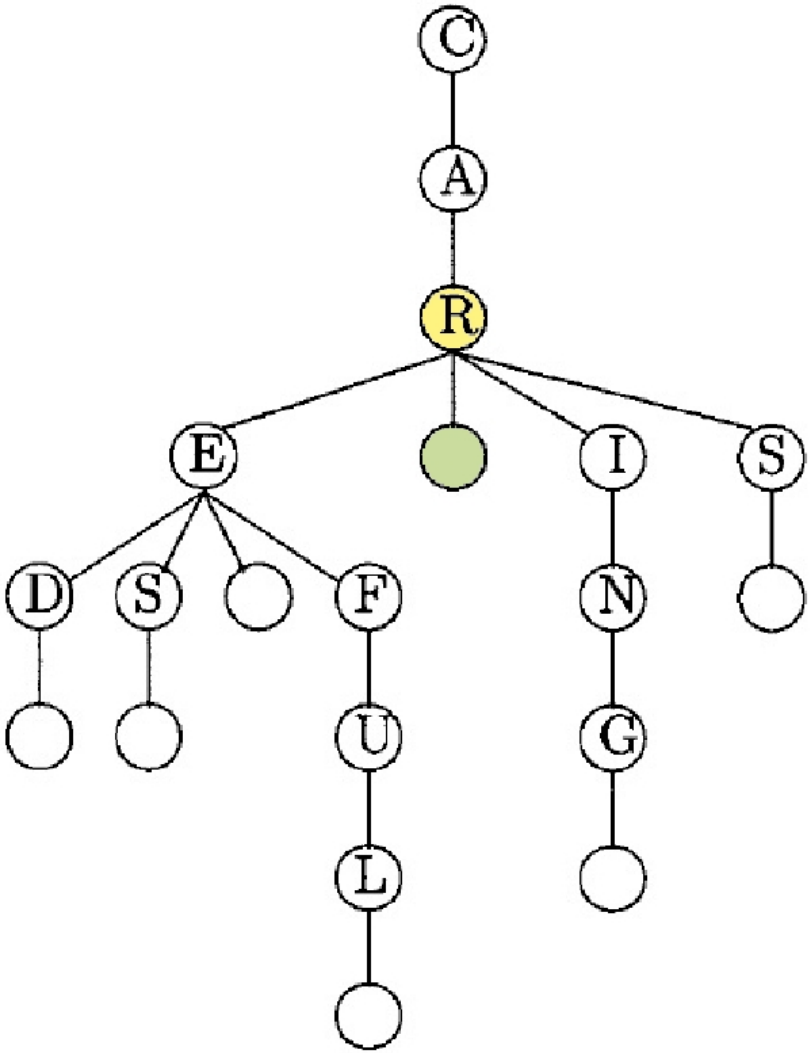


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1

Identifying affixes

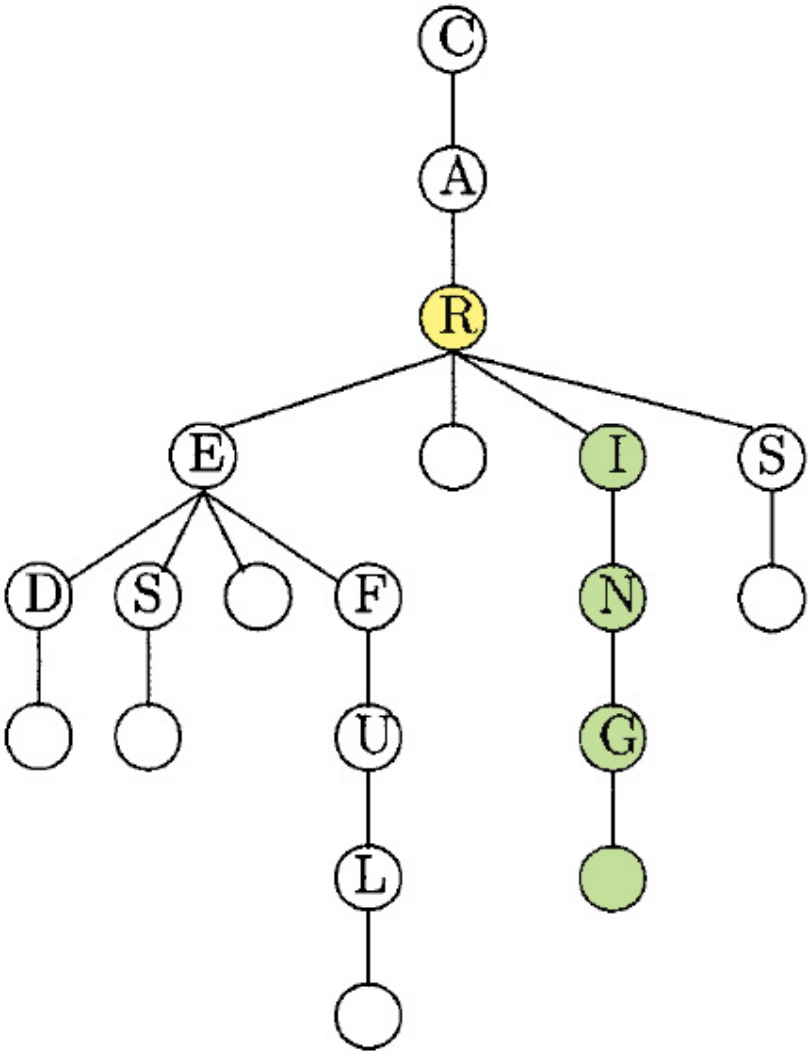


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1		

Identifying affixes

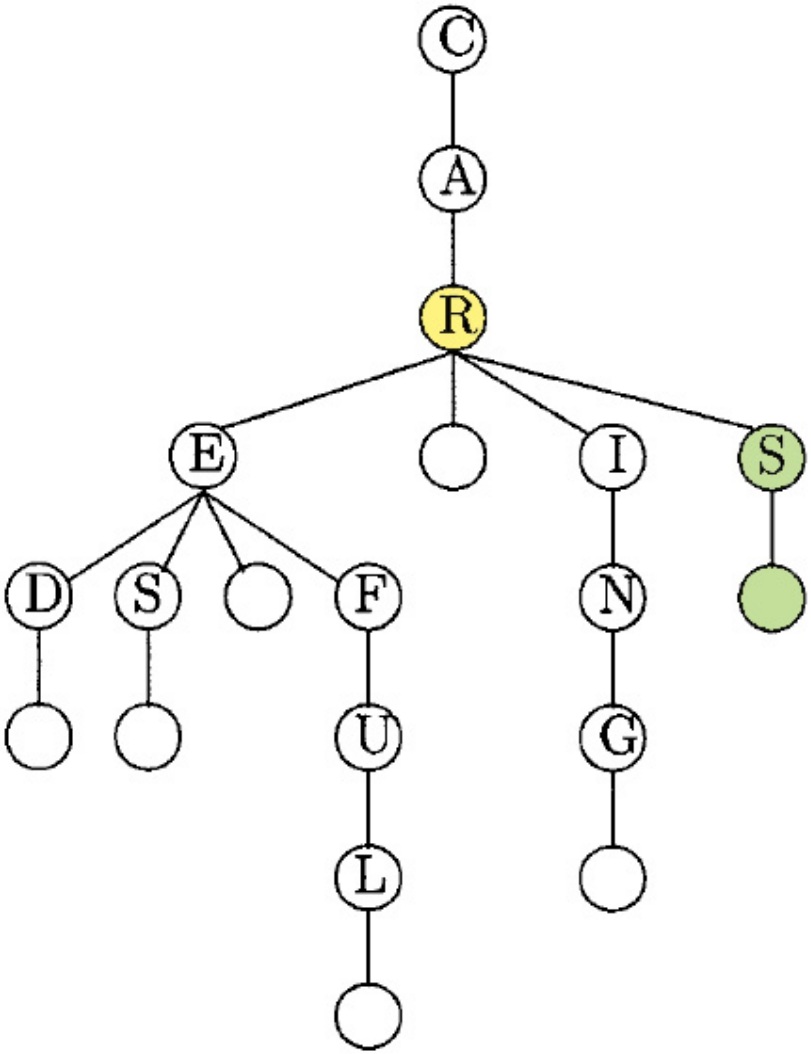


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1	ING	1

Identifying affixes

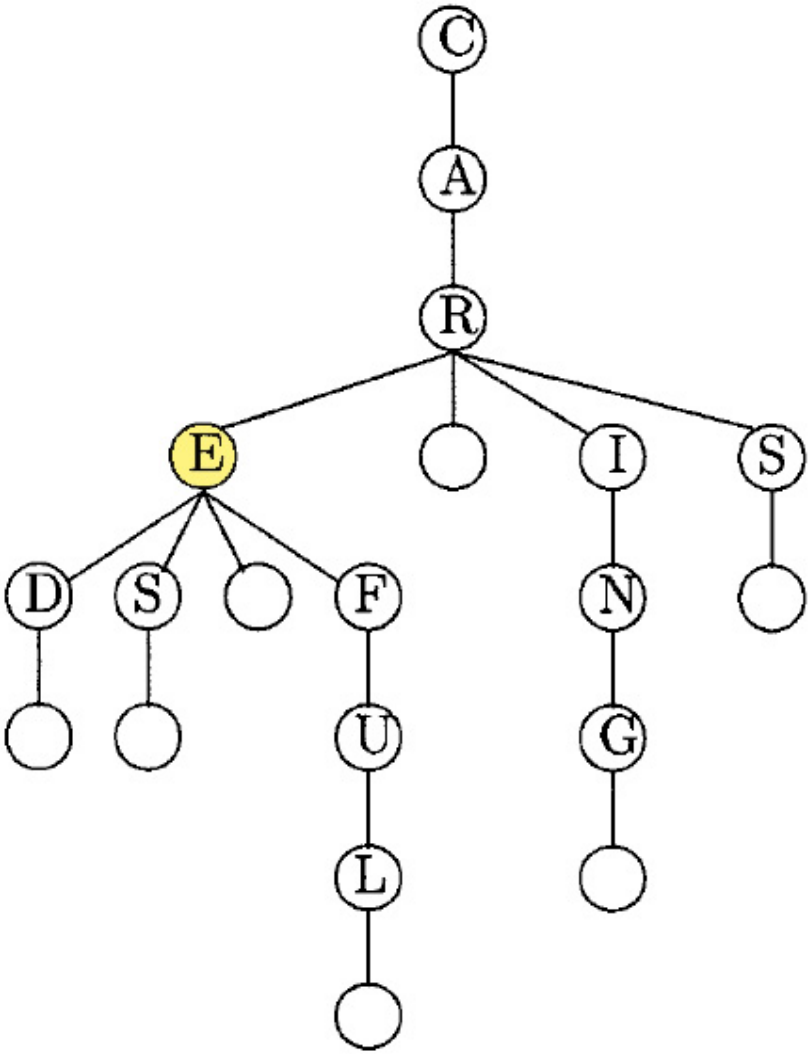


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1	ING	1
S	1		

Identifying affixes

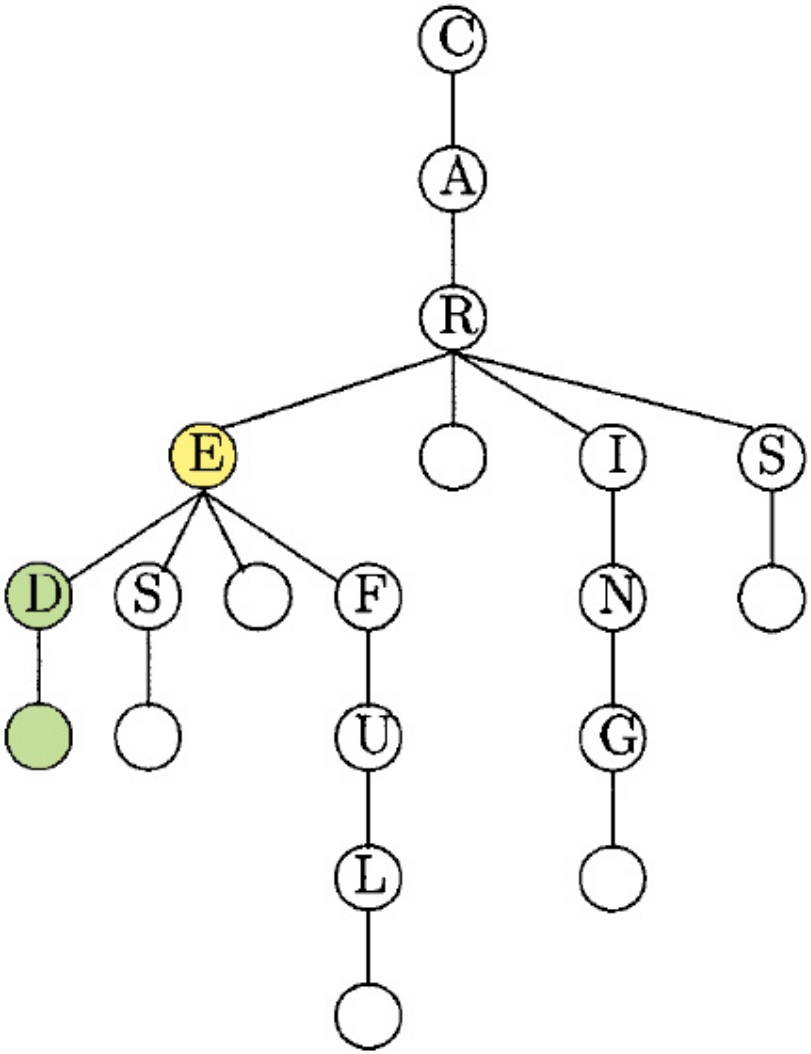


Then we find a branching point at the E in CARE

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1	ING	1
S	1		

Identifying affixes

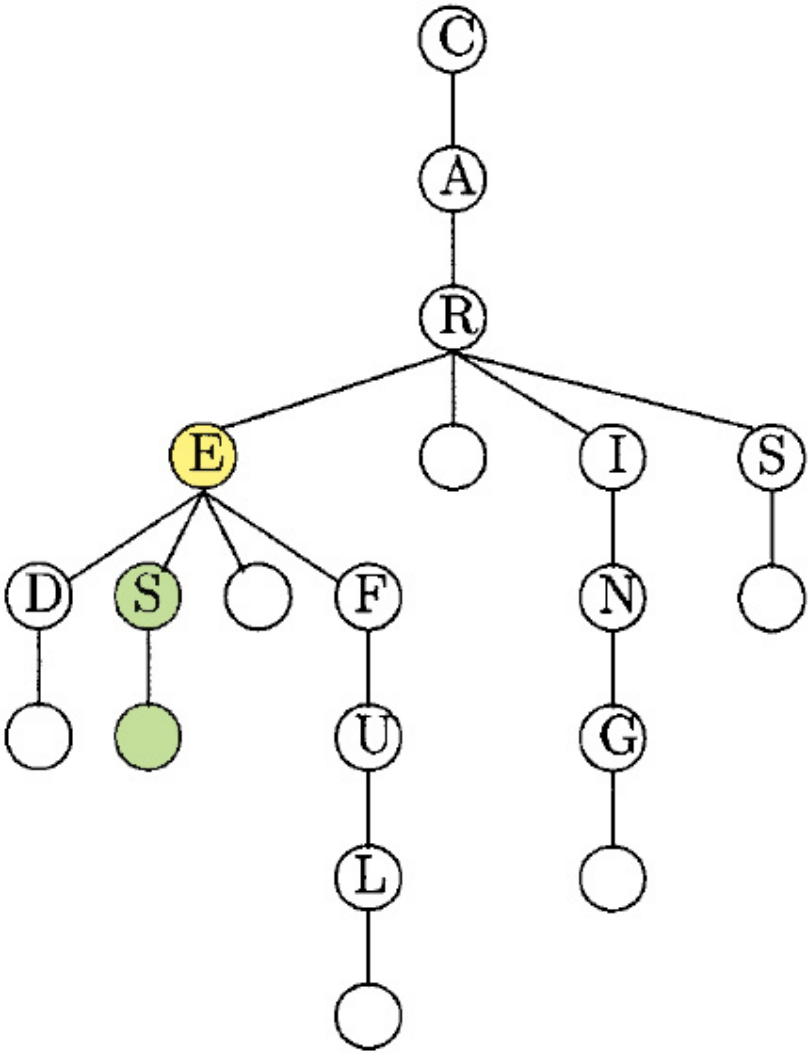


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1	ING	1
S	1	D	1

Identifying affixes

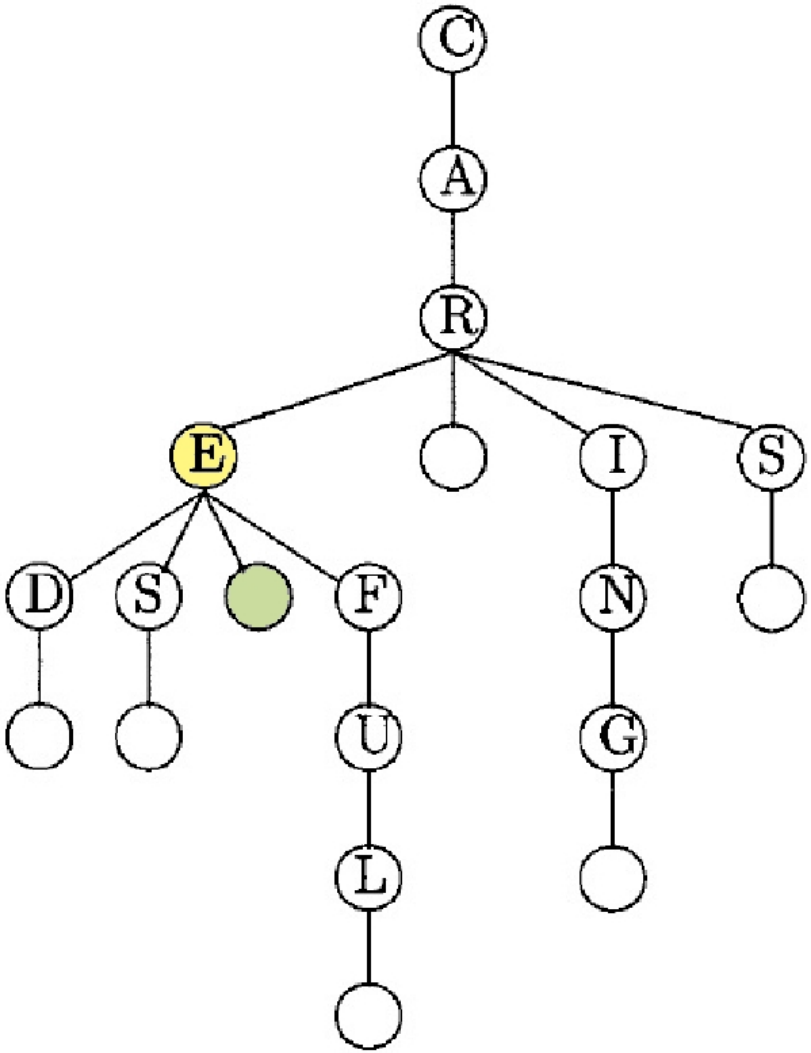


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	1	ING	1
S	2	D	1

Identifying affixes

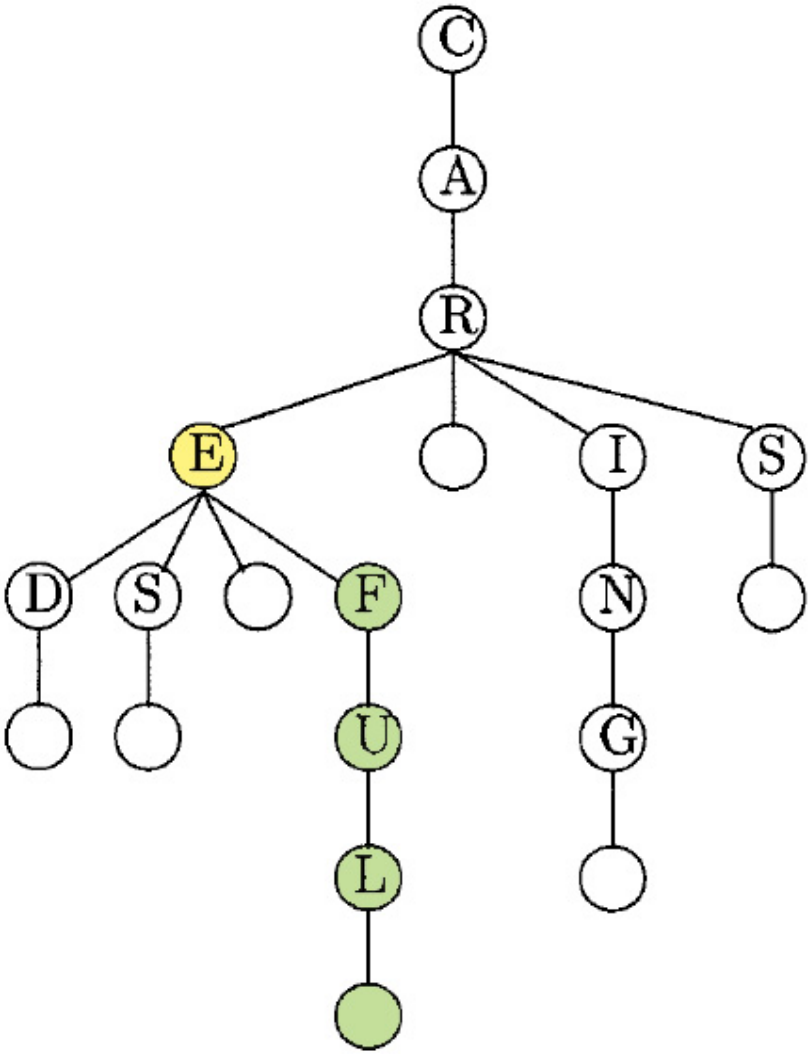


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	2	ING	1
S	2	D	1

Identifying affixes

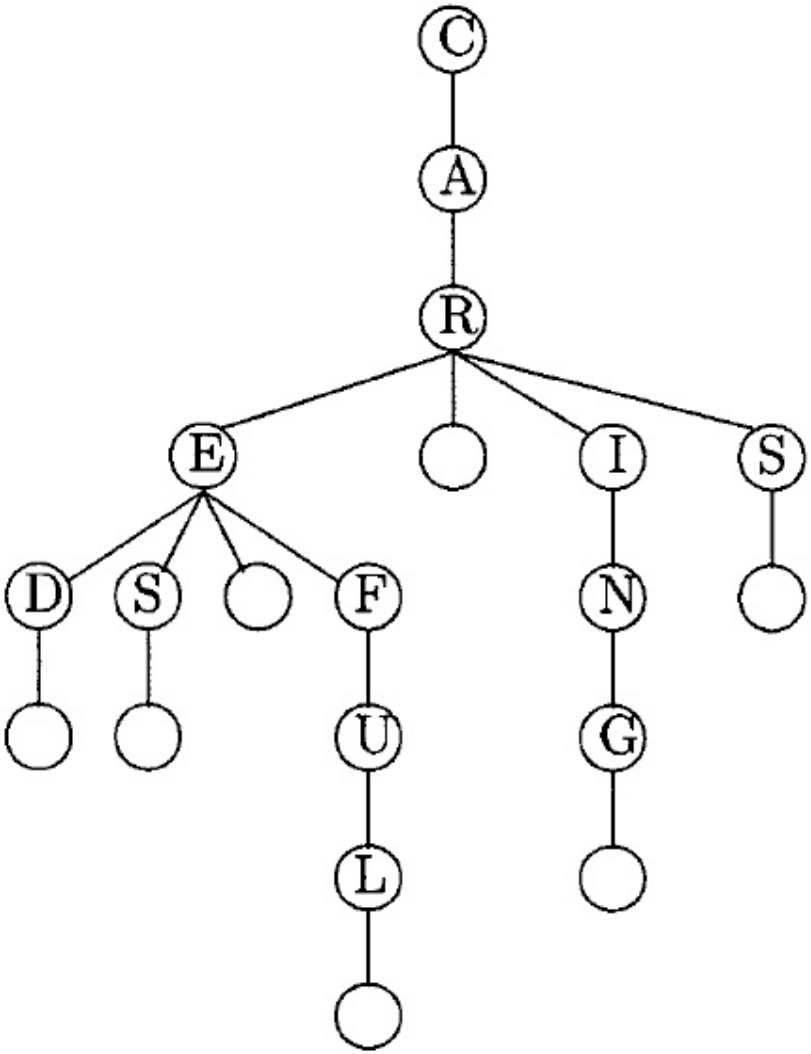


Then we add each remaining string to our affix inventory with its total count

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	2	ING	1
S	2	D	1
FUL	1		

Identifying affixes

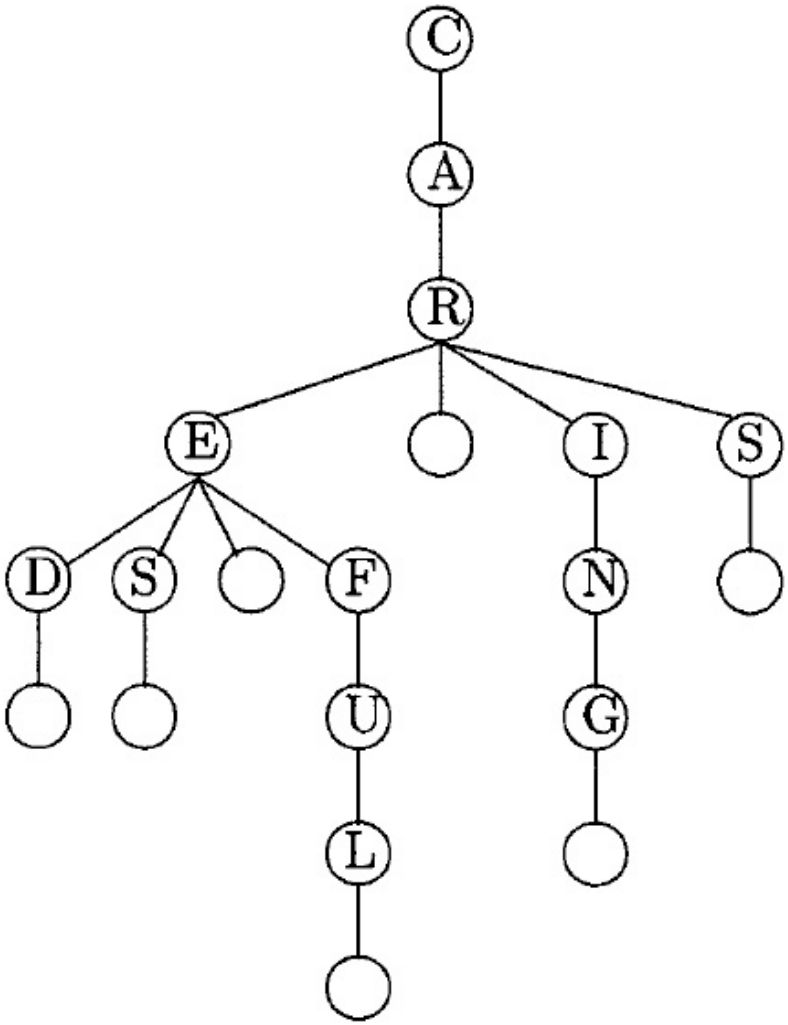


Keep only the K most frequent.
(Not really meaningful here.)

Candidate Suffix Inventory

ED	1	ES	1
E	1	EFUL	1
NULL	2	ING	1
S	2	D	1
FUL	1		

Finding PPMVs



- Identify all pairs of affixes which descend from the same node (e.g. {“s”, NULL}) and call these pairs **rules**
- Two words which share the same stem and affix rule form a **PPMV** (**p**air of **p**otential **m**orphological **v**ariants). For example, (“car”, “cars”)
- The **ruleset** of a rule is the set of all PPMVs that have that rule in common. Here, the ruleset of (“s”, NULL) would be the set {“cars/car”, “cares/care”}
- The algorithm finds the ruleset for each rule.

Computing Semantic Vectors

- Decide which of the rulesets that we have generated contain pairs of words which are semantically related.
- S&J don't compute cosine scores directly on each vector in the matrix; rather, they first apply singular value decomposition (SVD) to the matrix (aka Latent Semantic Analysis or LSA; Landauer et al 1988)
- LSA is beyond the scope of this discussion, but the general idea is that the matrix is projected (compressed) into a lower k-dimensional subspace such that the k dimensions of this new subspace are the k most informative dimensions.
- This results in a matrix of "semantic vectors".

Comparing Semantic Vectors

To determine if a pair of words in a PPMV are semantically related, they define a normalized cosine score, or **NCS**, written as $\overline{cos}$ between two semantic vectors, Ω_w and Ω_v.

To compute the NCS between to Ω_w and Ω_v:

1. Choose a set R of 200 random words
2. Compute the cosine similarity between Ω_w and each vector $\Omega_r \in R$, storing the mean (μ_w) and standard deviation (σ_w)
3. Repeat step 2, replacing Ω_w with Ω_v, computing μ_v and σ_v.

$$\overline{cos}(\Omega_v, \Omega_w) = \min_{y \in \{w,v\}} \left(\frac{cos(\Omega_v, \Omega_w) - \mu_y}{\sigma_y} \right)$$

Sample NCS scores

Below are normalized cosine scores computed by S&J:

PPMV	$\overline{cos}$
ally/allies	6.5
car/cars	5.6
dirty/dirt	2.4
rating/rate	0.97
car/cares	-0.14
car/caring	-0.71
car/cared	-0.96
ally/all	-1.3

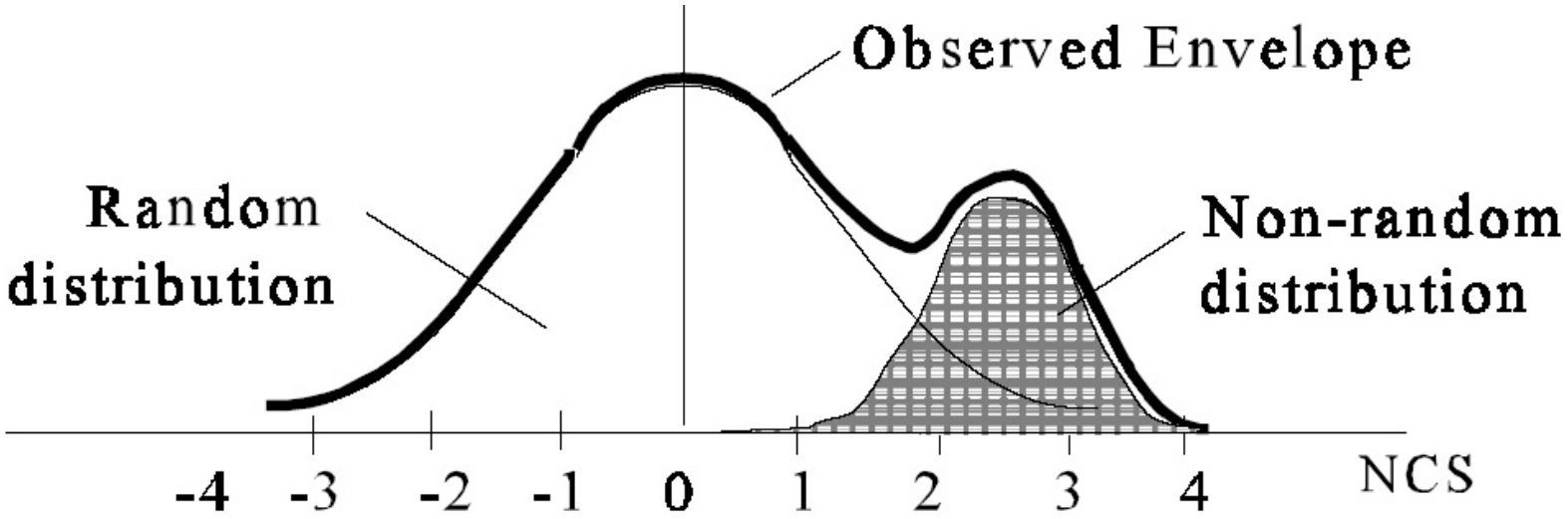
A score over 2.0 would be rare for a random event.

Ruleset-level Statistics

- We need to determine if a rule is valid, e.g. (“s”, NULL) [24946 unique pairs in BNC], or invalid, e.g. (“e”,”age”) {(“home”,“homage”), (“trie”, “triage”)} [115 unique pairs]
- Compute the NCS for all PPMVs of a particular rule.
- The NCS scores for invalid PPMVs should be distributed normally $N(0,1)$ ($\mu = 0$, $\sigma^2 = 1$)
- Calculate $Pr(true)$, the probability that a particular ruleset is valid (non-random).

Visualizing $Pr(true)$

For a particular PPMV, $Pr(true)$ computes the probability that the NCS of the PPMV was generated by the non-random (shaded) distribution.



▸ Subrules ▸ Results

Subrules

Consider the rule (“es”, NULL)

- This rule pairs together “car/cares” which have a low NCS.
- But this rule is sometimes valid (“church/churches”, “mash/mashes”, “miss”, “misses”)
- The problem is that we have to decide whether the rule (“es”, NULL) is valid based on members of the ruleset and there will be a lot of incorrect (“es”, NULL) matches (“hat/hates”, “cap/capes”, “sit/sites”)...
- ...So how can we remedy this?

Subrules

- Based on the intuition that these rules are phonological variations of other rules, we might expect to find that the (“es”, NULL) rule applies in only specific cases.
- If so, there should be specific environments where we’d find that there were higher than average NCS scores:

Rule/Subrule	Average	Std Dev	# instances
(“es”, NULL)	1.62	2.43	173
(“ches”, “ch”)	2.20	1.66	32
(“shes”, “sh”)	2.39	1.52	15
(“res”, “r”)	-0.69	0.47	6
(“tes”, “t”)	-0.58	0.93	11

Results

- Notice that the algorithm as we've described it so far, determines whether a pair of words (a PPMV) is valid.
- For pairs such as "concerning/concerned", "concerns/concerning" we might find both pairs are valid.
- However, reference data lists "concern" as the root of these forms
- So, they score their "conflation sets", for example {concern, concerned, concerns, concerning} (details skipped)

Results

- Recall that they set a threshold for determining whether or not to believe that a particular PPMV in a rule set is non-random.
- S&J(2001) call this value T_5.
- Higher values of T_5 make the algorithm more conservative.
- The choice of this value becomes irrelevant in S&J'01.

	(Goldsmith) Linguistica	S & J $T_5 = 0.5$	S & J $T_5 = 0.7$	S & J $T_5 = 0.85$
Precision	83.0%	85.0%	90.0%	92.6%
Recall	80.4%	81.8%	79.3%	76.6%
F-Score	81.6%	83.4%	**84.3%**	83.9%

Insights from Schone 2001

Schone evaluates standalone semantics on 5 languages in his thesis.

- English is “missing almost all irregular word forms”
- Spanish has “an incredible problem with deletions” (words omitted from the conflation sets)
- Dutch beats the baseline (no analysis) by only 10%

Pause

Overview

- Extends the Schone and Jurafsky (2000) work
- Includes additional measures because of the shortcomings of semantics alone.
- (“reusability”, “use”) is labeled as a morphological variant but is discarded since the words are not semantically similar enough.
- (“as”, “a”) is deemed acceptable because, since they appear so frequently, neither has much semantic information, so, in that respect, they are semantically very similar.
- Introduction of bad rules: “ho-/$\oslash$” $\Rightarrow$ “pi-/$\oslash$” for “hog/pig” which have very similar semantics [81 unique pairs].

Overview

1. Identify PPMV's, create semantic vectors and build conflation sets. (This is nearly identical to S&J'00)
2. Compute a probability based on the frequency of the rules
3. Compute a probability based on local syntactic information
4. Expand conflation sets using transitivity
5. Combine steps 1, 2, 3, and 4 to get the final answer

Example: Potential Rules

Rank	English	German	Dutch
1	-s⇒∅	-n⇒∅	-en⇒∅
2	-ed⇒-ing	-en⇒∅	-e⇒∅
4	-ing⇒∅	-s⇒∅	-n⇒∅
8	-ly⇒∅	-en⇒-t	de-⇒∅
12	C-⇒c-	-en⇒-te	-er⇒∅
16	re-⇒∅	1-⇒∅	-r⇒∅
20	-ers⇒-ing	er-⇒∅	V-⇒v-
24	1-⇒∅	1-⇒2-	-ingen⇒-e
28	-d⇒-r	ge-/-t⇒-en	ge-⇒-e
32	s-⇒∅	D-⇒d-	-n⇒-rs

Example: Potential Rules

Rank	English	German	Dutch
1	-s⇒∅	-n⇒∅	-en⇒∅
2	-ed⇒-ing	-en⇒∅	-e⇒∅
4	-ing⇒∅	-s⇒∅	-n⇒∅
8	-ly⇒∅	-en⇒-t	de-⇒∅
12	C-⇒c-	-en⇒-te	-er⇒∅
16	re-⇒∅	1-⇒∅	-r⇒∅
20	-ers⇒-ing	er-⇒∅	V-⇒v-
24	1-⇒∅	1-⇒2-	-ingen⇒-e
28	-d⇒-r	ge-/-t⇒-en	ge-⇒-e
32	s-⇒∅	D-⇒d-	-n⇒-rs

Valid circumfix in German

Example: Potential Rules

Rank	English	German	Dutch
1	-s⇒∅	-n⇒∅	-en⇒∅
2	-ed⇒-ing	-en⇒∅	-e⇒∅
4	-ing⇒∅	-s⇒∅	-n⇒∅
8	-ly⇒∅	-en⇒-t	de-⇒∅
12	C-⇒c-	-en⇒-te	-er⇒∅
16	re-⇒∅	1-⇒∅	-r⇒∅
20	-ers⇒-ing	er-⇒∅	V-⇒v-
24	1-⇒∅	1-⇒2-	-ingen⇒-e
28	-d⇒-r	ge-/-t⇒-en	ge-⇒-e
32	s-⇒∅	D-⇒d-	-n⇒-rs

Capitalization as a prefix

Example: Potential Rules

Rank	English	German	Dutch
1	-s⇒∅	-n⇒∅	-en⇒∅
2	-ed⇒-ing	-en⇒∅	-e⇒∅
4	-ing⇒∅	-s⇒∅	-n⇒∅
8	-ly⇒∅	-en⇒-t	de-⇒∅
12	C-⇒c-	-en⇒-te	-er⇒∅
16	re-⇒∅	1-⇒∅	-r⇒∅
20	-ers⇒-ing	er-⇒∅	V-⇒v-
24	1-⇒∅	1-⇒2-	-ingen⇒-e
28	-d⇒-r	ge-/-t⇒-en	ge-⇒-e
32	s-⇒∅	D-⇒d-	-n⇒-rs

Valid for some pairs (“baker/baked”, “adapter/adapted”)
But not for others (“fancier/fancied”, “bead/bear”)

Example: Potential Rules

Rank	English	German	Dutch
1	-s⇒∅	-n⇒∅	-en⇒∅
2	-ed⇒-ing	-en⇒∅	-e⇒∅
4	-ing⇒∅	-s⇒∅	-n⇒∅
8	-ly⇒∅	-en⇒-t	de-⇒∅
12	C-⇒c-	-en⇒-te	-er⇒∅
16	re-⇒∅	1-⇒∅	-r⇒∅
20	-ers⇒-ing	er-⇒∅	V-⇒v-
24	1-⇒∅	1-⇒2-	-ingen⇒-e
28	-d⇒-r	ge-/-t⇒-en	ge-⇒-e
32	s-⇒∅	D-⇒d-	-n⇒-rs

Never valid (“age/sage”, “eating/seating”, “lump/slump”)
There are 2198 of these pairs in the BNC.

Semantics

- Once we have the potential rules, the computation of the semantics is exactly as in SJ'00
- Also, to disambiguate with their new measures, they now call this measure Pr_{Sem} (the semantic probability)

Affix Frequencies: Pr_{Orth}

- Why? Rules which occur very frequently are very likely to be valid.
- According to their experiment, 99.7% of the PPMVs they propose in the ruleset s-/$\oslash$ are correct; however, their semantic measure selects only 92% of these PPMVs.
- Semanitcs seems to hurt the top 15 rules; after that, the semantics work well (Schone, 2001).
- Pr_{S-O} (combination of Pr_{Sem} and Pr_{Orth} gets them 3% more of the correct PPMVs than semantics alone (e.g. from 92% to 95%)

Local Syntactic Context: Computing Pr_{Syntax}

Using semantics alone, S&J would deem the bottom four PPMVs invalid. (I think Vegas/Vega should be invalid)

Word+s	Word	Pr	Word+s	Word	Pr
agendas	agenda	.968	legends	legend	.981
ideas	idea	.974	militias	militia	1.00
pleas	plea	1.00	guerrillas	guerrilla	1.00
seas	sea	1.00	formulas	formula	1.00
areas	area	1.00	railroads	railroad	1.00
Areas	Area	.721	pads	pad	.731
Vegas	Vega	.641	feeds	feed	.543

Local Syntactic Context: Computing Pr_{Syntax}

- To overcome this problem, we will again look at context.
- In Pr_{Sem}, we compared the context of the left half of a PPMV (e.g. “pads”) against the right half (e.g. “pad”).
- Now, we are going to compare the local context (± 2 words) of the left and right halves of our below threshold PPMV against the left and right halves of above threshold PPMVs, e.g. comparing “pads” to “railroads” and “pad” to “railroad”
- We’ll compute a probability, Pr_{Syntax} which correlates with how good a match we have against the above threshold PPMVs

Syntactic Context: Computing Pr_{Syntax}

Some sample contexts for the rule "-s $\Rightarrow \oslash$" :

Context for "-s"		Context for $\oslash$	
agendas are	seas were	a legend	this formula
two red pads	pleas have	militia is	an area
these ideas	other areas	railroad has	A guerrilla

- Note that the words we've retained in our context window (in blue) are the words the help disambiguate plural from singular.
- If a below threshold PPMV shows a similar pattern, then we are more likely to believe it is a morphologically valid pair.

Allomorphy

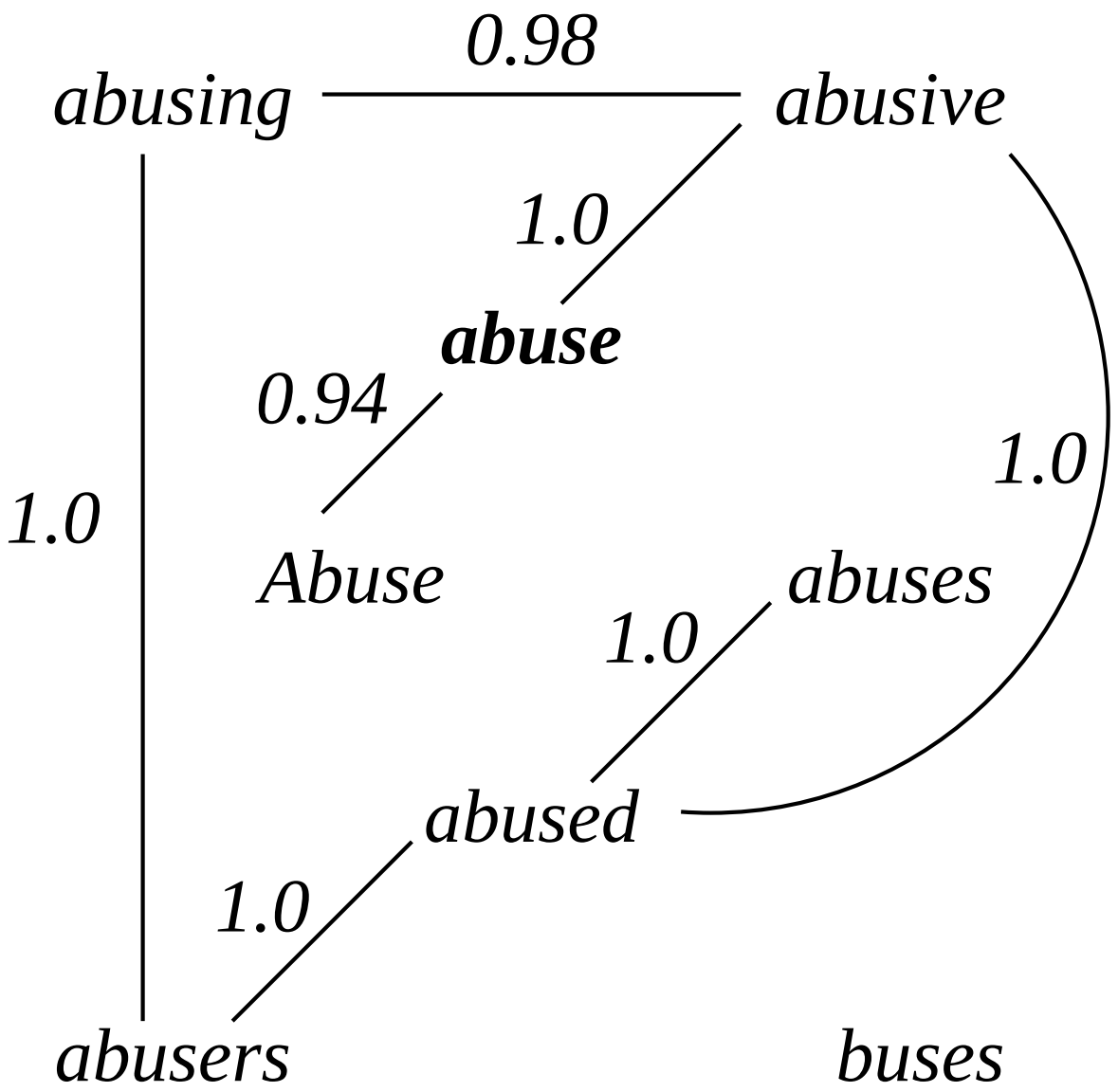
- Note that the ruleset “s”/NULL is not the only plural/singular pattern.
- We should be able to compare that rule’s ruleset against other rules’ rulesets to see if they, too, are modeling the same syntactic behavior

Rule	“Relative”	NCS	Rule	“Relative”	NCS
-s/⊘	-ies/-y	83.8	-ed/⊘	-d/⊘	95.5
-s/⊘	-es/⊘	79.5	-ing/⊘	-ing/-e	94.3
-ed/⊘	-ied/-y	81.9	-ing/⊘	-ting/⊘	.707

Transitive Closure

- If we knew that “abusers/abused” was a valid PPMV...
- ...and we knew that “abuses/abused” was a valid PPMV...
- ...but “abuses/abusers” was deemed invalid because it failed our other similarity measures...
- ...we would like to realize attempt to link “abuses” and “abusers” based on our validated link between each of those words with “abused”.

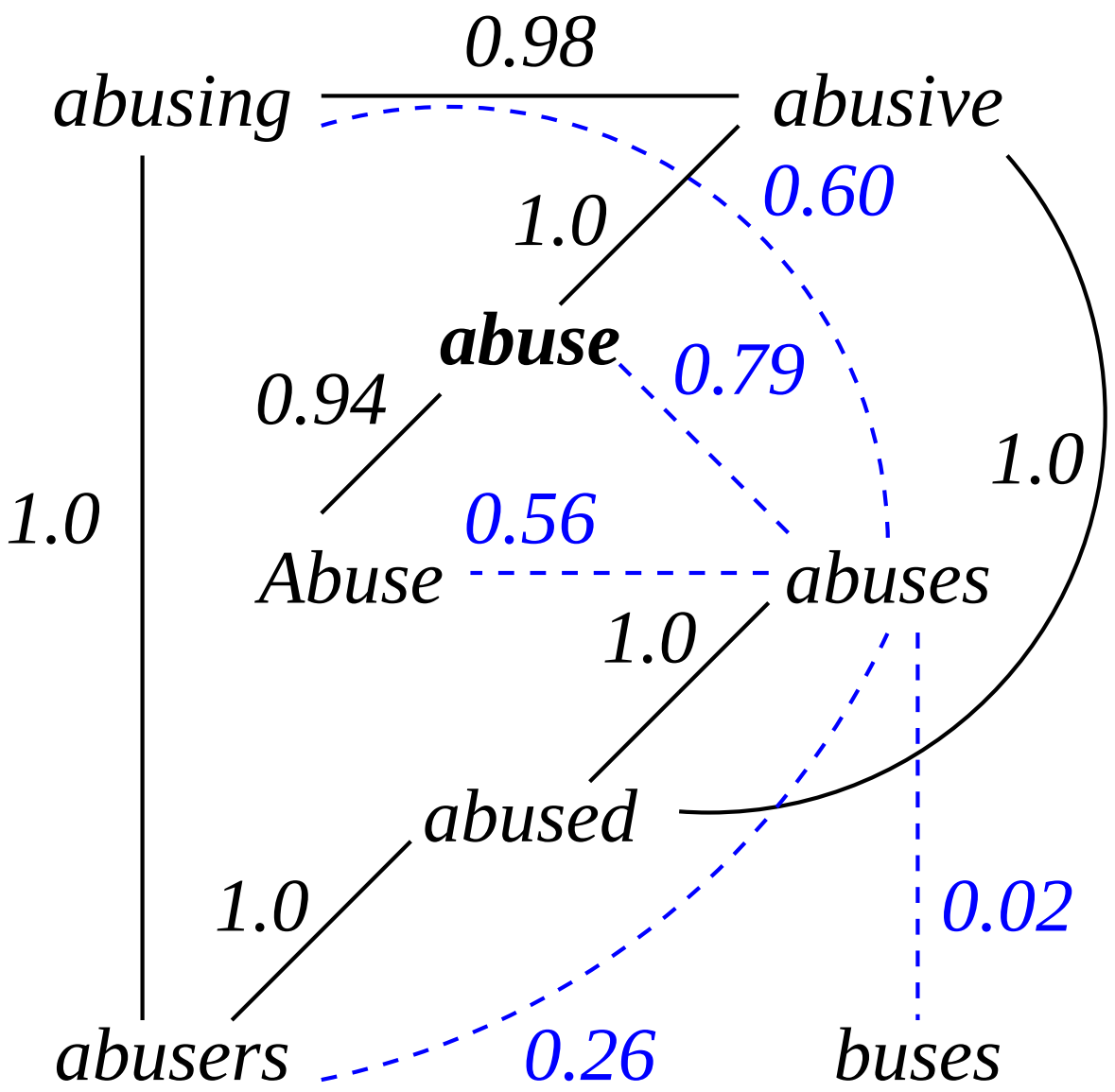
Transitive Closure



After computing $Pr(\text{valid})$, we can create an undirected graph showing the probabilities between PPMVs.

Here, for example, $Pr(\text{abuse} \Rightarrow \text{abusive}) = 1.0$

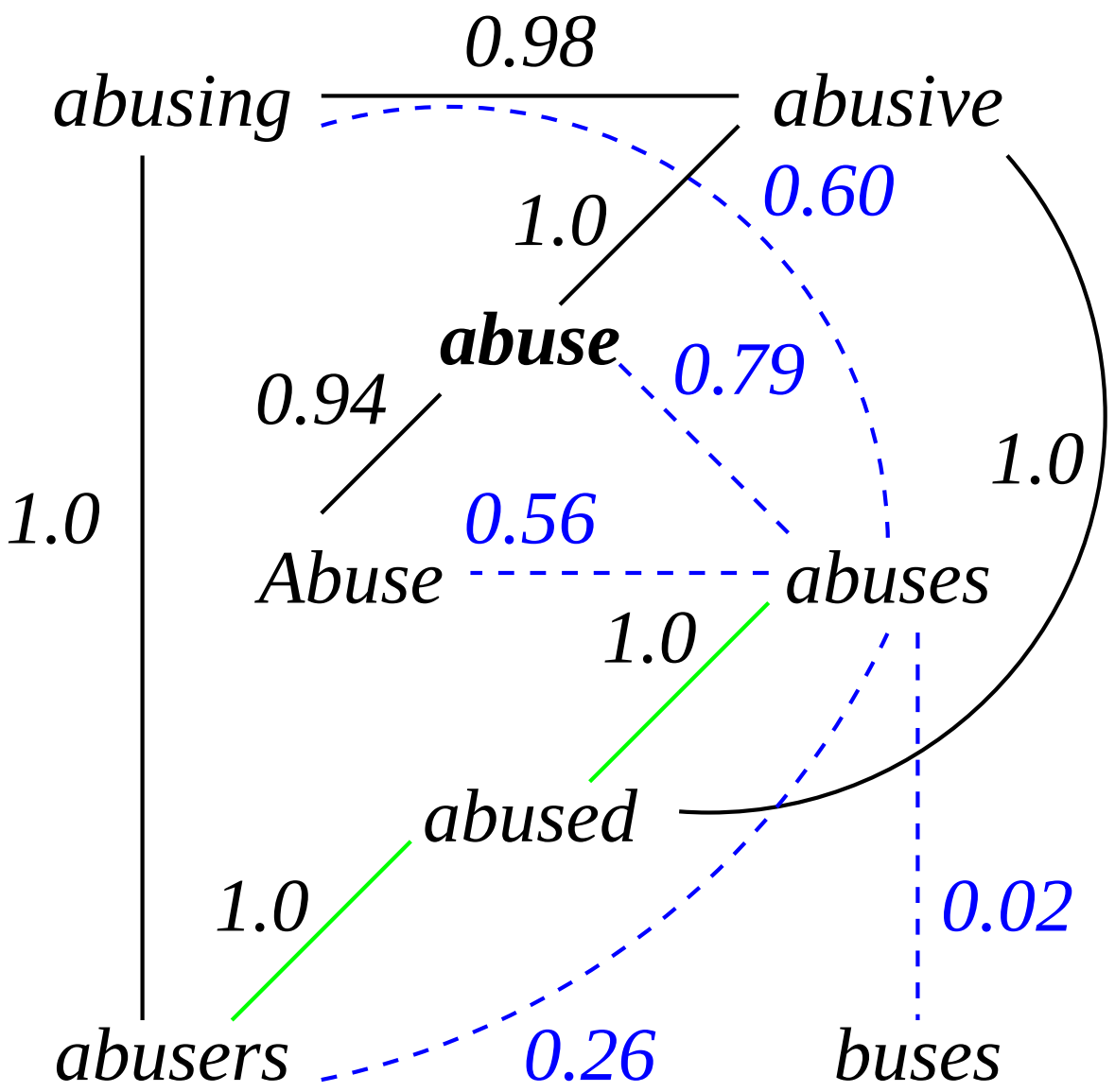
Transitive Closure



However, there are still valid pairs (X $\Rightarrow$ Y) which may be below our threshold of 0.85

Here, for example, "abuses" and "abusers" appear unrelated.

Transitive Closure



But if we can find a path (X $\Rightarrow$ Z) and (Z $\Rightarrow$ Y) such that both paths are above a threshold...

And such that the necessary rule "s $\Rightarrow$ rs" already exists in our inventory...

Then we deem it an acceptable PPMV.

Evaluation

- As in SJ'00, the evaluation is done using the conflation sets
- Evaluation is done in English, German and Dutch
- Graded separately for whether circumfixes were detected ("C") versus when only suffixes were detected ("S").
- Compares against a baseline ("None") where each word is its own conflation set
- Compares against SJ'00 (equivalent to SJ'01's Pr_{Sem})

Evaluation

	English		German		Dutch	
	S	C	S	C	S	C
Baseline	62.8	59.9	75.8	63.0	74.2	70.0
Goldsmith	81.8		84.0		75.8	
SJ'00	85.2		88.3		82.2	
$+Pr_{Orth}$	85.7	82.2	89.3	76.1	84.5	78.9
$+Pr_{Syntax}$	87.5	84.0	91.6	78.2	85.6	79.4
+transitive	**88.1**	84.5	**92.3**	78.9	**85.8**	79.6

Incorporation of these new features clearly helps, and seems to work reasonably well in German and Dutch; notice the circumfixation routine hurts performance.

Did we really get to the end?

Unbelievable!