

40 Years of Work in Computational Morphology

Richard Sproat*

Google, Research & Machine Intelligence, New York

AIMM 2019

Stony Brook University

<http://rws.xoba.com/aimm.pdf>

*with input from Christo Kirov and Kyle Gorman

Overview

- Some early work
- The finite state “revolution”
- Early neural systems
- More on finite-state methods
- Non-neural morphological induction
- The Deep Learning “revolution”

- Disclaimer:
 - I can't cover everything in 50 minutes so apologies if your favorite topic is skipped
 - There's a lot to cover so apologies if some of it goes by a bit quickly

Two early systems

- The Porter Stemmer (Porter, 1980)
 - Heavily used in early work on Information Retrieval
 - Main goals were reduction in size of lexicon and conflation of related forms
 - Rules for English were hard-coded in BCPL, an early precursor to C
- DECOMP (Allen, Hunnicutt & Klatt, 1987), which was much more informed by work in linguistics

DECOMP

- Described in Allen, Hunnicutt & Klatt (1987), but dating to the 1960's.
- Context: English Text-to-Speech synthesis (TTS) as part of the MITalk system
- Motivation:
 - Reduce storage: listing all words is too costly
 - Coverage: even a large lexicon cannot cover everything
 - Compute word pronunciations
- Approach:
 - Roots derived from Kučera and Francis (1967) - **early example of corpus-based methods**
 - Simple finite-state model of morphotactics
 - Right-to-left greedy partitioning
 - Hand tuned costs to favor analyses: e.g. favor suffixes and prefixes over roots
 - English spelling change rules: only a single letter at the end of a morph may be affected
- Claimed (by Klatt, 1987) to be able to handle 120K words from 12K entries

DECOMP example

scarcity

- Possible decompositions:
 - *scarce* + *ity*
 - *scar* + *city*
 - NB: only the former will give the right pronunciation (/skɛɹsɪti/)
- *scar*+*city* is tried first but involves two roots so is costly (cost = 234)
- *scarc(e)*+*ity* tried second and has a lower cost (cost = 136)

DECOMP is of some linguistic interest since it (along with other components of the MITalk system) was heavily influenced by early work on generative linguistics being done a few doors away.

Finite-state morphology: the key insight

C. Douglas Johnson (1972, p. 43):

“It is a well-established principle that any mapping whatever that can be computed by a finitely statable, well-defined procedure can be effected by a rewriting system ... Hence any theory which allows phonological rules to simulate arbitrary rewriting systems is seriously defective, for it asserts next to nothing about the sorts of mappings these rules can perform”

In other words: a model like *SPE* has essentially no predictive power.

But, in fact, virtually all phonological rules that had been formulated at the time were much more restricted than that: they represented *regular relations*

Regular languages and relations: 1 minute overview

Regular language over an alphabet Σ (from [Wikipedia](#)):

- The empty language \emptyset , and the empty string language $\{\epsilon\}$ are regular languages.
- For each $a \in \Sigma$ (a belongs to Σ), the [singleton](#) language $\{a\}$ is a regular language.
- If A and B are regular languages, then $A \cup B$ (union), $A \cdot B$ (concatenation), and A^* ([Kleene star](#)) are regular languages.
- No other languages over Σ are regular.

Regular relation:

- Take $a \in \{\Sigma \cup \epsilon\} \times \{\Sigma \cup \epsilon\}$; replace “language” in the above with “relation”

Important additional closure properties:

- Regular languages are closed under *intersection*
- Regular relations are closed under *composition* and *inversion*

Finite-state acceptors and transducers

- Regular languages are computable with *finite-state acceptors*
- Regular relations are computable with *finite-state transducers*
- The latter are just a generalization of the former so we will illustrate the latter with an example of a simple state machine that takes inputs and produces outputs.

A simple state machine

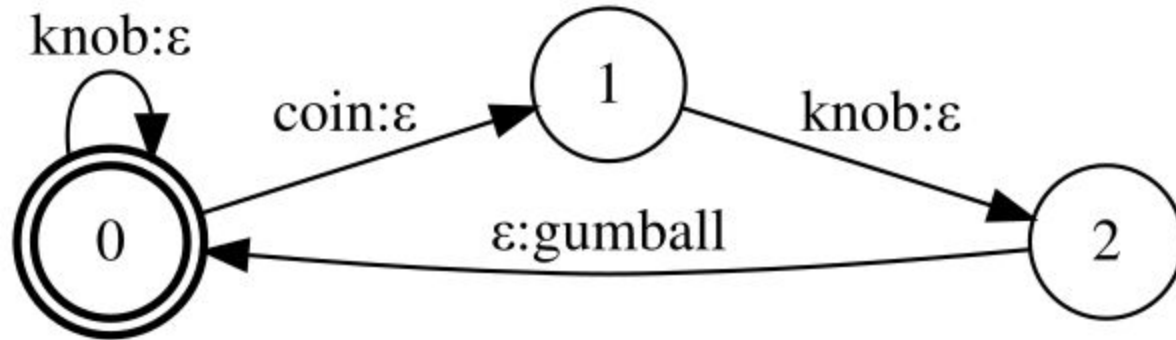


Credit: etsy.com

and thanks to
Kyle Gorman for
the example

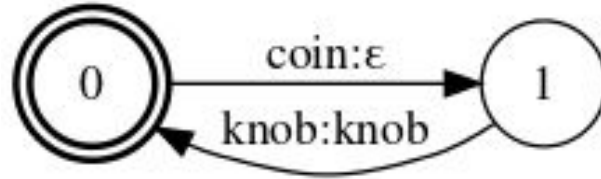
Finite state transducers

A simple state machine. Gumball machine G



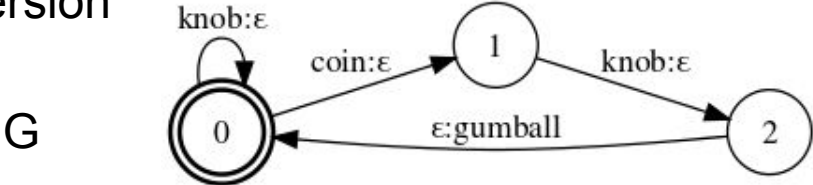
Finite-state transducers

Making a free version: Modification machine M

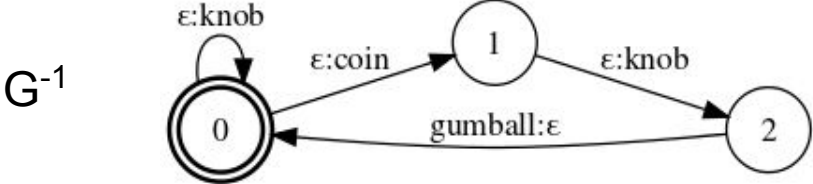


Finite-state transducers

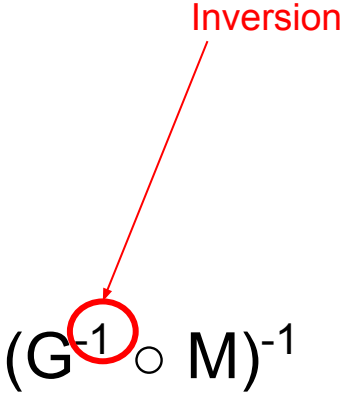
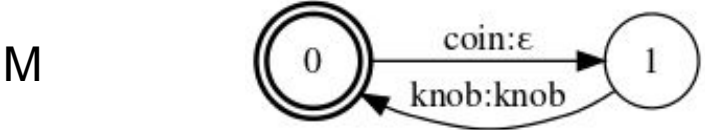
Making a free version



Inversion



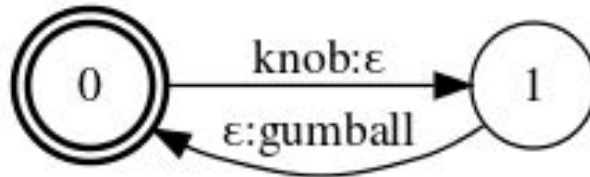
Composition



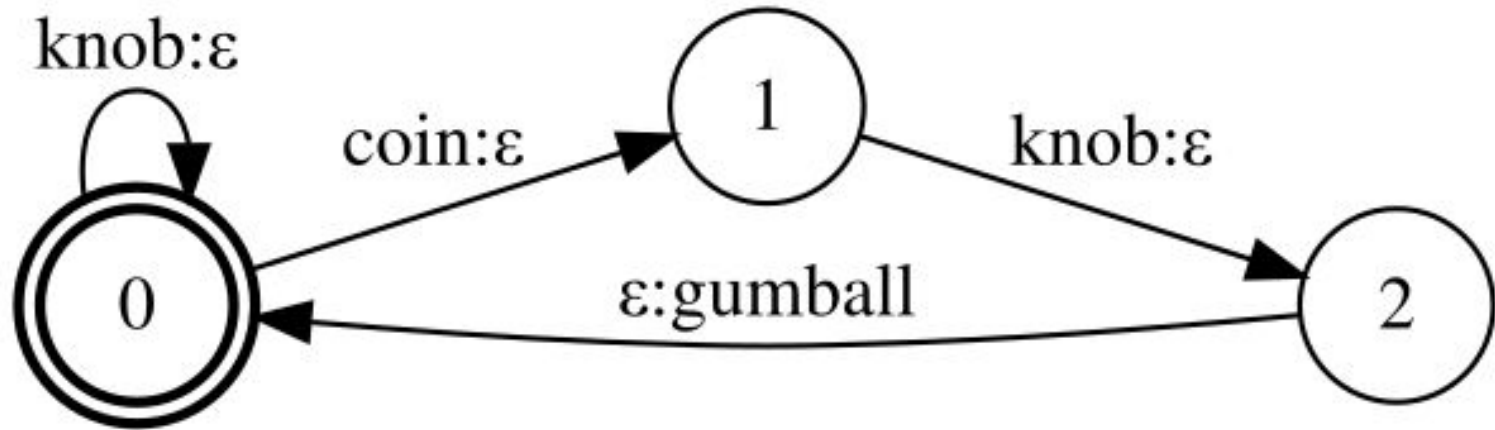
Finite-state transducers

Final free version

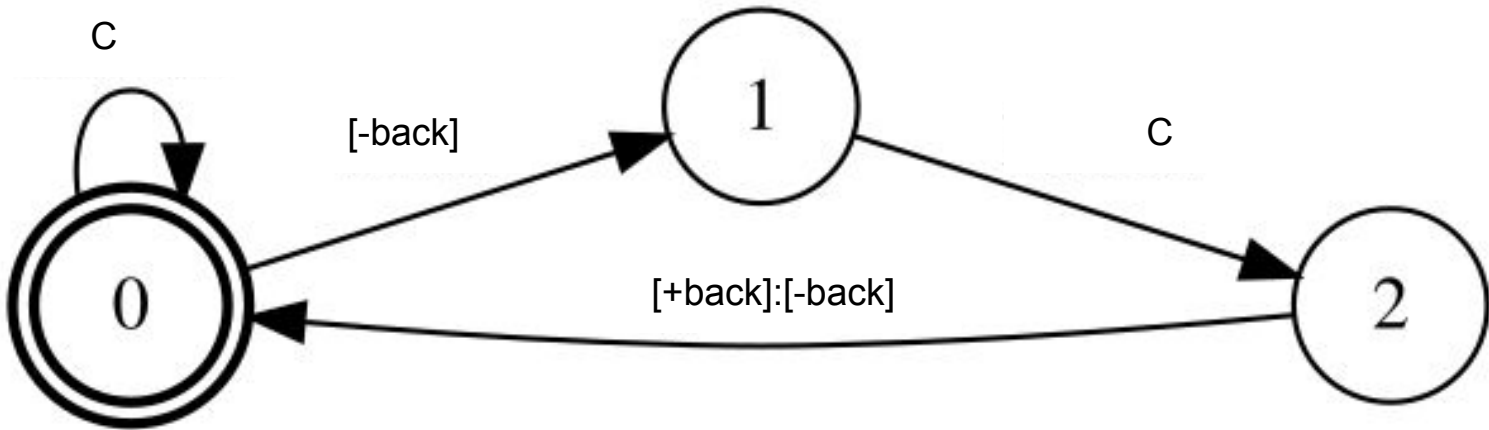
$$(G^{-1} \circ M)^{-1} =$$



Finite-state transducers



Finite state transducers: gumballs to vowel harmony



Finite-state transducers

- Returning to C.D. Johnson...
- FSTs can encode *cascades* of rules of the form:
 - $\varphi \rightarrow \psi / \lambda \text{ ___ } \rho$
- (Yes they can also implement OT)
- Weights on arcs can model probabilities, preferences, etc.
- So, if one has rules as above, one can *compile* them into FSTs ... if one has a compiler

Kaplan & Kay 1994

- Ron Kaplan and Martin Kay at Xerox PARC started working on this problem starting in the early 1980s
 - Actually they were not aware of Johnson's much earlier work when they started the project
- The algorithm was written up in an unpublished paper in 1983, which was only published in final form in 1994.
- The basic idea was as above:
 - Individual rules are implemented as FSTs
 - Rules in a phonological rule system would be cascaded together
- The problem was that machines at the time were way too underpowered.
 - Kaplan noted (1983, p.c.) that a non-trivial rule system ground a high-end "Dorado" to a halt:
 - A Dorado was about 16 Mhz or about 200 times slower than this Mac, with about 1/20,000 the amount of memory

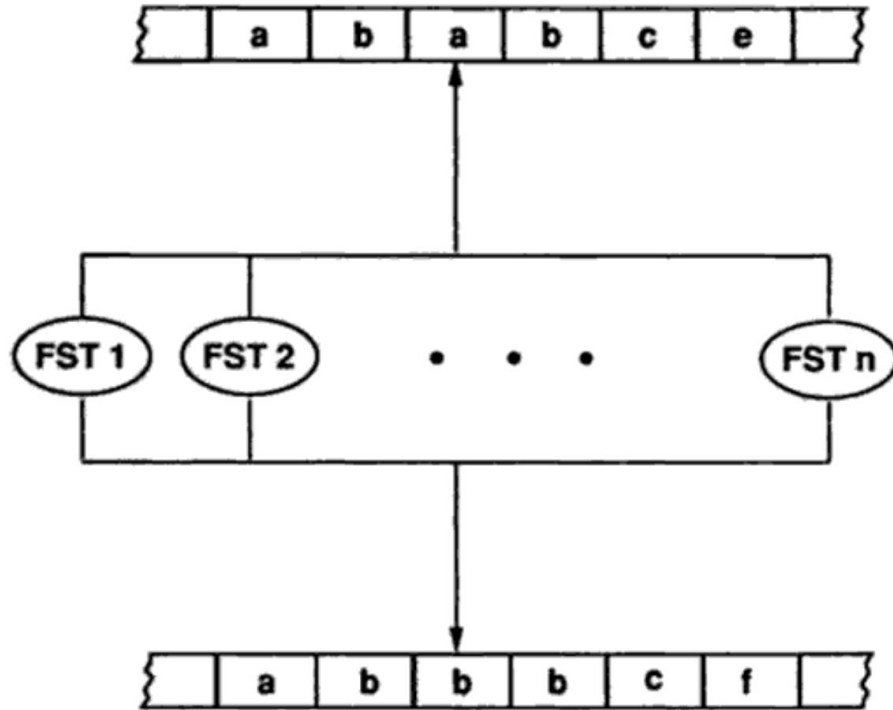
Koskenniemi 1983

- Kimmo Koskenniemi, then a PhD student at the University of Helsinki, wanted to build a morphological analyzer for Finnish:
 - Complex agglutinative affixation (mostly suffixation)
 - Productive phonological processes such as vowel harmony and consonant gradation
 - Arguably one of the first real NLP systems with wide coverage
- Given the (then) impracticality of Kaplan & Kay's approach, Koskenniemi came up with an alternative:
 - Don't compile a rule from a human-readable formulation: build the state machine by hand.
 - Avoid composition, since that was too expensive, instead:
 - Each *two-level* rule relates the *underlying* form to the *surface* form
 - All rules are synchronized by a process that is algorithmically equivalent to *intersection*
 - *k-length-difference-bounded* regular relations are closed under intersection (Roark & Sproat, 2007)

Two-level morphology

- Koskenniemi's "Two-level morphology" started a whole cottage industry through the mid 1990's, but it is important to bear in mind that it came about by a *historical accident*.
- Basic components:
 - Intersected two-level rules
 - Continuation lexica -- basically *tries* connected together

Two-level morphology: intersected FSTs



Two-level morphology: continuation lexicons

m-a-t-o-[+noun] s-[+pl]
o-a-d-[+noun] s-[+pl]
t-a-k-e-[+verb] s-[+3sg]

Lexical FSTs

t a k e s

Types of two level rules

All rules are of the form:

CorrespondencePair **operator** LeftContext ___ RightContext

Exclusion	$a:b \not\Leftarrow LC_RC$	<i>a</i> cannot become <i>b</i> in this context
Context restriction	$a:b \Rightarrow LC_RC$	<i>a</i> becomes <i>b</i> only in this context
Surface coercion	$a:b \Leftarrow LC_RC$	<i>a</i> must become <i>b</i> in this context
Composite	$a:b \Leftrightarrow LC_RC$	<i>a</i> becomes <i>b</i> iff in this context

Correspondences with classical rewrite rules

Exclusion	$a:b \not\Leftarrow \text{LC_RC}$	<i>a</i> cannot become <i>b</i> in this context
Context restriction	$a:b \Rightarrow \text{LC_RC}$	<i>a</i> becomes <i>b</i> only in this context
Surface coercion	$a:b \Leftarrow \text{LC_RC}$	<i>a</i> must become <i>b</i> in this context
Composite	$a:b \Leftrightarrow \text{LC_RC}$	<i>a</i> becomes <i>b</i> iff in this context

Obligatory rewrite rule

Correspondences with classical rewrite rules

Exclusion	$a:b \not\Leftarrow \text{LC_RC}$	a cannot become b in this context
Context restriction	$a:b \Rightarrow \text{LC_RC}$	a becomes b only in this context
Surface coercion	$a:b \Leftarrow \text{LC_RC}$	a must become b in this context
Composite	$a:b \Leftrightarrow \text{LC_RC}$	a becomes b iff in this context

Optional rule, and this is the only rule *in the system* that maps a to b

Correspondences with classical rewrite rules

Exclusion	$a:b \not\Leftarrow \text{LC_RC}$	a cannot become b in this context
Context restriction	$a:b \Rightarrow \text{LC_RC}$	a becomes b only in this context
Surface coercion	$a:b \Leftarrow \text{LC_RC}$	a must become b in this context
Composite	$a:b \Leftrightarrow \text{LC_RC}$	a becomes b iff in this context

Obligatory rule, and this is the only rule *in the system* that maps a to b .

Correspondences with classical rewrite rules

Exclusion	$a:b \not\Leftarrow \text{LC_RC}$	<i>a cannot become b in this context</i>
Context restriction	$a:b \Rightarrow \text{LC_RC}$	<i>a becomes b only in this context</i>
Surface coercion	$a:b \Leftarrow \text{LC_RC}$	<i>a must become b in this context</i>
Composite	$a:b \Leftrightarrow \text{LC_RC}$	<i>a becomes b iff in this context</i>

There is no rule *in the system* that maps a to b in this context.

Obvious limitations

- Well adapted for language with largely concatenative morphology, but less easy to handle non-concatenative morphology
 - As noted, intersection of regular relations is limited.
-
- But both of these issues were eventually overcome with more powerful machines and formalisms

Two-level morphology: consequences

- Morphological grammars for many languages using Koskenniemi's approach.
 - E.g., PC-KIMMO (Antworth, 1990) developed for use by field linguists
- Stimulated debate on whether deep representations were required in phonology
 - E.g., declarative “one-level phonology” (Bird & Ellison, 1994)
- Extensions include “two-level morphology with composition” (Karttunen, Kaplan & Zaenen, 1992)
- Karttunen (1998) on the “proper treatment” of optimality in phonological theory:
 - Interaction of GEN and constraints can be handled using **lenient composition**
 - OT and traditional generative phonology both combine regular relations using (lenient) composition: at the end of the day you end up with an FST
 - So what's the big theoretical difference?

FS morphology more generally: consequences

- A large number of toolkits available, among them:
 - XFST (Beesley & Karttunen, 2003)
 - Foma (Hulden, 2009)
 - Thrax (Tai, Skut & Sproat, 2011)
 - Pynini (Gorman, 2016)
- A much better understanding of how seemingly different morphological operations and theories relate to each other:
 - In Roark & Sproat (2007), we argue that the differences between, say, Item-and-Process theories and Item-and-Arrangement theories, don't amount to much of a computational distinction.

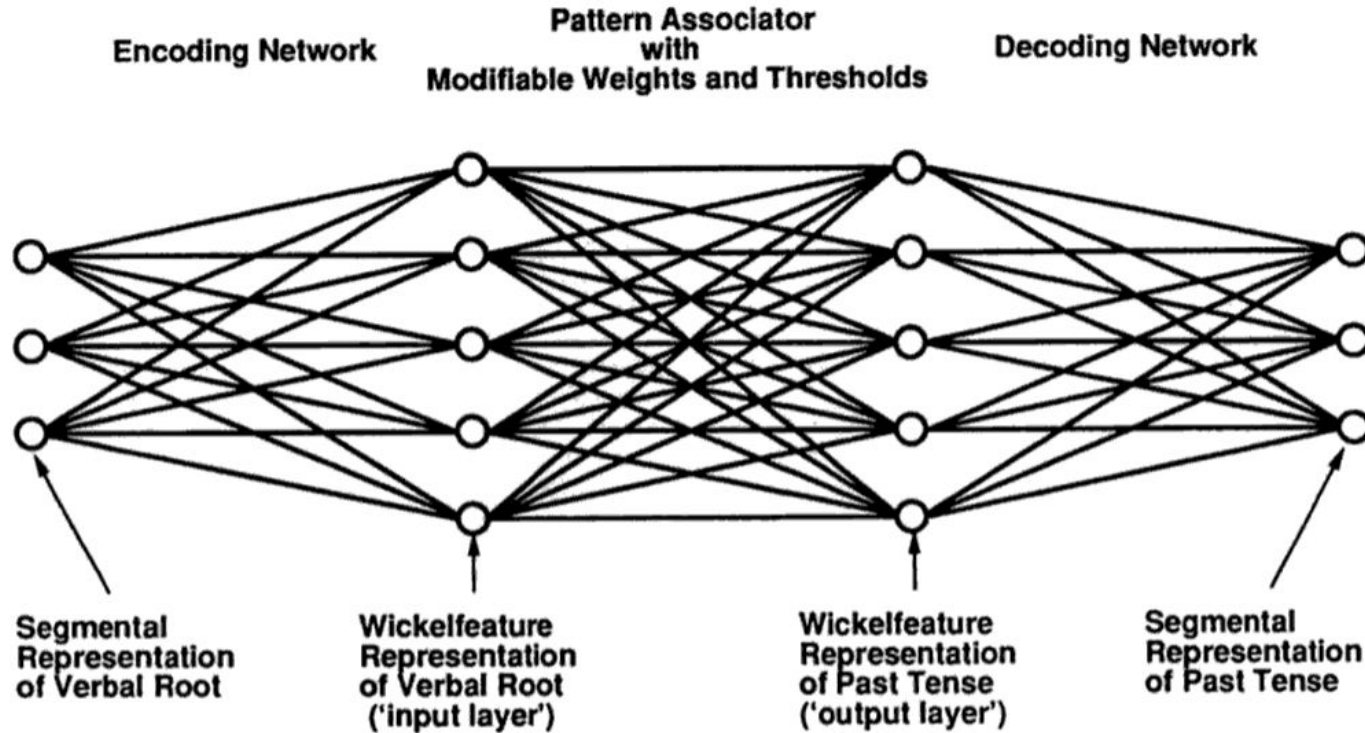
An interlude on an early neural model

- Rumelhart & McClelland's (1986) paper on learning the English past tense using a neural network:
 - Arguably one of the most controversial papers on language that was ever published
 - One of the first attempts to *learn* morphology using a computational system
 - They were specifically interested in modeling how children acquire the system
- Lots of fierce debate on both sides:
 - Searing critique by Pinker & Prince (1988)
 - But also a lot of support from the “natural phonology” camp

Basic components of R&M's system

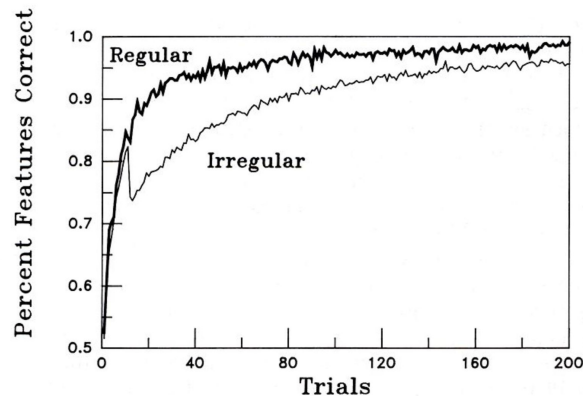
- Encoding and decoding in terms of *triphones* (which they term “Wickelphones”, after Wickelgren, 1969).
 - /kat/ → {#ka, kat, at#}
 - These are then converted into sets of binary “Wickelfeatures”
 - Decoding proved to be tricky since their network predicted weights for sets of Wickelfeatures, which then needed to be decoded into a string of phonemes.
 - They handled this by querying the output with various candidate decodings and finding the closest one
 - Noise, via feature blurring “to promote fast generalization”
 - Baayen, Chang & Blevins (2018) appear to have rediscovered the Wickelphone.
- A two-layer network — no hidden layer

Rumelhart & McClelland's network



Mimicking human learning

- The R&M model gets the famous U-shaped curve
 - But only because of a trick in the way the irregulars are presented
- The system does learn many of the past-tense forms
 - but also fails to learn others
 - **Note: there is no held-out data**



Some example outputs

guard	guard, guarded
type	typted
mail	mailed, membled
tour	toureder, toured
catch	catched
cling	clinged, clung
eat	ated

“Unlike the RM model, no adult speaker is utterly stumped in an unpressured naturalistic situation when he or she needs to produce the past tense form of *soak* or *glare*, none vacillates between *kid* and *kidded*, none produces *membled* for *mailed* or *toureder* for *toured*.” (Pinker & Prince, 1988)

Synopsis of Rumelhart & McClelland

- Pinker & Prince (1988) eviscerated the work, and argued that any system that does not distinguish between rules (for regular processes) and lexicalization (for irregular processes) is missing the point
- There was nothing critical about R&M's architecture: we (a summer intern, Dania Egedi, and I) were able to get very similar results with a simple three-layer feed-forward network that encoded segments straightforwardly as concatenated phonetic feature vectors. (Egedi & Sproat, 1988)
 - We also showed that some of their "results" were artifacts of the way they did their calculations of how much regularizing the network was doing

Synopsis of Rumelhart & McClelland

- R&M's work also generated a small cottage industry of papers trying to extend the approach, e.g.:
 - MacWhinney & Leinbach (1991)
 - Plunkett & Marchman (1991, 1993)
 - Plunkett & Juola (1999)
- See Kirov & Cotterell (2018) for a good formal characterization of the R&M work and an expanded discussion of subsequent work.
- Many of the problems with the early work have been addressed with more sophisticated networks: see below.

More finite state refinements

- Essentially all morphological operations can be reduced to a single finite-state operation: composition (Roark & Sproat, 2007)
 - Simple affixation
 - Infixation
 - Prosodic circumscription
 - Subtractive morphology
 - Root and pattern morphology
 - “Morphomic” operations
 - ...
- For example a simple suffix β can be represented by an FST
 - $S = \Sigma^*[\varepsilon : \beta]$, where
 - Suffixation to a stem Γ is then accomplished by computing $\Gamma \circ S$

More finite-state refinements

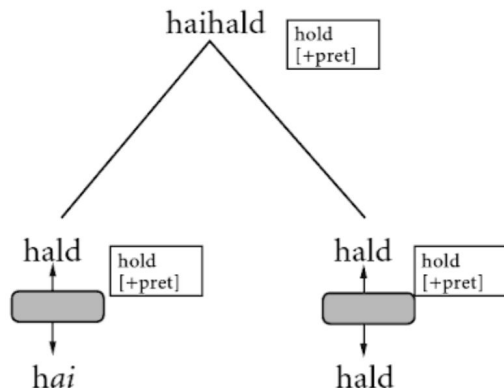
TABLE 2.2 Template- and non-template-providing affixes in Yowlumne

Root	Neutral Affixes		Template Affixes ^a	
	<i>-al</i> “dubitative”	<i>-t</i> “passive aorist”	<i>-inay</i> “gerundial” CVC(C)	<i>-ʔaa</i> “durative” CVCVV(C)
caw “shout”	caw-al	caw-t	caw-inay	cawaa-ʔaa-n
cuum “destroy”	cuum-al	cuum-t	cum-inay	cumuu-ʔaa-n
hoyoo “name”	hoyoo-al	hoyoo-t	hoy-inay	hoyoo-ʔaa-n
diiyl “guard”	diiyl-al	diiyl-t	diyl-inay	diyil-ʔaa-n
ʔilk “sing”	ʔilk-al	ʔilk-t	ʔilk-inay	ʔiliik-ʔaa-n
hiwiit “walk”	hiwiit-al	hiwiit-t	hiwt-inay	hiwiit-ʔaa-n

Suffixation of *-al*, *-t*, *-inay*, *-ʔaa* is just as before, except now Σ^* is replaced by appropriate FSTs to model the transformation.

More finite-state refinements

- But what about reduplication?
 - Since reduplication involves copying it cannot in general be handled by regular relations (though bounded reduplication can be so handled by brute force)
- Reduplication can be removed as a problem for morphological analysis by making it somebody else's problem:
 - Inkelas & Zoll's (2005) Morphological Doubling theory



Morph learning in the 2000's: Unsupervised methods

- Learning of morphological segmentation dates at least back to work of Zellig Harris (1955), who used an entropy-like measure to induce morph boundaries
- MacWhinney (1978) proposed a detailed computational model of morphological acquisition, but this was never implemented (AFAIK)
- Goldsmith (2001): *Linguistica*
- Morfessor (Creutz & Lagus, 2002) -- commonly used as a baseline
- Other work:
 - Yarowsky & Wicentowski (2001)
 - Schone & Jurafsky (2000)
 - Monson's (2008) ParaMor system

Linguistica: Goldsmith (2001)

- Collect a set of “signatures” from an unannotated corpus.
 - A couple of ways to do this, one that uses a mutual-information-like measure to find good suffixes and
 - An entropic measure to segment words with these suffixes

∅.ed.ing.s	<i>accent, afford, attempt</i>
's.∅.s	<i>adolescent, amendment, association</i>
∅.ed.er.ing.s	<i>attack, charm, flow</i>
∅.s	<i>aberration, abstractionist, accommodation</i>
e.ed.es.ing	<i>achiev, compris, describ</i>

Linguistica: Goldsmith (2001)

- Candidate evaluation

- The set of signatures and stems is a proposed morphology for the language
- The metric is *Minimum Description Length* of the corpus as compressed using the morphology

$$\sum_{w \in W} [w] [-\log(P(\sigma(w))) - \log(P(\text{stem}(w)|\sigma(w))) - \log(P(\text{suffix}(w)|\sigma(w)))]$$

Probability of the signature of the word.

Probability of the stem given the signature

- Some results for 1,000 English words:

- 829 good
- 52 wrong
- 36 failed to analyze
- 83 spurious analysis

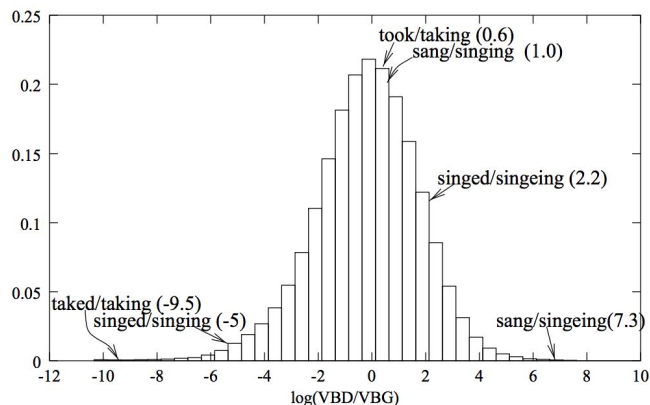
Probability of the suffix given the signature

- Goldsmith's claim: "knowledge of semantics and even grammar is unlikely to make the problem of morphology discovery significantly easier."

Yarowsky & Wicentowski (2001)

- Present a lightly supervised morphological induction system, where light supervision comprises:
 - A table of inflectional categories for the language
 - A list of candidate noun, verb and adjective roots, plus a method for guessing parts of speech
 - A list of consonants and vowels for the language
 - Optionally, a list of function words
- Unlike Linguistica, Y&W's method can learn non-affixal relationships such as *sing* ↔ *sang*
 - But how do you know it's not *sing* ↔ *singed*?
- Two methods:
 - *Cosine similarities between weighted contextual vectors* (similar to Schone & Jurafsky, 2000)
 - *Semantics does matter*
 - The frequency ratio of the two forms

Yarowsky & Wicentowski (2001)



- Iteratively train a Morphological Transformation model to yield a *morphological similarity*
- Uses the “pigeonhole principle” — better known as *morphological blocking* (Aronoff, 1976)

Combination of Similarity Models	# of Iterations	All Words (3888)	Highly Irregular (128)	Simple Concat. (1877)	Non-Concat. (1883)
FS (<i>Frequency Sim</i>)	(Iter 1)	9.8	18.6	8.8	10.1
LS (<i>Levenshtein Sim</i>)	(Iter 1)	31.3	19.6	20.0	34.4
CS (<i>Context Sim</i>)	(Iter 1)	28.0	32.8	30.0	25.8
CS+FS	(Iter 1)	32.5	64.8	32.0	30.7
CS+FS+LS	(Iter 1)	71.6	76.5	71.1	71.9
CS+FS+LS+MS	(Iter 1)	96.5	74.0	97.3	97.4
CS+FS+LS+MS	(Conv)	99.2	80.4	99.9	99.7

Table 9: Performance of combined alignment models on 4 classes of past-tense English verbs

Morfessor (Creutz & Lagus, 2002)

- Also uses Minimum Description Length (MDL) defined (for their best, recursive method) in terms of the cost of the text and the cost of the vocabulary of morph types
- Also try “sequential Maximum Likelihood”
- Outperforms Linguistica

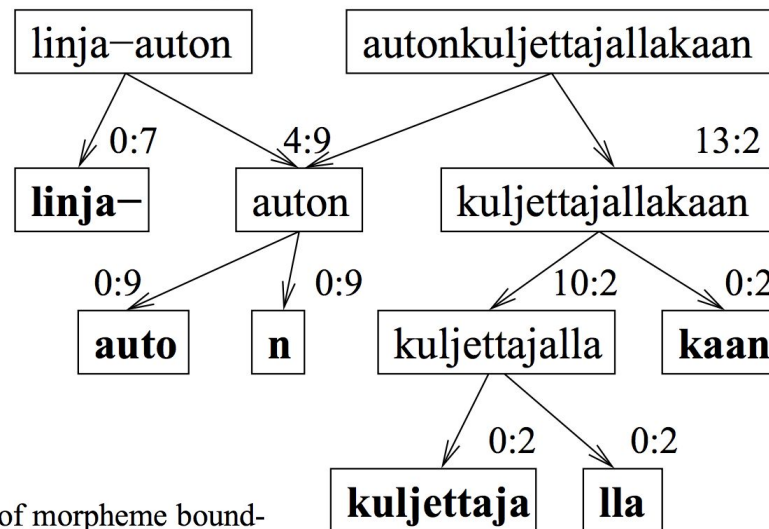


Table 3: Estimate of accuracy of morpheme boundary detection based on visual inspection of a sample of 2500 Finnish word tokens.

Method	Correct	Incomplete	Incorrect
Rec. MDL	49.6%	29.7%	20.6%
Seq. ML	47.3%	15.3%	37.4%
Linguistica	43.1%	24.1%	32.8%

Morfessor (Creutz & Lagus, 2002)

	Recursive MDL	Sequential ML	Linguistica
	affect affect + ing affect + ing + ly affect + ion affect + ion + ate affect + ion + s affect + s	affect affect + ing affect + ing + ly affecti + on affecti + on + at + e affecti + on + s affect + s	affect affect + ing affect + ing + ly affect + ion affect + ion + ate affect + ion + s affect + s
animal doctor	eläin + lääkäri	eläin + lääkäri	eläinlääkäri
	eläin + lääkäri + lle	eläin + lääkäri + lle	eläinlääkäri + lle
zool. museum	eläin + museo + n	eläin + museo + n	eläinmuseo + n
	eläin + museo + on	eläin + museo + on	eläinmuseo + on
ZOO	eläin + puisto + n	eläin + puisto + n	eläinpuisto + n
	eläin + puisto + sta	eläin + puisto + sta	eläinpuisto + sta
ZOO	eläin + tar + han	eläin + tarh + a + n	eläintarh + an

Today: Deep learning and morphology: Return to Supervised methods

- Deep learning has revolutionized natural language processing (as well as vision, *go* playing, self-driving cars ...)
 - Machine translation
 - Parsing
 - Speech recognition is moving more and more towards “end-to-end” neural systems
- Less interest in “psychological reality”
- Two problems:
 - The methods are very *data hungry*
 - It can be hard to fix things when they go wrong

Claim:

Deep learning methods are very good at *almost* learning the right function

A case where NMT misses the true function

Korean input	Transliteration	True value	Google NMT
억 (億)	eok	100,000,000 ($10^4 \times 10^4$)	billion
이억 (二億)	yi eok	200,000,000	twenty billion
삼억 (三億)	sam eok	300,000,000	three billion
사억 (四億)	sa eok	400,000,000	twelve
오억 (五億)	o eok	500,000,000	five hundred

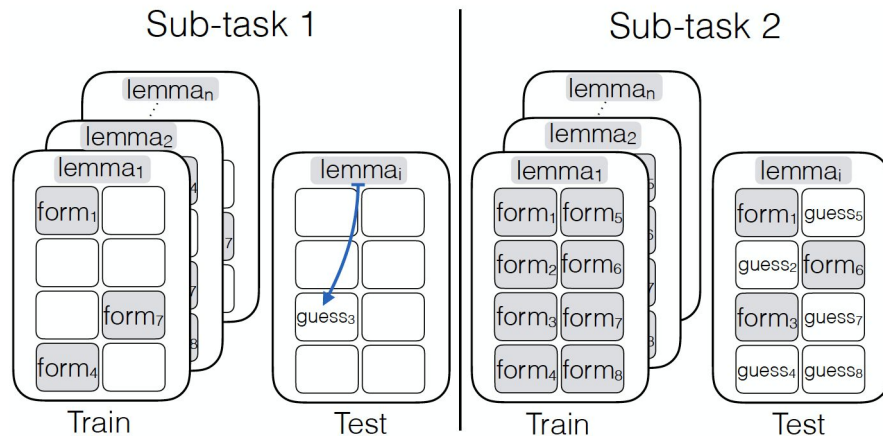
Kirov & Cotterell's redo of R&M

- Kirov & Cotterell (2018) present a reimplementaion of R&M using modern encoder-decoders
- No *eat* → *ated* errors:
 - The only errors in irregulars on *held out* data are regularizations
 - However there are some “membled” errors:
 - *thin* → *thun*
 - *try* → *traud*
 - *drawl* → *drooled*
- Displays “micro U-shaped” behavior (Plunkett & Marchman, 1991)

CLING		MISLEAD		CATCH		FLY	
#	output	#	output	#	output	#	output
5	[klɪŋd]	8	[mɪslɪ:dɪd]	7	[kætʃ]	6	[flaɪd]
11	[klʌŋ]	19	[mɪslɛd]	31	[kætʃ]	31	[flu:]
13	[klɪŋ]	21	[mɪslɛd]	43	[kɒt]	40	[flaɪd]
14	[klɪŋd]	23	[mɪslɛd]	44	[kætʃ]	42	[fleɪ]
18	[klʌŋ]	24	[mɪslɪ:dɪd]	51	[kætʃ]	47	[flaɪd]
21	[klɪŋd]	29	[mɪslɛd]	52	[kɒt]	56	[flu:]
28	[klʌŋ]	30	[mɪslɪ:dɪd]	66	[kætʃ]	62	[flaɪd]
40	[klʌŋ]	41	[mɪslɛd]	73	[kɒt]	70	[flu:]

SIGMORPHON 2017 (Cotterell et. al. 2017)

- Data from 52 languages mined from Wiktionary
 - Many errors in the data
- “Morphological reinflection” tasks:
 - Inflected form from lemma
 - Paradigm completion (or Paradigm Cell Filling; cf. Malouf 2016, inter alia)
- High (10K), medium (1K) and low (100) amounts of training data — all forms are given
 - See Silfverberg and Hulden (2018) for results when training paradigms are only partially filled.



SIGMORPHON 2017 (Cotterell et. al. 2017)

- All but one of the submitted systems was neural
- Most of the systems made use of auxiliary (real or synthetic) data to bias their systems towards *copying*.
- Without this, neural methods such as sequence-to-sequence models are not particularly good at latching onto the generalization that by default related morphological forms tend to share a lot of material
 - In other words the systems don't naturally pass the *wug* test
 - *Faithfulness* is not an inherent property of these systems

SIGMORPHON 2017 (Cotterell et. al. 2017)

- Some error analysis from the best system on Task 2, high condition:
 - The LMU (Ludwig-Maximilians-Universität München) system (Kann & Schütze, 2017), which uses an encoder-decoder network that is an ensemble of 5 RNN encoder-decoders (Kann & Schütze, 2016)
 - “our main focus is on task 2” (Kann & Schütze, 2017, p. 40)
 - Here we focus on LMU-2, which was usually the better of the two results that team submitted
 - **There is also an ongoing analysis of the errors by Kyle Gorman and colleagues**

Error analysis for Task 2: French (0.988 acc)

- Very few errors at all, but some odd ones:
 - *prêter* → **prêasas** (V;IND;PST;2;SG;PFV — correct *prêtas*)
 - 9 slots (out of 49) are given
 - Generalization from training example *écraser* → *écrasas*?
 - *neiger* → **nengé** (V.PTCP;PST — correct *neigé*)
 - 1 form (out of 11) is given
 - Possible confusion with *venger* → *vengé* from training?

Error analysis for Task 2: Irish (0.695 acc)

Lemma	Features	Desired form	Produced Form	Form(s) given
Rómhánach	VOC;SG	Rómhánaigh	Rhómhánaigh	Rómhánaigh (GEN;SG) Rómhánach (DAT;SG)
núicléas	VOC;SG	núicléis	nhúicléis	núicléis (NOM;PL) núicléas (NOM;SG)
stáisiún	VOC;SG	stáisiúin	shtáisiúin	stáisiúin (DAT; PL)

A prominent feature is overgeneralization of lenition (“aspiration”) represented in Irish orthography with <h>: *turas* (journey) → *a thurais*. This never happens with /r/, /l/, /n/ or with /s/ before a stop. In the training data there are:

- 12 instances of /r/ with no aspiration in the vocative
- 1 instance of /n/
- 10 instances of /sC/

Error analysis for Task 2: Latin (0.877 acc)

Lemma	Features	Desired form	Produced Form	Form(s) given
exemplar	ABL;PL	exemplāribus	exemplāriīs	exemplāria (VOC;PL)
consuetudo	NOM;SG	cōnsuētūdō	cōnsuētūden	cōnsuētūdine (ABL;SG)
inventor	NOM;PL	inventōrēs	inventā ¹	inventōrum (GEN; PL)
sinapis	ACC;SG	sināpem	sināpemrūpim ²	sināpis (GEN; SG)
monasterium	NOM;SG	monastēriō (!)	monastērgular ³	monastērium (GEN;SG) (!)

¹ Possible contamination from *elementum*, *elementa* in training?

² Contamination from one bad paradigm *rupes* → *rūpēs rūpīs* (ACC;PL) and *rupes* → *rūpem rūpim* (ACC;SG)

³ Four training examples where the NOM;SG is given as “Singular”

A synopsis

- Are there any “membled” errors?
 - Unclear, but there are some odd generalizations from minimal training examples
- Another way to think of it:
 - the model mostly does something reasonable and often the errors make sense
 - ...but occasionally it does something really odd
 - very sensitive to noise: a realistic acquisition algorithm *must* be robust to noise
- In their data preparation K&S reverse the strings in languages that are primarily prefixing (e.g. Navajo):
 - They seem to expect their model to focus on the end of whatever it is presented with
 - Cf. the poor generalization of the *initial* consonant mutations in Irish
- Is the problem of morphological induction solved?
 - Not yet: the random odd generalization suggests that current systems are not quite learning the function

Computational models and morphological theory

- Much recent work on morphological learning is, at least implicitly, more sympathetic to Word-and-Paradigm models: do we really need morphemes?
- Kirov & Cotterell (2018) note that R&M's system could easily learn unnatural alternations such as string-reversal, but that modern Encoder-Decoder models are probably less likely to be able to learn some mappings.
 - What mappings *can* they learn and how well do those correspond to attested patterns?
 - How many of the limitations on observed morphological operations are due to history?
- How well are neural models able to learn abstractions such as “morphomic” processes (Aronoff, 1993)?
 - Whatever your position on the “morphome debate”, it remains true that one needs some notion like: the following affixes x, y, z, \dots attach to stem X , where X may not be predictable in form

A final thought: Two kinds of “black box”...

- Machine learning has often been treated as a “black box”:
 - one applies some methods more or less off-the-shelf to some problem, without necessarily understanding why those methods might or might not be appropriate

A final thought: Two kinds of “black box”...

- With the advent of powerful deep learning methods and increasingly sophisticated toolkits we are seeing the inverse kind of “black box”:
 - machine learning experts who are hunting for nails to apply their hammers to
- Example: text-to-speech is just a sequence-to-sequence problem:
 - Text is the input sequence
 - Speech is the output sequence (of continuous vectors)
 - Hence systems like *Tacotron* (Wang et al, 2017) or *Char2Wav* (Sotelo et al. 2017)
 - Cf. a comment on <https://news.ycombinator.com/item?id=13992454> about the original *Tacotron* paper:

These new models are replacing almost everything that's been developed for TTS in the last 30 years with neural nets. So you don't need to study TTS anymore. Study neural nets and then read this paper and you will understand the state of the art in TTS.

...or, do we need (computational) linguists anymore?

- The real danger here is that when knowledge reduces to knowing everything there is to know about the latest neural approaches, there'll be nobody left who understands, if there is a problem, what went wrong.
- The best defense will be to get as many people who *do* understand the problem involved in the process.

References

- Jonathan Allen, M. Sharon Hunnicutt and Dennis Klatt. 1987. *From Text to Speech: the MITalk System*. Cambridge University Press, Cambridge.
- Evan Antworth. 1990. *PC-KIMMO: A Two-Level Processor for Morphological Analysis*. SIL Publications, Dallas.
- Mark Aronoff. 1976. *Word Formation in Generative Grammar*. MIT Press, Cambridge, MA.
- Mark Aronoff. 1993. *Morphology by itself: Stems and Inflectional Classes*. MIT Press, Cambridge, MA.
- R. Harald Baayen, Yu-Ying Chang and James Blevins. 2018. “Inflectional morphology with linear mappings.” *The Mental Lexicon* 13(2), 230-268.
- Kenneth Beesley and Lauri Karttunen, 2003. *Finite State Morphology*. CSLI Publications, Stanford.
- Steven Bird and T. Mark Ellison. 1994. “One-level phonology: Autosegmental Representations and Rules as Finite Automata.” *Computational Linguistics* 20(1), 55-90.
- Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqi, Sandra Kübler, David Yarowsky and Jason Eisner. 2017. “CoNLL-SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection in 52 Languages”. *SIGMORPHON*.
- Mathias Creutz and Krista Lagus. 2002. “Unsupervised discovery of morphemes.” *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, 21-30
- Dania Egedi and Richard Sproat. 1988. “Connectionist networks and natural language morphology.” Presented at the University of Maryland Conference on Grammar and Language Processing.

- John Goldsmith. 2001. “Unsupervised acquisition of the morphology of a natural language.” *Computational Linguistics* 27(2), 153-198.
- Kyle Gorman. 2016. “Pynini: A Python library for weighted finite-state grammar compilation.” *Proceedings of the ACL Workshop on Statistical NLP and Weighted Automata*, 75–80.
- Harald Hammarström. 2009. *Unsupervised Learning of Morphology and the Languages of the World*. PhD Dissertation. Chalmers University, Gothenburg, Sweden.
- Zellig Harris. 1955. “From Phoneme to Morpheme.” *Language* 31(2), 190-222.
- Mans Hulden. 2009. “Foma: a finite-state compiler and library.” EACL, 29-32.
- Sharon Inkelas and C. Zoll. *Reduplication: Doubling in Morphology*. Cambridge University Press, Cambridge, UK.
- C. Douglas Johnson. 1972. *Formal Aspects of Phonological Description*. Mouton, The Hague.
- Katharina Kann and Hinrich Schütze. 2016. “Single-Model Encoder-Decoder with Explicit Morphological Representation for Reinflection.” *ACL*, 555-560
- Katharina Kann and Hinrich Schütze. 2017. “The LMU System for the CoNLL-SIGMORPHON 2017 Shared Task on Universal Morphological Reinflection.” *SIGMORPHON*.
- Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3), 331–378.
- Lauri Karttunen, Ronald M. Kaplan, and Annie Zaenen. 1992. “Two-level morphology with composition.” *COLING*, 23-28.

- Lauri Karttunen. 1998. “The proper treatment of optimality in computational phonology.” Plenary talk at *FSMNLP*, Ankara, Turkey, 1-12.
- Christo Kirov and Ryan Cotterell. 2018. “Recurrent Neural Networks in Linguistic Theory: Revisiting Pinker and Prince (1988) and the Past Tense Debate.” *Transactions of the Association for Computational Linguistics*, 6, 651-665.
- Dennis Klatt. 1987. “Review of text-to-speech conversion for English.” *Journal of the Acoustical Society of America* 82, 737-793.
- Kimmo Koskenniemi. 1983. *Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production*. PhD Thesis, University of Helsinki.
- Henry Kučera and W. Nelson Francis. 1967. *Computational Analysis of Present-Day American English*. Brown University Press, Providence, RI.
- Brian MacWhinney, 1978. *The Acquisition of Morphophonology*. Monographs of the Society for Research in Child development. University of Chicago Press, Chicago.
- Brian MacWhinney and Jared Leinbach. 1991. “Implementations are not conceptualizations: Revising the verb learning model.” *Cognition*, 40(1), 121–157.
- Robert Malouf. 2016. “Generating morphological paradigms with a recurrent neural net.” *San Diego Linguistic Papers*, 6, 122-129.
- Christian Monson. 2008. *ParaMor: From Paradigm Structure to Natural Language Morphology Induction*. PhD Thesis, CMU.

- Steven Pinker and Alan Prince. 1988. “On language and connectionism: Analysis of a parallel distributed processing model of language acquisition.” *Cognition* 28(1), 73–193.
- Kim Plunkett and Virginia Marchman. 1991. “U-shaped learning and frequency effects in a multilayered perceptron: Implications for child language acquisition.” *Cognition*, 38(1), 43–102.
- Kim Plunkett and Virginia Marchman. 1993. “From rote learning to system building: Acquiring verb morphology in children and connectionist nets.” *Cognition*, 48(1), 21–69.
- Kim Plunkett and Patrick Juola. 1999. “A connectionist model of English past tense and plural morphology.” *Cognitive Science*, 23(4), 463–490.
- Martin Porter. 1980. An algorithm for suffix stripping. *Program* 14(3) 130–137.
- Brian Roark and Richard Sproat. 2007. *Computational Approaches to Morphology and Syntax*. Oxford University Press, Oxford.
- David E. Rumelhart and James L. McClelland. 1986. “On learning the past tenses of English verbs.” In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2, 216–271. MIT Press, Cambridge, MA.
- Patrick Schone and Daniel Jurafsky. 2000. “Knowledge-free induction of morphology using latent semantic analysis.” *CoNLL*, 67-72.
- Miikka Silfverberg and Mans Hulden. 2018. “An encoder-decoder approach to the paradigm cell filling problem.” *EMNLP*, 2883-2889.
- Terry Tai, Wojciech Skut and Richard Sproat. “Thrax: An Open Source Grammar Compiler Built on OpenFs.” *ASRU 2011*, Waikoloa Resort, Hawaii.

- Sotelo, Jose, Soroush Mehri, Kundan Kumar, João Felipe Santos, Kyle Kastner, Aaron Courville, and Yoshua Bengio. 2017. “Char2wav: End-to-end speech synthesis.” In *ICLR*.
- David Yarowsky and Richard Wicentowski. 2001. “Minimally supervised morphological analysis by multimodal alignment.” *ACL*, 207-216.
- Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, Rif A. Saurous. 2017. “Tacotron: Towards End-to-End Speech Synthesis.” <https://arxiv.org/abs/1703.10135>.
- Wayne Wickelgren. 1969. “Auditory or articulatory coding in verbal short-term memory.” *Psychological Review*, 76(2), 232-235.