Analyzing Ensemble and Backoff Lemmatization of Latin Text

Janaki Mohta
Institute for Computing in Research

July 28, 2025

Background: Linguistics

Scientific Study of Language

Rules for analyzing language — Computing
Natural Language Processing (NLP)

Background: Linguistics

Important first step: Normalization

Stemming

Walking
Walk | ing
Walk

Thing
Th | ing
Th

Background: Linguistics

Important first step: Normalization

Stemming

Walking
Walk | ing
Walk

Thing
Th | ing
Th

Lemmatization

"conventionally defined 'base' form"
(Sprugnoli et al., 2020)

Walking
"Walk" -ing
Walk

Thing
"Thing"
Thing

Linguistics in Latin

Strengths

- Standardization
- Precise word forms

Weaknesses

- Many endings
- Repeated endings

templum; stem templ-
interdum; stem interd-

Lemmatization stronger choice

CLTK Latin Lemmatization

The Lemmatizers

- Dict - Dictionary
- Unigram - Training
- Regexp - "Stemmer"
- Old Dict - Larger Dictionary
- Identity - Failsafe

  templi → templi

  interdum → interdum

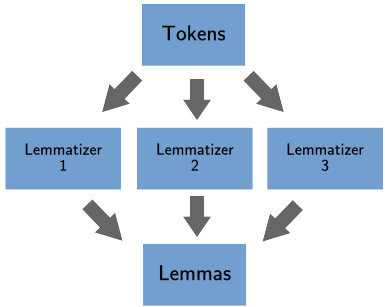
Backoff Lemmatization Pipeline

CLTK Latin Lemmatization

Ensemble: Promising Second Option?

Ensemble lemmatization with the Classical Language Toolkit (Burns, 2020)

Strong lemmatizers strengthening each other



CLTK Ensemble Lemmatization

Description in paper vs implementation in library

Lemmatizers Offered

- Dict
- Unigram
- Regexp
- Old Dict

Potential Benefits

- Nuance
- Customization
- Expandability

Why hasn't Ensemble overtaken Backoff?

Preparing for Tests

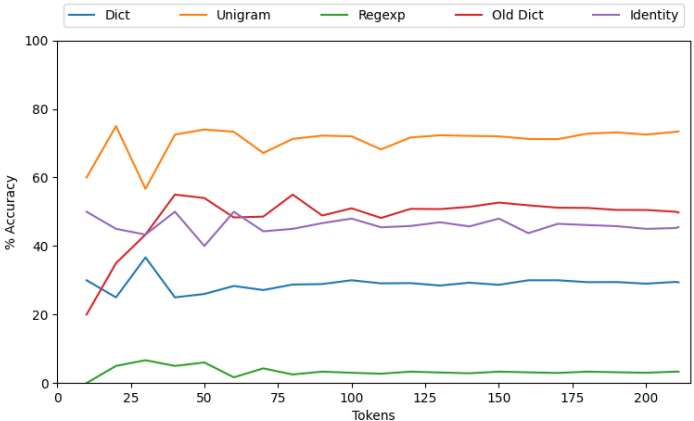
Isolating sub-lemmatizers

Getting a dataset

- Virgil's *Aeneid*
- Cicero's Catilinarian Orations
- **Caesar's *Commentarii de Bello Gallico***

Manual lemmatization: Time constraints

Testing Dataset Size

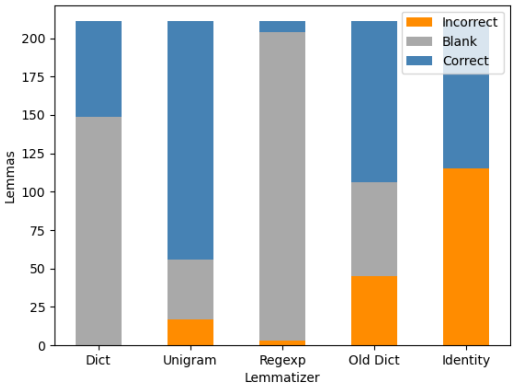


Random slicing

Stabilization: sign of consistency

Caveat: Repeated tokens

Individual Sub-Lemmatizers



Sub-Lemmatizer	Accuracy Ratio
Dict	inf.
Unigram	9.12
Regexp	2.33
Old Dict	2.33
Identity	0.835

Correct - Incorrect Ratio (Accuracy Ratio)

Why hasn't Ensemble overtaken Backoff?

Can the Backoff order itself be optimized?

Custom Backoff Pipelines

Pipeline	Accuracy
CLTK Backoff	0.853
CLTK Backoff without Dict	0.839
Min Error (Dict, Regexp, Unigram, Old Dict, Identity)	0.834
Max Correct (Dict, Unigram, Identity)	0.834

No pipelines as accurate, many pipelines close

Worse configuration of the same components

Can the Backoff order itself be optimized?

- General conclusion: No
- Sub-lemmatizers optimized for order

Custom Ensemble Pipelines

	Components	Accuracy
	CLTK Backoff	0.853
A	**Old Dict, Dict, Unigram, Regexp**	**0.796**
B	**Old Dict, Unigram, Regexp**	**0.796**
C	Old Dict, Dict, Unigram	0.777
D	Old Dict, Unigram	0.777
E	Dict, Unigram, Regexp	0.763

Similar to Backoff: Consistently slightly worse

Optimization for task

Dict underuse

Unigram: "est"
$\rightarrow$ "sum": 0.993, "edo": 0.007

A Common Theme

Why hasn't Ensemble overtaken Backoff?

Can the Backoff order itself be optimized?

The key: Optimization

Backoff pipeline specialized

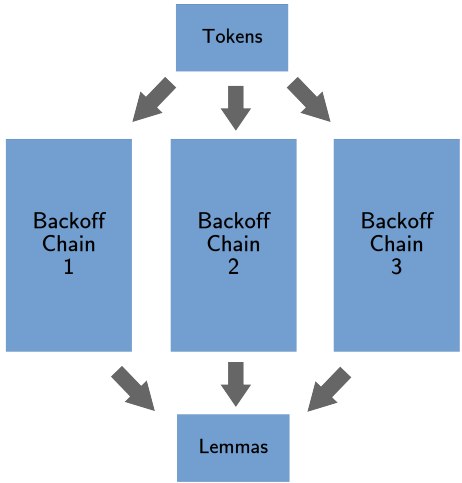
Ensemble in this form: Doomed to fail?

- Specialization
- Number of lemmatizers
- Complex weighting — worse Backoff

Stronger Ensemble Applications

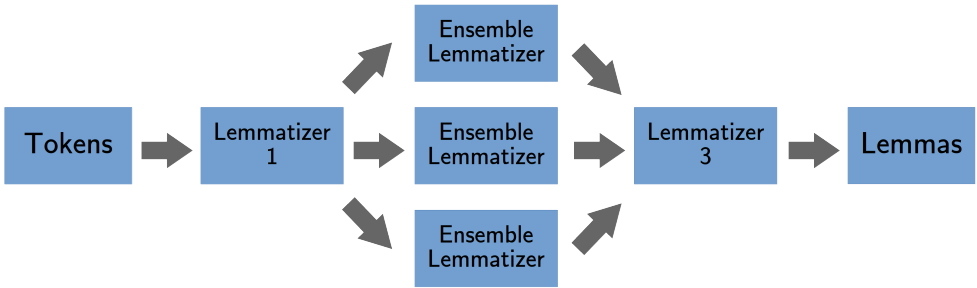
Multiple lemmatization pipelines

Burns's original proposal



Stronger Ensemble Applications

Strengthening segment of Backoff chain
Strong choice: Unigram



Next Steps

Testing Ensemble in stronger forms

Application: strengthening complex lemmatizers

- Stronger lemmatizers developed
- Easy increase in performance?
- Potentially diminishing returns

Thank You