Learning Morphological Rules for Amharic Verbs Using Inductive Logic Programming

Wondwossen Mulugeta
Addis Ababa University,
Addis Ababa, Ethiopia
wondisho@yahoo.com

Michael Gasser
Indiana University,
Bloomington, USA
gasser@cs.indiana.edu

LREC Conference
SALTMIL-AfLaT Workshop
Normalization of Under-Resources Languages
Istanbul, Turkey
May 22, 2012

Agenda

- Introduction
- Amharic Verb Morphology
- Machine Learning (ILP) of Morphology
- Experiment Setup and Data
- Experiments and Result
- Challenges
- Future work

Introduction

- Amharic is a Semitic language (27+ speakers)
  - Written using Amharic Fidel, ፊደል , which is syllabic script
    - (be=በ, bu=ቡ, bi=ቢ, ba=ባ, bE=ቤ, b=ብ, bo=ቦ)
- Many Computational Approaches of Morphology:
  - Rule based and Machine learning for many languages
    - HornMorpho: Finite State Transducer (Amharic, Tigrigna and Oromo)
  - Amharic morphology has so far been attempted using only rule-based methods.
- We have applied machine learning approach to the task
  - *This is a work on progress*
- The work is a contribution to learning of Morphology in general

Amharic Verbs

- Amharic Verbs convey:
  - lexical information, subject and object person, number, and gender; tense, aspect, and mood; various derivational categories such as passive, causative, and reciprocal; polarity (affirmative/negative); relativization; and a range of prepositions and conjunctions.
- Amharic Verb Morphology:
  - affixation, reduplication, and compounding (common to most)
  - The stems consist of a root + vowels + template merger
    - (e.g., ***sbr*** + **ee** + CVCVC, which leads to the stem ***seber*** 'broke')
- It is a non-concatenative process

Amharic Verbs…contd

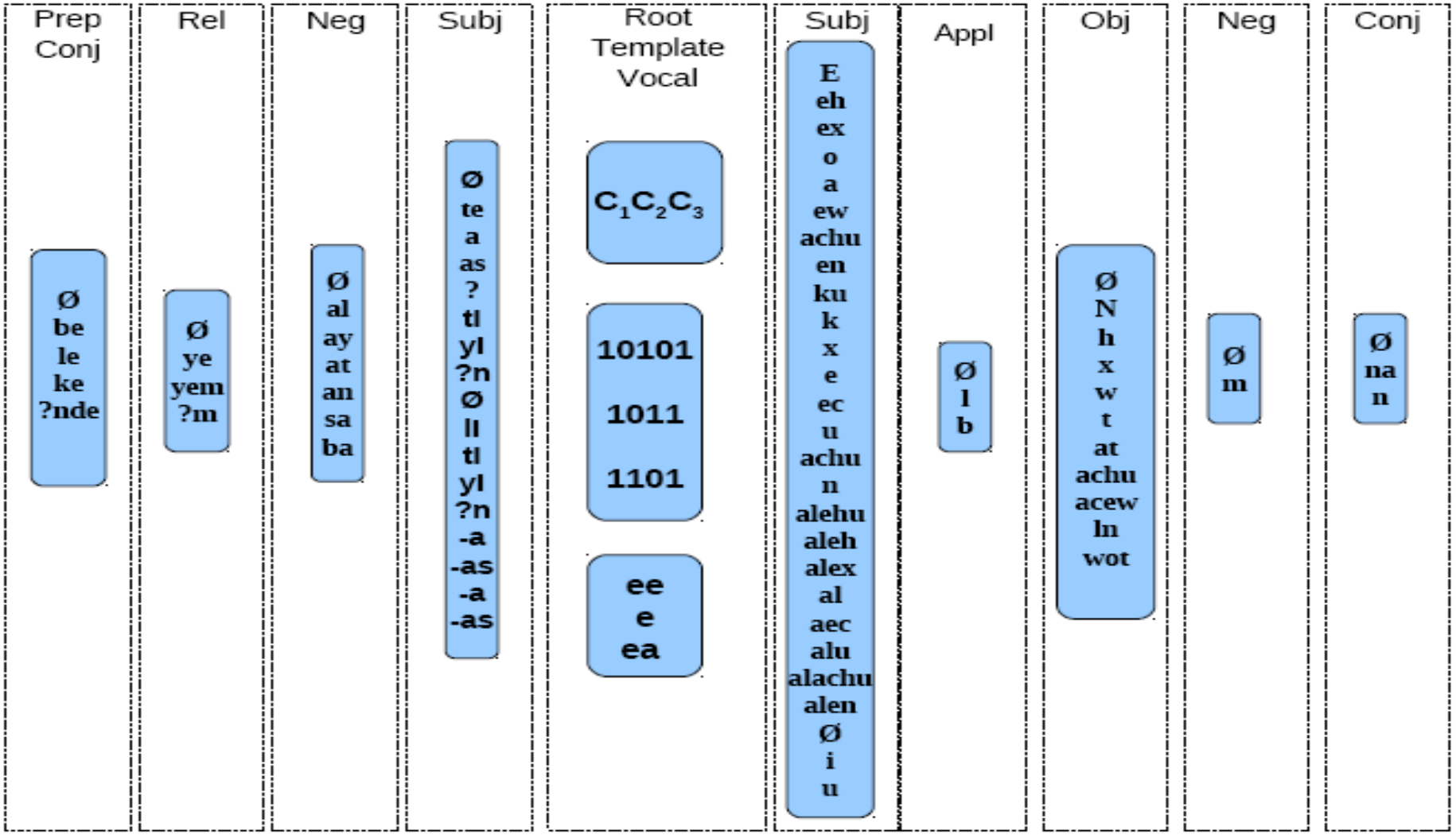
- Amharic verbs: *4 prefixes* and *5 suffixes*.
- The affixes have an intricate set of co-occurrence rule
- Grammatical features are shown using:
  - Affixes, Vowel sequence, Root template (CV pattern)

*?-**sebr**-alehu (እሰብራለሁ) → 1ˢ pers. sing. **simplex** imperfective*
*?-**seber**-alehu (እሰበራለሁ) → 1st pers. sing. **passive** imperfective*

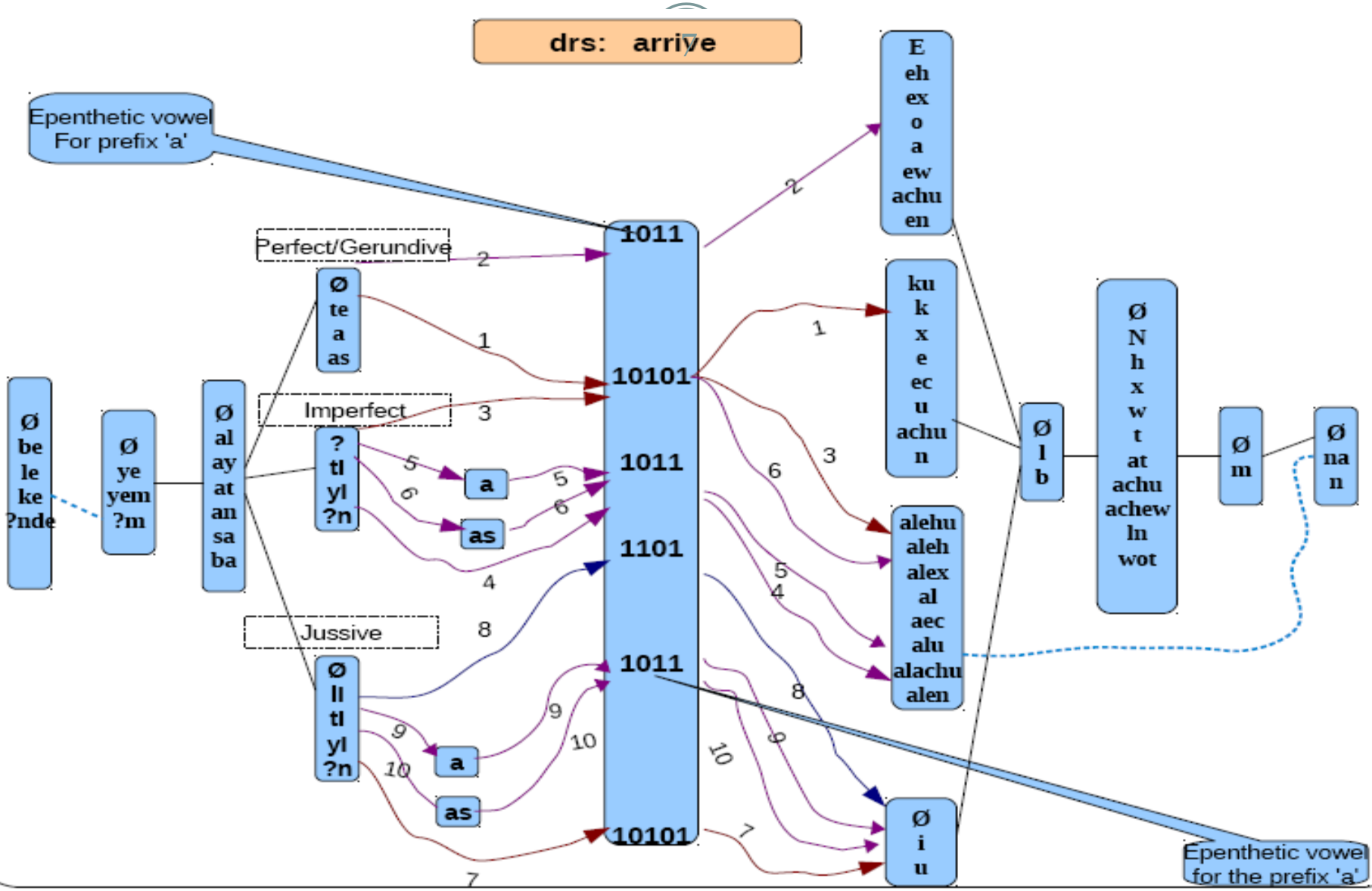
*te-**deres**-ku (ተደረስኩ) → 1st pers. sing. **passive** perfective*
***deres**-ku (ደረስኩ) → 1st pers. sing. **simplex** perfective*

The **Geez** script has been Romanized using the standard SERA for this experiment using our own Prolog script (lookup dictionary).

Amharic Verbs…contd



Amharic Verbs...contd



Amharic Verbs…contd

- Amharic morphology has alternation rules:
  - stem affix intersection points or within the stem itself

Word	Root	Feature
gdel (ግደል)	gdl(ግድል)	2nd person sing. masc. imperative
gdey (gdel-i) (ግደይ)	gdl(ግድል)	2nd person sing. fem. imperative
t-gedl-aleh (ትገድላለህ)	gdl(ግድል)	2nd person sing. masc. imperfect
t-gedy-alex (ትገድያለሽ)	gdl(ግድል)	2nd person sing. fem. imperfect

Another Example

ብ-አል-ሰብር-ል-አችሁ-ም

b-al-sebr-l-achu-m

even though I won't be break it for you(pl)

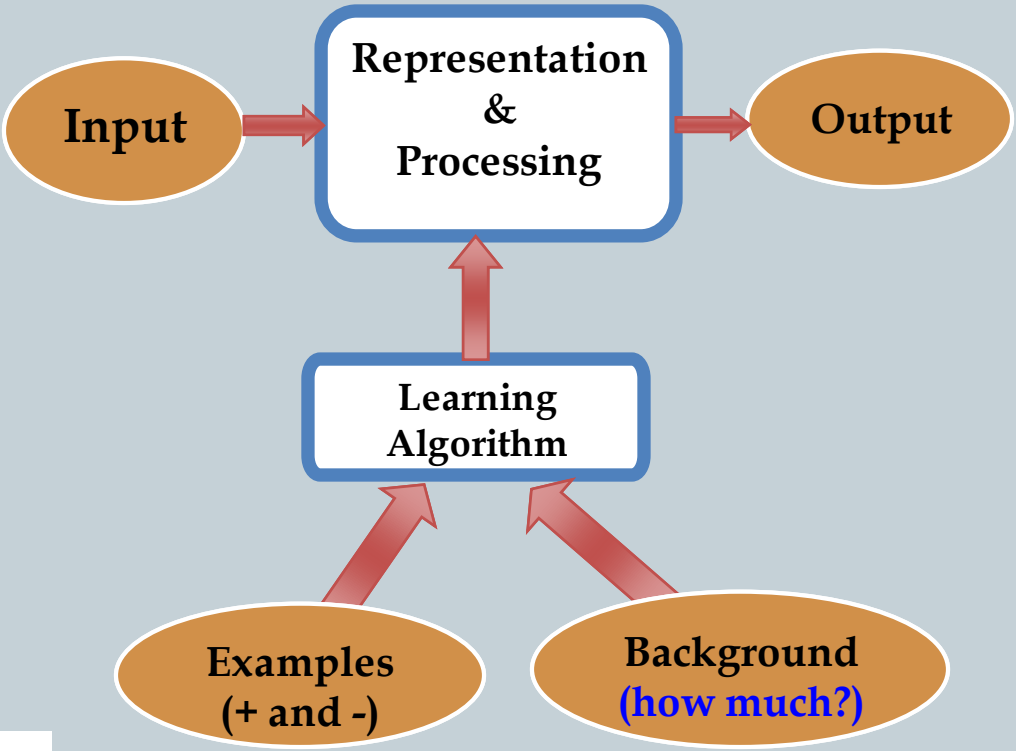
ባልሰብርላችሁም

Rule Based Morphology

- Finite State for morphology dominant after Koskenniemi's two level morphology .
  - Rule based
  - HornMorpho developed to analyze Amharic, Tigrigna and Oromiffa words
  - All rules need to de enumerated
  - knowledge-based: (HornMorpho experience)
    - difficult to debug,
    - hard to modify (to add new findings),
    - Difficult to adapt to other similar languages

ILP Framework

- ILP combines Logic and Programming
- Hypothesis is drawn from background knowledge and examples.
  - The examples (E), background knowledge (B) and hypothesis (H) are all logic programs.



Prolog Rules:

$$\left.\begin{array}{l} p :- q, r. \\ p :- s, t. \end{array}\right\} \quad p \Leftrightarrow (q \wedge r) \vee (s \wedge t)$$

ILP for Morphology

- ILP learning systems is Supervised
  - Supervised morphology learning systems are usually based on two-level morphology
  - These approaches differ in the level of supervision they employ to capture the rules.
    - word pairs
    - segmentation of input words
    - a stem or root and a set of grammatical features

Machine Learning (ILP) of Morphology

- Attempts to apply ILP to morphology
  - Kazakov, 2000, Manandhar et al, 1998, Zdravkova et al, 2005,
    - English, Macedonian
  - dealt with languages with relatively simple morphology
  - No root template issue (Vital for Amharic and similar languages)
  - Was possible to list all (most) examples?

- We have used CLOG ILP tool for our experiment
  - CLOG is a Prolog based ILP system,
  - Developed by Manandhar *et al* (1998),
  - Learn first order decision lists (rules)
  - Use only positive examples.
    - CLOG relies on output completeness

Experiment Setup and Data

- Learning Amharic morphological rules with ILP:
  - Training data prepared with the help of *HornMorpho*
  - *Used only **216** Amharic verbs*
  - background knowledge and the learning aspect.
    1) To handle stem extraction by identifying affixes,
    2) To identify root and vowel sequence
    3) To handle orthographic alternations
    4) To associate grammatical feature with word constituents

Training Data Format:

stem(Word, Stem, Root, Grammatical Features)

Training Example:

Stem([s,e,b,e,r,k,u],[s,e,b,e,r],[s,b,r] [1,1]).
stem([s,e,b,e,r,k],[s,e,b,e,r],[s,b,r], [1,2]).
stem([s,e,b,e,r,x],[s,e,b,e,r],[s,b,r], [1,3]).

Experiment Setup and Data

- Learning stem extraction:
  - ***set_affix*** predicate
    - Identify the prefix and suffixes of the input word.
    - Takes the *Word* and *Stem* and learns the affixes

```
set_affix(Word, Stem, P1,P2,S1,S2):-
    split(Word, P1, W11),
    split(Stem, P2, W22),
    split(W11, X, S1),
    split(W22, X, S2),
    not( (P1=[],P2=[],S1=[],S2=[])).
```

    - The utility predicate '*split*' segments any input string into all possible pairs of substrings.
    - ***sebr*** {([]-[*sebr*]), ([*s*]-[*ebr*]), ([*se*]-[*br*]), ([*seb*]-[*r*]), or ([*sebr*]-[])}.

Experiment Setup and Data

- Affix extraction example:

teseberku (seber) Vs tegedelku (gedel)

{break} {kill}

[],[teseberku],[]
[t],[eseberku],[]
[te],[seberku],[]
:
[te],[seber],[ku]
:
[te],[seber],[ku]
[te],[seber],[ku]
[],[teseber],[ku]
[],[teseberk],[k]

[te],[seber],[ku]
[te],[gedel],[ku]

Segment: [te], STEM, [ku]
CV Pattern: CVCVC, ee

More general rule that can be derived for the two words

[],[tegedelku],[]
[t],[egedelku],[]
[te],[gedelku],[]
:
[te],[gedel],[ku]
:
[te],[gedel],[ku]
[te],[gedel],[ku]
[],[tegedel],[ku]
[],[tegedelk],[u]

Experiment Setup and Data

- Learning Roots:
- ***root_vocal***, Predicate
  - Used to extract the root from examples by taking only the *Stem* and the *Root* (the second and third arguments)
  - The performs unconstrained permutation of the characters in the *Stem* until the first segment of the permutated string matches the *Root* character pattern provided in the example.

```
root_vocal(Stem,Root,Vowel):-
        merge(Stem,Root,Vowel).

merge([X,Y,Z|T],[X,Y|R],[Z|V]):-
        merge(T,R,V).
merge([X,Y|T],R,[X,Y|V]):-
        merge(T,R,V).
merge([X|Y],[X|Z],W) :-
        merge(Y,Z,W).
merge([X|Y],Z,[X|W]) :-
        merge(Y,Z,W).
```

Experiment Setup and Data

- Root Vocal and Template extraction”

seber (sbr) Vs gedl (gdl)

seber
sebre
:
:
sbree
:
:
seebr
ebres
esber
eesbr

Vowel Sequence: ***ee***
CV Pattern: ***CVCVC***

gdel
gdle
:
:
gdle
:
:
gedl
edlg
egdl
egld

Vowel Sequence: ***e***
CV Pattern: ***CVCC***

Experiment Setup and Data

- Learning stem internal alternations:
  - ***set_internal_alter*** predicate
    - This predicate works much like the '*set_affix*' predicate except that it replaces a substring which is found in the middle of *Stem* by another substring from *Valid_Stem*.
    - Required a different set of training data:

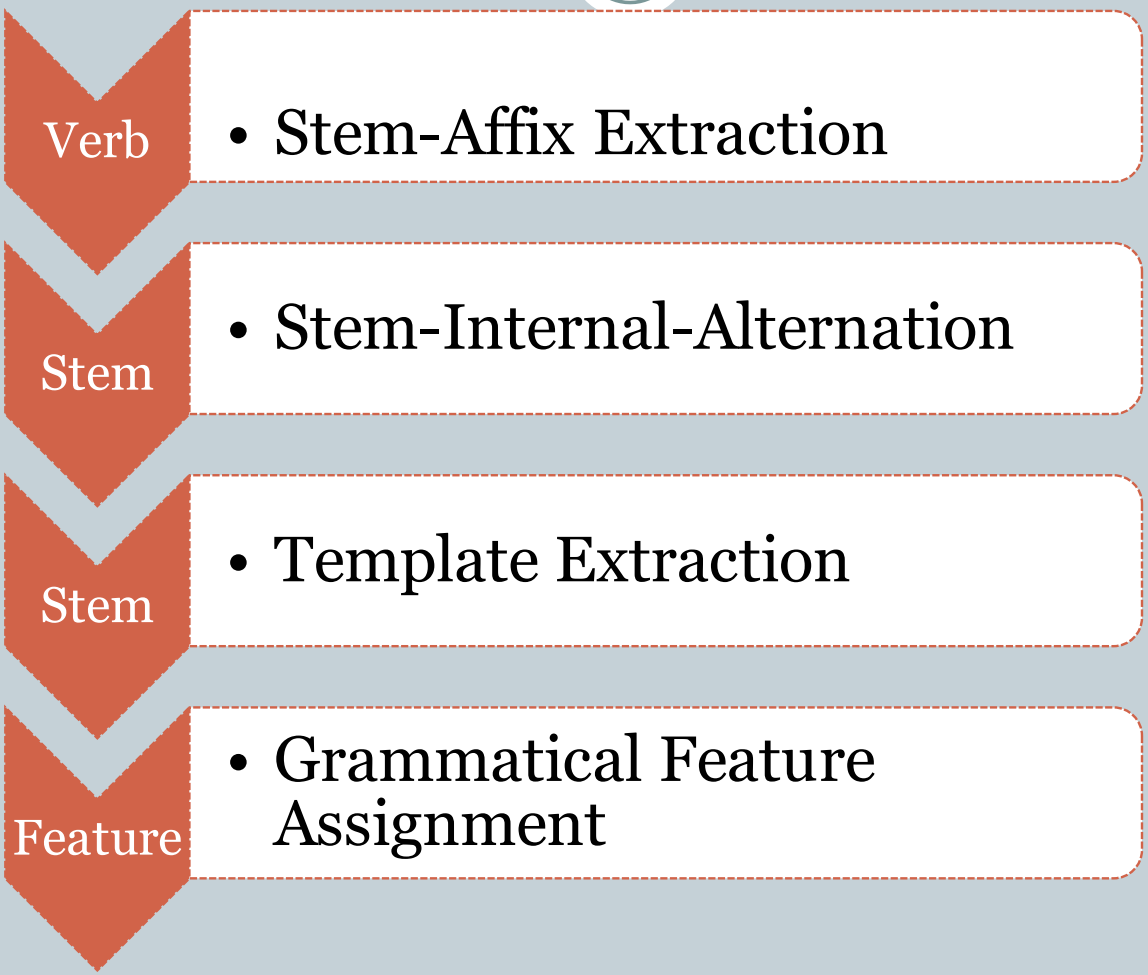
```
alter([h,e,d],[h,y,e,d]).
alter([m,o,t],[m,e,w,o,t]).
alter([s,a,m],[s,e,?,a,m]).
```

```
set_internal_alter(Stem,Valid_Stem,St1,St2):-
        split(Stem,P1,X1),
        split(Valid_Stem,P1,X2),
        split(X1,St1,Y1),
        split(X2,St2,Y1).
```

Experiments and Result

- To learn a set of rules, the predicate and arity for the rules must be provided for CLOG.
  - predicate schemas
    - ***rule(stem(_,_,_,_))*** for *set_affix* and *root_vocal*, and
    - ***rule(alter(_,_))*** for *set_internal_alter*.
- The training set contains 216 Amharic verbs.
- The example contains all possible combinations of tense and subject features.
- Each word is first Romanized, then segmented into the stem and grammatical features

Experiments and Result



Experiments and Result

- The training took less than a minute for Affix extraction
- 108 rules for affix extraction, one example

```
stem(Word, Stem, [2, 7]):-
        set_affix(Word, Stem, [y], [], [u], []),
        feature([2, 7], [imperfective, tppn]),
        template(Stem, [1, 0, 1, 1]).
```

Input Word: [y,m,e,k,r,u] {advise}
Set_affix *results in: Stem=[m,e,k,r] as to the above rule (removing[y] and [u])*
Template *will generate: 1,0,1,1 from Stem which is the same as in the rule*
Thus, ***Feature*** *will declare the word is Imperfective, Third Person Plural Neuter*

*Input Word: *[y,m,e,k,e,r,u] {advise} (valid but no such examples in the training)*
Set_affix *will result in: Stem=[m,e,k,e,r] according to the above rule*
Template *will generate: 1,0,1,0,1 from Stem which* ***will fail*** *the rule*

Experiments and Result

- 18 rules for root template extraction: one example

```
root(Stem, Root):-
        root_vocal(Stem, Root, [e, e]),
        template(Stem, [1, 0, 1, 0, 1]) .
```

Input Stem: [g,e,r,e,f] {beat}
root_vocal *results in:*
Root=[m,k,r] based on the number and type of vowels in the rule
Template *will generate: 1,0,1,0,1 from Stem which is the same as in the rule*

*Input Stem: *[g,a,r,e,f]*
root_vocal *results in: Root=[g,r,f] [a,e], which* ***fails*** *to meet the rule*

Experiments and Result

- 3 rules for internal stem alternation

```
alter(Stem,Valid_Stem):-
   set_internal_alter(Stem,Valid_Stem, [o], [e, w, o]).
```

Input Stem: [m,o,k] {hot}
Set_internal_alter *results in:*
Valid_Stem=[m,e,w,o,k] which will be further analyzed for validity later

Input Stem: [m,o,k,e,r] {try} (wrong alternation)
Set_internal_alter *results in:*
Valid_Stem=[m,e,w,o,k,e,r] but it will fail letter in the template extraction as [1,0,1,0,1,0,1] is not among the learned templates

Experiments and Result

- Test Date:
  - verbs in their third person singular masculine form
  - Source: Online, Armbruster (1908)
- The verbs are inflected for the eight subjects and four tense-aspect-mood features of Amharic
  - Total Test set:
    - 1,784 distinct verb forms
- Accuracy:
  - 1,552=86.9%

Experiments and Result

- Error Analysis
  - absence of similar examples
  - inappropriate alternation rule

Test Word	*Stem*	*Root*	*Feature*	
[s,e,m,a,c,h,u]	*[s,e,m,a,?]*	*[s,m,?]*	*perfective, sppn*	Correct Analysis
[l,e,g,u,m,u]	*[l,e,g,u,m]*	*NA*	*NA*	No Similar Example
[s,e,m,a,c,h,u]	*[s,e,y,e,m]*	*[s,y,m]*	*gerundive, sppn*	Wrong Alternation

Challenge

The current learning trend demands examples for all combination of features for word formation.

- **Every subject for all tense and voice**
- **For various root groups/radicals**
- **For all object/negative/applicative/conj combinations**
- **For all forms with alternation**

- Can we (do we have to) exhaustively list all the combination?
- What correlation do we have between the constituents?
- How can we learn these interactions?

……..

Future work

- We can not give all possible combinations for Amharic
- Won't be generic otherwise

More Generic Approach: (Genetic Programming)

Generalizing from partial data….some morphemes can decide some feature of the word independent of the other constituents!

- Rules like..if it has the prefix 'te' then the word is passive despite all the other morphemes and template structure

```
stem(A, B, C, D):-
        set_affix(A, B, [t,e], [], S, []),
        root_temp(B, C, [e, e]),
        template(B, [1, 0, 1, 0, 1]),
        feature(D, [Ten, passive, Sub]).
```

S, *B*, *C*, *Ten* and *Sub* are variables and they can take any forms but the word (if it a valid Amharic word) will be Passive

Acknowledgement

- Special Thanks to:
  - IDRC-ICT4D project hosted by Nairobi University, Kenya
    - For sponsoring my conference participation
    - For partially supporting this project

አመሰግናችኋለሁ

a-mesegn-a-chʷa-lehu

I Thank you all

wondisho@yahoo.com
wondgewe@indiana.edu
and
gasser@cs.indiana.edu