## 1、Introduction

Task:

Given a lemma (the dictionary form of a word) with its part-of-speech, generate a target inflected form.

Example:

Source form and features: release V;NFIN

Target tag: V; V.PTCP; PRS

Target form: releasing

Data Format:

The training and development data is provided in a simple utf-8 encoded text format where each line in a file is an example that consists of word forms and corresponding morphosyntactic descriptions (MSDs) provided as a set features, separated by semicolons. The fields on a line are **TAB**-separated.

For our task, the fields are: lemma, target form, MSD.

An example from the English training data:

touch touching V;V.PTCP;PRS

In the training data, all three fields are given. During the test phase, field 2 is omitted. For example:

touch V;V.PTCP;PRS

we need to generate target form by MSD(V;V.PTCP;PRS)

## 2、Experiments -Baseline methods:

As a first step, the system aligns input/output training examples using Levenshtein distance. For example, the example

schielen geschielt V.PTCP;PST

is aligned as follows:

\_\_schielen

geschielt\_

The system now assumes that each input-output pair can be broken up into a prefixation part, a stem part, and a suffixation part, based on where the inputs or outputs have initial or trailing zeroes:

Pr St Su

\_\_|schiele|n

ge|schielt|\_

After this, the system extracts a set of prefix-changing rules based on the Pr pairings, and a set of suffix-changing rules based on St+Su pairings.

In this example, the following suffix-changing rules are extracted:

n$ > $

en$ > t$

len$ > lt$

elen$ > elt$

ielen$ > ielt$

hielen$ > hielt$

chielen$ > chielt$

schielen$ > schielt$

Likewise, the only prefix-changing rule extracted is the following:

$ > $ge

For languages with few prefix changes, the only prefix rule will often be $ > $

All these rules are associated with the complete MSD of an example, in this case**V.PTCP;PST.**

At generation time, the longest suffix rule that applies to a lemma form to be inflected is used. For example, if we're asked to inflect kaufen into V.PTCP;PST, we may find that, for example, en$ > t$ is the longest-matching suffix rule among all rules for V.PTCP;PST, which transforms:

$kaufen$ > $kauft$

and, following this, we find the most frequently seen prefix-changing rule for the target MSD in question that is applicable, in this case $ > $ge and apply that:

$kaufen$ > $gekauft$

## 2、Experiments -improve

I found in the existing original baseline program,

The logic of Baseline is to first extract rules for each row of data in the trial data set，then set the alignment of the train data and the train label, and then identify the prefix suffix. Next, According to the inflect rule (MSD) like ***V.PTCP;PST,*** extract the rule.

In the test data, I need to find the most frequently seen prefix-changing rule for the target MSD. At this time, we can only match one target MSD

Just like following data:

***take took V;PST***

In this data, we need to look for rules in ***V;V.PTCP***

but actually, maybe rule of ***V;V.PTCP;PST*** also works for this data.

So I expanded matching rules, when the target MSD and any other MSD contained in each other, I would mix them as a whole .

Experiments have proved that this method has some improvement effects on English, irish, and Slovak.

In addition, in order to ensure that the first matched MSD is a priority match, I set a low weight value for the newly added MSD to avoid affecting the original result.

Another solution：

Another solution is similar to the idea of random forest in machine learning. First, I split each row of training data,

such as ***take took V; PST,*** which will be split into

***take V -> took***, ***take PST -> took***, ***take V; PST-> tool***

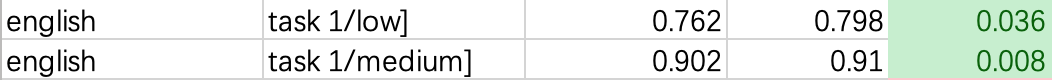
these three kinds of data, and replaced with the new data set.

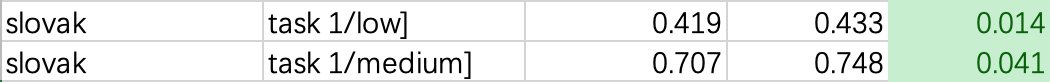
And in this step, randomly select 40% of the new data set as a

mini-data set to generate rules, use the generated rules to test the data, and get the label. The same step is executed 50 times, so for the same piece of test data, I will get 50 results, and select the most frequent one among them, and output it as the final result.

This solution is helpful to improve the accuracy of English language family, but other languages will have a negative impact(The codes for both solutions have been uploaded)

## 3、Evaluation and Error Analysis





For the above three languages, the third column is the accuracy obtained by the unmodified method, the fourth column is the modified, and the fifth column is the degree of change.

The current question is whether to generate rules only from prefixes and suffixes, but not from lemma itself. Some attempts have been made to use the method used in Semi-supervised learning of morphological paradigms and lexicons in the paper, but they did not succeed.

**4、Conclusion**

Basic conclusion: If we improve the algorithm from the perspective of adding matching rules, we can get some small improvements, but not very significant. If we want to improve from traditional methods, we need to study the structure of each language, not just Prefix suffix