# Use regular expression to find interesting pattern in English.

Finding common suffix is useful. Finding words with common suffix can help finding the stem of a word and help tagging words. So I developed a little Python program to help find word stems with common suffix in morphology translation.

For example, the naïve way of find words with specific suffix like ‘-s’ is using the regular expression ‘\b(\w)s\b’, but this cause problems, like word ‘basis’, which the last ‘s’ is not a suffix, will be identify as a suffix, and when processing word ‘achieves’, naïve approach of using regular expression lack the power of make distinct the between is the word with suffix ‘-s’ or ‘-es’.

In that situation, I propose a new approach to use regular expression to process finding common morphology suffix task. I call this look forward and back approach. Generally speaking, as large corpus it is, a word is likely to occur in the text as its transformation with a particular suffix occurs. So when a word with pattern of suffix is found, the regular expression will looking forward and back. If the word stem occur at least once, I assume that the word stem is a intact word instead of a part of a word which end with the same letters of the suffix.

To achieve this, I made some modification of the code in ‘regexs.py’ file. First, the regex code read each line from a file and finds the pattern in each line separately. To look forward and back in the entire corpus, it needs to search the pattern in a single string. Second, by experiment, I found the looking forward and back approach has some shortcomings. To overcome it and future the experiment, it needs some statistic functions.

The regular expression of finding a word stem occurs with a specific suffix before it occurred is ‘\b(\w+) \b.\*\b\1NEWSUFF\b’, in case of some morphology transformation like the word ‘company’ to ‘companies’, O introduce an old suffix in the regular expression, so the updated regular expression is ‘\b(\w+)OLDSUFF\b.\*\b\1NEWSUFF\b’, and for finding word stems with a specific new suffix occur before itself, the regular expression should be ‘\b(\w+)NEWSUFF\b.\*\b\1OLDSUFF\b’, to have the union of the two set, I combine those two regular expression like: ‘\b(\w+)OLDSUFF\b.\*\b\1NEWSUFF\b|‘\b(\w+)NEWSUFF\b.\*\b\2OLDSUFF\b’, this are the regular expression for find word stem with the old suffix occurs at least once beside the word stem with new suffix occurred. Beside, to find word stem with the old suffix occurs at least twice beside the word stem with new suffix, the regular expression is: ‘\b(\w+)OLDSUFF\b.\*\b\1OLDSUFF\b.\*\b\1NEWSUFF\b|\b(\w+)NEWSUFF\b.\*\b\1OLDSUFF\b.\*\b\1OLDSUFF\b|\b(\w+)OLDSUFF\b.\*\b\1NEWSUFF\b.\*\b\1OLDSUFF\b’, those regular expression is hard coded in my python code, each time running just replace the ‘OLDSUFF’ and ‘NEWSUFF’ part in the string with interesting suffixes.

To run this code, at least three system arguments is needed: the first one is the old suffix, which can be an empty string, the second argument is the new suffix, and from the third argument is the file path which contain the text. For examples, to find word with ‘-s’ suffix, the command is: ’python suffix.py ‘’ ‘s’ ./../gutenberg/\*.txt’, and the outcome is in Figure 1:

s

\b(\w+)\b.\*\b\1s\b|\b(\w+)s\b.\*\b\2\b

[('a', 2722), ('i', 1997), ('it', 166), ('mr', 107), ('son', 51), ('day', 38), ('offering', 24), ('shekel', 19), ('god', 18), ('king', 15), ('whale', 14), ('thing', 13), ('heaven', 12), ('one', 12), ('year', 12), ('face', 10), ('hand', 9), ('stone', 9), ('word', 9), ('your', 9), ('her', 8), ('house', 8), ('know', 8), ('tree', 8), ('tribe', 8), ('way', 8), ('lord', 7), ('servant', 7), ('time', 7), ('duke', 6), ('eye', 6), ('side', 6), ('curtain', 5), ('dream', 5), ('gate', 5), ('his', 5), ('land', 5), ('look', 5), ('nation', 5), ('priest', 5), ('cubit', 4), ('end', 4), ('fig', 4), ('friend', 4), ('hold', 4), ('lion', 4), ('lot', 4), ('name', 4), ('our', 4), ('praise', 4), ('prophet', 4), ('book', 3), ('come', 3), ('cover', 3), ('deal', 3), ('father', 3), ('hat', 3), ('heart', 3), ('judge', 3), ('light', 3), ('love', 3), ('month', 3), ('part', 3), ('post', 3), ('prince', 3), ('return', 3), ('ship', 3), ('talent', 3), ('tell', 3), ('their', 3), ('thought', 3), ('thousand', 3), ('work', 3), ('ye', 3), ('altar', 2), ('angel', 2), ('as', 2), ('beast', 2), ('bell', 2), ('board', 2), ('bone', 2), ('captive', 2), ('chariot', 2), ('cloud', 2), ('colour', 2), ('conjecture', 2), ('daughter', 2), ('ear', 2), ('equal', 2), ('feast', 2), ('find', 2), ('finger', 2), ('girl', 2), ('head', 2), ('horn', 2), ('idol', 2), ('jaw', 2), ('jew', 2), ('lamb', 2), ('leader', 2), ('letter', 2), ('live', 2), ('master', 2), ('measure', 2), ('member', 2), ('moment', 2), ('mother', 2), ('noise', 2), ('officer', 2), ('other', 2), ('own', 2), ('please', 2), ('province', 2), ('reed', 2), ('right', 2), ('river', 2), ('rod', 2), ('rose', 2), ('row', 2), ('sacrifice', 2), ('sail', 2), ('say', 2), ('second', 2), ('see', 2), ('seed', 2), ('shoe', 2), ('sister', 2), ('slave', 2), ('sleep', 2), ('song', 2), ('star', 2), ('state', 2), ('step', 2), ('stranger', 2), ('thank', 2), ('valley', 2), ('voice', 2), ('voyage', 2), ('walk', 2), ('wheel', 2), ('wind', 2), ('wing', 2), ('with', 2), ('absorb', 1), ('admire', 1), ('american', 1), ('anarchist', 1), ('ancient', 1), ('animal', 1), ('answer', 1), ('antichrist', 1), ('appear', 1), ('artist', 1), ('author', 1), ('basket', 1), ('bason', 1), ('beare', 1), ('become', 1), ('beget', 1), ('bird', 1), ('birth', 1), ('bishop', 1), ('blade', 1), ('blunder', 1), ('boat', 1), ('boil', 1), ('bottle', 1), ('brace', 1), ('breast', 1), ('build', 1), ('building', 1), ('bullock', 1), ('buy', 1), ('calm', 1), ('cannibal', 1), ('cape', 1), ('capitol', 1), ('card', 1), ('case', 1), ('catholic', 1), ('cave', 1), ('cease', 1), ('censer', 1), ('chaise', 1), ('chimney', 1), ('chowder', 1), ('christian', 1), ('circuit', 1), ('command', 1), ('companion', 1), ('compend', 1), ('constable', 1), ('continent', 1), ('contradiction', 1), ('corner', 1), ('corroborate', 1), ('countenance', 1), ('court', 1), ('cousin', 1), ('crew', 1), ('dame', 1), ('dance', 1), ('dart', 1), ('death', 1), ('decree', 1), ('delay', 1), ('depart', 1), ('disciple', 1), ('disguise', 1), ('ebook', 1), ('eel', 1), ('elder', 1), ('enter', 1), ('escape', 1), ('evidence', 1), ('exclude', 1), ('experience', 1), ('fade', 1), ('fail', 1), ('faith', 1), ('fantastic', 1), ('feed', 1), ('fine', 1), ('firebrand', 1), ('fit', 1), ('flock', 1), ('forehead', 1), ('front', 1), ('garden', 1), ('gift', 1), ('give', 1), ('globe', 1), ('graft', 1), ('grave', 1), ('groan', 1), ('guinea', 1), ('ha', 1), ('hanging', 1), ('harpooneer', 1), ('hearse', 1), ('heauen', 1), ('hebrew', 1), ('hee', 1), ('hill', 1), ('hip', 1), ('host', 1), ('household', 1), ('how', 1), ('hundred', 1), ('husband', 1), ('hut', 1), ('immerse', 1), ('indicate', 1), ('inhabitant', 1), ('innkeeper', 1), ('intimate', 1), ('joy', 1), ('kick', 1), ('knop', 1), ('lack', 1), ('landscape', 1), ('lark', 1), ('laver', 1), ('lay', 1), ('leak', 1), ('leaue', 1), ('leave', 1), ('lie', 1), ('line', 1), ('lust', 1), ('mast', 1), ('material', 1), ('me', 1), ('mean', 1), ('mechanic', 1), ('melon', 1), ('mesopotamian', 1), ('million', 1), ('minister', 1), ('motive', 1), ('mout', 1), ('movement', 1), ('mum', 1), ('needle', 1), ('network', 1), ('niece', 1), ('notion', 1), ('number', 1), ('o', 1), ('oblation', 1), ('occupant', 1), ('offence', 1), ('orb', 1), ('out', 1), ('painting', 1), ('palm', 1), ('pardon', 1), ('pay', 1), ('pearl', 1), ('penetrate', 1), ('person', 1), ('phase', 1), ('pillar', 1), ('pine', 1), ('place', 1), ('planet', 1), ('plant', 1), ('plough', 1), ('pomegranate', 1), ('pot', 1), ('power', 1), ('proverb', 1), ('puzzle', 1), ('question', 1), ('rabbit', 1), ('realm', 1), ('receive', 1), ('religion', 1), ('reubenite', 1), ('road', 1), ('rogue', 1), ('roof', 1), ('rule', 1), ('run', 1), ('sailor', 1), ('sake', 1), ('scale', 1), ('screw', 1), ('seat', 1), ('secret', 1), ('serjeant', 1), ('shaker', 1), ('shape', 1), ('shark', 1), ('shepherd', 1), ('sight', 1), ('sit', 1), ('sixpence', 1), ('smile', 1), ('snow', 1), ('soldier', 1), ('soul', 1), ('sound', 1), ('spirit', 1), ('splendor', 1), ('spout', 1), ('stage', 1), ('sting', 1), ('stop', 1), ('street', 1), ('strife', 1), ('sub', 1), ('survivor', 1), ('sweet', 1), ('swing', 1), ('sword', 1), ('table', 1), ('take', 1), ('tear', 1), ('tete', 1), ('think', 1), ('threshold', 1), ('thunderbolt', 1), ('tithe', 1), ('toe', 1), ('ton', 1), ('tongue', 1), ('treasure', 1), ('turtle', 1), ('two', 1), ('view', 1), ('vineyard', 1), ('vision', 1), ('vote', 1), ('vow', 1), ('want', 1), ('water', 1), ('wedding', 1), ('weight', 1), ('weston', 1), ('whoredom', 1), ('woe', 1), ('wonder', 1), ('wreath', 1)]

384

\b(\w+)\b.\*\b\1\b.\*\b\1s\b|\b(\w+)s\b.\*\b\1\b.\*\b\1\b|\b(\w+)\b.\*\b\1s\b.\*\b\1\b

[('i', 88), ('a', 44), ('it', 6), ('duke', 3), ('look', 3), ('son', 3), ('god', 2), ('king', 2), ('cape', 1), ('dream', 1), ('gift', 1), ('guinea', 1), ('heaven', 1), ('her', 1), ('lord', 1), ('month', 1), ('nation', 1), ('offering', 1), ('our', 1), ('thousand', 1), ('ye', 1)]

21

Figure 1

In the first block it prints out which word occur once and occurred with suffix ‘-s’ associate with its time of occurrence, and in the second block it print out word occurred twice.

I run the code with the following old and new suffix in Figure 2

|  |  |  |  |
| --- | --- | --- | --- |
| old suffix | new suffix | Number of word occurred once | Number of words occurred twice |
| Null | ed | 142 | 2 |
| Null | er | 49 | 0 |
| Null | es | 24 | 4 |
| Null | est | 26 | 2 |
| Null | ing | 99 | 1 |
| Null | ly | 34 | 1 |
| Null | ness | 19 | 1 |
| Null | s | 384 | 21 |
| y | ied | 11 | 0 |
| y | ier | 3 | 0 |
| y | ies | 22 | 1 |
| y | iest | 3 | 0 |

Figure 2

By finding those patterns, I found something interesting. Even though the looking forward and back approach can eliminate the possibility of retrieval some words with the ending letter same as the suffix, this approach cause some other problems. It will find a pair of different word, which one of it appending the suffix is the other word. For example, for suffix ‘-s’, the word ‘a’ as ‘as’ are a pair of different word which one of it append the suffix will be the other one.

And there is an interesting phenomenon of those kinds of pair of words. By looking for the count of occurrence, if a word stem’s occurrence is significantly greater than normal one. It is more likely to be this kind of words. For suffix ‘-ed’, the word stem ‘he’ occurs 12 times, which normal word only occur once. For suffix ‘-es’, the word stem ‘do’ occurs 41 times and ‘on’ occurs 8 times, which normal word only occur once. For suffix ‘-est’, the word stem ‘he’ occurs 13 times, which normal word only occur once. For suffix ‘-ly’, the word stem ‘on’ occurs 96 times, which normal word only occur once. For suffix ‘-ed’, the word stem ‘a’ occurs 2722 times, ‘i’ occurs 1997 times, ‘it’ occurs 166 times and ‘mr’ occurs 107 times, which normal word only occur once.

However, until now, I didn’t find any approach to supports that. By instinct, I guess if the founded word is actually with the given suffix, it occurrence may have some relation with its word stem. Since a morphology transformation in English between two word usually have a similar meaning, but like word pair ‘a’ and ‘as’, which each one has total different meaning with each other, the relation between the occurrence may be zero. But now I don’t know how to quantify the relation can to compare it. Future study can be apply to this area.