Extraction of Callsigns

Table of Contents

This is the second part of the exercise to be done until end of the lecture.

Learning Objectives

- Using STL
- Programming in a team, you may work in teams up to 5

Exercise at a glance

- Implement a real callsign extraction function
- Implement a more efficient callsign extraction
-

Detailed Exercise Description

Exercise 4-4: Who is doing what in your team

Send a short description, who will do what in your team, if you are working in a team. If you are working alone this is not necessary. It is currently enough to send a plan just for the part of the exercise described in this file.

If you are sending me this description until 2024-11-29 via email you have time for the whole exercise until 18-12-2024 otherwise until end of November.[1]

Exercise 4-5: Simple extraction of callsigns

Your input files have the same format as in exercise 4.1.

Assume the callsign is in the beginning, but not always the first words
Examples are

- "speed bird eight seven alfa sierra identified climb flight level two three zero". The expected output would be "BAW87AS". Here the callsign was in the very first words.
- "gruess gott austrian triple seven sierra identified climb flight level two three zero". The expected output would be "AUA777S". Here the callsign was not in the very first words.
- "bonjour air france eighty seven double alfa identified climb flight level two three zero". The expected output would be "AFR87AA". The numbers are not correctly pronounced (eighty seven said and not eight seven)
- "faroe line eight fox x-ray identified climb flight level two three zero". The expected output would be "FLI8FX". Foxtrot could be abbreviated by fox. The other letters are not abbreviated.
- "fox lima indiae eight fox x-ray identified climb flight level two three zero". The expected output would be "FLI8FX". Sometimes the 3-letter code is fully spelled.
-
- "easy foxtrot six alfa identified" or "easy jet foxtrot six alfa identified" The expected output would be in both cases "EZYF6A". Some three letter codes code have multiple combinations of word sequences.
- "oscar echo six kilo golf identified" The expected output would be in both cases "OE6KG". Small aircraft have no three-letter code, but have just 5 to 8 digits and letters.

The callsign could be also at the end. This is the case, when the pilot is speaking:

- "descending eight thousand feet direct dexon speed bird twenty nine seventeen". The expected output would be "BAW2917". The callsign is even at the end of the utterance.
- The callsign is not always in the beginning: "climbing flight level two three zero austrian triple seven sierra".

Exercise 4-5 Extraction of Multiple callsigns

- "speed bird twenty nine seventeen standby lufthansa four double alfa after air france taxi via november november 8

[1] I hope it is clear. It is a must to send that list. You can change at any time, but you should have already now an idea, what you want to do (and I can give then early feedback whether you are working in the right direction or not).

to delta four eight one". The expected output would be "BAW2917" and "DLH4AA". "air france" is not used as a callsign.

Often you have then the words "break break" or "standby2 in the utterance, but you cannot rely on, e.g. when the speech recognizer did not recognize these words.

Exercise 4-6 Use Context information

Use the information of the available callsigns in the air.

- "delta echo six delta golf identified" The expected output would be "DALE6DG". "delta" is a three-letter code (DAL) and "delta" is also a letter of the NATO alphabet.

The information of the available callsigns in the air at the time the utterance is spoken can be helpful, see format of input file in exercise 4-1. Here again an example:

```
2017-08-13__15-00-08-22:
  sky travel golf hotel dobry den praha radar radar contact descend flight level one six zero proceed papa romeo five three zero no speed restrictions
    Csgn:  BMS9977 TVS2GH DLH714 NTF316 OKIFR
    TVS2GH STATION RADAR
    TVS2GH INIT_RESPONSE
    TVS2GH DESCEND 160 FL
    TVS2GH DIRECT_TO PR530 none
    TVS2GH NO_SPEED_RESTRICTIONS
```

The mapping to TVS2GH is only possible, when you know, that TVS2GH is in the air, as specified after the key "Csgn"

The challenge is here that you generate for each callsign a container of word lists, which can be use to address a callsign. The file ExampleForWordLists.pdf (also on my homepage) contains examples for some callsigns. Air traffic controllers are very innovative. Even more combinations might exist.

Exercise 4-7 Use Context information for correcting errors of Speech Recognition

```
2017-08-13__15-00-08-22:
  sky travel three golf hotel dobry den praha radar radar contact descend flight level one six zero proceed papa romeo five three zero no speed restrictions
    Csgn:  BMS9977 TVS2GH DLH714 NTF316 OKIFR TVS3LK
    TVS2GH STATION RADAR
    TVS2GH INIT_RESPONSE
    TVS2GH DESCEND 160 FL
    TVS2GH DIRECT_TO PR530 none
    TVS2GH NO_SPEED_RESTRICTIONS
```

"three" is recognized, but "two" would be correct. Here it is possible to "guess" that TVS2GH might be correct. Be careful that you do not accept too many errors. It is always a trade-off between recognition rate and error rate. A low error rate is more important.

Use your implementation of the Levenshtein distance calculation for this task. Try to find out which words of the utterance could correspond to the callsign and then calculate the Levenshtein distance for all word combinations which could be (in the above case) created for the callsigns BMS9977, TVS2GH, DLH714, NTF316, OKIFR and TVS3LK. Take the nearest one, which is not too far away. Some heuristic could be helpful.

Also consider that in some cases the correct callsign is missing.

2017-08-13__15-00-08-22:
faro line two golf india dobry den praha radar radar contact descend flight level one six zero proceed papa romeo five three zero no speed restrictions
Csgn: BMS9977 TVS2GH DLH714 NTF316 OKIFR
TVS2GH STATION RADAR
TVS2GH INIT_RESPONSE
TVS2GH DESCEND 160 FL
TVS2GH DIRECT_TO PR530 none
TVS2GH NO_SPEED_RESTRICTIONS

Here TVS2GH is also the nearest, but maybe this is the wrong one.

Exercise 4-8 Implement a faster Levenshtein distance calculation

Consider the following example

2020-08-08__06-22-46-22
easy three eight alfa climb to altitude six thousand feet heading zero nine zero after southhampton climb to flight level nine zero
Csgn: BCS6789 BAW2642 BAW2606 BAW138 BAW366U BAW48CU BAW14K RYR5993 BAW664C BAW526 EZY51WY BCS6891 EZY15WG EZY1817 RYR4AZ EZY38HA EZY49BN EZY73FX EZY83FD EZY87ET EZY93EB FTL632 KLM1000 WZZ14B LZB851 REV72B RYR13ET RYR2315 TOM7EX RYR3QM WZZ19DD RYR46HL RYR4GQ RYR52AH UAL931 RYR5QP RYR6GM RYR62VX RYR64TT RYR81ZW WUK5V WUK934 WZZ102 WZZ183
EZY38HA CLIMB 6000 ft
EZY38HA HeADING 090 none
EZY38HA CLIMB 90 WHEN PASSING SAM

Here we have 45 callsigns in the list. Up to 150 might be possible.

We have more than 20 spoken words.

For some callsigns 100 possible word combinations exist with up to ten words.

The complexity of the Levenshtein distance algorithm is O(nxm), where n and m are the lengths of the two words, in the example 10 by 20.

If you have 50 callsigns and 100 word-combinations for each callsign you have to calculated 5000 Levenshtein distance each with an effort of 200. You are in the order of 1.000.000 string comparisons. And maybe you need to check for each of the Levenshtein distance calculation, if the Levenshtein distance is not too big. A simple threshold is maybe not a good solution. “lufthansa one” instead of “easy one" maybe worse than the distance of two in “air france one six alfa one” and “air france alfa six alfa one”.

Try to find something more efficient, which also finds (always)the best match and also implement a heuristic, which mostly finds the best match. Consider that you only need the Levenshtein distance and not the needed steps (function backtrace), at least not for all combinations.

Evaluation criteria

This will be a combination of how many correct callsigns you find and the how fast you are.

Therefore, first implement a naïve, but slow, algorithm for finding the best Levenshtein distance and then concentrate on a good heuristic, ignoring first to generate all the word sequences for a callsign.

It is not necessary to implement also a fast algorithm to find the best match, if the spoken words are correct, i.e. at least one word sequence for the given callsigns matches the word sequence.
Your algorithm is evaluated on known utterances (test cases), but also on unknown test cases, which I have on my computer.