Karlsruhe Institute of Technology

Online Neural Network-based Language Identification

Master's Thesis of

Daniel H. Draper

at the Department of Informatics
Institute for Anthropomatics and Robotics

Reviewer:	Dr.-Ing. Sebastian Stüker
Second reviewer:	
Advisor:	M.Sc. Markus Müller

12. December 2016 – 11. May 2017

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 12th of May, 2017

. . . . . .

(Daniel H. Draper)

Abstract

Zusammenfassung

Contents

List of Figures

List of Tables

1. Introduction

Language Identification (LID) describes the classification task of differentiating between spoken speech in different languages and being able to correctly classify which speech-segments consists of which language. Neural Networks refer to Artificial Neural Network's, a Machine Learning approach to classification tasks employed greatly throughout all sciences and especially in computer science and tasks concerned with the processing of spoken speech. This thesis tries to find a low-latency, fast, or "online", approach to Language Identification using the classification method of neural networks.

The following chapter gives an introductory view of the applications of Language Identification, and introduces the tasks this thesis tries to solve. It also presents related work and how this thesis can be put into perspective to those works. Afterwards we give preliminary theoretical explanations and definitions, including an introduction to Neural Networks in Sec. **??**. The chapter afterwards introduces the language identification tasks this thesis deals with, and the different data corpusses used, followed by our solution split into two chapters: the preprocessing and actual neural networks trained with its training results. Approaches used to further smooth over results in an online environment are then presented followed by the conclusion and an outlook.

The work done in this thesis uses the Janus Recognition Toolkit (jrtk)[1], an Automatic Speech Recognition toolkit developed in joint cooperation by the KIT and CMU. The jrtk offers a tcl/tk[2] script-based environment for the development of Automatic Speech Recognition systems, therefore source code in this thesis will consist of tcl/tk scripts with (some) janus-specific commands. The jrtk and tcl/tk are further explained in sec. **??**.

1.1. Applications

Automatic Speech Recognition (ASR) is used in many applications and devices today, especially in the rise of handheld mobile devices like smart-phones and tablets. It has progressed quickly in the last five years and has found commercial success. Famous examples include Google[3]'s "Ok, Google" and Apple[4]'s Siri. Which both include voice search[FHBM08] and a form of voice control, that even is extensible in the case of Google and Android e.g[bAO14]. Many other applications have emerged, including spoken language translation[5], especially relevant for this thesis in the realm of Lecture Translation[MNN+16] .

[1] Janus Recognition Toolkit: `http://isl.anthropomatik.kit.edu/cmu-kit/english/1406.php`
[2] Tcl/tk: `https://www.tcl.tk/`
[3] Google: `www.google.com`
[4] Apple: `www.apple.com`
[5] IWSLT: `iwslt.org`

The task of Language Identification can be applied in all of those fields, as Automatic Speech Recognition is mostly trained on one language and therefore requires a totally different setup of classifiers per language, making a manual change of language previous to recognition necessary. Robust and low-latency language identification would eliminate the need for this.

LID would be especially applicable in the realm Spoken language translation, as used for example in the European Parliament where already components of ASR and Machine Translation are employed and are being actively developed[6][VMH+05], and LID would further be able to automate these translation tasks.

This thesis will focus mostly on the KIT's lecture Translator[7] as the system trained was implemented for it. We believe our results are general enough to be transferable to other applications with small implementation-specific changes.

1.2. Related Work

This section takes a look at related work that shows different approaches of identifying Language in spoken speech and describe the differences between their work and our approach.

[6]TC-STAR: `tcstar.org`

[7]Lecture Translator: `https://lecture-translator.kit.edu`

2. Preliminary Definitions

In the following chapter we want to define and explain terms and concepts used throughout this thesis as well as give an outlook to related work and the general language identification approaches.

3. Language Identification Tasks

This chapter introduces the datasets used to train the networks employed in this approach. While Language Identification is applicable in many different scenarios, in this thesis the focus lies on trying to establish a low-latency online approach for recognizing the spoken language in a university-lecture environment. Because finding a suitable test setup for online data retrieval is hard the data used was cut to short lengths to make an evaluation as to correctness of the recognition possible in an ”online-like” scenario.

This means that the output of the net is evaluated after short samples of speech and therefore can be seen as indicative of online performance of the neural net.

3.1. Euronews 2014

Our first data set we retrieved from Euronews [1] 2014. Euronews is a TV channel that is broadcast in 13 different languages simultaneously both on TV and over the Web and is semi-automatically transcribed. The data corpus includes10 language (Arabian, German, Spanish, French, Italian, Polish, Portuguese, Russian, Turkish and English) with about 20 hours of data per language provided overall. Details of this breakdown can be seen in table 3.1.

Language	**Number of Speakers**	**Length overall**
Arabian	1055	
German	928	
Spanish	932	
French	1016	
Italian	935	
Polish	1229	
Portuguese	1062	
Russian	958	
Turkish	957	
English	928	
Overall	10000	

Table 3.1.: The Euronews corpus speaker breakdown with total utterances length

The speaker list was then split into three smaller datasets: the train set, development set and test set using the common Simple Random Sampling. The sizes were 80% train set,

[1] Euronews: http://www.euronews.com/

and 10% for both the development and test set. Table **??** shows the split data for the three sets.

3.2. Lecture Data

3.3. European Parliament

4. Feature Preprocessing

This chapter deals with the feature preprocessing used to form normal speech into feature vectors to be understood by neural networks. The setup we used is based on the standard capabilities of the Janus Recognition Toolkit. It is then run through a six layer Automatic Speech Recognition network that was pre-trained on 10 languages. The 2^{nd} last layer of the ASR net is a Bottleneck feature layer, where the feature vectors are extracted and then used as input for the trained Language Identification Network. The following sections describe this Feature Preprocessing for data as well as the first ASR network used to create the BNF features the LID net requires.

4.1. Feature Access

4.2. Feature Description

The extracted ADC features from the audio files are then used for further preprocessing. We first use a standard Mel filter bank to extract only the necessary coefficients from the ADC0 feature.

First a spectrum is applied to the ADC0 Feature, therefore calculating the Fast Fourier Transformation of the Digitalized Signal.

4.3. ASR BNF network

5. LID Network

This chapter describes the actual Language Identification Neural Network trained as well as the results of different network/data setups used. Most network experiments in the Network Geometry, meaning the number of hidden layers as well as the layout of the neurons in these layers were tried using the Euronews corpus. Results from this corpus were then transferred over to the other corpusses, meaning that the network layout that worked best for Euronews was then adjusted for the lecture data but otherwise the geometry was kept intact.

5.1. Basic Setup

Many different tools exist for deeplearning, the most acclaimed being Tensorflow [AAB+16], DL4J/ND4j[1] and Theano [BBB+11]. Pre-existing work on Language Identification using ASR BNFs used a python wrapper around Theano for training, that was developed by Jonas Gehring [Geh12]. This thesis continues the use of this wrapper. The basic layout of the network used and improved upon by this thesis were input vectors from the ASR net 4.1 with 966 coefficients. The net setup were 5 layers of denoising auto-encoders with each 1000 nodes and a tanh activation function using the mean squared error as the loss function.

The neural net was then trained using mini batches of size 2m and a learning Rate of 0.01. The pretrained net was then retrained with a 1000 neuron to 10 coefficient output to get to our 10 Languages as classes to classify against. In the basic setup this was trained using a learning rate of 1 and the exit condition of a minimum change of 0.005 / 0.0001 for the training/validation data respectively.

The beginning benchmark to improve upon was then set to the frame-based validation/-train error of 0.23 / 0.27 respectively.

The following sections describe different network layouts and changes we made to the training of the neural network and the improvements we managed to make upon our initial result.

5.2. Improving Network Layout

Different experiments were undertaken with the network layout. This includes a 6 hidden layer-pretraining as well as a change in the geometry. The differences in the frame-based errors can be seen in Table **??**

[1] DL4J: `https://deeplearning4j.org/index.html`

6. Output smoothing

While frame-based error rates on the training/validation sets were already sufficiently good from the LID networks, this of course is not a reliable indicator of real-world online performance, so the development setup consisted of (for each data corpus) a development set of speakers whose samples were run through the LID setup (Feature Preprocessing Sec. 4 → ASR BNF extraction Sec. 4.3 → LID network Sec. 5). The following smoothing approaches were applied in an ”online” fashion, meaning it was made sure they can be calculated “on-the-fly” while new data is still coming in.

6.1. Basic Test Filter

The first filter tried was a basic 5-Frame smoothing: It saves the value of the last direct outputs and only outputs a language if the last 5 direct outputs would have been the same. It also includes a filtering based on the actual output of the language ID neuron, only counting outputs higher than that. This of course means that the approach requires a 5 frame ($\approx 50ms$) “warmup” time, which still would make it usable in an online environment.

See Lst. 6.1 for the corresponding tcl/tk code for this basic filtering approach. The test setup used then goes through the entire development set of samples and counts the correctly/wrongly classified samples. This means that an extra amount of smoothing is included, but results should still be sufficiently general to be able to infer properties of employed filters, as they all include this extra smoothing. Table **??** shows a comparison of the basic filter with a bare setup only outputting the direct output of the LID net. Fig. **??** shows a comparison of the length of samples and the amount of correctly classified samples of this length for the LID net with the best evaluation results, the 6-layered geometry-adjusted lower learning rate net (See Sec. **??**)

```
proc filter{} {
    #setting up variables
     for {set i 0} {$i < 10} {incr i} {
        set totalM($i) 0
    }
    set lastFrameID -1
    set counter 0
    set currentOutput -1
    #Going through whole sample frame by frame in output layer of nn
       called nnBNF-> can be changed to work on continuously incoming
       data easily
    for {set i 0} { $i < [featureSetLID frameN nnBNF]} {incr i} {
        #we find the current output of the net
        set maxFrame [lindex [lsort -decreasing -real [featureSetLID
           frame nnBNF $i]] 0]
```

```
13          set maxFrameID [lsearch -real [featureSetLID frame nnBNF $i]
                $maxFrame]
14          #Actual Filtering: Only count if last 4 frames were also
                classified to be this language
15          if {$maxFrameID != $lastFrameID} {
16             set lastFrameID $maxFrameID
17             set counter 0
18          #Also filter for the actual output of the neuron
19          } elseif {$maxFrame >= 0.61} {
20             incr counter
21             if {$counter >= 5} {
22                set currentOutput $maxFrameID
23             }
24          }
25          #setting total classification amounts for current sample
26          if {$currentOutput != -1} {
27               set totalM($currentOutput) [expr {[set totalM(
                    $currentOutput)] + 1}]
28          }
29       }
30       set maxOverall -1
31       set maxID -1
32       #Now get the output for the whole sample (smoothing
33       for {set i 0} {$i < 10} {incr i} {
34          if {[set totalM($i)] > $maxOverall} {
35              set maxOverall [set totalM($i)]
36              set maxID $i
37          }
38       }
39       #print out the total classification for the entire sample
40       puts -nonewline "Overall we have classified as: "
41       #help function to print language name not id.
42       puts [getName $maxID]
43       return $maxID
44    }
```

Listing 6.1: Most basic filter employed to smooth output

6.2. Advanced Test Filter

The advanced test filter is based on the jrtk's *FILTER* capability. It automatically takes a defined amount of frames and calculates the weighted arithmetic mean with predefined weights for incoming audio. First tries were done using a basic filter setup of:

```
filter          nnFILTER   nnBNF  {-2 {1 2 3 2 1}}
```

Herein the context is 2 frames on each side of the current frame (the first parameter) with weights 1, 2 , 3 , 2 , 1 for the 5 frames respectively. Applying our basic setup from above on this of smoothing on this weighted average output it turned out, as expected, that the context of only 4 frames was too small.

It was adjusted to use a 10-frame based approach with descending weights for frames further out:

```
filter          nnFILTERREAL nnBNF {-5 {1 2 3 4 5 6 5 4 3 2 1}}
```

which produces better results and an improvement on the basic setup of about 0.5 % on average for the different net setups. Evaluation results are listed in Tab. **??**. The full tcl code for this can be found in the Appendix.

6.3. Variance Test Filter

As a next approach, the variance between the two most likely outputs was taken into account. This however, did not lead to an improvement in the robustness of the classification on the development set. The reason possibly being that the output of the 2^{nd} most likely neuron is not going to differ from the maximum output on many different language pairs: E.g. French/Italian, Russian/Polish, Italian/Portuguese. The full tcl code of this filter can be found in the appendix . The evaluation results can be seen in Tab. **??**

7. Conclusion

Bibliography

[AAB+16] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016.

[bAO14] N. bt Aripin and M. B. Othman. Voice control of home appliances using android. In *2014 Electrical Power, Electronics, Communicatons, Control and Informatics Seminar (EECCIS)*, pages 142–146, Aug 2014.

[BBB+11] James Bergstra, Frédéric Bastien, Olivier Breuleux, Pascal Lamblin, Razvan Pascanu, Olivier Delalleau, Guillaume Desjardins, David Warde-Farley, Ian Goodfellow, Arnaud Bergeron, et al. Theano: Deep learning on gpus with python. In *NIPS 2011, BigLearning Workshop, Granada, Spain*, volume 3. Citeseer, 2011.

[FHBM08] A.M. Franz, M.H. Henzinger, S. Brin, and B.C. Milch. Voice interface for a search engine, April 29 2008. US Patent 7,366,668.

[Geh12] Jonas Gehring. *Training deep neural networks for bottleneck feature extraction.* 2012. Karlsruhe, KIT, Pittsburgh, Carnegie Mellon Univ., Interactive Systems Laboratories, Masterarbeit, 2012.

[MNN+16] Markus Müller, Thai Son Nguyen, Jan Niehues, Eunah Cho, Bastian Krüger, Thanh-Le Ha, Kevin Kilgour, Matthias Sperber, Mohammed Mediani, Sebastian Stüker, and Alex Waibel. Lecture translator - speech translation framework for simultaneous lecture translation. In *Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 82–86, 2016.

[VMH+05] David Vilar, Evgeny Matusov, Saša Hasan, Richard Zens, and Hermann Ney. Statistical machine translation of european parliamentary speeches. In *Proceedings of MT Summit X*, pages 259–266, 2005.

A. Appendix

In this Appendix we present images, complete source code listings as well as a Glossary at the end.