Group Project on

Natural Language Processing (CS 491)

Academic Year- 2019-20

On

Text-Based Language Identification for Indian Languages

Submitted by

1. Anjali Mishra BT17GCS157
2. Arundhati Das BT17GCS016
3. Ayushi Kapoor BT17GCS020
4. Arushi Sehgal BT17GCS017

INDEX

Introduction:

Language identification has always been a challenging issue and an important research area in Natural Language Processing . It is the task of automatically detecting language(s) from a given text or document. In this work, we address the problem of detecting languages from the document that contain text from monolingual and multilingual documents. We have implemented our code in python using the iNLTK library which uses pre-trained language models in an ensemble to determine which language, or languages, an unknown input text/document is written in. Indian Language Identification is a prerequisite for many applications like detecting the source language for machine translation, information retrieval, summarization etc.Our language model is designed to identify five Indian Languages:Punjabi, Hindi, Sanskrit, Tamil, Bengali. Also in a multilingual society like India there is wide scope for automatic language identification since it would be a vital step in bridging the digital divide between the Indian masses and the world.

Problem statement:

Language Identification of Indian Languages of a given text document containing monolingual and multilingual texts . Our model is mainly focusing on identification of 5 Indian Languages:

1. Bengali,
2. Punjabi,
3. Tamil ,
4. Sanskrit and
5. Hindi.

Literature review:

Jeremy Howard, Sebastian Ruder(2018) the study proposed a Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method which can be applied to any task in NLP, and introduced the techniques that are the key for fine-tuning a language model. In this study the method significantly outperforms the state-of-the-art on six text classification tasks, reducing the error by 18-24 percent on the majority of datasets.

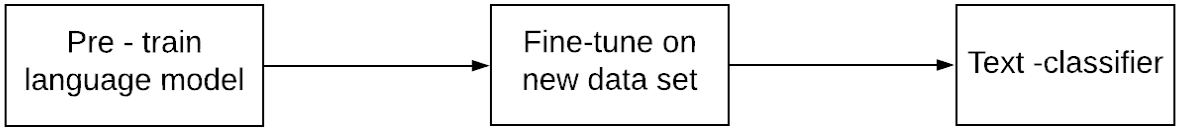
Indhuja K, P. C. Reghu Raj(2014) the study investigated the performance of statistical measures to determine the text-based language identification system, with an emphasis on five languages used in India based on Devanagari script - Hindi, Sanskrit, Marathi, Nepali and Bhojpuri. The

proposed system uses n-grams as a feature for classification. Language Identification is an important preprocessing step in many tasks of Natural Language Processing (NLP). The study showed the accuracy around 90 percent in two pair LID. It is observed that the differences between the languages within a language family and across language family cases are not very drastic. For example Hindi is actually inherited from Sanskrit, similarly Marathi, Bhojpuri are from Hindi. They show a narrow gap in language identification. Then they observed that average accuracy decreases from 87.2 to 83.5 when going from two pair LDI to three pair LDI . The average accuracy in 5 pairs is 80 percent much less than other pairs.

Proposed methodology:

Our Language Model is based on ULMfit Model and follows three step architecture:

1. **LM Pre-training:**The Language Model is trained on a general-domain corpus to capture general features of the language in different layers
2. **LM Fine-tuning:**We fine-tune the model on the target task dataset to learn its distributions
3. **Target Task Classifier:** The pretrained Language Model is expanded by two linear blocks so that the final output is a probability distribution over the sentiment labels.

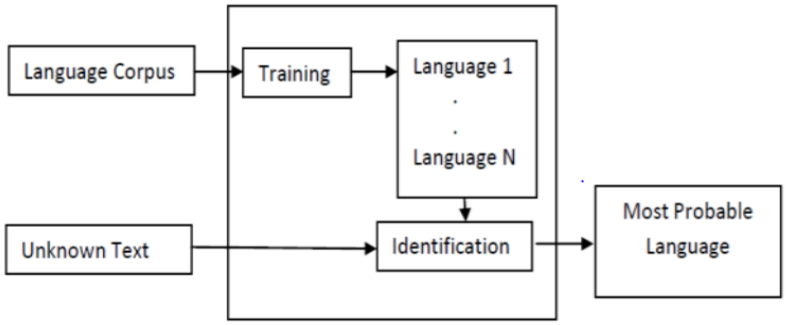


- **Workflow**

  Following are the three main steps are done namely:

  - **Training**:In this step we train our model with the language corpus of the above mentioned languages.
  - **Preprocessing**:After training all the preprocessing is done & necessary dependencies are installed to run the model.

- **Testing**:We test our model by giving an unknown text in any language(i.e monolingual/multilingual text document) to the model after which our model correctly identifies the language of the given text document.



❖ **Technology Used:**

Language: Python 3.6

Python IDE: Google colab

Libraries used: iNLTK library which is one of the most important library used for identifying the Indian languages as it supports 12 languages of Indian origin.It is a deep learning open source library.

Dataset: Wikipedia Articles, BBC News Articles, Movies Review, Shlokas.

Result and Analysis:

- Built a language identification model which can successfully identify monolingual & multilingual texts.
- Our language model has performed word based language identification from a given piece of text.
- Our model has implemented both language identification from a given sentence and as well as identifying the language from any text document.

Output Snapshots:

- Setup the language models

```
[ ]  from inltk.inltk import setup
     from inltk.inltk import identify_language
     from inltk.inltk import identify_language, reset_language_identifying_models
     from inltk.inltk import predict_next_words
     from inltk.inltk import remove_foreign_languages
     from inltk.inltk import get_sentence_encoding
     from inltk.inltk import get_similar_sentences
     from inltk.inltk import get_sentence_similarity
     setup('hi')
     setup('bn')
     setup('pa')
     setup('sa')
```

```
Done!
Done!
Done!
Done!
```

- Monolingual Language Identification
  - ❖ Reading from the text document

```
f = open('./text.txt', 'r')
file_contents = f.read()
print (file_contents)
```

```
कालिंजर दुर्ग, भारतीय राज्य उत्तर प्रदेश के बांदा जिले में स्थित एक दुर्ग है। बुन्देलखण्ड क्षेत्र में विंध्य पर्वत पर स्थित यह दुर्ग विश्व धरोहर स्थल खजुराहो से ९७.७(17.7)
प्राचीन काल में यह दुर्ग जेजाकभुक्ति (जयशक्ति चन्देल) साम्राज्य के अधीन था। बाद में यह १०(10)वीं शताब्दी तक चन्देल राजपूतों के अधीन और फिर रीवा के सोलंकियों
```

```
f = open('./tamil.txt', 'r')
file_contents1 = f.read()
print (file_contents1)
```

```
தலைவலி  என்றாலே உடனே மாத்திரை போடும் பழக்கம் பலரிடம் உள்ளது. அதுவும் பலரிடமும் கைவசம் இ(


40 வயதைத் தாண்டினால், நரம்புத் தளர்ச்சியில் கொண்டு போய் விட்டு விடும் என்பது பலருக்கு தெரிவதில்ல
```

  - ❖ Identification of language from the given text document

```
identify_language(file_contents)
```

```
'hindi'
```

```
identify_language(file_contents1)
```

```
'tamil'
```

❖ Sentence based language identification

```
identify_language("আবহাওয়া চমৎকার")
```

```
'bengali'
```

```
identify_language("ਤੁਸੀਂ ਕਿਵੇਂ ਹੋ")
```

```
'panjabi'
```

```
identify_language("भवान् संस्कृतं भाषते वा ")
```

```
'sanskrit'
```

- Multilingual Language Identification

```
import polyglot
from polyglot.utils import pretty_list
import pycld2 as cld2

mixed_text=("எப்படி இருக்கிறீர்கள்  আপনি কেমন আছেন আপ কैसे हैं")



isReliable, textBytesFound, details, vectors = cld2.detect(
    mixed_text, returnVectors=True
)
print(details)
```

```
(('TAMIL', 'ta', 47, 1024.0), ('BENGALI', 'bn', 29, 603.0), ('HINDI', 'hi', 22, 1638.0))
```

Concluding remarks (including shortcomings)

- Our model takes comparatively more time for identification of language from large dataset.
- Accuracy of Hindi language identification is less comparable in comparison to other languages.

References:

1. Akosu, Nicholas & Selamat, Ali. (2015). Word-length algorithm for language identification of under-resourced languages. Journal of King Saud University - Computer and Information Sciences. 28. 10.1016/j.jksuci.2014.12.004.
2. Indhuja K, Indu M, Sreejith C, P. C. Reghu Raj, 2014, Text Based Language Identification System for Indian Languages Following Devanagiri Script, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 03, Issue 04 (April 2014)

Annexure:

Language Identification Model has been implemented in google colab platform written in python language.

Source code file is located at:

https://colab.research.google.com/drive/1NrilCHrTe6R61Vzg9QJTTNSHdE50iwnY

Code:

```
#Install libraries and Dependencies
pip install inltk
pip install torch==1.3.1+cpu -
 https://download.pytorch.org/whl/torch_stable.html
pip install polyglot
pip install pycld2
pip install PyICU
from fastai.text import *
import sentencepiece as spm
from pathlib import Path
import aiohttp as aiohttp
import asyncio
from fastai.text import *
from inltk.tokenizer import LanguageTokenizer
```

```
from inltk.inltk import setup
from inltk.inltk import identify_language
from inltk.inltk import identify_language
from inltk.inltk import predict_next_words
from inltk.inltk import remove_foreign_languages
from inltk.inltk import get_sentence_encoding
from inltk.inltk import get_similar_sentences
from inltk.inltk import get_sentence_similarity
import os


#Training with Datasets
class LanguageCodes:
    bengali = 'bn'
    hindi = 'hi'
    panjabi = 'pa'
    sanskrit = 'sa'
    tamil = 'ta'

    def get_all_language_codes(self):
        return [self.bengali,  self.hindi,self.panjabi,
                self.sanskrit, self.tamil]


class LMConfigs:
    all_language_codes = LanguageCodes()
    lm_model_file_url = {
        all_language_codes.bengali: 'https://www.dropbox.com/s/4berhstpw83
6kcw/export.pkl?raw=1',
        all_language_codes.hindi: 'https://www.dropbox.com/s/sakocwz413eyz
t6/export.pkl?raw=1',
        all_language_codes.panjabi: 'https://www.dropbox.com/s/ejiv5pdsi2m
hhxa/export.pkl?raw=1',
        all_language_codes.sanskrit: 'https://www.dropbox.com/s/4ay1by5ryz
6k39l/sanskrit_export.pkl?raw=1',
        all_language_codes.tamil: 'https://www.dropbox.com/s/88klv70zl82u3
9b/export.pkl?raw=1',
    }
    tokenizer_model_file_url = {
        all_language_codes.bengali: 'https://www.dropbox.com/s/29h7vqme1kb
8pmw/bengali_lm.model?raw=1',

        all_language_codes.hindi: 'https://www.dropbox.com/s/xrsjt8zbhwo7z
xq/hindi_lm.model?raw=1',
```

```
        all_language_codes.panjabi: 'https://www.dropbox.com/s/jxwr9ytn0zf
zulc/panjabi_lm.model?raw=1',
        all_language_codes.sanskrit: 'https://www.dropbox.com/s/e13401nsek
ulq17/tokenizer.model?raw=1',
        all_language_codes.tamil: 'https://www.dropbox.com/s/jpg4kaqyfb71g
1v/tokenizer.model?raw=1',

    }

    def __init__(self, language_code: str):
        self.language_code = language_code

    def get_config(self):
        return {
            'lm_model_url': self.lm_model_file_url[self.language_code],
            'lm_model_file_name': 'export.pkl',
            'tokenizer_model_url': self.tokenizer_model_file_url[self.lang
uage_code],
            'tokenizer_model_file_name': 'tokenizer.model'
        }


class AllLanguageConfig(object):

    @staticmethod
    def get_config():
        return {
            'all_languages_identifying_model_name': 'export.pkl',
            'all_languages_identifying_model_url': 'https://www.dropbox.co
m/s/a06fa0zlr7bfif0/export.pkl?raw=1',
            'all_languages_identifying_tokenizer_name': 'tokenizer.model',
            'all_languages_identifying_tokenizer_url':
                'https://www.dropbox.com/s/t4mypdd8aproj88/all_language.mo
del?raw=1'
        }

#Tokenization
class LanguageTokenizer(BaseTokenizer):
    def __init__(self, lang: str):
        self.lang = lang
        self.sp = spm.SentencePieceProcessor()
        model_path = path/f'models/{lang}/tokenizer.model'
        self.sp.Load(str(model_path))

    def tokenizer(self, t: str) -> List[str]:
```

```
        return self.sp.EncodeAsPieces(t)

    def remove_foreign_tokens(self, t: str):
        local_pieces = []
        for i in self.sp.EncodeAsIds(t):
            local_pieces.append(self.sp.IdToPiece(i))
        return local_pieces


class SanskritTokenizer(LanguageTokenizer):
    def __init__(self, lang: str):
        LanguageTokenizer.__init__(self, lang)


class BengaliTokenizer(LanguageTokenizer):
    def __init__(self, lang: str):
        LanguageTokenizer.__init__(self, lang)


class HindiTokenizer(LanguageTokenizer):
    def __init__(self, lang: str):
        LanguageTokenizer.__init__(self, lang)


class PanjabiTokenizer(LanguageTokenizer):
    def __init__(self, lang: str):
        LanguageTokenizer.__init__(self, lang)


class TamilTokenizer(LanguageTokenizer):
    def __init__(self, lang: str):
        LanguageTokenizer.__init__(self, lang)


#Language Setup method
all_language_codes = LanguageCodes()

async def setup_language(language_code: str):
    lmconfig = LMConfigs(language_code)
    config = lmconfig.get_config()
    await download_file(config['lm_model_url'], path/'models'/f'{language_
code}', config["lm_model_file_name"])
    await download_file(config['tokenizer_model_url'], path/'models'/f'{la
nguage_code}',
```

```
                        config["tokenizer_model_file_name"])
    print('Done!')
    return True


def verify_language(language_code: str):
    lmconfig = LMConfigs(language_code)
    config = lmconfig.get_config()
    if (path/'models'/f'{language_code}'/f'{config["lm_model_file_name"]}'
).exists() and \
            (path/'models'/f'{language_code}'/f'{config["tokenizer_model_f
ile_name"]}').exists():
        return True
    else:
        return False
lcodes = LanguageCodes()
all_language_codes = lcodes.get_all_language_codes()


async def download(language_code: str):
    if language_code not in all_language_codes:
        raise Exception(f'Language code should be one of {all_language_cod
es} and not {language_code}')
    learn = await setup_language(language_code)
    return learn


def setup(language_code: str):
    asyncio.set_event_loop(asyncio.new_event_loop())
    loop = asyncio.get_event_loop()
    tasks = [asyncio.ensure_future(download(language_code))]
    learn = loop.run_until_complete(asyncio.gather(*tasks))[0]
    loop.close()


def check_input_language(language_code: str):
    if language_code not in all_language_codes:
        raise Exception(f'Language code should be one of {all_language_cod
es} and not {language_code}')
    if not verify_language(language_code):
        raise Exception(f'You need to do setup for the **first time** for
language of your choice so that '
                        f'we can download models. So, '
                        f'Please run setup({language_code}) first!')
```

```
def tokenize(input: str, language_code: str):
    check_input_language(language_code)
    tok = LanguageTokenizer(language_code)
    output = tok.tokenizer(input)
    return output

#Language Identification Method

def identify_language(input: str):
    asyncio.set_event_loop(asyncio.new_event_loop())
    loop = asyncio.get_event_loop()
    tasks = [asyncio.ensure_future(check_all_languages_identifying_model()
)]
    done = loop.run_until_complete(asyncio.gather(*tasks))[0]
    loop.close()
    defaults.device = torch.device('cpu')
    path = Path(__file__).parent
    learn = load_learner(path / 'models' / 'all')
    output = learn.predict(input)
    return str(output[0],output[1])


def remove_foreign_languages(input: str, host_language_code: str):
    check_input_language(host_language_code)
    tok = LanguageTokenizer(host_language_code)
    output = tok.remove_foreign_tokens(input)
    return output

#Setup All five Language Models
setup('hi')
setup('bn')
setup('pa')
setup('sa')
setup('ta')

#Read the contents of text file

f = open('./text.txt', 'r')

file_contents = f.read()

print (file_contents)


f = open('./tamil.txt', 'r')

file_contents1 = f.read()
```

```
print (file_contents1)

#Monolingual Language Identification(text Document)
identify_language(file_contents)
identify_language(file_contents1)


#Monolingual Language Identification(Given sentence)
identify_language("আবহাওয়া চমৎকার")
identify_language("तुमी किवें हो")
identify_language("भवान् संस्कृतं भाषते वा ")
#Multilingual language Identification
mixed_text=("எப்படி இருக்கிறீர்கள்  আপনি কেমন আছেন आप कैसे हैं")

isReliable, textBytesFound, details, vectors = cld2.detect(
    mixed_text, returnVectors=True
)
print(details)
```