Efficient Auto Language Detection

Overview (ChatGPT, Google Gemini, Google search)

In the NLP domain, several mainstream language detection methods are commonly used:

1. **N-gram Models**: analysing sequences of n characters or words. By creating frequency distributions of these n-grams, models can identify the language based on the likelihood of encountering specific n-grams in different languages.
2. **Character-Based Models**: These models focus on the distribution of characters in the text.
3. **Machine Learning Classifiers**: Algorithms such as Support Vector Machines (SVM), Random Forests, or deep learning models can be trained on features derived from the text (like n-grams, word embeddings) to classify languages.
4. **Deep Learning Models**: Recurrent Neural Networks (RNNs) or Transformers (like BERT) through extra fine-tuning.
5. **Pre-trained Language Models**: Models like FastText or language identification systems built on BERT and other transformer architectures, through fine-tune.
6. **Hybrid Approach**.

https://translatepress.com/docs/addons/automatic-user-language-detection/

https://engati.com/glossary/language-detection#:~:text=different%20approaches%20are:-,1.,Frequent%20Words%20Method

Overview: online APIs

Examples: AWS, Google Cloud, Microsoft Azure, IBM

Others:

- Intellexer, more industry-solution focused https://esapi.intellexer.com/
- EdenAI: https://www.edenai.co/feature/language-detection-apis
- Detect Language API: https://detectlanguage.com/

Could be expensive!!!

https://www.edenai.co/post/top-10-language-detection-apis

Open-sourced local solutions

Spacy FastLang: https://spacy.io/universe/project/spacy_fastlang

Fast language detection using FastText and Spacy.

EXAMPLE

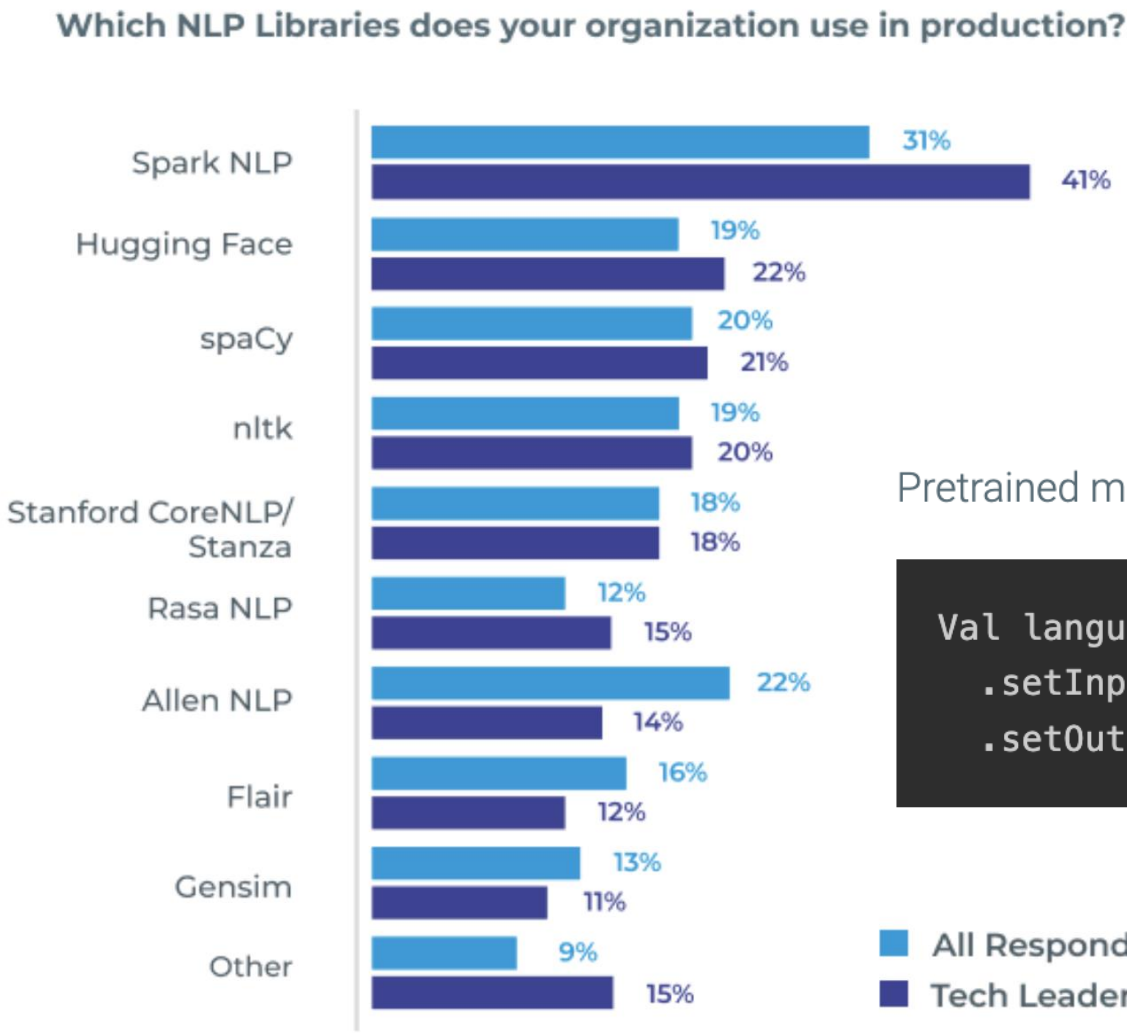
```
import spacy_fastlang

nlp = spacy.load("en_core_web_sm")
nlp.add_pipe("language_detector")
doc = nlp('Life is like a box of chocolates. You never kno

assert doc._.language == 'en'
assert doc._.language_score >= 0.8
```

Open-sourced local solutions

Spark NLP: https://sparknlp.org/, [code example]



Pretrained models can be loaded with `pretrained` of the companion object:

```
Val languageDetector = LanguageDetectorDL.pretrained()
  .setInputCols("sentence")
  .setOutputCol("language")
```

Open-sourced local solutions

langdetect: https://pypi.org/project/langdetect/

Basic usage

To detect the language of the text:

```
>>> from langdetect import detect
>>> detect("War doesn't show who's right, just who's left.")
'en'
>>> detect("Ein, zwei, drei, vier")
'de'
```

`langdetect` supports 55 languages out of the box (ISO 639-1 codes):

```
af, ar, bg, bn, ca, cs, cy, da, de, el, en, es, et, fa, fi, fr, gu, he,
hi, hr, hu, id, it, ja, kn, ko, lt, lv, mk, ml, mr, ne, nl, no, pa, pl,
pt, ro, ru, sk, sl, so, sq, sv, sw, ta, te, th, tl, tr, uk, ur, vi, zh-cn, zh-tw
```

Conclusion

- This is a well-studied domain, so the open-sourced approaches would be sufficient
- **FastLang** and **LangDetect** both looks well!

The key part is the running time estimation, I believe either should be the very fast.

I would recommend we have a small test dataset to test out the accuracy of the approaches and mention that in the manuscript.

Fast language detection using FastText and Spacy.

EXAMPLE

```
import spacy_fastlang

nlp = spacy.load("en_core_web_sm")
nlp.add_pipe("language_detector")
doc = nlp('Life is like a box of chocolates. You never kno

assert doc._.language == 'en'
assert doc._.language_score >= 0.8
```

To detect the language of the text:

```
>>> from langdetect import detect
>>> detect("War doesn't show who's right, just who's left.")
'en'
>>> detect("Ein, zwei, drei, vier")
'de'
```