

Protecting euRopean public spaces against Emergent hoStile drone thrEats thRough an adVanced multidimensional shield and cross-border intelligEnce





Multilingual **T**ext **A**nalysis

A Language detection testing tool

by LogObject

Draft 15.04.2026

Table of Contents

1. Motivation 2
2. Document links 3
3. About Fasttext 3
4. About MarianNMT 5
a. Local language pairs for offline execution 5

1. Motivation

To implement the Multilingual Text Analysis Component as a part of the PRESERVE Project LogObject (LOD) evaluates some opensource tools for language detection, preformatting and translation of a text based document.

FastText

The selected tool for the language detection system is **Fasttext** which was developed by Facebook. The job of such a language detection system is to detect the language of a given text document. The confidence value of the detected language will be provided additionally.

If a text contains more than one language a list of languages and their related confidence shall be provided. The language detection module is executed locally without using the internet.

MarianNMT

The tool for the translation of text from a given language to a destination language is MarianNMT. The supported language pairs for the translation are loaded locally without using the internet.

2. Document links

Topic	Link
Fasttext	**https**://fasttext.cc/
ISO 639 Language codes	https://www.iso.org/iso-639-language-code
MarianNMT	https://marian-nmt.github.io/
Blindgenerator	Generator for texts for testing purpose https://www.blindtextgenerator.de/
Language models	https://huggingface.co/Helsinki-NLP/opus-mt-ro-en test a pair to downlad e.g ro-en https://huggingface.co/models?search=opus-mt-ro

3. About Fasttext

FastText, a tool developed by Facebook AI Research for text classification and word vector generation, is highly efficient and capable of processing large amounts of text.

The following country codes are supported by FastText.

af als am an ar arz as ast av az azb ba bar bcl be bg bh bn bo bpy br bs bxr ca cbk ce ceb ckb co cs cv cy da de diq dsb dty dv el eml en eo es et eu fa fi fr frr fy ga gd gl gn gom gu gv he hi hif hr hsb ht hu hy ia id ie ilo io is it ja jbo jv ka kk km kn ko krc ku kv kw ky la lb lez li lmo lo lrc lt lv mai mg mhr min mk ml mn mr mrj ms mt mwl my myv mzn nah nap nds ne new nl nn no oc or os pa pam pfl pl pms pnb ps pt qu rm ro ru rue sa sah sc scn sco sd sh si sk sl so sq sr su sv sw ta te tg th tk tl tr tt tyv ug uk ur uz vec vep vi vls vo wa war wuu xal xmf yi yo yue zh

The result of the language detection (Button: Detect Languages) is a list of country codes, an confidence value for each code and english language names concerning ISO 639; https://www.iso.org/iso-639-language-code

However, there are technical and practical limits depending on **how FastText is used** (e.g., training vs. inference) and **the hardware** being used.

1. Text length per individual example (line)

- **Recommended:** Short to medium-length texts (e.g., sentences or paragraphs).
- **Technical limit:** FastText processes text line by line. A "line" typically represents a **document or sentence**.
    - There is **no hard character limit**, but very long texts (several thousand tokens) may cause memory or performance issues.
    - It is **recommended to use texts with 100–1,000 words** per line.
    - If you're dealing with very long texts (e.g., entire chapters or books), it's best to split them into smaller chunks.

2. Total size of the training dataset

- FastText is designed for large-scale data.
- **Millions of text lines** (examples) are possible.
- Memory usage depends on:
    - Vocabulary size
    - Number of classes
    - Model parameters like dim, epoch, lr, etc.
- You can easily process **gigabytes of text**, assuming your machine has enough RAM.

3. Model memory size

- FastText models are compact:
    - A standard model is usually **under 1 GB**.
    - With binary compression (-compress), even smaller.
- With higher dim values and very large vocabularies, the model can also grow to several GB.

Summary

Type of text size	Recommended / Possible
Length of a single text	Up to several thousand tokens, ideally <1,000 words

Type of text size	Recommended / Possible
Training corpus size	Millions of lines / several GB of text
Model size (memory)	Usually <1 GB, larger models (several GB) possible

4. About MarianNMT

a. Local language pairs for offline execution

Planned

#original

"ar-en", "ca-en", "de-en", "el-en", "es-en", "fr-en", "hi-en", "it-en", "ro-en", "ru-en", "tr-en", "zh-en",

"ar-de", "ca-de", "el-de", "en-de", "es-de", "fr-de", "hi-de", "it-de", "ro-de", "ru-de", "tr-de", "zh-de",

"ar-el", "ca-el", "de-el", "en-el", "es-el", "fr-el", "hi-el", "it-el", "ro-el", "ru-el", "tr-el", "zh-el",

"ar-ca", "de-ca", "el-ca", "en-ca", "es-ca", "fr-ca", "hi-ca", "it-ca", "ro-ca", "ru-ca", "tr-ca", "zh-ca",

"ar-es", "ca-es", "de-es", "el-es", "en-es", "fr-es", "hi-es", "it-es", "ro-es", "ru-es", "tr-es", "zh-es"

Successfully loaded

ar-de ar-es ca-es de-en en-de es-ca es-en fr-el hi-en it-en ru-es zh-de

ar-el ca-de de-ca de-es en-el es-de fr-ca fr-en it-ca it-es tr-en zh-en

ar-en ca-en de-el en-ca en-es es-el fr-de fr-es it-de ru-en tr-es

Currently loaded language pairs

Sorted by source language	Sorted by destination language
Arabisch (ar): ar-de, ar-es, ar-el, ar-en **Katalanisch (ca):** ca-es, ca-de, ca-en **Deutsch (de):** de-en, de-ca, de-es, de-el **Englisch (en):** en-de, en-el, en-ca, en-es **Spanisch (es):** es-ca, es-en, es-de, es-el **Französisch (fr):** fr-el, fr-ca, fr-en, fr-de, fr-es **Hindi (hi):** hi-en **Italienisch (it):** it-en, it-ca, it-es, it-de **Russisch (ru):** ru-es, ru-en **Türkisch (tr):** tr-en, tr-es **Chinesisch (zh):** zh-de, zh-en	**Deutsch (de):** ar-de, ca-de, es-de, fr-de, it-de, zh-de **Englisch (en):** ar-en, ca-en, es-en, fr-en, hi-en, it-en, ru-en, tr-en, zh-en **Spanisch (es):** ar-es, de-es, fr-es, it-es, ru-es, tr-es **Griechisch (el):** ar-el, de-el, en-el, es-el, fr-el **Katalanisch (ca):** de-ca, en-ca, fr-ca, it-ca

Not successfully loaded

ca: ar-ca, el-ca, hi-ca, ro-ca, ru-ca, tr-ca, zh-ca
de: el-de, hi-de, ro-de, ru-de, tr-de
el: ca-el, hi-el, it-el, ro-el, ru-el, tr-el, zh-el
en: el-en, ro-en
es: el-es, hi-es, ro-es, zh-es

Marian is an efficient, free Neural Machine Translation framework written in pure C++ with minimal dependencies. It is mainly being developed by the Microsoft Translator team.

Many academic (most notably the University of Edinburgh and in the past the Adam Mickiewicz University in Poznań) and commercial contributors help with its development.

It is currently the engine behind the Microsoft Translator Neural Machine Translation services and being deployed by many companies, organizations and research projects (see below for an incomplete list).

Translation Models which are developed with MarainNMT are free and usable under MIT Licence:

https://huggingface.co/Helsinki-NLP/opus-mt-de-en

Licence conditions: Include the copyright notice and the license in your product. Your imprint, license dialog, documentation or README should state: This product includes translation models from Helsinki-NLP (https://huggingface.co/Helsinki-NLP), licensed under the MIT License. Optional: You can also include the full license text of the MIT license.   Falls du das Modell in einem größeren Softwareprodukt, OEM-System oder behördlich regulierten Kontext einsetzen willst, ist es sinnvoll, mit der Rechtsabteilung Rücksprache zu halten – auch wenn die MIT-Lizenz üblicherweise keine rechtlichen Probleme verursacht.

Confidence Value based on the log-probability:

Der Confidence-Wert spiegelt die **durchschnittliche Wahrscheinlichkeit der vom Modell generierten Tokens** wider.
Dabei gilt:

- Jedes Token in der Ausgabe hat eine **Log-Wahrscheinlichkeit** (log prob), die angibt, wie wahrscheinlich es vom Modell an dieser Stelle erzeugt wurde.
- Die Confidence eines Satzes wird als **arithmetisches Mittel der Log-Wahrscheinlichkeiten der gewählten Tokens** berechnet.
- Diese durchschnittliche Log-Wahrscheinlichkeit wird anschließend mit exp() in eine **normale Wahrscheinlichkeit** umgewandelt (Wertebereich: 0–1).

📌 **Formell:**

Wenn y1,y2,...,yny_1, y_2, \dots, y_ny1,y2,...,yn die generierten Token sind und p(yi)p(y_i)p(yi) deren Wahrscheinlichkeiten:

confidence=exp⁡(1n∑i=1nlog⁡(p(yi)))\text{confidence} = \exp\left(\frac{1}{n} \sum_{i=1}^{n} \log(p(y_i))\right)confidence=exp(n1i=1∑nlog(p(yi)))

Das ist gleichbedeutend mit dem **geometrischen Mittel** der Wahrscheinlichkeiten.

---------- Ähnlcihkeits metriken

Welche Ähnlichkeitsmetriken können noch verwendet werden?

ChatGPT:

Sehr gute Frage! Wenn du das Flagging robuster machen willst – also erkennen möchtest, **ob die Übersetzung inhaltlich zu ähnlich oder identisch zum Quelltext ist**, obwohl ein hoher Confidence-Wert vorliegt – kannst du folgende **Ähnlichkeitsmetriken** in Betracht ziehen:

🔍 1. String-basierte Metriken

Metrik	Beschreibung	Bibliothek
Levenshtein-Distanz	Misst die minimale Anzahl an Einfüge-, Lösch- und Ersetzoperationen	python-Levenshtein
Jaccard-Ähnlichkeit	Verhältnis der gemeinsamen Tokens zu allen eindeutigen Tokens	sklearn, manuell
Cosine Similarity (TF-IDF)	Vergleicht Texte als Vektoren im TF-IDF-Raum	sklearn
Fuzzy Matching (ratio)	Ähnlichkeitswert auf Basis von Levenshtein – gut für kleine Texte	fuzzywuzzy (alt)

🔬 2. Semantische Metriken (Vektorraum-basiert)

Metrik	Beschreibung
Sentence-BERT Cosine Similarity	Misst Ähnlichkeit semantisch – ideal für echte Bedeutungserkennung

Metrik	Beschreibung
spaCy Similarity	Nutzt Wortvektoren aus geladenem spaCy-Modell
Universal Sentence Encoder	Vektorvergleich mit Google's semantischen Sätzen

---

✅ **Vorschlag: Kombination zweier Ansätze**

Beispiel: **Flagging**, wenn einer dieser Fälle zutrifft:

- Levenshtein-Ratio > 0.95
- UND Confidence > 0.8
- UND übersetzter Text ≈ Originaltext