fEMR Translation Tech Spec

David Jumper, Harrison Shu, Michael Hayes, Noah Tobinsky, Sean Hershey

March 21, 2024

platinum.cscaws.com:8443/projects/TEMR/summary
github.com/FEMR/femr/tree/temr-dev

Table of Contents

Introduction

Overview

tEMR is working on a dynamic translation service for physicians and healthcare coders who may write and read in different languages but must be on the same page. Our dynamic translation involves neural networks like Argos (OpenNMT) and MarianMT working in the background in Python.

Glossary

I18N / Internationalization - The process of designing a software application so that it can be adapted to various languages and regions without engineering changes

ICD-10 - A medical classification system designed to catalog health conditions by categories of similar scenarios, mapping complex scenarios to broader morbidities.

Neural Machine Translation - An artificial neural network used to predict the likelihood of a sequence of words, typically modeling and then translating entire sentences in a single integrated model.

ISO-639 - Standardized way to classify languages. Every language is mapped to and referenced with a two-letter code.

Context

The fEMR application captures text that is entered into the input fields and only displays it in the original imputed language. This causes many difficulties in effective ICD-10 encoding, and continuity of care for the patient when used by physicians. By knowing the language preference of the enterer and the accessor automatic dynamic translation can improve the native readability of medical records.

Goals

User Stories

- As a medical provider user, I want to be able to view the translated chief complaints text in the language of my user profile, so that I can understand chief complaints notes outside my language.
- As a medical provider user, I want to see the original untranslated chief complaint text.

- As a developer, I want to easily be able to test the performance of existing and new offline translation models.

Technical Requirements

- The system shall have average first-time translation times of 3 seconds
- The system shall have average subsequent translation times of 1 second
- The system shall be compatible with all major operating systems
- The system shall not store the dynamically translated text outside of the user's current session

Out of Scope

- Training our offline translation models might not be feasible, as we are limited to the language pairs that are supported by the translation library we use, and we would need to compare results for each language pair with Google translate (which isn't accurate 100% of the time either) or find another language reference.

Future Goals

Fall Quarter

- Construct an automatic translation testing program that outputs an spreadsheet report of different translation options and the translation runtimes for each option
- Utilize/ aggregate the translation testing data to determine the appropriate translation model to use for offline translation
- Have a prototype of translation functionality on the Medical page for Spanish-to-English translations

Winter Quarter

- Fully integrate dynamic offline translation throughout the entire fEMR kit and have patient data translated in real-time when the fEMR patient data is loaded
- Use Google Translate or a more accurate translation model for data in fEMR central

Spring Quarter

- Translation server startup before first translation reduces first time translation speed to an average of 1 second.
- Have project code from both fEMR teams merged and maintained, and have the merged code be stable.
- Investigate installation process and resource use on sample field laptop.

Assumptions

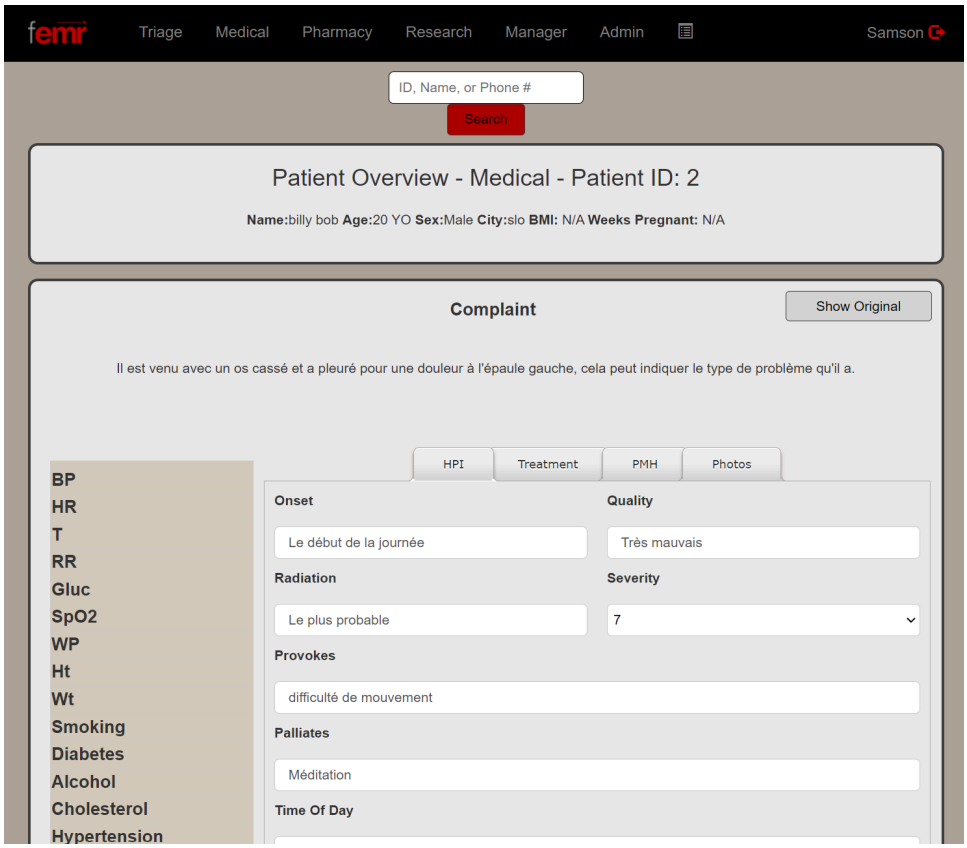
- While online time may exist to update packages the admin should not be burdened with constant installations.
- Patients are potentially going to be using fake names and personal information, so continuity might not always be feasible.

Solutions

Proposed Design

For our design, we will auto-translate complaints in the medical portal using triage user's entered language and the viewing user's language preference. This will be made possible by intercepting the complaint and routing it through a process builder supported python file that contains a neural network machine translation model.

Presentation Layer



Featured Technologies

- Play Framework
- MySQL
- ProcessBuilder (java.io)
- Argos Translate
- MarianMT

Alternate or Researched Technologies

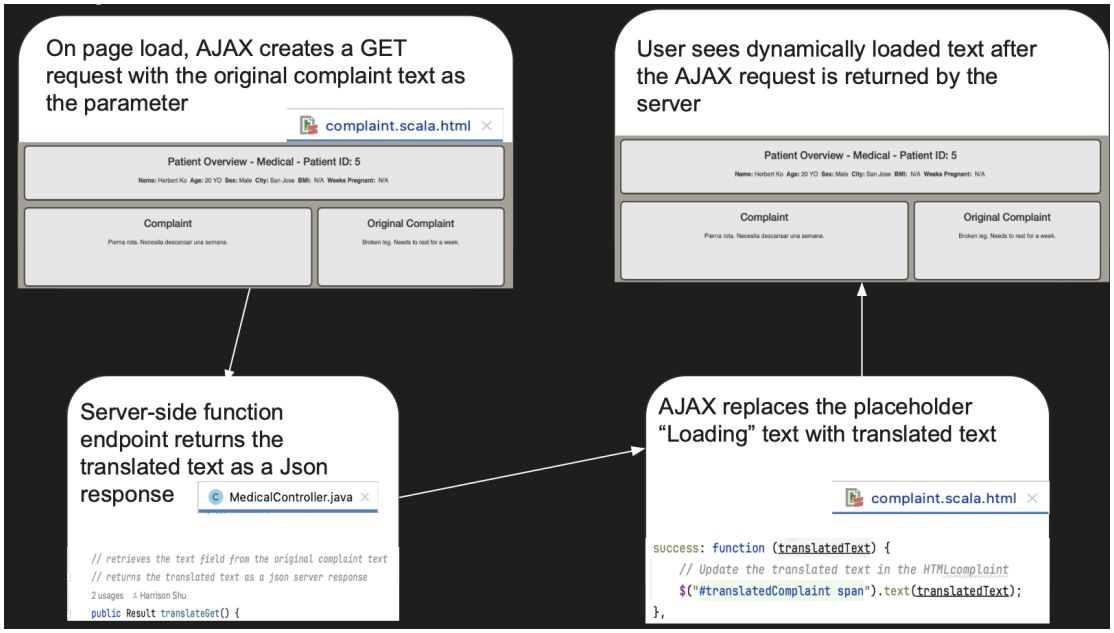
- Google Translate
- Lingua

Translation Libraries

Library	Pros	Cons
Argos Translate	Offline, Fast, and Lightweight	Limited Languages (42), Python, Only translates to and from English
Marian	Offline, Accuracy, Languages (192+)	Cons - Larger Model, Slower with some languages.
Google Translate	Accuracy, Languages (133)	Online
Lingua	Language Detection	Not Translation

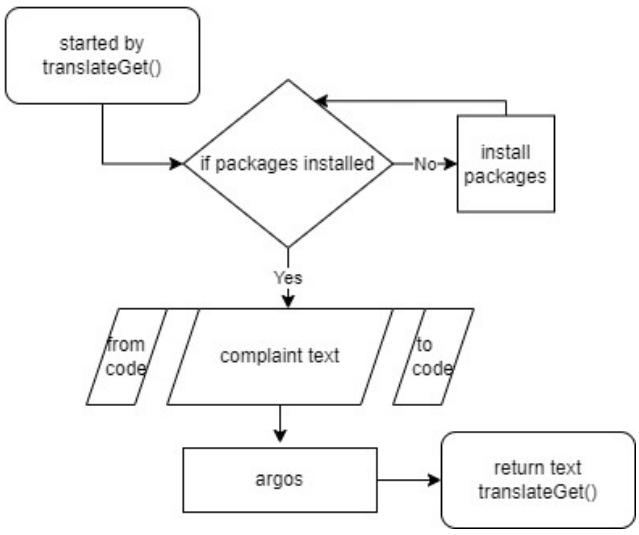
Structure Diagrams

Background Translation / Dynamically Loaded Text



On page load AJAX requests original complaint, then the server's function returns translated text, which is then used to replace the placeholder buffer animation.

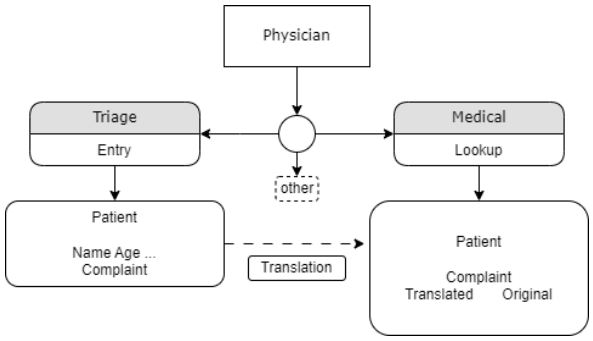
Translate Python Flowchart



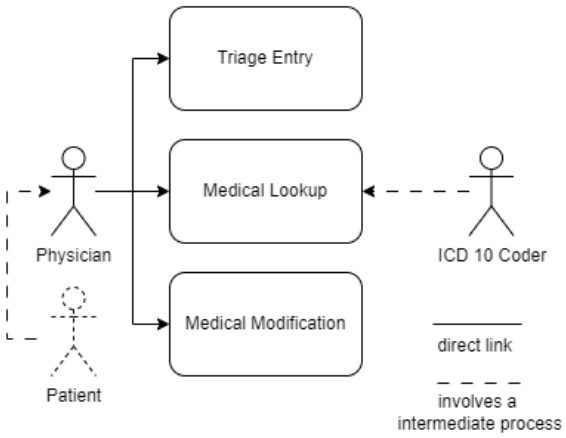
The python script checks if all necessary models are downloaded and configured (which will only be the case the first time the script is run on a new machine), then translates and returns with the Argos library.

Behavior Diagrams

Activity



Use Case



The physician is the only user who can change or enter complaints and their activity also includes looking up users for continuity of care. Patients interact with physicians in order to use fEMR and ICD-10 Coders will gain access to the medical lookup information through aggregate data compilation not fEMR UI itself. The intermediate process between triage and medical portals is accessing the database.

Database Model

- Blue Tables: Relational tables with their schema remained unchanged.
- Green Tables: Relational tables with changes to their schema. So far, these changes only include the addition of language codes.

Tables of interest center around cheif_complaints, where the complaint/s are stored, patient encounters where cheif_complaint objects link, and users not shown but important for future implementation of the user-defined language model.

Test Plan

For the backend, unit tests will be conducted on all pieces of code, ensuring a test coverage of over 90%. Play Framework has a tests location folder where we can add more tests as we implement features and units.

Further Considerations

Security Considerations

- Consider the privacy of the data being translated in cloud services.
- For offline translation be careful how complaints are cached and when they should be scrapped
- Work in the bounds of lowest common privilege and implement security at least as good as in place measures to the fEMR web application.

Privacy Considerations

- How does the solution protect user's data privacy?
    - Since our translation is using an offline translation model, the data remains locally stored. We do not store translated data outside of the user's current session

Regional Considerations

- We have the most widely spoken languages worldwide supported within the offline translation model. However, not all languages are supported and there is a small possibility to have issues if someone in a non-supported language uses fEMR. The next best language should be used in its place while it is requested for a language kit and added to the model.

Accessibility Considerations

- How accessible is the solution?
    - The goal is for our translation to be easily readable by any user who has set up their profile for a language they know. In cases where our translation fails, we are implementing a "show original" button that allows users to view the original, non-translated data. This should make our translation as accessible as it can be while functioning offline.
- What tools will you use to evaluate its accessibility?
    - We will use user testing to evaluate our tools' accessibility

Operational Considerations

- Does this solution cause adverse aftereffects?
    - Our translation will add some extra time when loading pages. We will limit this effect by using a fast translation model and possibly loading the original data while the translated data is still loading.
- How will the solution recover in case of failure?
    -
- How will operational costs be kept low while delivering increased value to users?
    - The translation will occur using an offline language model and downloaded language packs which allows for unlimited translations without cost

Risks

- What risks are being undertaken with this solution?
    - Our offline translation model when used on the stored data (i.e. medical notes) can sometimes produce incorrect translations that are more confusing than the original text. Our “show original” button should mitigate this risk.
- What is the cost-benefit analysis of taking these risks?
    - Since our translation has to be done offline, we don’t have any other options, so this risk is necessary for our solution.

Deliberation

Discussion

- The decision of which offline language model would be most suitable for the software is up for debate. Limitations such as fEMR kit speed, fEMR kit storage, and time availability of Providers should be taken into account for the choice of language model.

Open Questions

- How can we accurately user-test with real world bilingual or non-english languages?
- How should we balance the need for translation speed, accuracy, and language availability?

References and Acknowledgements

- References
    - https://www.playframework.com/documentation/3.0.x/Home
    - https://github.com/argosopentech/argos-translate
    - https://huggingface.co/docs/transformers/model_doc/marian
- Acknowledgments
    - Professor BJ Klinenberg
    - Andi Mastie
    - Sarah Draugelis

Thank you to the above people for the support and making our progress possible