A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green. They are positioned diagonally, with the blue one partially covering the green one.

# Using Transformers for Japanese-English Translation

Felice Forby

CSPB 4830 Natural Language Processing



# Problem Overview

- Machine translation for the Japanese-English language pair tends to be difficult, but newer transformer models have shown much better performance.
- Lack of models that are trained to handle specialized topics like patent and medical translation.
- Professional translation is expensive and time-consuming. AI models can help speed things up.



# Research questions

- What open-source Japanese-English machine translation transformer models are available?
- How do they perform? Are there common mistakes?
- Can fine-tuning the models improve their performance?
- How does the volume of data affect fine-tuning results?
- Can a small dataset make a difference?



# Approach

1. Find & evaluate open-source transformer models for JA-EN translation
2. Analyze common mistakes (e.g. grammar or subject matter related error)
3. Choose a model to fine-tune with custom data.
4. Experiment with how the volume of training data affect the fine-tuning results.



# Tools & Models

## Hugging Face framework and API

- Model setup
- Datasets
- Training (fine-tuning) and evaluation

## Models

- [Helsinki NLP opus-mt-ja-en](#)
- [FuguMT](#)
- [ElanMT](#)
- [Mbart-ja-en](#)
- Also tried larger [Facebook mBART-50 many to one multilingual machine translation model](#), but was difficult to work with due to computational demands



# Datasets

## Tatoeba dataset

- Multilingual, volunteer maintained by Tatoeba organization
- ~200,000 pairs for Japanese-English language pair
- Mostly shorter, simpler sentences

## Kyoto Free Translation Task (KFTT) dataset

- Originated from professionally translated encyclopedic articles about Kyoto, Japan
- Cleaned and processed for use in machine translation systems
- ~200,000 long, complex sentence pairs on a specialized topic

## Custom dataset focused on Ruby programming language

- Manually created from bilingual Ruby documentation websites
- Also included some synthetic data generated by Claude
- ~400 sentence pairs as of now



# Initial Analysis

## Current model evaluation

- Evaluated all 4 models using the 3 different datasets

## Common mistakes

- Failing to understand the implicit context in the Japanese
- Awkward literal translations.
- Complete breakdown of translation for complex sentences.
- Incorrect or misspelled word when dealing with specialized topics (history, programming languages)
- For Ruby documentation, off style for typical documentation text.

# Output examples with common errors

## Tatoeba

太陽電池で動く自動車を望んでいる。

- **Reference:** I want a car that runs on solar power.
- **Model:** I want a car that works on **solar cells**.

私の時計は10時です。

- **Reference:** It is ten o'clock by my watch.
- **Model:** **My watch is ten o'clock.**

## Custom dataset

Action Cableのチャンネルを通じて、異なるタイプのリアルタイム機能向けの特定のストリームを作成できます。

- **Reference:** Through channels in Action Cable, specific streams can be created for different types of real-time functionality.
- **Model:** **You can create** a specific stream for different types of real-time **functions** through the **ActionCile** channel.

## KFTT

しかし近世に至って表現豊かな楽器である三味線が渡来、完成され、更に箏、胡弓が加わり、これらによる新たな音楽が生まれた。

- **Reference:** However, the shamisen, an instrument full of expression, introduced from abroad and reached its full potential by the early Edo period, and moreover, the koto and kokyu were added and consequently a new kind of music developed.
- **Model:** However, in the next generation, new music has been born with a colorful musical instrument that has been completed and the bow added to it.
  - **Doesn't mention specific instruments, incorrect historic terminology, overall doesn't make sense.**



# Results: Initial Model Evaluation

**JA-EN Machine Translation Transformer Model Evaluation**

Model	Dataset	BLEU	chrF	Translation time (sec)
Helsinki-NLP/opus-mt-ja-en	Tatoeba	0.429	<b>60.66</b>	18.8
	KFTT	0.054	26.64	88.6
	Ruby	0.227	55.79	49.5
Mitsua/elan-mt-bt-ja-en	Tatoeba	<b>0.438</b>	60.49	14.3
	KFTT	0.191	<b>47.67</b>	52.1
	Ruby	0.345	64.58	26.66
staka/fugumt-ja-en	Tatoeba	0.427	59.34	19.4
	KFTT	<b>0.208</b>	47.37	94.8
	Ruby	<b>0.391</b>	<b>67.64</b>	44.5
ken11/mbart-ja-en	Tatoeba	0.261	48.41	24.7
	KFTT	0.033	23.00	62.5
	Ruby	0.128	44.24	30.9

\* Evaluated with the same 100 random samples from each dataset

## BLEU

0.30-0.40 = good, understandable

0.40-0.50 = high quality

0.50-0.60 = very high quality


## chrF

35-40 = acceptable, not great

40-45 = moderate quality

45-50 = high quality

> 55 = very high



## Experiments - Fine-tuning with different amounts of data

Fine-tuning one of the models (Helsinki NLP opus-mt-ja-en) using custom data


- Fine-tuned with 100, 200, 300, and all available samples (353 after test set split) from my custom Ruby programming dataset
- Scored each version of model with BLEU and chrF metrics



## Results: How training data volume affects fine-tuning results

### Helsinki Opus-mt-ja-en fine-tuned with varying number of Ruby-specific training samples

Number of samples	BLEU	chrF
Before fine-tuning	0.250	59.38
100	0.275	60.47
200	0.317	62.37
300	0.326	63.31
353 (All available)	<b>0.337</b>	<b>63.32</b>
Evaluated with test set from Ruby-specific dataset		




# Examples from how training data volume affects fine-tuning results

Action Cableは**WebSockets**をRailsアプリケーションの他の部分と **統合し**、リアルタイム機能を Rubyで記述できるようにします。

EN Reference: Action Cable **integrates WebSockets** with the rest of your Rails application, allowing real-time features to be written in Ruby.

- **Original Model Prediction:**
  - Action Cables **combine webSocks** with other parts of the Rails application and allow you to describe real-time features in Ruby.
- **After 100 samples:**
  - Action Cable **combines webSocks** with other parts of Rails applications and allows you to describe real-time features in Ruby.
- **After 200 samples:**
  - Action Cable **combines webSocks** with the rest of Rails applications so that they can describe real-time features in Ruby.
- **After 300 samples:**
  - Action Cable **combines webSockets** with the rest of Rails applications so that they can describe real-time features in Ruby.
- **After all samples:**
  - Action Cable **integrations WebSocks** with the rest of Rails applications so that they can describe real-time features in Ruby.



# Examples from how training data volume affects fine-tuning results

ブレースの間にあるものは(もし文字列でなければ)文字列に変換され、その外側の文字列の中に置き換えられます。

EN Reference: The bit between the **braces** is turned into a string (if it isn't one already) and then substituted into the outer string at that point.

- **Original Model Prediction:**
  - If it's not a string, then it's converted into a string, and replaced with an outside string. **(no mention of braces)**
- **After 100 samples:**
  - What's between the **buzzers** is converted to a string (if not a string) and replaced with an external string.
- **After 200 samples:**
  - What's between the **buzzers** is converted to a string (if not a string) and replaced with an outside string.
- **After 300 samples:**
  - What is between the **beerace** is converted to a string (if not a string) and replaced with an outside string.
- **After all samples:**
  - What is between the **braces** is converted to a string (if not a string) and replaced with an outside string.



# Challenges & Lessons Learned

- It's hard to find good evaluation and training data that the models haven't seen yet.
- Creating a custom dataset is very tedious and time-consuming, but can have high value.
- Large language models can be prohibitively resource intensive.



# Conclusion & Future Work

## Insights

- Even small transformer models do quite well with JA-EN translation.
- Volume of data does seem to affect fine-tuning results, with even a small increase in the number of examples improving scores.

## Future Work

- Can using a large amount of training data for fine-tuning negatively affect results or is more always better?
- Build on existing custom dataset to further improve the fine-tuned model.
- Redo experiments with more training examples and reevaluate with larger test set.