INFO 3350/6350

Lecture 20: Translation

To do

- Week 15: Social media analysis readings
- PS5 in flight
  - Due Thursday, 11/30, at 11:59pm
- Final exam/project released
  - Due Saturday, December 9, at **noon** per Registrar

In [1]:
```
# Install packages that we need (Colab only)
#!pip install sentencepiece
#!pip install transformers
```

In [2]:
```
# Import packages

from transformers import T5Tokenizer, T5ForConditionalGeneration, M2M100ForCondit
import torch
```

Load the model

We'll use **FLAN-T5 Large** and **M2M** for our translation task.

In [3]:
```
# Load the T5 model
device = "cpu"
model_id = "google/flan-t5-large"
model = T5ForConditionalGeneration.from_pretrained(model_id).to(device)
tokenizer = T5Tokenizer.from_pretrained(model_id)
```

Translation

A simple example to begin. Translate English into German. Many models are pretty good at this. T5 -- at the size we're using -- isn't great.

In [4]:
```
english_text = 'translate: English to German: I live in a small house on a large
inputs = tokenizer(english_text, return_tensors="pt")
input_ids = inputs.input_ids.to(device)
outputs = model.generate(input_ids, max_new_tokens=40)
print(tokenizer.batch_decode(outputs, skip_special_tokens=True))
```

```
['Ich wohne in einer kleinem Haus auf einem großen Hügel.']
```

In [5]:
```
# how about French?
english_text = 'translate: English to French: I live in a small house on a large
```

```
inputs = tokenizer(english_text, return_tensors="pt")
input_ids = inputs.input_ids.to(device)
outputs = model.generate(input_ids, max_new_tokens=40)
print(tokenizer.batch_decode(outputs, skip_special_tokens=True))
```

```
['Je vivo dans une petite maison sur un grand sol.']
```

Ehh ... J'habite une petite maison sur une grande colline?

In [6]:
```
# or Danish?
english_text = 'translate: English to Danish: I live in a small house on a large
inputs = tokenizer(english_text, return_tensors="pt")
input_ids = inputs.input_ids.to(device)
outputs = model.generate(input_ids, max_new_tokens=40)
print(tokenizer.batch_decode(outputs, skip_special_tokens=True))
```

```
['Je liv i en krabbet p en stor hy.']
```

Jeg bor i et lille hus på en stor bakke.

In [7]:
```
# Chinese?
english_text = 'translate: English to Chinese: I live in a small house on a large
inputs = tokenizer(english_text, return_tensors="pt")
input_ids = inputs.input_ids.to(device)
outputs = model.generate(input_ids, max_new_tokens=40)
print(tokenizer.batch_decode(outputs, skip_special_tokens=True))
```

```
['']
```

Try M2M

Meta's M2M is an encoder-decoder model developed for multilingual translation. It supports any-directional translation between 100 languages.

In [8]:
```
# Load models
m2m_model = M2M100ForConditionalGeneration.from_pretrained("facebook/m2m100_418M"
m2m_tokenizer = M2M100Tokenizer.from_pretrained("facebook/m2m100_418M")
```

In [9]:
```
# tokenize input
english_input = 'I live in a small house on a large hill.'
m2m_tokenizer.src_lang = "en"
encoded_en = m2m_tokenizer(english_input, return_tensors="pt")

# translate English to French
generated_tokens = m2m_model.generate(
    **encoded_en, # input ids and attention mask
    forced_bos_token_id=m2m_tokenizer.get_lang_id("fr"), # to French
    max_new_tokens=40
)
print(m2m_tokenizer.batch_decode(generated_tokens, skip_special_tokens=True))

# translate English to Danish
generated_tokens = m2m_model.generate(
    **encoded_en,
    forced_bos_token_id=m2m_tokenizer.get_lang_id("da"), # to Danish
    max_new_tokens=40
)
print(m2m_tokenizer.batch_decode(generated_tokens, skip_special_tokens=True))
```

```
['Je vis dans une petite maison sur une grande colline.']
['Jeg bor i et lille hus på en stor bjerg.']
```

In [10]:

```
# Hindi to English and to Chinese
hi_text = "जीवन एक चॉकलेट बॉक्स की तरह है।" # from the docs
chinese_text = "生活就像一盒巧克力。"
encoded_hi = m2m_tokenizer(hi_text, return_tensors="pt")
generated_tokens = m2m_model.generate(
    **encoded_hi,
    forced_bos_token_id=m2m_tokenizer.get_lang_id("en"), # to English
    max_new_tokens=40
)
print(m2m_tokenizer.batch_decode(generated_tokens, skip_special_tokens=True))

generated_tokens = m2m_model.generate(
    **encoded_hi,
    forced_bos_token_id=m2m_tokenizer.get_lang_id("zh"), # to Chinese
    max_new_tokens=40
)
print(m2m_tokenizer.batch_decode(generated_tokens, skip_special_tokens=True))
```

```
['Life is like a chocolate box.']
['生活就像一盒巧克力。']
```

In [11]:

```
# does the Chinese translation match?
m2m_tokenizer.batch_decode(generated_tokens, skip_special_tokens=True)[0] == chin
```

Out[11]:

```
True
```