18

Evaluating How Gender is Translated ?

He HUANG 22107447

October 9, 2023

1. Using the HuggingFace 🤗 API, translate the test set using a multilingual model such as mBART and a bilingual model (e.g. MarianMT).

Part1: mBART

```
!pip install transformers
!pip install sentencepiece

# access to CUDA if we have one
import torch
device = torch.device("cuda")
if torch.cuda.is_available() else torch.device("cpu")
device # return device(type='cuda)

# define the mBart model and tokenizer
from transformers import MBartForConditionalGeneration, MBart50TokenizerFast
mbart_name = "facebook/mbart-large-50-many-to-many-mmt"
mbart_model = MBartForConditionalGeneration.from_pretrained(mbart_name).to(
                                              device)
mbart_tokenizer = MBart50TokenizerFast.from_pretrained(mbart_name,
     src_lang="fr_XX")
```

```
# translate french sentences into english
def mbart_translate(fra_sent):
# encode the french sentence
  inputs = mbart_tokenizer(fra_sent["text"],  return_tensors="pt",
        truncation=True, max_length=512, padding=True).to(device)
# generate
  generated_tokens = mbart_model.generate(**inputs,
        forced_bos_token_id=mbart_tokenizer.lang_code_to_id["en_XX"])
# decode the english sentence
  eng_sent = mbart_tokenizer.batch_decode(generated_tokens,
        skip_special_tokens=True)
  return {"mbart_fr_en": eng_sent}
```

```
# using the map function
dataset = dataset.map(mbart_translate,batched=True, batch_size=8)
```

Part2: MarianMT

```
!pip install sacremoses

# define the MarianMT model and tokenizer
from transformers import MarianMTModel, MarianTokenizer
mt_name = "Helsinki-NLP/opus-mt-fr-en"
```

```
mt_model = MarianMTModel.from_pretrained(mt_name).to(device)
mt_tokenizer = MarianTokenizer.from_pretrained(mt_name)
```

```
# translate french sentences into english
def mt_translate(sent_fra):
# encode the french sentence
  inputs = mt_tokenizer(sent_fra["text"], return_tensors="pt", truncation=True
                                               , max_length=512, padding=True).to(
                                               device)
# generate
  generated_tokens = mt_model.generate(**inputs)
# decode the english sentence
  eng_sent = [mt_tokenizer.decode(g, skip_special_tokens=True) for g in
                                               generated_tokens]
  return {"mt_fr_en": eng_sent}
```

2. Evaluate the quality of the translation using the BLEU score both at the corpus level (i.e. considering all sentences as belonging to the same corpus) and for each gender (i.e. by separating sentences according to the grammatical gender of French sentences). What do you conclude ?

Part 1: Compute the BLEU score for the entire translated corpus

```
!pip install sacrebleu

import sacrebleu

# mbart
mbart_trans = dataset["text"]["mbart_fr_en"]
# fr->en
mbart_bleu = sacrebleu.corpus_bleu(mbart_trans, [ref_en]).score
print(f'mBART BLEU score: {mbart_bleu:.2f}')
# mBART BLEU score: 46.00

# MarianMT
mt_trans = dataset["text"]["mt_fr_en"]
# fr->en
marian_bleu = sacrebleu.corpus_bleu(mt_trans, [ref_en]).score
print(f'MarianMT BLEU score: {marian_bleu:.2f}')
# MarianMT BLEU score: 49.29
```

and what do you conclude ?

Part 2: For each gender
From the french corpus, we can observe that all these nouns matching the masculine or feminine are distributed equally throughout the sentence, also as presented in the article (Wisniewski et al. 2022a). The first sentence is masculine and the second is feminine. Some nouns, on the other hand, share a single word form for the feminine and masculine, such as ”accessoiriste”,”voyagiste”,”zoologiste”, etc. Based on this distinction between the grammatical gender, i categorized the sentences as belonging to the masculine, feminine, and epicene respectively.

```
# split gender list in french sentences
import re

epicene_phrase = []
masc_phrase = []
fem_phrase = []
```

```
# i considered that, the epicene sentences have the same derterminer and noun
# check if the first sentence is equal to the secode
def get_gender(list_fr):
  for i in range(len(list_fr)-1):
    if list_fr[i] == list_fr[i+1]:
      epicene_phrase.append(list_fr[i])
  # exclude the epicene phrase from the total list
  leftover = [ex for ex in list_fr if ex not in epicene_phrase]
  for i in range(len(leftover)):
    # if the index is odd
    if i % 2 != 0:
      fem_phrase.append(leftover[i])
    # if the index is even
    else:
      masc_phrase.append(leftover[i])
```

easier to do that with a dataframe

```
# filter the dataset according to each gender
masc_filterted = dataset.filter(lambda x: x["src_gender"] == "masculine")
fem_filterted = dataset.filter(lambda x: x["src_gender"] == "feminine")
epicene_filterted = dataset.filter(lambda x: x["src_gender"] == "epicene")
```

Here, i compute the proportion of sentences for each category:

```
# compute the proportion

masc_total = len(masc_filterted["text"])
masc_prop = masc_total / len(dataset["text"])
print(f"The proportion for masculine sentences
in total dataset: {masc_prop:.2%}")

fem_total = len(fem_filterted["text"])
fem_prop = fem_total / len(dataset["text"])
print(f"The proportion for feminine sentences
in total dataset: {fem_prop:.2%}")

epicene_total = len(epicene_filterted["text"])
epicene_prop = epicene_total / len(dataset["text"])
print(f"The proportion for epicene sentences
in total dataset: {epicene_prop:.2%}")
```

	examples	proportion
epicene	272	8.01
masculine	1561	45.99
feminine	1561	45.99

Then, based on this part, i mapped out the translations of English sentences and compute the BLEU score.

```
# mBART
# BLEU score for masculine and feminine sentences

mbart_bleu_m = sacrebleu.corpus_bleu(masc_filterted['text']['mbart_fr_en'], [
                                      masc_filterted['text']['ref']]).score

mbart_bleu_f = sacrebleu.corpus_bleu(fem_filterted['text']['mbart_fr_en'], [
                                      fem_filterted['text']['ref']]).score

print(f'mBART BLEU score for masculine: {mbart_bleu_m:.2f}')
print(f'mBART BLEU score for feminine: {mbart_bleu_f:.2f}')
```

```
# MarianMT
marian_bleu_m = sacrebleu.corpus_bleu(masc_filterted['text']['mt_fr_en'], [
                                      masc_filterted['text']['ref']]).score
marian_bleu_f = sacrebleu.corpus_bleu(fem_filterted['text']['mt_fr_en'], [
                                      fem_filterted['text']['ref']]).score

print(f'MarianMT BLEU score for masculine: {marian_bleu_m:.2f}')
print(f'MarianMT BLEU score for feminine: {marian_bleu_f:.2f}')
```

The results are as follows:

	mBART	MarianMT
corpus	46.00	49.29
masculine	49.89	53.07
feminine	42.05	45.54

and epicene ?

From the total text, the translation accuracy of the MarianMT is a little higher than that of mBART. The difference between the two models is not very significant. After splitting the French sentences into two gender sentences, for MarianMT itself, the accuracy of prediction for masculine sentences is better than the feminine sentences. And at this point, it's the same case for mBART.
In French, the way that express the feminine are more inexplicit than masculine. For instance, "le bagagiste" and "la bagagiste", only the determiner can match the gender. And the presence of prossessive pronoun "son" in French will also cause the ambiguity in translation.

3. Determine the proportion of sentences in which the gender is correctly translated according to the way it is expressed in French.

Gender can be marked in French in various ways (Wisniewski et al. 2022b):
· by the determiner (e.g la journaliste)
· by the noun (e.g l'actrice)
· by the determiner and the noun (e.g la traductrice)
· not be marked by determiner and noun (e.g. l'analyste)

```
# get french determiner and nouns
# get english possessive pronoun
# get possessive pronoun translated

def split_sent(examples):
  examples["det_fr"] = re.split("'|\s+", examples["text"])[0]
  examples["nouns_fr"] = re.split("'|\s+", examples["text"])[1]
  examples["pronoun_en"] = re.split(" ", examples["ref"])[-2]
  examples["pred_pronoun_mbart"] = re.split(" ", examples["mbart_fr_en"])[-2]
  examples["pred_pronoun_mt"] = re.split(" ", examples["mt_fr_en"])[-2]
  return examples
dataset = dataset.map(split_sent)
```

```
# convert to dataframe to compute each proportion

import pandas as pd
```

```
df = dataset["text"].to_pandas()

# epicene nouns
# if noun is epicene, return True, otherwise return False

df["fra_noun_epi"] = df["nouns_fr"].duplicated(keep=False)
df["eng_noun_epi"] = df["nouns_en"].duplicated(keep=False)

# according to the different way to express french gender
# take each subset from dataframe

# epicene
epicene_part = df[(df["fra_noun_epi"] == True) & (df["det_fr"] == "l")]
# det and noun
det_noun = df[(df["det_fr"] != "l") & (df["fra_noun_epi"] == False)]
# determiner only
det_mark = df[(df["fra_noun_epi"]) & (df["det_fr"] != "l")]
# noun only
noun = df[(df["det_fr"] == "l") & (df["fra_noun_epi"] == False)]
```

```
# proportion of each part in df
epicene_df = epicene_part.shape[0] / df.shape[0]
det_noun_df = det_noun.shape[0] / df.shape[0]
det_mark_df = det_mark.shape[0] / df.shape[0]
noun_df = noun.shape[0] / df.shape[0]
```

```
def compute(data):
  # compute the number of sentences
  # in which the processives pronouns are correctly translated
  # calculate the proportion in the corresponding gender sentences

  accuracy_mbart = (data["pred_pronoun_mbart"]
        == data["pronoun_en"]).sum() / len(data)
  accuracy_mt = (data["pred_pronoun_mt"]
        == data["pronoun_en"]).sum() / len(data)

  return f"Accuracy for mBART model:  {accuracy_mbart:.2%}
        Accuracy for MT model:{accuracy_mt:.2%}"
```

The results:

Source text	Proportion	mBART	MarianMT
epicene	8.01	45.96	49.26
det and noun	52.68	66.83	91.44
det only	24.51	74.76	90.14
noun only	14.79	66.14	78.09

From the proportion, there are more than half of the sentences are marked by the determiner and noun. For `mBART` itself, the accuracy in translation of the sentences that are only marked by the determiner, is higher than in other cases. For `MarianMT`, the accuracy of translation in sentences is significant when the sentence is expressed by determiner and noun.

4. Represent the distribution of the number of sub-word tokens in occupational nouns according to gender in English and French. What is the impact of the number of sub-word tokens on the quality of the translation of gender ?

```python
# obtain the occupational nouns into subtoken units
from transformers import AutoTokenizer
def get_subwords(examples):
  tokenizer = AutoTokenizer.from_pretrained("bert-base-multilingual-cased")
  fr_subwords = [tokenizer.tokenize(noun) for noun in examples["nouns_fr"]]
  en_subwords = [tokenizer.tokenize(noun) for noun in examples["nouns_en"]]
  examples["fr_subwords"] = fr_subwords
  examples["en_subwords"] = en_subwords
  return examples
dataset = dataset.map(get_subwords, batched=True)
```

```python
# compute the number of subwords
def compute_subwords(examples):
  fr_num_subwords = [len(subword) for subword in examples["fr_subwords"]]
  en_num_subwords = [len(subword) for subword in examples["en_subwords"]]
  examples["fr_num_subwords"] = fr_num_subwords
  examples["en_num_subwords"] = en_num_subwords
  return examples
dataset = dataset.map(compute_subwords, batched=True)
```

```python
# add french and english subwords and the number of subwords to the dataframe
df.loc[:, "fr_subwords"] = dataset["text"]["fr_subwords"]
df.loc[:, "en_subwords"] = dataset["text"]["en_subwords"]
df.loc[:, "fr_num_subwords"] = dataset["text"]["fr_num_subwords"]
df.loc[:, "en_num_subwords"] = dataset["text"]["en_num_subwords"]
```

```python
# groupby the number of subwords and compute the proportion
prop_fr_subwords = df.groupby("fr_num_subwords").count()['text']/len(df)
prop_en_subwords = df.groupby("en_num_subwords").count()['text']/len(df)
```

```python
# compute the accuracy according to the number of subwords
def compute_acc_sub(ds):
  accuracy_mbart = (ds["pred_pronoun_mbart"] == ds["pronoun_en"]).sum() /
                                                                    len(ds)
  accuracy_mt = (ds["pred_pronoun_mt"] == ds["pronoun_en"]).sum() / len(ds)
  return f"mBART model: {accuracy_mbart:.2%} MT model: {accuracy_mt:.2%}"
```

```python
subwords_acc = df.groupby("fr_num_subwords").apply(compute_acc_sub)
```

And here's the result:

Subwords	Proportion	mBART	MarianMT
1	6.36	63.89	84.72
2	28.04	65.34	89.92
3	39.06	65.99	85.14
4	20.88	68.69	82.65
5	4.41	81.33	83.33
6	1.06	77.78	83.33
7	0.08	33.33	33.33
8	0.05	50.00	50.00

There are two to four subwords which are relatively frequent in French occupational nouns. For the case of occupational nouns with two subwords, `MarianMT` had the highest accuracy.

When there are 5 subwords, `mBART` can reach an accuracy of 80%. In French, since feminine occupational nouns are inflected from masculine nouns by changing the form of the word. We can presume that those feminine occupational nouns are composed of more subwords. When there are more subwords, the accuracy tends to decrease for both models. In summary, translating French occupational nouns into English, the feminine are difficult to translate correctly, because occupational nouns do not reflect specific genders in Freench and there are some ambiguous expressions. And, morphologicallym words are consist of more than one morpheme.

ok

References

Wisniewski, Guillaume et al. (Dec. 2022a). "Analyzing Gender Translation Errors to Identify Information Flows between the Encoder and Decoder of a NMT System". In: *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, pp. 153–163. DOI: 10.18653/v1/2022.blackboxnlp-1.13. URL: https://aclanthology.org/2022.blackboxnlp-1.13.

— (2022b). "Biais de genre dans un système de traduction automatique neuronale : une étude des mécanismes de transfert cross-langue [Gender bias in a neural machine translation system: a study of crosslingual transfer mechanisms]". French. In: *Traitement Automatique des Langues, Volume 63, Numéro 1 : Varia [Varia]*. France: ATALA (Association pour le Traitement Automatique des Langues), pp. 37–61. URL: https://aclanthology.org/2022.tal-1.2.