hw3_final

March 23, 2022

1 SI630 Assignment 3 Data Annotation

1.1 Group 7

[2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy
import numpy as np
import krippendorff
import torch

from transformers import pipeline
from datasets import load_dataset

group_labels = pd.read_csv("Group7_annotation_final(1).csv")
user1 = group_labels[group_labels['annotator'] == 'user1']
user2 = group_labels[group_labels['annotator'] == 'user2']
user3 = group_labels[group_labels['annotator'] == 'user3']

hw_data = pd.read_csv("si630w22-hw3-data.csv")

train_df = pd.read_csv("si630w22-hw3-train.csv")
dev_df = pd.read_csv("si630w22-hw3-dev.csv")
my_group_labels = train_df[train_df['group'] == 'group_07']
other_group_labels = train_df[train_df['group'] != 'group_07']
```

1.2 Problem 6: Inter Annotator Agreement

1.2.1 6.1 Pearson's correlation r (nominal) and Krippendorff's (ordinal, nominal)

[3]:
```
user1_rating_series = user1.rating.reset_index().rating
user2_rating_series = user2.rating.reset_index().rating
user3_rating_series = user3.rating.reset_index().rating

user_ratings_nominal_df = pd.DataFrame({
    'user1': user1_rating_series,
    'user2': user2_rating_series,
    'user3': user3_rating_series
})

user_ratings_ordinal_df = user_ratings_nominal_df.copy()
user_ratings_ordinal_df[['user1','user2','user3']] =␣
 ↪user_ratings_ordinal_df[['user1','user2','user3']].astype(str)

r, p = scipy.stats.pearsonr(user2.rating, user3.rating)
print(r)
print(user2_rating_series.corr(user3_rating_series))

print()
print('Pearson\'s Correlation:')
print(user_ratings_nominal_df.corr())
print()


print("Krippendorff's alpha for nominal metric: ",
      krippendorff.alpha(reliability_data=[user1.rating, user2.rating, user3.
 ↪rating], level_of_measurement="nominal"))

print("Krippendorff's alpha for interval metric: ",
      krippendorff.alpha(reliability_data=[user1.rating, user2.rating, user3.
 ↪rating]))
# user_ratings_ordinal_df['user1'][0]
```

```
0.5901072744862975
0.5901072744862973

Pearson's Correlation:
          user1     user2     user3
user1  1.000000  0.513242  0.558525
user2  0.513242  1.000000  0.590107
user3  0.558525  0.590107  1.000000

Krippendorff's alpha for nominal metric:  0.451075921545643
Krippendorff's alpha for interval metric:  0.5467531727277909
```

1.2.2 P6 Answer

The correlation coefficient between user1 and user2, user1 and user3 and user2 and user3 are 0.513, 0.559 and 0.590 respectively, which means user1 has relatively low correlation with the other two users. Our group Krippendorff' s alpha for ordinal metric is: 0.583 and Krippendorff' s alpha for nominal metric: 0.451, which is not too bad, showing that both our guideline is not too unclear and we were able to maintain some kind of consistency annotating following our guide.

6.2

Even though our r and alpha are close in magnitude, we don't think there is any further interpretation for that closeness. The r is the correlation coefficient between each two users and Kirppendoff' s alpha can even be negative if the consistency is too bad.

Pearson's Correlation between any two annotators was consistently higher than the nomail metric yielded for Krippendorff's alpha. This is likely due to the effect where the group of annotators collectively constructed and followed a common guideline during the annotation activity, and that guideline exhibits more ordering tasks than classification tasks.

6.3

In this case, the ordinal metric Krippendorff's alpha should be applied in this setting. Nominal metric applies for categories that have no predetermined order like colors: red, green and blue. There is no order between categories. The ordinal metric for alpha (0.547) was higher than the nominal metric (0.451), which means that more agreement was observed when the ratings data was processed as ordinal values. Ordinal metric applies for cases like ranking based on a scale from 1 to 5 meaning that 1 is worse than 3, where we can establish an order between categories even though the magnitude of it doesn't mean much except for their order. So with knowledge about the annotation task, we should take Krippendorff's alpha for ordinal metric, which is 0.583.

1.3 Problem 7: Compute agreement of all other annotators on my group's items

[4]:
```
for group in set(train_df.group):
    group_data = train_df[train_df['group'] == group]
    data = []
    for i, v in enumerate(set(group_data.annotator_id)):
        rater = group_data[group_data['annotator_id'] == v].rating
        data.append(rater)
    print(f"Krippendorff's ordinal alpha for {group}:",
      krippendorff.alpha(reliability_data=data))

group_data = train_df[train_df["group"] == 'group_08']
```

```
Krippendorff's ordinal alpha for group_25: 0.8152188318771727
Krippendorff's ordinal alpha for group_18: 0.3394115959229721
Krippendorff's ordinal alpha for group_10: 0.22156061020132156
Krippendorff's ordinal alpha for group_15: 0.44608703758370494
Krippendorff's ordinal alpha for group_04: 0.6344654098794749
Krippendorff's ordinal alpha for group_14: 0.8501096667401515
Krippendorff's ordinal alpha for group_17: 0.5005291005291005
Krippendorff's ordinal alpha for group_03: 0.5177571757687727
Krippendorff's ordinal alpha for group_16: 0.48265882081901434
Krippendorff's ordinal alpha for group_01: 0.5496867847957907
Krippendorff's ordinal alpha for group_07: 0.6007873926711043
Krippendorff's ordinal alpha for group_23: 0.9576501779111142
Krippendorff's ordinal alpha for group_09: 0.6158133595623889
Krippendorff's ordinal alpha for group_24: 0.5934215579792441
Krippendorff's ordinal alpha for group_05: 0.3909864259661111
Krippendorff's ordinal alpha for group_19: 0.8036164603392939
Krippendorff's ordinal alpha for group_11: 0.24683527979531206
```

```
Krippendorff's ordinal alpha for group_21: 0.8327358321461928
Krippendorff's ordinal alpha for group_22: 0.2738881340794862
Krippendorff's ordinal alpha for group_08: -0.5262098618607629
Krippendorff's ordinal alpha for group_12: 0.7813640482277473
Krippendorff's ordinal alpha for group_02: 0.6551702202312144
Krippendorff's ordinal alpha for group_20: 0.6413369326408367
Krippendorff's ordinal alpha for group_13: 0.5265997616166396
```

1.3.1 P7 Answer

Based on the training set, my group (group 07) had achieved an agreement that lies somewhere in the middle of the pack (0.60). The lowest calculated agreement was negative (-0.52), which means the annotators did worse than randomly assigning labels to the data set, while the highest alpha was 0.95. Having examined other groups' guidelines, it was clear that some guidelines were easier to interpret from the standpoint of an annotator, while others felt ambiguous and thus prone to arriving at different understandings.

The Krippendorff's alpha for ordinal metric in the other group is 0.612, which is slightly higher than our group but also close in magnitude. The reason for their guideline is more consistent is that they divide the question into different categories, which is very helpful. For different kinds of questions, we may have different standards for the quality of the answer. For example, for an "asking for suggestions" question, we will expect the reply to elaborate the reason for that, however, for a "asking for personal experience" question, it is unreasonable to ask for a reason for that, which increases the inconsistency in our result.

1.4 Problem 8: Examining Inter-group Disagreement of top 10 comment ratings

```
[6]: merged_labels = pd.merge(my_group_labels, other_group_labels, how='inner',␣
      ↪on=['id'])

     merged_labels[merged_labels['id'] == 't3_n2agq3']

     my_group_grouped_mean = my_group_labels.groupby('id').rating.mean()
     other_group_grouped_mean = other_group_labels.groupby('id').rating.mean()

     inter_group_averages = []

     for _id in set(my_group_labels.id):
         my_avg = my_group_grouped_mean[_id]
         if _id in set(other_group_labels.id):
             other_avg = other_group_grouped_mean[_id]
             group = other_group_labels[other_group_labels['id'] == _id].group.
      ↪values[0]
         data = [_id, my_avg, other_avg, group]
         inter_group_averages.append(data)

     inter_group_averages[7] # id, my_avg, other_avg
```

```
inter_group_averages_df = pd.DataFrame(inter_group_averages, columns= ['id',␣
↪'my_avg', 'other_avg', 'group'])

inter_group_averages_df['diff'] = abs(inter_group_averages_df.my_avg -␣
↪inter_group_averages_df.other_avg)

inter_group_averages_df.sort_values(by="diff", ascending=False)[:5]
```

[6]:
```
              id  my_avg  other_avg     group  diff
278  t3_nefhky     5.0        1.0  group_24   4.0
13   t3_nb4000     5.0        2.0  group_01   3.0
277  t3_n94s6q     5.0        2.0  group_11   3.0
122  t3_n9xdlv     5.0        2.0  group_01   3.0
116  t3_nngwzf     5.0        2.0  group_08   3.0
```

1.4.1 Investigate the top 10 questions and replies that yielded the biggest differences

[7]:
```
for i in inter_group_averages_df.sort_values(by="diff", ascending=False)[:10].
↪values:
    print(f'Question ID: {i[0]}, group_07_avg = {i[1]}, {i[3]}_avg = {i[2]} ||␣
↪diff = {i[-1]}')
    q = hw_data[hw_data['question_id'] == i[0]].question_text.values[0]
    r = hw_data[hw_data['question_id'] == i[0]].reply_text.values[0]
    print(f'Q: {q}')
    print(f'R: {r}')
    print()
```

```
Question ID: t3_nefhky, group_07_avg = 5.0, group_24_avg = 1.0 || diff = 4.0
Q: What are some relationship "green flags" that your SO might be a plant?
R: If you sit at home all time and you don't socialize you become a potato… To
be exact a couch potato

Question ID: t3_nb4000, group_07_avg = 5.0, group_01_avg = 2.0 || diff = 3.0
Q: When looking for a partner does your type really matter to you or can you see
yourself dropping a few key wants in favour of realistic compromise?
R: I've not yet looked for a partner; however it's always been my perspective
that types really don't matter. Once I get to know people I don't find myself
consciously make decisions on whether or not to be attracted to them. And even
though I have certain things I find most attractive (like brunette, curly hair),
the women I've been attracted to haven't necessarily fit that.

Question ID: t3_n94s6q, group_07_avg = 5.0, group_11_avg = 2.0 || diff = 3.0
Q: How did Kant determine on which level to classify an act?
R: &gt; As I understand it, Kant determined that lying was wrong because if
everyone did it, we would hate that. No, this is not true. Insofar as we might
say that Kant anywhere defends the view that lying is always wrong, he would be
presumably doing so using one of the forms of the categorical imperative. Those
```

```
being, roughly: 1. Such an act would fall under a maxim which cannot be willed
to be a universal law. 2. Such an act treats a person as a mere means. 3. Such
an act is inconsistent with considering each rational will as a universally
legislating will. 4. Such an act would fall under a maxim which is inconsistent
with being a universally legislating member of a merely possible kingdom of
ends. I take it that, perhaps, you are imagining it violates the first one, but
the first one does not involve checking maxims against what people hate. There
are a few proposals about how the first formula works, but among them are things
like: the maxim cannot be willed to be a universal law because…(1) it contains
a logical contradiction, (2) in a world where this rule exists the proposed act
is imprudent, or (3) a world where this rule exists is generally inconsistent
with the agent's requirements for a world.

Question ID: t3_n9xdlv, group_07_avg = 5.0, group_01_avg = 2.0 || diff = 3.0
Q: Why can we eat pigs but not dogs when they are of comparable intelligence?
R: Dogs were bread and evolved as human friend and companion while pigs were
used as food , even in the wild they are food for other animals

Question ID: t3_nngwzf, group_07_avg = 5.0, group_08_avg = 2.0 || diff = 3.0
Q: What’s something you love telling your children?
R: "To make monkeys ask questions." As an answer when my actions are being
questioned. Works well with my husband too.

Question ID: t3_nl4icj, group_07_avg = 4.666666666666667, group_12_avg = 2.0 ||
diff = 2.666666666666667
Q: What is a cringy scene from a show/movie that no one talks about?
R: Stranger things at the end of the last season when that one girl is singing
the song during the intense scene.

Question ID: t3_n46meh, group_07_avg = 5.0, group_25_avg = 2.5 || diff = 2.5
Q: LPT Request: How to be more participative in classes?
R: If somebody else says something you can always say "I really like what x said
about y, I agree with this point a lot and &lt;something something that rehashes
what that person said&gt;"

Question ID: t3_no6uzu, group_07_avg = 5.0, group_01_avg = 2.5 || diff = 2.5
Q: How much personal information should you share as you move across friendship
tiers (i.e. acquaintance to friend to close friend to best friend, spouse or
family)?
R: That depends on how much trust are you willing to put in a certain people. If
you're not comfortable sharing something, that it's probably not a good idea to
do it.

Question ID: t3_nbonu6, group_07_avg = 5.0, group_08_avg = 2.5 || diff = 2.5
Q: How has the pandemic affected you in a positive way?
R: With the lack of social male exspectations I realized that I am trans

Question ID: t3_n5e2h1, group_07_avg = 5.0, group_01_avg = 2.5 || diff = 2.5
```

```
Q: What's a horrible way someone tried to be funny?
R: At a work function - I never go to these. I don't drink, I'm awkward in
social settings and don't handle noise/filtering multiple conversations very
well so overall it's a bad time. Coworkers convinced me to go. I went, and there
was a lull so I sat myself at the side and was checking something on my phone as
the comedian came out. He first told the bar to "shut the fuck up, it's my time
now" which went…brilliantly as you'd expect in a room full of half-drunk
military personnel. He then spotted me, and started kicking off that I should be
paying attention to his act. To be clear: this was a 1 hour set, not the actual
event we're there for. He started going off on my clothes, my hair, my weight
(Overweight, pre-transition transgender…) and I was just…shocked. Couldn't
respond, being called out in front of like 100 coworkers. Luckily one of my
coworkers stuck up for me and started telling him to leave me alone, pick on him
if he wants someone to fight with etc and the 'comedian' was booed off stage.
And that's why I now refuse point blank to attend work events.
```

1.4.2 8 Answer:

Question ID: t3_nefhky

Our group averaged a 5 while group 24 averaged a score of 1. The difference seems to lie in the definition of "relevant" content as outlined in both groups' guidelines. Such a concept is prone to different interpretations due to factors such as culture and background. With that said we believe a score of 5 is appropriate as the reply clearly addresses the question, and in a relevant way (albeit half jokingly).

Question ID: t3_n94s6q

Our group averaged a 5 while group 11 averaged a score of 2. Group 11's guideline differed from ours (7) where it prompts for the annotator to look for supplemental "context" in replies in order to score higher than a 3. Although both the question and the answer clearly lacks context to explain any foreign concepts, the answer directly addresses any of the concepts mentioned in the question such as "Kant" which is why we believe a score of 5 should stand.

Question ID: t3_nb4000

Our group averaged a 5 while group 1 averaged a score of 2. The difference is credited to group 1's use of character counts as a criteria in assessing the helpfulness of a comment while ours did not use it as an explicit requirement. The reply falls short of group 1's threshold for a helpful comment which is predominately why it did not score over a 2. We believe a score of 5 should stand given that the answer directly addresses the question and provides ample explanation despite the length of the composition.

Question ID: t3_nngwzf

Our group averaged a 5 while group 8 averaged a score of 2. The difference comes from group 8's guideline that outlines any replies without an explanation would lead to a 2. While our guidelins also mentions explanation and elaborateness, there is clearly a mismatch between how the two groups preceived what constituted valid explanation. Subjectively we think the reply provides relevant explanation, which is why a score of 5 should stay.

Question ID: t3_n9xdlv

Our group averaged a 5 while group 1 averaged a score of 2. Once again difference seems to stem from group 1's requirement involving character counts while our group's guidelines doesn't.

Question ID: t3_n7v3ck

Our group averaged a 4.67 while group 16 averaged a score of 2. Group 16's scoring guideline on vague / ambiguous terms seem to led them to reach a consensus of 2. While our guidlines also addresses the problem with vagueness (by deducting a point), we believe their punishment for vague answers seemed too significant all things considered. In addition the reply also does not go into great detail in explaining itself, which is why we believe that meeting at the middle with group 16 and reassigning it a score of 3 is appropriate here.

Question ID: t3_nl4icj

Our group averaged a 4.67 while group 12 averaged a score of 2. Group 12's guideline for a score of 2 states that a reply either has little or no relation to the question, or it does not elicit meaningful information in regards to the question; which we disagree with. This mismatch could be due to differing cultures/backgrounds of the annotators, which is why we still believe a score of 4/5 is appropriate for this item.

Question ID: t3_nom59y

Our group averaged a 2.33 while group 2 averaged a score of 5. This is an interesting one since the group clearly outlines how questions that do not answer the question belonging in the 1-2 categories, yet the reply averaged a 5 across all members. We believe that a lower score such as 1-3 should stand as the reply acknoledges the question, but does not supply any answers to address it.

Question ID: t3_no6uzu

Our group averaged a 5 while group 1 averaged a score of 2.5. Once again this is due to group 1 imposing a character count criteria while our group does not. We think our score of 5 is appropriate.

Question ID: t3_n3jjkf

Our group averaged a 5 while group 1 averaged a score of 2.5. Again, the character count criteria seems to be the reason why the cleraly relavant and sufficiently thourough reply only averaged a 2.5 by group 1. We believe our 5 should stand.

1.5 Problem 9: Imrpovements to own guildlines to increase annotator agreement or correct for gaps and edge cases.

1.5.1 9 Answer:

- Standardize character count as a criteria
  - it should be made apparent whether strict character counts should be employed as a criteria. This is most useful in assessing the degree to which a reply is thourough / explicative
- Standadize punishment toward vagueness / ambiguity
  - in the event that annotators are having trouble assessing any aspect of a reply due to vague / ambiguous / unclear wording, the course of action for such occurences should be made apparent to reduce variation in interpretations
- Better define relevance

– clearly outline what relevance means for the annotator: (Q&A subject agreement? Comprehensiveness? etc.)

1.6 Problem 10-12: Training a Classifier using HuggingFace Trainer Class

[1]:
```
def find_qa_text(question_id):
    try:
        return qa[qa['question_id'] == question_id].iloc[0].raw_data
    except:
        pass
```

[2]:
```
import pandas as pd
train = pd.read_csv("si630w22-hw3-train.csv")
dev = pd.read_csv("si630w22-hw3-dev.csv")
qa = pd.read_csv("si630w22-hw3-data.csv")
qa["raw_data"] = qa["question_text"] + qa["reply_text"]
# qa["raw_data"] = qa["question_text"]
train["qa_text"] = train["id"].apply(find_qa_text)
train["text"] = train["annotator_id"] * 5 + train["group"] * 5 +␣
 ↪train["qa_text"]
dev["qa_text"] = dev["id"].map(find_qa_text)
dev["text"] = dev["annotator_id"] * 5+ dev["group"] * 5 + dev["qa_text"]
final = pd.read_csv("si630w22-hw3-test.public.csv")
final["qa_text"] = final["id"].map(find_qa_text)
final["text"] = final["annotator_id"] * 5+ final["group"] * 5 + final["qa_text"]

# train["qa_text"] = train["id"].apply(find_qa_text)
# train["text"] = train["qa_text"]
# dev["qa_text"] = dev["id"].map(find_qa_text)
# dev["text"] = dev["qa_text"]
```

[3]:
```
def minus1(rating):
    return rating - 1.0
```

[4]:
```
# train_labels
```

[5]:
```
train_texts, train_labels = train["text"].tolist(), train["rating"].
 ↪apply(minus1).tolist()
test_texts, test_labels = dev["text"].tolist(), dev["rating"].apply(minus1).
 ↪tolist()
final_texts, final_labels = final["text"].tolist(), [0 for i in␣
 ↪range(len(final))]
```

[6]:
```
from transformers import AutoTokenizer
import torch
device = "cuda:0" if torch.cuda.is_available() else "cpu"
tokenizer = AutoTokenizer.from_pretrained("microsoft/MiniLM-L12-H384-uncased")
```

```
max_length = 512
train_encodings = tokenizer(train_texts, truncation=True, padding="max_length",␣
↪max_length=max_length)
test_encodings = tokenizer(test_texts, truncation=True, padding="max_length",␣
↪max_length=max_length)
final_encodings = tokenizer(final_texts, truncation=True, padding="max_length",␣
↪max_length=max_length)
# small_train_encodings = tokenizer(train_texts[0:100], truncation=True,␣
↪padding="max_length", max_length=max_length)
# small_test_encodings = tokenizer(test_texts[0:100], truncation=True,␣
↪padding="max_length", max_length=max_length)
```

[7]:
```
class HelpfulnessDataset(torch.utils.data.Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels

    def __getitem__(self, idx):
        item = {key: torch.tensor(val[idx]) for key, val in self.encodings.
↪items()}
        item['labels'] = torch.tensor(self.labels[idx])
        return item

    def __len__(self):
        return len(self.labels)

train_dataset = HelpfulnessDataset(train_encodings, train_labels)
test_dataset = HelpfulnessDataset(test_encodings, test_labels)
final_dataset = HelpfulnessDataset(final_encodings, final_labels)
```

[8]:
```
# small_train_dataset = HelpfulnessDataset(small_train_encodings,␣
↪train_labels[0:100])
# small_test_dataset = HelpfulnessDataset(small_test_encodings, test_labels[0:
↪100])
```

[9]:
```
from sklearn.metrics import accuracy_score
import numpy as np

def compute_metrics(pred):
  labels = pred.label_ids
  preds = np.round_(pred.predictions)
#   preds = pred.predictions.argmax(-1)
  # calculate accuracy using sklearn's function
  acc = accuracy_score(labels, preds)
  return {
      'accuracy': acc,
  }
```

[10]:
```
from torch import nn
from transformers import Trainer


class CustomTrainer(Trainer):
    def compute_loss(self, model, inputs, return_outputs=False):
        labels = inputs.get("labels")
        # forward pass
        outputs = model(**inputs)
        logits = outputs.get("logits")
        # compute custom loss (suppose one has 3 labels with different weights)
        loss_fct = torch.nn.MSELoss()
        loss = loss_fct(logits.view(-1, self.model.config.num_labels), labels.
 ↪view(-1))
        return (loss, outputs) if return_outputs else loss
```

[12]:
```
from transformers import AutoModelForSequenceClassification
model = AutoModelForSequenceClassification.from_pretrained("microsoft/
 ↪Multilingual-MiniLM-L12-H384", num_labels=1).to(device)
```

```
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at microsoft/Multilingual-MiniLM-L12-H384 and are newly
initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

[13]:
```
from transformers import Trainer, TrainingArguments

training_args = TrainingArguments(
    output_dir='./results',          # output directory
    num_train_epochs=6,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate␣
 ↪scheduler
    weight_decay=0.03,               # strength of weight decay
    logging_dir='./logs',            # directory for storing logs
    logging_steps=80,
)

trainer = CustomTrainer(
    model=model,                         # the instantiated Transformers model␣
 ↪to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=test_dataset,             # evaluation dataset
    compute_metrics=compute_metrics
)
```

```
[ ]: trainer.train()
```

```
/home/fangzhel/.local/lib/python3.8/site-
packages/transformers/optimization.py:306: FutureWarning: This implementation of
AdamW is deprecated and will be removed in a future version. Use the PyTorch
implementation torch.optim.AdamW instead, or set `no_deprecation_warning=True`
to disable this warning
  warnings.warn(
***** Running training *****
  Num examples = 17845
  Num Epochs = 6
  Instantaneous batch size per device = 16
  Total train batch size (w. parallel, distributed & accumulation) = 16
  Gradient Accumulation steps = 1
  Total optimization steps = 6696
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)

<IPython.core.display.HTML object>

Saving model checkpoint to ./results/checkpoint-500
Configuration saved in ./results/checkpoint-500/config.json
Model weights saved in ./results/checkpoint-500/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-1000
Configuration saved in ./results/checkpoint-1000/config.json
Model weights saved in ./results/checkpoint-1000/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([5])) that is different to the
input size (torch.Size([5, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-1500
Configuration saved in ./results/checkpoint-1500/config.json
Model weights saved in ./results/checkpoint-1500/pytorch_model.bin
```

```
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-2000
Configuration saved in ./results/checkpoint-2000/config.json
Model weights saved in ./results/checkpoint-2000/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([5])) that is different to the
input size (torch.Size([5, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-2500
Configuration saved in ./results/checkpoint-2500/config.json
Model weights saved in ./results/checkpoint-2500/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-3000
Configuration saved in ./results/checkpoint-3000/config.json
Model weights saved in ./results/checkpoint-3000/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([5])) that is different to the
input size (torch.Size([5, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-3500
Configuration saved in ./results/checkpoint-3500/config.json
Model weights saved in ./results/checkpoint-3500/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Saving model checkpoint to ./results/checkpoint-4000
```

```
Configuration saved in ./results/checkpoint-4000/config.json
Model weights saved in ./results/checkpoint-4000/pytorch_model.bin
/home/fangzhel/.local/lib/python3.8/site-packages/torch/nn/modules/loss.py:529:
UserWarning: Using a target size (torch.Size([16])) that is different to the
input size (torch.Size([16, 1])). This will likely lead to incorrect results due
to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
```

[ ]:
```
trainer.evaluate()
```

[ ]:
```
result = trainer.predict(final_dataset).predictions
```

[ ]:
```
results = np.reshape(result, 3821)
```

[ ]:
```
results = results.tolist()
```

[ ]:
```
final["predicted"] = results
```

[ ]:
```
final_rating = final.drop(["annotator_id","group",'qa_text','text'], axis=1)
```

[ ]:
```
final_rating.head(5)
```

[ ]:
```
rating_mean = final_rating.groupby('id').predicted.mean()
```

[ ]:
```
rating_mean = rating_mean.reset_index()
```

[ ]:
```
rating_mean.to_csv("submission.csv", index=False)
```

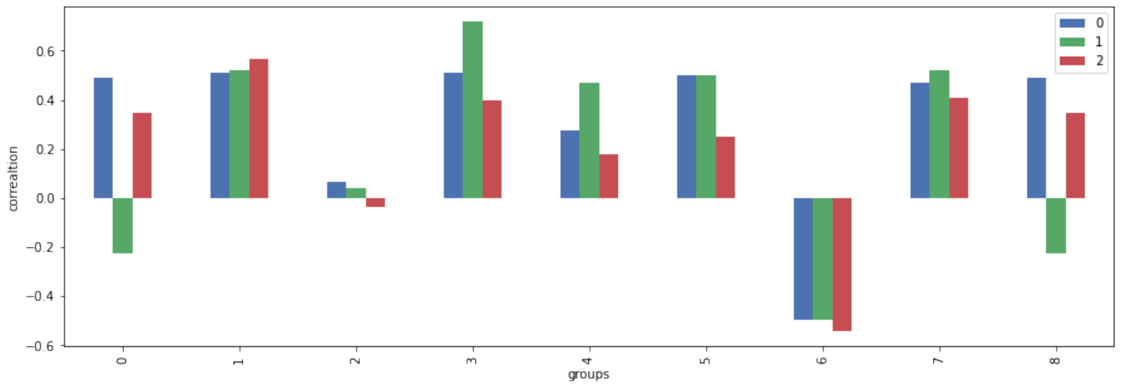
[ ]:
```
torch.save(model.state_dict(), "./model1")
```

1.7 Problem 13: Annotator Ablation Test

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

plt.style.use('seaborn-deep')

group_correlations = [[0.48951400645043536, -0.22837264256179246, 0.
 ↪3438469832402187],
 [0.5084151132399999, 0.5204087192234862, 0.5659612298927907],
 [0.06687308424459298, 0.03811909073425339, -0.03675495225115784],
 [0.5075811356037233, 0.7199960137102475, 0.39843548979370641],
 [0.27585145631477503, 0.4674992263392126, 0.17527538703825574],
 [0.5012456790702576, 0.5007186735306224, 0.2517263080847705],
 [-0.4979473531082712, -0.49684278745558474, -0.5441892868999019],
 [0.4699327154711153, 0.5203062782820229, 0.41029940141805965],
 [0.48951400645043536, -0.22837264256179246, 0.3438469832402187]]
```

```
cor_df = pd.DataFrame(group_correlations)

cor_df.plot.bar(figsize=(15,5), xlabel="groups", ylabel="correaltion")
```

```
[2]: <AxesSubplot:xlabel='groups', ylabel='correaltion'>
```


correaltion
groups