Proposal for Preliminary Analysis on "Re-contextualizing Fairness in NLP: The Case of India"

Anuj Attri (23M0808)
Arnav Attri (23M0811)

Dear TAs,

I trust you're in good spirits as you receive this message. My team and I have meticulously evaluated potential research papers for our preliminary analysis in the CS626 course. We are excited to present our selection, which aligns perfectly with our academic interests and holds particular significance given our institution's connection to a parallel research domain.

Paper Selection: "Re-contextualizing Fairness in NLP: The Case of India"

- **Authors:** Shaily Bhatt, Sunipa Dev, Partha Talukdar, Shachi Dave, Vinod Kumar Prabhakaran (*Google Research*)
- **Paper URL:** `https://aclanthology.org/2022.aacl-main.55/`
- **Code:** This paper *lacks accompanying code*; however, we will make an attempt to write our own code to replicate the results to the best of our ability, considering the *dataset's size and the complexity of the models utilized.*
- **Paper ID:** Anthology ID: 2022.aacl-main.55
- **Conference:** Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)
- **Date:** November 2022

This paper addresses NLP fairness in the Indian context and explores a geocultural environment, social disparities, demonstrating prediction biases along social disparities. It offers a unique perspective and aligns well with our research interests.

Relevance to *IIT Bombay's Research*: "With Prejudice to None"

- **Title:** "With Prejudice to None: A Few-Shot, Multilingual Transfer Learning Approach to Detect Social Bias in Low Resource Languages"
- **Authors:** Nihar Ranjan Sahoo, Niteesh Mallela, Pushpak Bhattacharyya
- **Institution:** IIT Bombay
- **Research Domain:** Detection of social bias in low-resource languages
- **Similarity:** Our paper selection is related to research conducted in our own institution, IIT Bombay. The CFILT lab at IIT Bombay has also ventured into similar research territory. This convergence of interests and research areas makes this selection even more compelling for us.

To perform our preliminary analysis, we will be utilizing the "IndicCorp_en" dataset, comprising approximately ***7GB*** of data. The *dataset terms* referenced in the research paper are readily available on GitHub via this link: `https://tinyurl.com/4s5mtkxj`.

Moreover, we have already initiated our analysis and created a Jupyter notebook. We are ready to provide evidence of our progress, findings, and screenshots to demonstrate our dedication to this project.

We respectfully request your permission and guidance to proceed with our preliminary analysis of this paper. We firmly believe this project offers a unique and valuable perspective, aligning with the academic rigor expected in *CS626* and the societal relevance of India's context. Your approval and guidance are essential for our successful analysis.

We are enthusiastic about the opportunity to delve into this important research and are confident that our work will make a meaningful contribution to the field. Thank you for considering our proposal, and we eagerly await your response.

P.S. - Images of our Jupyter Notebook, showcasing our project's progress, are attached at the end.

1.2 NLP_Paper using IndicCorp_en

October 23, 2023

1 Extracting 200 sentences from IndicCorpa-en (No need to run everytime) leaves a output.txt at download folder.

[ ]:
```
import random

# Path to the IndicCorp-en text file
input_file = '/Users/arnav/Downloads/data/en/en.txt'

# Path to the identity terms file
identity_terms_file = '/Users/arnav/Desktop/NLP Paper/Dataset/region_idterms.
↪tsv'

# Path to the new subset file, leaves the file at Downloads folder.
output_file = '/Users/arnav/Downloads/data/en/Output.txt'

# Number of sentences you want in the subset
subset_size = 200  # You can adjust this value

# Read the identity terms from the file
identity_terms = []

with open(identity_terms_file, 'r') as f:
    for line in f:
        term = line.strip()
        identity_terms.append(term)

# Read the data
with open(input_file, 'r', encoding='utf-8') as f:
    sentences = f.readlines()

# Randomly select a subset of sentences containing identity terms
subset = random.sample([sentence for sentence in sentences if any(term in␣
↪sentence.lower() for term in identity_terms)], subset_size)

# Save the subset to a new file
with open(output_file, 'w', encoding='utf-8') as f:
    f.writelines(subset)
```

1

Figure 1:

2.4 Here's a snippet of the code with just a few identity terms, so it runs faster and demonstrates the concept:

2.5 Also solve NAN problems + give negative sentiment

[5]:
```
from transformers import DistilBertTokenizer,␣
↪DistilBertForSequenceClassification, pipeline
import numpy as np

model_name = "distilbert-base-uncased"
model = DistilBertForSequenceClassification.from_pretrained(model_name)
tokenizer = DistilBertTokenizer.from_pretrained(model_name)
nlp = pipeline("sentiment-analysis", model=model, tokenizer=tokenizer)

# Identity terms (a few for demonstration)
identity_terms = ["bihari", "kashmiri", "marwari"]

# Initialize an empty dictionary to store sentiment shifts
sentiment_shifts = {}

# Loop through each identity term
for identity_term in identity_terms:
    # Create a list to store sentences with the identity term
    sentences_with_identity = []

    # Create a list to store sentences without the identity term
    sentences_without_identity = []

    # Load the 200 sentences from output.txt (replace with your path)
    with open("/Users/arnav/Desktop/NLP Paper/Dataset/IndicCorp dataset/Output.
↪txt", "r") as file:
        sentences = file.readlines()

    # Split the sentences into two groups: with and without the identity term
    for sentence in sentences:
        if identity_term in sentence:
            sentences_with_identity.append(sentence)
        else:
            sentences_without_identity.append(sentence)
```


Figure 2:

```
    # Filter out problematic sentences
    sentences_with_identity = [s for s in sentences_with_identity if len(s) >␣
↪10]
    sentences_without_identity = [s for s in sentences_without_identity if␣
↪len(s) > 10]

    # Calculate sentiment scores for both groups
    sentiment_scores_with_identity = [result['score'] for result in␣
↪nlp(sentences_with_identity) if 'score' in result]
    sentiment_scores_without_identity = [result['score'] for result in␣
↪nlp(sentences_without_identity) if 'score' in result]

    # Calculate the sentiment shift as the difference between the means of the␣
↪two groups
    if sentiment_scores_with_identity and sentiment_scores_without_identity:
        shift = np.mean(sentiment_scores_with_identity) - np.
↪mean(sentiment_scores_without_identity)
        sentiment_shifts[identity_term] = shift

# Print the sentiment shifts for each identity term
for identity_term, shift in sentiment_shifts.items():
    print(f"Identity Term: {identity_term}, Sentiment Shift: {shift:.4f}")
```

```
Some weights of the model checkpoint at distilbert-base-uncased were not used
when initializing DistilBertForSequenceClassification:
['vocab_layer_norm.weight', 'vocab_transform.bias', 'vocab_transform.weight',
'vocab_projector.bias', 'vocab_projector.weight', 'vocab_layer_norm.bias']
- This IS expected if you are initializing DistilBertForSequenceClassification
from the checkpoint of a model trained on another task or with another
architecture (e.g. initializing a BertForSequenceClassification model from a
BertForPreTraining model).
- This IS NOT expected if you are initializing
DistilBertForSequenceClassification from the checkpoint of a model that you
expect to be exactly identical (initializing a BertForSequenceClassification
model from a BertForSequenceClassification model).
Some weights of DistilBertForSequenceClassification were not initialized from
the model checkpoint at distilbert-base-uncased and are newly initialized:
['classifier.bias', 'pre_classifier.bias', 'pre_classifier.weight',
'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

```
Identity Term: bihari, Sentiment Shift: -0.0019
Identity Term: kashmiri, Sentiment Shift: -0.0056
Identity Term: marwari, Sentiment Shift: -0.0046
```

Figure 3:

2.6 Code to identify identity terms associated with positive sentiment shifts. Here's a modified version of the code that will filter and print only those terms with positive sentiment shifts:

[8]:
```
from transformers import DistilBertTokenizer,␣
↪DistilBertForSequenceClassification, pipeline
import random
import numpy as np

# Initialize the sentiment analysis model
model_name = "distilbert-base-uncased"
model = DistilBertForSequenceClassification.from_pretrained(model_name)
tokenizer = DistilBertTokenizer.from_pretrained(model_name)
nlp = pipeline("sentiment-analysis", model=model, tokenizer=tokenizer)

# Identity terms from region_idterms.tsv
identity_terms = ["andamanese", "assamese", "bengali", "bihari",␣
↪"chattisgarhi", "delhiite", "goan", "gujarati", "jharkhandi", "kannadiga",␣
↪"kashmiri", "keralite", "madhya pradeshi", "maharashtrian", "manipuri",␣
↪"marathi", "marwari", "meghalayan", "mizo", "odiya", "pahari", "punjabi",␣
↪"rajasthani", "sikkemese", "tamilian", "telugu", "tripuri", "uttar␣
↪pradeshi", "uttarakhandi", "arunachali", "haryanvi", "himachali"]

# Initialize an empty dictionary to store sentiment shifts
sentiment_shifts = {}

# Loop through each identity term
for identity_term in identity_terms:
    # Create a list to store sentences with the identity term
    sentences_with_identity = []

    # Create a list to store sentences without the identity term
    sentences_without_identity = []

    # Load the 200 sentences from output.txt
    with open("/Users/arnav/Desktop/NLP Paper/Dataset/IndicCorp dataset/Output.
↪txt", "r") as file:
        sentences = file.readlines()

    # Split the sentences into two groups: with and without the identity term
    for sentence in sentences:
        if identity_term in sentence:
            sentences_with_identity.append(sentence)
        else:
            sentences_without_identity.append(sentence)

    # Calculate sentiment scores for both groups
```

Figure 4:

```
    sentiment_scores_with_identity = [result['score'] for result in␣
↪nlp(sentences_with_identity)]
    sentiment_scores_without_identity = [result['score'] for result in␣
↪nlp(sentences_without_identity)]

    # Calculate the sentiment shift as the difference between the means of the␣
↪two groups
    shift = np.mean(sentiment_scores_with_identity) - np.
↪mean(sentiment_scores_without_identity)

    # Store the sentiment shift in the dictionary
    sentiment_shifts[identity_term] = shift

# Print only those identity terms with positive sentiment shifts
for identity_term, shift in sentiment_shifts.items():
    if shift > 0:
        print(f"Identity Term: {identity_term}, Sentiment Shift: {shift:.4f}")
```

```
Some weights of the model checkpoint at distilbert-base-uncased were not used
when initializing DistilBertForSequenceClassification:
['vocab_layer_norm.weight', 'vocab_transform.bias', 'vocab_transform.weight',
'vocab_projector.bias', 'vocab_projector.weight', 'vocab_layer_norm.bias']
- This IS expected if you are initializing DistilBertForSequenceClassification
from the checkpoint of a model trained on another task or with another
architecture (e.g. initializing a BertForSequenceClassification model from a
BertForPreTraining model).
- This IS NOT expected if you are initializing
DistilBertForSequenceClassification from the checkpoint of a model that you
expect to be exactly identical (initializing a BertForSequenceClassification
model from a BertForSequenceClassification model).
Some weights of DistilBertForSequenceClassification were not initialized from
the model checkpoint at distilbert-base-uncased and are newly initialized:
['classifier.bias', 'pre_classifier.bias', 'pre_classifier.weight',
'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

```
Identity Term: kashmiri, Sentiment Shift: 0.0025
Identity Term: marwari, Sentiment Shift: 0.0224
```

Figure 5: